FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Aplikace strojového učení pro analýzu bezpečnostních auditních záznamů v kontextu GDPR
Student:	Jakub Sekera
Vedoucí:	Mgr. Jakub Růžička
Studijní program:	Informatika
Studijní obor:	Počítačová bezpečnost
Katedra:	Katedra informační bezpečnosti
Platnost zadání:	Do konce zimního semestru 2019/20

Pokyny pro vypracování

Popište, jakým způsobem lze využít strojové učení v prostředí malých a středních podniků pro analýzu bezpečnostních auditních záznamů, za účelem vyhovění požadavkům GDPR, tak, aby zjednodušilo a/nebo automatizovalo práci bezpečnostního analytika:
* Jaké záznamy nařizuje GDPR firmám auditovat
* Jakým způsobem může strojové učení usnadnit identifikaci podezřelé aktivity

Nastudujte, popište a porovnejte supervised a unsupervised metody strojového učení využívané v oblasti auditu bezpečnostních záznamů. Vyberte jeden z přístupů, navrhněte a implementujte funkční prototyp detekující podezřelou aktivitu. Prototyp otestujte v prostředí malé/střední firmy a zhodnoťte naplnění cílů práce.

Výsledkem práce je open-source programový modul implementovaný za využití frameworku TensorFlow.

Seznam odborné literatury

Dodá vedoucí práce.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 12. září 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práca

Aplikácia strojového učenia pre analýzu bezpečnostných auditných záznamov v kontexte GDPR

Bc. Jakub Sekera

Katedra informační bezpečnosti
Vedúci práce: Mgr. Jakub Růžička

4. februára 2020

Poďakovanie

V prvom rade by som chcel poďakovať vedúcemu tejto diplomovej práce Mgr. Jakubovi Růžičkovi za jeho cenné rady, pripomienky a za množstvo konzultácií, ktoré sme spolu absolvovali. V druhom rade by som rád poďakoval Ing. Tomášovi Komárkovi za jeho cenné rady a pripomienky v oblasti strojového učenia a firme Cisco Systems, ktorá mi poskytla dátové sady. V neposlednom rade by som rád poďakoval mojej priateľke Zuzke, ktorá má podporovala počas môjho celého štúdia, ale aj mojim rodičom a starým rodičom za finančnú, ale aj mentálnu podporu, ktorá mi štúdium na vysokej škole výrazne zjednodušovala.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval(a) samostatne a že som uviedol(uviedla) všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, v znení neskorších predpisov. V súlade s ustanovením § 46 odst. 6 tohoto zákona týmto udeľujem bezvýhradné oprávnenie (licenciu) k užívaniu tejto mojej práce, a to vrátane všetkých počítačových programov ktoré sú jej súčasťou alebo prílohou a tiež všetkej ich dokumentácie (ďalej len „Dielo“), a to všetkým osobám, ktoré si prajú Dielo užívať.

Tieto osoby sú oprávnené Dielo používať akýmkoľvek spôsobom, ktorý neznižuje hodnotu Diela, a za akýmkoľvek účelom (vrátane komerčného využitia). Toto oprávnenie je časovo, územne a množstevne neobmedzené. Každá osoba, ktorá využije vyššie uvedenú licenciu, sa však zaväzuje priradiť každému dielu, ktoré vznikne (čo i len čiastočne) na základe Diela, úpravou Diela, spojením Diela s iným dielom, zaradením Diela do diela súborného či zpracovaním Diela (vrátane prekladu), licenciu aspoň vo vyššie uvedenom rozsahu a zároveň sa zaväzuje sprístupniť zdrojový kód takého diela aspoň zrovnateľným spôsobom a v zrovnateľnom rozsahu ako je zprístupnený zdrojový kód Diela.

V Prahe 4. februára 2020 . . . . . .

České vysoké učení technické v Praze
Fakulta informačních technologií
© 2020 Jakub Sekera. Všetky práva vyhradené.
Táto práca vznikla ako školské dielo na FIT ČVUT v Prahe. Práca je chránená medzinárodnými predpismi a zmluvami o autorskom práve a právach súvisiacich s autorským právom. Na jej využitie, s výnimkou bezplatných zákonných licencií, je nutný súhlas autora.

Odkaz na túto prácu

Sekera, Jakub. *Aplikácia strojového učenia pre analýzu bezpečnostných auditných záznamov v kontexte GDPR*. Diplomová práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2020.

Abstrakt

Teoretická časť tejto diplomovej práce je venovaná podrobnému popisu bezpečnostných auditných záznamov, log manažmentu a SIEM systémov, ale aj tomu ako je možné využiť strojové učenie k analýze záznamov a identifikácií podozrivej aktivity. Súčasťou teoretickej časti je aj podrobná analýza nariadenia GDPR vo vzťahu k strojovému učeniu a bezpečnostným auditným záznamom, ale aj popis odporúčaní a nariadení, ktoré sú spojené s týmto nariadením. Praktická časť tejto diplomovej práce navrhuje a implementuje funkčný prototyp (skript) schopný detegovať podozrivú aktivitu (anomálie) pomocou algoritmov strojového učenia zo záznamov vytvorených webovým proxy serverom. Navrhnutý a implementovaný skript je testovaný na reálnych dátach poskytnutých firmou Cisco Systems a je navrhnutý tak, aby mohol byť v budúcnosti (po ďalšom vývoji) súčasťou rôznych SIEM systémov ako programový modul. Výstupom skriptu sú rôzne metriky a grafy, ale hlavne súbor s detegovanými anomáliami, ktorý môže slúžiť bezpečnostným analytikom ako ďalší zdroj informácií a pomôcť im tak pri analýze a riešení rôznych bezpečnostných incidentov. Skript môže plniť aj funkciu „automatizovaného“ filtra a to tak, že z veľkého množstva záznamov vyfiltruje hrozby, ktoré sú relevantné (vzbudzujú podozrenie žeby mohli byť škodlivé) a ktoré môžu byť použité ako vstup do ďalších systémov určených k ich detailnejšej analýze.

Klíčová slova strojové učenie, detekcia anomálií, GDPR, SIEM systémy, log manažment, bezpečnostné auditné záznamy, web proxy záznamy

Abstract

The theoretical part of this master's thesis consists of a detailed description of security audit records, log management, and SIEM systems, but also how machine learning can be used to analyze records and identify suspicious activity. Moreover, it includes a detailed analysis of GDPR in relation to machine learning and security audit records, as well as a description of the recommendations and regulations associated with this regulation are included. The practical part of this master's thesis designs and implements prototype (script), which is able to detect suspicious activity with the help of machine learning from records created by web proxy servers. Designed and implemented script is tested on real data provided by Cisco System company and is designed to be part of various SIEM systems as a module in the future (after further development). The script outputs include various metrics and charts but mainly a file with detected anomalies. The file can serve as a source of information for security analysts to help them analyze and resolve various security incidents and alerts. The script can also be used as an "automated" filter because the script is able to filter threats from a large number of records that are relevant (might be harmful) and can be used as input to other systems designed to analyze these threats in more detail.

Keywords machine learning, anomaly detection, GDPR, SIEM systems, log management, security audit records, web proxy logs

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Od čias, kedy sa stali počítače súčasťou nášho každodenného života (práca, zábava, sociálne siete atď.) stúpa aj záujem útočníkov o to, aby mohli zarobiť peniaze alebo napáchať iné škody tým, že využijú rôzne zraniteľnosti počítačových systémov a sietí. Mnoho typov malvéru využíva internet práve k svojmu rozširovaniu, čo umožňuje útočníkom jednoduchšie a rýchlejšie vniknúť do rôznych systémov a sietí, a tým získať rôzne citlivé dáta ako sú osobné údaje užívateľov daného systému, čísla kreditných kariet atď. Nemusí to byť len o získavaní rôznych citlivých dát, ale môže to byť aj napr. rozosielanie nechcenej pošty, šírenie nevyžiadanej reklamy, ale aj využívanie výpočtového výkonu nakazených počítačov napr. na ťažbu kryptomien atp. Firmy sa snažia tomu brániť tým, že inštalujú do svojich firemných prostredí rôzne bezpečnostné prvky ako sú napr. antivírusové programy a zariadenia na detekciu a prevenciu prieniku. Avšak pri detekcií a prevencií prieniku môžu výrazne pomôcť aj tzv. SIEM systémy, ktoré dokážu analyzovať rôzne typy a druhy bezpečnostných auditných záznamov a na základe rôznych staticky definovaný pravidiel a filtrov vytvárať rôzne výstrahy a upozornenia, ktoré sú následne analyzované bezpečnostnými expertami. Tieto výstrahy a upozornenia pomáhajú bezpečnostným expertom identifikovať potenciálne útoky a zabrániť ich tak ďalšiemu šíreniu v prostredí firmy, resp. organizácie. Avšak tieto statické pravidlá resp. filtre musia byť neustále vylepšované na to, aby dokázali odhaliť škodlivú činnosť nových druhov hrozieb resp. malvéru, a práve to je jeden z prípadov, s ktorým nám môže pomôcť strojové učenie.

Strojové učenie ponúka v súčasnosti riešenia na rôzne problémy a je teda pochopiteľné, že je aplikované na oblasť počítačovej bezpečnosti. Na oblasť, ktorá poskytuje ohromné sady dát a informácií, ktoré sú pre strojové učenia tak dôležité, pretože z nich vie získať zaujímavé informácie a pomôcť tak bezpečnostným analytikom riešiť rôzne problémy a zjednodušiť ich život. Už v súčasnosti sa na riešenie problému detekcie malvéru, ktorá je úzko spojená s detekciou anomálií čoraz s vyšším úspechom používajú algoritmy strojového

učenia, avšak je potrebné dodať, že úspešnosť jednotlivých algoritmov detegovať podozrivú aktivitu je častokrát spojená s úrovňou kvality dát atp. Pomocou strojového učenia je možné detegovať podozrivú aktivitu a v ideálnom prípade odhaliť zdroj potenciálnej nákazy a uľahčiť identifikáciu podozrivej aktivity, čomu sa budeme venovať aj v tejto diplomovej práci.

Ciele práce

Medzi ciele tejto diplomovej práce patrí:

- V krátkosti popísať strojové učenie aj so zameraním sa na oblasť počítačovej bezpečnosti
- Priniesť čitateľovi obsiahlejší pohľad do problematiky záznamov, či už z pohľadu ich typov a formátov, ale aj z pohľadu ich zberu, spracovania a analýzy
- Popísať ako dokáže strojové učenie pomôcť bezpečnostným analytikom s identifikáciou podozrivej aktivity pri analýze záznamov
- Analyzovať nariadenie GDPR so zameraním sa na bezpečnostné auditné záznamy a strojové učenie
- V krátkosti sa venovať popisu oblasti anomálií, ich detekcie a popísať, a porovnať *supervised*, *semi-supervised* a *unsupervised* metódy, resp. techniky strojového učenia využívané v oblasti bezpečnostných auditných záznamov pri detekcií anomálií
- Navrhnúť a implementovať jednoduchý (*open-source*) prototyp skriptu, ktorý bude schopný detegovať podozrivú aktivitu, resp. anomálie
- Analyzovať poskytnuté záznamy, aplikovať, otestovať a vyhodnotiť existujúce algoritmy (modely) strojového učenia na dátach (bezpečnostných auditných záznamoch) z reálneho prostredia malých/stredných firiem pomocou navrhnutého skriptu

Štruktúra práce

Štruktúra práce vychádza z vyššie spomenutých cieľov tejto práce a obsahom jednotlivých kapitol je:

1. kapitola je venovaná všeobecnému úvodu do strojového učenia (základnému popisu, typom strojového učenia, *deep learning*, ale aj tomu ako je možné využiť strojové učenie v oblasti počítačovej bezpečnosti)

2. kapitola sa venuje popisu bezpečnostných auditných záznamov (ako sa záznamy delia podľa toho, kto je zdrojom ich vytvorenia), ďalej táto kapitola obsahuje popis log manažmentu, rôznych formátov záznamov, ale aj popis systémom určených na zber, spracovanie a analýzu záznamov (SIEM systémy) a záver tejto kapitoly je venovaný popisu využitia strojového učenia na analýzu a identifikáciu podozrivej aktivity

3. kapitola je venovaná problematike GDPR a jej obsahom sú základné informácie o nariadení GDPR, zaoberá sa vzťahom GDPR a strojového učenia, vzťahom GDPR a bezpečnostných auditných záznamov, vzťahom SIEM systémov a GDPR, ale popisuje aj rôzne odporúčania a nariadenia, a venuje sa aj kritike GDPR

4. kapitola je venovaná strojovému učeniu využívanému v oblasti auditu bezpečnostných záznamov, a konkrétne sa táto kapitola venuje anomáliám (ich definícií, rozdeleniu atď.), ich detekcií (rôznym technikám detekcie anomálií založených na strojovom učení), ale aj teoretickému popisu algoritmov použitých v praktickej časti tejto diplomovej práce

5. kapitola je venovaná praktickej časti tejto diplomovej práce a konkrétne návrhu (architektúre výsledného skriptu), hodnoteniu výkonnosti jednotlivých modelov, ale aj popisu použitých programovacích jazykov, knižníc a nástrojov

6. kapitola sa venuje aplikácií návrhu na poskytnuté záznamy od ich získavania, základného popisu, analýzy až po výsledky a vyhodnotenie jednotlivých modelov na týchto poskytnutých záznamoch

Kapitola 1

Strojové učenie

V tejto úvodnej kapitole sa budeme venovať všeobecnému popisu strojového učenia, typom algoritmom strojového učenia, obsahom tejto kapitoly je aj krátka sekcia o deep learning a v závere sa budeme venovať aplikácií strojového učenia na oblasť počítačovej bezpečnosti.

1.1 Všeobecný popis

Deterministické počítačové algoritmy pracujú na základe princípu determinizmu. To znamená, že na rovnaký vstup odpovedajú vždy rovnakým výstupom. Tento princíp avšak nie je vždy žiadúci a ani dostačujúci.

Marsland [1] začína svoju knihu príkladom, ktorý hovorí o tom, že si máme predstaviť, že vlastníme webovú stránku, ktorá predáva softvér a chceme pochopiť správanie sa našich zákazníkov na tejto webovej stránke a zvýšiť predaj svojich produktov. O každom návštevníkovi sa nám podarí zozbierať informácie ako je napr. typ operačného systému, typ internetového prehliadača, krajina z ktorej daný zákazník na našu webovú stránku pristupuje, akým spôsobom zaplatil atď. Na základe týchto dát sa snažíme odhadnúť a doporučiť zákazníkom produkty o ktoré by mohli mať záujem. Každý zákazník je však iný a my musíme zaistiť to, aby sme zákazníkovi odporučili produkt, ktorý si od nás s najväčšou pravdepodobnosťou kúpi. Práve v tomto prípade sa hodí algoritmus strojového učenia, ktorý narozdiel od klasických algoritmov dokáže zobrať do úvahy dáta špecifické pre každého užívateľa, ako je napr. história nákupov, jeho preferencie apod.

Podľa Chio a Freeman [2] je strojové učenie podoblasťou umelej inteligencie, ktorá sa zaoberá algoritmami a technikami umožňujúcim počítačovým systémom „učiť sa“. Pod pojmom učenie si musíme predstaviť schopnosť zovšeobecniť poznatky z dát a informácií získaných strojovým učením v minulosti. Na základe týchto poznatkov dokáže strojové učenie predpovedať budúce výstupy/výsledky. Strojové učenie je postavené na matematických princípoch, ktoré sú implementované v jednotlivých algoritmoch strojového učenia.

Forma generalizácie a princípu strojového učenia sa podľa Marsland [1] dá vysvetliť na nasledujúcom príklade hrania hry. Predstavte si, že hráte hru napr. *Scrabble*[1] proti počítači, ktorý ju ešte nikdy predtým nehral. Zo začiatku sa Vám možno bude dariť vyhrávať, avšak príde okamih (pravdepodobne po odohraní niekoľkých hier) kedy sa to zlomí a počítač začne vyhrávať, až do doby keď už sa Vám nepodarí vyhrať ani jednu hru. Túto stratégiu „učenia sa“ hraním hier použije počítač aj proti ďalšiemu hráčovi a nebude sa ju musieť učiť odznova, a to je práve spomínaná forma generalizácie.

Strojové učenie spája podľa Marsland [1] dohromady koncepty a nápady z rôznych oblastí ako je napr. neurológia a biológia, matematika, fyzika atp. Práve vďaka týmto konceptov a nápadom vznikli rôzne algoritmy strojového učenia dávajúce možnosť počítačom „učiť sa“. Príkladom sú neurónové siete v oblasti strojového učenia pre ktoré bol ľudský mozog a jeho fungovanie inšpiráciou.

1.2 Typy strojového učenia

Marsland [1] rozdeľuje metódy strojového učenia na:

- učenie s učiteľom (z angl. *supervised learning*)
- učenie bez učiteľa (z angl. *unsupervised learning*)
- *reinforcement learning*[2]
- evolučné učenie (z angl. *evolutionary learning*)

1.2.1 Supervised learning

V prípade *supervised learning* sa metóde poskytne tzv. trénovacia sada obsahujúca prípady u ktorých vie, aká je správna odpoveď, resp. vie, že danému vstupu odpovedá príslušný výstup. Na základe tejto trénovacej sady dokáže táto metóda svoje poznatky o danom probléme generalizovať[3]. Následne sa bude snažiť (s využitím poznatkov, ktoré sa naučila pomocou trénovacej sady) predpovedať výstupy u vstupoch, u ktorých nemá znalosť správného výstupu, resp. výsledku.

Tento proces učenia sa z trénovacej sady sa nazýva *supervised learning* preto, pretože je tu spojitosť s učiteľom, ktorý učí svojich študentov. Správne odpovede sú známe, metóda predpovedá výsledky na trénovacej sade a je opravovaná učiteľom.

[1] Klasická slovná hra na princípe tvorby krížovky.

[2] Reinforcement learning sa nachádza medzi supervised a unsupervised learning (pre bližšie informácie pozri sekciu 1.2.3).

[3] Schopnosť generovať „rozumné“ výstupy u vstupoch s ktorými sa počas procesu učenia nestretla.

Problémy, ktoré *supervised learning* rieši sa primárne delia na:

- **klasifikáciu** – algoritmus sa snaží priradiť hodnotu/objekt do príslušnej kategórie napr. klasifikácia kvetov do príslušných kategórií podľa ich druhu (napr. iris dáta set od Fisher [3])

- **regresiu** – algoritmus sa snaží odhadnúť spojitú hodnotu napr. odhad ceny nehnuteľnosti na základe lokality, počtu izieb, obývacej plochy atď.

Medzi algoritmy strojového učenia, ktoré patria do skupiny úloh regresie patrí napr. SVR, lineárna a logistická regresia, rozhodovacie stromy, náhodne lesy atď. Do skupiny úloh klasifikácie patria napr. algoritmy ako je K-NN, SVM, naivný Bayes, klasifikácia pomocou rozhodovacích stromov a náhodných lesov atď.

Caruana a Niculescu-Mizil [4] prinášajú rozsiahle empirické porovnanie *supervised* algoritmov strojového učenia.

1.2.2 Unsupervised learning

V prípade *unsupervised learning* nie sú k dispozícií správne odpovede ako to bolo v prípade *supervised learning*, a preto sa táto metóda snaží hľadať podobnosti medzi rôznymi vstupmi. Na základe podobností združuje vstupy, ktoré majú niečo spoločné do jednej skupiny.

Zhluková analýza tzv. *cluster analysis* je jedným zo základných princípov *unsupervised learning*. Základným cieľom zhlukovania je nájsť rôzne skupiny nachádzajúce sa v dátach. Zhlukovacie algoritmy to dosiahnu tak, že dokážu identifikovať rôzne štruktúry v dátach. Prvky toho istého klastra (resp. skupiny) sú si navzájom viac podobné ako tie prvky, ktoré patria do iného klastra (resp. skupiny).

Medzi algoritmy zhlukovej analýzy patrí napr. DBSCAN, K-means, hierarchical clustering, mean-shift atď. V prípade detekcií anomálií (oblasť bude bližšie popísaná v sekcií 4.3) je to napr. algoritmus LOF.

1.2.3 Reinforcement learning

Reinforcement learning je metóda strojového učenia nachádzajúca sa podľa Marsland [1] niekde medzi *supervised* a *unsupervised learning*. Táto metóda umožňuje učiť sa v interaktívnom prostredí prostredníctvom spätnej väzby z pokusov a omylov, a z vlastných akcií a skúsenosti. Algoritmus musí sám objaviť a vyskúšať rôzne možnosti až do chvíle kým nenájde tú správnu odpoveď.

Jedným z príkladov aplikácie a využitia *reinforcement learning* je vytvorenie počítačového programu AlphaGo. AlphaGo sa podľa Silver a kol. [5] stal

prvým počítačovým programom, ktorý dokázal poraziť svetového šampióna v hre Go[4] a to aj s využitím *reinforcement learning*.

Medzi algoritmy *reinforcement learning* patrí podľa Huang [6] napr. Q-learning, SARSA, DQN a DDPG.

1.2.4 Evolutionary learning

Evolutionary learning je podľa Marsland [1] metóda, ktorej algoritmy sú založené na princípoch biologickej evolúcie a tento princíp môžeme vnímať ako spôsob učenia sa. Organizmus sa časom zlepšuje a prispôsobuje prostrediu tak, aby šanca na jeho prežitie a možnosť mať potomkov bola čím ďalej tým väčšia. Algoritmy využívajú myšlienku funkcie *fitness*, čo je skóre vyjadrujúce kvalitu súčasného riešenia.

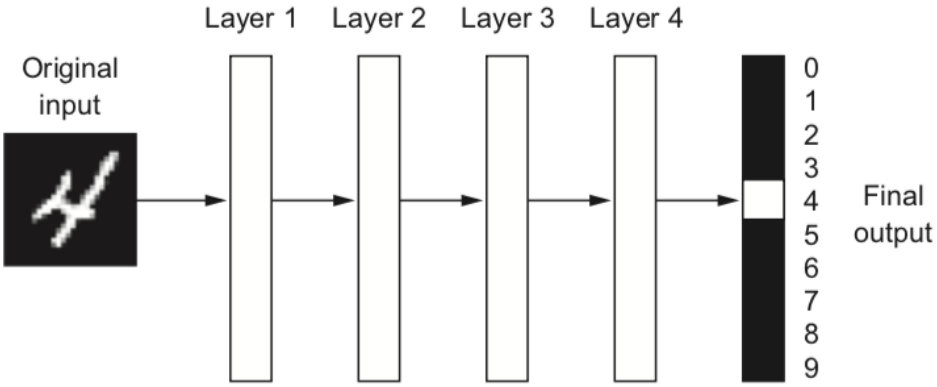
Medzi evolučné algoritmy patria napr. genetické algoritmy, genetické a evolučné programovanie, evolučné stratégie atď.

1.3 Deep learning

Podľa Chollet [7] je hlavným problémom strojového učenia a *deep learning* nájdenie vhodného a zmysluplného spôsobu transformácie dát tak, aby sme našli vhodné reprezentácie vstupných dát, ktoré nás privedú a priblížia k očakávanému výsledku. Reprezentácia dát je odlišný spôsob, ktorým sa môžeme pozerať na tie isté dáta. Niektoré úlohy, ktoré sa zdajú byť zložité pomocou jednej reprezentácie dát môžu byť riešené jednoduchšie pomocou nejakej inej reprezentácie dát. Strojové učenie je v podstate o hľadaní „užitočných“ reprezentácií vstupných dát v rámci preddefinovaného priestoru možností pomocou spätnej väzby. *Deep learning* je podmnožinou strojového učenia. Ide o nový prístup učenia sa rôznych reprezentácií dát, tak aby bol kladený dôraz na učenie sa po sebe nasledujúcich vrstiev čoraz významnejších reprezentácií. Slovo *deep* vo slovnom spojení *deep learning* neznamená hlbšie porozumenie daného problému, skôr ide o to koľko po sebe nasledujúcich vrstiev reprezentácie dát prispieva k výslednému modelu. Moderný *deep learning* často obsahuje desiatky alebo stovky týchto po sebe nasledujúcich vrstiev reprezentácie. V *deep learning* sú tieto tzv. vrstvené reprezentácie dát učené pomocou neurónových sietí.

Ako sieť niekoľkých vrstiev transformácií (pozri obr. 1.1) transformuje obraz číslice (v našom prípade číslicu 4) na to, aby rozpoznala aká číslica sa nachádza na obrázku? Na to sa musíme pozrieť na obr. 1.2, kde je možné vidieť *deep network*, ktorá transformuje digitálny obraz (na ktorom je zobrazená číslica 4) do reprezentácií, ktoré sa čoraz viac odlišujú od pôvodného obrazu a prinášajú čoraz viac informácií o konečnom výsledku.

[4]Strategická dosková hra pre dvoch hráčov.



Obr. 1.1: *Deep neural network* určená pre číslicovú klasifikáciu [7]

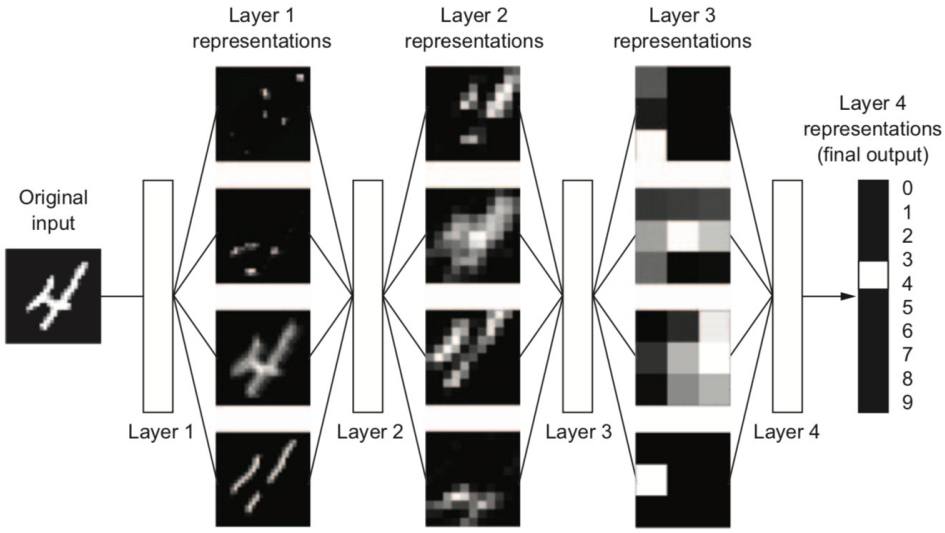
Deep network sa podľa Chollet [7] dá chápať aj ako viacstupňová informačno-destilačná operácia, kde informácie postupne prechádzajú jednotlivými filtrami a vychádzajú čoraz viac purifikované. Ide v podstate o viacstupňový spôsob učenia sa reprezentácií dát.

Pojem neurónová sieť síce pochádza z oblasti neurológie a niektoré koncepty *deep learning* boli čiastočne inšpirované tým, ako chápeme ľudský mozog, no aj napriek tomu modely *deep learning* nie sú modelmi ľudského mozgu. Chollet [7] tvrdí, že neexistuje dôkaz, ktorý by potvrdil, že mozog pracuje na podobných učiacich sa mechanizmoch, ktoré využívajú moderné *deep learning* modely. Pojmy neurónová sieť a *deep learning* (resp. *deep neural network*) sa častokrát zamieňajú a nesprávne používajú. Podľa Ramesh [8] má typická neurónová sieť 3 vrstvy (vstupnú, skrytú a výstupnú). V prípade, že pribúdajú ďalšie tzv. skryté vrstvy tak už hovoríme o *deep learning* (resp. *deep neural network*, obr. 1.3.

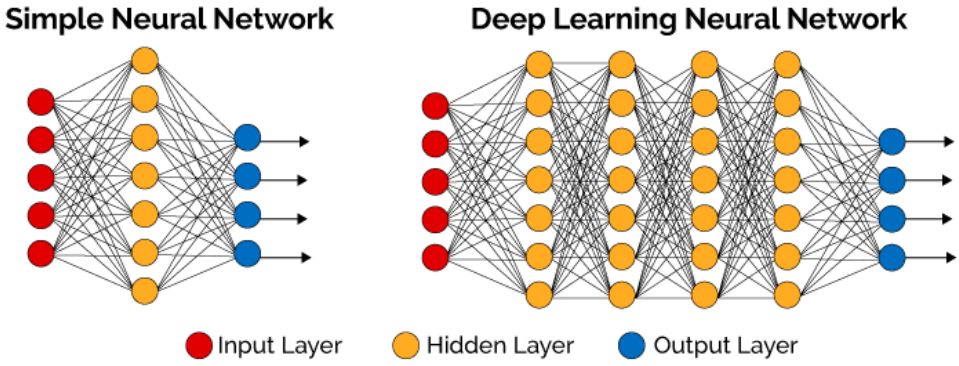
Podľa Polyakov [10] existujú tieto *deep learning* modely a algoritmy:

- pre úlohy regresie – algoritmy ANN, RNN, NTM, DNC
- pre úlohy klasifikácie – algoritmy ANN a CNN
- pre zhlukovú analýzu (tzv. clustering) – napr. algoritmus SOM
- pre odporúčacie systémy (tzv. association rule learning) – algoritmy DRBM, DBN a Stacked Autoencoder
- pre tzv. generatívne modely – algoritmy Variational Autoencoders, GAN a Boltzmann Machines

Koncepty *deep learning* už boli podľa Chollet [7] pochopené v roku 1989, ale *deep learning* začalo prosperovať a byť na vzostupe až od roku 2012,



Obr. 1.2: *Deep learning* reprezentácia dát naučená pomocou modelu číslicovej klasifikácie [7]



Obr. 1.3: Neurónová sieť vs. *deep neural network* [9]

pretože okrem teoretických poznatkov sú experimentálne zistenia a pokusy to, čo poháňa celé strojové učenie a *deep learning* dopredu. Algoritmické pokroky a pokroky v strojovom učení sú možné len vtedy, ak je k dispozícii:

- hardvér (dostupnosť väčšieho výpočtového výkonu)
- väčšie množstvo dát
- porovnávacie testy (tzv. *benchmarks*)

To nám dovoľuje experimentovať, skúšať nové nápady a vylepšovať staré myšlienky. Skutočným úzkym hrdlom v 90. rokoch 20. storočia a na začiatku 21. storočia bol nedostatok vhodných dát a nedostupnosť dostatočného výpočtového výkonu.

1.4 Oblasť počítačovej bezpečnosti

Práca s dátami ako je napr. predspracovanie, analýza atď. predstavuje v oblasti počítačovej bezpečnosti v posledných rokoch podľa Recorded Future [11] jednu z najväčších výziev. Analytici musia pracovať s veľkým množstvom dát, ktoré sa dennodenne produkuje a v globálnom merítku je toto množstvo nepredstaviteľné. V kombinácií s masívnym množstvom upozornení a záznamov, ktoré sa produkujú z rôznych systémov ako sú napr. rôzne webové aplikácie, antivírusové programy, sieťové prvky, e-mailové domény, doménové servery, firewall, systémy, ktoré zaznamenávajú rôzne užívateľské aktivity atp. je jasné, že pre analytikov je takmer nemožné pracovať v izolácií bez nejakej pomoci.

V oblasti počítačovej bezpečnosti sú to podľa Recorded Future [11] práve stroje a algoritmy strojového učenia, ktoré dokážu pomôcť s automatizáciou úloh, a tým priniesť analytikom dostatočné množstvo náhľadov/informácií, ktoré im dokážu výrazne uľahčiť ich prácu, pretože dokážu efektívnejšie a rýchlejšie rozpoznať čo je, a čo nie je škodlivá aktivita. Automatizáciou rôznych úloh a analýz dokážu analytici rýchlo odhaliť hrozby a izolovať situácie, ktoré potrebujú hlbšiu ľudskú analýzu.

Podľa Drinkwater [12] je jednou z najväčších výziev strojového učenia v oblasti počítačovej bezpečnosti schopnosť preniesť výsledky a implementácie algoritmov, ktoré prinášajú vynikajúce výsledky v testovacích prostrediach do reálnych a komplexných systémov a sietí.

Umelá inteligencia, ktorej podmnožinou je aj strojové učenie sa podľa Recorded Future [11] používa v oblasti počítačovej bezpečnosti na:

1. rozpoznávanie vzorov (z angl. *pattern recognition*) – napr. identifikácia phishing e-mailov a nevyžiadanej pošty (spam), identifikácia malvéru atď.

2. detekcia anomálií (z angl. *anomaly detection*) – detekcia nezvyčajných aktivít ako je napr. detekcia podvodov v online bankovníctve, podozrivé správanie sa užívateľov na sietí atď.

3. NLP (tzv. spracovanie prirodzeného jazyka) – konvertovanie neštruktúrovaného textu ako je napr. webová stránka do štruktúrovanej podoby, vhodnej pre strojové učenie (napr. spracovanie obrovského množstva dostupných údajov o jednotlivých hrozbách alebo analýza e-mailov za účelom identifikácie phishingu)

4. prediktívna analýza (z angl. *predictive analytics*) – spracovaním dát a identifikáciou vzoriek je možné robiť rôzne predikcie a identifikácie odľahlých hodnôt (napr. priradzovanie *predictive risk* skóre k entitám ako sú napr. IP adresy a ich následné blokovanie na základe daného skóre)

Firma Cisco [13] tvrdí, že strojové učenie dokáže pomôcť s:

- hľadaním hrozieb v sietí – detekcia hrozieb pomocou monitorovania správania sa siete za účelom nájdenia rôznych anomálií v sietí (detekcia hrozieb vo vnútri siete, neznámeho malvéru a porušenia rôznych pravidiel v rámci siete)

- ochranou užívateľov pri prehľadávaní webových stránok – predikcia škodlivých webových stránok analýzou internetovej aktivity za účelom identifikácie „útočných“ infraštruktúr

- poskytnutím malvérovej ochrany na koncových zariadeniach – identifikácia nových škodlivých súborov a aktivít, ktorá je založená na vlastnostiach a správaní sa známych škodlivých malvérov

- ochranou dát v cloud prostredí – analýza podozrivých pokusov o prihlásenie sa do cloud aplikácií, detekcia anomálií založených na lokalizácií (geografická poloha odkiaľ sa daný človek pripája) a analýza reputácie daných IP adries na to, aby sme mohli identifikovať hrozby a riziká s ktorými sa môžeme stretnúť v cloud aplikáciách a platformách

- detekciou malvéru v zašifrovanom toku dát – analýza šifrovaných dátových elementov využitím sieťovej telemetrie (napr. pomocou riešenia ETA (pozri [14]))

Niektoré vyššie spomenuté aplikácie strojového učenia na oblasť počítačovej bezpečnosti spolu s konkrétnymi príkladmi a implementáciami je možné nájsť v knihe od Chio a Freeman [2]. Aplikáciu strojového učenia v oblasti počítačovej bezpečnosti spolu s vyhodnotením jednotlivých algoritmov je možné takisto nájsť napr. v článku od Ford a Siraj [15].

Kapitola 2

Bezpečnostné auditné záznamy

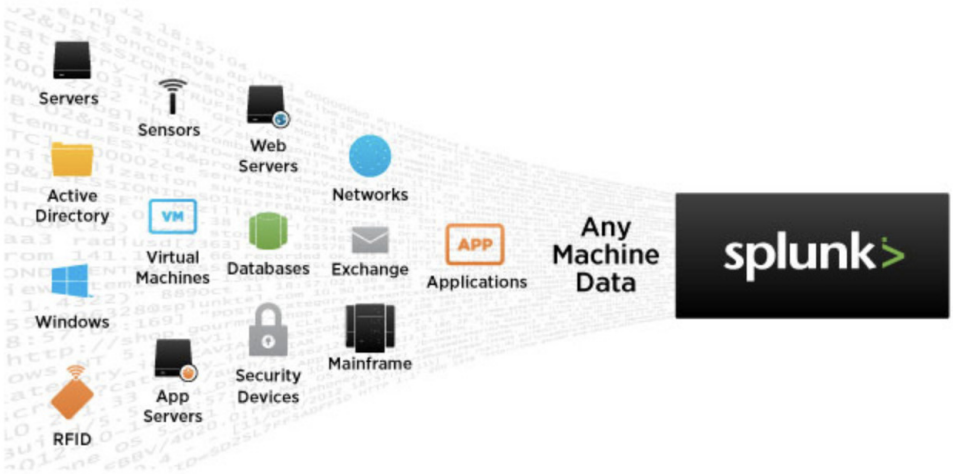
V úvode tejto kapitoly sa budeme venovať všeobecnému popisu záznamov, rozdeleniu záznamov do kategórií podľa toho, kto je zdrojom ich vytvorenia, ale aj rôznym formátom záznamov a tzv. logovacím úrovniam. V ďalšej časti tejto kapitoly sa budeme venovať popisu log manažmentu, systémom určených na zber záznamov, ale hlavne SIEM systémom a softvérom určených na spracovanie a analýzu záznamov. V závere tejto kapitoly sa budeme venovať spôsobu, akým môže strojové učenie uľahčiť identifikáciu podozrivej aktivity a zjednodušiť, prípadne automatizovať prácu bezpečnostných analytikov. Obsahom poslednej časti tejto kapitoly je krátke zhrnutie obsahu celej tejto kapitoly.

2.1 Záznam

Podľa Chuvakin, Schmidt, Phillips a kol. [16], ale aj podľa Kent a Souppay [17] a Al-Fedaghi a Mahdi [18] je záznam (tzv. *log*) zoznam udalostí, ktoré sa stali v rámci systémov a sietí danej organizácie, resp. jej infraštruktúry. Obsahom jednotlivých záznamov sú udalosti, kde každá jedna udalosť, ktorá sa v zázname nachádza obsahuje informácie o špecifickej udalosti, ktorá sa stala v danom systéme alebo sietí. Záznamy boli pôvodne určené na tzv. *troubleshooting* (riešenie problémov), no v súčasnosti je pomocou nich môžné napr. docieliť optimalizáciu výkonu systémov a sietí, zaznamenať akcie vykonávané užívateľmi v danom prostredí, ale aj detegovať rôzne podozrivé aktivity.

Záznam teda poskytuje časovú os jednotlivých udalostí, ktoré sa stali v čase postupne tak ako išli za sebou počas behu aplikácie/systému. Korelácia jednotlivých udalostí a záznamov dokáže priniesť odpoveď na to, prečo došlo napr. k poklesu výkonu daného systému, prečo sa nepodarilo danú aplikáciu aktualizovať alebo aj to koľkokrát a odkiaľ sa prihlásil konkrétny užívateľ do rôznych systémov za jeden deň apod.

V rámci organizácie a jej infraštruktúry existuje mnoho záznamov, ktoré obsahujú informácie týkajúce sa počítačovej bezpečnosti. Príkladom bezpečnostných záznamov sú auditné záznamy, ktoré napr. zaznamenávajú pokusy



Obr. 2.1: Zdroje/kategórie záznamov zbierané softvérom Splunk [19]

o autentifikáciu užívateľa do systému a záznamy bezpečnostných zariadení, ktoré zaznamenávajú podozrivú aktivitu, resp. možné útoky.

Podľa Kent a Souppay [17] a Al-Fedaghi a Mahdi [18] sa dajú záznamy podľa toho, kto je zdrojom ich vytvorenia rozdeliť do nasledujúcich kategórií:

1. Bezpečnostný softvér
2. Operačné systémy
3. Aplikácie

Okrem vyššie spomenutých kategórií existujú aj ďalšie kategórie záznamov, resp. zdroje, ktoré tieto záznamy generujú. Napríklad softvér Splunk spracováva kategórie záznamov, ktoré je možné vidieť na obr. 2.1.

2.1.1 Bezpečnostný softvér

Bezpečnostný softvér je hlavným zdrojom vytvárania bezpečnostných záznamov. Medzi typické sieťovo a klientsky založené bezpečnostné softvéry môže podľa Bhatt, Manadhata a Zomlot [20] a podľa Kent a Souppay [17] patriť:

- Antimalvér softvér – príkladom sú napr. antivírusové programy, ktoré zaznamenávajú pokusy o napadnutie systému, časy skenovania daného systému, inštancie detegovaného malvéru atď.

- IDS a IPS – zaznamenávajú detailné informácie o podozrivých aktivitách a detekciách útokov, v prípade IPS sú to aj záznamy o zastavení škodlivých aktivít

- VPN systémy – zaznamenávajú informácie o úspešných a neúspešných pokusoch o prihlásenie, dátum a čas pripojenia a odpojenia daného užívateľa od VPN a množstvo prenesených dát počas jednej tzv. *session*

- Webové proxy – uchovávajú záznamy o všetkých URL adresách na ktoré bolo pomocou nich pristúpené

- Softvér na správu zraniteľností – zaznamenávajú informácie o opravách, ktoré boli nainštalované a o stave zraniteľnosti každého zariadenia (známe zraniteľnosti a chýbajúce softvérové aktualizácie)

- Autentifikačné servery – zaznamenávajú každý pokus o prihlásenie, ktorý obsahuje zdroj (pôvod), užívateľské meno, dátum a čas, ale aj to či bol pokus úspešný alebo neúspešný

- Smerovače – v prípade, že sú smerovače nastavené tak, aby blokovali určitý typ dátového toku napr. pomocou ACL, tak dokážu zaznamenať stav, kedy dôjde k blokovaniu nežiadúcej aktivity, resp. toku dát, ale oveľa dôležitejšou funkciou smerovačov (z pohľadu záznamov) je to, že dokážu produkovať NetFlow[5]/IPFIX[6] záznamy

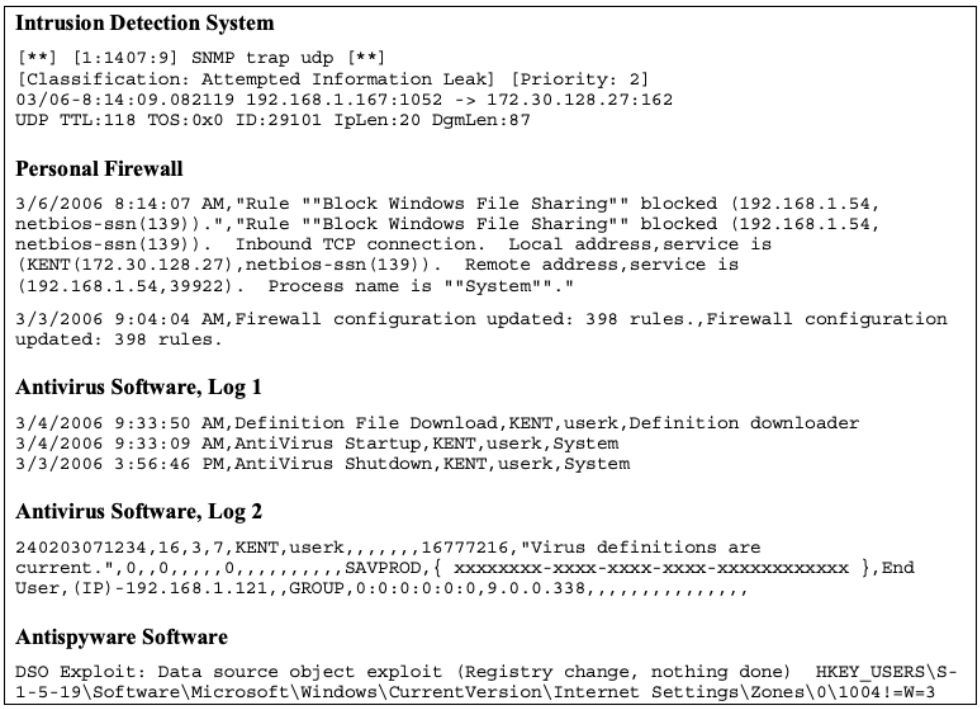
- Firewalls – dokážu povoliť prípadne blokovať rôzne aktivity na základe pravidiel, no narozdiel od smerovačov môžu sledovať stav toku dát a vykonávať kontrolu obsahu, a teda generovať detailnejšie záznamy ako smerovače

- Network Quarantine Servers – zaznamenávajú informácie o tom, ktorí užívatelia boli pri pokuse o pripojenie a prihlásenie sa do siete zablokovaný a čo bolo príčinou tejto blokácie

Na obr. 2.2 je možné vidieť niekoľko príkladov záznamov, ktoré boli vytvorené rôznymi bezpečnostnými softvérmi.

[5]Záznamy z protokolu NetFlow (vyvinutého spoločnosťou Cisco Systems, ktorý poskytuje možnosť zhromažďovať sieťovú prevádzku/prenos pri vstupe alebo výstupe z daného rozhrania) sa môžu používať na detekciu prieniku, či analýzu abnormálnej prevádzky v sieti (pozri [21]).

[6]Protokol vyvinutý pod organizáciou IETF [22] na základe potreby spoločného, univerzálneho štandardu pre export IP flows/tokov, ktorý bol inšpirovaný a je založený na Cisco NetFlow verzie 9.



Intrusion Detection System

```
[**] [1:1407:9] SNMP trap udp [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/06-8:14:09.082119 192.168.1.167:1052 -> 172.30.128.27:162
UDP TTL:118 TOS:0x0 ID:29101 IpLen:20 DgmLen:87
```

Personal Firewall

```
3/6/2006 8:14:07 AM,"Rule ""Block Windows File Sharing"" blocked (192.168.1.54,
netbios-ssn(139)).","Rule ""Block Windows File Sharing"" blocked (192.168.1.54,
netbios-ssn(139)).  Inbound TCP connection.  Local address,service is
(KENT(172.30.128.27),netbios-ssn(139)).  Remote address,service is
(192.168.1.54,39922).  Process name is ""System""."

3/3/2006 9:04:04 AM,Firewall configuration updated: 398 rules.,Firewall configuration
updated: 398 rules.
```

Antivirus Software, Log 1

```
3/4/2006 9:33:50 AM,Definition File Download,KENT,userk,Definition downloader
3/4/2006 9:33:09 AM,AntiVirus Startup,KENT,userk,System
3/3/2006 3:56:46 PM,AntiVirus Shutdown,KENT,userk,System
```

Antivirus Software, Log 2

```
240203071234,16,3,7,KENT,userk,,,,,,,16777216,"Virus definitions are
current.",0,,0,,,,,0,,,,,,,,,,SAVPROD,{ xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx },End
User,(IP)-192.168.1.121,,GROUP,0:0:0:0:0:0:0,9.0.0.338,,,,,,,,,,,,,,
```

Antispyware Software

```
DSO Exploit: Data source object exploit (Registry change, nothing done)  HKEY_USERS\S-
1-5-19\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\0\1004!=W=3
```

Obr. 2.2: Niekoľko príkladov záznamov vytvorených bezpečnostnými softvérmi [17]

2.1.2 Operačné systémy

Operačné systémy serverov, pracovných staníc, ale aj sieťových zariadení ako sú napr. smerovače zaznamenávajú rôzne informácie, ktoré sú spojené s počítačovou bezpečnosťou. Podľa Kent a Souppay [17] sa tieto informácie, resp. záznamy dajú rozdeliť do dvoch kategórií:

- Systémové udalostí – do tejto kategórie patrí napr. záznam o vypnutí systému alebo naštartovaní služby, teda udalostí vykonávané komponentami operačného systému
- Auditné záznamy – do tejto kategórie patrí napr. záznam o úspešných a neúspešných pokusoch o prihlásenie sa do systému, prístup k súborom, zmeny týkajúce sa správy účtov ako je vytvorenie/odstránenie účtu alebo zmena hesla atď., a teda akcie, ktorých iniciátorom je človek

Záznamy z operačných systémov sú primárne určené na identifikáciu a investigáciu podozrivých aktivít, ktoré sa týkajú konkrétneho zariadenia. Po tom ako nejaký bezpečnostný softvér odhalí podozrivú aktivitu tak pomocou záznamov z operačných systémov je možné získať ďalšie informácie, ktoré môžu pomôcť pri investigácií. Podľa Kent a Souppay [17] je formátom väčšiny

```
Event Type:  Success Audit
Event Source: Security
Event Category:     (1)
Event ID:    517
Date:        3/6/2006
Time:        2:56:40 PM
User:        NT AUTHORITY\SYSTEM
Computer:    KENT
Description:
The audit log was cleared
Primary User Name: SYSTEM      Primary Domain: NT AUTHORITY
Primary Logon ID: (0x0,0x3F7)  Client User Name: userk
Client Domain: KENT            Client Logon ID: (0x0,0x28BFD)
```

Obr. 2.3: Príklad bezpečnostného záznamu vytvoreného v operačnom systéme Windows [17]

záznamov, ktoré operačné systémy vytvorili, resp. vytvárajú syslog. Štandard syslog a jeho formát bude bližšie popísaný v podsekcií 2.4.1. Ďalšie záznamy môžu byť ukladané v proprietárnych formátoch ako to je napr. v prípade operačného systému Windows. Na obr. 2.3 je príklad bezpečnostného záznamu vytvoreného v operačnom systéme Windows.

Záznamy sa môžu buď rovno vypisovať na konzolu/terminál daného operačného systému, ukladať lokálne do súboru/súborov na disk a/alebo zároveň posielať do logovacích systémov. Logovacie systémy, resp. rôzne tzv. logovacie manažmenty dokážu s týmito záznamami pracovať a robiť na základe nich rôzne analýzy bezpečnostných udalostí/incidentov, generovať rôzne výstrahy atď.

Podľa Berman [23] patria medzi typické Linuxové súbory, v ktorých sa nachádzajú záznamy napr. **/var/log/messages**, kde sú všeobecné a systémové správy, **/var/log/auth.log** alebo **/var/log/secure**, kde sú záznamy o autentizácií atď.

V prípade rodiny operačných systémov Microsoft Windows slúži na prezeranie a prehľadávanie jednotlivých udalostí a záznamov nástroj Microsoft Event Viewer pomocou ktorého je možné preskúmať záznamy v jednotlivých kategóriách ako je bezpečnosť, administrácia, systém atp.

Sieťové zariadenia medzi ktoré patrí napr. smerovač, prepínač atď. majú vlastné špecializované operačné systémy, ktoré takisto zaznamenávajú rôzne udalosti. Na obr. 2.4 sú príklady záznamov vytvorených v operačnom systéme Cisco IOS.

Vyššie spomenuté príklady sú len ukážkami záznamov vytváraných niektorými operačnými systémami a nejedná sa o kompletný zoznam operačných systémov. U ďalších operačných systémov to môže byť inak a jednotlivé záznamy sa môžu od seba líšiť.

```
R1#
*Feb 14 09:38:48.132: %SYS-5-CONFIG_I: Configured from console by console
```

```
R1#
*Feb 14 09:40:09.325: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed
state to up
*Feb 14 09:40:10.326: %LINEPROTO-5-UPDOWN: Line protocol on Interface
GigabitEthernet0/1, changed state to up
```

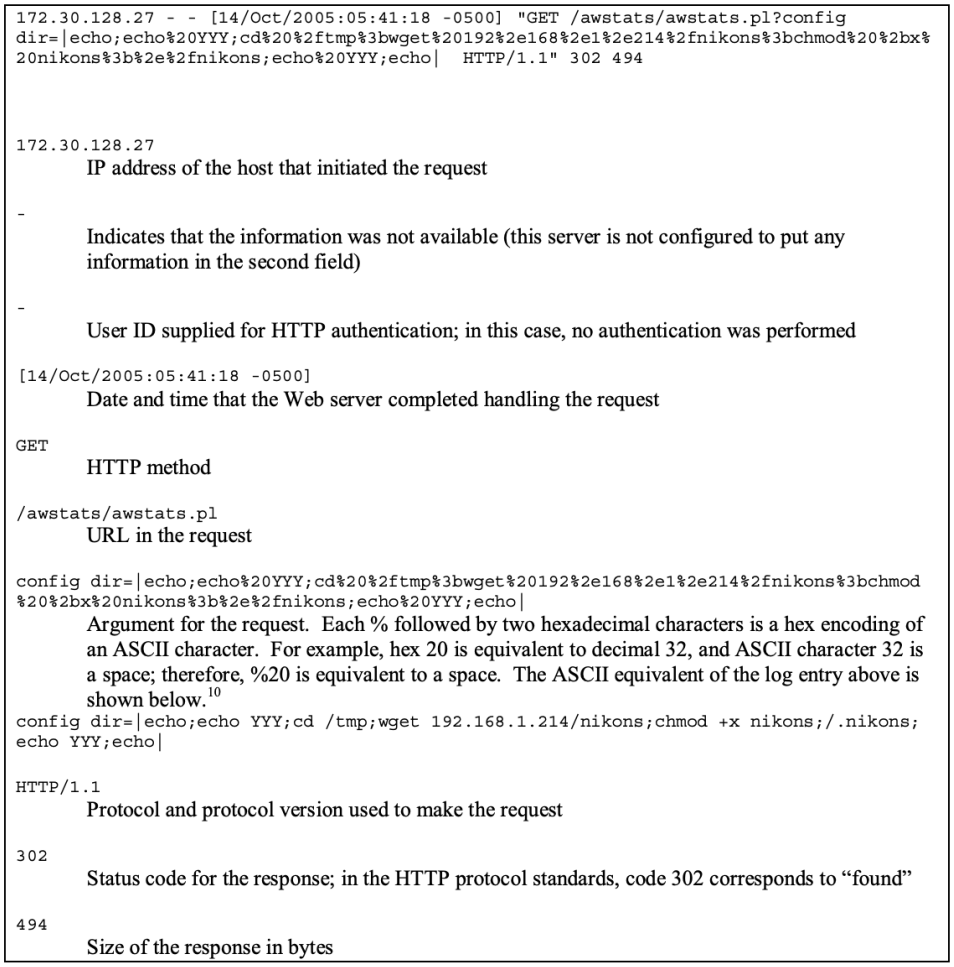
Obr. 2.4: Príklady záznamov vytvorených operačným systémom Cisco IOS [24]

2.1.3 Aplikácie

Niektoré aplikácie vytvárajú svoje vlastné záznamy a niektoré sa spoliehajú na tzv. logovacie schopnosti operačného systému, pod ktorým bežia. Informácie, ktoré sa ukladajú do jednotlivých záznamov sa líšia. Podľa Kent a Souppay [17], ale aj podľa Brown [25] sa dajú informácie, ktoré aplikácie ukladajú do záznamov rozdeliť do týchto nasledujúcich kategórií:

- Požiadavky klientov a odpovede serverov – napr. e-mail servery zaznamenávajú odosielateľa, príjemcu, názov e-mail, atď.; webové servery zaznamenávajú prístupy na každú jednu URL adresu a typy odpovedí, ktorými daný server odpovedá
- Informácie o účtoch – do tejto kategórie patrí napr. záznam o úspešných a neúspešných pokusoch o prihlásenie sa do aplikácie, zmeny týkajúce sa správy účtov ako je vytvorenie/odstránenie účtu v aplikácií alebo zmena hesla atď.
- Informácie o ladení (tzv. *debug information*) – pri diagnostike a riešení rôznych chýb, ktoré môžu pri behu aplikácie vznikať môžu tieto informácie pomôcť
- Informácie o používaní daného systému – napr. počet transakcií vykonaných za určitú časovú jednotku (minútu, hodinu atď.), ale aj veľkosť jednotlivých transakcií
- Dôležité systémové udalostí aplikacií – ako je napr. naštartovanie a vypnutie aplikácie, rôzne chyby a zlyhania aplikácie a dôležité konfiguračné zmeny

Väčšina záznamov vytváraných aplikáciami je v propretiárnom formáte, čo z nich robí záznamy, ktoré sa častokrát dajú problematicky použiť a navyše



```
172.30.128.27 - - [14/Oct/2005:05:41:18 -0500] "GET /awstats/awstats.pl?config
dir=|echo;echo%20YYY;cd%20%2ftmp%3bwget%20192%2e168%2e1%2e214%2fnikons%3bchmod%20%2bx%
20nikons%3b%2e%2fnikons;echo%20YYY;echo|  HTTP/1.1" 302 494
```

172.30.128.27
IP address of the host that initiated the request

-
Indicates that the information was not available (this server is not configured to put any information in the second field)

-
User ID supplied for HTTP authentication; in this case, no authentication was performed

[14/Oct/2005:05:41:18 -0500]
Date and time that the Web server completed handling the request

GET
HTTP method

/awstats/awstats.pl
URL in the request

config dir=|echo;echo%20YYY;cd%20%2ftmp%3bwget%20192%2e168%2e1%2e214%2fnikons%3bchmod
%20%2bx%20nikons%3b%2e%2fnikons;echo%20YYY;echo|
Argument for the request. Each % followed by two hexadecimal characters is a hex encoding of an ASCII character. For example, hex 20 is equivalent to decimal 32, and ASCII character 32 is a space; therefore, %20 is equivalent to a space. The ASCII equivalent of the log entry above is shown below.[10]

```
config dir=|echo;echo YYY;cd /tmp;wget 192.168.1.214/nikons;chmod +x nikons;/.nikons;
echo YYY;echo|
```

HTTP/1.1
Protocol and protocol version used to make the request

302
Status code for the response; in the HTTP protocol standards, code 302 corresponds to "found"

494
Size of the response in bytes

Obr. 2.5: Príklad záznamu z webového servera spolu s vysvetlením [17]

dáta, ktoré obsahujú sú kontextovo závisle, a preto je potrebné viacero zdrojov na preskúmanie ich obsahu. Na obr. 2.5 je záznam z webového servera spolu s vysvetlením, čo znamenajú jednotlivé položky.

