þÿDistribuovaný systém klasifika
þÿpro VoIP infrastrukturu vyu~ív
protokol SIP

2017-02-13T19:01:51Z

VYSOKÁ ŠKOLA BÁŇSKÁ–TECHNICKÁ UNIVERZITA OSTRAVA
VŠB–TECHNICAL UNIVERSITY OF OSTRAVA

FAKULTA ELEKTROTECHNIKY A INFORMATIKY
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

KATEDRA TELEKOMUNIKAČNÍ TECHNIKY
DEPARTMENT OF TELECOMMUNICATIONS

Distribuovaný systém klasifikace útoků pro VoIP infrastrukturu využívající protokol SIP

Distributed System for Attack Classification in VoIP Infrastructure Based on SIP Protocol

DIZERTAČNÍ PRÁCE
DISSERTATION THESIS

AUTOR PRÁCE — Ing. Jakub Šafařík
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Miroslav Vozňák PhD.
SUPERVISOR

OSTRAVA, 2016

PODĚKOVÁNÍ

Rád bych tímto poděkoval vedoucímu dizertační práce, panu docentovi Ing. Miroslavu Vozňákovi, Ph.D., za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Děkuji i celému týmu LIPTEL za podporu a pomoc související i nesouvisející s touto prací. Velké poděkování patří mé přítelkyni a rodině za pečlivé přečtení disertační práce, trpělivost a emoční i materiální podporu při studiu.

Ostrava ....

(podpis autora)

PROHLÁŠENÍ

Prohlašuji, že jsem svou dizertační práci vypracoval samostatně pod vedením vedoucího dizertační práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.
Jako autor uvedené dizertační práce dále prohlašuji, že v souvislosti s vytvořením této dizertační práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Ostrava ....
(podpis autora)

ABSTRAKT

Dizertační práce se zaměřuje na strojové metody klasifikace SIP útoků. Data o VoIP útocích jsou získána distribuovanou sítí detekčních sond s honeypot aplikacemi. Zachycené útoky následně zpracovává centralizovaný expertní systém Beekeeper. Tento systém provádí transformaci dat a jejich klasifikaci algoritmy strojového učení. V práci rozebírám různé typy těchto algoritmů, využívající učení bez i s učitelem, kdy nejlepších výsledků klasifikace dosahuje MLP neuronová síť. Tato neuronová síť je blíže popsána a testována v různých konfiguracích a nastaveních. Výsledná implementace obsahuje i techniky k vylepšení přesnosti, které stávající implementace nevyužívají. V práci seznamuji čtenáře se SIP protokolem, VoIP útoky a současným stavem na poli detekce těchto útoků. Navrhované řešení spoléhá na nasazení expertního systému Beekeeper s distribuovanou sítí detekčních sond. Koncept systému Beekeeper má modulární design s moduly pro agregaci a čištění dat, analýzu a vyhodnocení útoku, monitoring stavu jednotlivých sond, webové rozhraní pro komunikaci s uživateli atd. Různorodost a široká škála dostupných sond umožňuje jejich snadné nasazení v cílové síti, přičemž vyhodnocení nežádoucího provozu provádí autonomně systém Beekeeper. Díky modulární architektuře však není nutné omezovat funkci tohoto systému jen na detekci útoků. Věrohodnost a přesnost klasifikace útoků neuronovou sítí byla ověřena srovnáním s ostatními algoritmy strojového učení a výhody modelu byly popsány.

KLÍČOVÁ SLOVA

Analýza provozu, bezpečnost, honeypot, klasifikace útoků, neuronová síť, SIP, strojové učení

ABSTRACT

The dissertation thesis focuses on machine learning methods for SIP attack classification. VoIP attacks are gathered with various types of detection nodes through a set of a honeypot applications. The data uncovered by different nodes collects centralized expert system Beekeeper. The system transforms attacks to the database and classifies them with machine learning algorithms. The thesis covers various supervised and unsupervised algorithms, but the best results and highest classification accuracy achieves MLP neural network. The neural network model is closely described and tested under varying condition and settings. The final neural network implementation contains the latest improvements for enhancing the MLP accuracy. The thesis familiarizes the reader with SIP protocol, VoIP attacks and the current state of the art methods for attack detection and mitigation. I propose the concept of a centralized expert system with distributed detection nodes. This concept also provides techniques for attack aggregation, data cleaning, node state monitoring, an analysis module, web interface and so on. The expert system Beekeeper is a modular system for attack classification and evaluation. Various detection nodes enable easy deployment in target network by the administrator, while the Beekeeper interprets the malicious traffic on the node. But the general nature and modularity of the expert system Beekeeper allow it to be used in other cases as well. The reliability and accuracy of the neural network model are verified and compared with other machine learning available nowadays. The benefits of proposed model are highlighted.