2.2 Auditné záznamy

V oblasti počítačovej bezpečnosti sa auditné záznamy podľa Berman [23] používajú hlavne na forenznú a bezpečnostnú analýzu, ale aj napr. pri vyšetrovaní trestných činov.

Gregory [26] tvrdí, že každý auditný záznam by mal minimálne obsahovať:

- **dátum a čas** (tzv. *timestamp*) – presný dátum a čas udalosti s prihliadnutím na časovú zónu

- **užívateľa alebo systémový učet** – ID užívateľa, prípadne meno, ktoré užívateľ v systéme používa alebo systémový účet, ktorý je asociovaný s danou udalosťou
- **použité zariadenie** – napr. zdrojová a cieľová IP adresa alebo inú identifikačnú informáciu (*terminal session ID*, webový prehliadač atď.), ktorá ukáže odkiaľ sa užívateľ prihlásil v čase keď sa daná udalosť stala
- **názov služby/príkazu/aplikácie/udalosti** – medzi udalosti môže napr. patriť zmena hesla používateľa do systému apod.

V prípade dátumu a času je veľmi dôležité, aby sa pri spracovaní auditných záznamov použil rovnaký formát tzv. *timestamp* . Túto synchronizáciu je možné docieliť naprieč zariadeniami, servermi a aplikáciami pomocou časového protokolu NTP [23]. Auditné záznamy a ich formát sa líši naprieč aplikáciami, zariadeniami, rôznymi systémami a operačnými systémami, no ich spoločnou vlastnosťou je to, že obsahujú informácie o tom kto, kedy a čo spravil, a prípadne ako sa systém zachoval.

Auditné záznamy musia byť chránené pred zmenou, odstránením a manipuláciou. Pre auditné záznamy by malo byť podľa Gregory [26] charakteristické:

- nemožnosť zmeny – nikto by nemal byť schopný zmeniť auditný záznam a ideálne by mali byť auditné záznamy ukladané na neprepisovateľné médium
- nemožnosť odstránenia – auditný záznam by nemalo byť možné odstrániť
- nemožnosť neautorizovaného vytvorenia – schopnosť vytvoriť auditný záznam by mali mať len autorizovaní jedinci alebo systémy a navyše samotné vytvorenie auditného záznamu by malo vytvoriť novú udalosť, ktorá zachytí to, že bol vytvorený nový auditný záznam

Podľa Berman [23] patrí medzi najväčšie prínosy auditných záznamoch či už sa jedná o dátové centrá, servery, pracovné stanice alebo aj záznamy z aplikácií:

- podpora zodpovednosti – auditné záznamy môžu byť spolu s kontrolou prístupu nástroj na to, aby sme dokázali identifikovať používateľov, ktorí sú podozrivý napr. z nesprávnej modifikácie prístupových práv alebo údajov
- rekonštrukcia rôznych udalostí – záznamy môžu prispieť k zisteniu a pochopeniu toho ako sa daná udalosť stala, a prípadne môžu proaktívne zabrániť budúcim zlyhaniam

- bezpečnosť a forenzná analýza – záznamy môžu ukázať kto a kedy skopíroval, modifikoval, vymazal špecifické súbory, ale aj to či niekto získal neautorizovaný prístup k užívateľskému účtu, prípadne či nedošlo k neoprávnenej eskalácií práv daného užívateľa

Auditné záznamy sú takisto primárnym cieľom útočníkov, ktorí sa snažia ukryť svoju identitu po preniknutí do daných systémov a vymazať stopy, ktoré po sebe zanechali. Na to, aby sme útočníkom zabránili vykonávať tieto škodlivé aktivity musí byť podľa Berman [23] zaistená dostatočná kontrola prístupu k auditným záznamom ako aj nemožnosť zmeny a odstránenia záznamov.

Každý systém, či už sa jedná o operačný systém, aplikáciu, bezpečnostný softvér atď. si môže záznamy, ktoré generuje ukladať lokálne napr. do súboru (záznamy zostávajú na danom zariadení) alebo druhou možnosťou je odosielať tieto záznamy do ďalších systémov ako sú napr. log manažment softvéry a SIEM (pozri sekciu 2.7), ktoré dokážu agregovať záznamy z viacerých zdrojov, analyzovať a korelovať jednotlivé záznamy naprieč rôznymi systémami atď.

Na záver tejto sekcie je ešte potrebné dodať jednu dôležitú informáciu a to síce, že v tejto diplomovej práci pod pojmom **bezpečnostné auditné záznamy** označujeme všetky kategórie a typy záznamov (pozri tabuľku 3.3), ktoré majú spojitosť s počítačovou bezpečnosťou a z ktorých je možné detegovať rôzne podozrivé aktivity/udalosti.

2.3 Log manažment

Počet, objem a rozmanitosť bezpečnostných záznamov sa podľa Kent a Souppay [17] výrazne mení a pri počte zariadení aj významne zvyšuje, a preto vznikla potreba logovacieho manažmentu a rôznych SIEM systémov. V oblasti logovania záznamoch existujú rôzne auditné logovacie systémy, ktoré sa nachádzajú na sieťových prvkoch a zariadeniach, v rámci rôznych aplikácií a operačných systémov. V rámci tzv. *stand-alone*[7] systémov a IoT zariadení existujú ešte rôzne podtypy záznamov, ktoré môžu zbierať špecifické typy udalostí ako sú napr. rôzne bezpečnostné a systémové udalosti, a udalostí o špecifických službách [23].

Obsahom procesu zaznamenávania záznamov a log manažmentu je aj agregácia veľkého množstva dát z rôznych zdrojov a to prináša výzvy, ktoré musia podľa Berman [23] systémy log manažmentu a tzv. SIEM systémy riešiť - zber dát, ich ukladanie a ochrana, rozbor, syntaktická analýza dát, a prípadne ich ďalšia hlbšia analýza.

[7]Hardvér alebo softvér, ktorý dokáže pracovať nezávisle od iného hardvéru alebo softvéru.

Kent a Souppay [17], ale aj Bhatt, Manadhata a Zomlot [20] tvrdia, že výzvy, ktoré musí log manažment riešiť sa dajú rozdeliť do nasledujúcich kategórií:

- **generovanie a ukladanie záznamov**
  - veľa zdrojov vytvárajúcich záznamy – jeden zdroj dokáže generovať niekoľko záznamov a v systéme je takýchto zdrojov niekoľko
  - nekonzistentný obsah záznamov – zdroje, ktoré vytvárajú záznamy častokrát zaznamenávajú len informácie, ktoré oni považujú za dôležité, a teda môže dochádzať k nekozistentnosti medzi zaznámami, k odlišnej reprezentácií dát, k malému množstvu zaznamenaných informácií, prípadne k informáciam, ktoré nie sú použiteľné apod.
  - nekonzistentné časové razítka (tzv. *timestamp*) – každý systém generujúci záznamy používa vo väčšine prípadov svoj interný zdroj času, ktorý môže byť nepresný
  - nekonzistentné formáty záznamov – záznamy sú ukladané do súborov vo formáte CSV, XML atp., do databáz, pomocou syslogu, protokolu SNMP, ale aj do binárnych súborov; niektoré záznamy sú v štandardných formátoch, iné v propretiárnych atď.
  - potreba definovať obdobie počas ktorého budú záznamy uchovávané (kompromis medzi nákladmi na skladovanie a požiadavkami na analýzu záznamov), ale aj dodržiavanie rôznych predpisov ako je napr. vymazanie citlivých údajov po určitom čase atď.
- **ochrana záznamov** – záznamy obsahujú častokrát citlivé informácie o systémoch, sieťach a aplikáciách, a preto musia byť chránené napr. pred neoprávneným prístupom, porušením integrity záznamov atď.
- **korelácia, vizualizácia a analýza záznamov** – bez hlbšej analýzy záznamov (napr. korelácia a identifikácia tzv. vzorov), bez vizualizácie záznamov, ktorá pomáha pri zhromažďovaní rôznych poznatkov z veľkého množstva záznamov, resp. dát sa hodnota a prínos záznamov výrazne znižuje

Pri výbere riešenia v oblasti logovacieho manažmentu a logovania záznamov je potrebné sa podľa Berman [23] zaoberať:

- normalizáciou dát – na to, aby bola analýza dát efektívna je potrebná ich normalizácia
- syntaktickou analýzou dát – zjednodušuje čítanie, vyhľadávanie a analýzu dát

- notifikáciami, resp. upozorňovaním (tzv. *alerting*) – kľúčový prvok auditu, ktorý uľahčuje proaktívny prístup; príkladom môže byť napr. notifikácia v prípade, že užívateľ vykonal neautorizovanú akciu alebo zadal nesprávne heslo niekoľkokrát za sebou v priebehu jednotiek/desiatok sekúnd

- bezpečnosťou – dôležitou súčasťou celého systému je bezpečný prenos jednotlivých záznamov od zdroja (ktorý vytvára záznamy) k cieli (logovaciemu systému), ktorý docielime využitím rôznych šifrovacích algoritmov pri prenose a uložením rôznych auditných záznamov na bezpečné médium v zašifrovanej podobe

- koreláciou dát – schopnosť vytvárať korelačne pravidlá je takisto dôležitá, pretože systémy musia byť schopné identifikovať postupnosť a nadväznosť jednotlivých udalostí

2.3.1 Infraštruktúra log manažmentu

Podľa Kent a Souppay [17] sa infraštruktúra log manažmentu skladá z hardvéru, softvéru, sietí a médií používaných na generovanie, prenos, ukladanie, analýzu a likvidáciu záznamov. Infraštruktúra sa zvyčajne skladá z týchto troch úrovní:

1. **generovanie záznamov** – táto prvá vrstva obsahuje tzv. hostiteľov, ktorí generujú jednotlivé záznamy; niektorí z týchto tzv. hostiteľov poskytujú svoje záznamy cez sieť tzv. *log* serverom (druhá vrstva) pomocou logovacích klientov, iní poskytujú svoje záznamy napr. pomocou API

2. **analýza a ukladanie záznamov** – táto druhá vrstva sa skladá z jedného alebo viacerých tzv. log serverov, ktoré prijímajú dáta od hostiteľov z prvej vrstvy; servery, ktoré prijímajú záznamy z viacerých zdrojov sa nazývajú *collectors* alebo *aggregators*; záznamy sú buď ukladané priamo na dané *log* servery alebo na separátne databázové servery

3. ***log monitoring*** – táto tretia vrstva obsahuje tzv. *console* (ovládací panel napr. vo forme webového rozhrania), ktorý slúži na monitorovanie a preskúmanie záznamov, a výsledkov z automatizovanej analýzy

Kent a Souppay [17] tvrdia, že jednotlivé typické funkcie infraštruktúry log manažmentu sa dajú rozdeliť do týchto jednotlivých kategórií:

- **všeobecné funkcie**

  - *log parsing* – extrahovanie dát zo záznamov, ktoré môžu byť použité v ďalšom logovacom procese

  - filtrovanie udalostí – odstránenie duplicitných záznamov a záznamov, ktoré neprinášajú žiadnu pridanú hodnotu
  - agregácia udalostí – záznamy, ktoré sú si podobné sa združujú do jedného záznamu, ktorý obsahuje napr. počet výskytov danej udalosti v jednotlivých záznamoch

- **funkcie úložiska**
  - rotácia záznamov (tzv. *log rotation*) – založenie nového tzv. *log* súboru v prípade, že ten starý *log* súbor už je kompletný (šetrenie miesta na lokálnom disku)
  - archivácia záznamov – uchovávanie záznamov na dlhšie časové obdobie, súvisí s rotáciou záznamov
  - kompresia záznamov – ukladanie záznamov vo forme, ktorá nemení ich obsah a šetrí miesto na disku, častokrát vykonávaná pri rotácií a archivácií záznamov
  - redukcia záznamov – odstránenie niekoľkých nepotrebných položiek zo záznamu, čím vznikne nový záznam, ktorý je menší
  - konverzia záznamov – konverzia záznamov z jedného formátu do druhého, napr. z formátu CSV do formátu XML, ktorej obsahom môže byť filtrácia, agregácia a normalizácia záznamov
  - normalizácia záznamov – každá položka záznamu je konvertovaná do konkrétnej reprezentácie dát napr. ukladanie dátumu a času v jednotnom formáte
  - kontrola integrity záznamu - bezpečnostný prvok, ktorý zaistí detekciu zmeny záznamu

- **funkcie analýzy**
  - korelácia udalostí – hľadanie vzťahov medzi dvoma alebo viacerými záznamami, najznámejšou formou je tzv. *rule-based* korelácia, ale korelácie je možné vykonávať aj pomocou štatistických metód a vizualizačných nástrojov
  - zobrazenie a prezeranie záznamov v užívateľský prívetivom formáte
  - *log reporting* – sumarizácia dôležitých aktivít, ktoré sa stali počas určitej doby a zobrazenie výsledkov analýzy jednotlivých záznamov

- **funkcie likvidácie**
  - vymazávanie záznamov – vymazávanie všetkých záznamov, ktoré už nie sú viac potrebné, pretože nie sú dôležité alebo tých, ktoré už boli archivované

Podľa Berman [23] existuje veľké množstvo riešení, resp. softvérov, ktoré poskytujú unifikovaný systém a ktoré obsahujú väčšinu prípadne všetky vyššie spomínané funkcie. Príkladom je napr. open-source riešenie tzv. Elastic stack - ELK (Elasticsearch, Logstash a Kibana), ale aj rôzne SIEM systémy, ktoré sú viac prispôsobené na použitie v oblasti počítačovej bezpečnosti. Benefitom týchto platforiem a riešení je poskytnutie centralizovaného riešenia pre agregáciu, spracovanie, ukladanie (v dostatočnom detaile a po určitú časovú dobu), analýzu záznamov (identifikácia bezpečnostných incidentov, podvodných aktivít atp.) atď.

Popisu týchto systémov a riešení spolu s ich využitím sa budeme venovať v podsekciách 2.7.1 a 2.7.2.

2.4 Rôzne formáty záznamov

Ako už bolo vyššie spomínané formáty záznamov sa môžu naprieč rôznymi systémami a softvérmi, ktoré tieto záznamy vytvárajú líšiť a navyše existujú štandardné, ale aj proprietárne formáty záznamov. V nasledujúcich sekciách sa budeme venovať popisu formátu syslog, rôznych ďalších formátov záznamov webových serverov, ale aj štruktúre NetFlow záznamu, ktorý je primárne generovaný smerovačmi alebo NetFlow sondami[8]. Tento zoznam formátov záznamov, ale nie je konečný, pretože na ukladanie záznamov sa môžu použiť aj iné formáty ako je napr. XML, JSON atď.

Kvôli tomu, že sa formáty záznamov líšia sa samotná analýza záznamov stáva zložitejšou, preto je potrebné pred samotnou analýzou záznamov tieto záznamy spracovať na spoločný formát, čo je možné docieliť napr. pomocou *log parserov*.

2.4.1 Syslog

Syslog podľa Gerhards [27] priradzuje každej správe, resp. záznamu prioritu na základe dôležitosti týchto dvoch nasledujúcich atribútov:

1. typ správy (tzv. *facility*) – príkladom sú napr. *kernel message* (0), *authorization message* (4) atď. celkovo ich syslog definuje 24 (0-23) a ich presný zoznam je možné nájsť na strane č. 10 v zdroji [27]

2. závažnosť (tzv. *severity*) – každý záznam, resp. správa má priradenú tzv. *severity* hodnotu (pozri tabuľku 2.1, čím nižšie číslo, tým viac je daná správa dôležitejšia)

[8]Server vybavený špeciálne upraveným a optimalizovaným operačným systémom Linux a softvérom na generovanie NetFlow štatistík.

Numerický kód	Severita	Význam
0	Emergency	systém je nepoužiteľný
1	Alert	akcia musí byť vykonaná okamžite
2	Critical	kritický stav
3	Error	chybový stav
4	Warning	výstraha/upozornenie
5	Notice	normálny, ale významný stav
6	Informational	informačné správy
7	Debug	*debug-level* správy

Tabuľka 2.1: Syslog formát - numerické kódy, názvy a ich význam

```
<34>1 2003-10-11T22:14:15.003Z mymachine.example.com su - ID47
- BOM'su root' failed for lonvick on /dev/pts/8
```

Obr. 2.6: Príklad záznamu vo formáte syslog podľa RFC 5424 [27]

Následne sa tzv. *priority number* vypočíta ako:

$$priority\_number = facility\_number * 8 + severity\_number$$

Na príklade na obr. 2.6 je možné vidieť, že šlo o neuspešný pokus používateľa lonvick prepnúť sa na super užívateľa (*root*). Z formátu záznamu je možné zistiť, že *priority number* je 34, a tým pádom facility číslo je 4 a severity číslo je 2, čo značí, že ide o tzv. kritický stav podľa tabuľky 2.1.

Viac informácií o formáte syslog je možné nájsť už v spomínanom RFC 5424 (pozri [27]).

2.4.2 Typické formáty záznamov webových serverov

Webový server si zaznamenáva jednotlivé žiadosti o prístupy na jednotlivé stránky (tzv. *page requests*) vo forme záznamov. Obsahom záznamov môže byť napr. typ a verzia webového prehliadača cez ktorý bolo na danú stránku pristúpené a z akej IP adresy, dátum a čas prístupu apod. Podľa Jansen [28] patria medzi populárne formáty záznamov webových serverov NCSA Common Log, NCSA Combined Log, NCSA Separate Log formát a W3C Extended Log file formáty. Podľa Helmy, Wahab, Norzali a kol. [29] patrí medzi známe formáty záznamov webových serverov ešte aj IIS Log File formát.

Všetky formáty záznamov sú v ASCII textovom formáte. V prípade NCSA Log file a W3C Extended Log file formátov platí, že sa roky ukladajú vo formáte štyroch číslíc, v prípade IIS formátu to je vo formáte dvoch číslic po rok 1999, a vo formáte štyroch číslic po roku 1999. Na druhej strane pre formáty NCSA a IIS platí to, že údaje, resp. jednotlivé položky, ktoré sa pre jednotlivé

požiadavky zaznamenávajú zostávajú stále rovnaké. Formát W3C Extended dovoľuje na rozdiel od predchádzajúcich formátov definovať, resp. vyberať jednotlivé položky, ktoré chceme pre požiadavky zaznamenávať.

2.4.2.1 NCSA Log file formát

„*NCSA Common Log formát (taktiež známy ako Access Log formát) obsahuje len základne informácie o žiadosti prístupu na stránku. Obsahuje IP adresu klienta, identifikátor klienta, používateľské meno návštevníka, dátum a čas, HTTP požiadavku (tzv. HTTP request), stavový kód HTTP a počet bajtov prenesených počas požiadavku/žiadosti. Combined Log formát obsahuje rovnaké informácie ako ten predchádzajúci formát (common), ale naviac obsahuje: tzv. referral URL*[9]*, informácie o webovom prehliadači návštevníka, informácie o jeho operačnom systéme a cookie* [10]*. Separate log formát obsahuje rovnaké informácie ako ten predchádzajúci formát (combined), ale rozdeľuje tieto informácie, resp. celý záznam do troch separátnych súborov: access log, referal log a agent log. Položky dátumu a času sú v každom z týchto troch tzv. log files rovnaké.*“[11].

Na obr. 2.7 je možné vidieť príklad záznamu (resp. žiadosti prístupu na stránku) vytvoreného v jednotlivých formátoch NCSA. Naviac platí, že nie všetky položky musia obsahovať nejaké informácie. Pre položky, ktoré neobsahujú žiadne dáta, resp. informácie (na obr. 2.7 to je prípad položky identifikátor klienta) existuje špeciálny znak -, ktorý plní funkciu tzv. zástupného symbolu (z angl. *placeholder*).

2.4.2.2 W3C Extended Log file formát

„*W3C Extended Log file formát je prispôsobiteľný ASCII formát s rôznymi vlastnosťami, resp. položkami, ktoré môže záznam v danom formáte obsahovať. Môžeme zaznamenávať položky, ktoré považujeme za dôležité a súčasne obmedziť veľkosť daného záznamu tým, že vynecháme niektoré nežiadúce položky. Jednotlivé položky sú od seba oddelené medzerami a čas je uložený vo formáte UTC.*“[12]

Na obr. 2.8 je možné vidieť záznam vo formáte W3C Extended. Zoznam jednotlivých položiek, ktoré záznam zaznamenáva začínajú na riadku, ktorý začína reťazcom **#Fields**. Jedná sa teda o položky ako je čas, IP adresa klienta, názov použitej metódy protokolu HTTP, URI stem (detaily pozri dokumentáciu [31]), stavový kód HTTP a verzia protokolu HTTP.

[9] Adresa URL z ktorej bola daná webová stránka navštívená.

[10] Krátke textové súbory vytvárané webovým serverom a ukladané v počítači prostredníctvom webového prehliadača.

[11] Preložené autorom tejto diplomovej práce zo zdroja [28], str. 25.

[12] Preložené autorom tejto diplomovej práce zo zdroja [30].

NCSA Common Log	111.222.125.125 - jimjansen [10/Oct/2009:21:15:05 +0500] "GET /index.html HTTP/1.0" 200 1043
NCSA Combined Log	111.222.125.125 - jimjansen [10/Oct/2009:21:15:05 +0500] "GET /index.html HTTP/1.0" 200 1043 "http://ist.psu.edu/faculty_pages/jjansen/" "Mozilla/4.05 [en] (WinNT; I)" "USERID=CustomerA; IMPID=01234"
NCSA Separate Log	Common Log: 111.222.125.125 - jimjansen [10/Oct/2009:21:15:05 +0500] "GET /index.html HTTP/1.0" 200 1043 Referral Log: [10/Oct/2009:21:15:05 +0500] "http://ist.psu.edu/faculty_pages/jjansen/" Agent Log: [10/Oct/2009:21:15:05 +0500] "Microsoft Internet Explorer - 7.0"

Obr. 2.7: Príklad záznamu (žiadosť prístupu na stránku) vytvoreného v rôznych formátoch NCSA [28]

```
#Software: Internet Information Services 6.0
#Version: 1.0
#Date: 2001-05-02 17:42:15
#Fields: time c-ip cs-method cs-uri-stem sc-status cs-version
17:42:15 172.16.255.255 GET /default.htm 200 HTTP/1.0
```

Obr. 2.8: Príklad záznamu (žiadosť prístupu na stránku) vytvoreného vo formáte W3C Extended[30]

2.4.2.3 IIS Log File formát

„*IIS formát je fixný ASCII formát, ktorý nie je možné modifikovať. IIS formát zaznamenáva viac informácií ako NCSA Common formát. Tento formát obsahuje základné položky ako je IP adresa užívateľa, používateľské meno, dátum a čas požiadavky, stavový kód a počet prijatých bajtov. Navyše, IIS formát obsahuje položky ako je uplynutý čas, počet poslaných bajtov, akciu (napr. prevzatie vykonané príkazom GET) a cieľový súbor. Jednotlivé položky sú oddelené čiarkami, čo robí tento formát jednoduchším na čítanie oproti ostatným ASCII formátov, ktoré na oddeľovanie jednotlivých položiek používajú medzery.*“[12]

Na obr. 2.9 je možné vidieť dva príklady záznamov, ktoré sú uložené vo formáte IIS.

```
192.168.114.201, -, 03/20/01, 7:55:20, W3SVC2, SALES1, 172.21.13.45, 4502, 163, 3223, 200, 0, GET, /DeptLogo.gif, -,
172.16.255.255, anonymous, 03/20/01, 23:58:11, MSFTPSVC, SALES1, 172.16.255.255, 60, 275, 0, 0, 0, PASS, /Intro.htm, -,
```

Obr. 2.9: Príklady záznamov (žiadost prístupu na stránku) vytvorených vo formáte IIS [30]

2.4.3 NetFlow záznam

NetFlow záznam obsahuje dôležité štatistiky o prevádzke/prenose v počítačovej sieti a je možné ho exportovať (záznamy zvyčajne exportuje smerovač alebo NetFlow sonda) a poslať do tzv. NetFlow kolektora[13] v rôznych formátoch. Štruktúra NetFlow záznamu je určená verziou protokolu NetFlow. Podľa Cisco Systems [21] je najobľúbenejším a najpoužívanejším formátom (informácia z októbra 2018) NetFlow formát záznamu verzie 5. Existuje, ale aj NetFlow formát záznamu verzie 9, ktorý je flexibilnejší a obsahuje všetky položky definované vo verzií 5 a navyše ďalšie voliteľné položky (IPv6 adresy a porty, štítky protokolu MPLS atď.). Na záver je potrebné dodať, že existuje aj formát IPFIX, ktorý ako už bolo spomenuté je založený na NetFlow formáte verzie 9.

Ako je možné vidieť na obr. 2.10 jeden záznam NetFlow poskytuje značné množstvo informácií, medzi ktoré patrí napr. počet bajtov a paketov v danom toku/flow[14], informácie o smerovaní, vstupné a výstupné rozhranie atď. (niektoré tzv. exportéry uvádzajú aj hodnotu zdrojového a cieľového autonómneho systému[15]).

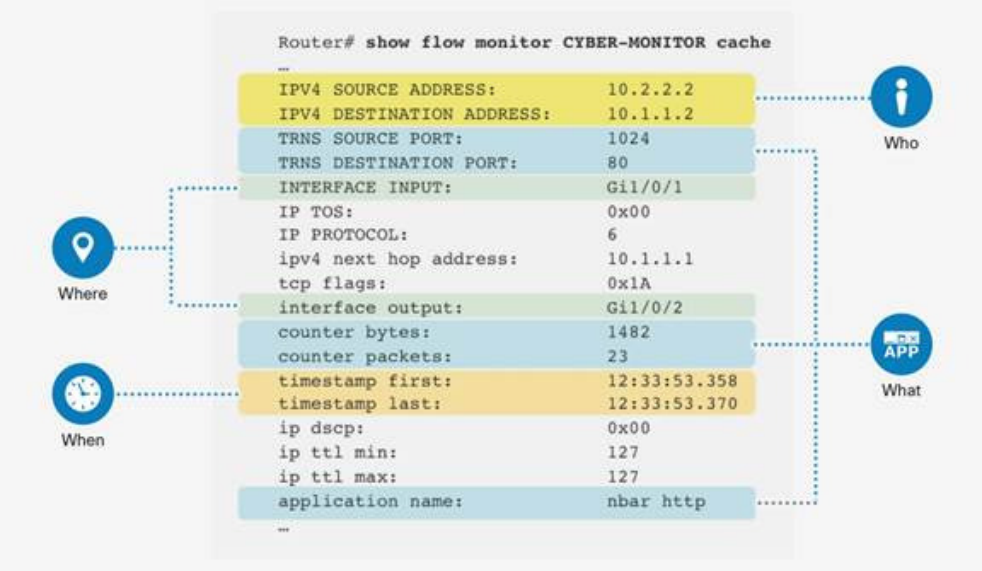
Na záver je potrebné dodať, že existuje aj protokol pcap, ktorý sa používa na zachytávanie sieťového prenosu (paketov) a na rozdiel od Netflow (ktorý nezachytáva pakety, ale „len“ zobrazuje informácie o jednotlivých tokoch/flows) zachytáva tento pcap jednotlivé pakety vrátane tzv. *payload*[16]. Vďaka svojím rozdielom sú NetFlow (zobrazuje informácie/štatistiky o tzv. flows) a pcap (poskytuje hĺbkovú analýzu sieťovej komunikácie) najužitočnejšími vtedy, ak sa používajú spolu.

[13]Nástroj, ktorý spracováva NetFlow záznamy za účelom vykonania analýzy, ale aj prezentácie v užívateľsky prívetivom formáte.

[14]Tok/flow je v terminologií NetFlow definovaný ako sekvencia paketov s rovnakou zdrojovou/cieľovou IP adresou, zdrojovým/cieľovým portom a číslom protokolu. Pre každý tok sa zaznamenávajú informácie ako je doba jeho vzniku, dĺžka trvania, počet prenesených paketov a bajtov atď.

[15]Počítačová sieť alebo skupina počítačových sietí, ktoré sú všetky riadené a kontrolované jedným subjektom alebo organizáciou.

[16]Dátový obsah, užitočné informácie prenášané pri prenose dát (časť prenášaných dát, ktorá signalizuje hlavný účel prenosu.



Obr. 2.10: Príklad záznamu vytvoreného vo formáte NetFlow [21]

2.5 Logovacie úrovne

Podobne ako to platí u formátoch jednotlivých záznamov, že si každý systém, resp. aplikácia môže vymyslieť svoj „vlastný“ formát v ktorom bude vytvárať a následne ukladať jednotlivé záznamy, tak podobne to je aj v prípade tzv. logovacích úrovni. Platí, že si v podstate každý systém, resp. aplikácia môže vymyslieť svoje „vlastné“ tzv. logovacie úrovne. Závažnosť (z angl. *severity*) záznamov (v prípade syslog formátu), resp. logovacie úrovne, ktoré sú obsahom jednotlivých záznamov sú určené (slúžia) na kategorizáciu jednotlivých záznamov podľa závažnosti, a to už či z pohľadu prevádzky daného systému a/alebo z pohľadu bezpečnosti.

V prípade záznamov vo formáte syslog sa definujú (ako už bolo vyššie spomínané) tzv. *severity* levely (pozri sekciu 2.1). Rôzne ďalšie logovacie softvéry ako je napr. log4j a log4net používajú tzv. logovacie úrovne, ktoré definujú úrovne od najzávažnejšieho FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL až po OFF. Bližší popis jednotlivých tzv. logovacích úrovni je možné nájsť napr. v zdroji [32].

Vyššie spomenuté *severity* levely, resp. logovacie úrovne boli len ukážkou toho ako sa môže závažnosť jednotlivých udalostí zaznamenávať, no v dalších iných systémoch a softvéroch budú podľa Dietrich [33] koncepty veľmi podobné.

2.6 Systémy určené na zber záznamov

Lokálne ukladanie záznamov napr. do súboru alebo do databázy, ktorá sa nachádza lokálne na danom systéme, ktorý záznamy zaznamenáva so sebou prináša rôzne nevýhody ako je napr.:

- pri veľkom počte systémov je prezeranie jednotlivých záznamov ktoré sú uložené na jednotlivých systémoch takmer nemožné a zaberá to veľa času
- korelácia záznamov naprieč rôznymi systémami je nemožná, pretože jeden systém nemá záznamy toho druhého a naopak atď.
- v prípade, že dôjde k poškodeniu lokálneho úložiska (na ktoré sa záznamy ukladajú) a neexistuje žiadna záloha môžu byť záznamy nenávratne preč alebo napr. v prípade sieťových zariadení, ktoré môžu záznamy ukladať na tzv. volatilné médium dôjde k strate záznamov takmer ihneď po vypnutí/reštarte daného zariadenia

Systémy určené na zber záznamov sú práve riešením na tieto vyššie spomínané problémy. Tieto tzv. *log collectors* dokážu zbierať záznamy z rôznych zdrojov naprieč IT prostredím danej organizácie a preposielať ich do ďalších systémov, ktoré dokážu tieto záznamy spracovať, analyzovať, vizualizovať atď.

Logovacích nástrojov existuje veľké množstvo a každý má svoje výhody, ale aj nevýhody. V následujúcich dvoch podsekciách sú popísané nástroje syslog-ng a fluentd, ale napr. v prípade syslog-ng existujú ešte dva podobné projekty a to rsyslog a syslog. Na druhej strane existujú tzv. *application logging frameworks*, resp. knižnice jednotlivých programovacích jazykov. Tieto nástroje slúžia na štandardizáciu procesu logovania záznamov v aplikáciach. Aplikácie sú napísané v rôznych programovacích jazykoch, a preto napr. v prípade programovacieho jazyku Java je to logovací nástroj Log4j prípadne Log4j 2, v prípade programovacieho jazyku C++ je to Pantheios atď.

Na záver je potrebné dodať, že v oblasti počítačových sietí sa o zber NetFlow záznamov starajú NetFlow sondy, ktoré sú špecializované zariadenia slúžiace na monitorovanie a export NetFlow štatistík. Tieto sondy je možné pripojiť v sieti do ľubovoľného bodu a následne odosielať exportované štatistiky do NetFlow kolektora, ktorý buď býva súčasťou typickej architektúry SIEM systémov (pozri sekciu 2.7 a obr. 2.15) alebo vo forme integrácie do SIEM systémov napr. pomocou riešenia od firmy Flowmon [34], ktoré je kompatibilné s rôznymi SIEM systémami ako je napr. LogRhythm (2.7.1.2), ArcSight (2.7.1.3) atď.

2.6.1 Syslog-ng

Syslog-ng [35] je aplikácia, resp. softvér, ktorej hlavným cieľom je zber záznamov z rôznych zariadení a systémov (operačné systémy, aplikácie atď.) a ich

následne ukladanie na jeden centrálny tzv. *log* server. Posielaním záznamov (vytvárajú sa na jednotlivých systémoch a zariadeniach) na vzdialené tzv. *log* servery je možné docieliť centrálny zber a ukladanie záznamov. Využitím protokolov TCP a TLS je možné zaistiť to, že sa nestratí žiaden záznam pri prenose a navyše bude komunikácia šifrovaná, pretože obsahom záznamov môžu byť citlivé informácie. Navyše ukladanie záznamov v centrálnej lokalite zjednodušuje proces archivácie záznamov, a tým je možné splniť rôzne politiky a regulácie rôznych organizácií.

Syslog-ng obsahuje aj tzv. *built-in* parsery pomocou ktorých je možné si poradiť s neštrukturovanými záznamami, ktorých je častokrát veľké množstvo. Syslog-ng podporuje syslog formáty podľa RFC 3164 [36], RFC 5424 (ktorý už bol spomínaný v podsekcií 2.4.1), ale podporuje aj JSON formát a tzv. journald formát správ. Syslog-ng dokáže konvertovať záznamy do predefinovaných formátoch, filtrovať záznamy na základe určitých jednoduchý kritérií atď., avšak nie je určený na analýzu záznamov.

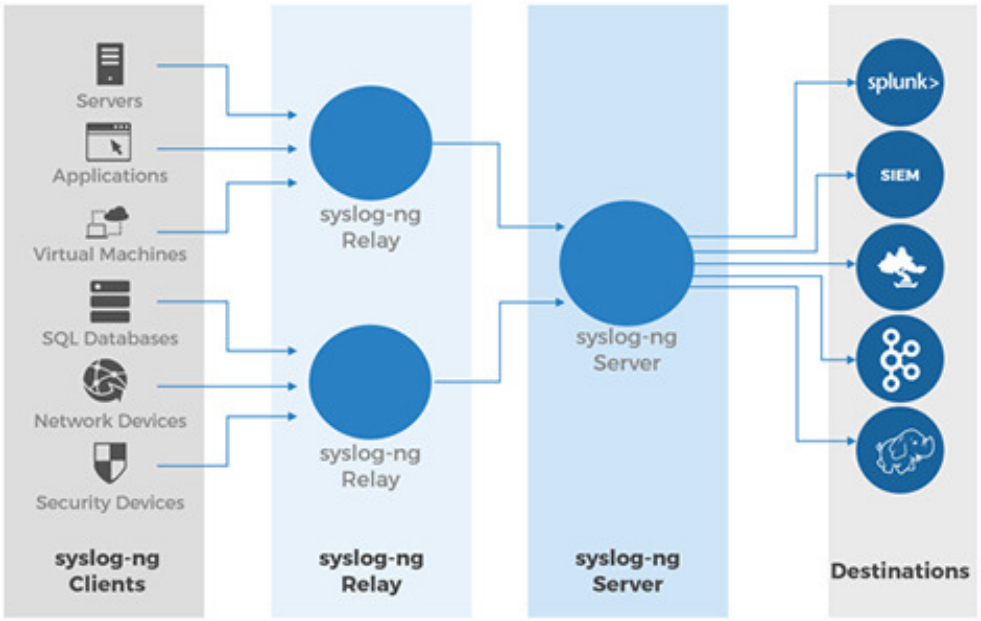
V syslog architekúre existujú tzv. syslog-ng klienti, ktorými sa označujú zariadenia na ktorých beží aplikácia syslog-ng, ktorej úlohou je zber záznamov z rôznych aplikácií, súborov a ďalších zdrojov, ktoré sú súčasťou systému na ktorom beží daný syslog-ng klient. Následne môžu byť záznamy posielané do tzv. syslog-ng *relay* (určených na tzv. *buffering* záznamov v prípade krátkodobých výpadkov) prípadne syslog-ng klienti rovno posielajú všetky dôležité záznamy do vzdialeného syslog-ng servera. Syslog-ng server ich uloží a/alebo prípadne rovno pošle do systémov, ktoré sú určené na ďalšie spracovanie, analýzu, vizualizácií záznamov (viac informácií v sekcií 2.7).

2.6.2 Fluentd

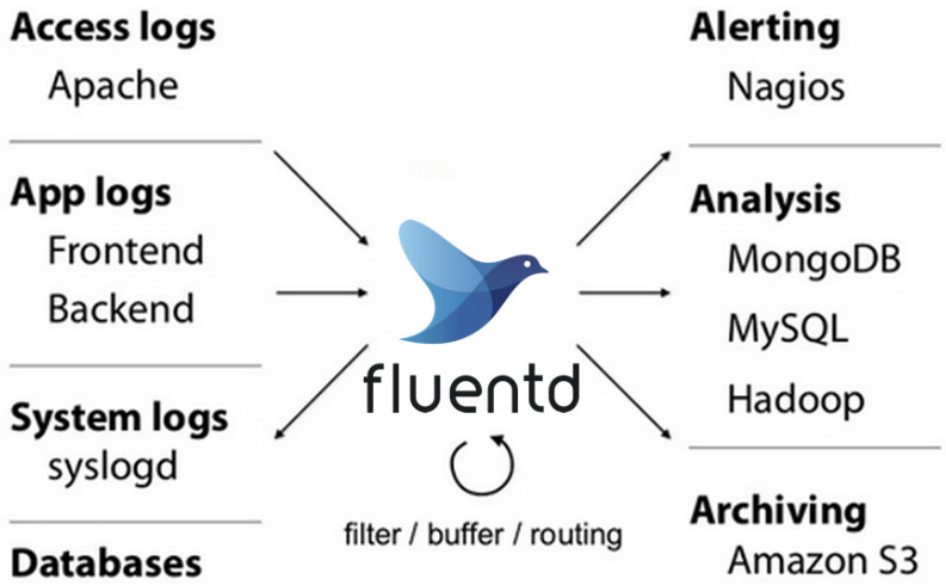
Fluentd [37] je *open-source* softvér, ktorý umožňuje zber dát, resp. záznamov z rôznych zdrojov a ich následnú transformáciu primárne do JSON formátu. Následne dokáže tieto záznamy posielať do rôznych analytických nástrojov ako je napr. ElasticSearch atď. Navyše existuje množstvo tzv. *pluginov*, ktoré umožňujú rozšíriť funkcionalitu softvéru fluentd. Na obr. 2.12 je možné vidieť ako softvér fluentd funguje.

2.7 SIEM systémy a softvéry určené na spracovanie a analýzu záznamov

Na začiatku tejto sekcie sa budeme venovať motivácií zavedenia SIEM systémov, ich definícií, architektúre, ale aj hlavným funkciám a výzvam, ktoré SIEM systémy musia riešiť. Ďalej popíšeme niektoré vybrané SIEM systémy od firiem ako je IBM (QRadar), LogRhythm (LogRhythm Security Intelligence Platform) a Micro Focus (ArcSight ESM). V tejto sekcií sa budeme venovať aj tzv. Elastic Stack, ktorý je spojením *open-source* produktov ElasticSearch,



Obr. 2.11: Architektúra syslog-ng [35]



Obr. 2.12: Architektúra fluentd [37]

Logstash a Kibana. Na konci tejto sekcie s budeme venovať produktom na spracovanie a analýzu záznamov v cloud platformách[17], konkrétne sa bude jednať o AWS a Microsoft Azure.

Na záver je potrebné dodať, že SIEM systémy a *log* manažment produkty (softvéry) sa už v súčasnosti veľmi prelínajú, avšak ešte stále existujú kategórie „čistého“ log manažment softvéru, ktoré sa neusilujú o vízie, resp. funkcie strojového učenie a umelej inteligencie, ktoré už majú rôzne SIEM softvéry a SOC[18]. Príkladom sú napr. produkty ako je Papertrail, Graylog a od spoločnosti SolarWinds napr. Kiwi Syslog Server a Security Event Manager, ktorým sa, ale táto diplomová práca venovať nebude.

2.7.1 SIEM

V minulosti existovalo podľa Bhatt, Manadhata a Zomlot [20] len málo bezpečnostných nástrojov, ktoré by sa starali o bezpečnosť danej organizácie. Boli to napr. *firewall*, IDS, antivírusové programy atď., a každý z týchto systémov mal svoje vlastné užívateľské rozhranie. Postupne sa tieto nástroje začali čoraz viac používať a začali sa objavovať aj ďalšie rôzne nástroje, a práve preto podľa Bhatt, Manadhata a Zomlot [20] vznikli dva problémy:

1. príliš veľké množstvo rôznych užívateľských rozhraní
2. neexistencia nástrojov na koreláciu udalosti medzi rôznymi bezpečnostnými nástrojmi

Práve SIEM systémy boli navrhnuté tak, aby tieto problémy, resp. výzvy dokázali riešiť. SIEM je manažement bezpečnostných informácií a udalostí, ktorý je podľa Micro Focus [38] spojením a určitou kombináciou týchto dvoch systémov:

1. SIM (Security Information Management) - SIM systémy, ktoré často bežia ako softvéroví agenti na monitorovaných zariadeniach slúžia na zber bezpečnostných záznamov napr. z IDS, proxy serverov atď. a ich ukladanie do centrálneho úložiska na neskoršiu analýzu, obsahom niektorých SIM systémov je aj normalizácia dát, resp. záznamov pred ich odoslaním do centrálneho úložiska
2. SEM (Security Event Management) - SEM systémy sú určené na identifikáciu, analýzu a na monitorovanie hrozieb v reálnom čase, na vizualizáciu dát a koreláciu udalostí, ale aj na vytváranie upozornení (z angl. *alert*) v reálnom čase

[17] Cloud platforma poskytuje nástroje, ktoré podniky potrebujú pre vývoj, nasadenie a chod aplikácií bez nákladov a zložitosti tvorby, a riadenia potrebnej platformy a infraštruktúry.

[18] Riešenie, ktoré zaisťuje komplexnú centralizáciu riadenia bezpečnostných udalosti a incidentov v jednom bode s cieľom minimalizácie reakčnej doby na incidenty.

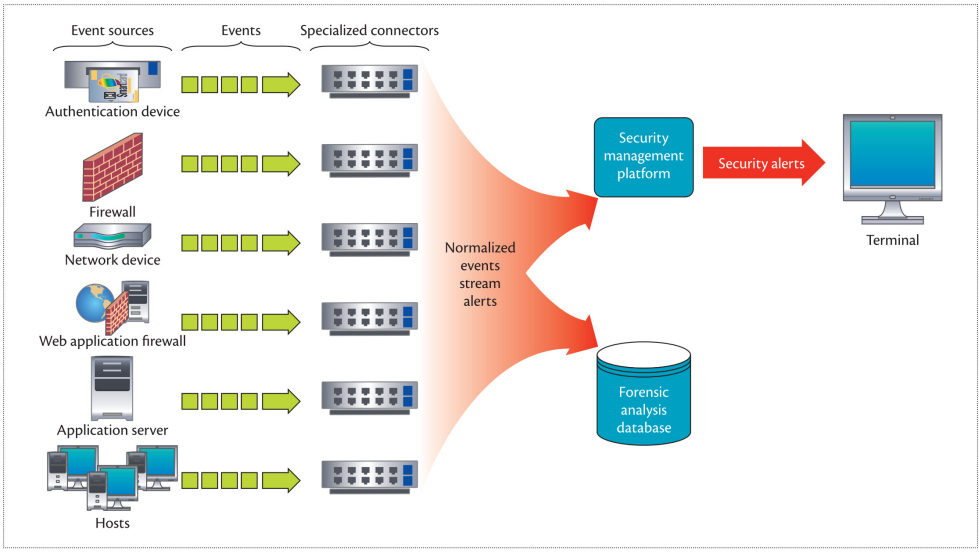
SIEM sú podľa Bhatt, Manadhata a Zomlot [20] dôležitým nástrojom, resp. veľmi dôležitou súčasťou tzv. SOC, pretože dokážu zbierať, normalizovať a analyzovať rôzne bezpečnostné udalosti, ktoré sú generované z rôznych zdrojov a dokážu pomôcť CSIRT[19] tímom, ktorí sú zodpovedný za prijímanie, posudzovanie a reagovanie na rôzne aktivity a incidenty, ktoré sa týkajú počítačovej bezpečnosti.

Na obr. 2.13 je možné vidieť jednotlivé komponenty typickej architektúry SIEM systémov podľa Bhatt, Manadhata a Zomlot [20]. Ako je možné vidieť na tomto obrázku tak SIEM systémy prijímajú vstupy z rôznych zariadení ako sú napr. sieťové prvky, aplikačné servery, *firewall* atď. Každé zariadenie generuje udalosti v rôznych formátoch a úlohou SIEM systémov je normalizovať tieto rôzne reprezentácie dát do jednotného formátu, aby sa napr. zjednodušilo ich ďalšie spracovanie, tvorba rôznych pravidiel atď. Ako je znázornené na obr. 2.13 tak to prijímanie jednotlivých udalosti prípadne záznamov je úlohou tzv. SIEM konektorov, ktoré vykonávajú syntaktickú analýzu udalosti/záznamov a konvertujú ich do jednotného formátu. Po tom ako sú jednotlivé udalosti, resp. záznamy tzv. normalizované posielajú sa do tzv. *forensic analysis* databáze a do platformy riadenia bezpečnosti (tzv. *security management platform*). Platforma udržiava a analyzuje jednotlivé udalosti (tie ktoré boli pozorované v posledných hodinách), a ak je potrebné tak archivačná databáza uchováva jednotlivé udalosti na dlhšie časové obdobie v prípade potreby ďalšieho forenzného vyšetrovania. Jednotlivé pravidlá vytvorené v tzv. manažment platforme sa aplikujú pravidelne na jednotlivé prichádzajúce udalosti a v prípade, že niektoré z pravidiel vytvorí nové upozornenie (z angl. *alert*) tak dôjde k tomu, že sa toto nové upozornenie pošle do terminálu SIEM systému na analýzu SOC analytikom. Jednotlivé pravidla sa snažia hľadať v jednotlivých udalostiach rôzne škodlivé aktivity ako je napr. vysoký počet neúspešných pokusoch o prihlásenie sa do systému, zvýšený počet tzv. HTTP požiadaviek na známe škodlivé stránky atp. Pravidlá môžu byť podľa Bhatt, Manadhata a Zomlot [20] generované, resp. vytvárané:

1. bezpečnostným analytikom

2. SIEM systémom, ktorý môže algoritmicky generovať pravidlá z jednotlivých udalosti (napr. pomocou tzv. *pattern mining* (pozri Mooney a Roddick [39])), prípadne niektoré pravidlá môžu využívať detekciu anomálií (práve to sú miesta, kde môže strojové učenie uľahčiť identifikáciu podozrivej aktivity pozri sekciu 2.8)

[19] Skratka bezpečnostného tímu, ktorý je určený pre koordináciu riešenia bezpečnostných počítačových incidentov.



Obr. 2.13: Typická architektúra SIEM systémov[20]

Medzi hlavné funkcie SIEM systémov patrí podľa Bhatt, Manadhata a Zomlot [20], ale aj podľa Scarfone [40]:

- agregácia dát z rôznych zdrojov ako sú napr. rôzne sieťové zariadenia, servery, databázy, operačné systémy, aplikácie atď.
- korelácia podobných udalosti generovaných z viacerých rôznych zdrojov, ktorá je dôležitá napr. pri detekcií škodlivej aktivity
- automatická analýza korelovaných udalostí a generovanie upozornení v reálnom čase na bezpečnostné hrozby v systéme
- zobrazovanie rôznych informácií vo forme informačných grafov, čo poskytuje pohľad na aktuálny stav daného systému ako celku
- generovanie tzv. *compliance reportov* (HIPAA, SOX, PCI DSS atď.) pre účely auditu

Na druhej strane medzi tzv. operačné, resp. prevádzkové výzvy, ktoré musia CSIRT, resp. bezpečnostní analytici pri používaní SIEM systémov riešiť patria podľa Bhatt, Manadhata a Zomlot [20]:

1. vytváranie a používanie jednotlivých pravidiel – problémom je hlavne vysoký počet tzv. falošných poplachov, ktoré majú vytvorené SIEM pravidlá tendenciu spúštať, a preto musia mať tieto pravidlá extrémne

nízky FPR[20] na to, aby boli použiteľné v praxi, no na druhej strane, ak bezpečnostní analytici napíšu veľmi špecifické pravidlá na zachytenie daného špecifického útoku nemusia zachytiť iné formy útokov (práve to je jedno z miest, kde môže strojové učenie uľahčiť identifikáciu podozrivej aktivity (pozri sekciu 2.8))

2. nedostatok tzv. kontextových informácií – aktivity ako je napr. zálohovanie, testovanie a tzv. *patching*[21] môžu častokrát spustiť alarmy, ktoré sú určené na detekciu narušenia bezpečnosti a vytvoriť tak zbytočnú réžiu, ktorá je spojená so „zbytočnou" investigáciou (riešením daného *false positive* alarmu)

3. nevyužívanie tzv. *long-term* dát – väčšina bezpečnostných analytikov sa zameriava na funkcionalitu tzv. *short-term* (krátkodobého) upozorňovania a ignoruje, resp. nevyužívajú funkciu tzv. *long-term* (dlhodobej) retencie záznamov (analytici zvyčajne monitorujú úzke časové okno jednotlivých udalosti), čím sa obmedzuje schopnosť zachytiť pomaly sa rozvíjajúce útoky najmä tzv. APT[22], a preto by bezpečnostní analytici mali venovať väčšiu pozornosť tzv. retenčným funkciám SIEM systémov a vytvoriť analytické riešenie (napr. opäť pomocou strojového učenia), ktoré môže pomôcť pri odhaľovaní jednotlivých vzorov tzv. pomalých útokov, resp. APT atď.

Medzi technické výzvy, ktoré musia SIEM systémy takisto riešiť patrí podľa Bhatt, Manadhata a Zomlot [20] zber udalosti, resp. záznamov, ich ukladanie, analýza a vizualizácia. Bližší popis jednotlivých výziev je možné nájsť v sekcií *log* manažment (pozri sekciu 2.3).

Podľa Bhatt, Manadhata a Zomlot [20] v časoch keď sa v infraštruktúrach rôznych firiem a organizácií nenachádzalo veľa systémov, aplikácií, sieťových prvkoch atď. a generovalo sa malé množstvo záznamov, resp. udalosti tak SIEM systémy neboli populárne medzi administrátormi a bezpečnostnými analytikmi. Avšak potom čo začali firmy a organizácie expandovať začal narastať aj počet generovaných záznamov, resp. udalosti z rôznych zdrojov v rámci firiem a organizácií atď., a tým pádom narastal aj záujem o SIEM systémy. V súčasnosti sú podľa Bhatt, Manadhata a Zomlot [20] SIEM systémy pravdepodobne najdôležitejšími nástrojmi SOC, aj preto, že sa stali nenahraditeľným nástrojom na zvládnutie rôznych bezpečnostných udalosti a incidentov, ktoré

[20]Podiel negatívnych prípadov, ktoré sú nesprávne identifikované ako pozitívne prípady (výsledok testu, ktorý nesprávne indikuje, že je prítomná konkrétna podmienka alebo atribút).

[21]Súbor zmien, ktorý vedie napr. k aktualizácií systému/softvéru, k jeho oprave, prípadne k zlepšeniu jeho jednotlivých funkcií.

[22]Typ útoku pri ktorom osoba alebo skupina osôb získa neoprávnený prístup do systému a po určitú časovú jednotku bude „skrytá", resp. neodhalená.

Obr. 2.14: Magic Quadrant SIEM systémov [41]

sa v rámci organizácií, resp. firiem objavujú. Bhatt, Manadhata a Zomlot očakávajú, že tento trend (záujem o SIEM systémy) bude naďalej pokračovať.

Podľa firmy Gartner[23] a jej magického kvadrantu (obr. 2.14) z decembra 2018 patrili medzi lídrov v oblasti SIEM systémov produkty od firiem ako je Splunk, IBM, LogRhythm atď. V nasledujúcich podsekciách budú popísané jednotlivé vybrané produkty, ktoré patria do kvadrantu lídrov, a to IBM QRadar, LogRhythm Security Intelligence Platform, ale aj ArcSight od firmy Micro Focus (ktorá spadá pod koncern Hewlett-Packard), ktorý síce do kvadrantu lídrov nepatrí, ale patrí do tzv. *challengers*, a je prítomný v laboratóriu etického hackování na Fakulte informačních technológií ČVUT.

[23] Americká spoločnosť zaoberajúca sa výskumom a poradenstvom v oblasti IS/ICT technológií.

2.7.1.1 IBM QRadar

IBM QRadar Security Information and Event Management (SIEM) dokáže podľa IBM [42] detegovať a prioritizovať hrozby, konsolidovať záznamy z tisícok zariadení, koncových bodov a aplikácií, ktoré sú distribuované v celej sietí a nasledne korelovať rôzne informácie a agregovať súvisiace udalosti do jednotlivých výstrah, čím pomôže bezpečnostným tímom rýchlo reagovať a znížiť dopady rôznych incidentov. QRadar je navrhnutý tak, aby poskytoval centralizovaný náhľad do záznamov a bezpečnostných dát celého podniku. Tento produkt obsahuje rôzne zabudované prípady užitia (z angl. *use cases*), pravidlá a politiky určené na rôzne korelácie v reálnom čase, a to všetko na detekciu známych, ale aj neznámych hrozieb (identifikácie zmien v správaní pomocou rôznych metód detekcie anomálií). QRadar poskytuje aj transparentnosť, zodpovednosť a merateľnosť, čím dokáže splniť rôzne regulačné podmienky a *compliance* požiadavky[24].

Medzi základne funkcie QRadar patrí podľa IBM [42]:

- prijímanie veľké množstva dát z *on-premise*[25] a cloudových zdrojov
- *built-in* analytika (sada pravidiel pre detekciu najčastejších hrozieb)
- korelovanie rôznych udalostí a informácií za účelom prioritizácie hrozieb
- automatické tzv. parsovanie a normalizácia záznamov
- tzv. *threat intelligence* a podpora pre STIX/TAXII[26]
- podpora zhody tzv. (compliance) s GDPR, FISMA, SOX atď.
- integrácia so 450 riešeniami - pomocou tzv. Device Support Modules (DSMs)
- pomocou tzv. *User Behavior Analytics* modulu je možné analyzovať správanie používateľov a odhaliť rizikové profily používateľov, ktoré sa v sietí nachádzajú

IBM QRadar SIEM je hlavným prvkom tzv. IBM QRadar Security Intelligence Platform, ktorá spája log manažment, SIEM, analýzu sietí a správania užívateľov, *vulnerability management*[27], *threat intelligence*[28] a rôzne investigatívne riešenie poháňané algoritmami umelej inteligencie a strojového učenia

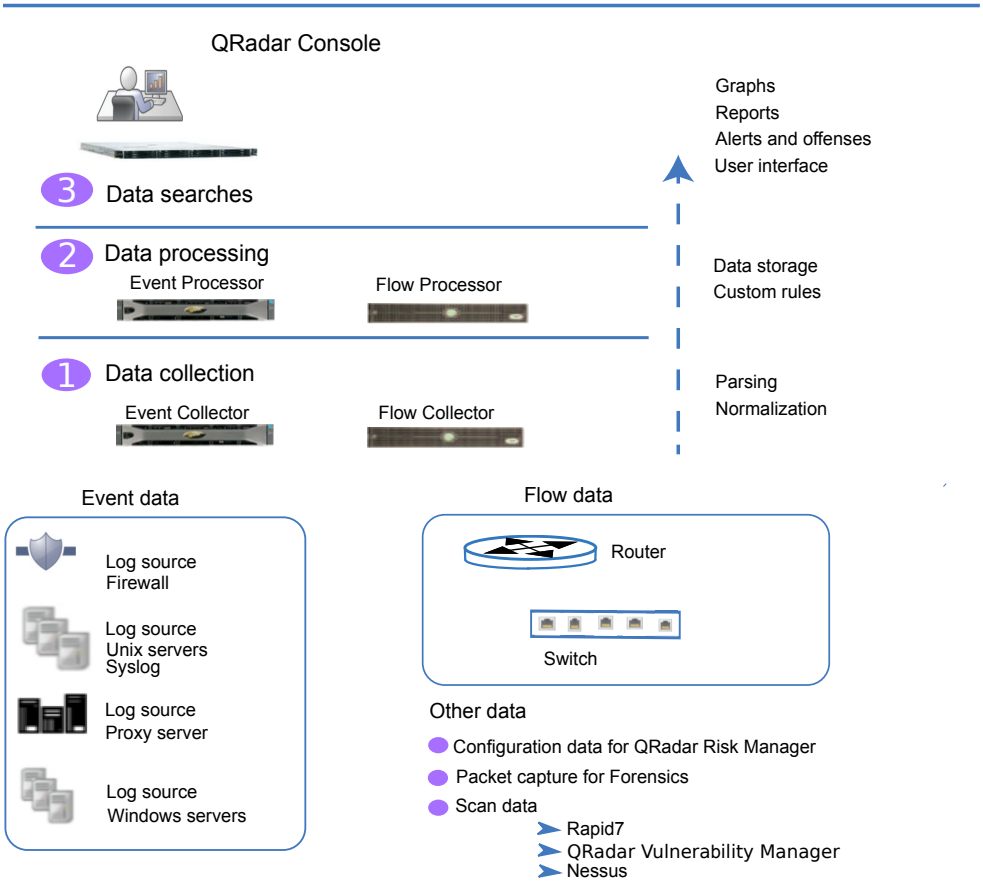
[24]GDPR, FISMA, SOX, HIPAA, ISO 27001, PCI DSS atď.

[25]Softvér, ktorý je nainštalovaný lokálne na serveroch prípadne počítačoch.

[26]Štandardy vyvinuté v snahe zlepšiť prevenciu a zmiernenie kybernetických útokov.

[27]Cyklický postup identifikácie, klasifikácie, stanovovania priorít, náprav a zmierňovania zraniteľností softvéru.

[28]Informácie, ktoré organizácia používa na porozumenie, prípravu, prevenciu a identifikáciu počítačových hrozieb.



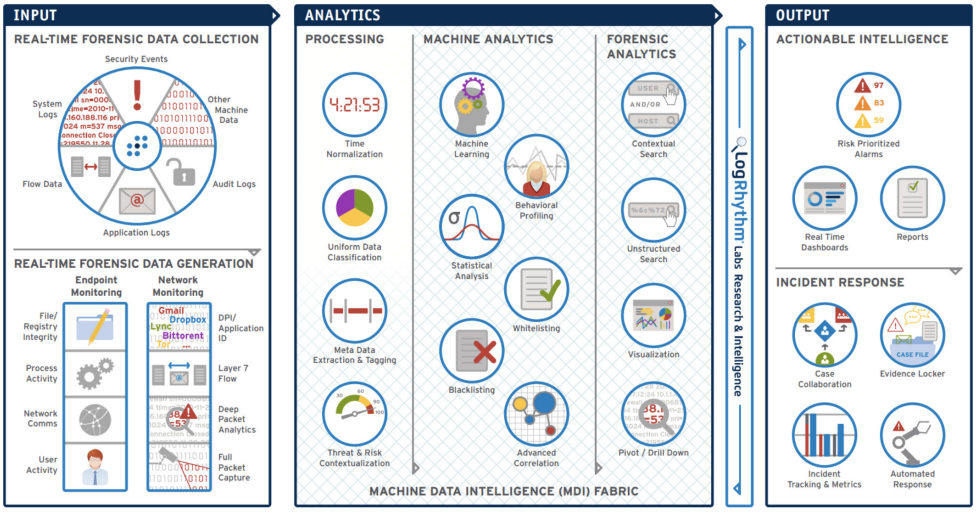
Obr. 2.15: Architektúra QRadar [43]

(jedná sa teda hlavne o algoritmy pre detekciu anomálií) do jednej platformy, ktorá je spravovaná z jedného rozhrania. Na obr. 2.15 je možné vidieť architektúru QRadar.

2.7.1.2 LogRhythm Security Intelligence Platform

LogRhythm [44] poskytuje platformu pod názvom Security Intelligence Platform, ktorá obsahuje riešenie pre riadenia životného cyklu hrozieb, SIEM, *log* manažment, monitorovanie koncových zariadení a sietí vrátane forenznej analýzy a bezpečnostných analytických nástrojov. Táto platforma dokáže detegovať širokú škálu indikátorov potenciálnej kompromitácie tzv. IoC[29], čím umožňuje okamžitú reakciu a aplikáciu preventívnych opatrení.

[29]Dôkaz, ktorý indikuje, že došlo ku kybernetickému útoku.



Obr. 2.16: LogRhythm Security Intelligence Platform [45]

Podľa LogRhythm [44] patria medzi kľúčové vlastnosti produktu:

- parsovanie záznamov a normalizácia pravidiel pre viac ako 700 unikátnych operačných systémov aplikácií, zariadení, databázy atď., a schopnosť zbierať a spracovať 300 000 správ za sekundu
- analýza veľkého objemu dát a ich vizualizácia
- pokročilá korelácia a rozpoznávanie vzorov
- detekcia anomálií chovania na úrovni užívateľov (*User Behavior Anomaly Detection*), sietí (*Network Behavior Anomaly Detection*), ale aj koncových zariadení (*Host Behavior Anomaly Detection*)
- automatizované nástroje pre riadenie zhody (z angl. *compliance*) s ISO 27001, PCI, SOX, HIPAA, FISMA, GLBA a ďalšími

Na obr. 2.16 je možné vidieť architektúru LogRhythm Security Intelligence platformy.

2.7.1.3 ArcSight ESM

ArcSight Enterprise Security Manager je podľa Micro Focus[30] [46] platforma určená na detekciu hrozieb v reálnom čase, na analýzu rôznych udalostí, ktorá poskytuje centralizovaný pohľad do rôznych prostredí v rámci organizácie, čím dokáže zjednodušovať niektoré procesy v organizácií. Pomocou detekcie hrozieb dokážu SOC tímy riešiť rôzne incidenty rýchlo a minimalizovať falošné

[30]Firma, pod ktorú tento softvér patrí.

poplachy. Tento nástroj dokáže analyzovať dáta z viac ako 500 rôznych zariadení a tzv. *ArcSight's ADP SmartConnectors* podporujú rôzne formáty od natívnych Windows udalostí, syslogu, záznamov z *firewall*, Netflow, XML/JSON atď.

Medzi niektoré z hlavných benefitov podľa Micro Focus [46] patrí:

- korelácia v reálnom čase – korelácia až 100 000 udalostí za sekundu v rámci organizácie
- kategorizácia a normalizácia – zozbierané záznamy sa konvertujú do univerzálneho CEF formátu, ktorý sa používa v tomto SIEM produkte a ktorý bol vyvinutý spoločnosťou Micro Focus
- modulárny rozvoj obsahu – možnosť zdieľania vlastného obsahu (vytvorených pravidiel, správ tzv. *reportov*, *dashboard* atď.) jednoduchou formou pomocou modulov a následne nasadenie týchto modulov na iné systémy, prípadne ich zdieľanie s ArcSight komunitou
- integrácia s produktom ArcSight Investigate – pokročilá analytická platforma na vytvorenie rýchleho a intuitívneho vyhľadávania a vizualizácie dát
- integrácia medzi ArcSight a tzv. zariadeniami tretích strán (z angl. *third-party*) – táto integrácia umožňuje spúšťať príkazy na tzv. *third-party* zariadeniach z ArcSight konzole (spustenie príkazu na zariadení a následne poslanie výstupu naspäť do konzole pre účely ďalšej analýzy)

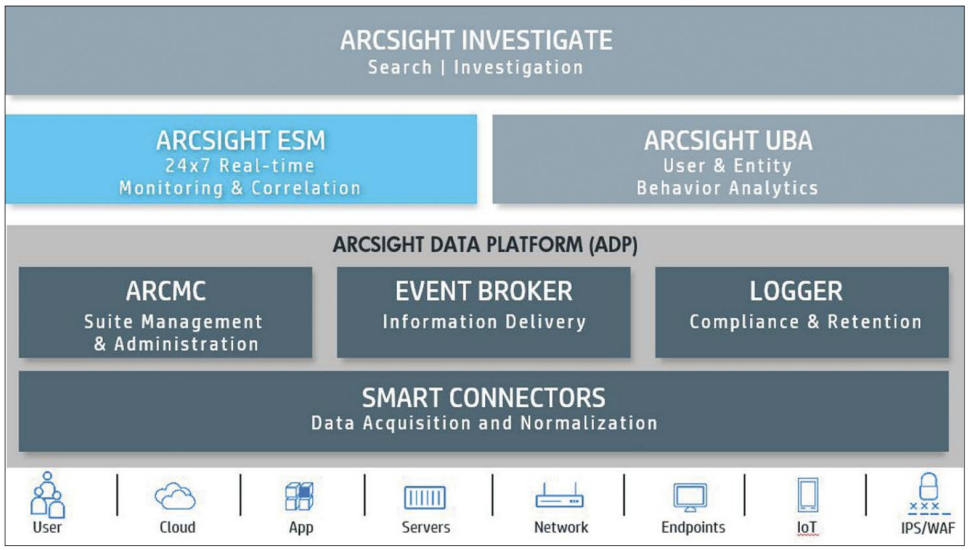
Na obr. 2.17 je možné vidieť portfólio ArcSight, ktorého súčasťou je produkt ArcSight ESM, ktorému sme sa venovali v tejto sekcií.

2.7.1.4 Záverečné zhrnutie a porovnanie

K popisu vyššie spomínaných SIEM produktov, resp. systémov (ich základných vlastností, hlavné benefity atď.) sme využili oficiálne texty z webových stránok jednotlivých firiem, ktoré tieto produkty predávajú, resp. sú ich vlastníkmi (*product overview*/*white paper* atď.). Bez hlbšej analýzy jednotlivých generických a niekedy aj „marketingových" termínov, ktoré sú využívané pri popise benefitov jednotlivých produktov sa dajú ťažko medzi sebou jednotlivé produkty porovnať. Cieľom tejto diplomovej práce nie je hlbšia analýza a porovnanie SIEM systémov, ale len úvod a popis ich základných vlastností a benefitov – „obecná" rešerš.

Na základe vyššie uvedených skutočností patria medzi spoločné vlastnosti vyššie spomínaných SIEM produktov:

- podpora rôznych formátov záznamov a druhov udalostí z rôznych zdrojov



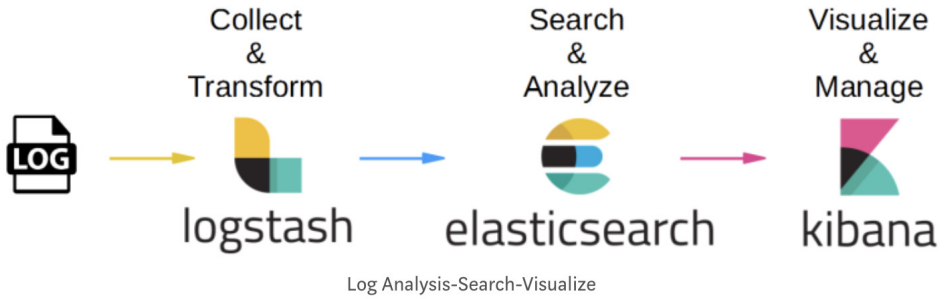
Obr. 2.17: Portfólio ArcSight [46]

- tzv. parsovanie a normalizácia záznamov
- korelácia rôznych udalostí v rámci organizácie
- vyhľadávanie, analýza a vizualizácia dát
- podpora zhody (súladu) (z angl. *compliance*) s FISMA, SOX, HIPA, ISO 27001 atď.

Oblasťou v ktorej sa od seba jednotlivé produkty výraznejšie odlišujú sa zdá byť oblasť umelej inteligencie, resp. strojového učenia, a to tým, že jednotlivé produkty obsahujú rôzny počet funkcionalít strojového učenia a navyše sú tieto funkcionality v rôznych fázach vývoja, resp. použiteľnosti a využiteľnosti. Primárne sa jedná o detekciu anomálií na úrovni sietí, užívateľov a koncových zariadení. Takže dá sa povedať, že vyššie spomenuté spoločné vlastnosti sú vlastnosťami, ktoré sú štandardizované a líšia sa len v maličkostiach a na druhej strane oblasť strojového učenia a jej integrácia do SIEM systémov je v aktívnom vývoji a rozvoji. Na záver je potrebné dodať, že sa jednotlivé produkty môžu líšiť v spôsobe ich nasadenia (*cloud*/*on-premise*), ale aj tým pre koho sú primárne určené, či pre malé, stredné alebo veľké podniky, či organizácie.

2.7.2 ELK

ELK Stack [47] alebo tzv. Elastic Stack (názov, ktorý je používaný v súčasnosti) je spojením troch *open-source* projektov - Elasticsearch, Logstash a Kibana. Elastic Stack je podľa Elahi [48] kompletným riešením analýzy záznamov,



Obr. 2.18: Elastic Stack architektúra [48]

ktorý pomáha pri vyhľadávaní, analýze a vizualizácií záznamov generovaných z rôznych zdrojov a v rôznom formáte. Je aj mechanizmom umožňujúcim prehľadávanie všetkých záznamov z jedného miesta a identifikáciu problémov naprieč viacerými systémami tým, že koreluje rôzne záznamy.

Medzi mnohé scenáre na ktoré sa dá Elastic Search použiť patrí podľa Elahi [48]:

- logovanie a analýza záznamov – bezpečnostné analýzy, detekcia podvodov (z angl. *fraud detection*) atď.
- metriky – agregácia údajov na základe hľadaného výrazu (pozri [49])
- full-text vyhľadávanie – vyhľadávač (tzv. *search engine*) určený napr. na vyhľadávanie určitých slov a slovných spojení v záznamoch

Ako je možné vidieť na obr. 2.18 tak Elastic Stack sa skladá z nasledujúcich troch produktov/modulov:

1. Logstash – *open-source* softvér, ktorý centralizuje, resp. získava dáta a záznamy z rôznych zdrojov (webové záznamy z webového servera Apache, NetFlow záznamy, tzv. aplikačné logy napr. z log4j, syslog atď.), následne ich transformuje a ukladá
2. ElasticSearch – distribuovaný, tzv. RESTful vyhľadávací a analytický nástroj, ktorý je základným prvkom Elastic Stack
3. Kibana – slúži na vizualizáciu dát (ako modul pre ElasticSearch)

2.7.3 Spracovanie a analýza záznamov v cloud platformách

Podľa Partlow [50] organizácie využívajú *cloud* platformy ako je AWS alebo Microsoft Azure na tzv. *hosting* svojich webových služieb a tieto platformy sa osvedčili ako efektívne, pretože v niektorých prípadoch dokážu znižovať

náklady, poskytujú lepšiu škálovateľnosť ako *on-premise* riešenia, sú spoľahlivejšie apod.

Log manažment je rovnako dôležitý v *cloud* prostredí ako aj v prostredí dátového centra, ktorý beží *on-premise*, a túto skutočnosť si uvedomujú aj verejný poskytovatelia *cloud* služieb, a preto aj oni poskytujú určité typy služieb log manažmentu. V nasledujúcich podsekciách bude popísané spracovanie a analýza záznamov v jednotlivých *cloud* platformách.

Na záver je potrebné dodať, že SIEM systémy sú potrebné v prípade, keď má daná organizácia vo svojej infraštruktúre (vo svojom prostredí) veľké množstvo systémov, a teda veľké množstvo rôznych zdrojov záznamov atď. Avšak v prípade, že celá infraštruktúra beží na niektorej z *cloud* platforiem (napr. AWS, Microsoft Azure atď.) tak si teoreticky organizácie v niektorých prípadoch (nepotrebujú pokročilú analytiku záznamov apod.) môžu vystačiť s nástrojmi (pozri nižšie), ktoré vo svojom prostredí pre *log* manažment poskytujú jednotliví poskytovatelia týchto *cloud* platforiem, pretože aj tak celá infraštruktúra beží v ich prostredí a všetky záznamy sa nachádzajú na jednom mieste.

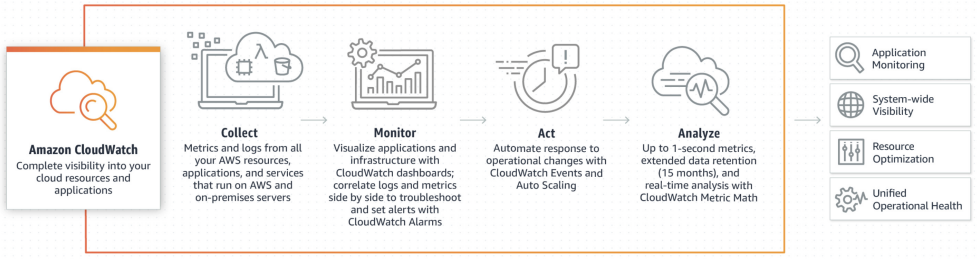
2.7.3.1 AWS

AWS poskytuje užívateľom na spracovanie, analýzu a vizualizáciu záznamov, ale aj na sledovanie aktivít užívateľov dve konkrétne riešenia a to CloudWatch, ktorý poskytuje viditeľnosť (z angl. *visibility*) naprieč *cloud* zdrojmi a aplikáciami, a CloudTrail, ktorý dokáže sledovať aktivitu užívateľov a používanie API v rámci tejto *cloud* služby, resp. platformy.

Amazon CloudWatch [51] je monitorovacia a riadiaca služba umožňujúca:

- pristupovať k dátam z jednej platformy – umožňuje zbierať metriky a záznamy zo všetkých AWS zdrojov, aplikácií a služieb, ktoré bežia na AWS platforme, ale aj na *on-premise* serveroch[31]
- jednoduchšie zhromaždovať vlastné a špecifické metriky z rôznych AWS zdrojov – natívna integrácia s viac ako 70 AWS službami
- získať viditeľnosť naprieč aplikáciami, infraštruktúrou a službami – korelácia a vizualizácia metrík, a záznamov za účelom rýchleho určenia a vyriešenia rôznych problémov
- znižovať celkové náklady – optimalizovať aplikačné a prevádzkové zdroje napr. automatickým vypínaním nepoužívaných zdrojov, monitorovaním zdrojov so sekundovou granularitou atď.
- pracovať so záznamami - umožňuje skúmať, analyzovať a vizualizovať záznamy, čo uľahčuje riešenie rôznych prevádzkových problémov atď.

[31] Servery, ktoré sa nachádzajú v dátovom centre danej organizácie, a teda nie sú umiestnené u niekoho iného, kto by sa staral o ich údržbu a monitorovanie.



Obr. 2.19: Architektúra Amazon CloudWatch [51]

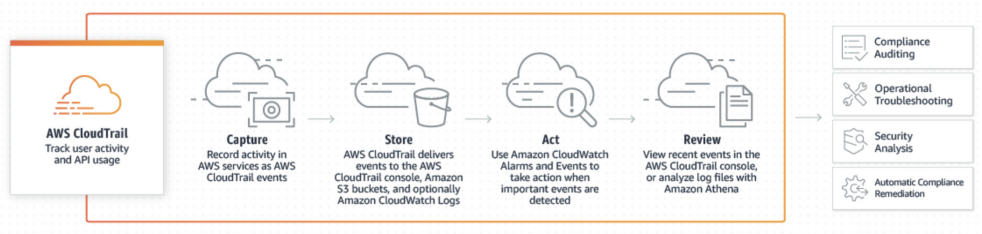
Služba Amazon CloudWatch sa v prostredí AWS *cloud* platformy používa hlavne na monitorovanie a následne na riešenie rôznych problémov v rámci infraštruktúry, optimalizáciu zdrojov, monitorovanie aplikácií, ale aj na zber, analýzu a vizualizáciu záznamov. Na obr. 2.19 je možné vidieť architektúru Amazon CloudWatch služby.

AWS CloudTrail [52] je služba, ktorá poskytuje správu a kontrolu dodržiavania predpisov (z angl. *compliance*), operačný audit a audit rizík AWS účtov. Pomocou tejto služby je možné zaznamenávať a monitorovať akcie (poskytuje históriu udalosti jednotlivých aktivít), ktoré sú spojené s konkrétnym účtom naprieč celou AWS infraštruktúrou.

Medzi hlavné benefity, ktoré táto služba poskytuje patrí:

- dodržiavanie pravidiel (z angl. *compliance*) – zjednodušenie dodržiavania pravidiel, resp. nariadení v oblasti auditu automatickým zaznamenávaním a ukladaním udalosti, resp. akcií, ktoré boli vykonané daným AWS účtom
- bezpečnostná analýza a riešenie problémov – zachytením celej histórie udalosti, resp. akcií, ktoré boli vykonané daným AWS účtom je možné objaviť a vyhľadať rôzne bezpečnostné a prevádzkové problémy
- identifikácia užívateľov, resp. účtov – zaznamenávanie zdroja (užívateľ, resp. účet), zdrojovej IP adresy z ktorej bol príkaz, tzv. API *call* v rámci danej AWS infraštruktúry vykonaný, ale aj čas, kedy došlo k uskutočneniu danej akcie
- bezpečnostná automatizácia – sledovanie a automatické reagovanie na rôzne aktivity účtu, ktoré môžu ohroziť bezpečnosť AWS zdrojov

Služba AWS CloudTrail sa v prostredí AWS *cloud* platformy používa hlavne na podporu tzv. *compliance*, detekciu exfiltrácie dát, bezpečnostnú analýzu, ale aj na riešenie rôznych prevádzkových problémov, ktoré sa môžu vyskytnúť v rámci AWS infraštruktúry. Na obr. 2.20 je možné vidieť architektúru tejto AWS CloudTrail služby.



Obr. 2.20: Architektúra AWS CloudTrail [52]

2.7.3.2 Microsoft Azure

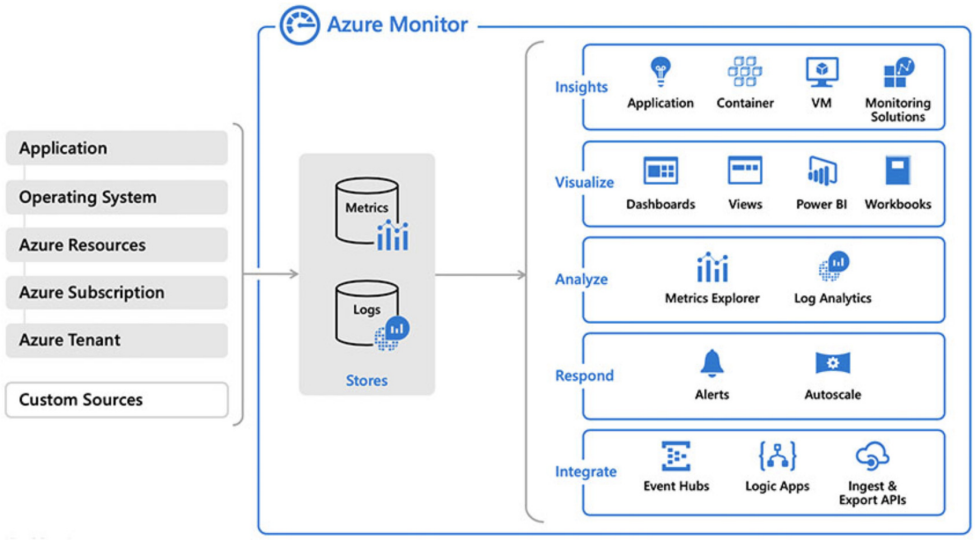
V prostredí *cloud* platformy Microsoft Azure sa používa nástroj Microsoft Azure Monitor [53], ktorý zhromažďuje a analyzuje telemetrické dáta z rôznych zdrojov v rámci Azure *cloud* platformy, ale aj z *on-premise* prostredí. Azure Monitor umožňuje analyzovať dáta, nastavovať rôzne upozornenia, získať kompletný prehľad a zaistiť maximálny výkon a dostupnosť jednotlivých aplikácií, ale aj aktívne identifikovať potenciálne hrozby a problémy s využitím strojového učenia. Pomocou integrovaného vyhľadávania a rôznych náhľadov je možné analyzovať milióny záznamov naprieč celou infraštruktúrou. Medzi hlavné funkcie tohto nástroja patrí podľa Microsoft [53]:

- unifikácia – možnosť ukladania a analýzy telemetrických dát z jedného centralizovaného miesta
- inteligencia – odhalenie rôznych hrozieb pomocou pokročilého analytického modulu, interaktívneho vyhľadávacieho jazyka a strojového učenia
- otvorenosť – integrácia s nástrojmi pre správu problémov, IT služieb, udalostí a informácií o zabezpečení

Na obr. 2.21 je možné vidieť architektúru Microsoft Azure Monitor nástroja.

2.8 Využitie strojového učenia k analýze záznamov a identifikácií podozrivej aktivity

Podľa Unomaly [54] patrí pravidelná analýza záznamov k najúčinnejšiemu spôsobu ako identifikovať rôzne chyby v systémoch (konfiguračné chyby, či softvérové chyby atď.). Taktiež pomocou tejto analýzy záznamov je možné identifikovať napr. rôzne zneužitia systémov a podozrivé aktivity ešte predtým ako spôsobia bezpečnostný incident alebo napr. degradáciu nejakej služby. Obsahy väčšiny záznamov sa častokrát opakujú a vyzerajú „normálne“, preto opakované prezeranie a pozeranie sa na prvý pohľad na tie isté záznamy,



Obr. 2.21: Architektúra Azure Monitor [53]

resp. ich obsahy je aj podľa Taylor [55] strata času a plytvanie zdrojmi, a práve preto môže byť efektívnejším spôsobom analýzy využitie strojového učenia, ktoré dokáže identifikovať rôzne anomálie a podozrivé aktivity. Vytváraním rôznych aplikácií, ktoré používajú algoritmy strojového učenia s ohľadom na počítačovú bezpečnosť má podľa Endler [56] znížiť únavné a časovo náročné úlohy, ktoré musia robiť napr. bezpečnostní analytici pri analýze záznamov.

Naučiť sa „čítať“, resp. porozumieť záznamom je podľa Berman [23] zručnosť, ktorá môže zabrať veľa času a hlavne v prípade, keď je potrebné sa vyznať v rôznych formátoch v ktorých môžu byť jednotlivé záznamy uložené. Rozmanitosť nástrojov, ktoré so záznamami pracujú taktiež tento problém nezjednodušuje. Tieto výzvy môžu častokrát odradiť bezpečnostných analytikov pri rôznych analýzach a znižovať hodnotu záznamov. Strojové učenie môže podľa Taylor [55] uľahčiť identifikáciu podozrivej aktivity a zefektívniť analýzu záznamov pomocou:

- klasifikácie dát, resp. záznamov pomocou techník a metód *supervised learning* – vstupom je záznam a výstupom rozhodnutie, či ide o „typický“ záznam alebo o anomáliu, ktorú je potrebné ďalej analyzovať
- zoskupením veľkého množstva neštruktúrovaných dát do tzv. „zmysluplných/interpretovateľných“ skupín – všetko ostatné, čo netvorí skupinu, resp. „padne“ mimo skupiny je považované za podozrivé
- predpovedaním možných dôsledkov rôznych útokov alebo incidentov

Medzi niektoré konkrétne implementácie a aplikácie algoritmov strojového učenia na analýzu záznamov patrí:

1. predikcia útokov – podľa Jain, Patnaik, Ichalkaranje [57] je väčšina útokov identifikovaná, resp. detegovaná až po tom čo napácha nejaké škody (idea predikcie útokov je postavená na tom, že tzv. trénovacie dáta budú generované na základe udalosti, ktoré sa už udiali (vytvorené analýzou siete a zachytávaním sieťovej prevádzky) a dáta v reálnom čase budú používané na predikciu útokov ešte predtým ako sa stanú) (pozri [57] str. 231) použitie SVM ako algoritmu strojového učenia na klasifikáciu alebo napr. použitie tzv. kontextového klasifikátora udalosti založeného na algoritmoch umelých neurónových sietí (ANN) (pozri Suarez-Tangil, Palomar, Ribagorda, a kol. [58])

2. detekcia útokov/zneužitia (z angl. *misuse detection*), škodlivého správania a hrozieb – napr. pomocou algoritmov neurónových sietí aplikovaných na auditné záznamy operačného systému Solaris (pozri Endler [56]), využitím logistickej regresie strojového učenia aplikovanej na auditné záznamy operačného systému Windows (pozri Berlin, Slater, Saxe [59]) alebo podľa Mayhew, Atighetchi, Adler a kol. [60] použitie strojového učenia na detekciu hrozieb pri analýze objemných dátových zdrojov

3. detekcia anomálií – napr. detekcia anomálií v sietí Malaiya, Kwon, Kim a kol. [61] a Bilge, Balzarotti, Robertson a kol. [62], pomocou záznamov z aplikačných systémov Kuna, García-Martinez a Villatoro [63] alebo analýza systémových záznamov za účelom detekcie anomálií pomocou troch *supervised* a troch *unsupervised* typov strojového učenia S. He, Zhu, P. He a kol. [64] alebo pomocou *deep learning* Du, Li, Zheng a kol. [65], a Tuor, Kaplan, Hutchinson a kol. [66]

Ako už bolo spomínané v podsekcií 2.7.1 (popis typickej architektúry SIEM systémov), že v prípade, ak niektoré z pravidiel (ktoré sú pravidelné aplikované na prichádzajúce udalosti) vytvorí upozornenie/výstrahu tak, bezpečnostní analytici zo SOC následne túto výstrahu preskúmajú a rozhodnú, či šlo v skutočnosti o škodlivú aktivitu, resp. hrozbu alebo nie. Identifikácia podozrivej aktivity pomocou manuálneho vytvorenia jednoduchých, ale aj komplexných pravidiel v rôznych SIEM systémoch je obtiažná, pretože bezpečnostný analytik častokrát nevidí súvislosti medzi jednotlivými udalosťami, chýba mu kontext a niekedy aj dostatočná znalosť daného prostredia, nemusí vytvoriť konečný výčet jednotlivých pravidiel a hrozieb, čím dôjde k tomu, že niektoré hrozby a útoky nemusia byť zachytené, navyše je manuálne vytváranie pravidiel časovo náročné. To má za následok podľa Feng, Wu a Liu [67], ale aj podľa Bhatt, Manadhata a Zomlot [20] (pozri 2.7.1), že výstrah je obrovské množstvo (čo presahuje schopností bezpečnostných analytikov riešiť

tieto všetky výstrahy) a navyše väčšina z týchto výstrah sú tzv. falošné poplachy. Aj vďaka tejto skutočnosti môže dôjsť k tomu, že niektoré potenciálne hrozby nebudú zachytené, prípadne dôjde ku kompromitácií systémov.

Podľa Feng, Wu a Liu [67] je strojové učenie spôsob, ktorým je možné zredukovať počet falošných výstrah/upozornení a zlepšiť produktivitu bezpečnostných analytikov SOC tým, že systémy, resp. algoritmy strojového učenia generujú komplexné tzv. užívateľské risk skóre[32], pomocou ktorého je možné lepšie detegovať správanie sa užívateľov. Komplexné užívateľské risk skóre sa generuje na základe rôznych informácií, napr. na základe počtu výstrah, ktoré boli generované daným užívateľom za jeden deň, výsledkov analýz tvorených SOC tímami atď. Na základe tohto skóre dokážu bezpečnostní analytici prioritizovať jednotlivé investigácie tým, že začnú s vyšetrovaním výstrah, ktoré majú najvyššie skóre, čo môže zvýšiť efektivitu ich práce, optimalizovať postupnosť riešenia jednotlivých výstrah a v neposlednom rade zlepšiť bezpečnosť celej organizácie.

Niektoré z vyššie spomenutých konkrétnych implementácií a aplikácií algoritmov strojového učenia pre analýzu záznamov sú veci, ktoré už niektoré SIEM systémy v určitej forme ponúkajú (avšak napr. v prípade spomínaného užívateľského risk skóre nemusia používať strojové učenie, ale „obyčajnú“ štatistiku atď.). Každopádne platí, že umelá inteligencia/strojové učenie je v oblasti analýzy záznamov smer, ktorým sa všetci uberajú. Rivas [68] z GB Advisors tvrdí, že spojenie umelej inteligencie, strojového učenia a rôznych SIEM riešení dokáže zvýšiť efektivitu bezpečnostných tímov prostredníctvom detekcie zraniteľností, útokov a rôznych hrozieb, ale hlavne tzv. predikciou neznámych hrozieb s minimálnou intervenciou bezpečnostných analytikov. Toto spojenie môže podľa Rivas [68] pomôcť pri:

- určovaní vzťahu medzi rôznymi anomáliami
- zbieraní, spracovávaní a analýze veľkého množstva dát, resp. záznamov
- optimalizácií tzv. UEBA modulu, ktorý je určený na detekciu tzv. nepravidelných vzorov (z angl. *patterns*) v správaní jednotlivých užívateľov
- prevode reaktívneho prístupu analýzy záznamov k proaktívnemu prístupu
- redukcií falošných výstrah

V oblasti počítačovej bezpečnosti, konkrétne pri aplikácií algoritmov strojového učenia v oblasti analýzy záznamov je častokrát podľa Scarfone [69], ale aj podľa Chio a Freeman [2] ťažké rozlíšiť, čo je a čo nie je škodlivé, pretože

[32] Ide o „inteligentné“ zoskupovanie rôznych upozornení, ktoré súvisia s aktivitou daného používateľa počas určitého časového obdobia.

to častokrát závisí na konkrétnom prostredí, v ktorom sa dané algoritmy strojového učenia aplikujú, ale aj na konkrétnom kontexte, ktorý je veľmi dôležitý a nevyhnutný na odlíšenie (napr. či séria prihlásení sa na server je spôsobená systémovým administrátorom, ktorý sa snaží pracovať prostredníctvom vzdialeného prístupu alebo ide o exfiltráciu dát vykonávanú útočníkom atď.).

So všetkým potenciálom, ktoré so sebou strojové učenie prináša nemôžeme zabúdať na to, že väčšina funkcií, ktoré so sebou strojové učenie prináša stále vyžaduje interakciu človeka, resp. bezpečnostných analytikov. Podľa Reichenberg [70] strojové učenie nemá nahradzovať bezpečnostných analytikov, ale malo by im slúžiť ako nástroj, ktorý im pomôže robiť rýchlejšie a lepšie rozhodnutia, byť viac proaktívnejší, ale aj pomáhať pri automatizácií jednotlivých úloh a detekcií incidentov, ktoré by bez strojového učenia neboli možné.

2.9 Zhrnutie

Záznam je zoznam udalostí, ktoré sa stali v rámci systémov a sietí danej organizácie, resp. jej infraštruktúry. Zdrojom záznamov sú bezpečnostné softvéry (antimalvér softvér, autentifikačné servery atď.), operačné systémy (systémové udalostí a auditné záznamy) a aplikácie (informácie o používaní daných aplikácií, dôležité systémové udalosti aplikácií atď.). Existujú rôzne štandardné, ale aj proprietárne formáty záznamov, príkladom je napr. syslog, rôzne formáty webových serverov, XML, JSON atď. Na druhej strane existujú aj tzv. logovacie úrovne, ktoré sú obsahom jednotlivých záznamov a ktoré sú určené na kategorizáciu jednotlivých záznamov podľa závažnosti, a to už či z pohľadu prevádzky daného systému a/alebo z pohľadu bezpečnosti.

Pre *log* manažment je kľúčové:

- správne nastavené časové značky na všetkých zdrojoch (synchronizovaný čas a jeho jednotný formát)
- normalizácia a syntaktická analýza záznamov
- dostatočná kapacita úložiska na ukladanie záznamov, pravidelné posielanie záznamov do centrálneho zariadenia a ich uchovávanie po určitú dobu
- dostupnosť záznamov aj v prípade poruchy systému (zálohovanie)
- korelácia, vizualizácia a pravidelná analýza záznamov
- zaistenie bezpečnosti a integrity záznamov (ochrana pred zneužitím, zmenou alebo odstránením)

O zber záznamov z rôznych zdrojov naprieč IT prostredím danej organizácie a ich preposielaním do ďalších systémov, ktoré dokážu tieto záznamy

spracovať, analyzovať, vizualizovať sa starajú systémy určené na zber záznamov (z angl. *log collectors*), konkrétne ide napr. o syslog-ng a fluentd, ale existujú aj ďalšie iné systémy.

SIEM systémy sú dôležitým nástrojom, resp. veľmi dôležitou súčasťou tzv. SOC tímov, pretože dokážu zbierať, normalizovať a analyzovať rôzne bezpečnostné udalosti, ktoré sú generované z rôznych zdrojov a dokážu pomôcť CSIRT tímom, ktorí sú zodpovedný za prijímanie, posudzovanie a reagovanie na rôzne aktivity a incidenty, ktoré sa týkajú počítačovej bezpečnosti. Medzi hlavné funkcie SIEM systémov patrí napr. agregácia dát z rôznych zdrojov, korelácia podobných udalosti, automatická analýza záznamov, ale aj rôzne vizualizácie týchto záznamov. Na druhej strane musia SIEM systémy riešiť výzvy, ktoré súvisia s vytváraním pravidiel (ich vytváranie môže byť niekedy časovo náročné) a ich používaním/aplikovaním (pravidlá môžu vytvárať falošné upozornenia/výstrahy, ale aj to, že nemusia byť dostatočné komplexné), nedostatkom tzv. kontextových informácií atď. V *cloud* platformách ako je napr. AWS alebo Microsoft Azure je *log* manažment rovnako dôležitý ako v prostredí dátového centra, ktorý beží *on-premise* a túto skutočnosť si uvedomujú aj poskytovatelia verejných *cloud* služieb, ktorí poskytujú systémy, resp. služby na spracovanie a analýzu záznamov vo svojich prostrediach.

Strojové učenie by malo pomôcť pri analýze záznamov a identifikácií podozrivých aktivít bezpečnostným analytikom tým, že dokáže predikovať a detegovať známe, ale aj neznáme útoky a hrozby, proaktívne blokovať podozrivé aktivity (takže analytici sa môžu viac venovať investigácií a tzv. *incident response*), detegovať rôzne anomálie, ale aj redukovať falošné výstrahy a zlepšiť tým produktivitu bezpečnostných analytikov. Na druhej strane by strojové učenie nemalo nahradzovať bezpečnostných analytikov, ale malo by im slúžiť ako nástroj, ktorý im pomôže s identifikáciou podozrivých aktivít, ale aj umožní robiť rýchlejšie a lepšie rozhodnutia, a tým im zjednoduší a/alebo automatizuje ich prácu.

Kapitola 3

GDPR

Na začiatku tejto kapitoly sa budeme venovať krátkemu úvodu do problematiky GDPR, vzťahu GDPR, strojového učenia a osobných údajov. V ďalších častiach tejto kapitoly sa budeme venovať vzťahu GDPR a bezpečnostných auditných záznamov, GDPR a SIEM systémov a v neposlednom rade kritike GDPR. V poslednej sekcií zhrnieme obsah celej tejto kapitoly.

3.1 Základné informácie

Nariadenie Európskeho parlamentu a Rady (EÚ) 2016/679 z 27. apríla 2016 o ochrane fyzických osôb pri spracúvaní osobných údajov a o voľnom pohybe takýchto údajov, ktorým sa ruší smernica 95/46/ES (všeobecné nariadenie o ochrane údajov), ktoré je známe pod skratkou GDPR predstavuje podľa Úřadu pro ochranu osobních údajů (ÚOOÚ) [71]:

„*predstavuje právny rámec ochrany osobných údajov platný na celom území EÚ, ktorý háji práva ich občanov proti neoprávnenému zaobchádzaniu s ich dátami a osobnými údajmi. GDPR preberá všetky doterajšie zásady ochrany a spracovania údajov, na ktorých systém únie ochrany osobných údajov stojí a potvrdzuje, že ochrana cestuje cez hranice súčasne s osobnými údajmi. V súlade s tým ďalej obecné nariadenie rozvíja a posiluje práva ľudí, ktorí sú dotknutý spracovaním, a to v oboch zložkách: mať, resp. získavať) informácie o tom, ktoré ich údaje sú spracovávané a prečo, a domáhať sa dodržiavania pravidiel, vrátane nápravy stavu. GDPR kladie systematický dôraz na vymáhateľnosť práv ľudí a povinností správcov (zodpovedných za spracovanie). Obsahuje preto prepracovanejšie a náročnejšie pravidlá pre zvláštne kategórie údajov a spracovania, a súčasne vymáha od správcov a spracovávateľov výrazne aktívnejší prístup, najmä sa jedná o to, že pred zahájením nového spracovania je potrebné posúdiť vplyv jednotlivých spracovaní na ochranu osobných údajov (DPIA) a zvoliť vhodné nástroje ochrany údajov, za určitých podmienok si vyžiadať predbežnú konzultáciu u dozorného úradu. Kľúčom k nastavovaniu povinností pro správcov je rizikovosť, ktorá je odvodená z rozsahu spracovania,*

spracovávaných osobných údajov a používaných technológií. Správcovia a spracovávatelia sú za určitých podmienok povinný menovať poverenca pre ochranu osobných údajov. Podrobnejšie sú stanovené povinnosti pri zabezpečení spracovania a po novom je zavedená povinnosť ohlasovať prípady porušenia zabezpečenia osobných údajov dozornému úradu a občanom, ktorých sa porušenie zabezpečenia týka.“ [33]

GDPR je v celej EÚ jednotne účinné a vymáhateľné od 25. mája 2018 a týka sa všetkých organizácií, ktoré spracovávajú osobné údaje občanov EÚ. GDPR tak nahradzuje a predovšetkým zjednocuje zákony o ochrane osobných údajov všetkých členských zemí EÚ.

Osobné údaje, ktorých sa primárne GDPR týka sú v tomto nariadení (článok 4 na str. 33 pozri [72]) definované ako:

„*Akékoľvek informácie týkajúce sa identifikovanej alebo identifikovateľnej fyzickej osoby (ďalej len „dotknutá osoba“); identifikovateľná fyzická osoba je osoba, ktorú možno identifikovať priamo alebo nepriamo, najmä odkazom na identifikátor, ako je meno, identifikačné číslo, lokalizačné údaje, online identifikátor, alebo odkazom na jeden či viaceré prvky, ktoré sú špecifické pre fyzickú, fyziologickú, genetickú, mentálnu, ekonomickú, kultúrnu alebo sociálnu identitu tejto fyzickej osoby.*“

Recitál č. 30 na str. 6 (pozri [72]) hovorí o online identifikátoroch:

„*Fyzickým osobám môžu byť pridelené online identifikátory, ktoré poskytujú ich prístroje, aplikácie, nástroje a protokoly, ako napríklad IP adresa, cookies, alebo iné identifikátory, ako napríklad štítky na rádiofrekvenčnú identifikáciu. Tieto môžu zanechávať stopy, ktoré sa najmä v kombinácii s jedinečnými identifikátormi a inými informáciami získanými zo serverov môžu použiť na vytvorenie profilov fyzických osôb a na ich identifikáciu.*“

V kontexte IP adries je potrebné pripomenúť rozhodnutie Súdneho dvora EU vo veci C-582/14 Patrick Breyer v. Bundesrepublik Deutschland (pozri [73]), ktoré poukazuje na problematiku dynamických IP adries, ktoré v spojení s ďalšími údajmi môžu predstavovať osobné údaje.

GDPR článok 9 na str. 38 (pozri [72]) definuje osobitnú kategóriu osobných údajov (citlivé osobné údaje) ako:

„*osobné údaje, ktoré odhaľujú rasový alebo etnický pôvod, politické názory, náboženské alebo filozofické presvedčenie alebo členstvo v odborových organizáciách, a spracúvanie genetických údajov, biometrických údajov na individuálnu identifikáciu fyzickej osoby, údajov týkajúcich sa zdravia alebo údajov týkajúcich sa sexuálneho života alebo sexuálnej orientácie fyzickej osoby.*“

ICO[34] [74] tvrdí, že tzv. pseudonymizované údaje sú stále považované za osobné údaje, aj napriek tomu, že dokážu sťažiť identifikáciu jednotlivcov, čím

[33] Preložené autorom tejto diplomovej práce zo zdroja [71].

[34] Nezávislý orgán Spojeného kráľovstva zriadený na podporu dodržiavania informačných práv vo verejnom záujme, podporuje otvorenosť verejných orgánov a súkromie jednotlivcov.

znižujú riziká spojené s ochranou súkromia. Na druhej strane údaje, ktoré sa podľa ICO dajú skutočne anonymizovať nepodliehajú nariadeniu GDPR.

V niektorých prípadoch je ťažké určiť, či sú dané údaje považované za osobné údaje, preto je podľa ICO potrebné s informáciami zachádzať opatrne, uistiť sa, že existuje jasný dôvod na ich spracovanie, ale najmä zabezpečiť ich bezpečné ukladanie a zlikvidovanie. ÚOOÚ, ktorý plní funkciu kontroly, dozorného a konzultačného úradu, a informačného kanálu (v Českej republike) vytvoril základný jednoduchý návod, resp. pravidlá ochrany osobných údajov, ktoré sú využiteľné malými správcami údajov, živnostníkmi, či menšími podnikmi. Obsahom návodu je napr. to, že spracovanie údajov nesmie nadmerne zasahovať do súkromia, účel spracovaných údajov musí byť jasný, spracovanie údajov musí byť legitímne a nesmie byť v rozpore s právnymi predpismi, či morálkou atď. Tento návod, ktorý obsahuje desať bodov je možné nájsť v Desateru spracovaní pro správce [75], publikované ÚOOÚ.

Prejsť a rozobrať všetky body GDPR nie je cieľom tejto diplomovej práce, a preto po krátkom úvode do problematiky GDPR sa budeme v nasledujúcich sekciách venovať vzťahu GDPR, osobných údajov a strojového učenia, vzťahu GDPR a auditných záznamov, ale aj vzťahu GDPR a SIEM systémov, a kritike GDPR. Viac informácií o GDPR je možné nájsť buď priamo v nariadení (pozri [72]), na webovej stránke ÚOOÚ [76], kde je mimo iného možné požiadať aj o konzultáciu a poradenstvo. Rôzne pokyny, odporúčania a osvedčené postupy v oblasti GDPR poskytuje a prináša Európsky výbor pre ochranu osobných údajov (nástupca Pracovnej skupiny pre ochranu údajov, známej pod skratkou WP29) (pozri [77]), ktorý mimo iného prispieva ku konzistentnému uplatňovaniu pravidiel ochrany osobných údajov v celej EÚ.

3.2 GDPR a strojové učenie

V tejto sekcií sa budeme venovať vzťahu strojového učenia a GDPR v zmysle ako strojové učenie pristupuje a pracuje s osobnými údajmi, ktoré sú pod ochranou GDPR.

Podľa Európského výboru pre ochranu osobných údajov [78] sa tzv. profilovanie a automatizované rozhodnutia používajú, resp. aplikujú na rôzne odvetvia ako je napr. bankový a finančný sektor, oblasť zdravotnej starostlivosti, marketingu atď. Pokroky a možnosti umelej inteligencie a strojového učenia uľahčili vytváranie profilov a automatizovaných rozhodnutí, čím môže potenciálne dôjsť k významnému ovplyvňovaniu práv a slobôd jednotlivcov.

Analýzu dopadu strojového učenia na tzv. profilovanie jednotlivcov[35] v kontexte GDPR je možné nájsť v článku [79], ktorého autormi sú Kamarinou, Millard a Singh, a práve v tejto sekcií budeme primárne vychádzať z tohto zdroja,

[35]Používanie osobných údajov na vyhodnotenie, resp. analýzu a predikciu zdravia, ekonomickej situácie atď.

ale aj z dokumentu Pokyny k automatizovanému individuálnímu rozhodování a profilování pro účely nařízení 2016/679 (pozri [78]).

3.2.1 Vymedzenie pojmov

Podľa GDPR článku 4 na str. 33 (pozri [72]) je spracúvanie:

„*operácia alebo súbor operácií s osobnými údajmi alebo súbormi osobných údajov, napríklad získavanie, zaznamenávanie, usporadúvanie, štruktúrovanie, uchovávanie, prepracúvanie alebo zmena, vyhľadávanie, prehliadanie, využívanie, poskytovanie prenosom, šírením alebo poskytovanie iným spôsobom, preskupovanie alebo kombinovanie, obmedzenie, vymazanie alebo likvidácia, bez ohľadu na to, či sa vykonávajú automatizovanými alebo neautomatizovanými prostriedkami.*“

GDPR článok 6 na str. 36 (pozri [72]) hovorí o tom, že spracúvanie je zákonné iba vtedy a iba v tom rozsahu, keď je splnená aspoň jedna zo šiestich podmienok definovaných v tomto GDPR článku:

(a) dotknutá osoba vyjadrila súhlas so spracúvaním svojich osobných údajov

(b) spracúvanie je nevyhnutné na plnenie zmluvy

(c) spracúvanie je nevyhnutné k splneniu zákonnej povinnosti prevádzkovateľa

(d) spracúvanie je nevyhnutné, aby sa ochránili životne dôležité záujmy dotknutej osoby alebo inej fyzickej osoby

(e) spracúvanie je nevyhnutné na splnenie úlohy realizovanej vo verejnom záujme alebo pri výkone verejnej moci zverenej prevádzkovateľovi

(f) spracúvanie je nevyhnutné na účely oprávnených záujmov, ktoré sleduje prevádzkovateľ alebo tretia strana

Jednou z podmienok (konkrétne bod f) je, že spracúvanie je nevyhnutné na účely oprávnených záujmov, ktoré sleduje prevádzkovateľ alebo tretia strana. Túto podmienku je možné uplatniť napr. v prípade, že zaznamenávame osobné údaje (napr. online identifikátory ako je napr. IP adresa atď.) do záznamov, resp. auditných záznamov v rôznych systémoch s cieľom predísť, resp. lepšie identifikovať rôzne kybernetické útoky ako je napr. neoprávnený prístup, exfiltrácia dát apod.

Profilovanie je podkategóriou automatizovaného spracúvania a GDPR článok 4 na str. 33 (pozri [72]) definuje profilovanie ako:

„*akákoľvek forma automatizovaného spracúvania osobných údajov, ktoré pozostáva z použitia týchto osobných údajov na vyhodnotenie určitých osobných aspektov týkajúcich sa fyzickej osoby, predovšetkým analýzy alebo predvídania*

aspektov dotknutej fyzickej osoby súvisiacich s výkonnosťou v práci, majetkovými pomermi, zdravím, osobnými preferenciami, záujmami, spoľahlivosťou, správaním, polohou alebo pohybom.“

Profilovanie znamená zhromažďovanie informácií o jednotlivcovi (alebo skupine jednotlivcov) a následné vyhodnotenie charakteristík (napr. vzorov správania) za účelom zaradenia ho/ich do určitej kategórie alebo skupiny na základe ktorej je možné analyzovať alebo predpovedať napr. schopnosť vykonávať jednotlivé úlohy, záujmy, správanie atď.

GDPR článok 22 na str. 46 (pozri [72]) hovorí o automatizovanom individuálnom rozhodovaní, vrátanie profilovania:

„*Dotknutá osoba má právo na to, aby sa na ňu nevzťahovalo rozhodnutie, ktoré je založené výlučne na automatizovanom spracúvaní, vrátane profilovania, a ktoré má právne účinky, ktoré sa jej týkajú alebo ju podobne významne ovplyvňujú.*“

To sa, ale neuplatňuje ak je rozhodnutie:

(a) nevyhnutné na uzavretie alebo plnenie zmluvy medzi dotknutou osobou a prevádzkovateľom

(b) povolené právom Únie alebo právom členského štátu, ktorému prevádzkovateľ podlieha a ktorým sa zároveň stanovujú aj vhodné opatrenia zaručujúce ochranu práv a slobôd a oprávnených záujmov dotknutej osoby

(c) založené na výslovnom súhlase dotknutej osoby

Pokyny k automatizovanému individuálnímu rozhodování a profilování pro účely nařízení 2016/679 [78] na strane 7 tvrdia, že tzv. profilovanie nie je len o samotnom rozhodnutí, ale aj o zhromažďovaní údajov pre potreby profilovania a vytváraní profilov ako takých, a o automatizovanej analýze pre určenie vzájomných súvislostí.

Podľa Kamarinou, Millard a Singh [79] je vhodné rozdeliť celý proces profilovania na tri fázy:

1. zber dát – osobné údaje by sa mali zhromažďovať iba na konkrétne, explicitné a legitímne účely a navyše dotknutá osoba má právo namietať proti spracúvaniu údajov (GDPR článok 21 na str. 45 (pozri [72]))

2. vývoj daného modelu pomocou algoritmov strojového učenia

3. samotné rozhodovanie

Podľa Kamarinou, Millard a Singh [79] sa ochrana, ktorá súvisí s rozhodovaním na základe automatického profilovania nemusí uplatniť v prípade, že sú osobné údaje anonymizované, a teda, že nie je možné identifikovať jednotlivcov v procese profilovania.

3.2.2 Rozhodovací proces

Na to, aby nebol aplikovaný GDPR článok 22 na str. 46 (pozri [72]), ktorý hovorí automatizovanom individuálnom rozhodovaní vrátane profilovania, tak by musela byť uplatnená jedna z troch výnimiek spomínaných v podsekcií 3.2.1 alebo by podľa Kamarinou, Millard a Singh [79] musel byť ľudský zásah v danom procese vecný, tzn. že ľudia by museli mať „skutočný“ vplyv na výsledok rozhodovacieho procesu. Napríklad v prípade, že daný rozhodovací proces závisí len na výstupe nejakého algoritmu strojového učenia a človek len potvrdí tento výstup algoritmu strojového učenia bez nejakej analýzy (výstup nie je žiadnym kritickým spôsobom posudzovaný človekom) tak táto vykonaná akcia nebude predstavovať „skutočný“ vplyv na výsledok rozhodovacieho procesu. GDPR článok 22 na str. 46 (pozri [72]), ale na druhej strane nešpecifikuje, či rozhodnutie proti ktorému sú dotknuté osoby chránené musí byť práve to konečné rozhodnutie alebo jednotlivé kroky vykonávané počas automatizovaného spracúvania, navyše GDPR nešpecifikuje, či samotné rozhodnutie musí byť urobené človekom, alebo či ho môže urobiť stroj. Na druhej strane recitál č. 71 na str. 14 (pozri [72]) tvrdí, že dotknutá osoba by na rozhodnutie ktoré bolo spracúvaním dosiahnuté mala mimo iného dostať vysvetlenie tohto rozhodnutia. Čo môže byť v prípade algoritmov strojového učenia niekedy problematické, pretože jednotlivé algoritmy strojového učenia môžu byť založené na veľmi odlišných modeloch. Napr. v prípade algoritmu rozhodovacích stromov (z angl. *decision tree*) môže byť jednoduchšie argumentovať a vysvetliť ako sa algoritmus rozhodol, a prečo k danému záveru prišiel (samozrejme to potom závisí aj na veľkosti a komplexnosti daného rozhodovacieho stromu). Na druhej strane niektoré algoritmy môžu pracovať ako tzv. *black box* ako to napr. platí u *deep learning* modelov, pretože ich závery a výsledky nie je možné podľa Kamarinou, Millard a Singh [79] legitimovať deduktívnym spôsobom.

3.2.3 Posúdenie vplyvu na ochranu osobných údajov - DPIA

Podľa GDPR článku 35(1) na str. 53 (pozri [72]):

„*Ak typ spracúvania, najmä s využitím nových technológií a s ohľadom na povahu, rozsah, kontext a účely spracúvania pravdepodobne povedie k vysokému riziku pre práva a slobody fyzických osôb, prevádzkovateľ pred spracúvaním vykoná posúdenie vplyvu plánovaných spracovateľských operácií na ochranu osobných údajov. Pre súbor podobných spracovateľských operácií, ktoré predstavujú podobné vysoké riziká, môže byť dostatočné jedno posúdenie.*“

Ako je vyššie spomenuté tak toto posúdenie je nutné v prípadoch, ktoré môžu mať za následok vysoké riziko pre práva a slobody fyzických osôb a následne by malo slúžiť predovšetkým na zmiernenie, či elimináciu rizík, ale aj ako dokumentácia pre dozorný úrad.

Posúdenie vplyvu na ochranu osobných údajov, ktoré je uvedené vyššie sa vyžaduje najmä v prípadoch:

(a) systematického a rozsiahleho hodnotenia osobných aspektov týkajúcich sa fyzických osôb, ktoré je založené na automatizovanom spracúvaní vrátane profilovania a z ktorého vychádzajú rozhodnutia s právnymi účinkami týkajúcimi sa fyzickej osoby alebo s podobne závažným vplyvom na ňu

(b) spracúvania vo veľkom rozsahu osobitných kategórií údajov (citlivé osobné údaje)

(c) systematického monitorovania verejne prístupných miest vo veľkom rozsahu

ÚOOÚ vydal dokument k povinnosti vykonávať posúdeniu vplyvu na ochranu osobných údajov (DPIA) (pozri [80]), ktorého obsahom je hlavne zoznam druhov spracúvania, resp. stanovenie kritérií (určenie rizikovosti spracúvania (operácií spracúvania) osobných údajov), podľa ktorých je nutné sa zaoberať tzv. DPIA. Celkovo sa v tomto dokumente nachádza 15 kritérií a jednotlivé kritéria môžu nadobúdať tzv. kritické hodnoty (červené), významné hodnoty (žlté) alebo nízke hodnoty (zelené). Na to, aby správca spadal do kategórie s vysokým rizikom musia byť hodnoty minimálne dvoch kritérií kritické (červené) alebo jedna hodnota kritéria kritická (červená) a zároveň minimálne 5 hodnôt kritérií musí nadobúdať významnú (žltú) hodnotu.

Spôsob akým sa strojové učenie vyvíja a používa môže viesť k vyvolaniu požiadavku na vykonanie tzv. DPIA. Je preto potrebné zvážiť konkrétne fakty rôznych scenárov strojového učenia na určenie toho, či sa DPIA vyžaduje alebo nie [79] a vykonať tzv. analýzu kritérií podľa dokumentu, ktorý je spomenutý vyššie. Pri analýze kritérií by sme v tomto prípade určite narazili na dôležité kritéria ako sú napr. sústavnosť spracúvania osobných údajov, rozsah spracovania osobných údajov, údaje zhromažďované o subjektoch údajov atď.