KEYWORDS

Attack classification, honeypot, machine learning, neural network, security, SIP, traffic analysis

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM SYMBOLŮ A ZKRATEK

Seznam zkratek

ANNv2	Konkrétní implementace neuronové sítě v jazyce JAVA, označována také jako druhá generace MLP
API	Application Programming Interface
ARM	Advanced RISC Machine – označení architektury procesorů
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
AUC	Area under curve
B2BUA	Back-To-Back User Agent
C4.5	Algoritmus tvorby rozhodovacích stromů
C&C	Command and control – označení pro řízení botnetu
CEC	Cross–entropy cost – chybová funkce neuronové sítě
CLI	Command Line Interface
CPU	Central Processing Unit
CSV	Comma Separated Values
DDoS	Distributed denial of service
DET	Decision error tradeoff
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DoS	Denial of service
ED	Euclidean distance
EM	Expectation maximisation
FTP	File Transfer Protocol
GD	Gradient descent
GMM	Gaussian mixture model
GPL	General Public License
H.323	doporučení definující protokoly pro audiovizuální komunikaci

HD	Hellinger distance
HEP	Homer Encapsulation Protocol
HTTP	Hypertext Transfer Protocol - protokol pro přenos HTML dokumentů
ICMP	Internet Control Message Protocol
ID3	Iterative Dichotomiser 3 – algoritmus tvorby rozhodovacích stromů
IDS	Intrusion detection system
IETF	The Internet Engineering Task Force - organizace vyvíjející internetové standardy
IM	Instant Messaging
IPS	Intrusion prevention system
IREP	Incremental Reduced Error Pruning
IRSF	International Revenue Sharing Fraud
IVR	Interactive Voice Response
JDBC	Java Database Connectivity
JSON	JavaScript Object Notation
LTE	Long Term Evolution - vysokorychlostní přenos dat v mobilních sítích
LuCI	OpenWrt Configuration Interface
LVQ	Leraning Vector Quantisation
MBS	Mini batch size - velikost fragmentu učící sady
MD	Manhattan distance
MITM	Man in the middle
MLP	Multilayer percetron
MLPv1	Konkrétní implementace neuronové sítě MLP v jazyce JAVA, označována také jako první generace MLP
MVC	Model View Controller – Návrhový vzor
MVC	Model View Controller
PBX	Private branch exchange – pobočková ústředna
PSTN	Public Switched Telephone Network - veřejná telefonní síť
QC	Quadrature cost – chybová funkce neuronové sítě

RAM	Random Access Memory
RAMFS	typ souborového systému, kdy jsou data ukládány do RAM
REST	Representational State Transfer
ROC	Receiver Operating Characteristic
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
SABU	Sdílení a Analýza Bezpečnostních Událostí
SDHC	Secure Digital High Capacity
SDP	Session Description Protocol
SGD	Stochastic gradient descent
SIP	Session initiation protocol
SIPS	SIP Secure
SMTP	Simple Mail Transfer Protocol - protokol pro přenos zpráv elektronické pošty
SNMP	Simple Network Management Protocol
SNMP	Simple network managemnt protocol
SOM	Self-organizing map
SPADE	Statistical packet anomaly detection engine
SPIT	SPAM over IP telephony
SPIT	Spam over IP Telephony
SRTP	Secure Real-time Transport Protocol
SSH	Secure Shell
SSL	Secure socket layer
TCP	Transmission Control Protocol
TDoS	Telephony Denial of Service – označení pro DoS pouze v rámci telefonie
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
TLS	Transport layer security
UA	User Agent

UAC	User Agent Client
UAS	User Agent Server
UC	Unified Communications – platforma sjednocené komunikace
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VFDT	Very Fast Decision Trees
VoIP	Voice over Internet Protocol
VoLTE	Voice over LTE
VUC	Volume under curve

Použité symboly

$(w^{l+1})^T[-]$	Transponovaná matice všech vah do následující vrstvy
$\bar{x}[-]$	Aritmetický průměr
$\beta[-]$	Parametr důležitosti *Recall*, $\beta \in < 1, +\infty)$
$\Delta w_{ij}[-]$	Změna váhy mezi neurony i a j
$\Delta w'_{ij}[-]$	Změna váhy mezi neurony i a j z předchozího kroku
$\delta_i^l[-]$	Chyba neuronu i ve vrstvě l
$\delta_j^l[-]$	Chyba neuronu j ve vrstvě l
$\delta^{l+1}[-]$	Vektor chyb neuronů následující vrstvy
$\eta[-]$	Učící koeficient, learning rate
$\lambda[-]$	Parametr vlivu L2 regularizace
$\mu[-]$	Momentum
$\nabla_y E[-]$	Vektor gradientů
$\sigma[-]$	Směrodatná odchylka
$\sigma'(z^L)[-]$	Vektor derivací aktivací neuronů vrstvy L
$\theta[-]$	Prahová hodnota neuronu (bias)
$ACC[-]$	Accuracy – přesnost klasifikace
$c[-]$	Strmost sigmoidu