Podľa GDPR článku 35(7) na str. 54 (pozri [72]) by samotné posúdenie malo obsahovať:

(a) systematický opis plánovaných spracovateľských operácií a účely spracúvania, vrátane prípadného oprávneného záujmu, ktorý sleduje prevádzkovateľ

(b) posúdenie nutnosti a primeranosti spracovateľských operácií vo vzťahu k účelu

(c) posúdenie rizika pre práva a slobody dotknutých osôb

(d) opatrenia na riešenie rizík vrátane záruk, bezpečnostných opatrení a mechanizmov na zabezpečenie ochrany osobných údajov a na preukázanie

súladu s týmto nariadením, pričom sa zohľadnia práva a oprávnené záujmy dotknutých osôb a ďalších osôb, ktorých sa to týka

Na druhej strane v prípade oprávneného záujmu, kedy sa napr. zaznamenávajú IP adresy do logovacích súborov kvôli bezpečnosti nie je vypracovanie posúdenia nutné.

V prípade, že správca, či spracovávateľ spadá do kategórie, kedy musí vypracovať DPIA mal by mať jednotlivé rizikové procesy zdokumentované, a to konkrétne ako a prečo s osobnými údajmi zachádza, kto má k ním prístup a ako sú zabezpečené.

3.3 GDPR a bezpečnostné auditné záznamy

Keďže záznamy, resp. bezpečnostné auditné záznamy obsahujú rôzne informácie o činnostiach na danom systéme a ich obsahom môžu byť aj osobné údaje je preto potrebné sa zaoberať vzťahu GDPR a bezpečnostných auditných záznamov. Napr. v prípade webových serverov môžu byť podľa Bateman [81] záznamy určené na identifikáciu:

- osoby, ktorá navštívila danú webovú stránku
- polohy (lokalizácia) návštevníka danej webovej stránky
- aktivity návštevníka na danej webovej stránke

Súčasťou vyššie spomenutých záznamov sú častokrát aj osobné údaje a navyše jednotlivé záznamy môžu obsahovať IP adresy, ktoré ako už bolo vyššie spomenuté (pozri sekciu 3.1) sú považované za online identifikátory a v kombinácií s jedinečnými identifikátormi a inými informáciami získanými z iných systémov môžu byť použité na vytvorenie tzv. profilov fyzických osôb a na ich identifikáciu. Ak by sme, ale na druhej strane všetky osobné údaje zo záznamov odstránili naše záznamy by významne stratili na hodnote a v niektorých prípadoch by sa dokonca mohli stať nepoužiteľnými. Našťastie, podľa Bateman [81] GDPR umožňuje legitímne používať a spracúvať osobné údaje (ktoré môžu byť obsahom záznamov, resp. bezpečnostných auditných záznamov), v prípade, že sú dodržiavané zásady ochrany a spracúvania osobných údajov o ktorých hovorí GDPR článok 5 na str. 35 a 36 (pozri [72]):

1. Zákonnosť, spravodlivosť a transparentnosť[36] – informovanosť užívateľov o spôsobe spracúvania ich osobných údajov, existencia právneho základu pre ich spracúvanie atď.

[36]Pracovná skupina pre ochranu údajov vydala Pokyny k transparentnosti podľa nariadenia 2016/679 (pozri [82]).

2. Obmedzenie účelu – získavané na konkrétne určené, výslovne uvedené a legitímne účely a nesmú sa ďalej spracúvať spôsobom, ktorý nie je zlučiteľný s týmito účelmi

3. Minimalizácia údajov – osobné údaje musia byť primerané, relevantné a obmedzené na rozsah, ktorý je nevyhnutný vzhľadom na účely, na ktoré sa spracúvajú, Pracovná skupina pre ochranu údajov [78] na str. 11 hovorí o tom, že správcovia by mali byť schopný jasne vysvetliť dôvod získavania a uchovávania osobných údajov alebo prípadne zvážiť použitie súhrnných, anonymizovaných, alebo pseudonymizovaných údajov

4. Správnosť, resp. presnosť – osobné údaje by mali byť správne a podľa potreby aktualizované, a musí byť zabezpečené, aby sa osobné údaje (ktoré sú nesprávne z hľadiska účelov, na ktoré sa spracúvajú) bezodkladne vymazali alebo opravili

5. Minimalizácia uchovávania – ukladanie záznamov len na dobu potrebnú na účely, na ktoré sa osobné údaje spracúvajú a prípadne automatické mazanie záznamov v pravidelných časových intervaloch

6. Integrita a dôvernosť – spracúvané spôsobom, ktorý zaručuje primeranú bezpečnosť osobných údajov, vrátane ochrany pred neoprávneným alebo nezákonným spracúvaním a náhodnou stratou, zničením alebo poškodením, a to prostredníctvom primeraných technických alebo organizačných opatrení

ÚOOÚ vydal základný jednoduchý návod využiteľný malými správcami údajov, živnostníkmi či menšími podnikmi, ktorý hovorí o tom ako zaobchádzať s osobnými údajmi (pozri [75]), a ktorý primárne vychádza zo spomenutého GDPR článku 5 na str. 35 a 36.

GDPR na jednej strane dáva množstvo pokynov pokiaľ ide o ochranu údajov. Tieto pokyny sú zosumarizované v GDPR článku 25 na str. 48 (pozri [72]), kde sa hovorí o špecificky navrhnutej a štandardnej ochrane údajov[37], a ide teda o prijatie a vykonanie primeraných technických a organizačných opatrení ako je:

- pseudonymizácia a minimalizácia údajov

- začlenenie do spracúvania nevyhnutné záruky s cieľom splniť požiadavky tohto nariadenia a chrániť práva dotknutých osôb

[37]Pracovná skupina pre ochranu údajov vydala Pokyny k špecificky navrhnutej a štandardnej ochrane údajov podľa nariadenia 2016/679 (pozri [83]).

- spracúvanie len osobných údajov, ktoré sú nevyhnutné pre každý konkrétny účel spracúvania a na uvedenú povinnosť sa vzťahuje:
  - množstvo získaných osobných údajov
  - rozsah ich spracúvania
  - doba ich uchovávania
  - ich dostupnosť

Na druhej strane nariadenie GDPR vyžaduje aj ochranu týchto údajov, resp. vyžaduje zaistiť tzv. primeranú úroveň bezpečnosti spracúvania[38], ktorá vychádza z GDPR článku 32 na str. 51 a 52 (pozri [72]), a ide teda opäť o prijatie primeraných technických a organizačných opatrení, ktoré zahŕňajú:

- pseudonymizáciu a šifrovanie osobných údajov
- schopnosť zabezpečiť trvalú dôvernosť, integritu, dostupnosť a odolnosť systémov spracúvania a služieb
- schopnosť včas obnoviť dostupnosť osobných údajov a prístup k nim v prípade fyzického alebo technického incidentu
- proces pravidelného testovania, posudzovania a hodnotenia účinnosti technických a organizačných opatrení na zaistenie bezpečnosti spracúvania

Šifrovanie, a teda bezpečnosť spracúvania by podľa Tankard [84] malo byť štandardným nástrojom na ochranu všetkých údajov, ktoré sa v organizácií ukladajú, resp. nachádzajú a rovnako to platí aj pri ich prenose. Napríklad v prípade, že dôjde k úniku dát a tieto dáta boli šifrované (samozrejme za predpokladu správne implementovaného šifrovania) tak podľa Tankard [84] nevzniká povinnosť oznámiť tento únik dotknutým osobám. Aj napriek tomu, že sú údaje šifrované je podľa Tankard vhodné minimalizovať množstvo zbieraných údajov, čím sa znižuje pravdepodobnosť porušenia požiadavky GDPR, ktorá hovorí o minimalizácií údajov, ale samozrejme aj množstvo údajov, ktoré je potrebné chrániť. Šifrovanie samo o sebe nestačí, a preto organizácie musia zaistiť, aby bola zaistená kontrola prístupu k údajom po ich dešifrácií atď.

Na preukázanie súladu s vyššie spomenutými požiadavkami (článok 25 a 32 nariadenia GDPR) sa môže použiť **schválený certifikačný mechanizmus** (pozri sekciu 3.3.2) alebo **zaistením potrebnej dokumentácie a prístupu k činnostiam spracúvania tak, aby tento súlad bolo možné posúdiť**

[38] „*Pri posudzovaní primeranej úrovne bezpečnosti sa prihliada predovšetkým na riziká, ktoré predstavuje spracúvanie, a to najmä v dôsledku náhodného alebo nezákonného zničenia, straty, zmeny, neoprávneného poskytnutia osobných údajov, ktoré sa prenášajú, uchovávajú alebo inak spracúvajú, alebo neoprávneného prístupu k takýmto údajom. - GDPR článok 32(2).*“

napr. v rámci kontroly dozorného orgánu. V prípade požiadaviek v oblasti bezpečnosti spracúvania sa na preukázanie súladu môže použiť aj **schválený kódex správania** (v prípade, že pre danú oblasť existuje) (pozri sekciu 3.3.1).

Na záver je potrebné dodať, že GDPR článok 33 na str. 52 (pozri [72]) hovorí o tom, že „*v prípade porušenia ochrany osobných údajov prevádzkovateľ bez zbytočného odkladu a podľa možnosti najneskôr do 72 hodín po tom, čo sa o tejto skutočnosti dozvedel, oznámi porušenie ochrany osobných údajov dozornému orgánu*“ a GDPR článok 34 na str. 52 (pozri [72]) hovorí, že „*v prípade porušenia ochrany osobných údajov, ktoré pravdepodobne povedie k vysokému riziku pre práva a slobody fyzických osôb, prevádzkovateľ bez zbytočného odkladu oznámi porušenie ochrany osobných údajov dotknutej osobe.*“

Z vyššie spomenutého vyplýva, že organizácie, resp. prevádzkovatelia musia byť schopní zachytiť to, že došlo k porušeniu ochrany osobných údajov, a práve záznamy, resp. bezpečnostné auditné záznamy (ktoré je potrebné uchovávať a analyzovať) im môžu pomôcť k tomu, aby boli schopní zachytiť a preukázať ako, a kedy došlo k úniku dát, resp. k porušeniu ochrany osobných údajov. Na druhej strane je potrebné mať na pamäti, že súčasťou týchto bezpečnostných auditných záznamov môžu byť aj osobné, resp. citlivé údaje.

3.3.1 Kódex správania

Kódexy správania a monitorovanie schválených kódexov správania sa nachádza v GDPR článkoch 40 a 41 na str. 56 až 58 (pozri [72]). Podľa ÚOOÚ [85] kódex správania definuje základné zásady, postupy a požiadavky na spracúvanie osobných údajov v konkrétnom odvetví. Z toho vyplýva, že je určený skupine správcov alebo spracovávateľov rovnakého typu (napr. banky, poisťovne apod.). Kódex správania musí byť spracovaný tak, aby pokryl požiadavky upravené GDPR pre spracúvanie osobných údajov konkrétneho druhu a navyše zásady, požiadavky a postupy musia byť formulované natoľko konkrétne, aby ich plnenie bolo overiteľné v rámci monitorovania, ktoré je vykonávané nezávislým subjektom. Nezávislý subjekt musí byť akreditovaný Úřadem pro ochranu osobních údajů, ktorý musí preukázať odborné znalosti v oblasti, pre ktorú je kódex určený, znalosť a skúsenosti s vykonávaním auditu atď. V prípade orgánov verejnej moci musí byť monitorovanie zaistené v rámci vnútorných kontrolných mechanizmov.

Neexistuje povinnosť prihlásiť sa k dodržiavaniu kódexu správania, ak sa však správca alebo spracovávateľ prihlási, je povinný sa podrobiť pravidelnému monitorovaniu kódexu správania nezávislým subjektom, ktoré sa vykonáva v pravidelných jednoročných až dvojročných intervaloch.

ÚOOÚ vydal metodiku [86] (metodickú príručku verzie 2.0) k vytváraniu kódexov správania v súlade s GDPR. Cieľom materiálu je vysvetliť ich podstatu, priblížiť a pomôcť verejnosti s lepšou orientáciou v tejto oblasti, naznačiť

základne pravidlá a odporučiť praktické postupy. Obsahom je aj stručný popis mechanizmu monitorovania kódexu. ÚOOÚ kódexy nespracováva, ale vo fáze prípravy poskytuje potrebné konzultácie a následne predložený kódex posúdi, a vydá stanovisko o tom, či je návrh kódexu v súlade s nariadením a prípadne ho aj následne schváli.

Európsky výbor pre ochranu osobných údajov hraje významnú úlohu pri uplatňovaní GDPR v oblasti kódexu správania. Jedná sa hlavne o metodický návod (Guidelines on Codes of Conduct and Monitoring Bodies under Regulation 2016/679) (pozri [87]), ktorý bol prijatý 19. 02. 2019. ÚOOÚ upozorňuje, že je nutné sa tzv. zladiť s týmto návodom, ktorý pripravil Európsky výbor pre ochranu osobných údajov, aby nedochádzalo k vytvárania neschváliteľných kritérií a rýchlym zmenám v podmienkach akreditácie subjektov pre monitorovanie kódexu a v štruktúre kódexu správania.

V čase písania tejto podsekcie (30.08.2019) sa nám nepodarilo nájsť žiaden kódex správania, ktorý by bol schválený ÚOOÚ v Českej republike. Na stránkach Ministerstva práce a sociálních věcí [88] sa nám, ale podarilo nájsť dokument, ktorý je označený ako *Doporučený postup č. 02/2018 - MPSV - GDPR - Kodex chování* a na str. 5 tohto dokumentu sa píše, že „*Podle aktuálních vyjádření ÚOOÚ se však nepředpokládá vydání jeho schválení před koncem prvního pololetí 2019.*“. Vyzerá to ale tak, že ani do tejto doby nebol schválený. Ak by sme sa pozreli na iné lokálne iniciatívy napr. Úrad na ochranu osobných údajov Slovenskej republiky má na svojich stránkach [89] zverejnený jeden kódex správania, konkrétne *Kódex správania pre spracúvanie osobných údajov advokátmi.*

3.3.2 Schválený certifikačný mechanizmus

Certifikácia a certifikačné subjekty sa nachádzajú v GDPR článkoch 42 a 43 na str. 58 až 60 (pozri [72]). Podľa ÚOOÚ [90] je osvedčenie (certifikát) o ochrane osobných údajov dokument, ktorý je vydaný subjektom pre vydávanie osvedčení (certifikačný orgán), ktorým subjekt (správca, spracovávateľ atď.) preukazuje zaistenie súladu s požiadavkami nariadenia GDPR. Tak ako to platilo v prípade kódexov správania tak aj v tomto prípade platí, že neexistuje žiadna povinnosť žiadať o vydanie osvedčenia, resp. certifikátu, jedná sa o jednu z voliteľných variant medzi ktoré mimo iného (ako už bolo vyššie spomenuté) patrí dodržiavanie kódexu správania a zaistenie potrebnej dokumentácie a prístupu k činnostiam spracúvania.

Osvedčenie (certifikát) o ochrane osobných údajov môže slúžiť ako:

- doklad toho, že činnosti spracúvania, ktoré sú vykonávané správcom alebo spracovávateľom sú v súlade s nariadením GDPR
- doklad, ktorým je možné preukázať to, že pri správnom nastavení parametrov produktu alebo služby je daný produkt alebo služba v súlade s nariadením GDPR

- doklad, ktorý môže zjednodušiť prenos osobných údajov do zahraničia tým, že sa preberajúca osoba (spracováva osobné údaje v zemi, kde nie je zaistená úroveň ochrany osobných údajov) preukáže týmto platným dokladom (osvedčením)

Spomínané osvedčenie môžu vydávať len subjekty pre vydávanie osvedčení tzv. certifikačné orgány, ktoré sú pre túto činnosť akreditované. Za akreditáciu subjektov pre vydávanie osvedčení bude, resp. už je v Českej republike zodpovedný vnútroštátny akreditačný orgán – Český institut pro akreditaci, o.p.s.

ÚOOÚ zverejnil 13.12.2017 návrh kritérií pre vydávanie osvedčení (certifikátov) a návrh kritérií pre akreditáciu subjektov pre vydávanie osvedčení (certifikátov) [91], ktorý bol vytvorený v spolupráci s Českým institutem pro akreditaci. Významnú úlohu pri tvorbe kritérií opäť hraje Európsky výbor pre ochranu osobných údajov, ktorý vydal tieto dva dokumenty:

- tzv. vodítka týkajúce sa certifikačných kritérií [92] (vydané 04.06.2019)
- tzv. vodítka týkajúce sa akreditačných kritérií [93] (vydané 04.06.2019)

V čase písania tejto podsekcie (31.08.2019) mal ÚOOÚ na svojich stránkach (pozri [90]) zverejnený harmonogram z dňa 27.10.2017, ktorý tvrdil/-tvrdí, že vyššie spomínané kritéria (zverejnené ÚOOÚ 13.12.2017) budú odovzdané Českému institutu pro akreditaci, o.p.s. až po finálnej úprave, a teda až potom, čo Európsky výbor pre ochranu osobných údajov vydá tzv. vodítka a ÚOOÚ ich zohľadní vo svojom materiály. Európsky výbor pre ochranu osobných údajov svoje vodítka už vydal (04.06.2019), ale v súčasnosti to vyzerá tak, že ÚOOÚ ich vo svojom materiály ešte nezohľadnil, a teda ani neodovzdal Českému institutu pro akreditaci. Tým pádom nie je možné v súčasnej dobe žiadať o akreditáciu, a tým pádom nie je možné žiadať ani o vydanie osvedčenia (certifikátu) k určitému produktu, službe alebo spracúvaniu. ÚOOÚ tvrdí, že v okamihu, keď to bude možné tak bude verejnosť informovaná.[39]

3.3.3 Odporúčania a nariadenia

V súčasnej dobe to vyzerá tak, že neexistuje žiaden schválený kódex správania a ani žiaden schválený certifikačný mechanizmus. Jedinou cestou je ísť formou zaistenia potrebnej dokumentácie (podľa návrhu [91] by mohlo ísť o bod 4.7.4 na str. 26) a prístupu k činnostiam spracúvania tak, aby tento súlad bolo možné posúdiť napr. v rámci kontroly dozorného orgánu a/alebo sa

[39]Stránka bola aktualizovaná 30.09.2019 a obsahuje nové informácie o tom, že ÚOOÚ tieto tzv. vodítka vydané Európskym výborom pre ochranu osobných údajov zohľadnil a čaká sa na posúdenie týchto navrhovaných požiadaviek pre vydávanie osvedčení Európskym výborom pre ochranu osobných údajov a tieto požiadavky budú odovzdané Českému institutu pro akreditaci, o.p.s. až po ich schválení daným orgánom.

oprieť o rôzne odporúčania, a nariadenia, ktoré existujú. Nasledujúce podsekcie sa budú venovať rôznym odporúčaniam a nariadeniam, ktoré sa týkajú, resp. súvisia s bezpečnostnými auditnými záznamami.

3.3.3.1 Odporúčania

Podľa Thies [94] patrí medzi osvedčené postupy v oblasti *log* manažment, ktoré dokážu pomôcť so zaistením súladu s nariadením GDPR:

1. centralizácia ukladania záznamov
2. odstraňovanie záznamov, ktoré sa ukladajú lokálne na jednotlivé servery (súvisi s 1. bodom) – záznamy uložené na centrálnom úložisku by mali byť odstránené, resp. by sa ani nemali ukladať na lokálne servery, aby sa predišlo duplikácií dát
3. štruktúrovanie záznamov – štruktúrované záznamy uľahčujú napr. anonymizáciu citlivých údajov, ktorá je spomenutá v ďalšom bode
4. anonymizácia citlivých dát, ktoré sú obsahom záznamov – napr. pomocou šifrovania alebo priamo odstránením týchto dát (tento spôsob, ale môže znížiť hodnotu záznamov)
5. šifrovanie záznamov pred prenosom do centrálneho úložiska

Podľa Nguyen [95] patrí medzi osvedčené postupy:

1. uchovávanie záznamov len po dobu, po ktorú sú potrebné
2. šifrovanie záznamov, monitorovanie a obmedzenie prístupu k ním
3. brať súlad s nariadením GDPR ako investíciu

Národné centrum kybernetickej bezpečnosti [96] (ako príklad lokálnej iniciatívy v Českej republike) vydalo odporúčanie na minimálne požiadavky pre záznamy, ktoré musia byť zaistené pre spoľahlivú ex-post analýzu kybernetických bezpečnostných incidentov, ktoré vychádza už zo spomínaného dokumentu NIST800-92 [17]. V tomto odporúčaní sa nachádzajú jednotlivé skupiny záznamov, ktoré by mohli firmám slúžiť ako vodítko na to, aké záznamy by mohli v rámci súladu s GDPR zaznamenávať. Je však dôležité podotknúť, že sa nejedná o nariadenie, ale len o odporúčanie.

Tabuľka 3.1 špecifikuje jednotlivé skupiny záznamov a stanovuje koeficient (počet dní) pre jednotlivé skupiny, kde jednotlivé skupiny sa podľa NCKB [96] delia do nasledujúcich kategórií:

- SEC – obsahom tejto skupiny je bezpečnostný softvér, IPS/IDS systémy, VPN, web proxy atď.

- OS – obsahom tejto skupiny sú servery, pracovné stanice a sieťové prvky a jedná sa o dva typy záznamov:
  - systémové udalosti – spustenie/zastavenie služby, vypnutie/zapnutie stanice apod.
  - udalosti auditu – prístupy k súborom, pokusy o ne/úspešné prihlásenia sa do systému, zmeny v nastaveniach apod.
- APP – obsahom tejto skupiny sú záznamy, ktoré zaznamenávajú chod aplikácií:
  - komunikácia klienta so serverom (C <> S) – klientské požiadavky prijaté serverom a ich odpovede
  - informácie o učte (ACC) – informácie o prihlásení sa k aplikácií alebo službe, zmeny v účtoch a v oprávneniach apod.
  - údaje o aktivite užívateľov (Aktivita) – napr. počet transakcií a ich objem
  - významné prevádzkové akcie – spustenie alebo ukončenie aplikácie, pády aplikácie alebo jej významné zmeny

Tabuľka 3.2 obsahuje odporúčania (minimálne požiadavky) pre rôzne kategórie záznamov (KII, VIS a ostatné), kde počet dní je možné nájsť v tabuľke 3.1.

Odporúčané minimálne požiadavky pre záznamy, ktoré sú definované v tabuľke 3.2 definuju aj to, ako často by sa mal vykonávať zber jednotlivých záznamov (Ako často je potrebné zasielať jednotlivé záznamy do *log* manažment). Pravidelnosť zberu jednotlivých záznamov, resp. zasielania sa líši podľa toho do akej skupiny/kategórie patria jednotlivé systémy. Pre KII (proces určenia daného systému do tejto kategórie je možné nájsť v zdroji [97]) je to najneskôr každých 5 minút, pre VIS (proces určenia daného systému do tejto kategórie je možné nájsť v zdroji [98]) je to každých 15 až 60 minút a pre kategóriu ostatné to je 3 až 24 hodín.

3.3.3.2 Nariadenia

GDPR nenariaďuje firmám, aké záznamy, resp. aké kategórie (skupiny) záznamov majú auditovať/zaznamenávať, ale hovorí o tom, ako je potrebné so záznamami, ktorých obsahom môžu byť aj osobné údaje pracovať (chrániť, spracovávať atď.) a aké všetky podmienky je potrebné splniť na to, aby bolo vyhovené požiadavkám GDPR.

Ak by sme sa pozreli do dokumentu Kritéria pro vydávání osvědčení a kritéria pro akreditaci (KVO) [91] (ktorý je v čase písania diplomovej práce 31.08.2019 stále návrhom a ktorý už bol spomínaný vyššie) a na **kritéria hodnotenia (certifikačné požiadavky)**, a konkrétne na bod 4.7, ktorý začína

Tabuľka 3.1: Jednotlivé skupiny záznamov a ich koeficienty (počet dní) [96]

SEC	**AV**	**IDS/IPS**	**Vzdial. prístup**	**Web proxy**	**Autentiz. server**	**Vulnerability management**	**Smerovače a prepínače**	**Radius**
Počet dní	30	30	30	30	30	30	30	30
OS	**Systém**	**Audit**						
Počet dní	30	30						
APP	**C <>S**	**ACC**	**Aktivita**	**Akce**				
Počet dní	7	7	1	30				

Tabuľka 3.2: Odporúčané minimálne požiadavky pre záznamy rôznych kategórií (KII, VIS a ostatné) [96]

Kategórie	**Ostatné**	**VIS**	**KII**
Retencia dát SEC	min. 1 * počet dní	min. 3 * počet dní	min. 6 * počet dní
Retencia dát OS	min. 1 * počet dní	min. 3 * počet dní	min. 6 * počet dní
Retencia dát APP	min. 1 * počet dní	min. 3 * počet dní	min. 6 * počet dní
Rotácia záznamov	každý týždeň alebo po dosiahnutí 25 MB	medzi 6-24 hodinami alebo pri dosiahnutí 5 MB	medzi 15-60 minútami alebo pri dosiahnutí 1 MB
Ako často zasielať záznamy do log manažment	každých 3-24 hodín	každých 15-60 minút	najneskôr každých 5 minút
Kontrola integrity (rotácia)	voliteľná	áno	áno
Šifrovanie záznamov	voliteľné	áno	áno
Šifrovaný prenos záznamov do log manažment	voliteľný	áno	áno

na str. 22 a hovorí o tom, či je zaistené riadenie bezpečnosti osobných údajov v súlade s GDPR článkami 24 (Zodpovednosť prevádzkovateľa), 25 (Špecificky navrhnutá a štandardná ochrana údajov) a 32 (Bezpečnosť spracúvania) zistili by sme, že je potrebné dodržať nasledujúce body, resp. požiadavky:

- údaje sú spracovávané spôsobom, ktorý zaistí ich náležité zabezpečenie pred:
  - neoprávneným a protiprávnym spracúvaním
  - náhodnou stratou
  - zničením
  - poškodením
- 1. varianta – požiadavky sú určené normou ČSN ISO/IEC 27001 Informační technologie – bezpečnostní techniky – Systém řízení bezpečnosti informací – Požadavky
  2. varianta – jednotlivé požiadavky sú postupne vymenované v spomínanom dokumente [91] od str. 23 po 26, nás zaujíma bod 4.7.3 Súbor technických a organizačných opatrení a konkrétne bod 4.7.3.4, ktorý hovorí o logovaní a monitorovaní:
     - existujú opatrenia na overenie prístupu užívateľov a tieto opatrenia sú testované
     - prístup je monitorovaný a zaznamenávaný:
       * je uplatnené tzv. on-line monitorovanie a sledovanie
       * je uplatňované zaznamenávanie činností všetkých užívateľov
       * zaznamenávanie je škálovateľné (podľa činnosti, doby a obsahu uchovávania záznamov)
       * záznam je zabezpečený (vrátane prístupu)
       * vytváranie záznamov nemôže byť blokované
       * sú nasadené nástroje pre jednoduchú analýzu záznamov
       * zavedenie a konfigurácia prístupových práv k zaznamenávaniu (logovaniu) je vykonávaná oprávnenou osobou
       * údaje záznamov sú bezpečne zničené po uplynutí doby ich uchovávania

Podľa Tankard [84], ale aj napr. podľa Asociace za lepší ICT řešení [99] použitie, resp. dodržanie normy ISO 27001 (prípadne podľa dokumentu Kritéria pro vydávání osvědčení a kritéria pro akreditaci (KVO) [91] špecifickejšie národná norma, ktorá bola spomenutá vyššie – ČSN ISO/IEC 27001 Informační technológie – bezpečnostní techniky – Systém řízení bezpečnosti informací – Požadavky) dokáže pomôcť zaistiť princípy, resp. požiadavky,

ktoré sú obsahom GDPR a to tak, že touto normou budú dosiahnuté vhodné technické a organizačné opatrenia, resp. odporučenia na ochranu informácií (osobných údajov), ktoré môžu byť obsahom záznamov (bezpečnostných auditných záznamov).

V auguste 2019 vyšla norma pod označením ISO 27701, ktorá podľa Tayllorcox [100] dokáže pomôcť organizáciám splniť požiadavky GDPR. Táto nová norma je založená na požiadavkách, princípoch, postupoch a pravidlách pre riadenie informačnej bezpečnosti podľa normy ISO 27001. ISO 27701 túto normu doplňuje, resp. jedná sa o rozšírenie požiadaviek na ochranu osobných údajov. Ide teda v oblasti ochrany osobných údajov o úplne nový certifikačný mechanizmus, ktorým je možné preukázať splnenie GDPR.

Ako už z vyššie uvedeného vyplýva, tak GDPR nenariaďuje firmám aké záznamy majú auditovať, resp. zaznamenávať, a preto vznikla tabuľka 3.3 (ktorá primárne vychádza z vlastnej interpretácie autora tejto diplomovej práce, ale aj zo zdroja [101] a [102]. Táto tabuľka je prehľadom toho, aké typy záznamov (zdroje) by mohli dané organizácie zaznamenávať a aké typy detekovaných udalostí je možné v jednotlivých záznamov nájsť z pohľadu rôznych bezpečnostných a systémových udalostí. Tabuľka bude slúžiť aj ako podklad pre praktickú časť tejto diplomovej práce z hľadiska výberu typu (zdroja, prípadne zdrojov) záznamov, ktorým sa budeme v praktickej časti venovať.

Na záver je potrebné dodať, že tabuľka 3.3 nie je konečným zoznamom všetkých typov záznamov (zdrojov), ktoré existujú a častokrát to závisí na konkrétnych prípadov použitia, potrieb a úrovne bezpečnostného monitorovania jednotlivých organizácií a je len na nich, ktoré typy (zdroje) záznamov sa rozhodnú zaznamenávať a následne analyzovať.

3.4 SIEM systémy a GDPR

Podľa Vijayan [103], ale aj podľa Boucas [104] môžu SIEM systémy výrazne pomôcť organizáciám pri plnení požiadaviek nariadenia GDPR, resp. pomôcť s tým, aby boli organizácie v súlade s týmto nariadením. Na druhej strane netreba zabúdať na to, že záznamy, ktoré sú v rámci SIEM systémov zbierané a spracúvavané, môžu obsahovať osobné údaje, čo môže priniesť organizáciám určité riziko, ale neznamená to, že musí automaticky dôjsť k porušeniu požiadaviek GDPR (legitímny záujem – GDPR článok 6). Preto je dôležité pochopiť potenciálne príležitosti a zmierniť hrozby, ktoré so sebou môžu priniesť SIEM systémy pri plnení požiadaviek GDPR. Podľa Boucas [104], ale aj podľa Subha [105] dokážu SIEM systémy, čo sa týka dodržiavania požiadaviek GDPR pomôcť s:

- zabezpečením integrity osobných údajov (článok 32) a to tým, že je možné zistiť kto a aké operácie vykonáva nad záznamami a overiť tak, či to je alebo nie je legitímne

Tabuľka 3.3: Prehľad typov (zdrojov) záznamov a detegovaných udalostí

Typ záznamu (zdroj)	**Typ detegovanej udalosti**
HTTP request (web access logs)	zneužívanie zraniteľností týkajúcich sa webových stránok (*web security vulnerabilities*)
Linux/Windows (Security) logs	neautorizované/neúspešné pokusy o prihlásenie sa do systému, útoky hrubou silou atď.
Web proxy logs	prítomnosť škodlivého softvéru, "neobvyklé"správanie sa pri prehľadávaní stránok atď.
System logs (HDFS, BGL atd.)	abnormálne správanie sa systému (odhalenie zlyhania systému apod.)
Firewall logs	detekcia škodlivej aktivity v sieti danej organizácie
DNS logs	exfiltrácia dát a/alebo prítomnosť škodlivého softvéru (malvér)
Network devices logs	zmeny konfigurácií na jednotlivých zariadeniach, zlyhanie hardvéru atp.
Authentication logs	detekcia zlyhania autentifikácie, činnosti súvisiace s autorizáciou atď.
Application logs	chybové udalosti vznikajúce pri behu danej aplikácie, prihlásenia sa do aplikácie atď.
NetFlow, IPFIX atď.	detekcia neoprávneného prieniku, malvéru, či abnormálnej prevádzky na sieti

- zabezpečením dôvernosti, integrity a dostupnosti systémov a služieb, ktoré spracúvajú osobné údaje, tým, že SIEM systémy neustále sledujú bezpečnostné udalostí, ktoré sa dejú naprieč celou infraštruktúrou danej organizácie v ktorej sú uložené, a v ktorej sa spracúvajú tieto osobné údaje
- detekciou, prevenciou a investigáciou úniku dát (s čím súvisi aj nasledujúci bod)
- prípadným oznamovaním závažného úniku dát orgánom EÚ, kedy je potrebné poskytnúť podrobné informácie týkajúce sa rozsahu úniku dát, počtu dotknutých záznamov atď. (GDPR článok 33), a to tak, že pomocou SIEM systémov je napr. možné zistiť kedy došlo k úniku dát, aké dáta boli odcudzené atď.
- plnením požiadavky GDPR, ktorá sa týka práva na vymazanie (právo „na zabudnutie“, GDPR článok 17) a to tak, že pomocou SIEM systémov, resp. záznamov, ktoré sa v nich nachádzajú je možné overiť, či došlo k odstráneniu požadovaných dát, ale aj to kto a kedy túto operáciu odstránenia dát vykonal (súvisí s prvým bodom)

Ako už bolo vyššie spomenuté tak obsahom záznamov môžu byť osobné údaje, čo potenciálne môže spôsobiť problémy pri súlade s GDPR. Na zmiernenie tohto rizika sa podľa Boucas [104] organizácie môžu rozhodnúť použiť pseudonymizáciu, čím sa znižuje riziko identifikácie konkrétnej osoby a/alebo použiť šifrovanie záznamov, resp. dát, čím sa osobné údaje (ktoré sú obsahom záznamov) chránia pri ich uchovávaní. Navyše SIEM systémy dokážu sledovať prístupy k miestam, kde sú záznamy, resp. dáta uložené a v prípade detekcie neoprávneného prístupu alebo neoprávnených pokusoch o presun týchto dát túto skutočnosť oznámiť. Podľa Vijayan [103] organizácie vo väčšine prípadoch nepotrebujú súhlas jednotlivca, resp. jednotlivcov na zhromažďovanie a spracúvanie ich dát, pretože sa tieto dáta, resp. záznamy zhromažďujú a spracúvajú za legitímnym účelom, ktorým je bezpečnosť. Na druhej strane aj naďalej platí, že postup musí byť zadokumentovaný a transparentný.

Na záver je potrebné dodať, že podľa Boucas [104] SIEM systémy dokážu pomôcť s niektorými špecifickými technickými a bezpečnostnými požiadavkami, ktoré so sebou GDPR prináša, no treba mať na pamäti, že nasadenie SIEM systému nemusí automaticky znamenať splnenie všetkých požiadaviek tohto nariadenia.

3.5 Kritika GDPR

Už len z pohľadu toho, že vznikol Európsky výbor pre ochranu osobných údajov (nástupca Pracovnej skupiny pre ochranu údajov, známej pod skratkou WP29, pozri [77]), ktorý prináša rôzne pokyny, odporučenia a osvedčené

postupy v oblasti GDPR a prispieva ku konzistentnému uplatňovaniu pravidiel ochrany osobných údajov v celej EÚ sa dá konštatovať, že samotný výklad GDPR môže byť v mnohých prípadoch nejasný. Je preto potrebné si pomáhať inými zdrojmi ako sú už spomínané stanoviská Európskeho výboru pre ochranu osobných údajov, ktoré výklad GDPR „upresňujú“ a rozširujú. Pri výklade GDPR častokrát pomáhajú rôzne lokálne iniciatívy jednotlivých štátov ako to je napr. v prípade Českej republiky ÚOOÚ [76] alebo v prípade Spojeného kráľovstva nezávislý orgán (ICO) zriadený na podporu dodržiavania informačných práv vo verejnom záujme, ktorý podporuje otvorenosť verejných orgánov a súkromie jednotlivcov.

GDPR z hľadiska bezpečnosti ochrany osobných údajov síce so sebou na jednej strane prináša rôzne zásady ochrany a spracúvania osobných údajov (GDPR článok 5), pokyny štandardnej ochrany údajov (GDPR článok 25), či vyžaduje zaistiť primeranú úroveň bezpečnosti spracúvania (GDPR článok 32) a hovorí aj o tom ako je možné preukázať súlad s týmito požiadavkami (pomocou schváleného kódexu správania, schváleného certifikačného mechanizmu, či zaistením potrebnej dokumentácie a prístupu k činnostiam spracúvania). Na druhej strane je potrebné povedať, že do dnešného dňa (31.08.2019) neexistuje v Českej republike žiaden schválený kódex správania a ani žiaden schválený certifikačný mechanizmus, a teda nie je jasne povedané podľa čoho je potrebné postupovať pri preukázaní súladu s GDPR. V tomto prípade sa je potrebné držať rôznych doporučení, resp. nariadení a ISO noriem, ktoré boli vyššie spomenuté, ale ani to nemusí zaistiť zbavenie sa firmy zodpovednosti za možné nesplnenie a nedodržanie cieľov tohto GDPR nariadenia.

Tieto moje slová potvrdzuje aj napr. vyjadrenie firmy LogManager, ktorá vo svojom *whitepaper* (pozri [106]) tvrdí, že táto GDPR regulácia nie je zrovna napísaná zrozumiteľným úradným jazykom a že mnoho organizácií rieši otázku aké opatrenia, a v akých oblastiach sú podľa požiadaviek GDPR povinný dodržiavať. Organizácie sa takisto zaoberajú otázkou aké systémy a riešenia im môžu dodržiavanie týchto požiadaviek zaistiť.

3.6 Zhrnutie

GDPR je v celej EÚ jednotne účinné a vymáhateľné od 25. mája 2018 a nahradzuje, a zjednocuje zákony o ochrane osobných údajov všetkých členských zemí EÚ. Nariadenie obsahuje 173 recitálov a 99 článkov, v ktorých sa nachádzajú definície osobných údajov, online identifikátorov, citlivých osobných údajov, všeobecných ustanovení, zásad, práv atď.

S informáciami, ktoré môžu obsahovať osobné údaje je potrebné zachádzať opatrne, uistiť sa, že existuje jasný dôvod na ich spracovanie, ale najmä zabezpečiť ich bezpečné ukladanie a zlikvidovanie. Spracovanie údajov nesmie nadmerne zasahovať do súkromia, účel spracovaných údajov musí byť jasný, spracovanie údajov musí byť legitímne a nesmie byť v rozpore s právnymi pred-

pismi, či morálkou atď. V Českej republike plní funkciu kontroly, dozorného a konzultačného úradu, a informačného kanálu ÚOOÚ. Rôzne odporúčania a osvedčené postupy v oblasti GDPR poskytuje a prináša Európsky výbor pre ochranu osobných údajov, ktorý prispieva ku konzistentnému uplatňovaniu pravidiel ochrany osobných údajov v celej EÚ.

Z GDPR článku 22 na str. 47 (pozri [72]) vyplýva, že v prípade strojového učenia a automatického spracúvania pri rozhodovacom procese musí existovať ľudský zásah alebo musí byť uplatnená jedna z troch výnimiek spomínaných v podsekcií 3.2.1, a navyše ak dotknutá osoba požiada o to, aby jej bolo vysvetlené prečo sa prišlo k danému rozhodnutiu tak musí byť táto požiadavka vyhovená, čo môže byť niekedy problém v prípade niektorých algoritmoch strojového učenia. Spôsob, akým sa strojové učenie vyvíja a používa môže viesť k vyvolaniu požiadavku na vykonanie DPIA, ktoré je nutné v prípadoch, ktoré môžu mať za následok vysoké riziko pre práva a slobody fyzických osôb.

Medzi všeobecné odporúčania v oblasti GDPR a auditných záznamov patrí napr. neukladať záznamy, ak to nie je potrebné (minimalizácia údajov a uchovávania), šifrovanie dát, resp. záznamov pri prenose a pri ukladaní na dané úložisko, obmedzeniu prístupu k jednotlivým záznamom, ale aj zaznamenávanie krokov, ktoré boli vykonané na zabezpečenie dát v prípade, že dôjde k ich úniku, aby sa prípadne bolo možné chrániť.

GDPR nenariaďuje firmám, aké záznamy, resp. aké kategórie (skupiny) záznamov majú auditovať, a preto vznikla tabuľka 3.3, ktorá je prehľadom toho, aké typy záznamov (zdroje) by mohli dané organizácie zaznamenávať a aké typy detekovaných udalostí je možné v jednotlivých záznamov nájsť z pohľadu rôznych bezpečnostných a systémových udalostí. Toto nariadenie skôr hovorí o tom, ako je potrebné so záznamami, ktorých obsahom môžu byť aj osobné údaje pracovať (chrániť, spracovávať atď.).

GDPR na jednej strane dáva množstvo pokynov pokiaľ ide o ochranu údajov, na druhej strane vyžaduje aj ochranu týchto údajov, resp. zaistenie primeranej úrovne bezpečnosti spracúvania. Avšak samotné nariadenie poskytuje len obecný návod, ako tieto opatrenia implementovať a na preukázanie súladu s požiadavkami GDPR hovorí toto nariadenie o schválenom certifikačnom mechanizme a schválenom kódexe správania, ale v súčasnej dobe to vyzerá tak, že v praxi zatiaľ nič podobné neexistuje.

Na druhej strane existujú rôzne NIST, ISO normy (bezpečnostné štandardy), odporúčania a nariadenia, ktorých splnením, resp. dodržaním je možné splniť GDPR požiadavky. Súčasťou odporúčaní je napr. pravidelnosť rotácie záznamov, retencia dát, ale aj napr. to, ako často zasielať záznamy do *log* manažment. Zber záznamov a ich následná analýza je jedným z podporných prostriedkov na to, aby mohol byť preukázaný súlad s nariadením GDPR, pričom, ale si treba dať pozor na to do akej skupiny/kategórie (KII, VIS alebo ostatné) patria jednotlivé systémy danej organizácie/firmy.

Zaistiť princípy, resp. požiadavky, ktoré sú obsahom GDPR by podľa Tankard [84], ale aj napr. podľa Asociace za lepší ICT řešení [99] a podľa doku-

mentu Kritéria pro vydávání osvědčení a kritéria pro akreditaci (KVO) [91] malo zaistiť dodržanie normy ISO 27001, ktorou budú dosiahnuté vhodné technické a organizačné opatrenia. Je potrebné dodať, že v auguste 2019 vyvinula ISO nový bezpečnostný štandard v kategórií ISO 27000 noriem pod označením ISO/IEC 27701, ktorý je rozšírením štandardov ISO 27001 a 27002. Podľa Tayllorcox [100] práve implementácia ISO noriem 27001 a tejto novej 27701 normy pomôže splniť legislatívne požiadavky na ochranu osobných údajov, tak ako to GDPR vyžaduje. Týmito medzinárodne uznávanými certifikátmi je možné deklarovať, že sú zavedené technické, procesné, organizačné a personálne opatrenia v oblasti ochrany osobných údajov. Podľa Tayllorcox [100] sa tak v súčasnosti jedná o jediný možný spôsob splnenia požiadaviek regulačných a kontrolných orgánov v oblasti spracúvania osobných údajov.

Na záver by som rád dodal, že autor tejto diplomovej práce nemá právnické vzdelanie, a pre bližšie a presnejšie informácie v konkrétnych prípadoch je potrebné vyhľadať odborníka, ktorý je oboznámený s týmto GDPR nariadením a prekonzultovať s ním jednotlivé kroky.

Kapitola 4

Strojové učenie využívané v oblasti auditu bezpečnostných záznamov

Všeobecnému popisu využitia strojového učenia na analýzu záznamov a identifikáciu podozrivej aktivity sme už venovali v sekcií 2.8. V tejto kapitole sa budeme primárne venovať detekcií podozrivej aktivity, konkrétne oblasti detekcie anomálií, ale aj konkrétnym algoritmom, ktoré budú použité pri implementácií praktickej časti tejto diplomovej práce. V poslednej sekcií sa budeme venovať krátkemu zhrnutiu tejto kapitoly.

4.1 Úvod

Podľa Chio a Freeman [2] sa prípady použitia strojového učenia v oblasti počítačovej bezpečnosti dajú rozdeliť do dvoch kategórií, a to na rozpoznávanie vzorov (z angl. *pattern recognition*) a na detekciu anomálií (z angl. *anomaly detection*). Hranica, ktorá tieto dve kategórie pomyselne rozdeľuje je nejasná, avšak každá z úloh, ktorá spadá do jednotlivých kategórií má jasne rozlíšiteľný cieľ.

Cieľom rozpoznávania vzorov je nájsť v dátach určité (explicitné alebo skryté) charakteristiky, ktoré sa následne môžu použiť na naučenie algoritmu na rozpoznávanie iných foriem údajov/dát, ktoré vykazujú rovnakú množinu charakteristík. Na druhej strane cieľom detekcie anomálií je vytvoriť pojem toho, čo je normálne, čo popisuje väčšinu (podľa Chio a Freeman to je 95%) daného súboru údajov a následne budú odchýlky akéhokoľvek druhu od tohto normálneho správania brané ako anomálie.

Keďže detekcia podozrivej aktivity vedie na detekciu anomálií v nasledujúcich sekciách tejto kapitoly sa budeme bližšie venovať práve tejto problematike.

4.2 Anomálie

V tejto a v nasledujúcej sekcií budeme primárne vychádzať z prieskumu/prehľadu od Chandola, Banerjee a Kumar [107] *Anomaly Detection : A Survey.*

Za zmienku, ale stojí aj *Deep learning for anomaly detection: A Survey* od Chalapathy a Chawla [108], ktorá je štruktúrou veľmi podobná, a ktorá používa rovnaký prístup ako vyššie spomínaný prieskum/prehľad, s tým rozdielom, že sa venuje neurónovým sieťam.

4.2.1 Definícia a rozdelenie anomálií

Hawkins [109] definuje anomáliu (odľahlú hodnotu) ako:

„*Udalosť/pozorovanie, ktoré sa od ostatných udalostí/pozorovaní líši natoľko, že vzbudzuje podozrenie, že bola/bolo generované iným mechanizmom.*“[40]

Podľa Chandola, Banerjee a Kumar [107] sú anomálie vzory v dátach, ktoré nevyhovujú presne definovanej predstave normálneho správania.

Anomálie sa môžu nachádzať v dátach z rôznych dôvodov ako je napr. prítomnosť škodlivej aktivity, porucha systému atď., a ich spoločnou vlastnosťou je to, že sú zaujímavé pre analytikov, a práve ich zaujímavosť a relevantnosť je kľúčovou vlastnosťou detekcie anomálií (pozri sekciu 4.3).

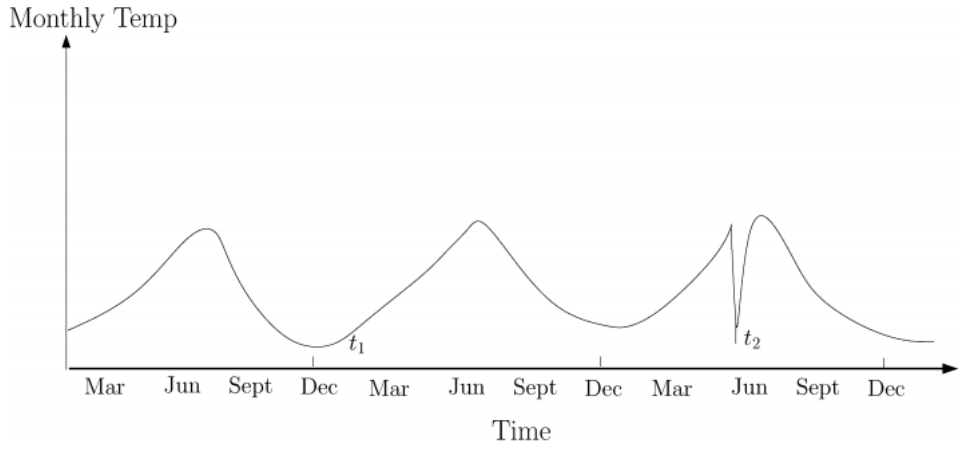
Anomálie sa podľa Chandola, Banerjee a Kumar [107], ale aj podľa Chalapathy a Chawla [108] dajú rozdeliť do nasledujúcich troch kategórií:

- bodové anomálie (z angl. *point anomalies*)
- podmienené/kontextové anomálie (z angl. *conditional/contextual anomalies*)
- skupinové/kolektívne anomálie (z angl. *group/collective anomalies*)

4.2.1.1 Bodové anomálie

Bodové anomálie sú považované za najjednoduchší typ anomálií a väčšina dostupnej literatúry sa podľa Chalapathy a Chawla [108] zameriava práve na tento typ anomálií. Bodové anomálie predstavujú nezrovnalostí/odchýlky, ktoré sa objavujú v dátach náhodne a nemusia mať žiaden osobitný výklad (bez kontextu). Príkladom môže byť zisťovanie podvodov s kreditnými kartami na základne odoslanej sumy a to tak, že ak sa odoslaná suma (transakcia) z účtu danej osoby výrazne líši od bežného rozsahu výdavkov (transakcií) danej osoby bude táto odoslaná suma (transakcia) považovaná za bodovú anomáliu.

[40] Preložené autorom tejto diplomovej práce.



Obr. 4.1: Kontextová anomália v čase t_2 v teplotnej časovej rade [107]

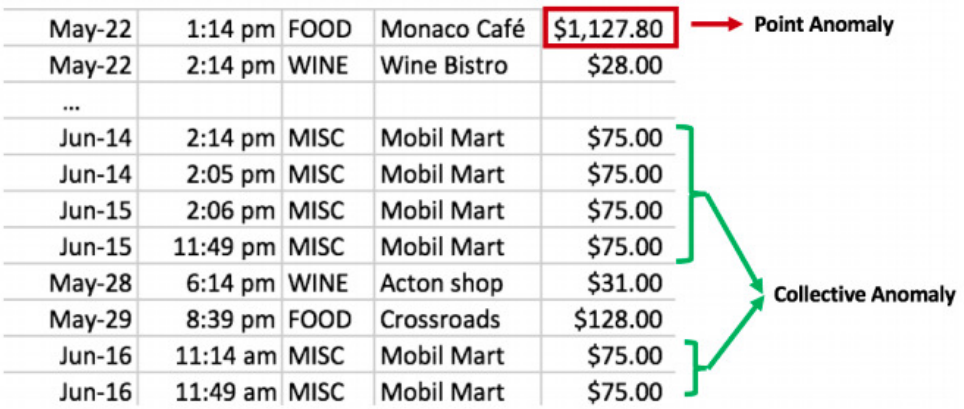
4.2.1.2 Podmienené/kontextové anomálie

Udalosť/inštancia sa považuje za podmienenú/kontextovú anomáliu v prípade, že sa táto udalosť/inštancia dá považovať za anomálnu v určitom špecifickom kontexte, ktorý je špecifikovaný ako súčasť formulácie problému. Tento typ anomálie sa identifikuje na základe kontextuálnych a tzv. behaviorálnych znakov, resp. atribútov. Kontextové atribúty sa používajú na určenie kontextu (v prípade časových rad je čas kontextovým atribútom) a behaviorálne atribúty definujú nekontextové charakteristiky inštancie (v oblasti podvodov s kreditnými kartami sú za behaviorálne atribúty považované napr. suma transakcie, príjemca atď.).

Pri tomto type anomálií je tzv. anomálne správanie určované pomocou hodnôt behaviorálnych atribútov v konkrétnom špecifickom kontexte. To znamená, že daná inštancia/udalosť môže byť v určitom kontexte považovaná za anomáliu, ale tá istá inštancia/udalosť (z hľadiska behaviorálnych atribútov) môže byť považovaná za normálnu v inom kontexte. Na obr. 4.1 je možné vidieť príklad kontextovej anomálie. Na obrázku je zobrazená teplotná časová rada za posledné roky v danej oblasti (pravdepodobne mierne podnebné pásmo). Z obrázka je možné vidieť, že v čase t_1 a t_2 je teplota rovnaká, ale teplota v čase t_1 je počas zimného obdobia (v decembri) považovaná za normálnu, ale rovnaká hodnota teploty v čase t_2, ale už v inom kontexte (počas letného obdobia v júni) bude považovaná za anomálnu.

4.2.1.3 Skupinové/kolektívne anomálie

Za skupinové/kolektívne anomálie sú považované skupiny jednotlivých udalostí/inštancií u ktorých platí, že každá jedná udalosť/inštancia z danej skupiny sa izolovane javí ako normálna udalosť/inštancia, ale v prípade, že pozo-



Obr. 4.2: Bodová a kolektívna anomália z oblasti detekcie podvodov s kreditnými kartami [108]

rujeme tieto udalosti/inštancie v rámci danej skupiny tak vykazujú nezvyčajné charakteristiky. Na obr. 4.2 je možné vidieť príklad bodovej a skupinovej/kolektívnej anomálie.

4.2.2 Výzvy a problémy

Ako už bolo vyššie spomínané tak anomálie sú vzory v dátach, ktoré nevyhovujú presne definovanej predstave normálneho správania. Úlohou detekcie anomálií by malo byť definovanie regiónu/regiónov, ktoré predstavujú normálne správanie a každú udalosť/inštanciu, ktorá do tohto regiónu/regiónov nepatrí označiť za anomáliu. V skutočnosti to, ale podľa Chandola, Banerjee a Kumar [107] také jednoduché nie je, pretože existuje niekoľko faktorov, ktoré detekciu anomálií komplikujú:

- hranica medzi normálnym a anomálnym (neobvyklým) správaním býva častokrát nepresná, vďaka čomu môže byť normálne správanie, ktoré je blízko tejto hranice považované za anomálne
- pri detekcií anomálií v oblasti škodlivých aktivít sa útočníci snažia prispôsobiť tak, aby sa abnormálne (neobvyklé) správanie javilo ako normálne, čím sa detekcia stáva oveľa zložitejšou
- vďaka vývoju normálneho správania nemusí byť v budúcnosti jeho súčasný pojem dostatočne charakteristický
- pojem anomália sa líši naprieč rôznymi aplikačnými doménami (aplikácia techniky/algoritmu vyvinutého v jednej doméne nemusí byť dostačujúca v inej doméne)

- (ne)dostupnosť anotovaných/oštítkovaných (z angl. *labeled*) dát na trénovacie a validačné účely rôznych modelov, ktoré sú určené na detekciu anomálií je zvyčajne hlavným problémom
- dáta častokrát obsahujú šum, ktorý býva podobný skutočným anomáliam, a preto je niekedy ťažké ho odlíšiť a odstrániť

Aj vďaka vyššie spomínaným výzvam a problémom (v tej najobecnejšej forme) nie je detekcia anomálií jednoduchou záležitosťou. Väčšina techník, ktoré v oblasti detekcie anomálií existujú rieši špecifickú formuláciu daného problému, ktorá sa odvíja od rôznych faktorov ako je napr. povaha vstupných údajov (vo všeobecnosti súbor inštancií údajov - objekt, vektor, udalosť atď.). Každú dátovú inštanciu je možné popísať pomocou súboru atribútov - premenná, pole atď. a atribúty môžu byť rôznych typov - binárne, kategorické alebo kontinuálne). Medzi ďalšie faktory patrí (ne)dostupnosť označených resp. oštítkovaných (z angl. *labeled*) dát (štítky označujú, či je daná inštancia normálna alebo anomálna), kategória anomálií (bodové, podmienené a skupinové) atď. Tieto jednotlivé faktory sa určujú podľa aplikačnej domény, v ktorej sa anomálie detegujú.

V nasledujúcich sekciách sa budeme venovať detekcií anomálií (obecne) a konkrétnym algoritmom, ktoré budú použité a aplikované na reálna dáta (záznamy) v praktickej časti tejto diplomovej práce.

4.3 Detekcia anomálií

Chandola, Banerjee a Kumar [107] definujú detekciu anomálií ako:

„*Detekcia anomálií je spojená s problémom nájdenia vzorov v dátach, ktoré nezodpovedajú očakávanému správaniu.*“[41]

Zo všetkých pojmov, ktoré sa v oblasti detekcie anomálií používajú sú **anomálie** a **odľahlé hodnoty** termíny, ktoré sa podľa Chandola, Banerjee a Kumar [107] používajú najčastejšie v súvislosti s detekciou anomálií. Pri detekcií anomálií je veľmi dôležité rozlišovať medzi dvoma pojmami, a to síce medzi pojmom **detekcia odľahlých hodnôt** (z angl. *outlier detection*), ktorá je taktiež známa ako tzv. *unsupervised* detekcia anomálií a **detekcia novosti** (z angl. *novelty detection*), ktorá je taktiež známa ako *semi-supervised* detekcia anomálií. Úlohou detekcie novosti je naučiť sa reprezentáciu „bežných“ (normálnych) vzorov s použitím dát, ktoré neobsahujú žiadne odľahlé hodnoty, a teda pri tréningu daného modelu sú použité len normálne dáta (dáta, ktoré neobsahujú odľahlé/anomálne hodnoty). Na druhej strane pri detekcií odľahlých hodnôt sa pri „učení“ používajú dáta, ktoré obsahujú ako normálne tak aj odľahlé/anomálne hodnoty. Na záver je potrebné dodať, že oba pojmy sú formami detekcie anomálie a budeme sa im venovať v tejto sekcií (pozri nižšie).

[41] Preložené autorom tejto diplomovej práce.

Dôležitosť detekcie anomálií je spôsobená tým, že ak sa už anomália v danom systéme nachádza, tak to väčšinou môže znamenať dôležitú a častokrát kritickú informáciu (v rôznych aplikačných doménach) ako je napr. to, že neobvyklý prenos na sieti by mohol znamenať exfiltráciu dát apod.

V oblasti detekcie anomálií nie je podľa Chandola, Banerjee a Kumar [107] získanie oštítkovaných (z angl. *labeled*) dát/inštancií, kde navyše platí, že sú tieto dáta oštítkované správne a obsahujú všetky typy anomálneho správania, jednoduchou a lacnou záležitosťou, pretože štítkovanie sa častokrát robí manuálne na základe doménovej expertízy daného odborníka. Získať dátovú sadu, ktorá obsahuje štítky pre normálne dáta/inštancie je ľahšie ako získať dátovú sadu, ktorá obsahuje štítky pre anomálne dáta/inštancie a to už len vďaka tomu, že anomálie majú často dynamický charakter, a je ťažké získať takú dátovú sadu, ktorá by pokrývala každé možné anomálne správanie, ktoré sa v danej dátovej sade môže vyskytnúť.

Na základe toho v akom rozsahu sú oštítkované/označené (z angl. *labeled*) dáta (normálne a anomálne inštancie) sa podľa Chandola, Banerjee a Kumar [107] techniky/metódy detekcie anomálií (pozri obr. 4.3) delia (podľa toho v akom móde fungujú) na:

1. *supervised* – predpokladá dostupnosť oštítkovanej (označenej) trénovacej dátovej sady pre normálne a anomálne inštancie/triedy

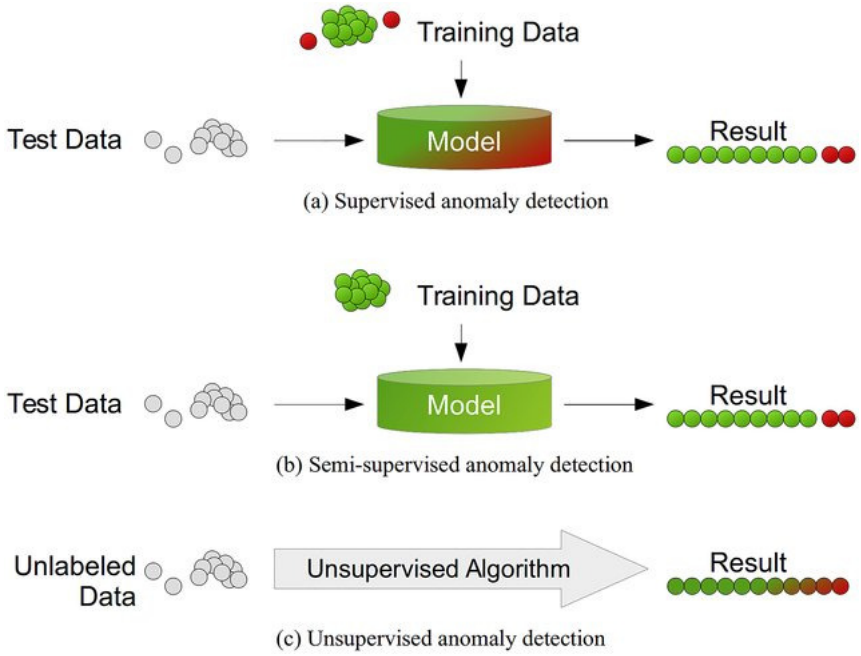
2. *unsupervised* – predpokladá, že oštítkovaná (označená) trénovacia sada nie je vôbec k dispozícií

3. *semi-supervised* – predpokladá, že je k dispozícií trénovacia sada, ktorá má štítky len pre normálne inštancie/triedy

Anomálie je taktiež možné detegovať pomocou tzv. statických pravidiel, ktoré sú vytvárané doménovými expertmi (analytikmi). Častokrát sú, ale tieto pravidlá zložité a príliš špecifické na to, aby sa dali použiť v „reálnom“ svete. Práve v tomto prípade dokáže pomôcť strojové učenie, a preto budeme v nasledujúcich sekciách rozoberať techniky detekcie anomálií založené na strojovom učení, ďalším technikám/metódam detekcie anomálií ako sú napr. štatistické metódy detekcie anomálií (na základe testovania modelu/hypotézy) sa v tejto diplomovej práci venovať nebudeme.

4.3.1 Techniky detekcie anomálií založené na strojovom učení

V tejto podsekcií sa budeme venovať technikám a metódam detekcie anomálií, a to konkrétne *supervised*, *unsupervised* a *semi-supervised* prístupu. Na konci tejto podsekcie porovnáme jednotlivé prístupy medzi sebou a vyberiem ten, ktorý použijeme pri implementácií praktickej časti tejto diplomovej práce.



Obr. 4.3: Rôzne techniky detekcie anomálií [110]

4.3.1.1 Supervised

Ako už bolo vyššie spomínané táto technika predpokladá dostupnosť trénovacej dátovej sady, ktorá má oštítkované jednotlivé inštancie ako pre normálnu tak aj pre anomálnu triedu. Po tom, čo je celá trénovacia dátová sada oštítkovaná sa problém detekcie anomálií redukuje na klasifikačný problém (vytvorenie prediktívneho modelu pre normálne a anomálne triedy. Následne sa inštancie (tie ktoré sa na tvorbe modelu nepodieľali) zaradzujú/klasifikujú (na základe vytvoreného modelu – klasifikátora) do jednotlivých tried. Používajú sa klasifikačné metódy ako sú napr. SVM, Decision Trees, ale aj neurónové siete atď.

S týmto prístupom, ale existuje viacero problémov:

- získanie oštítkovanej/označenej dátovej sady (najmä pre triedu anomálií) je zvyčajne náročné (navyše získanie presných a reprezentatívnych štítkov je niekedy až nemožné)

- slabá robustnosť – modely bojujú s problémom overfitting[42], pretože síce sa pri trénovaní/vytváraní modelu naučili rozoznávať určité vzory (z angl. *patterns*), ale už pri testovaní bojujú s novými typmi anomálií, ktoré pri tréningu nevideli

[42]Nadmerné prispôsobenie sa trénovacím dátam.

4.3.1.2 Unsupervised

Táto technika (ako už bolo vyššie spomenuté) nevyžaduje oštítkovanú trénovaciu dátovu sadu, a preto je podľa Chandola, Banerjee a Kumar [107] najrozšírenejšou technikou v oblasti detekcie anomálií. Táto *unsupervised* technika detekcie anomálií je založená na predpoklade, že v testovacích dátach sa normálne inštancie vyskytujú oveľa častejšie (vytvárajú určitý druh rozpoznateľného vzoru) ako anomálne inštancie. Ak tento predpoklad nie je pravdivý/splnený, tak dochádza k vysokému počtu falošných poplachov. Medzi metódy *unsupervised techniky* detekcie anomálií patria metódy založené na klastroch (z angl. *clustering-based methods*) ako je napr. DBSCAN atď.

4.3.1.3 Semi-Supervised

Techniky detekcie anomálií, ktoré fungujú v *semi-supervised* móde/režime predpokladajú (ako už bolo vyššie spomenuté), že je k dispozícií trénovacia sada, ktorá má štítky pre normálne inštancie/triedy. Typický prístup, ktorý sa v tomto prípade používa je ten, že sa vytvorí model pre triedu, ktorá zodpovedá normálnemu správaniu a následne sa tento model použije/aplikuje na testovaciu sadu, ktorá obsahuje ako normálne tak aj anomálne inštancie na to, aby sa identifikovali anomálie v tejto testovacej sade.

Na záver je potrebné dodať, že väčšina *semi-supervised* techník môže byť prispôsobená tak, aby pracovala v *unsupervised* režime a to tak, že v trénovacom procese sa použijú vzorky neoznačených (neoštítkovaných) dát. Tento prístup, ale predpokladá, že testovacia množina obsahuje veľmi málo anomálií a že model, ktorý bol vytvorený počas tréningu je odolný (robustný) voči týmto anomáliám. Týmto spôsobom sa v podstate dá testovať robustnosť daného modelu, resp. algoritmu a to tak, že do trénovacej sady (ktorá obsahuje len inštancie, ktoré patria do normálnej triedy) sa zámerne „zamiešajú“ inštancie, ktoré patria do anomálnej triedy.

4.3.1.4 Porovnanie techník

Medzi výhodu *supervised* prístupu, resp. tejto techniky patrí to, že testovacia fáza je v porovnaní s trénovacou fázou pomerne rýchla, pretože každá testovacia inštancia sa porovnáva už s predpočítaným modelom, čo platí aj v prípade *semi-supervised* prístupu. Medzi hlavnú nevýhodu *supervised* prístupu patrí to, že výkonnosť modelu/modelov závisí od dostupnosti správne oštítkovanej trénovacej dátovej sady, ktorá obsahuje štítky ako pre normálnu, tak aj pre anomálnu triedu. V prípade *semi-supervised* prístupu je predpokladom dostupnosť štítkov len pre normálnu triedu a v prípade *unsupervised* prístupu nie je oštítkovaná trénovacia sada vôbec k dispozícií, čo je jedna z hlavných výhod tejto *unsupervised* techniky (kedže získanie oštítkovanej dátovej sady býva náročné) . Medzi ďalšie výhody *unsupervised* techniky patrí napr. to, že je menej náchylná na tzv. *overfitting* narozdiel od *supervised* techniky. Nevýhodou

tejto *unsupervised* techniky je to, že zvyčajne nie je až tak účinná ako *supervised* technika a za určitých podmienok môže dochádzať k vysokému počtu falošných poplachov. Z vyššie uvedeného vyplýva, že jedným z hlavným faktorom pri výbere jednej z techník pri detekcií anomálií je práve (ne)dostupnosť oštítkovanej/označenej dátovej sady.

Autorovi tejto diplomovej práci boli poskytnuté oštítkované dátové sady (z reálneho prostredia) ako pre normálnu tak aj pre anomálnu triedu (pre viac informácií pozri sekciu 6.2) na to, aby bolo možné vyhodnotiť a porovnať jednotlivé algoritmy strojového učenia. V tomto prípade sa naskytla možnosť použiť *supervised* prístup, ale tento prístup nebude použitý hlavne kvôli tomu, že použitie *supervised* techniky je pre oblasť detekcie anomálií atypické, čo potvrdzujú aj Chandola, Banerjee a Kumar [107] a to tým, že sa touto technikou vo svojom prieskume hlbšie nezaoberajú. Táto diplomová práca sa preto bude ďalej venovať už len *semi-supervised* technike detekcie anomálií (detaily v kapitole 5), ktorá je tiež známa pod pojmom detekcia novosti (z angl. *novelty detection*)

Popis použitých „klasických“ algoritmov strojového učenia (k-NN, Local Outlier Factor a Isolation Forest), ale aj algoritmu z oblasti *deep learning* (neurónových sieti) Autoenkódera je možné nájsť v sekcií *Použité algoritmy* (pozri sekciu nižšie 4.4).

4.3.2 Výstup detekcie anomálií

Neoddeliteľnou a veľmi dôležitou súčasťou detekcie anomálií je interpretácia výstupu jednotlivých metód, resp. techník detekcie anomálií. Výstupy, ktoré sú produkované jednotlivými metódami môžu byť dvoch typov:

- **štítky** – každej testovacej inštancií je pridelená jedna z dvoch možných hodnôt (hodnota/vzorka je považovaná za normálnu alebo za anomálnu)

- **skóre** – každej testovacej inštancií je pridelené anomálne skóre (spôsoby pomocou ktorých sa anomálne skóre počíta sa líšia v závislosti na použitých algoritmoch) na základe ktorého je možné rozhodnúť ako veľmi je, či nie je daná inštancia anomálna (na rozhodnutie sa môže použiť nejaká prahová hodnota (z angl. *threshold*) alebo sa vyberie množina najanomálnejších (najabnormálnejších) inštancií)

Z vyššie uvedeného delenia vyplýva to, že analytik musí nielen správne nastaviť jednotlivé parametre jednotlivých modelov/techník (algoritmov) detekcie anomálií, ale aj zvoliť vhodnú prahovú hodnotu (z angl. *threshold*) a následne správne interpretovať výstup na základe ktorého je potom možné urobiť ďalšie dôležité rozhodnutia.

4.4 Použité algoritmy

V praktickej časti diplomovej práci budú použité nižšie uvedené algoritmy v tzv. *semi-supervised* režime (móde) a to tak, že na tréning (vytvorenie modelu) budú použité výhradne inštancie, ktoré patria do normálnej triedy a na vyhodnotenie modelu, resp. na testovanie budú použité inštancie z oboch tried (ako z normálnej tak aj z anomálnej triedy).

4.4.1 k-NN

Algoritmus k-NN (k-najbližších susedov) a jeho využitie na detekciu anomálií je založené na predpoklade, že anomálií sa v dátach nachádza oveľa menej ako normálnych (neanomálnych) dát a navyše platí, že anomálne dáta sa od tých normálnych dát výrazne líšia resp., že vzdialenosť anomálnych bodov k ich najbližším bodom/susedom bude v priemere výrazne väčšia ako u normálnych bodoch.

Knorr a Ng [111] definovali (na základe vzdialenosti) odľahlý bod ako:

„*Bod* $\boldsymbol{p}$ *v dátovej sade je odľahlým bodom vzhľadom k parametrom* $\boldsymbol{k}$ *(počet susedov daného bodu* $\boldsymbol{p}$*) a* $\boldsymbol{d}$ *(vzdialenosť), ak maximálne* $\boldsymbol{k}$ *bodov v danej dátovej sade je vo vzdialenosti* $\boldsymbol{d}$ *alebo menšej od daného bodu* $\boldsymbol{p}$“[43]

Vzdialenosť sa môže medzi dvoma bodmi počítať rôznymi spôsobmi, podľa [113] je najbežnejším spôsobom počítania (metrikou) vzdialenosti medzi dvoma bodmi štandardná Euklidovská vzdialenosť.

Ramaswamy, Rastogi a Shim [112] tvrdia, že aj napriek tomu, že vyššie spomínaná definícia má svoje výhody (jednoduchá, intuitívna a v niektorých prípadoch má aj nízke výpočtové nároky pre pomerne veľké dátové sady), tak má aj určité nedostatky:

- užívateľ musí zadať parameter d (vzdialenosť), ktorý je v niektorých prípadoch ťažko určiť
- neposkytuje hodnotenie pre odľahlé hodnoty (stupeň odľahlosti)
- počet buniek v tzv. *cell-based* algoritme rastie exponenciálne

Kvôli týmto nedostatkom Ramaswamy, Rastogi a Shim [112] definovali odľahlý bod (aj preto, že užívateľa zvyčajne zaujíma $\mathbf{n}$ bodov, ktoré sú považované za najväčšie/najodľahlejšie anomálne body) ako:

„*Vzhľadom k parametrom* $\boldsymbol{k}$ *a* $\boldsymbol{n}$*, bod* $\boldsymbol{p}$ *je odľahlým bodom, ak maximálne* $(n-1)$ *ďalších bodov v danej dátovej sade má pre* D^k *(vzdialenosť bodu* $\boldsymbol{p}$ *k jeho* $\boldsymbol{k}$*-tému najbližšiemu susedovi) vyššiu hodnotu ako* $\boldsymbol{p}$“[44]

[43] Definícia bola prebraná a preložená autorom tejto diplomovej práce zo zdroja Ramaswamy, Rastogi a Shim [112], kde sa uvádza, že táto definícia sa mierne líši od pôvodnej definície (ktorej autormi sú vyššie spomínaný autori Knorr a Ng), ale je jej ekvivalentom.

[44] Definícia bola prebraná a preložená autorom tejto diplomovej práce zo zdroja Ramaswamy, Rastogi a Shim [112].

Na rozdiel od predchádzajúcej definície je v tejto definícií detekcia odľahlého bodu založená na vzdialenosti bodu p od jeho k-tého najbližšieho suseda. Hodnota $D^k(p)$ je teda mierou, resp. ukazateľom toho, ako veľmi je alebo nie je daný bod p odľahlým bodom. Body s vyššou hodnotou $D^k(p)$ patria do redších zhlukov, a preto sú zvyčajne silnejšími odľahlými bodmi ako body, ktoré majú nižšie hodnoty $D^k(p)$ a ktoré patria do hustejších zhlukov.

Na záver je potrebné dodať, že na vzdialenosť daného bodu p k jeho k-tému najbližšiemu susedovi sa dá pozerať ako na odľahlé (anomálne) skóre na základe ktorého je možné detegovať anomálie.

4.4.2 Local Outlier Factor

Breuniq, Kriegel, Ng a Sander [114] tvrdia, že pri identifikácií odľahlých hodnôt je pre mnohé scenáre vhodnejšie každému objektu, resp. bodu priradiť tzv. stupeň odľahlosti. Tento stupeň odľahlosti je definovaný ako faktor lokálnej odľahlosti (LOF) daného objektu. Algoritmus LOF odhaduje hustotu v okolí daného bodu a porovnáva ju s hustotou v okolí najbližších susedov. Tento algoritmus je založený na predpoklade, že u normálnych bodov sú hustoty veľmi podobné, ale u anomálných bodov výrazne odlišné (nižšie).

Algoritmus LOF odhaduje hustotu v okolí bodu pomocou špeciálnej vzdialenosti označovanej ako *reachability distance*. Táto špeciálna vzdialenosť je počítaná ako maximum z dvoch hodnôt a je definovaná ako:

$$\text{reach-dist}_k(p, o) = max\{\text{k-distance}(o), d(p, o)\}, \tag{4.1}$$

kde *k-distance(o)* je vzdialenosť ku k-tému najbližšiemu susedovi bodu o a $d(p, o)$ je vzdialenosť bodu p od bodu o. Dá sa teda povedať, že ak sa bod p nachádza v blízkosti k susedov bodu o, tak výsledkom *reach-dist(p, o)* bude *k-distance(o)*, v opačnom prípade to bude skutočná vzdialenosť bodu p od bodu o, a to síce (*d(p, o)*).

Hustota v okolí daného bodu (označovaná ako lrd - *local reachability distance*) sa vypočíta ako spriemerovaná hodnota *reachability distance* ku všetkým k najbližším susedom daného bodu a výslednou hodnotou lrd je potom inverzná hodnota tejto spriemerovanej hodnoty (pozri rovnicu 4.2).

$$\text{lrd}_{MinPts}(p) = 1/\left(\frac{\sum_{o \in N_{MinPts}(p)} \text{reach-dist}_{MinPts}(p, o)}{\mid N_{MinPts}(p) \mid}\right), \tag{4.2}$$

kde $MinPts$ špecifikuje minimálny počet objektov, takže $N_{MinPts}(p)$ sa dá označiť, resp. interpretovať ako množina $MinPts$ najbližších susedov daného bodu p. Táto lrd hodnota bude vysoká pre body, ktoré ležia vo vnútri zhluku (s veľkou hustotou bodov v ich okolí) a nízka pre body, ktoré sú mimo jednotlivé zhluky.

Výsledný faktor lokálnej odľahlosti daného bodu (v našom prípade p) známy pod skratkou LOF sa vypočíta ako:

$$\mathrm{LOF}_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{\mid N_{MinPts}(p) \mid} \tag{4.3}$$

LOF je v podstate priemerný pomer hodnôt lrd susedov daného bodu p k hodnote lrd daného bodu p alebo inak povedané LOF sa počíta ako priemerná hustota v okolí najbližších susedov daného bodu delená hustotou v okolí daného bodu.

Ná základe tejto LOF hodnoty je možné urobiť nasledujúce rozhodnutie:

- $LOF(p) \approx 1$ znamená, že bod p má podobnú hustotu v okolí ako jeho susedia (normálny bod)
- $LOF(p) < 1$ znamená, že bod p má vyššiu hustotu v okolí ako jeho susedia (normálny bod)
- $LOF(p) \gg 1$ znamená, že bod p má výrazne nižšiu hustotu v okolí ako jeho susedia (odľahlý/anomálny bod)

Hodnota LOF sa dá interpetovať aj ako anomálne skóre, na základe ktorého je možné určiť či je alebo nie je daný bod v dátovej sade odľahlým (anomálnym) bodom. Pomocou algoritmu LOF je teda takisto možné detegovať anomálie.

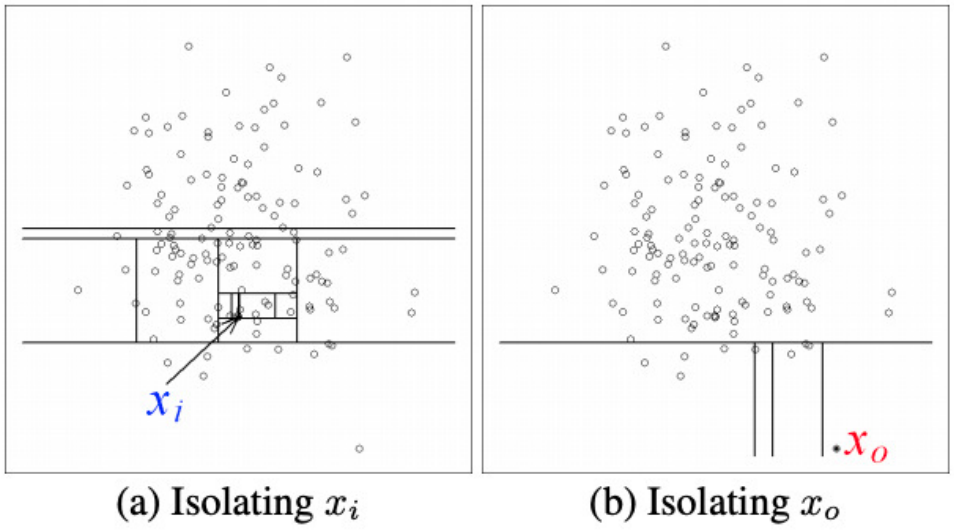
4.4.3 Isolation forest

Isolation forest [115] je založený na nasledujúcich dvoch predpokladoch:

1. počet anomálnych bodov/inštancií nachádzajúcich sa v dátach je v skutočnosti oveľa menší ako počet normálnych bodov/inštancií
2. hodnoty atribútov/príznakov jednotlivých anomálnych bodov sa výrazne líšia od hodnôt atribútov/príznakov jednotlivých normálnych bodov

Základnou myšlienkou algoritmu Isolation forest (ktorá vychádza hlavne z druhého predpokladu) je to, že zatiaľ čo normálne body je od seba možné oddeliť/izolovať len veľmi ťažko, anomálne body je možné oddeliť/izolovať od normálnych bodov pomerne jednoducho. Pri náhodnom rozdeľovaní priestoru bodov budú anomálne body oddelené od normálnych (s vyššou pravdepodobnosťou) už po pár krokoch.

Túto myšlienku je možné demonštrovať pomocou príkladu (pozri obr. 4.4), ktorý vizualizuje náhodné rozdeľovanie priestoru bodov. Normálny bod x_i vyžaduje na oddelenie/izoláciu od ostatných bodov viac oddielov/časti ako anomálny bod x_o. V tomto príklade sú jednotlivé oddiely/časti generované



Obr. 4.4: Vizualizácia náhodného rozdeľovania priestoru pre normálny a anomálny bod [115]

spôsobom, že sa náhodne vyberie atribút/príznak, a potom sa následne náhodne vyberie hodnota z rozsahu hodnôt daného atribútu/príznaku, ktorá sa použije ako rozdeľovacia hodnota na základe ktorej sa daný priestor rozdelí na dve časti a pokračuje sa rekurzívne ďalej. Vďaka tomu, že rekurzívne rozdeľovanie priestoru môže byť reprezentované stromovou štruktúrou, tak počet oddielov/častí potrebných na izoláciu konkrétneho bodu je ekvivalentný dĺžke cesty od koreňového uzla až po koncový uzol. Z toho vyplýva, že anomálne body budú mať kratšie dĺžky ciest ako normálne body.

Isolation forest je rovnako ako náhodny les (z angl. *random forest*) tvorený súborom rozhodovacích stromov[45]. Isolation forest narozdiel od náhodného lesa (ktorého výstupom, resp. výslednou predpoveďou je agregácia predpovedí jednotlivých stromov, v prípade klasifikácie to je väčšina predpovedí („hlasov“) a v prípade regresie priemer) počíta dĺžku cesty, ktorá je potrebná na izolovanie jednotlivých bodov/inštancií.

Tento algoritmus funguje v dvoch fázach:

1. z náhodne vybraných bodov/inštancií sa vytvorí definovaný počet stromových štruktúr, ktoré sú podobné binárnym vyhľadávacím stromom

2. u každého bodu/inštancie sa počíta dĺžka cesty jednotlivými vytvorenými stromami, ktorá je definovaná ako počet hran z koreňového uzla až po externý uzol (list)

[45]Rozhodovací strom [116] je množinou uzlov a hrán zaradených do hierarchickej štruktúry.

Liu, Ting a Zhou [115] definujú anomálne skóre s inštancie x ako:

$$s(x, \psi) = 2^{-\frac{E(h(x))}{c(\psi)}}, \tag{4.4}$$

kde ψ je počet vzorkov/inštancií určených pre tvorbu stromu (v zdroji [115] označovaný ako *subsampling size*), $h(x)$ je dĺžka cesty inštancie x, $E(h(x))$ je priemer $h(x)$ pre všetky stromy a $c(n)$ je priemerná dĺžka cesty neúspešného vyhľadávania v binárnom vyhľadávacom strome.

Pre každú dátovú inštanciu, resp. bod je vypočítané anomálne skóre s, na základe ktorého je možné urobiť nasledujúce rozhodnutie:

- bod/inštancia sa považuje za anomálnu, ak je jej anomálne skóre veľmi blízko hodnoty 1
- bod/inštancia sa považuje za normálnu, ak je jej anomálne skóre oveľa menšie ako hodnota 0.5
- v prípade, že u všetkých bodov/inštancií platí, že anomálne skóre $s \approx 0.5$ tak potom sa dá povedať, že v danej dátovej sade sa nenachádza žiadny anomálny bod/inštancia

Výhodou Isolation forest algoritmu je to, že má nízke pamäťové nároky a lineárnu výpočtovú zložitosť vzhľadom k počtu bodov.

V skutočnosti, z dôvodu veľkej nerovnováhy jednotlivých tried (normálnej a anomálnej), kedy platí, že anomálna trieda obsahuje oveľa menej bodov/inštancií ako tá normálna, sa priemerná hodnota anomálneho skóre naprieč celou dátovou sadou blíži k 0 a nie k 1. Cieľom analytika (experta) je experimentálne zvoliť najoptimálnejší prah (z angl. *threshold*), ktorý bude slúžiť ako hranica, ktorá bude od seba správne oddeľovať normálne a anomálne body resp. inštancie.

4.4.4 Autoenkóder

Autoenkóder [117] je typ neurónovej siete pre ktorý je typické to, že počet jeho výstupov sa rovná počtu jeho vstupov (výstupná vrstva má rovnaký počet uzlov (neurónov) ako vstupná vrstva) a pozostáva z dvoch hlavných častí:

1. kóder
2. dekóder

Následujúci text bude vychádzať zo sekcie 2.2 (*Autoencoder and anomaly detection*) od An a Cho [117], ale aj zo sekcie 3.2 (*Autoencoder*) od Fan [118].

Pre jednoduchosť budeme uvažovať najjednoduchšiu formu autoenkódera s jednou skrytou vrstvou (z angl. *hidden layers*), taktiež nazývaný ako *vanilla autoencoder*, ktorý má kóder definovaný ako (pozri rovnicu 4.5) a dekóder ako

(pozri rovnicu 4.6), kde platí, že $k, d \in \mathbb{N}$, $W \in \mathbb{R}^{k,d}$ je matica váh o rozmeroch $k \times d$, $b \in \mathbb{R}^k$ (o rozmeroch $k \times 1$) je bias kódera, $b' \in \mathbb{R}^d$ (o rozmeroch $d \times 1$) je bias dekódera, vstupný vektor $x \in \mathbb{R}^d$ o rozmeroch $d \times 1$ (kde d je počet príznakov (z angl. *features*)), σ je funkcia nelineárnej transformácie (nelineárna aktivačná funkcia[46]), $h \in \mathbb{R}^k$ (o rozmeroch $k \times 1$) a $z \in \mathbb{R}^d$ (o rozmeroch $d \times 1$).

$$h = f(x) = \sigma(Wx + b) \tag{4.5}$$

$$z \ = g(h) = \sigma(W^\top h + b') \tag{4.6}$$

$$L_{MSE}(x, z) = \|x - z\|^2 = \frac{1}{d} \sum_{j=1}^{d} (x_j - z_j)^2 \tag{4.7}$$

Kóder transformuje dáta (mapuje vstupný vektor x dimenzie d) do h (vektor označovaný ako „skrytá“ reprezentácia (z angl. *latent representation/space*)) redukovanej/zmenšenej dimenzie k, a platí, že $k < d$. Naopak dekóder vykonáva rekonštrukciu, ktorá mapuje dáta (vektor h) zo zmenšenej dimenzie k naspäť do vektora z pôvodnej dimenzie d. Na obr. 4.5 je možné vidieť architektúru najjednoduchšej formy autoenkódera[47], ale aj spôsob počítania výstupu kódera h, konkrétne h_1.

Cieľom trénovacieho procesu autoenkódera je minimalizovať stratovú funkciu (z angl. *loss function*) (pozri rovnicu 4.8).

$$L(x, g(f(x))), \tag{4.8}$$

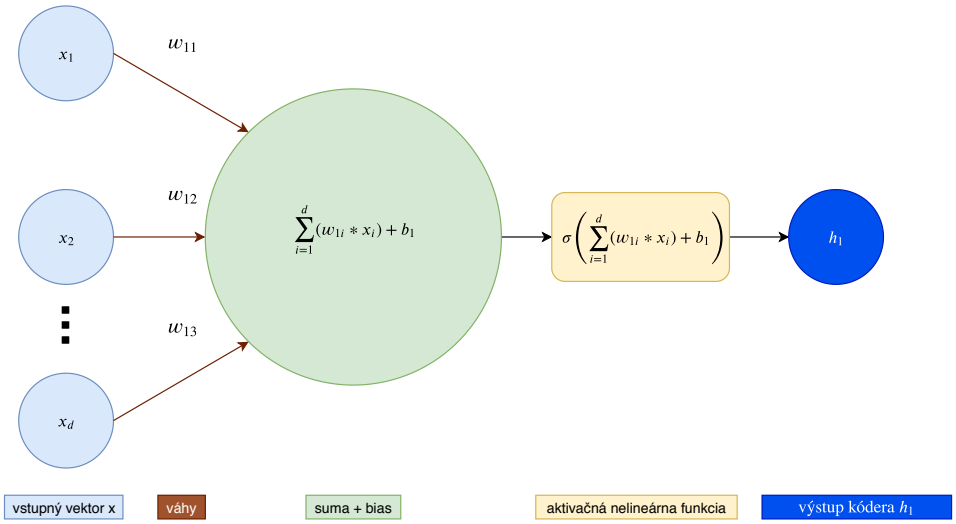
ktorá meria rekonštrukčnú chybu (chybu rekonštrukcie), teda rozdiel medzi vstupným vektorom x a rekonštruovaným vektorom z. Autoenkóder je trénovaný hľadaním optimálnych riešení pre W, b a b', ktoré minimalizujú stratovú funkciu. Na meranie rozdielu medzi vstupným vektorom x a rekonštruovaným vektorom z sa používa stredná kvadratická chyba (z angl. *mean squared error*) (pozri rovnicu 4.7). Algoritmus trénovania autoenkódera je možné nájsť nižšie (pozri alg. 1), kde f_ϕ a g_θ sú viacvrstvové neurónové siete.

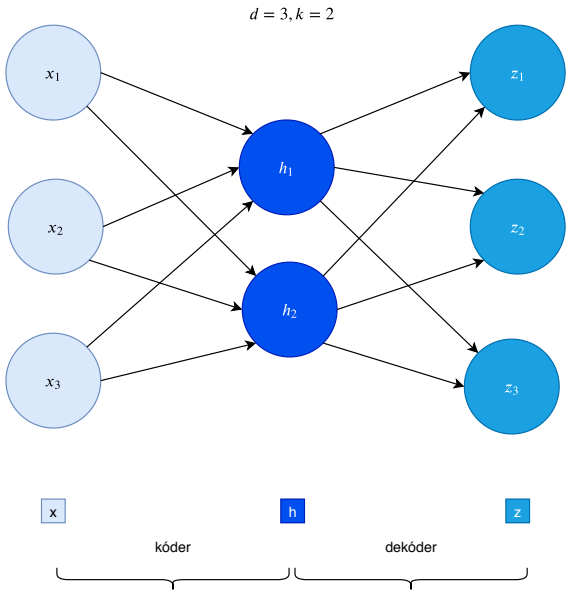
Detekcia anomálií pomocou autoenkódera je založená na *semi-supervised* technike (na trénovanie autoenkódera sa používa výhradne normálna dátová sada[48], pre bližšie informácie pozri podsekciu 4.3.1.3). Po tréningu daného autoenkódera nastáva fáza v ktorej sa rekonštrukčná chyba používa ako anomálne skóre na základe ktorého je možné detekovať anomálie, a to vďaka tomu, že autoenkóder dokáže normálne dáta (body reprezentované vektormi) rekonštruovať s malou rekonštrukčnou chybou, zatiaľ čo u anomálnych dát (bodov) to platiť nebude. Body u ktorých je rekonštrukčná chyba vysoká

[46] Príkladom môže byť hyperbolický tangens (tanh) alebo sigmoid (pozri Aggarwal [119]), ale aj ReLU (REctified Linear Unit) atď.

[47] Architektúra, nakreslená pomocou nástroja draw.io [120]

[48] Dátova sada, ktorá neobsahuje anomálie.





Obr. 4.5: Architektúra najjednoduchšej formy autoenkódera

Algorithm 1 Trénovací algoritmus – autoenkóder

VSTUP: *dátová sada* $x^{(1)}, ..., x^N$
VÝSTUP: *kóder* f_ϕ, *dekóder* g_θ
$\phi, \theta \leftarrow$ *incializuj parametre*
repeat
    $E = \sum_{i=1}^{N} \|x^{(i)} - g_\theta(f_\phi(x^{(i)}))\|^2$ *vypočítaj sumu rekonštrukčnej chyby*
    $\phi, \theta \leftarrow$ *aktualizuj parametre pomocou gradientov hodnoty E napr. pomocou metódy Stochastic Gradient Descent*
until konvergencia parametrov ϕ, θ

(v závislosti na prahovej hodnote z angl. *threshold*) budú považované za anomálie. Jednotlivé kroky algoritmu detekcie anomálií je možné nájsť nižšie (pozri alg. 2).

Algorithm 2 Algoritmus detekcie anomálií – autoenkóder

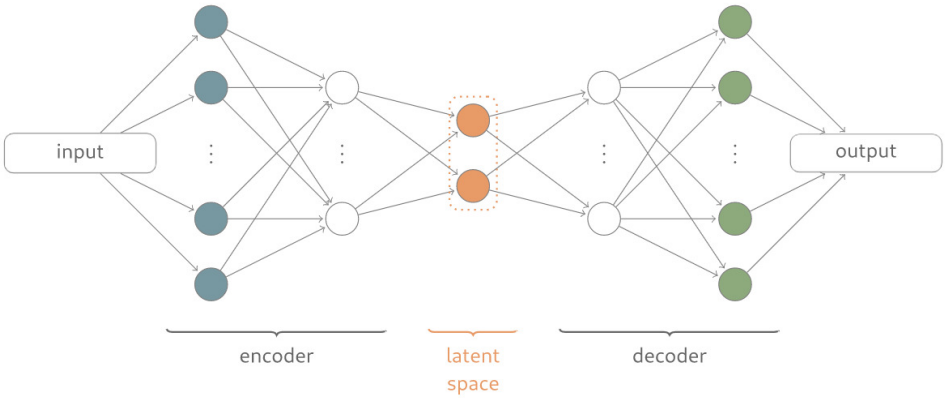
VSTUP: *normálna dátova sada* X, *anomálna dátova sada* $x^{(i)}$ *i = 1, ..., N; threshold* α
VÝSTUP: *rekonštrukčná chyba* $\|x - z\|^2$
$\phi, \theta \leftarrow$ *natrénuj autoenkóder použitím normálnej datovej sady* X
for `i=1 to N` **do**
    rekonštrukčná chyba$(i) = \|x^{(i)} - g_\theta(f_\phi(x^{(i)}))\|^2$
    if *rekonštrukčná chyba*$(i) > \alpha$ **then**
        x^i *je anomália*
    else
        x^i *nie je anomália*
    end if
end for

Na obr. 4.6 je možné vidieť príklad viacvrstvového autoenkódera, ktorý obsahuje tri skryté vrstvy (z angl. *hidden layers*).

4.5 Zhrnutie

Anomálie sú vzory v dátach, ktoré nevyhovujú presne definovanej predstave normálneho správania. Nachádzajú sa v dátach z rôznych dôvodov ako je napr. prítomnosť škodlivej aktivity, porucha systému atď., a delia sa na bodové, podmienené/kontextové, a skupinové/kolektívne anomálie.

Vďaka výzvam a problémom ako je napr. to, že hranica medzi normálnym a anomálnym (neobvyklým) správaním býva častokrát nepresná, (ne)dostupnosť oštítkovaných (z angl. *labeled*) dát, ale aj vďaka tomu, že sa útočníci snažia prispôsobovať tak, aby ich nebolo ľahké odhaliť nie je detekcia anomálií jednoduchou záležitosťou.



Obr. 4.6: Viacvrstvový (z angl. *multilayer*) autoenkóder [121]

Pri detekcií anomálií je veľmi dôležité rozlišovať medzi **detekciou odľahlých hodnôt** (z angl. *outlier detection*), ktorá je taktiež známa ako tzv. *unsupervised* technika detekcie anomálií a **detekciou novosti** (z angl. *novelty detection*), ktorá je taktiež známa ako *semi-supervised* technika detekcie anomálií. Dôležitosť detekcie anomálií je spôsobená skutočnosťou, že ak sa už anomália v danom systéme nachádza tak to väčšinou môže znamenať dôležitú a častokrát kritickú informáciu (v rôznych aplikačných doménach) ako je napr. to, že neobvyklý prenos na sieti by mohol znamenať exfiltráciu dát apod. Techniky detekcie anomálií založené na strojovom učení sa na základe toho v akom rozsahu sú dostupné oštítkované/označené dáta delia na *supervised*, *semi-supervised* a *unsupervised*. Jedným z hlavným faktorom pri výbere jednej z techník pri detekcií anomálií je práve (ne)dostupnosť oštítkovanej/označenej dátovej sady.

Neoddeliteľnou a veľmi dôležitou súčasťou detekcie anomálií je interpretácia výstupu jednotlivých metód, resp. techník detekcie anomálií. Výstupy, ktoré sú produkované jednotlivými metódami môžu byť dvoch typov, a to štítky (z angl. *labels*) alebo tzv. anomálne skóre.

Pri implementácií praktickej časti tejto diplomovej práce budeme používať *semi-supervised* techniku detekcie anomálií a využijeme/aplikujeme „klasické“ algoritmy strojového učenia medzi ktoré patrí k-NN, Local Outlier Factor Isolation Forest, ale aj algoritmus z oblasti *deep learning* (neurónových sieti), konkrétne Autoenkóder.

Kapitola 5

Návrh a implementácia

V úvode tejto kapitoly sa budeme venovať návrhu praktickej časti tejto diplomovej práce, konkrétne výslednému skriptu, ale aj krátkemu popisu toho, ako by tento skript mohli používať užívatelia. V ďalších častiach sa budeme venovať hodnoteniu výkonnosti jednotlivých modelov a popisu použitých programovacích jazykov, knižníc a nástrojov. V závere sa budeme venovať zhrnutiu celej tejto kapitoly.

5.1 Návrh

Pri návrhu výsledného skriptu, ktorého obsahom je celá tzv. *pipeline*[49] strojového učenia sme vychádzali z bodov, ktoré sú uvedené na začiatku kapitoly č. 2 v knihe od Géron [122] a medzi tieto body patrí:

1. získavanie dát
2. analýza, vizualizácia a popis poskytnutých dát
3. príprava a predspracovanie dát (extrakcia príznakov, úprava rozsahu dát apod.)
4. výber a trénovanie modelov
5. ladenie a hľadanie optimálnych hyperparametrov[50] pre jednotlivé modely
6. aplikácia a vyhodnotenie jednotlivých modelov

[49] Séria po sebe nasledujúcich krokov, ktoré zahrňujú v prípade strojového učenia získavanie údajov ich spracovanie, testovanie algoritmov atp.

[50] Parametre, ktorých hodnoty sú nastavené pred samotným začiatkom procesu trénovania/učenia.

Spôsob iterácie, resp. vývoja celého nášho projektu (praktickej časti diplomovej práce) vychádza z vyššie spomínaných bodov a pozostáva z nasledujúcich krokov:

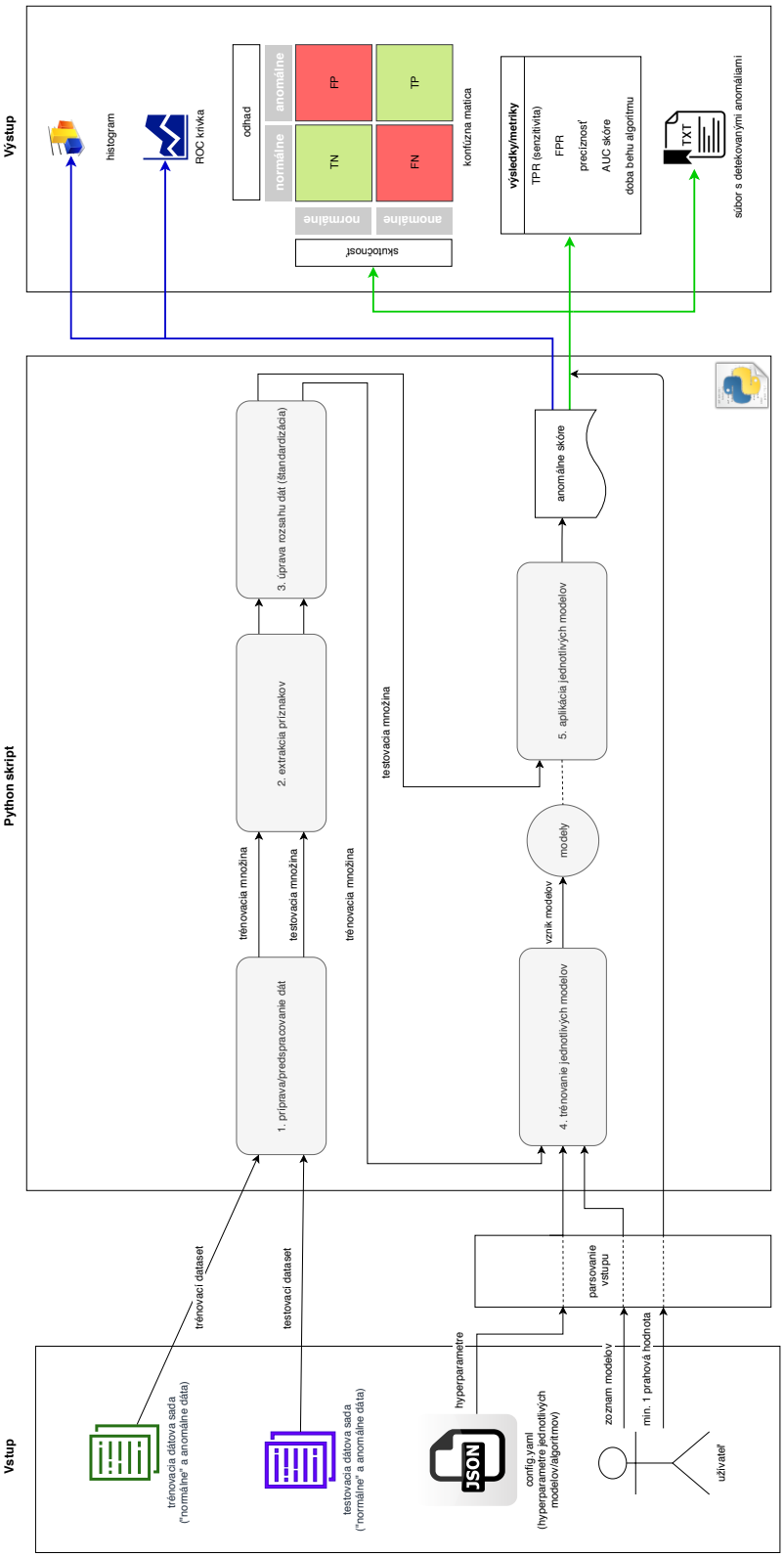
1. Jupyter notebook[51]
   a) analýza a vizualizácia poskytnutých dát (konkrétne trénovacej množiny)
   b) príprava a predspracovanie dát (trénovacej množiny), ktoré pozostáva z extrakcie príznakov, úpravy rozsahu dát apod.
   c) výber modelov (k-NN, Local Outlier Factor, Isolation forest a Autoenkóder, pozri sekciu 4.4) a ich trénovanie na konkrétnom rozdelení trénovacej množiny s použitím predvolených hyperparametrov
   d) aplikácia a vyhodnotenie jednotlivých modelov na časti trénovacej množiny (experimenty) pomocou hodnotenia výkonnosti jednotlivých modelov (detaily v sekcií 5.2)
2. Jupyter notebook
   a) ladenie a hľadanie „optimálnych“ hyperparametrov pre jednotlivé modely na trénovacej množine pomocou krížovej validácie a algoritmu RandomSearch (detaily v podsekcií 6.3.3)
   b) analýza a vizualizácia poskytnutých dát (konkrétne testovacej množiny)
3. Python[52] skript – výsledný skript, ktorý vychádza z predchádzajúcich dvoch Jupyter notebookov, ktoré na seba nadväzujú, s tým rozdielom, že obsahom výsledného skriptu nie je analýzy a vizualizácia poskytnutých dát, a ani proces ladenia a hľadania „optimálnych“ hyperparametrov pre jednotlivé modely, ale len výstup tohto procesu („optimálne“ hyperparametre)

Z vyššie uvedených bodov vyplýva, že tzv. softvérové prototypovanie[53] prebiehalo v jednotlivých Jupyter notebookoch, ktoré na seba nadväzujú. Následne bol tento kód (doplnený o konfiguračný súbor, parsovanie vstupu a vytvorenie súboru s detekovanými anomáliami) zreplikovaný do výsledného (finálneho) skriptu pre potenciálnu integráciu tohto skriptu/modelu s iným nástrojom, resp. softvérom. Jednotlivé Jupyter notebooky a výsledný finálny skript je možné nájsť v prílohe tejto diplomovej práce, v priečinku *impl* (pozri dodatok D). Užívateľskú príručku je možné nájsť v dodatku C.

[51] Krátky popis nástroja Jupyter notebook je možné nájsť v podsekcií 5.3.4.

[52] Krátky popis programovacieho jazyka Python je možné nájsť v podsekcií 5.3.1.

[53] Vytváranie prototypov softvérových aplikácií, tj. neúplných verzií softvérového programu.



Obr. 5.1: Architektúra skriptu

Na obr. 5.1 je možné vidieť architektúru celého výsledného skriptu, ktorá sa skladá z nasledujúcich troch častí:

1. vstup, ktorý sa skladá z:
   a) trénovacej dátovej sady
   b) testovacej dátovej sady
   c) konfiguračného súboru vo formáte YAML, ktorý obsahuje množinu hyperparametrov pre jednotlivé modely (ak nie je tento súbor definovaný na vstupe tak sa použije množina hyperparametrov, ktorá sa nachádza priamo v kóde)
   d) vstupu od používateľa[54]:
      i. zoznam modelov, ktoré sa budú trénovať a následne vyhodnocovať
      ii. minimálne jedna prahová hodnota (z angl. *threshold*) (detaily v podsekcií 5.2.1)
2. hlavná časť (Python skript) – vychádza z oboch spomínaných Jupyter notebookov
3. výstup, ktorý sa skladá z:
   a) histogramu
   b) výsledkov/metrík, ktoré pozostávajú z TPR (senzitivita (z angl. *recall/sensitivity*)), FPR, precíznosť (z angl. *precision*), AUC skóre (detaily v podsekcií 5.2), ale aj z doby behu algoritmu
   c) konfúznej matice (detaily v podsekcií 5.2.1)
   d) ROC krivky (detaily v podsekcií 5.2.2)
   e) súboru, ktorý obsahuje detegované anomálie

Aplikáciu jednotlivých bodov, resp. vývoj a realizáciu celej tejto tzv. *pipeline* strojového učenia aplikovanú na konkrétny druh záznamov (*web proxy* záznamov), ktorej výstupom je výsledný skript je možné nájsť v nasledujúcej kapitole 6 aj s podrobným popisom jednotlivých krokov. Pri návrhu a architektúre výsledného skriptu sme mysleli aj na používateľov, ktorí by mohli tento výsledný skript používať. Predstava je taká, že v budúcnosti (po ďalších iteráciách tohto projektu) by tento skript mohol byť súčasťou (mohol byť integrovaný do) rôznych SIEM systémov ako programový modul alebo by prípadne mohol fungovať samostatne. Na vstup tento skript dostane bezpečnostné auditné záznamy (v súčasnosti *web proxy* záznamy) spolu s ďalšími

[54]Súčasťou vstupu od používateľa je aj argument, ktorý indikuje to, či je alebo nie je na vstupe (k dispozícií) oštítkovaná dátová sada (detaily v užívateľskej príručke v dodatku C).

vstupmi od užívateľa a na základe toho, ale aj ďalších informácií je jeho výstupom súbor s anomáliami (v našom prípade web proxy záznamy, ktoré tento skript, resp. algoritmy strojového učenia považujú za anomálne, u ktorých je predpoklad, že budú aj škodlivé). Následne by bezpečnostní experti, ktorých je už aj tak v malých a stredných firmách málo mohli použiť tento súbor ako ďalší zdroj informácií, ktorý im môže pomôcť pri analýze záznamov, ale aj pri rozhodovaní a riešení rôznych bezpečnostných incidentov. Cieľom nie je bezpečnostných expertov nahradiť, ale priniesť im ďalší zdroj informácií na základe ktorých by sa mohli lepšie rozhodovať a dokázalo im to zjednodušiť ich prácu. Z pohľadu GDPR je tento postup, resp. aplikácia strojového učenia (v prípade, že záznamy obsahujú osobné údaje) v súlade s GDPR článkom 6, pretože sa uplatňuje jedna z podmienok (detaily v podsekcií 3.2.1), ktorá hovorí o tom, že spracúvanie, ktorého sa aplikácia strojového učenia týka je zákonné, ak je nevyhnutné na účely oprávnených záujmov, ktoré sleduje prevádzkovateľ alebo tretia strana, čo je v prípade detekcie anomálií a konkrétne detekcie prítomnosti škodlivého softvéru, „neobvyklého“ správania sa pri prehľadávaní stránok apod. (pozri tabuľku 3.3) splnené.

Druhou možnosťou ako by sa tento skript mohol využívať pri detekcií anomálií by mohol byť ten, že by tento skript plnil funkciu „automatizovaného“ filtra, ktorý na vstupe prijíma všetky bezpečnostné auditné záznamy (v našom prípade web proxy záznamy) a jeho výstupom je súbor s detegovanými anomáliami (web proxy záznamy u ktorých vzniklo podozrenie žeby mohli byť škodlivé), avšak ich počet by bol nižší ako na vstupe. Následne by tento „zredukovaný“ výstup (počet záznamov určených k ďalšiemu spracovaniu by bol menší) bol vstupom do ďalších systémov, resp. nástrojov určených k detailnejšej analýze bezpečnostných auditných záznamov.

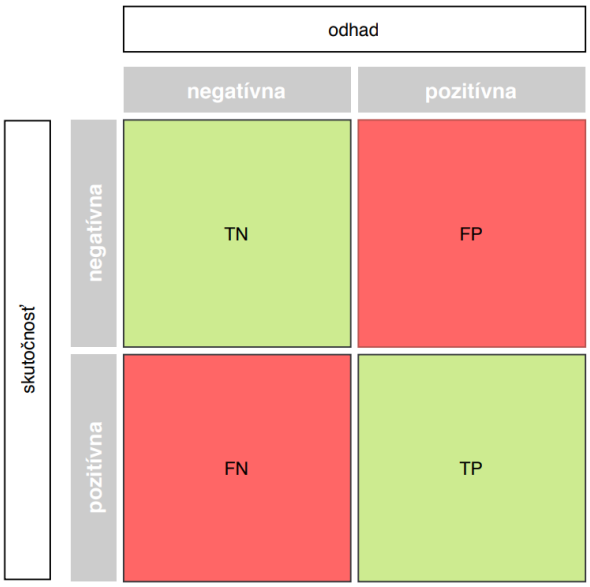
Na záver tejto sekcie je potrebné dodať, že jednotlivé kroky, resp. architektúra skriptu je navrhnutá tak, aby ju bolo možné aplikovať (po ďalších iteráciách/vývoji/zmenách) na ďalšie iné typy/druhy záznamov napr. NetFlow, systémové záznamy apod. (pozri tabuľku 3.3). Testovací režim v produkčnom prostredí a zbieranie spätnej väzby od užívateľov nie je súčasťou tejto práce, avšak bolo by to ďalším logickým krokom po tom, čo sme vytvorili a otestovali tento prototyp na novej dátovej sade.

5.2 Hodnotenie výkonnosti jednotlivých modelov

V tejto sekcií si ukážeme ako zistiť, resp. vyjadriť a vyhodnotiť kvalitu, a výkonnosť jednotlivých modelov pomocou rôznych nástrojov a metrík.

5.2.1 Konfúzna matica

Jednotlivé modely a ich výkonnosť budeme hodnotiť pomocou rôznych metrík odvodených z tzv. konfúznej matice (z angl. *confusion matrix*). Táto konfúzna matica sumarizuje výsledok klasifikácie klasifikátora a má riadky indexované



Obr. 5.2: Konfúzna matica

podľa tried výstupnej veličiny (skutočnosť) a stĺpce podľa tried, ktoré daný model predpovedal (odhad/predikcia). Na obr. 5.2 je možné vidieť príklad konfúznej matice pre binárny klasifikačný problém a obsahom tejto matice sú nasledujúce informácie:

1. TN (správne negatívny) – počet odhadov správne zaradených do negatívnej triedy
2. TP (správne pozitívny) – počet odhadov správne zaradených do pozitívnej triedy
3. FP (nesprávne pozitívny) – počet odhadov nesprávne zaradených do pozitívnej triedy (chyba prvého druhu)
4. FN (nesprávne negatívny) – počet odhadov nesprávne zaradených do negatívnej triedy (chyba druhého druhu)

Na základe týchto informácií sa dajú vypočítať rôzne metriky, my ale uvedieme len tie, ktoré budeme používať v ďalších častiach tejto diplomovej práce. Medzi metriky používané v ďalších častiach tejto diplomovej práce patrí:

- TPR – táto metrika je pomerom počtu vzoriek označených za správne pozitívne (TP) k počtu správne pozitívnych (TP) a nesprávne negatívnych (FN), ktorá hovorí o schopnosti modelu správne odhadnúť (klasifikovať) všetky pozitívne vzorky ako pozitívne (koľko zo všetkých pozitívnych vzoriek bolo aj modelom klasifikovaných ako pozitívnych)

$$TPR\ (senzitivita) = \frac{TP}{TP + FN} \tag{5.1}$$

- FPR – táto metrika je pomerom počtu vzoriek označených za nesprávne pozitívne (FP) k počtu nesprávne pozitívnych (FP) a správne negatívnych (TN), ktorá hovorí o neschopnosti modelu správne klasifikovať negatívne vzorky

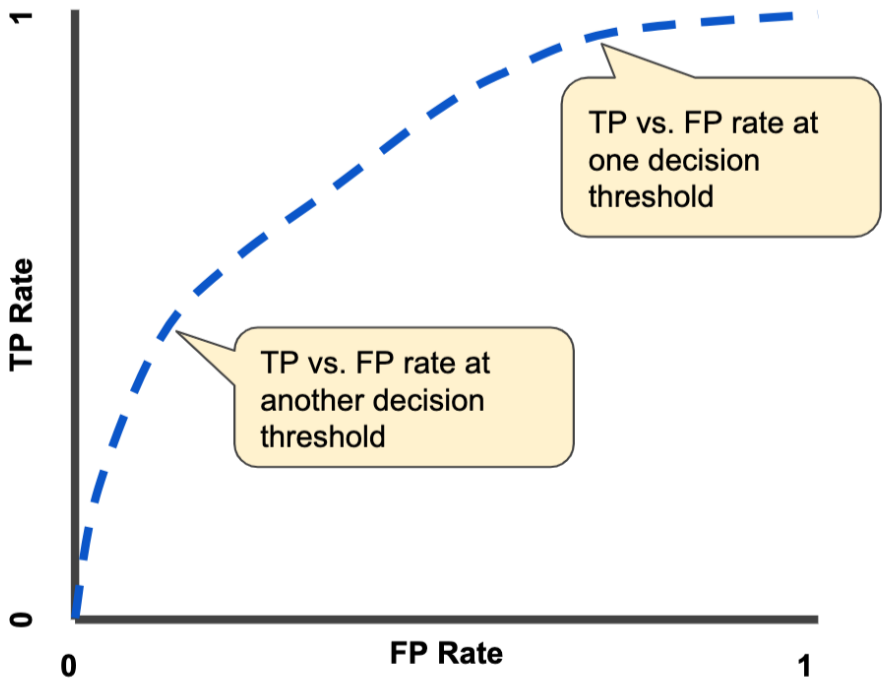
$$FPR = \frac{FP}{FP + TN} \tag{5.2}$$

- precíznosť – táto metrika je pomerom počtu vzoriek označených za správne pozitívne (TP) k počtu správne pozitívnych (TP) a nesprávne pozitívnych (FP), ktorá hovorí o schopnosti modelu správne klasifikovať pozitívne vzorky (koľko z pozitívnych vzoriek (klasifikovaných modelom) patrí aj v skutočnosti do pozitívnej triedy)

$$precíznosť = \frac{TP}{TP + FP} \tag{5.3}$$

Keďže pri našej implementácií a realizácií detekcie anomálií pomocou jednotlivých modelov strojového učenia nie sú výstupom detekcie anomálií štítky (z angl. *labels*), ale anomálne skóre, musíme nájsť a zvoliť spôsob, ktorým dokážeme určiť to, či je daná inštancia normálna alebo anomálna, a až následne vytvoriť (na základe tejto informácie) vyššie spomínanú konfúznu maticu. Cieľom je určiť najoptimálnejší prah resp. prahovú hodnotu (z angl. *threshold*), ktorá bude slúžiť ako hranica, ktorá bude od seba (správne) oddeľovať normálne a anomálne inštancie. Na to použijeme štatisticky nástroj nazývaný kvantil, resp. v našom prípade presnejšie percentil.