C_{pred}	Predikovaná třída útoku
C_{real}	Reálná třída útoku
$E_{CEC}[-]$	Chybová funkce, cross–entropy
$E_{rate}[-]$	Error rate – chybovost klasifikace
$f_i^N[-]$	Normalizovaná hodnota vlastnosti f_i pro vstupní vektor
$F_m[-]$	F–measure
$FN[-]$	False negative, chyba 2. řádu
$FP[-]$	False positive, chyba 1. řádu
$G_m[-]$	G–mean
$H^2[-]$	Hellinger distance
$max(F_i)[-]$	Maximum všech vlastností indexu i pro vstupní vektory
$min(F_i)[-]$	Minimum všech vlastností indexu i pro vstupní vektory
$N_C[-]$	Počet útoků ostatních tříd ve vstupních datech vzhledem k třídě C
$NPE[-]$	Nejnižší počet epoch nutných k naučení modelu
$o_j[-]$	Očekávaný výstup neuronu
$p[-]$	Počet vzorů trénovací množiny
$P_C[-]$	Počet útoků třídy C ve vstupních datech
P_{HD}	Rozložení dat během trénovacího období
$PPE[-]$	Průměrný počet epoch nutných k naučení modelu
$PRE[-]$	Precision – preciznost klasifikace
Q_{HD}	Rozložení dat během testovacího období
$REC[-]$	Recall – senzitivita
$SPE[-]$	Specificity – specifičnost
$TN[-]$	True negative
$TP[-]$	True positive
$w_i[-]$	Váha spojení neuronové sítě
$x_i[-]$	Vstup neuronu
$x_{max}[-]$	Maximum $\max_x : x \in X, X = x_1, ..., x_n$

$x_{min}[-]$	Minimum $\min_x : x \in X, X = x_1, ..., x_n$
$y[-]$	Výstup neuronu
$y^L[-]$	Vektor výstupů neuronů vrstvy L
$y_i^{l-1}[-]$	Výstup neuronu j ve vrstvě $l-1$
$z[-]$	Vnitřní potenciál (aktivace) neuronu

1 ÚVOD

Služba přenosu hovoru prostřednictvím IP sítí – VoIP je v dnešní době často nasazovaná technologie unifikace hlasových hovorů, video hovorů, telekonferencí i přenosů mediálního obsahu obecně. Důvody širokého rozšíření jsou nízké provozní náklady, využití stávající datové infrastruktury a pokročilé funkce a vlastnosti nedostupné pro předchozí generaci PSTN. Jedním z nejčastěji používaných protokolů pro IP telefonii se stal SIP, zejména díky otevřenosti tohoto protokolu. S přicházejícím rozmachem vysokorychlostních bezdrátových datových přenosů ještě více vzroste zastoupení SIP protokolu na trhu, a to prostřednictvím chytrých telefonů, tabletů či jiných přenosných zařízení. Společnost Juniper ve své zprávě uvádí, že během roku 2017 dosáhne počet uživatelů využívajících SIP protokol miliardy uživatelů pouze na mobilních zařízeních. [9]

Právě kvůli zvětšujícímu se počtu společností nabízejících VoIP služby na bázi SIP protokolu, tak i rozšiřující se nabídce klientských aplikací či hardwarových zařízení, roste i zájem útočníků o tuto službu. Z hlediska útočníků se nejčastěji jedná o znemožnění přístupu k dané službě (DoS, DDoS), zneužití nebo neoprávněné užívaní služby, krádeže osobních údajů, odposlech účastníků, atd. Situaci zhoršuje i podobnost SIP protokolu s textovými protokoly HTTP a SMTP. Většina zranitelností těchto protokolů je dále aplikovatelná i na VoIP infrastrukturu spoléhající na SIP protokol. Během prvotního definování SIP protokolu a principu jeho funkcí nebyl brán velký zřetel na bezpečnost tohoto protokolu, což se snažila postupně vyřešit řada rozšíření. Nehledě na tato rozšíření zůstává architektura SIP velmi zranitelná, jak prokazují i předcházející výzkumy v laboratoři skupiny LIPTEL (mimo jiné například [Saf04, Saf07, Saf10]) či dalších výzkumných týmů [32, 43].