Po tom, čo daný model resp. algoritmus strojového učenia priradí každej jednej inštancii anomálne skóre určíme/zvolíme percentil, resp. rôzne percentily a vypočítame hodnoty týchto percentilov, ktoré budú slúžiť ako prahové hodnoty na základe ktorých vytvoríme konfúzne matice (hodnoty nižšie ako táto prahová hodnota budú označené ako normálne a hodnoty, ktoré budú vyššie ako táto prahová hodnota budú označené ako anomálne), pomocou ktorých budeme vyhodnocovať jednotlivé modely pre rôzne prahové hodnoty, resp. pre rôzne hodnoty percentilov (pozri sekciu 6.4).



Obr. 5.3: Ukážka ROC krivky [123]

5.2.2 ROC krivka

Na hodnotenie výkonnosti jednotlivých modelov budeme používať aj ROC krivku [123] (a z nej vypočítané AUC skóre), ktorá sa používa na hodnotenie binárneho klasifikátora (v prípade detekcie anomálií normálne vs. anomálne vzorky). ROC krivka je graf, ktorý znázorňuje výkon daného klasifikačného modelu pri všetkých klasifikačných prahoch (z angl. *threshold*) a je vyjadrením vzťahu medzi TPR (na osi y) a FPR (na osi x), príklad tejto ROC krivky je možné nájsť na obr. 5.3. Pri postupnom znižovaní klasifikačného prahu dochádza k tomu, že sa zvyšuje TPR alebo FPR, alebo oba hodnoty naraz.

Obsah plochy pod ROC krivkou (AUC) je dôležitá metrika, ktorú budeme používať v ďalších častiach tejto práce na vyhodnotenie výkonnosti jednotlivých modelov. Na základe AUC je možné sumarizovať výkonnosť daného modelu do jedného čísla. Hodnota AUC je pravdepodobnosť s akou je daný model schopný správne klasifikovať danú vzorku, resp. je vyjadrením toho, ako dokáže model rozlišovať medzi jednotlivými triedami.

5.3 Použité programovacie jazyky, knižnice a nástroje

V nasledujúcich sekciách sa budeme venovať popisu programovacieho jazyka Python, ale aj popisu ďalších dôležitých nástrojov a knižníc použitých v tejto diplomovej práci.

5.3.1 Python

Python [124] je zrozumiteľný, výkonný, objektovo orientovaný a dynamicky interpretovaný programovací jazyk, ktorý vyniká svojou jednoduchou syntaxou. Tento programovací jazyk bol vytvorený Guido Van Rossum v roku 1991. Okrem typovej kontroly podporuje tento jazyk rôzne programovacie paradigma od objektovo orientovaného, imperatívneho, procedurálneho, či funkcionálneho. Python je možné takisto voľne upravovať a znova distribuovať, pretože už od začiatku je vyvíjaný ako slobodný *open-source* projekt. Jednou z jeho ďalších výhod je jednoduchá dostupnosť veľkého množstva balíčkov, ktoré existujú pre rôzne platformy (Windows, Mac OS X, Linux a Unix), a ich jednoduchá inštalácia pomocou nástroja *pip*[55], ktorý je štandardnou súčasťou inštalácie. Vyššie spomínané funkcie a vlastnosti predurčujú tento programovací jazyk k širokému uplatneniu a použitiu v krátkych tzv. *snippet*[56], v rozsiahlejších programoch a vo webových aplikáciách, ale aj na použitie v oblasti strojového učenia (napr. známa knižnica Scikit-learn). Tieto slová podporuje aj webová služba GitHub[57], ktorá zverejnila The State of the Octoverse (pozri [125]), v ktorej hovorí o tom, že v rebríčku popularity programovacích jazykov v roku 2019 predbehol Python Javu a je na druhom mieste po JavaScripte, podľa počtu prispievateľov do jednotlivých repozitárov. Podľa tzv. TIOBE Indexu (pozri [126]), ktorý je ukazovateľom popularity programovacích jazykov sa Python umiestňuje takisto na popredných priečkach.

V tejto diplomovej práci bol použitý programovací jazyk Python verzie 3.7 na implementáciu praktickej časti tejto diplomovej práce. Najdôležitejšie Python knižnice, ktoré boli v tejto práci použité sú popísané v podsekciách nižšie.

5.3.1.1 SciPy

Scipy [127] je ekosystém programovacieho jazyka Python pozostávajúci z nasledujúcich *open-source* knižníc:

- NumPy – určená pre vedcov a analytikov, ktorá mimo iného definuje typ pre n-rozmerné homogénne pole (najčastejšie čísel) a API pre prácu s takýmto poľom atď.

[55] Správca balíčkov pre moduly programovacieho jazyka Python.
[56] Malý znovu použiteľný kus programového kódu.
[57] Webová služba, ktorá podporuje vývoj softvéru pomocou verzovacieho nástroje Git.

- IPython – poskytuje bohatú architektúru pre interaktívne výpočty, a to napr. výkonný interaktívny *shell*, Python *kernel* pre Jupyter notebook atď.
- SciPy – poskytuje mnoho užívateľsky prívetivých a efektívnych nástrojov určených pre rôzne matematické oblasti ako je napr. štatistika, optimalizácia, lineárna algebra atď.
- Sympy – určená pre symbolickú matematiku
- Matplotlib – určená na vykresľovanie 2D grafov
- pandas – poskytuje vysokovýkonné a ľahko použiteľné dátové štruktúry a nástroje na analýzu dát

V tejto diplomovej práci bola knižnica pandas použitá hlavne v spojení s dátovou štruktúrou DataFrame, ktorá bola využívaná na ukladanie a prácu s danými dátovými sadami, knižnica matplotlib na vykresľovanie 2D grafov (ROC krivky, ale aj konfúznej matice), ale aj knižnica numpy.

5.3.1.2 Seaborn

Seaborn [128] je knižnica, ktorá je určená na vizualizáciu dát, resp. na tvorbu štatistickej grafiky v programovacom jazyku Python. Táto knižnica je založená na knižnici Matplotlib a je úzko integrovaná s dátovými štruktúrami z knižnice pandas. Poskytuje vysokoúrovňové rozhranie na vykresľovanie atraktívnych a informatívnych štatistických grafov.

V tejto diplomovej práci bola knižnica seaborn použitá primárne na vizualizáciu distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (histogram).

5.3.1.3 Scikit-learn

Scikit-learn [129] je knižnica strojového učenia, ktorá bola implementovaná na použitie v programovacom jazyku Python a ktorá je založená, resp. používa NumPy, SciPy a Matplotlib knižnice. Táto knižnica poskytuje implementáciu bežne používaných algoritmov strojového učenia určených na klasifikáciu, regresiu, zoskupovanie (z angl. *clustering*), redukciu dimenzie (z angl. *dimensionality reduction*) spolu s nástrojmi na vyhodnotenie a vizualizáciu výsledkov.

Táto diplomová práca využíva scikit-learn knižnicu pri vytváraní ROC krivky, na výpočet AUC skóre, ale aj pri vytváraní konfúznej matice, pri štandardizácií dát a vytvorení trénovacej a testovacej množiny v prípade trénovacej dátovej sady, ale aj na rozdelenie trénovacej množiny na k rovnakých častí (použitých pri krížovej validácií).

Na záver je dôležité poznamenať, že PyOD *open-source framework* (pozri 5.3.3) používa pri implementácií algoritmov Isolation Forest, kNN a Local Outlier Factor modulov práve scikit-learn knižnicu.

5.3.2 TensorFlow a Keras

TensorFlow [130] je *open-source* platforma určená pre oblasť *deep learning* a neurónových sietí. Táto platforma, resp. knižnica obsahuje komplexný a flexibilný ekosystém rôznych nástrojov, knižníc, ale aj komunitných zdrojov, čo umožňuje vedcom, ale aj vývojárom vytvárať a nasadzovať aplikácie založené na najmodernejších riešeniach v oblasti strojového učenia. Táto platforma poskytuje stabilné API pre programovacie jazyky Python a C++. TensorFlow bol pôvodne vyvinutý výskumníkmi a inžiniermi z Google Brain [131] len na interné použitie za účelom vykonávania výskumu v oblasti strojového učenia a *deep learning* (resp. *deep neural networks*), avšak v roku 2015 bol TensorFlow vydaný pod licenciou *Apache License 2.0* [132].

Keras [133] je *model-level* knižnica vyvinutá Francois Chollet, ktorá poskytuje vysokoúrovňové API určené na vývoj modelov neurónových sietí. Táto knižnica sa nezaoberá činnosťami na nízkej úrovni (práca s tenzormi[58] atď.), ale namiesto toho sa spolieha na špecializovanú, dobre optimalizovanú knižnicu, resp. *framework*, ktorý manipuluje s tenzormi, a ktorý slúži ako *backend*[59] Keras. Keďže Keras je modulárna knižnica tak dokáže „fungovať“ nad rôznymi *backend*. V súčasnosti existujú tri *backend* implementácie určené pre Keras:

1. TensorFlow – *framework* vyvinutý spoločnosťou Google, ktorý je určený na manipuláciu s tenzormi
2. CNTK – jednotný súbor nástrojov určených pre oblasť *deep learning* vyvinutý spoločnosťou Microsoft
3. Theano – *framework* vyvinutý LISA LAB na Université de Montréal, ktorý je určený na manipuláciu s tenzormi

Keras bol vyvinutý s cieľom, aby pomocou neho bolo možné jednoducho a rýchlo experimentovať s algoritmami v oblasti *deep learning* a medzi jeho hlavné výhody patrí to, že umožňuje jednoduché a rýchle vytváranie prototypov, ale aj to, že podporuje implementácie rôznych typov neurónových sietí.

Do verzie 1.1.0 Keras platilo, že predvoleným *backend* pre Keras bol Theano, ale od verzie 1.1.0 sa to zmenilo a predvoleným *backend* sa pre Keras stal TensorFlow, čím sa používanie Keras a TensorFlow zvýšilo, pretože nebolo možné mať nainštalovaný Keras bez toho, aby nebol nainštalovaný TensorFlow, ale aj vďaka tomu, že sa Keras stával čoraz obľúbenejším medzi používateľmi kvôli jednoduchosti jeho API. Vo verzií TensorFlow 1.10.0 bol predstavený *submodul* **tf.keras**, čo bol prvý krok k tomu, aby sa Keras priamo integroval do samotného TensorFlow. V septembri 2019, keď spoločnosť Google vydala TensorFlow verzie 2.0 sa Keras stal oficiálnym vysokoúrovňovým

[58] Objekt v matematike, ktorý je zovšeobecnením pojmu vektor

[59] Výpočtový stroj/systém, ktorý vytvára sieťový graf/topológiu, spúšťa optimalizátory a vykonáva rôzne operácie s číslami.

API pre TensorFlow. Na oficiálnych stránkach Keras [133] sa v súčasnosti píše odporúčanie, ktoré hovorí o tom, aby používatelia, ktorí používajú tzv. *multi-backend* Keras, ktorý používa TensorFlow *backend* začali používať tf.keras (súčasťou TensorFlow verzie 2.0), ktorý je lepšie udržiavaný a má lepšiu integráciu s funkciami TensorFlow. Na záver je potrebné dodať, že posledné vydanie Keras verzie 2.3.0 (ktorý nie je súčasťou TensorFlow) bolo posledným vydaním tzv. *multi-backend* Keras a *multi-backend* Keras je nahradený vyššie spomínaným tf.keras. Inými slovami sa dá povedať, že užívatelia pôvodného *multi-backend* Keras by mali začať používať TensorFlow verzie 2.0 a tf.keras pre ich ďalšie projekty v oblasti *deep learning* a neurónových sietí [134] aj z toho dôvodu, že chyby budú v pôvodnom Keras (toho, ktorý nie je súčasťou TensorFlow) opravované len do apríla roku 2020.

Podľa webovej služby GitHub, ktorá zverejnila The State of the Octoverse (pozri [125]) patrí TensorFlow do Top 5 *open-source* projektov za rok 2019, čo sa týka počtu prispievateľov. Podľa Belani [135], ale aj napr. podľa Maayan [136] patrí TensorFlow (spolu s Keras) do Top 5 *deep learning framework* za rok 2019.

V tejto diplomovej práci bola použitá knižnica Keras z *framework* TensorFlow (modul tf.keras) a konkrétne upravený Autoencoder modul z PyOD *framework* (detaily v nasledujúcej podsekcií 5.3.3).

5.3.3 PyOD

PyOD je *open-source framework* (vytvorený autormi Zhao, Nasrullah a Li pozri [137]) obsahujúci komplexnú a škálovateľnú sadu nástrojov napísaných v jazyku Python slúžiacu na detekciu odľahlých objektov (anomálií), ktorý je možné jednoducho nainštalovať pomocou nástroja *pip*. Medzi jeho hlavné vlastnosti patrí napr. unifikované API, detailná dokumentácia, ale aj množstvo (viac ako 20) naimplementovaných algoritmov z oblasti detekcie anomálií (pozri [138]). Autori tvrdia, že tento *framework* vznikol hlavne preto, pretože v programovacom jazyku Python, ktorý je jedným z najpoužívanejších jazykov, chýbala špecializovaná sada nástrojov (v oblasti strojového učenia), ktorá by bola určená pre oblasť detekcie anomálií. Síce existuje v programovacom jazyku Python knižnica strojového učenia akou je napr. scikit-learn (pozri podsekciu 5.3.1.3), ale táto knižnica sa špecificky nezaoberá detekciou anomálií. PyOD *framework* je založený, resp. využíva NumPy, SciPy, ale aj spomínanú scikit-learn knižnicu. Dokonca platí, že API PyOD *framework* bolo inšpirované dizajnom API scikit-learn a niektoré algoritmy ako je napr. Isolation Forest, kNN, ale aj Local Outlier Factor sú implementované s využitím práve tejto scikit-learn knižnice. Autoenkóder model je v tomto *open-source framework* implementovaný (ku dňu 5. januára) s využitím *multi-backend* Keras, ktorý používa TensorFlow *backend*, a teda sa ešte stále nepoužíva „nový“ tf.keras, ktorý je súčasťou TensorFlow verzie 2.0. Autor tejto diplomovej práce na tento fakt poukázal (pozri *pull request* [139]) a autor PyOD *framework* so zmenami

súhlasil, ale zmeny by sa mali prejaviť až v najbližších týždňoch/mesiacoch. Zmeny, ktoré sú v danom *pull request* uvedené boli otestované a autor tejto diplomovej práce lokálne zmenil príslušné riadky kódu v danom Autoenkóder module, a teda sa priamo používa už „nový“ tf.keras z TensorFlow. Na záver je potrebné dodať, že PyOD framework cituje čoraz viac vedeckých prác (pozri [140]).

V tejto diplomovej práci boli z PyOD *framework* využité implementácie (moduly) algoritmov (kNN, Isolation Forest, Local Outlier Factor a upravený Autoenkóder).

5.3.4 Jupyter notebook

Jupyter notebook [141] je webová *open-source* aplikácia/nástroj/konzola, ktorá umožňuje vytvárať a zdieľať kód, ale aj dokumenty. Jedná sa o užitočný nástroj, ktorý poskytuje prostredie, v ktorom je možné spustiť kód, resp. jednotlivé príkazy a pozrieť sa na ich výstupy, komentovať kód, pridávať rôzne poznámky a vysvetlenia (v Markdown[60]), ale aj vizualizovať dáta (pomocou grafov), a to všetko bez opustenia daného prostredia. Vďaka týmto vlastnostiam je Jupyter notebook používaný vedcami a dátovými analytikmi pri vykonávaní tzv. *workflow*[61], ktoré zahrnujú čistenie dát, štatistické modelovanie, vytváranie a trénovanie modelov strojového učenia, vizualizáciu údajov atď.

Jednotlivé príkazy (kusy kódu) sú vykonávané (individuálne) v bunkách, čím je možné logicky oddeliť jednotlivé časti celého dokumentu/kódu, ale aj napr. otestovať konkrétny blok kódu v danom projekte bez toho, aby sa musel vykonávať celý Jupyter notebook odznova. Vďaka tomu, že sú Jupyter notebooky takto interaktívne a flexibilné môžu byť napr. používané aj v pedagogickom prostredí pri vysvetľovaní a demonštrovaní jednotlivých postupov, a algoritmov.

Samotný Jupyter, ktorého názov vznikol z názvov troch rôznych programovacích jazykov **JU**lia, **PYT**hon a **R**, je napísaný v programovacom jazyku Python, ale podporuje aj iné jazyky ako je napr. Java, JavaScript atď. (pozri [142]), kde sa nachádza kompletný zoznam podporovaných kernelov[62].

V tejto diplomovej práci bol Jupyter notebook použitý v spojení s Python jádrom (kernelom) na prototypovanie[63], spúšťanie rôznych experimentov a na vizualizáciu výsledkov.

[60] Značkovací jazyk, ktorý slúži na úpravu obyčajného textu a jeho následný prevod na formátovaný text (umožňuje vyznačiť v texte nadpisy a zoznamy, doplniť odkazy atp.)

[61] Pracovný a technologický postup (schéma vykonávania komplexnej činnosti).

[62] Program, ktorý spúšťa a preveruje/skúma kód užívateľa.

[63] Činnosť používaná v procese vývoja softvéru a jedná sa o vytváranie prototypov aplikácií - neúplných verzií aplikácií/programov.

5.3.5 Ostatné knižnice

V tejto sekcií budú spomenuté všetky ostatné Python balíčky, ktoré boli primárne použité pri extrakcií príznakov (pozri sekciu 6.3.2) u web proxy záznamov. Tieto balíčky by sa dali rozdeliť do nasledujúcich kategórií podľa toho na čo boli použité:

- na manipuláciu s URL adresami – *tldextract* a urllib.parse (konkrétne *urlparse*)
- na manipuláciu s WHOIS databázami (resp. informáciami) – ipwhois (konkrétne *IPWhois*), *geoip2.database* a *whois*
- na manipuláciu s IP adresami – *ipaddress* a *socket*
- na manipuláciu so súbormi vo formáte JSON a YAML – *json*, resp. *yaml*

Okrem vyššie spomínaných balíčkov bola ešte použitá knižnica *timeit*[64], na meranie toho, ako dlho bežia jednotlivé algoritmy strojového učenia.

5.4 Zhrnutie

Pri návrhu skriptu, ktorého obsahom je celá tzv. *pipeline* strojového učenia sme vychádzali z bodov, ktoré sú uvedené na začiatku kapitoly č. 2 v knihe od Géron [122]. Celú praktickú časť sme vyvíjali v dvoch krokoch, prvým krokom bolo softvérové prototypovanie pomocou dvoch Jupyter notebookov a druhým krokom bola implementácia výsledného skriptu v programovacom jazyku Python (architektúru skriptu je možné nájsť na obr. 5.1). Výsledný skript bol navrhnutý tak, aby mohol byť po prípadných ďalších iteráciách integrovaný napr. do SIEM systémov a jeho výstup (súbor s detegovanými anomáliami) mohol slúžiť bezpečnostným analytikom ako ďalší zdroj informácií, ktorý im môže zjednodušiť ich prácu napr. pri analýze záznamov, ale aj pri rozhodovaní a riešení rôznych bezpečnostných incidentov. Druhá možnosť je tá, žeby tento skript plnil funkciu „automatizovaného“ filtra, ktorého výstupom je súbor s detegovanými anomáliami (web proxy záznamy, ktoré vzbudzujú podozrenie žeby mohli byť škodlivé). Následne by tento „zredukovaný“ výstup (počet záznamov určených k ďalšiemu spracovaniu by bol menší) mohol byť použitý ako vstup do ďalších systémov určených k detailnejšej analýze záznamov.

Výkonnosť jednotlivých modelov bude na konci ďalšej kapitoly v sekcií 6.4 hodnotená pomocou metrík ako je AUC skóre, TPR, FPR, precíznosť, ale aj na základe doby behu jednotlivých algoritmov. Praktická časť tejto diplomovej práce bola implementovaná pomocou programovacieho jazyka Python, detekcia anomálií bola vykonávaná za pomoci *framework* PyOD a softvérové prototypovanie prebiehalo (ako už bolo vyššie spomínané) pomocou Jupyter notebooku.

[64] Konkrétne funkcia *default_timer* z tejto knižnice.

Kapitola 6

Analýza záznamov a vyhodnotenie modelov

V úvodnej časti tejto kapitoly sa budeme venovať získavaniu a základnému popisu poskytnutých (*web proxy*) záznamov (popis jednotlivých príznakov, zdroj záznamov atp.). V ďalšej časti tejto kapitoly sa budeme venovať aplikácií návrhu (tento návrh bol vytvorený v kapitole 5) na poskytnuté bezpečnostné auditné záznamy, konkrétne na *web proxy* záznamy (od analýzy a predspracovania až po aplikáciu a vyhodnotenie jednotlivých modelov). Následne sa budeme venovať výsledkom a vyhodnoteniu jednotlivých modelov pomocou ROC kriviek a AUC skóre, pomocou rôznych metrík ako je TPR, FPR a precíznosť pre jednotlivé prahové hodnoty, ale aj pomocou doby behu jednotlivých algoritmov. V závere sa budeme venovať zhrnutiu celej tejto kapitoly.

6.1 Získavanie záznamov

Na úplnom začiatku písania tejto diplomovej práce bola firma, ktorá mala poskytnúť záznamy vo fáze testovania nasadenia log manažment riešenia od firmy Graylog a bolo takmer isté, že sa toto riešenie nasadí a bude poskytnutý „testovací“ účet pomocou ktorého bude možné analyzovať rôzne záznamy, ktoré sa v danej organizácií vyskytujú a následne ich „použiť“ a analyzovať v tejto diplomovej práci. Bohužiaľ z rôznych dôvodov sa nasadenie daného riešenia neuskutočnilo a po dohode s vedúcim tejto diplomovej práce sme sa rozhodli, že sa budeme snažiť získať záznamy od inej firmy, a to konkrétne od firmy, ktorá poskytuje riešenie v oblasti SIEM systémov.

Táto firma poskytuje riešenie (softvér) v oblasti SIEM systémov a má rôznych zákazníkov (napr. školy, malé, stredné, ale aj veľké firmy a organizácie). Tým pádom bola ideálnym kandidátom na to, aby niektorí z ich zákazníkov (firiem, prípadne škôl) mohol poskytnúť záznamy, na to, aby ich bolo možné „použiť“ a analyzovať v tejto diplomovej práci. Po vzájomných

rozhovoroch vyšlo najavo, že nám táto firma poskytne kontakt na školu, ktorá nám následne poskytne záznamy, ktoré bude možné analyzovať. Bohužiaľ už po prvom stretnutí začali vznikať rôzne problémy a ani v tomto prípade sa nepodarilo získať žiadne záznamy.

Nakoniec sa nám podarilo získať *web proxy* záznamy z reálneho prostredia od firmy Cisco Systems, konkrétne od jedného z jej zákazníkov (stredná firma). Tento druh záznamov je mimo iného možné nájsť aj v tabuľke 3.3, kde je možné nájsť typ/y udalosti, ktoré je možné detegovať z týchto záznamov. Nasledujúce sekcie budú venované týmto *web proxy* záznamom od ich základného popisu, aplikácií návrhu na tieto poskytnuté záznamy až po výsledky a vyhodnotenie jednotlivých modelov.

6.2 Základný popis poskytnutých záznamov

Vďaka tomu, že nám spoločnosť Cisco Systems, Inc. poskytla nasledujúcu (oštikovanú) dátovú sadu (z reálneho prostredia) bude možné otestovať a vyhodnotiť jednotlivé algoritmy strojového učenia (oblasť detekcie anomálií). *Web proxy* záznamy boli oštítkované pomocou Cognitive Intelligence produktu od firmy Cisco (detaily v podsekcií 6.2.2) a ako už bolo spomenuté v podsekcií 4.3.1.4 tak pri „implementácií“ detekcie anomálií zvolíme (sme zvolili) *semi-supervised* prístup.

Zjednodušene by sa dalo povedať, že *web proxy* server slúži ako brána medzi webovými prehliadačmi v lokálnej (miestnej) sieti a Internetom, vďaka čomu obsahujú tieto *web proxy* záznamy hodnotné informácie pomocou ktorých je možné detegovať rôzne podozrivé aktivity. *Web proxy* záznamy obsahujú informácie týkajúce sa odchádzajúcej a prichádzajúcej tzv. *web-based* komunikácie, príkladom môže byť napr. komunikácia hostiteľov v lokálnej sieti s hostiteľmi, ktorí sú umiestnený v externých sieťach (Internet). Na základe týchto informácií bude možné detegovať napr. prítomnosť škodlivého softvéru, „neobvyklé“ správanie sa pri prehľadávaní stránok atď. (pozri tabuľku 3.3). Táto poskytnutá dátová sada obsahuje údaje a informácie, ktoré sú pod NDA[65], a preto nemôže byť nikde zverejnená. Avšak všetky informácie o tejto dátovej sade, ktoré môžu byť zverejnené (nepodliehajú NDA) ako je napr. popis jednotlivých stĺpcov (príznakov), rôzne štatistiky o dátovej sade atď. budú v tejto diplomovej práci uvedené.

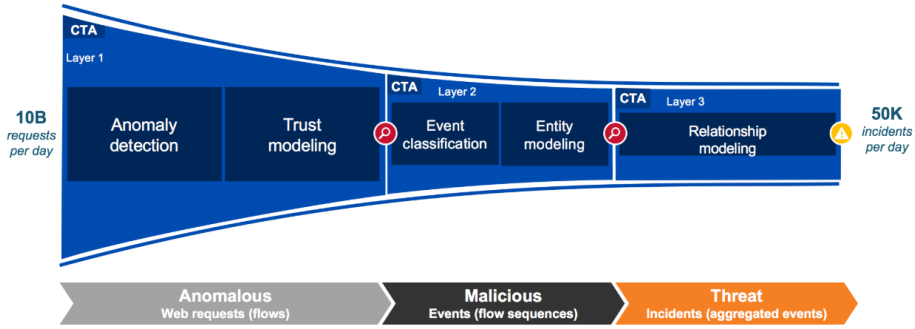
[65] Zmluva dvoch strán o nezdieľaní informácií s tretími stranami.

6.2.1 Popis jednotlivých príznakov

Poskytnutá dátová sada obsahuje na každom jednom riadku jeden *web proxy* záznam, ktorý je reprezentovaný pomocou 12 rôznych druhov/typov informácií:

1. *timestamp* – časová značka vzniku *web proxy* záznamu vo formáte *timestamp* (napr. 1575578160763 – po prevode do UTC formátu (2019-12-05 20:36:00))
2. *URL* – URL adresa na ktorú bol vykonaný daná požiadavka (napr. `https://fit.cvut.cz`)
3. *sourcePort* – zdrojový port zdroja požiadavku (typicky 443 pre HTTPs a 80 pre HTTP)
4. *csFlowBytes* – počet prenesených bajtov od zdroja k cieli (napr. 1223)
5. *scFlowBytes* – počet prenesených bajtov od cieľa k zdroji (napr. 3288)
6. *UserAgent* – HTTP User-Agent hlavička na identifikáciu webového prehliadača, operačného systému atď. (napr. Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko)
7. *contentTypeString* – Content-Type získaný z MIME hlavičky – typ internetového média (napr. text/plain; charset=utf-8)
8. *reputation* – reputačné skóre získané z webového proxy servera CWS
9. *magicContentTypeString* – Content-Type získaný pomocou knižnice libmagic – typ média (napr. text/plain; charset=utf-8)
10. *httpMethod* – metóda HTTP protokolu (napr. GET, POST atď.)
11. *ServerIP* – IP adresa servera, alebo cieľa pripojenia (napr. 147.32.232.248)
12. *label* – príznak, ktorého obsahom sú tzv. identifikátory (štítky), ktoré definujú to, či daný *web proxy* záznam je „normálny“ (nebol detegovaný ako malvér) alebo podozrivý (škodlivý) a navyše, či tento podozrivý (škodlivý) *web proxy* záznam patrí do jednej z n škodlivých kategórií (jednotlivé škodlivé kategórie budú bližšie popísané v podsekcií 6.3.1), príkladom môže byť *label* napr. GOOD, PUA1, INF2 atď.

Jednotlivé druhy/typy informácií budú buď. priamo použité ako príznaky pre algoritmy strojového učenia alebo budú použité na extrakciu ďalších príznakov, alebo nebudú vôbec použité (detaily a doplňujúce informácie v podsekcií 6.3.1).



Obr. 6.1: Architektúra Cognitive Intelligence produktu [143]

6.2.2 Zdroj záznamov a ich štítkovanie

Zdrojom poskytnutých *web proxy* záznamov je jeden zo zákazníkov (stredných firiem) firmy Cisco Systems, Inc. Poskytnuté *web proxy* záznamy sú rozdelené na dva súbory, podľa toho z akého časového obdobia pochádzajú. Prvý súbor obsahuje *web proxy* záznamy konkrétneho zákazníka z časového obdobia (2019-12-04 23:38:00 až 2019-12-05 23:38:59) a časť tohto súboru bude použitá ako trénovacia/učiaca množina pre algoritmy strojového učenia. Druhý súbor obsahuje *web proxy* záznamy toho istého zákazníka, avšak už z iného časového obdobia, a to konkrétne (2019-12-11 23:34:00 až 2019-12-12 23:37:59), tento súbor bude použitý ako testovacia množina a budú na ňom otestované a vyhodnotené jednotlivé algoritmy strojového učenia.

Samotné štítkovanie (z angl. *labeling*), teda označovanie, resp. rozdeľovanie jednotlivých *web proxy* záznamov na legitímne („normálne“) a na podozrivé (škodlivé) bolo vykonané pomocou Cognitive Intelligence produktu od firmy Cisco Systems, Inc. Tento produkt sa primárne zameriava na detekciu škodlivého softvéru a škodlivej komunikácií na sieti (dokáže spracovávať a analyzovať *web proxy* a NetFlow záznamy). Využívaním viacerých vrstiev predspracovania, detekcie anomálií a klasifikácie sa rozlišuje normálne správanie od toho škodlivého. V prípade, že sa zistí škodlivá aktivita, tak tento produkt vytvorí incident na základe ktorého môže správca postupovať pri riešení tejto infekcie. Architektúru Cognitive Intelligence produktu je možné nájsť na obr. 6.1. Bližšie informácie o tomto Cognitive Intelligence produkte je možné nájsť v Cisco Blog od Rehák a Anderson [143].

Na záver je potrebné dodať, že boli poskytnuté *web proxy* záznamy, ktoré „vstupujú“ do Cognitive Intelligence produktu, a teda nie sú žiadnym spôsobom upravené (resp. „obohatené“). Získané štítky pre jednotlivé *web proxy* záznamy pochádzajú z poslednej Layer 3 vrstvy (pozri obr. 6.1), v našom prípade pochádzajú jednotlivé štítky z externého *blacklist*, ktorý sa vytvára z jednotlivých incidentov. Prvý súbor, ktorého časť bude použitá ako trénova-

cia množina obsahuje 67 944 017 *web proxy* záznamov. Druhý súbor, ktorého časť bude použitá ako testovacia množina obsahuje 61 953 677 *web proxy* záznamov. Bližšie informácie o jednotlivých súboroch a záznamoch budú uvedené v podsekcií 6.3.1.

6.3 Aplikácia návrhu na poskytnuté záznamy

V tejto sekcií sa budeme venovať *web proxy* záznamov od ich analýzy až po aplikáciu a vyhodnotenie jednotlivých modelov a budeme vychádzať z návrhu popísaného v kapitole č. 5. Všetky jednotlivé kroky, resp. ich implementáciu je možné nájsť v nasledujúcich súboroch, ktoré sa nachádzajú v prílohe tejto diplomovej práce v adresári *src* a sú podrobne okomentované spolu s interpretáciou jednotlivých výstupov:

1. Jupter notebook (1_iteration_web_proxy_dataset.(ipynb|html)) – obsahuje analýzu a vizualizáciu trénovacej množiny *web proxy* záznamov (z 5. decembra), extrakciu príznakov, ale aj vyhodnotenie jednotlivých modelov na časti trénovacej množiny s predvolenými hyperparametrami

2. Jupter notebook (2_iteration_web_proxy_dataset.(ipynb|html)) – je pokračovaním 1. Jupyter notebook a obsahuje analýzu a vizualizáciu testovacej množiny *web proxy* záznamov (z 12. decembra), ale hlavne krížovú validáciu a hľadanie „optimálnych parametrov“ (detaily v sekcií 6.3.3)

3. Python skript (web_proxy_logs_demo.py) a jeho príslušné súbory – skript, ktorý je spojením oboch Jupyter notebookov a jeho výstupy (výsledky) sú obsahom sekcie 6.4

6.3.1 Analýza a predspracovanie dát

Pri analýze štítkov, konkrétne príznaku *label* sme zistili, že tento príznak obsahuje niekoľko (v závislosti na tom, či ide o trénovací alebo testovací súbor) unikátnych identifikátorov malvéru, z ktorých je možné odvodiť (pomocou ďalších informácií od firmy Cisco Systems, Inc.) informácie ako je napr. *severita*, *risk* skóre, kategória malvéru apod., avšak presné názvy týchto identifikátorov nie je možné zverejniť, pretože sú pod NDA. Na základe príznaku *label* sme vytvorili (s pomocou vyššie spomínaných informácií) ďalšie tri príznaky, ktoré budú primárne slúžiť pri analýze záznamov, ale aj pri vyhodnotení jednotlivých modelov.

Tieto príznaky môžu nadobúdať nasledujúce hodnoty:

- *outcome*
  - **legitimate** – označuje tzv. legitímne („normálne“) *web proxy* záznamy
  - **malicious** – označuje tzv. podozrivé (škodlivé) *web proxy* záznamy
- *category*
  - **PUA** – aplikácie, ktoré obsahujú kód pre vykonávanie rôznej neželanej činnosti akou je napr. zobrazovanie reklám, inštalovanie panelov nástrojov atď. alebo inej nejasnej činnosti, ktorá je zo strany užívateľa nežiadúca
  - **ad injector** – aplikácie, ktoré napr. vkladajú nežiadúce reklamy na webové stránky pri ich prehliadaní, zakrývajú alebo dokonca nahradzujú jednotlivé reklamy ich vlastnými reklamami apod.
  - **information stealer** – aplikácie, ktoré sa snažia (zo systému) získať a odoslať informácie o užívateľovi (prihlasovacie informácie ako sú napr. používateľské mená a heslá apod.) za účelom zneužitia
  - **malvertising** – (*malicious advertising*), aplikácie resp. škodlivé reklamy, ktoré sa nachádzajú na rôznych (aj legitímnych) webových stránkach na to, aby šírili medzi používateľmi malvér
  - **cryptojacking** – (*cryptomining malware*), aplikácie alebo súčasti škodlivého softvéru vyvinuté na to, aby využívali zdroje daného systému na ťažbu kryptomien bez výslovného súhlasu používateľa
  - **malicious content distribution** – aplikácie, resp. hrozby, ktoré súvisia s webovými stránkami používanými na distribúciu PUA
  - **undefined**– aplikácie, resp. hrozby, ktoré nepatria (neboli kategorizované) do vyššie spomínaných kategórií
  - **non-malicious** – v tomto prípade sa nejedná o hrozbu, ale ide o legitímne *web proxy* záznamy (**non-malicious** kategória sa rovná **legitimate** *outcome*)
- *risk_level*
  - **no risk** – aplikácie/programy bez rizika (legitímne *web proxy* záznamy)
  - **low** – škodlivé aplikácie/programy s nízkou úrovňou rizika
  - **medium** – škodlivé aplikácie/programy so strednou úrovňou rizika
  - **high** – škodlivé aplikácie/programy s vysokou úrovňou rizika

Prvý súbor s *web proxy* záznamami pochádzajúci z časového obdobia 2019-12-04 23:38:00 až 2019-12-05 23:38:59 obsahuje (ako už bolo vyššie spomenuté) celkovo 67 944 017 *web proxy* záznamov. Z toho 67 914 842 *web proxy* záznamov patrí (bolo kategorizovaných produktom Cognitive Intelligence) do kategórie legitímnych („normálnych“) a zvyšok (29 175) *web proxy* záznamov patrí do kategórie podozrivých (škodlivých). Aby extrakcia príznakov netrvala veľmi dlho, ale hlavne kvôli tomu, že jednotlivé modely budeme trénovať a vyhodnocovať na lokálnom stroji (obmedzujúcimi faktormi sú pamäťové nároky a výpočtový výkon) sme sa rozhodli, že do trénovacej množiny vyberieme náhodným výberom z celkového počtu 67 914 842 legitímnych („normálnych“) *web proxy* záznamov 3 012 293 vzoriek, čo tvorí niečo medzi 4 až 5% z celkového počtu legitímnych („normálnych“) *web proxy záznamov*. Z celkového počtu 29 175 *web proxy záznamov*, ktoré patria do kategórie podozrivých (škodlivých) zoberieme zo všetkých kategórií (okrem jedného druhu škodlivých *web proxy* záznamov, ktorý má nízku úroveň rizika a patrí do kategórie PUA) všetky vzorky a zo spomínaného druhu zoberieme cca 8%, čo tvorí niečo vyše ako 1 800 *web proxy* záznamov. Celkový počet *web proxy záznamov*, ktoré patria do kategórie podozrivých (škodlivých) bude 7 580. Tieto jednotlivé čísla sme u tejto trénovacej množiny vybrali týmto spôsobom preto, aby sme pri tzv. $5k$ Fold krížovej validácií (detaily v podsekcií 6.3.3) mohli trénovať/učiť modely strojového učenia na 75% legitímnych („normálnych“) *web proxy* záznamoch, čo tvorí cca 2,25 milióna vzoriek a validovať tieto modely strojového učenia na 25% legitímnych („normálnych“) *web proxy* záznamoch, čo tvorí cca 750 000 vzoriek, ktoré budú „zmiešané“ so všetkými (7 580) *web proxy* záznamami, ktoré patria do kategórie podozrivých (škodlivých), čo zodpovedá výslednému pomeru (podozrivé : legitímne) 1 : 100[66], a to konkrétne 7 580 : 750 000. Rozdelenie, resp. jednotlivé počty podozrivých (škodlivých) *web proxy* záznamov (v trénovacej množine) na základe vyššie spomínaných príznakov je možné nájsť v tabuľke 6.1. Stĺpec *hrozby* je vyjadrením toho, koľko tzv. unikátnych identifikátorov malvéru sa nachádza v danej kategórií (napr. v prípade kategórie **ad injector** sa v trénovacej množine nachádza 5 rôznych unikátnych identifikátorov danej kategórie malvéru).

Druhý súbor s *web proxy* záznamami pochádzajúci z časového obdobia 2019-12-11 23:34:00 až 2019-12-12 23:37:59 (ako už bolo vyššie spomenuté) obsahuje celkovo 61 953 677 *web proxy* záznamov. Z toho 61 933 823 *web proxy* záznamov patrí (bolo kategorizovaných produktom Cognitive Intelligence) do kategórie legitímnych („normálnych“) a zvyšok (19 854) *web proxy* záznamov patrí do kategórie podozrivých (škodlivých). Z dôvodu toho, že druhý súbor, resp. jeho časť bude použitá ako testovacia množina a vyhodnotenie trvá kratšiu dobu ako trénovanie sme sa rozhodli, že z celkového

[66] V praxi sa častokrát používajú pomery 1 : 10 00 alebo aj 1 : 10 000, ale z dôvodu vyššie spomínaných dvoch faktorov (pamäťové nároky a výpočtový výkon) sme si to nemohli dovoliť.

kategória	**risk level**				**hrozby**
	low	medium	high	no risk	
PUA	3506	1978	0	0	5
ad injector	0	1290	0	0	5
information stealer	0	20	382	0	2
malvertising	158	0	0	0	3
cryptojacking	14	0	0	0	1
malicious content distr.	0	0	0	0	0
undefined	232	0	0	0	5
non-malicious	0	0	0	3012293	0
súčet	**3910**	**3288**	**382**	**3012293**	**21**

Tabuľka 6.1: Rozdelenie trénovacej množiny

kategória	**risk level**				**hrozby**
	low	medium	high	no risk	
PUA	15705	2280	0	0	5
ad injector	0	1136	0	0	7
information stealer	0	9	161	0	2
malvertising	45	0	0	0	3
cryptojacking	469	0	0	0	1
malicious content distr.	3	0	0	0	1
undefined	46	0	0	0	5
non-malicious	0	0	0	1988076	0
súčet	**16268**	**3425**	**161**	**1988076**	**24**

Tabuľka 6.2: Rozdelenie testovacej množiny

počtu 19 854 *web proxy* záznamov, ktoré patria do kategórie podozrivých (škodlivých) zoberieme všetky vzorky, a aby bol zachovaný pomer (1 : 100) tak ako je to v prípade validačnej množine (pozri vyššie) tak z celkového počtu 61 933 823 legitímnych („normálnych“) *web proxy* záznamov vyberieme náhodne 1 988 076 vzoriek, čo je cca 3% z celkového počtu legitímnych („normálnych“) *web proxy* záznamov, čo zodpovedá výslednému pomeru (podozrivé : legitímne) 1 : 100, a to konkrétne 19 854 : 1 988 076. Rozdelenie, resp. jednotlivé počty podozrivých (škodlivých) *web proxy* záznamov (v testovacej množine) na základe vyššie spomínaných príznakov je možné nájsť v tabuľke 6.2, stĺpec *hrozby* (rovnako ako v predchádzajúcej tabuľke) je vyjadrením toho, koľko tzv. unikátnych identifikátorov danej kategórie malvéru sa nachádza v danej kategórií (napr. v prípade kategórie **ad injector** sa v testovacej množine nachádza 7 rôznych unikátnych identifikátorov malvéru).

```
timestamp       URL     sourcePort      csFlowBytes     scFlowBytes     UserAgent       contentTypeString       reputation
magicContentTypeString  httpMethod      ServerIP        label   outcome category        risk_level

2019-12-01 10:40:59     https://google.sk       443     1723    3288    Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
application/json        -4.08   text/plain; charset=utf-8       GET     8.8.8.8 GOOD    legitimate      non-malicious   no risk
```

Obr. 6.2: Príklad web proxy záznamu

Pri následnej analýze oboch súborov (už pracujeme len zo spomínanými časťami jednotlivých súborov) sme zistili, že príznaky *magicContentTypeString*, *contentTypeString* a *UserAgent* obsahujú veľké množstvo chýbajúcich hodnôt (viac ako 50%). Tieto príznaky sme preto odstránili, keďže príznak u ktorého je väčšina hodnôt umelo doplnená má zanedbateľný, či až nechcený prínos pri učení jednotlivých algoritmov strojového učenia. Príznaky *timestamp*, *csFlowBytes*, *scFlowBytes* a *sourcePort* nám pri testovaní nepriniesli žiadne informácie, a preto sme ich takisto odstránili, a budeme pracovať len s príznakmi *URL*, *reputation*, *httpMethod*, *ServerIP* a samozrejme *label*, *outcome*, *category* a *risk_level*. Na niektoré z týchto príznakov „aplikujeme“ extrakciu príznakov (pozri nasledujúcu sekciu 6.3.2).

Pri analýze príznaku *URL* sme zistili, že niektoré *web proxy* záznamy obsahujú v doménovej časti URL adresy IP adresu, čo nie je žiadúce, a preto použijeme tzv. *reverse DNS lookup*, ktorý dokáže „preložiť“ IP adresu na názov domény alebo prípadne hostiteľa, a tým získame želanú (preloženú) doménovú časť danej URL adresy. Celý tento „preklad“ trval pre oba súbory a pre dané IP adresy dohromady cca 1 hodinu. Pri ďalšej analýze príznaku *URL* sme sa zamerali na tzv. URI scheme, ktorá pomáha identifikovať zdroj napr. http, https, ftp apod. Z analýzy sme zistili, že vyše 95% *web proxy* záznamov „obsahuje“ v časti URI scheme HTTPs a zvyšok (niečo malo než 5%) obsahuje v časti URI scheme HTTP, čo v tomto prípade (výrazne) ovplyvňuje extrakciu príznakov, pretože nemôžeme napr. extrahovať príznaky z URL, ktoré sú spojené s dĺžkou názvu súboru, argumentu, adresára apod. Avšak dalo by sa povedať, že táto situácia (tento pomer HTTPs : HTTP vo *web proxy* záznamoch) je už v dnešnej dobe bežná vec, no na druhej strane je potrebné dodať, že nám to detekciu anomálií len sťažuje. Na obr. 6.2 je možné vidieť ukážku/príklad *web proxy* záznamu, ktorý bol umelo vytvorený, avšak štruktúra a obsah zodpovedá *web proxy* záznamov z reálneho prostredia, ktoré sme získali od firmy Cisco Systems.

6.3.2 Extrakcia príznakov a úprava rozsahu dát

Pri extrakcií príznakov sme (ako už bolo vyššie spomínané) vychádzali z príznakov *httpMethod*, *URL* a *ServerIP*. Z týchto príznakov vznikli (extrakciou) nasledujúce nové príznaky (niektoré z príznaky boli inšpirované prehľadom od Sahoo, Liu a Hoi [144]):

1. *httpMethod*
    - Method_freq – príznak, ktorý využíva frekvenciu jednotlivých kategórií tzv. HTTP *request methods* (v tomto prípade CONNECT, GET atď.)
2. *URL*
    - length_of_URL – príznak, ktorý reprezentuje dĺžku (počet znakov) URL reťazca
    - number_of_special_characters – príznak, ktorý reprezentuje počet špeciálnych znakov v danom URL reťazci
    - number_of_sub_domains – príznak, ktorý reprezentuje počet subdomén v danom URL reťazci
    - length_of_primary_domain – príznak, ktorý reprezentuje dĺžku (počet znakov) primárnej domény v danom URL reťazci
    - length_of_subdomains – príznak, ktorý reprezentuje dĺžku (počet znakov) subdomén v danom URL reťazci
    - alexa_position – príznak, ktorý je vyjadrením pozície konkrétnej domény prípadne subdomény v súbore, ktorý obsahuje tzv. Alexa Traffic Rank (detaily budú uvedené nižšie)
    - page_rank – príznak, ktorý udeľuje tzv. Open PageRank konkrétnej doméne prípadne subdoméne na základe informácií v súbore, ktorý obsahuje tento Open PageRank (detaily budú uvedené nižšie)
    - domain_age – príznak, ktorý je vyjadrením „veku“ danej domény (v dňoch), ktorý je vypočítaný z dátumu registrácie/vytvorenia tejto domény (detaily budú uvedené nižšie)
3. *ServerIP*
    - ASN_freq – príznak, ktorý využíva frekvenciu jednotlivých ASN čísiel (detaily budú uvedené nižšie)

Príznak *alexa_position* vznikol z tzv. Alexa Traffic Rank, ktorý je navrhnutý tak, aby „meral“ popularitu jednotlivých webových stránok. Tento Alexa Traffic Rank je vypočítaný na základe počtu denných návštevníkov a počtu zobrazení jednotlivých stránok počas trojmesačného obdobia a je poskytovaný (prístup k zoznamu) službou Alexa Top Sites [145]. O jednotlivých webových stránkach sú okrem Alexa Traffic Rank k dispozícií aj ďalšie informácie ako je napr. počet zobrazení jednotlivých webových stránok na milión používateľov, priemerný počet zobrazení danej webovej stránky na jedného používateľa, ale aj percento používateľ navštevujúcich danú webovú stránku atď. Táto služba je, ale platená, a preto sme k nej prístup nemali (maximálne

si je možné zobraziť TOP 50 webových stránok pozri [146]. Avšak podarilo sa nám nájsť tzv. Alexa *static* súbor (obsahuje zoznam 594 915 webových stránok), ktorý síce neobsahuje priamo Alexa Traffic Rank, ale zoznam webových stránok, ktoré sú podľa tohto Alexa Traffic Rank zoradené a tento súbor by mal byť aktualizovaný každé tri mesiace (zoznam je možné stiahnuť z odkazu [147]). Túto informáciu (pozíciu) danej webovej stránky v tomto súbore použijeme ako príznak. V prípade, že sa daná webová stránka v súbore nenachádzala bola jej priradená pozícia 594 916, čo značí, že je to prvá pozícia, ktorá sa už v danom súbore nenachádza.

Príznak *page_rank* je podobný príznaku *alexa_position*, avšak v tomto prípade ide o tzv. Open PageRank[148], ktorý je založený na *open-source* dátach, a je teda voľné dostupný pre každého, buď pomocou API alebo ako stiahnuteľný zip súbor. My sme zvolili druhú možnosť a súbor stiahli z odkazu [149], súbor obsahuje zoznam 10 milióna webových stránok a u každej webovej stránky je k dispozícií spomínaný Open PageRank, ktorý reprezentujeme ako príznak *page_rank*. V prípade, že sa daná webová stránka v súbore nenachádzala bola jej priradená hodnota Open PageRank rovná 0.

Príznak *domain_age* je vytvorený z príznaku *URL* (z URL adresy) pomocou Python balíčka *whois*, ktorý dokáže získať informácie o dátume registrácie danej webovej stránky, resp. domény. Konkrétna hodnota daného príznaku (*domain_age*) je počítaná nasledujúcim vzorcom:

$$domain\_age = registration\_date - UNIX\ Epoch\ time, \tag{6.1}$$

kde *domain_age* je „vek“ danej domény v dňoch, *registration_date* je dátum registrácie danej domény získanej pomocou Python balíčka *whois* a *UNIX Epoch time* je počet sekúnd uplynulých od okamihu koordinovaného svetového času (UTC) 00:00:00 1. januára 1970. V prípade, že sa Python balíčku *whois* nepodarilo získať informáciu o dátume registrácie danej domény, čo sa stalo u cca 1% prípadov) boli tieto hodnoty nahradené strednou hodnotou tohto *domain_age* príznaku. Celý tento proces získavania dátumu registrácie trval pre oba súbory a pre dané domény cca 8 hodín.

Príznak *ASN_freq* je vytvorený z príznaku *ServerIP* a je založený na jednotlivých frekvenciách čísiel autonómnych systémov (ASN), ktoré sú obsahom jednotlivých súborov. Číslo ASN je možné získať „preložením“ IP adresy (v našom prípade IP adresy servera) na konkrétne číslo ASN a to pomocou WHOIS[67] databázy, ktorá tieto informácie obsahuje. Na získavanie týchto informácií bola použitá databáza GEOIP2 Lite od spoločnosti MaxMind, a to konkrétne GeoLite2 ASN. Na získanie prístupu k tejto databáze bolo potrebné vytvoriť účet, ktorý je avšak zadarmo (bližšie informácie sú uvedené v zdroji [150]). V prípade, že sa tieto informácie (ASN číslo) nepodarilo z tejto da-

[67]WHOIS je služba, ktorá umožňuje získať informácie o vlastníkovi, registrátorovi domény, čísle ASN atď.

tabázy získať, bol použitý (ako druhá možnosť[68]) Python balíček *IPWhois*, ktorý je takisto schopný tieto informácie získať. V prípade, že sa ani pomocou spomínaného Python balíčka nepodarilo získať informáciu o ASN čísle bola hodnota nedefinovaná a bola pre túto hodnotu vytvorená špeciálna kategória.

Následne boli na základe vypočítaného štandardného korelačného koeficientu (pomocou korelačnej matice) medzi všetkými spomínanými (číselnými) príznakmi a cieľovým príznakom *outcome* na trénovacej množine vybrané príznaky, ktoré sa budú ďalej používať a na základe ktorých budú trénované a vyhodnotené jednotlivé modely. Medzi tieto príznaky patrí *domain_age*, *reputation*, *page_rank*, *ASN_freq*, *alexa_position*, *length_of_subdomains*, *Method_freq* a *length_of_primary_domain*.

Pri úprave rozsahu dát (resp. pri škálovaní jednotlivých numerických príznakov) sme použili *štandardizáciu*, aby hodnoty danej vlastnosti/príznaku mali nulovú strednú hodnotu a rozptyl rovný 1.

6.3.3 Výber a trénovanie jednotlivých modelov

Po extrakcií príznakov, výbere príznakov a úprave rozsahu dát nasleduje samotný výber a trénovanie jednotlivých modelov. Ako už bolo spomínané zaoberali sme sa tzv. *semi-supervised* technikou detekcie anomálií a anomálie sme sa snažili detegovať pomocou 4 algoritmov, a to konkrétne pomocou k-NN, Local Outlier Factor, Isolation forest a Autoenkóder (pozri sekciu 4.4). Z *framework* PyOD (pozri podsekciu 5.3.3) sme použili moduly, ktoré implementácie jednotlivých algoritmov obsahujú. Obsahom týchto modulov sú aj predvolené tzv. hyperparametre, ktoré sa nastavujú ešte pred samotným trénovaním. Pri trénovaní jednotlivých modelov sme experimentovali aj s týmito predvolenými hyperparametrami, ale nakoniec sme sa rozhodli, že sa budeme snažiť nájsť množinu optimálnych hyperparametrov (optimalizácia a ladenie hyperparametrov) pre jednotlivé modely pomocou krížovej validácie, aby sme predišli tzv. *overfittingu*, ale aj kvôli tomu, aby sme lepšie „vyhodnotili“ jednotlivé množiny hyperparametrov pre jednotlivé modely.

Ako už bolo vyššie spomenuté pri výbere optimálnych hyperparametrov, ale aj na to, aby sme predišli tzv. *overfittingu* použijeme k-fold krížovú validáciu. Táto metóda rozdeľuje dátovu sadu (trénovaciu/učiacu množinu) na k rovnako veľkých disjunktných podmnožín s tým, že vždy sa jedna z podmnožín použije ako testovacia (validačná) množina (na tejto množine sa testuje výkonnosť a presnosť daného modelu) a zvyšných $k-1$ podmnožín sa použije ako trénovacia množina na trénovanie daného modelu. Tento proces sa opakuje k - krát (vždy iné podmnožiny) a jeho výstupom je tzv. generalizačná chyba algoritmu/modelu, ktorá je priemerom predikčných chýb vypočítaných na jednotlivých validačných podmnožinách. Pri „hľadaní“ množiny optimálnych hyperparametrov sme použili n-krát opakovanú ($n = 3$) kFold ($k = 5$)

[68]Táto možnosť bola pri testovaní prvou možnosťou, avšak počas získavania informácií sme narazili na obmedzenie tzv. *rate limiting*.

krížovú validáciu a množinu optimálnych hyperparametrov sme hľadali na základe AUC skóre, ale aj na základe toho ako dlho bežali jednotlivé modely/algoritmy. Celý tento proces, ktorý sa delí na 7 krokov, a ktorý sa pre každý jeden z modelov opakuje (samozrejme s inými množinami jednotlivých hyperparametrov) je možné nájsť na obr. 6.3.