Problematikou bezpečnosti SIP protokolu se zabývá řada odborných publikací a článků. Existují doporučené postupy a bezpečnostní politiky snižující rizika napadení i ovlivnění VoIP infrastruktury. Nicméně právě díky otevřenosti protokolu SIP, nejednotnosti jednotlivých implementací serverů i širokému portfoliu dostupných klientů (ať už softwarových či hardwarových) podporujících různé vlastnosti a rozšíření SIPu, lze tyto navrhované metodiky v rámci infrastruktury nasadit v omezené míře.

Tato disertační práce přináší řešení vzniklé situace díky nasazení distribuovaného systému pro sběr informací o útocích na VoIP infrastrukturu, stejně tak jako automatizovaný systém pro klasifikaci útoků a vyhodnocení těchto útoků. Centralizovaný server umožňuje jednoduchou interpretaci detekovaných útoků, poslouží administrátorům sítí pro vyhodnocení bezpečnostních hrozeb a zvýšení současného zabezpečení. Díky umístění sond v různých sítích (oddělených logicky i lokalitou) je možné nasadit opatření proti útokům i v infrastruktuře, v níž se útok ještě nevyskytl.

Navrhovaný systém se skládá z několika částí. Nejdůležitější částí je server, sloužící pro agregování informací z jednotlivých sond. Tento server provádí také analýzu detekovaných útoků a jejich klasifikaci. Pro klasifikování útoků využívám řadu algoritmů strojového

učení, jakými jsou neuronové sítě (MLP), samoorganizační Kohonenovy mapy (SOM), rozhodovací stromy (DT), aj. Různé metody klasifikace umožní zlepšení přesnosti a zvýšení věrohodnosti stanovených klasifikací.

Další částí navrhovaného systému jsou variabilní sondy, prostřednictvím nichž probíhá sběr informací o útocích. Primárním zdrojem jsou honeypot aplikace, emulující chování produkčních programů, které obsahují bezpečnostní slabiny. Analýzu chování útočníků a metod jejich útoků bude provádět již zmíněná serverová část. Bezpečnostní sondy jsou dostupné v různých variantách: od konkrétní aplikace, předpřipravené obrazy či předkonfigurovaná hardwarová zařízení. Softwarové vybavení sond využívá existujících open-source aplikací a operačních systémů.

Poslední součástí navrhovaného bezpečnostního řešení je zavedení zabezpečeného přenosu informací o útocích mezi sondami a serverem. V neposlední řadě probíhá také monitoring jednotlivých sond. Výsledkem je tedy distribuovaný systém sbírající a vyhodnocující informace o útocích na SIP infrastrukturu s možností exportu výsledků do různých formátů i systémů.

2 RELAČNÍ PROTOKOL SIP

Následující kapitola slouží jako nezbytný úvod do problematiky VoIP infrastruktury založené na protokolu SIP. Důkladný popis vlastností SIP je nad rámec dizertační práce, avšak základní popis problematiky je nutný pro pochopení zranitelností, zabezpečení i možností detekce útoků na SIP.

Konkrétně jsou popsány základní vlastnosti a rysy SIP protokolu, následované přehledem různých typů útoků využívaných proti VoIP infrastruktuře.

2.1 Vlastnosti SIP

Protokol SIP byl definován v rámci dokumentu RFC 3261 roku 2002 [50] skupinou IETF. Od této doby zůstává původní jádro protokolu beze změn, průběžně jej však doplňovali další autoři v rámci rozšiřujících RFC dokumentů.

SIP protokol byl navržen jako jednodušší varianta protokolu H.323. Zároveň byl zveřejněn jako open-source software, který mohl do své VoIP infrastruktury implementovat kdokoliv. Cílem bylo jak zvýšení interoperability mezi jednotlivými výrobci a implementacemi VoIP služeb, tak i nezávislost na proprietárních protokolech jednotlivých dodavatelů.

Postupem času se SIP protokol stával vedoucím protokolem na poli VoIP komunikací a dnes se jedná o jeden ze základních kamenů většiny VoIP systémů. Účelem SIP protokolu je přenos signalizace VoIP pro vytvoření, ukončení a řízení jednotlivých multimediálních relací mezi účastníky (nalezení a stav účastníka, sestavení, řízení a modifikace spojení, ...). SIP zprávy standardně využívají protokolu UDP (server naslouchá na portu 5060), ale mohou využívat i TCP či šifrované TLS spojení (označován jako SIPS podobně jako je tomu u HTTP/HTTPS, server naslouchá na portu 5061).

V rámci VoIP služeb je SIP využíván ve spolupráci s dalšími protokoly. RTP zajišťuje paketové doručování multimediálních dat (tedy audia i videa). Byl definován v RFC 1889 a později nahrazen RFC 3350. Šifrovaná varianta SRTP je definována v RFC 3711. Protokol RTCP slouží k řízení RTP relace na základě sledování kvality tohoto spojení. Posledním zmíněným protokolem je SDP, který je určen k vyjednání parametrů pro přenos médií mezi účastníky, používaného transportního protokolu, typu kodeku, atd.

Hlavními znaky SIP protokolu je především jeho textová forma a komunikace na bázi požadavků a odpovědí. Nezanedbatelná je také podobnost s protokoly HTTP a SMTP, z nichž protokol vychází [13, 50].

2.1.1 Adresace - SIP URI

Pro účely adresace v rámci SIP se využívá URI, jako je tomu i u adresáta u SMTP. Dle RFC 3261 je SIP URI definována následovně:

```
sip:user:password@host:port;uri-parameters?headers
```

Účel jednotlivých částí je patrný z jejich názvu. U vynechaných částí se použijí výchozí hodnoty, pokud je to možné (např. pro *port* je výchozí hodnota 5060). Část *password* se doporučuje nespecifikovat. Výsledné URI mají většinou jednoduchý tvar a mohou vypadat například takto (příklady byly převzaty z RFC 3261 [50]):

```
sip:bob@biloxi.com
sips:bob@biloxi.com
sip:2125551212@example.com
sip:+1-212-555-1212:1234@gateway.com;user=phone
sip:alice@192.0.2.4:5060
```

2.1.2 Prvky SIP architektury

Dle specifikace se architektura SIP skládá z 5 logických prvků. U serverových komponent však bývá obvyklé sloučení více funkcionalit do jednoho serveru.

- User Agent (UA) – Jakýkoliv klient (aplikace či zařízení), který vytváří SIP spojení, např.: IP telefon, softphone aplikace, IM klient, mobilní zařízení či aplikace nebo gateway pro přístup do PSTN. Označován také jako koncový bod.
- Proxy server – Obsluhuje a směruje SIP požadavky jednotlivých agentů (UA). Může vystupovat jako klient i server zároveň. Zároveň plní i funkci kontroly práv (např. zda může daný uživatel provést volání), interpretuje SIP zprávy a může přepisovat specifické části SIP zpráv.
- Redirect server – Umožňuje přesměrování příchozích požadavků od klientů na alternativní URI. Snižuje zátěž na proxy serveru.
- Registrar server – Slouží pro zpracování REGISTER požadavků. Mapuje klienty (UA), resp. jejich SIP URI k jejich aktuálnímu umístění v síti (IP adresa, username, port, ...)
- Location service – Lokalizační službu využívají předchozí typy serverů. Udržuje informace o umístění klienta a SIP proxy serverů. Typicky bývá přímo součástí Registrar serveru.

Základními prvky SIP sítě jsou tedy User agents a jednotlivé SIP servery. Koncové body, podobně jako je tomu i u SIP proxy, se chovají jako klient (UAC) i server (UAS), v závislosti na tom, zda požadavky vysílají nebo přijímají (odpovědi vice versa). Každé koncové zařízení musí být spárováno s konkrétním SIP účtem (označován dále i jako SIP user nebo SIP Extension).

- UAC – Označení klientské části, vysílá požadavky a přijímá odpovědi.
- UAS – Označení serverové části, přijímá požadavky a odesílá odpovědi.
- B2BUA – Slouží k přemostění komunikace (např. provádí transkódování). Udržuje si informace o dialogu podobně jako SIP proxy.

Rozdíl mezi B2BUA a SIP proxy spočívá v obsluze spojení. B2BUA se chová jako prostředník, příchozí spojení ukončuje a sestavuje další pro spojení s cílovým klientem. SIP proxy tyto zprávy přeposílá. Z pohledu UA však není patrné, zda komunikuje s B2BUA či SIP proxy.

Proxy servery dělíme na dvě skupiny dle množství uchovávaných informací o spojení. První skupinou jsou bezestavové (Stateless) servery, které neuchovávají stavové informace o probíhající komunikaci. Používají se pro vyvažování zátěže v síti a jejich hlavní výhodou je rychlost zpracování SIP zpráv.

Stavový server (Stateful) udržuje informace o průběhu komunikace až do jejího ukončení. Výhodou je monitoring stavu spojení, detekce replikace zpráv, možnost větvení či přesměrování provozu, atd. Protože si server uchovává informace o každém navázaném spojení, je z bezpečnostního hlediska náchylnější k útokům cílených na vyčerpání zdrojů cíle [32].