Obsahom jednotlivých krokov celého procesu krížovej validácie je:

1. rozdelenie celej trénovacej množiny na normálne (legitímne) a podozrivé (škodlivé) *web proxy* záznamy
2. opakované náhodné rozdelenie celej množiny (na obrázku označenej ako N, ktorá obsahuje len normálne (legitímne) *web proxy* záznamy), čím vzniknú 3 náhodné rozdelené množiny kvôli opakovanej krížovej validácií
3. vytvorenie množiny hyperparametrov (množina je vytváraná algoritmom RandomSearch, ktorý vyberá z rozsahu definovaných hodnôt pre jednotlivé hyperparametre), ktorá bude vstupovať do každej z troch krížových validácií
4. proces samotnej krížovej validácie (na 75% normálnych (legitímnych) *web proxy* záznamoch budeme daný model trénovať a na zvyšných 25% normálnych (legitímnych) spojených so 100% podozrivými (škodlivými) *web proxy* záznamami tento model validovať)
5. výpočet spriemerovaného AUC skóre pre každú z troch krížových validácií, keďže výstupom každej z nich je 5 hodnôt (používame tzv. 5 kFold krížovú validáciu) AUC skóre
6. výpočet výsledného spriemerovaného AUC skóre, ktoré sa bude počítať zo spriemerovaných AUC skóre vypočítaných v predchádzajúcom kroku (používame opakovanú krížovú validáciu) a na základe tohto výsledného spriemerovaného AUC skóre sa bude hľadať optimálna množina hyperparametrov pre jednotlivé modely
7. od bodu č.3 opakujeme celý tento proces s tým, že do jednotlivých krížových validácií vstupuje iná množina hyperparametrov pre jednotlivé modely

Všetky experimenty a vyhodnotenia jednotlivých modelov, ktoré sa budú v nasledujúcich sekciách nachádzať boli spúšťané na prenosnom počítači, ktorý mal nasledujúce parametre:

- procesor – 1,6 GHz Dual-Core Intel Core i5
- veľkosť a frekvencia operačnej pamäte – 16 GB 2133 MHz
- grafická karta – Intel UHD Graphics 617 1536 MB

Obr. 6.3: Krížová validácia

Pri rôznych experimentoch s jednotlivými algoritmami strojového učenia sme prišli na to, že doba behu (trénovanie modelu a prideľovanie anomálneho skóre) u algoritmu k-NN bola niečo vyše 1 hodiny a 35 minút, a u algoritmu Local Outlier Factor to bolo podobne, a to síce niečo vyše 1 hodiny a 17 minút. Z toho dôvodu sme sa rozhodli, že pre tieto dva algoritmy nebudeme púšťať proces krížovej validácií a použijeme predvolené hyperparametre. Proces krížovej validácie sme teda použili len u algoritmu Isolation forest (presnejšie 3-krát opakovanú 5k Fold krížovú validáciu) a u algoritmu Autoenkóder (presnejšie 1-krát opakovanú 5k Fold krížovú validáciu). U algoritmu Autoenkóder sme použili len 1-krát opakovanú krížovú validáciu a to z toho dôvodu, že pri experimentoch s 3-krát opakovanou krížovou validáciou sa výsledne spriemerované AUC skóre pre jednotlivé krížové validácie výrazne nemenilo, ale aj preto, žeby to bolo časovo náročné. V prípade krížovej validácie aplikovanej na algoritmus Isolation Forest (proces krížovej validácie bol spustený 20-krát) sa výsledné spriemerované AUC skóre pohybovalo v intervale 0.8 až 0.836 a jednotlivé doby behu algoritmu v intervale od 45 sekúnd až po 450 sekúnd. V prípade algoritmu Autoenkóder (proces krížovej validácie bol spustený 30-krát sa výsledné spriemerované AUC skóre pohybovalo v intervale 0.842 až 0.864 a jednotlivé doby behu algoritmu v intervale 240 sekúnd až 2158 sekúnd. Najlepšie výsledné spriemerované AUC skóre bolo dosiahnuté na nasledujúcich hyperparametroch (pozri nižšie) a konkrétne výsledky je možné nájsť v tabuľke 6.3 a všetky uvedené výsledky v tejto aj v nasledujúcej sekcií budú zaokrúhľované na tisíciny.

algoritmus	**výsledné spriemerované AUC skóre**	**doba behu [s]**
k-NN	x	x
LOF	x	x
IForest	0.836	215.649
Autoenkóder	0.864	265.232

Tabuľka 6.3: Výsledné najlepšie spriemerované AUC skóre dosiahnuté pri krížovej validácií na „optimálnych“ hyperparametroch jednotlivých modelov

- k-NN
  - n_neighbors: 5
- Local Outlier Factor
  - n_neighbors: 15
- Isolation forest
  - max_features: 4
  - numbers_of_estimators: 100

- Autoenkóder
  - batch_size: 512
  - dropout_rate: 0.4
  - epochs: 10
  - hidden_activation: relu
  - hidden_neurons: [8, 2, 2, 8]
  - l2_regularizer: 0.4
  - optimizer: sgd
  - output_activation: softmax

Po tom, čo sa nám podarilo nájsť jednotlivé množiny optimálnych hyperparametrov pre jednotlivé modely sme začali s vyhodnotením týchto modelov na testovacej množine. Aplikácia a vyhodnotenie modelov bude popísané v nasledujúcej sekcií. Implementáciu (kód) vyššie spomínaného procesu je možné nájsť v prílohe, v adresári src, v podadresári impl a konkrétne adresár jupyter, a 2.iterácia (2_iteration_web_proxy_dataset.(ipynb|html) Jupyter notebook).

Na záver je potrebné dodať, že „hľadanie“ najoptimálnejších hyperparametrov pre jednotlivé modely/algoritmy strojového učenia nebolo cieľom tejto diplomovej práce, avšak ukázali sme spôsob a nástroj, ktorým to je možné dosiahnuť.

6.3.4 Aplikácia a vyhodnotenie jednotlivých modelov

Po výbere resp. nájdení „optimálnych“ hyperparametrov prebiehala aplikácia a vyhodnotenie jednotlivých modelov nasledujúcim spôsobom. Trénovanie jednotlivých modelov sa uskutočnilo na trénovacej množine (z 5. decembra 2019) z ktorej sa na trénovanie „použili“ len normálne (legitímne) *web proxy* záznamy (*semi-supervised* technika) a celkový počet vzoriek bol 3 012 293. Jednotlivé modely boli následne „aplikované“ na testovaciu množinu (z 12. decembra), ktorá obsahovala 1 988 076 normálnych (legitímnych) a 19 854 podozrivých (škodlivých) *web proxy* záznamov. Každému vzorku, resp. *web proxy* záznamu pridelili jednotlivé modely príslušné anomálne skóre a na základe tohto anomálneho skóre, ale aj na základe informácie do ktorej z tried (negatívnej alebo pozitívnej) patrí príslušný *web proxy* záznam bola vykreslená ROC krivka a vypočítané AUC skóre na základe čoho sme mohli vyhodnotiť a porovnať jednotlivé modely medzi sebou. Následne na základe definovanej prahovej hodnote (z angl. *threshold*) bola vykreslená konfúzna matica a z nej vypočítané ďalšie metriky ako je TPR, FPR, precíznosť, ale aj TPR vypočítané pre každú zo škodlivých kategórií a tzv. *risk* levelov pomocou

ktorých je takisto možné porovnávať jednotlivé modely medzi sebou. V neposlednom rade sme merali aj uplynutý čas behu jednotlivých modelov/algoritmov. Výsledky a vyhodnotenie jednotlivých modelov sa nachádza v nasledujúcej sekcií 6.4.

6.4 Výsledky a vyhodnotenie jednotlivých modelov

Táto sekcia je rozdelená do jednotlivých podsekcií nasledujúcim spôsobom:

- v podsekcií 6.4.1 sa budeme venovať komentovaniu vytvorených ROC kriviek pre jednotlivé modely, resp. algoritmy strojového učenia, komentovaniu AUC skóre pre jednotlivé modely, ale aj dobám behu jednotlivých modelov
- v podsekcií 6.4.2 sa budeme venovať komentovaniu výsledkov, ktoré závisia od zvolenej prahovej hodnoty a výsledky ukážeme pre dve rôzne prahové hodnoty (konkrétne 0.6 a 0.8)

Na záver je potrebné dodať, že všetky grafy (ROC krivky, histogramy, ale aj konfúzne matice) a výsledky (doba behu jednotlivých algoritmov, TPR, precíznosť a FPR) pre jednotlivé algoritmy strojového učenia je možné nájsť aj v priloženom CD (pozri dodatok D). Avšak obsahom priloženého CD nie sú súbory s detegovanými anomáliami (tak ako to je znázornené na obr. 5.1) pre jednotlivé prahové hodnoty, a to z toho dôvodu, že obsahom súborov sú jednotlivé *web proxy* záznamy (detegované ako anomálne/škodlivé *web proxy* záznamy), ktoré ako už bolo spomínané sú pod NDA.

6.4.1 ROC krivky, AUC skóre a doby behov jednotlivých modelov

Ako už bolo vyššie spomínané v tejto podsekcií sa budeme venovať komentovaniu jednotlivých ROC kriviek „vytvorených“ jednotlivými modelmi, ale budeme sa venovať aj komentovaniu AUC skóre a jednotlivým dobám behu algoritmov. Všetky ROC krivky je možné nájsť v dodatku B. Z dôvodu toho, že algoritmy Isolation forest a Autoenkóder nedokázali správne detegovať (ako sa ukáže v podsekcií 6.4.1.2) jeden konkrétny typ hrozby, ktorá patrí do škodlivej kategórie PUA budeme u všetkých výsledkov a grafov stále zobrazovať výsledky, resp. grafy so všetkými škodlivými kategóriami, ale aj bez tohto konkrétneho druhu hrozby, ktorá patrí do škodlivej kategórie PUA a budeme ju označovať ako **PUA kategórie č. 1.** (detaily v podsekcií 6.4.1.2). Na záver je potrebné dodať, že táto konkrétna PUA hrozba má najnižší *risk* level a tzv. *severity* level spomedzi všetkých škodlivých kategórií a druhov hrozieb, ktoré sa v danej testovacej sade nachádzajú.

6.4.1.1 k-NN a Local Outlier Factor

V prípade algoritmu k-NN ani jeden z „problémov“ (problém s detekciou PUA kategórie č. 1 u algoritmov Isolation forest a Autoenkóder, a problém s detekciou inej PUA kategórie u algoritmu Local Outlier Factor, ktoré budú analyzované v nasledujúcich častiach textu) nenastal a tento algoritmus dosahuje spomedzi všetkých modelov najlepšie výsledky. ROC krivky je možné nájsť na obr. B.1, ale aj na obr. B.3. Aj napriek tomu, žeby sa podľa obr. B.2 a obr. B.4, kde sú vykreslené distribúcie anomálneho skóre pre legitímnu a podozrivú (škodlivú) triedu mohlo zdať, že legitímna/normálna trieda a škodlivá trieda majú rovnaké anomálne skóre (v okolí bodu 0 na osi x), nie je tomu tak. V skutočnosti je medzi týmito triedami veľmi malý rozdiel v anomálnom skóre (stotiny), ale „stačí“ to na to, aby tieto dve triedy boli od seba oddelené správne.

Ak sa pozrieme na obr. B.5, resp. na ROC krivku pre algoritmus Local Outlier Factor uvidíme tam významný schod, resp. skok z hodnoty FPR približne 0.05 na 1, ktorý je spôsobený významnou zmenou FPR medzi dvoma prahovými hodnotami (pri minimálnom, resp. žiadnom rozdiely príslušných TPR hodnôt), čo by mohlo znamenať, že sa dátach v nachádza normálna (legitímna) trieda, ktorá má výrazne zastúpenie a veľmi podobné resp. rovnaké anomálne skóre, ale súčasne v dobe keď je FPR blízko hodnote 1 sa ešte mení (zvyšuje) TPR (spomínaný schod), čo môže znamenať, že sa v dátach nachádza kategória škodlivej triedy resp. kategórie tejto triedy, ktoré majú nižšie anomálne skóre ako normálna trieda, o čom sa je možné presvedčiť na obr. B.6, ale aj na obr. B.8, kde je možné vidieť, že v skutočnosti existuje kategória (kategórie) škodlivej triedy, ktorá má nižšie anomálne skóre ako normálna (legitímna) trieda. Po odstránení PUA kategórie č. 1 (ako je možné vidieť na obr. B.7) tento schod „nezmizol“ a navyše (pri porovnaní ROC kriviek B.5 a B.7) hodnota TPR (ak je FPR približne 0.5) klesla. Z tejto skutočnosti sa dá usúdiť to, že v tomto prípade nemá algoritmus Local Outlier Factor problém detegovať PUA kategóriu č. 1, keďže po odstránení tejto kategórie škodlivej triedy hodnota TPR klesla, čo znamená, že sme odstránili kategóriu pozitívnej (škodlivej) triedy, ktorú algoritmus detegoval správne. Pri hlbšej analýze sme zistili, že ide opäť o hrozbu typu PUA, avšak v tomto prípade sa jedná o iný typ hrozby spadajúcej do tejto PUA kategórie, ktorá má vyšší *risk* level (*medium risk* level), čo potvrdzujú aj tabuľky B.1 a B.6), kde je možné vidieť, že algoritmus Local Outlier Factor má problém správne detegovať škodlivé kategórie, ktoré majú *medium risk* level. To, že ide o hrozbu, ktorá spadá do škodlivej kategórie PUA a má *medium risk* level potvrdzujú tabuľky B.3 a B.8.

6.4.1.2 Isolation forest a Autoenkóder

Ak sa pozrieme na obr. B.9, kde je znázornená ROC krivka pre algoritmus Isolation forest uvidíme tam významný skok, resp. nárast hodnoty TPR približne z 0.3 na 0.95, ktorý je spôsobený významnou zmenou TPR medzi dvoma prahovými hodnotami, pričom rozdiel medzi príslušnými hodnotami FPR je minimálny, resp. žiadny (ak sa hodnota TPR zvyšuje tak buď sa hodnota FPR takisto zvyšuje alebo zostáva rovnaká). To by mohlo znamenať, že sa v testovacej množine nachádza konkrétna kategória pozitívnej (škodlivej) triedy, ktorej väčšina (alebo všetky) vzorky majú rovnaké anomálne skóre. Navyše by to mohlo znamenať, že je táto kategória škodlivej triedy v tejto trénovacej množine výrazne zastúpená, čo by sa mohlo prejaviť tak veľkou zmenou hodnoty TPR medzi dvoma príslušnými prahovými hodnotami, pri minimálnom (žiadnom) rozdiely medzi príslušnými FPR hodnotami. Ak sa pozrieme na distribúciu anomálneho skóre pre legitímnu a škodlivú triedu (obr. B.10) zistíme, že náš predpoklad je správny, a to síce, že sa v danej testovacej množine nachádza kategória (prípadne kategórie) škodlivej triedy, pretože na histograme je možné vidieť v okolí hodnoty -0.17 na osi x významný *peak*. Ak sa pozrieme na konkrétne rozdelenie a zastúpenie jednotlivých škodlivých kategórií v testovacej množine (tabuľka 6.2) zistíme, že kategória PUA, ktorá obsahuje 5 rôznych hrozieb typu PUA je zastúpená počtom 15 705 vzoriek a druhá najpočetnejšia škodlivá kategória je *ad injector* s počtom 1 136 vzoriek, čo by mohlo znamenať, že PUA je škodlivou kategóriou, ktorú je potrebné preskúmať, resp. presnejšie aké anomálne skóre je prideľované tejto konkrétnej škodlivej kategórií. Pri analýze anomálneho skóre, ktoré je prideľované tejto PUA kategórií sme zistili, že v tejto PUA kategórií sa nachádza jeden druh hrozby typu PUA, ktorý je zastúpený celkový počtom 14 597 vzoriek a že väčšina týchto vzoriek má rovnaké anomálne skóre, čím je možné vysvetliť tak významnú zmenu (skok) v hodnote TPR, v ROC krivke medzi dvoma prahovými hodnotami, pri minimálnom (žiadnom) rozdiely medzi príslušnými FPR hodnotami, a práve túto kategóriu (ako už bolo vyššie spomenuté) označujeme ako **PUA kategórie č. 1**. V prípade algoritmu Autoenkóder sa vyššie spomínaná skutočnosť takisto vyskytla ako je možné vidieť na obr. B.13 a na distribúcií anomálneho skóre (obr. B.14).

V oboch prípadoch je tento jav (výrazný nárast hodnoty TPR pri minimálnych (žiadnych) rozdieloch príslušných FPR hodnôt a príslušných prahových hodnôt) „nežiadúci“, pretože značí to o tom, že algoritmy nedokážu v tomto prípade správne detegovať konkrétnu hrozbu, ktorá patrí do škodlivej kategórie PUA. Na obr. B.11 je teda možné vidieť ROC krivku a na obr. B.12 distribúciu anomálneho skóre pre legitímnu a škodlivú triedu, pre algoritmus Isolation forest a na obr. B.15 ROC krivku a distribúciu anomálneho skóre na obr. B.16 pre algoritmus Autoenkóder už bez spomínanej PUA kategórie č. 1. To, že algoritmy Isolation forest a Autoekóder nie sú schopné správne detegovať tento druh hrozby patriacej do kategórie PUA sa presvedčíme aj

v podsekcií 6.4.2, kde si ukážeme metriky ako je TPR, precíznosť a FPR pre jednotlivé prahové hodnoty.

6.4.1.3 AUC skóre a doby behov jednotlivých algoritmov

Dosiahnuté AUC skóre pre jednotlivé algoritmy strojového učenia je možné nájsť v tabuľke 6.4 ako pre všetky škodlivé kategórie, tak aj pre prípad, že sme odobrali PUA kategóriu č. 1. Zmena (nárast) v AUC skóre, v prípade algoritmov Isolation forest a Autoenkóder potvrdzuje naše zistenie na základe príslušných ROC kriviek a to síce, že PUA kategória č. 1 bola kategóriou, ktorú oba algoritmy nedokázali správne detegovať, resp. klasifikovať.

V prípade algoritmov k-NN a Local Outlier Factor došlo po odstránení PUA kategórie č. 1 k poklesu AUC skóre, čo sa dá vysvetliť tak, že oba algoritmy dokázali spomínanú PUA kategóriu č. 1 správne detegovať. Výrazný pokles v AUC skóre u algoritmu Local Outlier Factor takisto potvrdzuje naše zistenie na základe príslušnej ROC krivky a príslušných tabuliek, a to síce, že tento algoritmus nie je schopný správne detegovať inú hrozbu zo škodlivej kategórie PUA, ktorá má, ale na rozdiel od kategórie PUA č. 1. *medium risk* level.

algoritmus	**AUC skóre**	
	všetky škodlivé kategórie	bez PUA kategórie č. 1
k-NN	0.980	0.978
LOF	0.918	0.793
IForest	0.623	0.828
Autoenkóder	0.639	0.848

Tabuľka 6.4: Výsledné AUC skóre podľa použitých algoritmov

Ak sa pozrieme na doby behov jednotlivých algoritmov (pozri tabuľku 6.5) zistíme, že najdlhšiu dobu behu dosiahol algoritmus k-NN, nasledovaný algoritmom Local Outlier Factor, Isolation forest a Autoenkóder, avšak rozdiel v dobe behu medzi algoritmom Isolation forest a Autoenkóder bol minimálny. Z vyššie uvedených výsledkov (AUC skóre) a z doby behu jednotlivých algoritmov sa dá usúdiť, že pri výbere toho „správneho“ algoritmu, resp. modelu máme dve možnosti a to síce, že buď chceme dosiahnuť lepšie výsledky za cenu dlhšej doby behu algoritmu, alebo horšie výsledky za cenu kratšej doby behu algoritmu. Použiteľnosť nášho skriptu a výber toho „správneho“ algoritmu, resp. modelu v produkčnom prostredí samozrejme závisí aj od toho aký hardvér (výpočtový výkon) máme k dispozícií, ale aj od toho v akom veľkom množstve prichádzajú na vstup jednotlivé (v našom prípade) *web proxy* záznamy. Z vyššie uvedených skutočností teda vyplýva, že použiteľnosť nášho skriptu v produkčnom prostredí je potrebné posudzovať prípad od prípadu a až pravdepodobne testovací režim v danom prostredí nám pomôže s výberom

toho „správneho“ algoritmu, resp. modelu pre dané prostredie v spojení s tým, čo chceme dosiahnuť, resp. aké sú očakávania.

doba behu	**algoritmus**			
	k-NN	LOF	IForest	Autoenkóder
trénovanie (*fit*) [s]	7971.273	8249.149	208.294	223.714
prideľovanie anomálneho skóre (*decision_function*) [s]	6165.658	2771.590	67.034	54.821

Tabuľka 6.5: Doba trénovania modelov a prideľovania anomálneho skóre podľa použitých algoritmov v sekundách

6.4.2 Výsledky pre jednotlivé prahové hodnoty

V tejto podsekcií si ukážeme ako sa menia hodnoty jednotlivých metrík, ak sa zvyšuje, resp. znižuje prahová hodnota (z angl. *threshold*). Nebudeme sa zaoberať hľadaním „najoptimálnejšej“ prahovej hodnoty, len si ukážeme výsledky metrík pre dve konkrétne prahové hodnoty 0.6 a 0.8. Výber tej „správnej“ prahovej hodnoty závisí od viacerých faktorov, resp. od toho, čo chceme dosiahnuť. Ako sme už viackrát spomínali výstupom skriptu (okrem rôznych metrík) je aj súbor s detegovanými anomáliami (*web proxy* záznamy, ktoré vzbudzujú podozrenie, že môžu byť škodlivé). Predstava je taká, že tento výstup by mal byť buď využívaný bezpečnostnými analytikmi ako sekundárny zdroj informácií alebo ako „zredukovaný“ výstup, ktorý bude vstupom do ďalších systémov určených na detailnejšiu analýzu bezpečnostných auditných záznamov (v tomto prípade *web proxy* záznamov). Z toho dôvodu by sme mali preferovať a zvoliť takú prahovú hodnotu, ktorá bude mať vyššiu hodnotu TPR aj za cenu vyššej hodnoty FPR.

6.4.2.1 Výsledky (metriky) detekcie anomálií podľa použitých algoritmov

Z tabuliek 6.6 a 6.7 sa dá vypozorovať to, že ak sa znižuje prahová hodnota tak (v prípade algoritmov Isolation forest a Autoenkóder) rastie TPR a aj FPR. To, že sa v prípade algoritmov k-NN a Local Outlier Factor nezmenilo ani TPR a ani FPR je spôsobené tým, ako počítame prahovú hodnotu (na základe percentilu), percentil sa síce zmenil, ale jeho vypočítaná hodnota sa nezmenila, čo znamená, že prahová hodnota sa v skutočnosti vôbec neposunula, resp. klasifikujeme rovnako v oboch prípadoch a to z dôvodu distribúcie anomálneho skóre pre legitímnu a škodlivú triedu, pretože legitímna (normálna) trieda má rovnaké, resp. veľmi podobné anomálne skóre a spôsob, ktorým počítame prahovú hodnotu (percentil) nám v tomto prípade túto prahovú hodnotu vôbec „neposúva“. Aby došlo k zmene, resp. posunu prahovej hodnoty museli by sme pravdepodobne vybrať percentil 0.9 a vyšší. Nízka hodnota metriky precíznosť (z angl. *precision*) by sa dala vysvetliť vysokým počtom FP a teda, že vzorky, ktoré v skutočnosti patria do negatívnej triedy

sú klasifikované ako pozitívne. Na druhej strane sa táto nízka hodnota tejto metriky dá odôvodniť pri pomere pozitívne : negatívne (podozrivé : legitímne) 1 : 100 a pri použití danej *semi-supervised* metódy, resp. techniky.

Tabuľky 6.6 a 6.7 mimo iného potvrdzujú naše slová uvedené v predchádzajúcich podsekciách 6.4.1.1 a 6.4.1.2 a to síce, že algoritmy k-NN a Local Outlier Factor sú schopné správne detegovať škodlivú kategóriu PUA pod označením PUA kategórie č. 1, keďže po jej odstránení došlo k poklesu hodnoty TPR. Na druhej strane algoritmy Isolation forest a Autoenkóder nie sú schopné správne detegovať túto kategóriu, keďže po jej odstránení sa v oboch prípadoch hodnota TPR zvýšila.

algoritmus	všetky škodlivé kategórie			bez PUA kategórie č. 1		
	TPR	**precíznosť**	**FPR**	**TPR**	**precíznosť**	**FPR**
k-NN	0.999	0.211	0.037	0.997	0.066	0.037
LOF	0.948	0.170	0.046	0.811	0.044	0.046
IForest	0.287	0.007	0.398	0.889	0.006	0.398
Autoenkóder	0.279	0.007	0.397	0.880	0.006	0.397

Tabuľka 6.6: Výsledky (metriky) detekcie anomálií podľa použitých algoritmov pre prahovú hodnotu 0.6

6.4.2.2 Výsledky (metriky TPR) podľa jednotlivých risk levelov

Ak by sme sa zamerali na metriku TPR podľa jednotlivých *risk* levelov (tabuľky B.1 a B.6) zistili by sme, že na zvolených prahových hodnotách nemá ani jeden z použitých algoritmov problém detegovať škodlivú (škodlivé) kategóriu (kategórie), ktoré majú vysoký (*high risk*) level. V našej testovacej množine (pozri tabuľku 6.2) má tento vysoký *risk* level len *information stealer*. Naopak algoritmy Isolation forest a Autoenkóder majú problém detegovať škodlivé kategórie, ktoré majú nízky (*low risk*) level, čo sa dalo očakávať, pretože už niekoľkokrát spomínaná PUA kategória č. 1 má práve tento nízky

algoritmus	všetky škodlivé kategórie			bez PUA kategórie č. 1		
	TPR	**precíznosť**	**FPR**	**TPR**	**precíznosť**	**FPR**
k-NN	0.999	0.211	0.037	0.997	0.066	0.037
LOF	0.948	0.170	0.046	0.811	0.044	0.046
IForest	0.239	0.012	0.200	0.781	0.010	0.198
Autoenkóder	0.248	0.012	0.199	0.768	0.010	0.199

Tabuľka 6.7: Výsledky (metriky) detekcie anomálií podľa použitých algoritmov pre prahovú hodnotu 0.8

(*low risk*) level. Avšak okrem problému s detekciou tejto PUA kategórie dosahujú algoritmy Isolation forest a Autoenkóder (podľa TPR (*risk* level) pozri tabuľky B.1 a B.6) pre *medium* a *high risk* level na konkrétnych dvoch prahových hodnotách (primárne teda u prahovej hodnoty 0.6) porovnateľné hodnoty/výsledky s algoritmami k-NN a Local Outlier Factor. Tieto tabuľky mimo iného potvrdzujú naše slová uvedené v podsekcií 6.4.1.1, ale aj v podsekcií 6.4.2.1 a to síce, že algoritmus Local Outlier Factor má problém detegovať typ hrozby škodlivej kategórie PUA, ktorá má *medium risk* level.

6.4.2.3 Výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov

Ak by sme sa ešte detailnejšie zamerali na schopnosť jednotlivých algoritmov detegovať jednotlivé kategórie škodlivej triedy podľa jednotlivých kategórií a *risk* levelov, a konkrétne na metriku TPR (pozri tabuľky B.2, B.3, B.4 a B.5 pre prahovú hodnotu 0.6, a tabuľky B.7, B.8, B.9 a B.10 pre prahovú hodnotu 0.8) opäť by sa potvrdila vyššie spomínaná skutočnosť, a to síce, že algoritmy Isolation forest a Autoenkóder nie sú schopné správne detegovať kategóriu PUA č. 1, ktorá má najnižší *risk* level, ale aj to, že algoritmus Local Outlier Factor nie je schopný detegovať škodlivú kategóriu PUA, ktorá má *medium risk* level. Pri ďalšej analýze sme zistili, že algoritmy Isolation forest a Autoenkóder nie sú schopné detegovať kategóriu škodlivej triedy *cryptojacking* (počet vzoriek v testovacej množine je 469) pre oba prahové hodnoty.

Pri snahe analyzovať dôvod toho, že algoritmy Isolation forest a Autoenkóder neboli schopné detegovať PUA kategórie č. 1 sme prišli na to, že niektoré z príznakov (z angl. *features*) obsahovali hodnoty, ktoré boli porovnateľné, resp. typické pre normálnu (legitímnu) triedu a nie pre pozitívnu (škodlivú) triedu. Jedným z príkladom je napr. vysoký tzv. Alexa Traffic Rank u príznaku *alexa_position* apod., čo by mohol byť jeden z dôvodov neschopnosti detegovať danú PUA kategórie č. 1. Pri snahe analyzovať neschopnosť detegovať kategóriu škodlivej triedy *cryptojacking* sme prišli na to, že takisto niektoré z príznakov obsahovali hodnoty porovnateľné, resp. typické pre normálnu (legitímnu) triedu a nie pre pozitívnu (škodlivú) triedu. Jedným z príkladov bol napr. príznak *page_rank*, ale aj vyššia hodnota príznaku *reputation*, čo by mohol byť takisto jeden z dôvodov neschopnosti detegovať danú škodlivú kategóriu.

6.5 Zhrnutie

Aj napriek komplikáciám s ktorými sme sa museli počas získavania záznamov vysporiadať sa nám podarilo získať bezpečnostné auditné záznamy (*web proxy* záznamy) z reálneho prostredia od firmy Cisco Systems, konkrétne od jedného z jej zákazníkov (stredná firma). Poskytnuté záznamy sú rozdelené do dvoch

súborov, podľa toho z akého časového obdobia pochádzajú. Prvý súbor, ktorého časť bola použitá ako trénovacia množina pochádza z 5. decembra 2019 a druhý súbor, ktorého časť bola použitá ako testovacia množina pochádza z 12. decembra 2019. Obsahom každého jedného riadku súboru je jeden *web proxy* záznam, ktorý je reprezentovaný pomocou 12 rôznych druhov/typov informácií a súčasťou sú aj štítky, ktoré označujú to, či je daný *web proxy* záznam legitímny (normálny) alebo škodlivý (podozrivý). Škodlivé *web proxy* záznamy ešte obsahujú informáciu o tom do ktorej zo 7 škodlivých kategórií patria, ale aj to aké majú *risk* skóre, resp. do akej kategórie (levelu) podľa tohto *risk* skóre patria. Po analýze *web proxy* záznamov, ktorej súčasťou bolo napr. zistenie konkrétneho rozdelenia trénovacej a testovacej množiny prišla na rad extrakcia príznakov a úprava rozsahu dát. Pri extrakcií príznakov sme vychádzali z rôznych informácií, ale primárne z adresy URL (príznak *URL*) a z IP adresy servera (príznak *ServerIP*). Po extrakcií príznakov nasledoval výber „najlepšej“ množiny príznakov na základe vypočítaného štandardného korelačného koeficientu a tzv. štandardizácia dát. Ďalším krokom bolo trénovanie jednotlivých modelov, resp. hľadanie množiny „optimálnych“ hyperparametrov pre jednotlivé modely pomocou krížovej validácie.

Po výbere resp. nájdení „optimálnych“ hyperparametrov prebiehala aplikácia a vyhodnotenie jednotlivých modelov, a to tým spôsobom, že na trénovacej množine (z 5. decembra 2019) a konkrétne len na legitímnych *web proxy* záznamoch (celkový počet vzoriek bol 3 012 293) sa uskutočnilo trénovanie jednotlivých algoritmov, resp. modelov a na testovacej množine (z 12. decembra 2019), kde už boli ako legitímne tak aj škodlivé *web proxy* záznamy (celkový počet legitímnych vzoriek bol 1 988 076 a škodlivých 19 854) sa uskutočnilo testovanie (prideľovanie anomálneho skóre jednotlivým *web proxy* záznamom). Na základe anomálneho skóre (spolu s informáciou do ktorej z tried (pozitívna alebo negatívna) patrí príslušný *web proxy* záznam) bola vykreslená ROC krivka, vypočítané AUC skóre, ale aj rôzne iné metriky ako TPR, precíznosť, FPR pre konkrétne dve prahové hodnoty 0.6 a 0.8. Doba behu jednotlivých algoritmov bola takisto jednou z metrík, ktorú sme merali.

Najlepšie AUC skóre bolo dosiahnuté algoritmom k-NN, ktorý bol nasledovaný algoritmom Local Outlier Factor, Autoenkóder a Isolation forest. Najdlhšiu dobu behu dosiahol algoritmus k-NN, nasledovaný algoritmom Local Outlier Factor, Isolation forest a Autoenkóder. Z vyššie uvedených výsledkov (AUC skóre) a z doby behu jednotlivých algoritmov sa dá usúdiť, že pri výbere toho „správneho“ algoritmu, resp. modelu máme dve možnosti a to síce, že buď chceme dosiahnuť lepšie výsledky za cenu dlhšej doby behu algoritmu, alebo horšie výsledky za cenu kratšej doby behu algoritmu, čo ovplyvňuje aj použiteľnosť daného skriptu v produkčnom prostredí. Algoritmom Isolation forest a Autoenkóder sa nepodarilo detegovať kategóriu škodlivej triedy, ktorú sme označili ako PUA kategórie č. 1, ale ani kategóriu škodlivej triedy *cryptojacking* pre jednotlivé prahové hodnoty. Na druhej strane algoritmu Local Outlier Factor sa nepodarilo detegovať kategóriu škodlivej triedy, resp. konkrétne

hrozbu PUA, ktorá má *medium risk* level. Pri snahe analyzovať dôvod toho, prečo sa jednotlivým algoritmom nepodarilo detegovať konkrétne hrozby sme prišli na to, že niektoré z príznakov obsahovali hodnoty typické pre normálnu triedu. Hľadaním „najoptimálnejšej“ prahovej hodnoty sme sa v tejto diplomovej práci nezaoberali, ale keďže výstupom nášho skriptu je súbor s detegovanými anomáliami a tento výstup bude buď použitý bezpečnostnými analytikmi ako sekundárny zdroj informácií alebo bude slúžiť ako vstup do ďalších systémov určených k detailnejšej analýze bezpečnostných auditných záznamov mali by sme zvoliť takú prahovú hodnotu, ktorá bude mať vyššiu hodnotu TPR aj za cenu vyššej hodnoty FPR.

Záver

Jedným z hlavných cieľov teoretickej časti tejto diplomovej práce bolo analyzovať nariadenie GDPR so zameraním sa na to, aké záznamy nariaďuje GDPR firmám auditovať. Pri analýze sme zistili, že vyslovene neexistuje žiadny článok resp. ustanovenie v nariadení GDPR, ktoré by hovorilo o tom aké záznamy nariaďuje GDPR firmám auditovať a z toho dôvodu sme sa venovali popisu rôznych odporúčaní a nariadení, ktoré je možné nájsť v podsekcií 3.3.3). Dôležitým výstupom tejto analýzy je tabuľka 3.3, ktorá prináša prehľad typov (zdrojov) záznamov a detegovaných udalostí, ktorá slúžila ako podklad pre praktickú časť tejto diplomovej práce. Jedným z ďalších cieľov teoretickej časti tejto diplomovej práce bolo priniesť čitateľovi obsiahlejší pohľad do problematiky záznamov, či už z hľadiska ich typov a rôznych formátov, ale aj z pohľadu ich zberu, spracovania a analýzy, čomu sa venujeme v kapitole 2. Využitiu strojového učenia k analýze záznamov a identifikácií podozrivej aktivity sa venujeme v sekcií 2.8.

Jedným z hlavných cieľov praktickej časti tejto diplomovej práce bolo vytvoriť *open-source* programový modul pomocou ktorého bude možné detegovať a identifikovať podozrivú aktivitu a ktorý zjednoduší, prípadne automatizuje prácu bezpečnostných analytikov. Pri návrhu a architektúre výsledného skriptu sme sa primárne zamerali na bezpečnostných analytikov, ktorí by mohli tento skript používať. Skript dostane na vstup bezpečnostné auditné záznamy (konkrétne *web proxy* záznamy) a jeho výstupom (okrem rôznych metrík a grafov) je súbor s detegovanými anomáliami (*web proxy* záznamy, ktoré vzbudzujú podozrenie žeby mohli byť škodlivé). Následne by tento výstup (súbor) mohol byť použitý ako sekundárny zdroj informácií, ktorý dokáže bezpečnostným analytikom pomôcť pri analýze záznamov a pri rozhodovaní a riešení rôznych bezpečnostných incidentov. Druhou možnosťou je využiť tento skript ako „automatizovaný“ filter, ktorý dokáže z veľkého množstva bezpečnostných auditných záznamov (v našom prípade *web proxy* záznamov) vyfiltrovať hrozby, ktoré sú relevantné a ktoré môžu byť použité ako vstup do ďalších systémov určených k ich detailnejšej analýze.

Pri aplikovaní návrhu a testovaní výsledného skriptu na *web proxy* záznamoch, ktoré pochádzali z reálneho prostredia od firmy Cisco Systems sme prišli na to, že najlepšie výsledky (*semi-supervised* technikou) dosahoval algoritmus k-NN, nasledovaný algoritmom Local Outlier Factor, Autoenkóder a Isolation forest. Horšie výsledky u algoritmov Autoenkóder a Isolation forest boli spôsobené neschopnosťou algoritmov detegovať konkrétny druh hrozby, ktorá patrí do kategórie PUA. Pri výbere jednotlivých modelov, resp. algoritmov a ich nasadenie do produkčného prostredia nás okrem dosiahnutých výsledkov zaujíma aj doba behu jednotlivých algoritmov, ktorá je takisto veľmi dôležitá. Hoci algoritmus k-NN dosiahol najlepšie výsledky z pohľadu metrík ako je AUC skóre, TPR atp. je potrebné dodať, že jeho doba behu bola najdlhšia spomedzi všetkých spomínaných algoritmov. Samozrejme všetko závisí aj od výpočtového výkonu, ale aj od toho v akom veľkom množstve prichádzajú na vstup jednotlivé *web proxy* záznamy. Z vyššie uvedených skutočností teda vyplýva, že použiteľnosť nášho skriptu v produkčnom prostredí je potrebné posudzovať prípad od prípadu a až pravdepodobne testovací režim v danom prostredí nám pomôže s výberom toho „správneho“ algoritmu, resp. modelu pre dané prostredie v spojení s tým, čo chceme dosiahnuť, resp. aké sú očakávania.

Ďalším logickým krokom by bolo spustiť tento skript (po ďalších iteráciách) v produkčnom prostredí (v testovacom režime) a zbierať spätnú väzbu od užívateľov, na základe ktorej by sme mohli tento skript ďalej vyvíjať. V súčasnom stave by rozšírenie výsledného skriptu mohlo obsahovať vylepšenia týkajúce sa extrakcie príznakov, a teda konkrétne prísť s novými príznakmi, ktoré by mohli detekciu anomálií, resp. detekciu podozrivej aktivity vo *web proxy* záznamoch vylepšiť. Keďže návrh výsledného skriptu bol navrhnutý tak, aby po ďalších iteráciách a vývoji mohol byť aplikovaný aj na ďalšie typy bezpečnostných auditných záznamov bolo by vhodné realizovať tento návrh napr. na NetFlow, systémové záznamy atp., avšak výraznou zmenou by musela prejsť extrakcia príznakov, ale aj ďalšie časti návrhu.

Výslednú implementáciu prototypu skriptu aplikovanú na *web proxy* záznamy spolu so všetkými výsledkami, ktorých súčasťou sú grafy a tabuľky je možné nájsť na priloženom médiu. Výsledky je takisto možné nájsť aj v dodatku tejto diplomovej práce. Na záver je potrebné dodať, že obsahom dodatku a ani priloženého média nie sú *web proxy* záznamy, ktoré podliehajú NDA. V prípade, že si užívateľ chce vyskúšať výsledný skript na „svojich“ *web proxy* záznamoch je potrebné postupovať podľa užívateľskej príručky, ktorá sa nachádza v dodatku C.

Oblasť strojového učenia a obzvlášť oblasť detekcie anomálií boli pre mňa veľkou výzvou, ale aj obrovskou skúsenosťou. Výzvy a problémy, s ktorými som sa stretol počas riešenia tejto diplomovej práce boli niečím, s čím som sa počas môjho štúdia na obore počítačová bezpečnosť veľmi nestretol, no na druhej strane je vidieť, že využitie strojového učenia má v oblasti počítačovej bezpečnosti svoj potenciál.

Literatúra

[1] Marsland, S.: *Machine learning: An algorithmic perspective*. London: Chapman and Hall/CRC, druhé vydání, 2014, ISBN 9781466583337, 1, 4–9 s.

[2] Chio, C.; Freeman, D.: *Machine Learning & Security*. O'Reilly, 2017, ISBN 9781491979907, 9, 12 s.

[3] Fisher, R.: UCI Machine Learning Repository - Iris data set [online]. 2017, [cit. 2019-05-02]. Dostupné z: `https://archive.ics.uci.edu/ml/datasets/Iris`

[4] Caruana, R.; Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning - ICML '06*, New York, New York, USA: ACM Press, 2006, ISBN 1595933832, s. 161–168, doi:10.1145/1143844.1143865. Dostupné z: `http://portal.acm.org/citation.cfm?doid=1143844.1143865`

[5] Silver, D.; Schrittwieser, J.; Simonyan, K.; aj.: Mastering the game of Go without human knowledge. *Nature Publishing Group*, ročník 550, 2017, doi:10.1038/nature24270. Dostupné z: `https://www.nature.com/articles/nature24270.pdf`

[6] Huang, S.: Introduction to Various Reinforcement Learning Algorithms. Part I (Q-Learning, SARSA, DQN, DDPG) [online]. *Medium*, 2018, [cit. 2019-07-15]. Dostupné z: `https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-i-q-learning-sarsa-dqn-ddpg-72a5e0cb6287`

[7] Chollet, F.: *Deep learning with Python*. New York: Manning Publications, 2017, ISBN 9781617294433, 6, 8–9, 20 s.

[8] Ramesh, V.: A Review on Application of Deep Learning in Thermography. *International Journal of Engineering and Management Research*, ročník 7, č. 3, 2017: s. 489–493, ISSN 2394-6962. Dostupné z: www.ijemr.nethttp://www.ijemr.net/DOC/AReviewOnApplicationOfDeepLearningInThermography.PDF

[9] Upwork: Log Analytics With Deep Learning And Machine Learning - Hiring — Upwork [online]. [cit. 2019-07-19]. Dostupné z: https://www.upwork.com/hiring/for-clients/log-analytics-deep-learning-machine-learning/

[10] Alexander Polyakov: Machine Learning for Cybersecurity 101 – Towards Data Science [online]. 2018, [cit. 2019-06-02]. Dostupné z: https://towardsdatascience.com/machine-learning-for-cybersecurity-101-7822b802790b

[11] The Recorded Future Team: Machine Learning: Practical Applications for Cybersecurity [online]. 2018, [cit. 2019-06-01]. Dostupné z: https://www.recordedfuture.com/machine-learning-cybersecurity-applications/

[12] Drinkwater, D.: 5 top machine learning use cases for security [online]. 2017, [cit. 2019-06-01]. Dostupné z: https://www.csoonline.com/article/3240925/5-top-machine-learning-use-cases-for-security.html

[13] Cisco Systems: What Is Machine Learning in Security? - Cisco [online]. [cit. 2019-06-01]. Dostupné z: https://www.cisco.com/c/en/us/products/security/machine-learning-security.html

[14] Cisco Systems: Encrypted Traffic Analytics. Technická zpráva, Cisco Systems, 2019. Dostupné z: https://www.cisco.com/c/dam/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encrytd-traf-anlytcs-wp-cte-en.pdf

[15] Ford, V.; Siraj, A.: Applications of Machine Learning in Cyber Security. In *27th International Conference on Computer Applications in Industry and Engineering*, October 2014, 2014, ISBN 9781880843970, str. 6.

[16] Chuvakin, A. A.; Schmidt, K. J.; Phillips, C.; aj.: *Logging and log management : the authoritative guide to understanding the concepts surrounding logging and log management.* Syngress, 2013, ISBN 1597496367, 2–3 s.

[17] Kent, K.; Souppaya, M.: Guide to Computer Security Log Management Recommendations of the National Institute of Standards and Technology. Technická zpráva, National Institute of Standards and Technology,

2006. Dostupné z: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-92.pdf

[18] Al-Fedaghi, S.; Mahdi, F.: Events Classification in Log Audit. *International Journal of Network Security & Its Applications (IJNSA)*, ročník 2, č. 2, 2010: s. 58–60, doi:10.5121/ijnsa.2010.2205.

[19] Splunk: Log Management — Log Analysis Monitoring Software — Splunk [online]. [cit. 2019-06-17]. Dostupné z: https://www.splunk.com/en_us/solutions/solution-areas/log-management.html

[20] Bhatt, S.; Manadhata, P. K.; Zomlot, L.: The Operational Role of Security Information and Event Management Systems. *IEEE Security & Privacy*, ročník 12, č. 5, sep 2014: s. 35–41, ISSN 1540-7993, doi:10.1109/MSP.2014.103. Dostupné z: http://ieeexplore.ieee.org/document/6924640/

[21] Cisco Systems: Network as a Security Sensor White Paper - Cisco [online]. 2015, [cit. 2019-11-26]. Dostupné z: https://www.cisco.com/c/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/white-paper-c11-736595.html

[22] Claise, B.; Trammell, B.; Aitken, P.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. *IETF RFC*, 2013: s. 1–63. Dostupné z: https://tools.ietf.org/html/rfc7011

[23] Berman, D.: Using Audit Logs for Security and Compliance — Logz.io [online]. 2018, [cit. 2019-06-07]. Dostupné z: https://logz.io/blog/audit-logs-security-compliance/

[24] Molenaar, R.: Cisco IOS Syslog Messages — NetworkLessons.com [online]. 2018, [cit. 2019-06-16]. Dostupné z: https://networklessons.com/cisco/ccie-routing-switching/cisco-ios-syslog-messages

[25] Brown, A.: Application Logging: What, When, How - DZone Java [online]. [cit. 2019-07-17]. Dostupné z: https://dzone.com/articles/application-logging-what-when

[26] Gregory, P.: *CISSP Guide to Security Essentials*. Cengage Learning, druhé vydání, 2015, ISBN 1285060423, 122,123 s.

[27] Gerhards, R.; GmbH, A.: RFC 5424 The Syslog Protocol. *Network Working Group, IETF*, 2009: s. 10–11. Dostupné z: https://tools.ietf.org/pdf/rfc5424.pdf

[28] Jansen, B. J.: *Understanding User-Web Interactions via Web Analytics: Bernard J. (Jim) Jansen, Pennsylvania State University*, ročník #6. Morgan & Claypool, 2009, ISBN 9781598298512, 25–26 s.

[29] Helmy, M.; Wahab, A.; Norzali, M.; aj.: Data Pre-processing on Web Server Logs for Generalized Association Rules Mining Algorithm. In *PROCEEDINGS OF WORLD ACADEMY OF SCIENCE, ENGINEERING AND TECHNOLOGY VOLUME 36 DECEMBER 2008 ISSN 2070-3740*, 2008, str. 8.

[30] Microsoft: IIS Log File Formats — Microsoft Docs [online]. 2017, [cit. 2019-06-20]. Dostupné z: https://docs.microsoft.com/en-us/previous-versions/iis/6.0-sdk/ms525807(v=vs.90)

[31] Microsoft: [MS-WMLOG]: cs-uri-stem — Microsoft Docs [online]. 2019, [cit. 2019-06-21]. Dostupné z: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-wmlog/fc4b49d7-e83f-4389-8063-414f8bad80dd

[32] Tutorialspoint: log4j Logging Levels [online]. [cit. 2019-06-22]. Dostupné z: https://www.tutorialspoint.com/log4j/log4j_logging_levels.htm

[33] Dietrich, E.: Logging Levels: What They Are and How They Help You — Scalyr [online]. 2017, [cit. 2019-06-22]. Dostupné z: https://www.scalyr.com/blog/logging-levels/

[34] Flowmon Networks a.s.: Flowmon & SIEM – Seamless Integration [online]. [cit. 2019-11-26]. Dostupné z: https://www.flowmon.com/getattachment/e6126e56-e6c4-4079-8185-03d87b1818b4/Flowmon-a-SIEM-systemy.aspx

[35] Syslog-ng: syslog-ng Open Source Edition 3.16 - Administration Guide [online]. [cit. 2019-06-23]. Dostupné z: https://www.syslog-ng.com/technical-documents/doc/syslog-ng-open-source-edition/3.16/administration-guide/3

[36] Lonvick, C.: Network Working Group C. Lonvick Request for Comments: 3164 Cisco Systems Category: Informational. *RFC*, 2001: str. 29. Dostupné z: https://tools.ietf.org/pdf/rfc3164.pdf

[37] Fluentd: What is Fluentd? — Fluentd [online]. [cit. 2019-06-23]. Dostupné z: https://www.fluentd.org/architecture

[38] Micro Focus: SIM, SEM, and SIEM: Definitions and Choosing the Right Enterprise Solution - Micro Focus Community - 1794733 [online]. [cit. 2019-09-30]. Dostupné z: https://community.microfocus.com/t5/

Security-Blog/SIM-SEM-and-SIEM-Definitions-and-Choosing-the-Right-Enterprise/ba-p/1794733

[39] Mooney, C. H.; Roddick, J. F.: Sequential pattern mining – approaches and algorithms. *ACM Computing Surveys*, ročník 45, č. 2, feb 2013: s. 1–39, ISSN 03600300, doi:10.1145/2431211.2431218. Dostupné z: http://dl.acm.org/citation.cfm?doid=2431211.2431218

[40] Scarfone, K.: SIEM benefits include efficient incident response, compliance [online]. 2015, [cit. 2019-06-15]. Dostupné z: https://searchsecurity.techtarget.com/feature/Three-enterprise-benefits-of-SIEM-products

[41] Gartner: Magic Quadrant for Security Information and Event Management [online]. [cit. 2019-07-11]. Dostupné z: https://www.rapid7.com/contentassets/103ddfbbf31943f183d6344f33a9c2b0/2018-siem-mq.jpg

[42] IBM: IBM QRadar SIEM Solution Brief. Technická zpráva, IBM, 2019. Dostupné z: https://www.ibm.com/downloads/cas/RLXJNX2G

[43] IBM: QRadar architecture overview [online]. [cit. 2019-07-11]. Dostupné z: https://www.ibm.com/support/knowledgecenter/en/SS42VS_7.3.2/com.ibm.qradar.doc/c_qradar_deployment_guide_arch.html

[44] LogRhythm: SIEM / Security Intelligence for MSP / MSSPs [online]. Technická zpráva, LogRhythm, 2015, [cit. 2019-09-30]. Dostupné z: https://gallery.logrhythm.com/data-sheets/security-intelligence-for-msp-data-sheet.pdf

[45] Comguard: Security Intelligence Platform [online]. Technická zpráva, Comguard, 2018, [cit. 2019-09-01]. Dostupné z: https://www.comguard.cz/www/upload/products/documents/20181228022741365.pdf

[46] MicroFocus: ArcSight Enterprise Security Manager [online]. 2018, [cit. 2019-07-09]. Dostupné z: https://www.microfocus.com/media/flyer/arcsight_enterprise_security_manager_ds.pdf

[47] Elastic: Feature-rich products [online]. [cit. 2019-08-31]. Dostupné z: https://www.elastic.co/products/

[48] Elahi, U.: Elastic Stack — A Brief Introduction – Hacker Noon [online]. 2018, [cit. 2019-06-15]. Dostupné z: https://hackernoon.com/elastic-stack-a-brief-introduction-794bc7ff7d4f

[49] Elastic: Aggregations [online]. [cit. 2019-08-31]. Dostupné z: https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html{#}search-aggregations

[50] Partlow, J.: Cloud Platform Security and Monitoring Make Security Possible ™. Technická zpráva, ReliaQuest, 2018. Dostupné z: https://dsimg.ubm-us.net/envelope/400093/570523/ReliaQuest-WhitePaper-CloudPlatformSecurityAndMonitoring.pdf

[51] AWS: Amazon CloudWatch - Application and Infrastructure Monitoring [online]. [cit. 2019-07-14]. Dostupné z: https://aws.amazon.com/cloudwatch/

[52] AWS: AWS CloudTrail – Amazon Web Services [online]. [cit. 2019-07-14]. Dostupné z: https://aws.amazon.com/cloudtrail/

[53] Microsoft: Přehled služby Azure Monitor — Microsoft Docs [online]. [cit. 2019-07-14]. Dostupné z: https://docs.microsoft.com/cs-cz/azure/azure-monitor/overview

[54] Unomaly: Unomaly — Algorithmic monitoring for the modern team [online]. [cit. 2019-07-18]. Dostupné z: https://unomaly.com/product/log-analysis-best-practice/

[55] Taylor, T.: Machine Learning and Log Analysis — Sumo Logic [online]. [cit. 2019-07-19]. Dostupné z: https://www.sumologic.com/blog/machine-learning-log-analysis/

[56] Endler, D.: Intrusion detection. Applying machine learning to Solaris audit data. In *Proceedings 14th Annual Computer Security Applications Conference (Cat. No.98EX217)*, IEEE Comput. Soc, 1998, ISBN 0-8186-8789-4, str. 1, doi:10.1109/CSAC.1998.738647. Dostupné z: http://ieeexplore.ieee.org/document/738647/

[57] Jain, L. C.; Patnaik, S.; Ichalkaranje, N.: Intelligent computing, communication and devices: Proceedings of ICCD 2014, volume 1. In *Advances in Intelligent Systems and Computing*, 2015, ISBN 9788132220114, ISSN 21945357, str. 231, doi:10.1007/978-81-322-2012-1.

[58] Suarez-Tangil, G.; Palomar, E.; Ribagorda, A.; aj.: Providing SIEM systems with self-adaptation. *Information Fusion*, ročník 21, jan 2015: s. 145–158, ISSN 1566-2535, doi:10.1016/J.INFFUS.2013.04.009. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1566253513000535

[59] Berlin, K.; Slater, D.; Saxe, J.: Malicious Behavior Detection using Windows Audit Logs. 2015, 1506.04200v2. Dostupné z: https://arxiv.org/pdf/1506.04200.pdf

[60] Mayhew, M.; Atighetchi, M.; Adler, A.; aj.: Use of machine learning in big data analytics for insider threat detection. In *MILCOM 2015 - 2015 IEEE Military Communications Conference*, IEEE, oct 2015, ISBN 978-1-5090-0073-9, s. 915–922eA, doi:10.1109/MILCOM.2015.7357562. Dostupné z: http://ieeexplore.ieee.org/document/7357562/

[61] Malaiya, R. K.; Kwon, D.; Kim, J.; aj.: An Empirical Evaluation of Deep Learning for Network Anomaly Detection. In *2018 International Conference on Computing, Networking and Communications, ICNC 2018*, January 2019, 2018, ISBN 9781538636527, s. 893–898, doi: 10.1109/ICCNC.2018.8390278.

[62] Bilge, L.; Balzarotti, D.; Robertson, W.; aj.: Disclosure: Detecting botnet command and control servers through large-scale NetFlow analysis. In *ACM International Conference Proceeding Series*, 2012, ISBN 9781450313124, s. 129–138, doi:10.1145/2420950.2420969.

[63] Kuna, H.; García-Martinez, R.; Villatoro, F.: Outlier detection in audit logs for application systems. *Information Systems*, ročník 44, aug 2014: s. 22–33, ISSN 03064379, doi:10.1016/j.is.2014.03.001. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0306437914000404

[64] He, S.; Zhu, J.; He, P.; aj.: Experience Report: System Log Analysis for Anomaly Detection. *Proceedings - International Symposium on Software Reliability Engineering, ISSRE*, 2016: s. 207–218, ISSN 10719458, doi: 10.1109/ISSRE.2016.21.

[65] Du, M.; Li, F.; Zheng, G.; aj.: DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning. In *ACM SIGSAC Conference*, 2017, ISBN 9781450349468, str. 14, doi:10.1145/3133956.3134015. Dostupné z: https://acmccs.github.io/papers/p1285-duA.pdf

[66] Tuor, A.; Kaplan, S.; Hutchinson, B.; aj.: Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data Streams. 2017, 1710.00811v2.

[67] Feng, C.; Wu, S.; Liu, N.: A user-centric machine learning framework for cyber security operations center. In *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, IEEE, jul 2017, ISBN 978-1-5090-6727-5, s. 173–175, doi:10.1109/ISI.2017.8004902.

[68] Rivas, G.: AI and SIEM: Increase the efficiency of your IT Security Team [online]. [cit. 2019-07-19]. Dostupné z: https://www.gb-advisors.com/ai-and-siem/

[69] Scarfone, K.: Give your SIEM system a power boost with machine learning [online]. [cit. 2019-07-20]. Dostupné z: https:

//searchsecurity.techtarget.com/tip/Give-your-SIEM-system-a-power-boost-with-machine-learning

[70] Reichenberg, N.: What Machine Learning Means for Security Operations — Siemplify [online]. [cit. 2019-07-20]. Dostupné z: https://www.siemplify.co/blog/what-machine-learning-means-for-security-operations/

[71] Úřad pro ochranu osobních údajů: GDPR (obecné nařízení): Úřad pro ochranu osobních údajů [online]. [cit. 2019-06-23]. Dostupné z: https://www.uoou.cz/gdpr/ds-3938/p1=3938

[72] Európsky parlament a Európska rada: NARIADENIE EURÓPSKEHO PARLAMENTU A RADY (EÚ) 2016/679. Technická zpráva, Európsky parlament a Európska rada, 2016. Dostupné z: https://eur-lex.europa.eu/legal-content/SK/TXT/PDF/?uri=CELEX:32016R0679&from=SK

[73] Nešpůrek, R.: Rozhodnutí Breyer a dynamická IP adresa jako osobní údaj [online]. 2017, [cit. 2019-08-25]. Dostupné z: https://www.pravniprostor.cz/clanky/obcanske-pravo/rozhodnuti-breyer-a-dynamicka-ip-adresa-jako-osobni-udaj

[74] Information Commissioner's Office: What is personal data? [online]. [cit. 2019-08-25]. Dostupné z: https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/key-definitions/what-is-personal-data/

[75] Úřad pro ochranu osobních údajů: Desatero zpracování pro správce: GDPR (obecné nařízení): Úřad pro ochranu osobních údajů [online]. [cit. 2019-06-23]. Dostupné z: https://www.uoou.cz/desatero-zpracovani-pro-spravce/ds-4821/archiv=0&p1=3938

[76] Úřad pro ochranu osobních údajů: Úřad pro ochranu osobních údajů: Titulní stránka [online]. [cit. 2020-01-18]. Dostupné z: https://www.uoou.cz/

[77] Európsky výbor pre ochranu osobných údajov: Obecné nařízení o ochraně osobních údajů: pokyny, doporučení, osvědčené postupy [online]. 2018, [cit. 2019-08-25]. Dostupné z: https://edpb.europa.eu/our-work-tools/general-guidance/gdpr-guidelines-recommendations-best-practices_cs

[78] PRACOVNÍ SKUPINA PRO OCHRANU ÚDAJŮ: Pokyny k automatizovanému individuálnímu rozhodování a profilování pro účely nařízení 2016/679. Technická zpráva, PRACOVNÍ SKUPINA PRO

OCHRANU ÚDAJŮ, 2018. Dostupné z: https://www.uoou.cz/assets/File.ashx?id_org=200144&id_dokumenty=31893

[79] Kamarinou, D.; Millard, C.; Singh, J.: Machine Learning with Personal Data. *Queen Mary University of London, School of Law: Legal Studies Research Paper*, ročník 246, 2016: s. 1–23.