2.1.3 Struktura SIP zpráv, požadavky a odpovědi

Jak již bylo zmíněno v kapitole 2.1, SIP komunikace probíhá na bázi jednotlivých požadavků a odpovědí (request/response). Každá SIP zpráva se skládá z hlavičky a těla. V hlavičce SIP požadavku a odpovědi je vždy vložena informace o použité verzi SIP protokolu, konkrétně řetězec `SIP/2.0`. V případě požadavku (request) je uvedena ještě metoda a SIP URI. Odpověď (response) obsahuje kód a textový popis odpovědi.

Obr. 2.1: První řádky ze SIP zprávy pro požadavek a odpověď.

Syntaxe:

```
Method Sip-URI Sip-Version
Sip-Version Status-Code Reason-Phrase
```

Například:

```
INVITE sip:1234@asterisk.vsb.cz SIP/2.0
SIP/2.0 100 Trying
```

Dále obsahuje SIP hlavička řadu polí pro nastavení specifických hodnot ve tvaru klíč–hodnota. Výčet jednotlivých polí je uveden v příloze A.1 a zkráceně v následující kapitole věnující se bezpečnosti SIP protokolu. Hlavním rozlišovacím nástrojem mezi jednotlivými SIP požadavky je označení SIP metody. Tabulka 2.1 uvádí jednotlivé typy metod používaných uvnitř SIP požadavků definovaných v základních i rozšiřujících RFC doporučeních.

Každá z odpovědí na SIP požadavek obsahuje stavový kód a odůvodnění, určené pro lepší čitelnost a vysvětlení stavového kódu. První číslice stavového kódu značí třídu odpovědi, následující dvojčíslí pak konkrétní stav (podobně jako je tomu i u HTTP protokolu).

Tab. 2.1: Přehled SIP metod[13].

SIP request	Účel	Definován
REGISTER	Registrace SIP uživatele v daném umístění.	RFC 3261
INVITE	Zahájení spojení mezi koncovými body.	RFC 3261
ACK	Potvrzení (Acknowledgement) a odpověď na INVITE.	RFC 3261
BYE	Ukončení sestaveného spojení mezi účastníky.	RFC 3261
CANCEL	Ruší INVITE požadavek, nemá vliv na sestavené spojení.	RFC 3261
OPTIONS	Určuje SIP zprávy a kodeky podporované účastníky spojení.	RFC 3261
PRACK	Provizorní ACK pro potvrzení provizorních odpovědí.	RFC 3262
REFER	Předání hovoru a kontaktů externím zdrojům.	RFC 3515
MESSAGE	Pro přenos instant messages.	RFC 3248
NOTIFY	Informace o změně stavu, které nejsou vázány k určité relaci.	RFC 3265
SUBSCRIBE	Přihlášení k odběru NOTIFY požadavků.	RFC 3265
PUBLISH	Publikování události (event) na server.	RFC 3903
UPDATE	Aktualizace parametrů session klientem.	RFC 3311
INFO	Přenos dalších informací v session.	RFC 6086

Tab. 2.2: Třídy stavových kódů SIP odpovědí[50].

Stavový kód	Třída	Význam
1xx	Provizorní odpověď	Požadavek byl přijat a je dále zpracováván.
2xx	Úspěch	Požadavek byl přijat, zpracován a akceptován.
3xx	Přesměrování	Je třeba další akce pro dokončení požadavku.
4xx	Chyba klienta	Špatná syntax požadavku nebo požadavek nelze obsloužit.
5xx	Chyba serveru	Provedení validního požadavku selhalo.
6xx	Globální chyba	Požadavek nelze zpracovat na žádném serveru.

2.1.4 Rozšíření SIP protokolu na trhu

VoIP služby částečně předběhly svou dobu a byly nasazovány především ve firemních prostředích. Ačkoli se tato technologie zdála být v posledních letech v útlumu, používáme ji v každodenním životě častěji, než bychom čekali. Rozvoj kvalitního vysokorychlostního datového připojení LTE umožnil nástup služeb jako VoLTE.

Během let vznikla řada řešení pro potřeby IP telefonie. Jsou jimi různá hardwarová zařízení, software klienti pro různé operační systémy i platformy (mobilní telefony, chytré telefony, weboví klienti). Platforma chytrých telefonů, kvalitní pokrytí datovým připojením, nárůst počtu webových aplikací a hlavně jednoduchost použití značně rozšiřuje počet nasazení SIP ústředen.

Díky otevřenosti SIP protokolu a dostupnosti SIP proxy jako open–source softwaru, roste počet služeb a aplikací s integrací multimediální komunikace. Především řada oblíbených aplikací pro IM přidala v posledních letech podporu videohovorů. Kromě použití

proprietárních protokolů je této funkcionality často dosaženo prostřednictvím VoIP na bázi SIP protokolu.

Na vzestupu jsou také platformy pro sjednocenou komunikaci (UC a kolaboraci – od textové komunikace, hovorů, videohovorů, konferenčních hovorů, webových klientů, propojení s PSTN až po sdílení aplikací, prezentací či pracovní plochy. Zástupcem UC platformy postavené na SIP protokolu je například Microsoft Lync.

Do budoucnosti předpokládám integraci do dalších nástrojů a služeb. Použitá technologie bude pro uživatele naprosto transparentní a bez nutnosti využití speciálního hardware či software, což zvyšuje nároky na zabezpečení a monitoring multimediálních služeb na straně poskytovatelů.

2.2 Bezpečnost SIP

Bezpečnost je zajisté jednou z nejdůležitějších oblastí nasazení jakékoliv služby, VoIP nevyjímaje. Většina dostupných serverových aplikací podporuje důkladné zabezpečení a většina sítí je dnes do jisté míry chráněna.

Stále se však můžeme setkat s laxním přístupem některých poskytovatelů, kteří nemají své služby dostatečně zabezpečeny nebo jsou zabezpečeny špatně. Jedním z důvodů může být i komplexnost zabezpečení spravované sítě a specializované znalosti bezpečnostních správců ohledně provozovaných služeb.

Dalším důvodem nedostatečného zabezpečení bývá zpravidla i nízký rozpočet nebo použité technologie. U technologií se nejedná pouze o prvky aktivní bezpečnosti (jako např. firewall), ale také o zařízení provozovaná v síti. Pokud hardwarové telefony nepodporují šifrování, nelze ho v síti nasadit bez aktualizace či výměny těchto zařízení, což nemusí být vždy možné.

Bezpečnost bývá často přehlížena a podhodnocena, protože její výsledky nejsou zjevně patrné a investice do zabezpečení nepřináší žádné technologické výhody. Pravá cena zabezpečeného systému se projeví až po úspěšném útoku, kdy dojde k úniku důvěrných dat, finanční újmě či poškození dobrého jména [42].

Hackingu se dnes nevěnuje jen pár jedinců, ale často se jedná o velmi dobře organizované skupiny pracující s jasným cílem za účelem zisku. Takovéto skupiny disponují nejen zkušenými a vzdělanými členy, ale také dostatečnými prostředky, hardwarovým vybavením i výpočetním výkonem. Pro tyto organizované skupiny následně není problém prolomit ochranu ani dobře zabezpečených systémů [36].

Roste i počet zařízení provozovaných v rámci zabezpečených sítí, které mohou být potenciálně napadené malwarem (např. chytré telefony zaměstnanců, tablety, jejich notebooky, atd.). Problémem může být i použití cloudových služeb či využívání neautorizovaného software pro komunikaci.

Nově se také objevují skupiny politicky motivovaných útočníků, označovaných jako hacktivisté. Současně se zvyšuje úroveň a počet sofistikovaných nástrojů pro hacking a

zjednodušuje se nasazení malware k obětem. U VoIP hrozí také větší nebezpečí zneužití některého z postranních kanálů.

U VoIP systémů jsou obvyklé pokusy o volání na placená čísla nebo do zahraničí. Potvrzují to zprávy ze zahraničí i prostředí České republiky. Hodnota způsobených škod přitom každoročně roste a překonává svým objemem i škody způsobené zneužitím platebních karet [13]. Zpráva ČTÚ uvádí celkovou škodu 21 milionů Kč za rok 2010 v rámci 4 případů zneužití pobočkových ústředen [26].

Tab. 2.3: Přehled cílů a bezpečnostních hrozeb [13].

Cíl útoku	Hrozby
Aplikace	Volání na placené linky, vishing, ...
Protokol	SPIT, fuzzing, flood attacks, ...
Operační systém	Buffer overflow, worms, DoS, ...
Server	Kompromitování, DoS, ...
Podpůrné systémy	SQL injection, DHCP exhaustion, ...
Síť	UDP flood, ICMP unreacheable, ...
Politiky a postupy	Slabá hesla, špatná práva, ...
Fyzické zabezpečení	Nezabezpečený přístup k serveru, ...

Dalším typem je cílené zneužití napadeného systému a jeho úprava pro získání přístupu k neoprávněným operacím a následnému využívání napadaného operátora k hovorům (na účet napadeného). Problémy způsobuje i odposlech hovorů, průmyslová špionáž či poškození dobrého jména. Tyto případy nebývají mediálně známé a postižení informace o útoku veřejně nesdělují.