[80] Úřad pro ochranu osobních údajů: K povinnosti provádět posouzení vlivu na ochranu osobních údajů (DPIA). Technická zpráva, Úřad pro ochranu osobních údajů, 2018. Dostupné z: https://www.uoou.cz/assets/File.ashx?id_org=200144&id_dokumenty=29003

[81] Bateman, R.: GDPR and Log Data - TermsFeed [online]. 2018, [cit. 2019-07-06]. Dostupné z: https://www.termsfeed.com/blog/gdpr-log-data/

[82] Európsky výbor pre ochranu osobných údajov: PRACOVNÍ SKUPINA PRO OCHRANU ÚDAJŮ ZŘÍZENÁ PODLE ČLÁNKU 29. Technická zpráva, Európsky výbor pre ochranu osobných údajov, 2017. Dostupné z: http://ec.europa.eu/newsroom/article29/news.cfm?item_type=1358&tpa_id=6936

[83] European Data Protection Board: Guidelines 4/2019 on Article 25 Data Protection by Design and by Default. Technická zpráva, European Data Protection Board, 2019.

[84] Tankard, C.: What the GDPR means for businesses. *Network Security*, ročník 2016, č. 6, jun 2016: s. 5–8, ISSN 1353-4858, doi:10.1016/S1353-4858(16)30056-3. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1353485816300563

[85] Úřad pro ochranu osobních údajů: Kodexy chování [online]. 2018, [cit. 2019-08-30]. Dostupné z: https://www.uoou.cz/kodexy-chovani/d-29493/p1=3938

[86] Úřad pro ochranu osobních údajů: Kodexy chování metodická příručka verze 2.0. Dostupné z: https://www.uoou.cz/assets/File.ashx?id_org=200144&id_dokumenty=37244

[87] Európsky výbor pre ochranu osobných údajov: Guidelines 1/2019 on Codes of Conduct and Monitoring Bodies under Regulation 2016/679 [online]. 2019, [cit. 2019-08-30]. Dostupné z: https://edpb.europa.eu/sites/edpb/files/consultation/edpb-20190219_guidelines_coc_public_consultation_version_en.pdf

[88] Ministerstvo práce a sociálních věcí České republiky: Doporučený postup č. 02/2018 - MPSV - GDPR - Kodex chování [online]. [cit. 2019-09-30]. Dostupné z: https://www.mpsv.cz/cs/13916

[89] Úrad na ochranu osobných údajov Slovenskej republiky: Schválené kódexy správania — Úrad na ochranu osobných údajov Slovenskej republiky [online]. [cit. 2019-08-30]. Dostupné z: https://dataprotection.gov.sk/uoou/sk/content/schvalene-kodexy-spravania

[90] Úřad pro ochranu osobních údajů: Certifikace, vydávání osvědčení. Technická zpráva, Úřad pro ochranu osobních údajů, 2017. Dostupné z: https://www.uoou.cz/certifikace-vydavani-osvedceni/d-27300/p1=3938

[91] Úřad pro ochranu osobních údajů: Kritéria pro vydávání osvědčení a kritéria pro akreditaci (KVO). Technická zpráva, Úřad pro ochranu osobních údajů, 2017. Dostupné z: https://www.uoou.cz/assets/File.ashx?id_org=200144&id_dokumenty=27997

[92] European Data Protection Board: Guidelines 1/2018 on certification and identifying certification criteria in accordance with Articles 42 and 43 of the Regulation Version history. Technická zpráva, European Data Protection Board, 2019. Dostupné z: https://edpb.europa.eu/sites/edpb/files/files/file1/edpb_guidelines_201801_v3.0_certificationcriteria_annex2_en.pdf

[93] European Data Protection Board: Guidelines 4/2018 on the accreditation of certification bodies under Article 43 of the General Data Protection Regulation (2016/679) Version history. Technická zpráva, European Data Protection Board, 2019. Dostupné z: https://edpb.europa.eu/sites/edpb/files/files/file1/edpb_guidelines_201804_v3.0_accreditationcertificationbodies_annex1_en.pdf

[94] Thies, S.: GDPR: Top 5 Logging Best Practices [online]. 2018, [cit. 2019-08-31]. Dostupné z: https://sematext.com/blog/gdpr-top-5-logging-best-practices/

[95] Nguyen, T.: GDPR Log Management – Compliant Logging Best Practices [online]. 2018, [cit. 2019-08-31]. Dostupné z: https://logdna.com/blog/best-practices-for-gdpr-logging/

[96] Národní Centrum Kybernetické Bezpečnosti: Doporučení na minimální požadavky pro logy, které musí být zajištěny pro spolehlivou ex-post analýzu kybernetických bezpečnostních incidentů. Technická zpráva, Národní Centrum Kybernetické Bezpečnosti, 2016. Dostupné z: https://www.govcert.cz/download/doporuceni/container-nodeid-1259/logmngmntfinal.pdf

[97] Národní úřad pro kybernetickou a informační bezpečnost: Kritická informační infrastruktura. Technická zpráva, Národní úřad pro kybernetickou a informační bezpečnost, 2018. Dostupné z: https://www.govcert.cz/download/kii-vis/Schema_KII.pdf

[98] Národní úřad pro kybernetickou a informační bezpečnost: Významné informační systémy. Technická zpráva, Národní úřad pro kybernetickou a informační bezpečnost, 2018. Dostupné z: https://www.govcert.cz/download/kii-vis/Schema_VIS.pdf

[99] Asociace za lepší ICT řešení: Oficiální GDPR certifikace [online]. [cit. 2019-08-31]. Dostupné z: https://lepsi-reseni.cz/ochrana-osobnich-udaju-gdpr/oficialni-gdpr-certifikace/

[100] TAYLLORCOX Czech: ISO 27701 Practitioner [online]. 2019, [cit. 2019-09-30]. Dostupné z: https://www.tx.cz/isms/iso-iec-27701-practitioner

[101] Advoqt: Identify all available log sources – SIEM Best Practices [online]. 2019, [cit. 2019-09-30]. Dostupné z: https://www.advoqt.com/identify-all-available-log-sources-siem-best-practices-2/

[102] Sparenberg, K.: Top 10 Log Sources You Should Monitor [online]. 2018, [cit. 2019-09-30]. Dostupné z: https://www.dnsstuff.com/top-10-log-sources-you-should-monitor

[103] Vijayan, J.: How to get your SIEM up to speed for GDPR [online]. 2018, [cit. 2019-09-01]. Dostupné z: https://techbeacon.com/security/how-get-your-siem-speed-gdpr

[104] Boucas, E.: Importance of Using SIEM for GDPR Compliance [online]. 2018, [cit. 2019-09-01]. Dostupné z: https://www.cpomagazine.com/cyber-security/importance-of-using-siem-for-gdpr-compliance/

[105] Subha: How to leverage SIEM to meet the GDPR's requirements [online]. 2018, [cit. 2019-09-01]. Dostupné z: https://blogs.manageengine.com/it-security/2018/07/20/leverage-siem-meet-gdprs-requirements.html

[106] LogManager: LOGmanager a soulad s požadavky GDPR. Technická zpráva, LogManager, 2018. Dostupné z: https://www.logmanager.cz/wp-content/uploads/2018/11/CZ_Whitepaper_GDPR_a_LOGmanager_v2.1.pdf

[107] Chandola, V.; Banerjee, A.; Kumar, V.: Anomaly detection: A survey. *ACM Reference Format*, ročník 41, č. 15, 2009: s. 1–11, doi: 10.1145/1541880.1541882. Dostupné z: http://doi.acm.org/10.1145/1541880.1541882

[108] Chalapathy, R.; Chawla, S.: Deep Learning for Anomaly Detection: A Survey. *Cluster Computing*, jan 2019: s. 1–9, 1901.03407. Dostupné z: http://arxiv.org/abs/1901.03407

[109] Hawkins, D. M.: *Identification of Outliers*. Dordrecht: Springer Netherlands, 1980, ISBN 978-94-015-3996-8, doi:10.1007/978-94-015-3994-4. Dostupné z: http://link.springer.com/10.1007/978-94-015-3994-4

[110] Goldstein, M.; Uchida, S.: A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLoS ONE*, ročník 11, č. 4, apr 2016, ISSN 19326203, doi:10.1371/journal.pone.0152173.

[111] Knorr, E. M.; Ng, R. T.: Algorithms for Mining Datasets Outliers in Large Datasets. *24th International Conference on Very Large Data Bases*, 1998: str. 393. Dostupné z: http://www.vldb.org/conf/1998/p392.pdf

[112] Ramaswamy, S.; Rastogi, R.; Shim, K.: Efficient algorithms for mining outliers from large data sets. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, ročník 29, č. 2, 2000: s. 427–429, ISSN 01635808, doi:10.1145/335191.335437.

[113] learn developers, S.: 1.6. Nearest Neighbors — scikit-learn 0.21.3 documentation [online]. 2019, [cit. 2019-12-02]. Dostupné z: https://scikit-learn.org/stable/modules/neighbors.html{#}unsupervised-neighbors

[114] Breuniq, M. M.; Kriegel, H. P.; Ng, R. T.; aj.: LOF: Identifying density-based local outliers. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, ročník 29, č. 2, 2000: s. 93–96, ISSN 01635808, doi:10.1145/335191.335388. Dostupné z: https://www.dbs.ifi.lmu.de/Publikationen/Papers/LOF.pdf

[115] Liu, F. T.; Ting, K. M.; Zhou, Z.-H.: Isolation-based Anomaly Detection. In *ACM Transactions on Knowledge Discovery from Data 6(1):1-39*, 2012, s. 3–4. Dostupné z: https://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/tkdd11.pdf

[116] Safavian, S. R.; Landgrebe, D.: A Survey of Decision Tree Classifier Methodology. *IEEE Transactions on Systems, Man and Cybernetics*, ročník 21, č. 3, 1991: str. 660, ISSN 21682909, doi:10.1109/21.97458.

[117] An, J.; Cho, S.: SNU Data Mining Center 2015-2 Special Lecture on IE Variational Autoencoder based Anomaly Detection using Reconstruction Probability. *The Journal of Machine Learning Research*, 2015: s. 3–4.

[118] Fan, Y. J.: Autoencoder Node Saliency: Selecting Relevant Latent Representations. *Pattern Recognition*, 2019: s. 644–645, 1711.07871v2.

[119] Aggarwal, C. C.: *Outlier Analysis*. Cham: Springer International Publishing, druhé vydání, 2017, ISBN 978-3-319-47577-6, 1–481 s., doi:10.1007/978-3-319-47578-3. Dostupné z: https://rd.springer.com/content/pdf/10.1007http://link.springer.com/10.1007/978-3-319-47578-3

[120] Alder, G.: draw.io – Online Diagramming [online]. 2019, [cit. 2019-12-11]. Dostupné z: https://about.draw.io/

[121] Spinner, T.; Körner, J.; Görtler, J.; aj.: Towards an Interpretable Latent Space : an Intuitive Comparison of Autoencoders with Variational Autoencoders. *IEEE VIS 2018*, oct 2018: str. 1. Dostupné z: https://thilospinner.com/towards-an-interpretable-latent-space/

[122] Géron, A.: *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*. O'Reilly Media, 2019, ISBN 9781492032649, 41 s. Dostupné z: http://shop.oreilly.com/product/0636920142874.do

[123] Google Developers: Classification: ROC Curve and AUC — Machine Learning Crash Course — Google Developers [online]. 2019, [cit. 2020-01-15]. Dostupné z: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

[124] Python Software Foundation: About Python™ — Python.org [online]. [cit. 2019-11-10]. Dostupné z: https://www.python.org/about/

[125] Github: The state of the octoverse 2019 [online]. [cit. 2019-11-09]. Dostupné z: https://octoverse.github.com/

[126] TIOBE Software BV: TIOBE Index — TIOBE - The Software Quality Company [online]. 2019, [cit. 2019-11-09]. Dostupné z: https://www.tiobe.com/tiobe-index/

[127] developers, S.: SciPy.org — SciPy.org [online]. [cit. 2019-11-10]. Dostupné z: https://www.scipy.org/

[128] Waskom, M.: Seaborn: Statistical Data Visualization — Seaborn 0.9.0 Documentation [online]. [cit. 2019-11-10]. Dostupné z: https://seaborn.pydata.org/

[129] learn developers, S.: scikit-learn: machine learning in Python — scikit-learn 0.21.3 documentation [online]. [cit. 2019-11-10]. Dostupné z: https://scikit-learn.org/stable/

[130] Abadi, M.; Agarwal, A.; Barham, P.; aj.: TensorFlow [online]. [cit. 2019-11-10]. Dostupné z: https://www.tensorflow.org/

[131] Google Brain Team: Brain Team – Google Research [online]. [cit. 2020-01-25]. Dostupné z: https://research.google/teams/brain/

[132] Apache: Apache License, Version 2.0 [online]. 2004, [cit. 2020-01-25]. Dostupné z: https://www.apache.org/licenses/LICENSE-2.0http://www.apache.org/licenses/LICENSE-2.0

[133] Chollet, F.; Tang, Y.; Studer, M.; aj.: Home - Keras Documentation [online]. [cit. 2019-11-10]. Dostupné z: https://keras.io/

[134] Rosebrock, A.: Keras vs. tf.keras: What's the difference in TensorFlow 2.0? - PyImageSearch [online]. [cit. 2019-11-10]. Dostupné z: https://www.pyimagesearch.com/2019/10/21/keras-vs-tf-keras-whats-the-difference-in-tensorflow-2-0/

[135] Belani, G.: Top 5 Deep Learning Frameworks for 2019 — Springboard Blog [online]. 2019, [cit. 2019-11-10]. Dostupné z: https://www.springboard.com/blog/deep-learning-frameworks/

[136] Maayan, G.: Top Deep Learning Frameworks of 2019 [online]. [cit. 2019-11-10]. Dostupné z: https://rubikscode.net/2019/08/12/top-deep-learning-frameworks-of-2019/

[137] Zhao, Y.; Nasrullah, Z.; Li, Z.: PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research*, ročník 20, 2019: s. 1–7, ISSN 1533-7928, 1901.01588. Dostupné z: https://pyod.readthedocs.iohttp://arxiv.org/abs/1901.01588

[138] Zhao, Y.: Welcome to PyOD documentation! — pyod 0.7.5 documentation [online]. [cit. 2019-11-19]. Dostupné z: https://pyod.readthedocs.io/en/latest/

[139] Sekera, J.: Switch from (pure) keras based on tensorflow to the keras loaded from the tensorflow library by sekerjak · Pull Request #145 · yzhao062/pyod [online]. [cit. 2020-01-25]. Dostupné z: https://github.com/yzhao062/pyod/pull/145

[140] Zhao, Y.: Scientific Work Using or Referencing PyOD [online]. [cit. 2019-12-10]. Dostupné z: https://pyod.readthedocs.io/en/latest/pubs.html{#}scientific-work-using-or-referencing-pyod

[141] Project Jupyter: Project Jupyter — Home [online]. 2017, [cit. 2019-11-10]. Dostupné z: https://jupyter.org/

[142] Avila, D.; Bussonnier, M.; Corlay, S.; aj.: Jupyter kernels · jupyter/jupyter Wiki · GitHub [online]. [cit. 2019-11-09]. Dostupné z: https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

[143] Rehak, M.; Anderson, B.: Securing Encrypted Traffic on a Global Scale - Cisco Blog [online]. 2018, [cit. 2020-01-06]. Dostupné z: https://blogs.cisco.com/security/securing-encrypted-traffic-on-worldwide-scale

[144] Sahoo, D.; Liu, C.; Hoi, S. C. H.: Malicious URL Detection using Machine Learning: A Survey. 2019, 1701.07179v3. Dostupné z: https://arxiv.org/pdf/1701.07179.pdf

[145] Amazon Web Services: AWS Marketplace: Alexa Top Sites [online]. [cit. 2020-01-07]. Dostupné z: https://aws.amazon.com/marketplace/pp/B07QK2XWNV

[146] Amazon Web Services: Alexa Top Sites [online]. [cit. 2020-01-07]. Dostupné z: https://www.alexa.com/topsites

[147] AWS: Alexa Traffic Rank [online]. [cit. 2020-01-07]. Dostupné z: http://s3.amazonaws.com/alexa-static/top-1m.csv.zip

[148] DomCop: What is Open PageRank? [online]. [cit. 2020-01-07]. Dostupné z: https://www.domcop.com/openpagerank/what-is-openpagerank

[149] DomCop: Open PageRank [online]. [cit. 2020-01-07]. Dostupné z: https://www.domcop.com/files/top/top10milliondomains.csv.zip

[150] MaxMind: Significant Changes to Accessing and Using GeoLite2 Databases — MaxMind Blog [online]. [cit. 2020-01-07]. Dostupné z: https://blog.maxmind.com/2019/12/18/significant-changes-to-accessing-and-using-geolite2-databases/

Dodatok **A**

Zoznam použitých skratiek

ACL - Access Control List

ANN - Artificial Neural Network

API - Application Programming Interface

APT - Advanced Persistent Threat

ASCII - American Standard Code for Information Interchange

AWS - Amazon Web Services

CEF - Common Event Format

CNN - Convolutional Neural Networks

CSIRT - Computer Security Incident Response Team

CSV - Comma-Separated Values

CWS - Cisco Web Security

ČSN - Česká Technická Norma

DBN - Deep Belief Network

DBSCAN - Density-Based Spatial Clustering of Applications with Noise

DDPG - Deep Deterministic Policy Gradient

DNC - Differentiable Neural Computer

DPIA - Data Protection Impact Assessment

DQN - Deep Q Network

DRBM - Deep Restricted Boltzmann Machine

ETA - Encrypted Traffic Analytics

EÚ - Európska Únia

FISMA - Federal Information Security Management Act

FN - False Negative

FP - False Positive

FPR - False Positive Rate

GAN - Generative Adversarial Networks

GDPR - General Data Protection Regulation

GLBA - Gramm–Leach–Bliley

HIPAA - Health Insurance Portability and Accountability Act

HTML - Hypertext Markup Language

HTTP - Hypertext Transfer Protocol

HTTPs - Hypertext Transfer Protocol Secure

ICO - Information Commissioner's Office

ICT - Information and Communication Technologies

IDS - Intrusion Detection Systems

IETF - Internet Engineering Task Force

IoC - Indicator of compromise

IOS - Internetwork Operating System

IoT - Internet of Things

IPFIX - IP Flow Information Export

IPS - Intrusion Prevention Systems

IS - Information System

ISO - International Organization for Standardization

JSON - JavaScript Object Notation

KII - Kritická Informační Infrastruktura

k-NN - k-Nearest Neighbors

LOF - Local Outlier Factor

MIME - Multipurpose Internet Mail Extensions

MPLS - Multiprotocol Label Switching

MTA - Mail Transfer Agent

NCKB - Národní Centrum Kybernetické Bezpečnosti

NCSA - National Center for Supercomputing Applications

NDA - Non-disclosure agreement

NLP - Natural Language Processing

NTM - Neural Turing Machines

NÚKIB - Národní úřad pro kybernetickou a informační bezpečnost

PCI DSS - Payment Card Industry Data Security Standard

PUA - Potentially Unwanted Application

REST - REpresentational State Transfer

RFC - Request For Comments

RNN - Recurrent Neural Network

ROC - Receiver Operating Characteristic

SARSA - State-Action-Reward-State-Action

SEM - Security Event Management

SIM - Security Information Management

SIEM - Security Information and Event Management

SNMP - Simple Network Management Protocol

SOC - Security Operations Center

SOM - Self-organized Maps

SOX - Sarbanes-Oxley

STIX - Structured Threat Information eXpression

SVM - Support Vector Machines

SVR - Support Vector Regression

TAXII - Trusted Automated eXchange of Intelligence

TLS - Transport Layer Security

TP - True Positive

TPR - True Positive Rate

TN - True Negative

UEBA - User and Entity Behavioral Analytics

URI - Uniform Resource Identifier

URL - Uniform Resource Locator

ÚOOÚ - Úřad pro ochranu osobních údajů

UTC - Coordinated Universal Time

VIS - Významné Informační Systémy

VPN - Virtual Private Network

XML - Extensible Markup Language

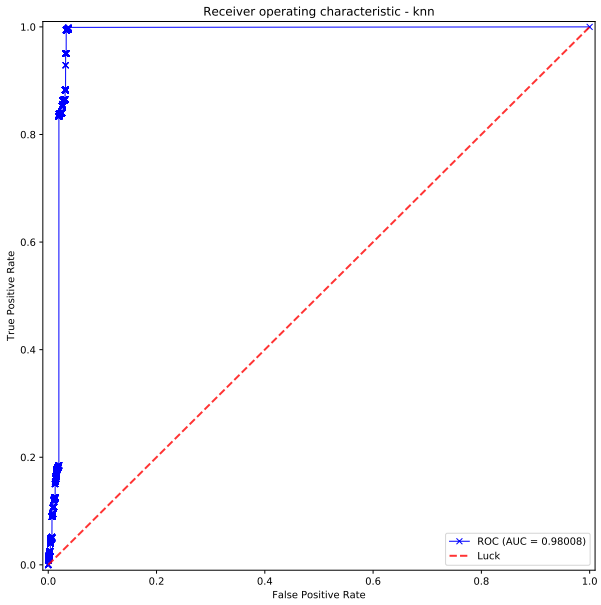
YAML - YAML Ain't Markup Language

Dodatok **B**

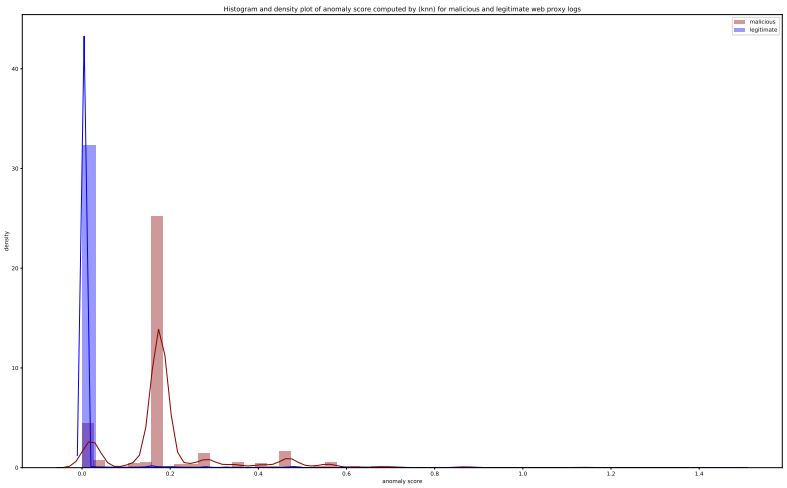
Výsledky jednotlivých modelov

Obsahom tohto dodatku sú:

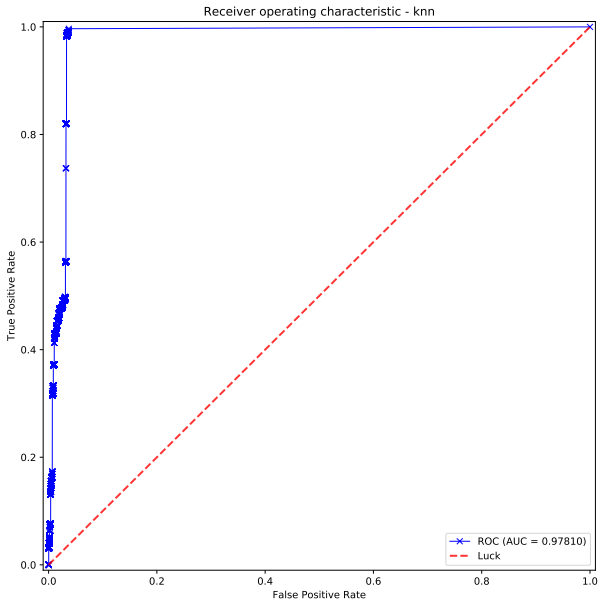
1. ROC krivky a histogramy (vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu) pre jednotlivé algoritmy a pre všetky škodlivé kategórie
   a) k-NN
   b) Local Outlier Factor
   c) Isolation forest
   d) Autoenkóder
2. ROC krivky a histogramy (vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu) pre jednotlivé algoritmy a bez PUA kategórie č. 1
   a) k-NN
   b) Local Outlier Factor
   c) Isolation forest
   d) Autoenkóder
3. Výsledky (metriky TPR) podľa jednotlivých risk levelov pre prahové hodnoty 0.6 a 0.8 pre jednotlivé algoritmy
4. Výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahové hodnoty 0.6 a 0.8 pre jednotlivé algoritmy



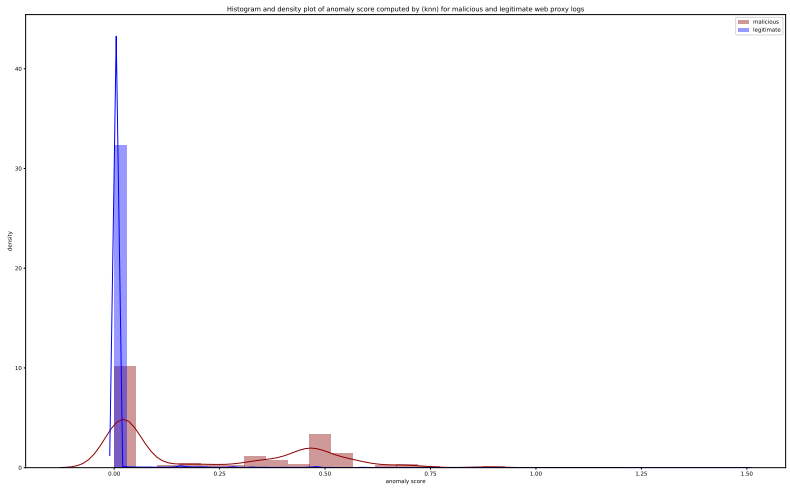
Obr. B.1: k-NN - ROC krivka (všetky škodlivé kategórie)



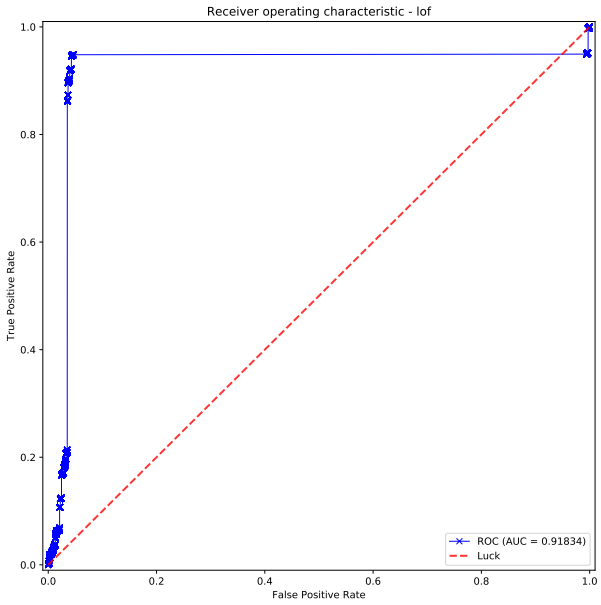
Obr. B.2: k-NN - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (všetky škodlivé kategórie)



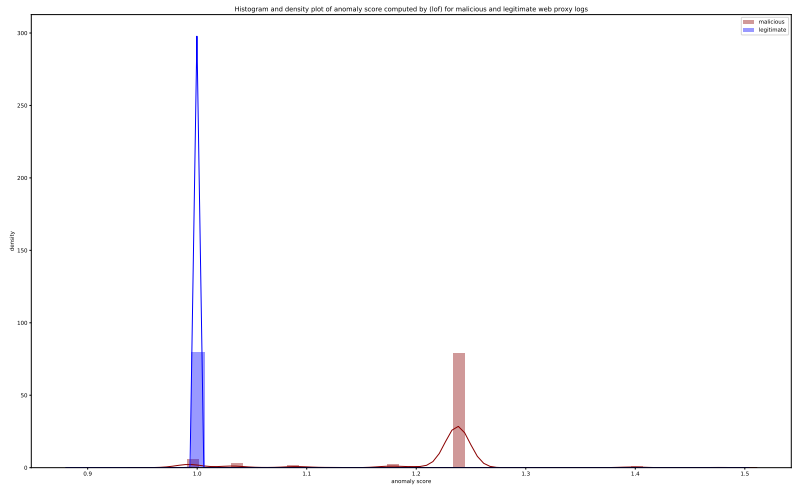
Obr. B.3: k-NN - ROC krivka (bez PUA kategórie č. 1)



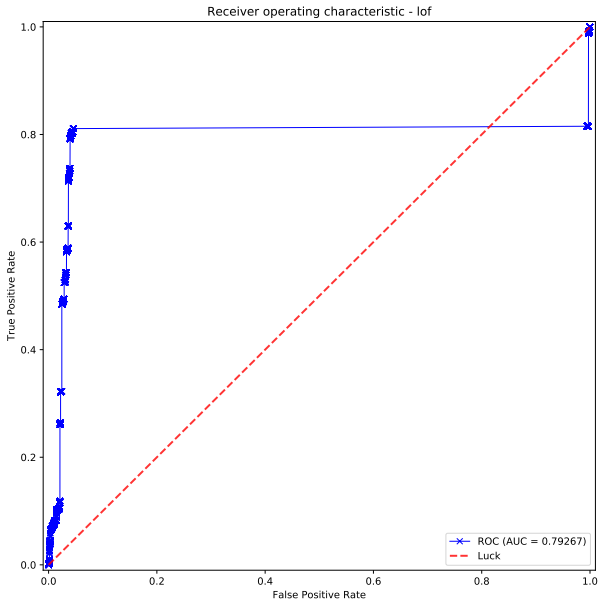
Obr. B.4: k-NN - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (bez PUA kategórie č. 1)



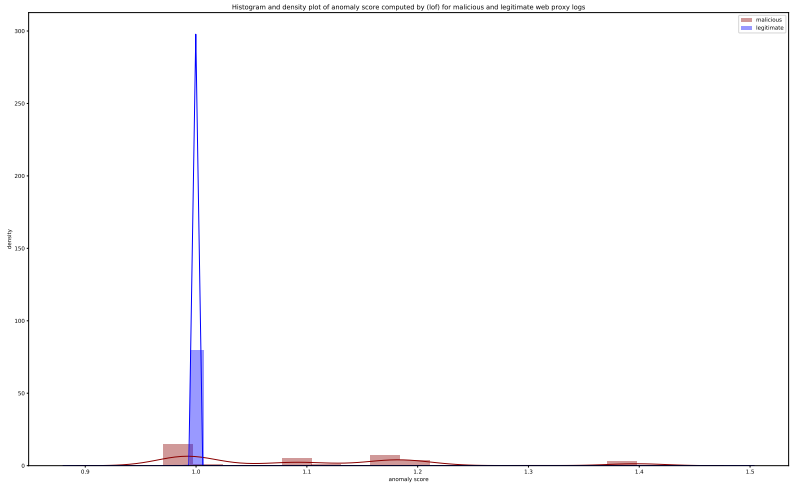
Obr. B.5: Local Outlier Factor - ROC krivka (všetky škodlivé kategórie)



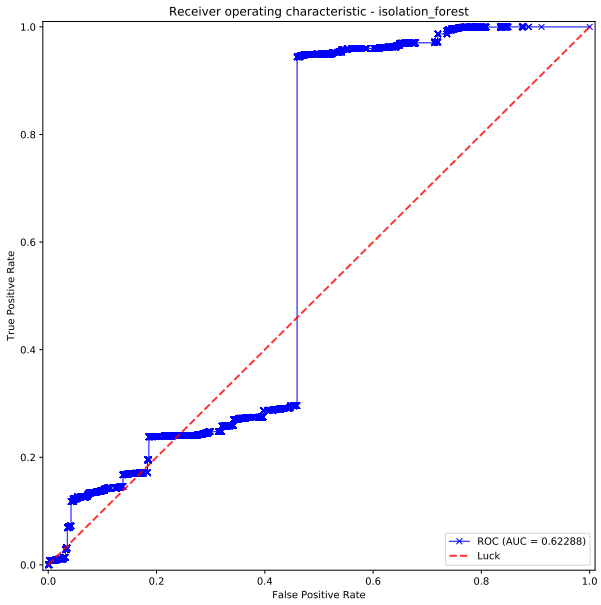
Obr. B.6: Local Outlier Factor - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (všetky škodlivé kategórie)



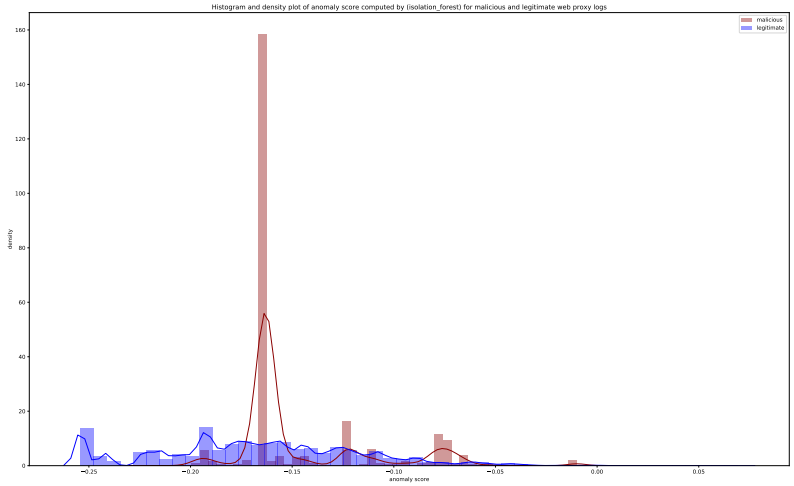
Obr. B.7: Local Outlier Factor - ROC krivka (bez PUA kategórie č. 1)



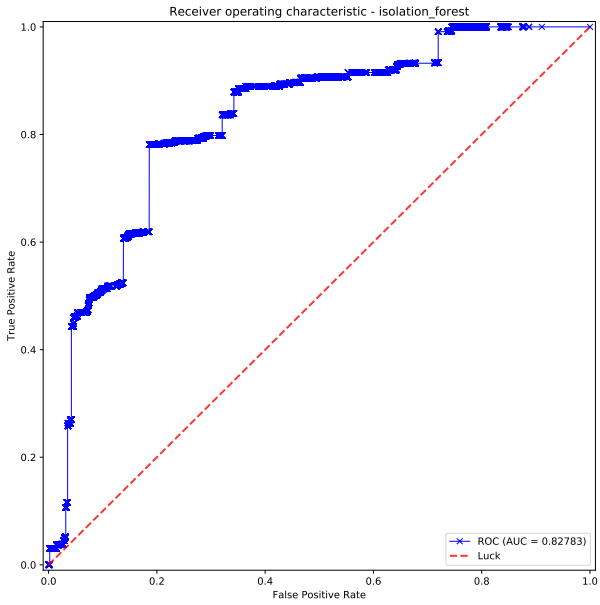
Obr. B.8: Local Outlier Factor - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (bez PUA kategórie č. 1)



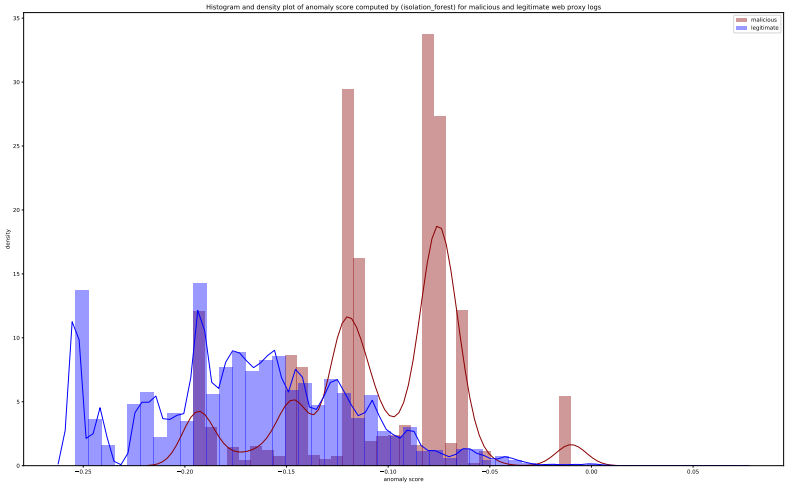
Obr. B.9: Isolation forest - ROC krivka (všetky škodlivé kategórie)



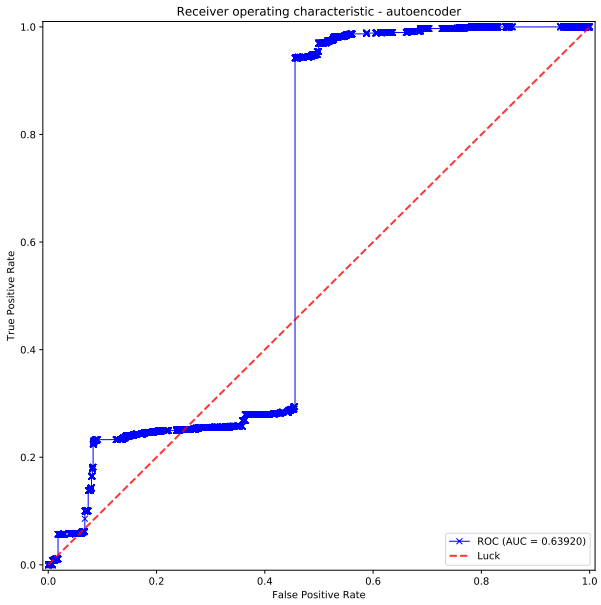
Obr. B.10: Isolation forest - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (všetky škodlivé kategórie)



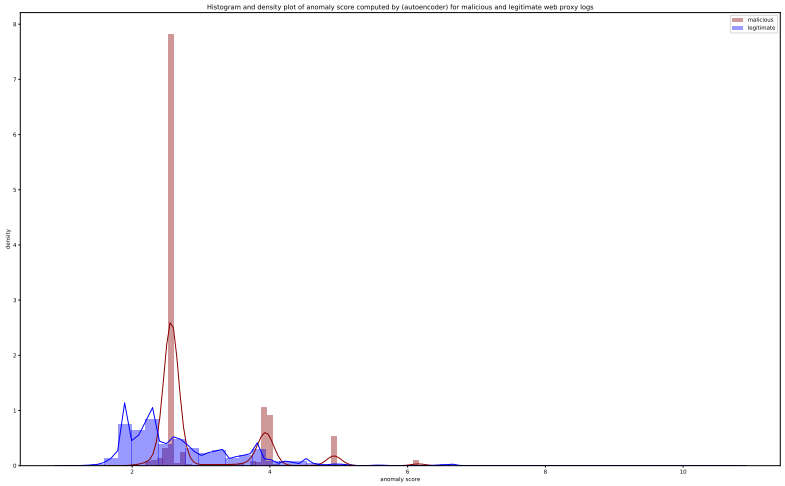
Obr. B.11: Isolation forest - ROC krivka (bez PUA kategórie č. 1)



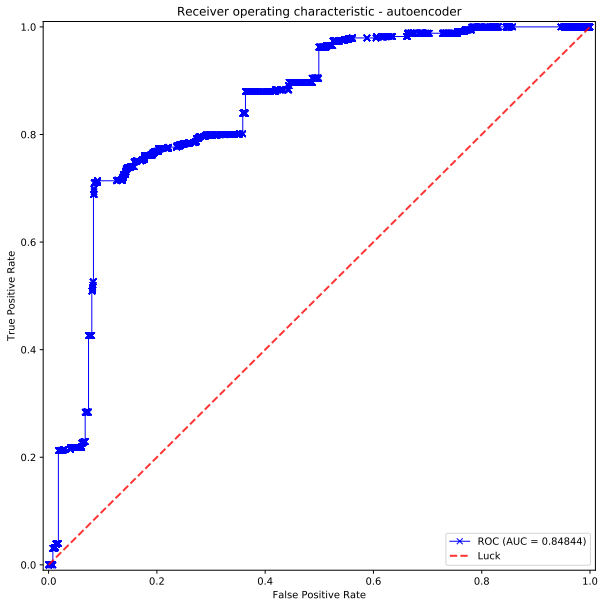
Obr. B.12: Isolation forest - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (bez PUA kategórie č. 1)



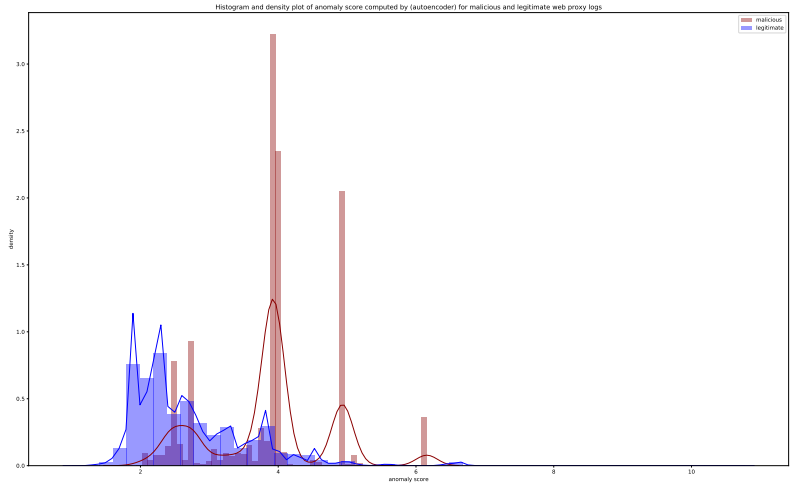
Obr. B.13: Autoenkóder - ROC krivka (všetky škodlivé kategórie)



Obr. B.14: Autoenkóder - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (všetky škodlivé kategórie)



Obr. B.15: Autoenkóder - ROC krivka (bez PUA kategórie č. 1)



Obr. B.16: Autoenkóder - vizualizácia distribúcie anomálneho skóre pre legitímnu a škodlivú triedu (bez PUA kategórie č. 1)

algoritmus	všetky škodlivé kategórie			bez kategórie PUA č. 1		
	TPR (risk level)			**TPR (risk level)**		
	low	medium	high	low	medium	high
k-NN	1.0	0.996	1.0	0.996	0.996	1.0
LOF	0.994	0.726	1.0	0.966	0.726	1.0
IForest	0.138	0.962	1.0	0.730	0.962	1.0
Autoenkóder	0.131	0.950	1.0	0.724	0.950	1.0

Tabuľka B.1: Výsledky (metriky TPR) podľa jednotlivých risk levelov pre prahovú hodnotu 0.6

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	1.0	1.0	1.0	x	1.0	1.0	1.0	x
ad injector	0.989	x	0.989	x	0.989	x	0.989	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	0.867	0.867	x	x	0.867	0.867	x	x
cryptojacking	1.0	1.0	x	x	1.0	1.0	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	1.0	1.0	x	x	1.0	1.0	x	x

Tabuľka B.2: k-NN – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.6

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	0.947	0.998	0.594	x	0.727	1.0	0.594	x
ad injector	0.989	x	0.989	x	0.989	x	0.989	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	0.867	0.867	x	x	0.867	0.867	x	x
cryptojacking	0.915	0.915	x	x	0.915	0.915	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	0.783	0.783	x	x	0.783	0.783	x	x

Tabuľka B.3: Local Outlier Factor – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.6

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	0.245	0.136	0.998	x	0.999	1.0	0.998	x
ad injector	0.889	x	0.889	x	0.889	x	0.889	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	1.0	1.0	x	x	1.0	1.0	x	x
cryptojacking	0.055	0.055	x	x	0.055	0.055	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	0.826	0.826	x	x	0.826	0.826	x	x

Tabuľka B.4: Isolation forest – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.6

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	0.238	0.129	0.984	x	0.989	1.0	0.984	x
ad injector	0.880	x	0.880	x	0.880	x	0.880	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	1.0	1.0	x	x	1.0	1.0	x	x
cryptojacking	0.055	0.055	x	x	0.055	0.055	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	0.609	0.609	x	x	0.609	0.609	x	x

Tabuľka B.5: Autoenkóder – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.6

algoritmus	všetky škodlivé kategórie			bez kategórie PUA č. 1		
	TPR (risk level)			**TPR (risk level)**		
	low	medium	high	low	medium	high
k-NN	1.0	0.996	1.0	0.996	0.996	1.0
LOF	0.994	0.726	1.0	0.966	0.726	1.0
IForest	0.110	0.814	1.0	0.694	0.814	1.0
Autoenkóder	0.127	0.787	1.0	0.708	0.787	1.0

Tabuľka B.6: Výsledky (metriky TPR) podľa jednotlivých risk levelov pre prahovú hodnotu 0.8

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	1.0	1.0	1.0	x	1.0	1.0	1.0	x
ad injector	0.989	x	0.989	x	0.989	x	0.989	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	0.867	0.867	x	x	0.867	0.867	x	x
cryptojacking	1.0	1.0	x	x	1.0	1.0	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	1.0	1.0	x	x	1.0	1.0	x	x

Tabuľka B.7: k-NN – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.8

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	0.947	0.998	0.594	x	0.727	1.0	0.594	x
ad injector	0.989	x	0.989	x	0.989	x	0.989	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	0.867	0.867	x	x	0.867	0.867	x	x
cryptojacking	0.915	0.915	x	x	0.915	0.915	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	0.783	0.783	x	x	0.783	0.783	x	x

Tabuľka B.8: Local Outlier Factor – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.8

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	0.220	0.111	0.970	x	0.980	1.0	0.970	x
ad injector	0.499	x	0.499	x	0.499	x	0.499	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	0.622	0.622	x	x	0.622	0.622	x	x
cryptojacking	0.0	0.0	x	x	0.0	0.0	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	0.435	0.435	x	x	0.435	0.435	x	x

Tabuľka B.9: Isolation forest – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.8

kategória	všetky škodlivé kategórie				bez PUA kategórie č. 1			
	TPR	**TPR (risk level)**			**TPR**	**TPR (risk level)**		
		low	med.	high		low	med.	high
PUA	0.234	0.126	0.974	x	0.983	1.0	0.974	x
ad injector	0.409	x	0.409	x	0.409	x	0.409	x
inf. stealer	1.0	x	1.0	1.0	1.0	x	1.0	1.0
malvertising	0.711	0.711	x	x	0.644	0.644	x	x
cryptojacking	0.049	0.049	x	x	0.049	0.049	x	x
mal. cont. dist.	1.0	1.0	x	x	1.0	1.0	x	x
undefined	0.435	0.435	x	x	0.435	0.435	x	x

Tabuľka B.10: Autoenkóder – výsledky (metriky TPR) podľa jednotlivých kategórií a risk levelov pre prahovú hodnotu 0.8

Dodatok C

Užívateľská príručka

Celý vývoj výsledného skriptu a príslušných Jupyter notebookov prebiehal na operačnom systéme macOS Catalina verzie 10.15.2. Nasledujúci postup, ktorý je obsahom užívateľskej príručky bol takisto otestovaný na tomto operačnom systéme. Ďalšie *Unix-like* systémy otestované neboli, ale pravdepodobne až na zmenu cesty k balíčku *pyod* v bode 5 by nižšie uvedený postup mal byť rovnaký, resp. veľmi podobný.

Je dôležité poznamenať, že súčasťou priloženého CD **nie sú** *web proxy* záznamy, pretože podliehajú NDA. Tým pádom nie je možné spustiť/otestovať výsledný skript, pretože nie sú splnené všetky prerekvizity spomenuté v bode 3. Po splnení všetkých prerekvizít spomenutých v bode 3 bude možné výsledný skript spustiť/otestovať, avšak za predpokladu, že *web proxy* záznamy budú vo formáte uvedenom v bodoch 3d a 3e. Navyše platí, že formát *web proxy* záznamov sa odvíja od toho, či sú alebo nie sú k dispozícií štítky, resp. oštítkované *web proxy* záznamy. *Web proxy* záznamy (použité v tejto diplomovej práci) boli exportované zo systému/produktu CWS (Cloud Web Security) od firmy Cisco Systems a obsahujú okrem iných príznakov aj „špecifický“ príznak *reputation*, no je potrebné dodať, že tento príznak by sa dal podobne „vyrobiť“ aj z iných systémov, ktoré spracovávajú *web proxy* záznamy. Z toho vyplýva, že je buď potrebné exportovať *web proxy* záznamy zo spomínaného produktu alebo zabezpečiť to, aby *web proxy* záznamy mali rovnaký formát ako je uvedené v bode 3d a 3e.

V prípade, že nie sú k dispozícií *web proxy* záznamy odporúčame sa pozrieť na jednotlivé Jupyter notebooky vo formáte HTML (aj s detailnými komentármi), ktoré sú súčasťou priloženého CD v adresári **src/impl/jupyter** (ich obsah je uvedený v podsekcií 5.1).

Postup inštalácie, ktorého súčasťou je aj zoznam prerekvizít je popísaný v nasledujúcich krokoch:

1. skopírujeme adresár **src** (z priloženého CD) na cieľový systém, na ktorom chceme nainštalovať výsledný skript

2. prejdeme do adresára **impl** za predpokladu, že máme terminál otvorený v adresári, kde sa nachádza adresár **src**

   ```
   $ cd src/impl
   ```

3. splníme **prerekvizity**, ktoré musí obsahovať adresár **web_proxy_logs**, ktorý sa nachádza v príslušnom **impl** adresári

   a) adresár **alexa** musí obsahovať súbor pomenovaný ako *top-1m.csv*, tento súbor je súčasťou priloženého CD (zoznam je možné stiahnuť z odkazu [147])

   b) adresár **open_page_rank** musí obsahovať súbor pomenovaný ako *top10milliondomains.csv*, tento súbor je súčasťou priloženého CD (súbor/zoznam je možné stiahnuť z odkazu [149])

   c) adresár **GeoLite2-ASN** musí obsahovať súbor pomenovaný ako *GeoLite2-ASN.mmdb*, tento súbor **nie je** súčasťou priloženého CD (databázu je možné stiahnuť po registrácií z portálu [150])

   d) adresár **train_dataset** musí obsahovať súbor, ktorý je pomenovaný ako *train_dataset.txt*, tento súbor **nie je** súčasťou priloženého CD z dôvodu NDA, avšak v prípade, že chceme otestovať výsledný skript musíme mať *web proxy* záznamy v nasledujúcom formáte:

      i. v prípade, že máme k dispozícií oštítkovanú dátovú sadu s *web proxy* záznamami musí tento súbor obsahovať všetky príznaky uvedené v podsekcií 6.2.1 a príznaky (*label*, *outcome*, *category* a *risk level*) uvedené v podsekcií 6.3.1 a príslušné hodnoty, ukážka toho, ako by taký *web proxy* záznam mal vyzerať je na obr. 6.2 (jednotlivé položky sú od seba oddelené pomocou tabulátora)

      ii. v prípade, že nemáme k dispozícií oštítkovanú dátovú sadu **nemusí** tento súbor obsahovať príznaky *label*, *outcome*, *category* a *risk level*

   e) adresár **test_dataset** musí obsahovať súbor, ktorý je pomenovaný ako *test_dataset.txt* a ktorého štruktúra je rovnaká ako je uvedené v predchádzajúcom bode, (tento súbor **nie je** súčasťou priloženého CD z dôvodu NDA)

   f) do adresárov **whois_ASN** a **whois_reg_date** vytvorí skript príslušné súbory na základe toho, že budú dostupné vyššie spomínané adresáre a ich obsah (obsah adresárov **nie je** obsahom priloženého CD z dôvodu NDA)

4. za predpokladu, že máme nainštalovaný *python3* a príslušný *pip* (správca balíčkov pre moduly programovacieho jazyka Python) vytvoríme virtuálne prostredie a nainštalujeme potrebné Python balíčky

```
$ python3 -m pip install --user -U virtualenv
$ python3 -m virtualenv env
$ source env/bin/activate
$ python3 -m pip install -U -r requirements.txt
```

5. vykonáme zmenu v kóde balíčka *pyod*, v prípade, že sa ešte neuzavrel nasledujúci *pull request* `https://github.com/yzhao062/pyod/pull/145`

   ```
   vi env/lib/python3.7/site-packages/pyod/models/auto_encoder.py
   ```

   ```
   -from keras.models import Sequential
   -from keras.layers import Dense, Dropout
   -from keras.regularizers import l2
   -from keras.losses import mean_squared_error

   +from tensorflow.keras.models import Sequential
   +from tensorflow.keras.layers import Dense, Dropout
   +from tensorflow.keras.regularizers import l2
   +from tensorflow.keras.losses import mean_squared_error
   ```

6. prejdeme do adresára **script**

   ```
   $ cd script
   ```

7. spustíme skript

   - nápoveda

     ```
     $ python3 web_proxy_logs_demo.py --help
     ```

   - 1. príklad (v prípade, že nie je k dispozícií oštítkovaná dátová sada)

     ```
     $ python3 web_proxy_logs_demo.py --threshold 0.6 0.8
     ↪  --model AE IF KNN LOF --labels no --train_dataset
     ↪  "../web_proxy_logs/train_dataset/train_dataset.txt"
     ↪  --test_dataset
     ↪  "../web_proxy_logs/test_dataset/test_dataset.txt"
     ```

   - 2. príklad (v prípade, že oštítkovaná dátová sada je k dispozícií)

     ```
     $ python3 web_proxy_logs_demo.py --hyperparameters
     ↪  config.yaml --threshold 0.4 0.6 --model AE LOF
     ↪  --labels yes --train_dataset
     ↪  "../web_proxy_logs/train_dataset/train_dataset.txt"
     ↪  --test_dataset
     ↪  "../web_proxy_logs/test_dataset/test_dataset.txt"
     ```

8. výstup skriptu sa ukladá do adresára **results** a jeho štruktúru je možné vidieť v obsahu priloženého CD (pozri dodatok D), v prípade, že nie je k dispozícií oštítkovaná dátová sada bude výstupom „len“ súbor s detegovanými anomáliami bez grafov a metrík

9. (voliteľné) v prípade, že sa chceme pozrieť na jednotlivé Jupyter notebook/y, zmeníme adresár a spustíme Jupyter notebook

```
$ cd ../jupyter
$ jupyter notebook
```

Na záver je potrebné dodať, že obsahom priloženého CD je súbor *README.md*, ktorého obsahom je užívateľská príručka v anglickom jazyku.

Dodatok D

Obsah priloženého CD

```
DP_Sekera_Jakub_2020
├── src
│   ├── impl ..... praktická časť diplomovej práce
│   │   ├── jupyter ..... jednotlivé Jupyter notebooky
│   │   ├── requirements.txt ..... zoznam potrebných Python balíčkov
│   │   ├── script ..... výsledný skript a príslušné súbory
│   │   │   ├── anomaly_detection ..... zdrojové kódy implementácie
│   │   │   │   ├── __init__.py
│   │   │   │   ├── dataloader.py ..... načítanie vstupu/dát
│   │   │   │   ├── feature_extraction.py ..... extrakcia príznakov
│   │   │   │   ├── input_parsing.py ..... spracovanie vstupu
│   │   │   │   ├── models.py ..... jednotlivé modely
│   │   │   │   └── visualization_and_results.py . vizualizácia a výsledky
│   │   │   ├── config.yaml ..... hyperparametre pre jednotlivé modely
│   │   │   ├── results ..... výsledky jednotlivých modelov
│   │   │   │   ├── autoencoder
│   │   │   │   ├── isolation_forest
│   │   │   │   ├── knn
│   │   │   │   └── lof
│   │   │   └── web_proxy_logs_demo.py ..... hlavná časť/súbor skriptu
│   │   ├── thesis ..... zdrojová forma práce vo formáte LaTeX
│   │   └── web_proxy_logs ... adresár obsahujúci adresáre s prerekvizitami
│   ├── LICENSE ..... GNU GPL licencia v3.0
│   └── README.md ..... základné informácie o skripte
└── text ..... text práce
    └── DP_Sekera_Jakub_2020.pdf ..... text práce vo formáte PDF
```

- `src`
  - `impl`
    - `script`
      - `results`
        - `kNN`
          - `all_categories` ..... všetky škodlivé kategórie
            - `0.6` ..... výsledky pre prahovú hodnotu 0.6
              - `confusion_matrix.pdf` ..... konfúzna matica
              - `results.txt` ..... metriky TPR, precíznosť a FPR
            - `0.8` ..... výsledky pre prahovú hodnotu 0.8
              - `confusion_matrix.pdf` ..... konfúzna matica
              - `results.txt` ..... metriky TPR, precíznosť a FPR
            - `histogram.pdf` ... vizualizácia distribúcie anomálneho skóre
            - `results.txt` ..... AUC skóre a doba behu algoritmu
            - `roc_curve.pdf` ..... ROC krivka
          - `without_PUA_category_1` ..... bez PUA kategórie č. 1
            - `0.6` ..... výsledky pre prahovú hodnotu 0.6
              - `confusion_matrix.pdf` ..... konfúzna matica
              - `results.txt` ..... metriky TPR, precíznosť a FPR
            - `0.8` ..... výsledky pre prahovú hodnotu 0.8
              - `confusion_matrix.pdf` ..... konfúzna matica
              - `results.txt` ..... metriky TPR, precíznosť a FPR
            - `histogram.pdf` ... vizualizácia distribúcie anomálneho skóre
            - `results.txt` ..... AUC skóre a doba behu algoritmu
            - `roc_curve.pdf` ..... ROC krivka