Slabinou může být i nasazení open-source ústředny. Přestože bývá otevřený software častěji záplatován, mohou ho, na rozdíl od proprietárního software, útočníci snadno zkoumat, testovat a hledat slabiny ve zdrojovém kódu. Stejně tak je dostupná i výchozí konfigurace, administrátorské účty, hesla. Ponechání těchto výchozích nastavení a používání starých verzí (kde jsou již známy případné slabiny) je významnou bezpečnostní hrozbou.

Útoky vůči SIP infrastruktuře lze rozdělit podle několika kategorií. Rozdělení podle cíle útoku uvádí tabulka 2.3 a toto rozdělení budu respektovat i ve výpisu jednotlivých typů útoků. Dále lze útoky dělit podle cíle útoku, složitosti, dopadu na cíl nebo míry rizika.

2.2.1 Typy hrozeb pro SIP infrastrukturu

Existuje mnoho typů bezpečnostních hrozeb, následující kapitola se proto zabývá útoky ohrožujícími přímo VoIP infrastrukturu. Z hlediska bezpečnosti je pro SIP důležitých několik faktorů. Jedním z nejpodstatnějších je textová podoba SIP protokolu a jeho podobnost s protokoly HTTP a SMTP. Díky této podobnosti lze aplikovat útočné postupy

používané proti těmto protokolům také na SIP. Následující souhrn útoků je rozdělen podle charakteru útoku.

Průzkumné útoky

Útoky z této kategorie bývají častým předvojem následujících kategorií. Cílem bývá jak celá VoIP infrastruktura, tak i konkrétní provozovaná zařízení a pobočková ústředna.

Zdrojem základních informací o provozovaném software mohou být veřejně dostupné informace z webových stránek, DNS nebo WHOIS databáze. Ačkoli nemusí být informace o interně provozované VoIP ústředně či jejím zabezpečení přímo uvedena, útočníci mohou využít i netradiční kanály jako nabízené pracovní pozice se specifikovanými nároky na znalosti konkrétních bezpečnostních systémů či VoIP služeb. Vodítkem mohou být i profily správců a zaměstnanců na specializovaných stránkách profesních sociálních sítí [13]. Metodami k získání informací jsou:

- WHOIS analýza
- DNS analýza
- Google hacking

Zdrojem informací o provozovaných zařízeních, serverech, službách a software jsou různé typy skenování. Uveďme například:

- ICMP ping sken
- ARP sken
- SNMP sken
- skenování portů
  - TCP sken – např. TCP SYN
  - UDP sken
- identifikace hosta – zjištění provozovaného OS, verze, spuštěných služeb
- nalezení telefonů – identifikace spuštěných zařízení
- Wardialing – technika vyhledání aktivních čísel nebo účtů

Skenování nemusí probíhat pouze na úrovni koncové sítě. V roce 2011 byl zaznamenán průběh skenování celého rozsahu IPv4 adres pomocí botnetu Sality zaměřeného na SIP servery (využívající SIP Register zpráv). Nejsou známy konkrétní cíle tohoto skenování ani počet nalezených serverů či klientů [15].

Výsledný seznam serverů mohl být prodán na černém trhu, zranitelné servery napadeny. Mnou nasazené honeypoty (více v kap. 6.1) zachycují první útoky už za méně než 20 minut po připojení do sítě [Saf11].

Jakmile útočníci dokončí základní oskenování sítě a nalezených zařízení, pokračují důkladnějším skenováním, útokem pro nalezení jednotlivých účtů, atd.

- Banner grabbing – technika identifikace služby pomocí zaslané odpovědi při připojení na otevřený port.

- Identifikace SIP proxy.
- Identifikace účtů – SIP Extension Enumeration.
  - Pomocí SIP REGISTER zpráv.
  - Pomocí SIP INVITE zpráv.
  - Pomocí SIP OPTIONS zpráv.
- Nalezení a identifikace služeb pro podporu VoIP infrastruktury – např.: TFTP serveru.
- Identifikace SIP proxy pomocí známých prvků konkrétních implementací – např.: obsazovací tón, obsluha hovoru, specifické nahrávky a chování IVR systému.

Nástroje pro zmíněné útoky jsou např. nmap, snmpwalk, snmp-probe, sipscan, sipcicious, sivus, atd. [13]. Proti části těchto útoků se nelze už z jejich podstaty bránit, předznamenávají ale případné další útoky a pro útočníky představují cenný zdroj dat o cílové infrastruktuře a zařízení v ní provozovaných.

Manipulace se SIP zprávami

Útoky manipulující s částmi SIP zprávy jsou spojeny s textovou podstatou SIP zpráv. Tabulka 2.4 obsahuje základní přehled polí v SIP hlavičce. Úplný výčet a konkrétní pravidla výskytu podle specifikace jsou uvedena v příloze A.1.

Tab. 2.4: Pole hlavičky podporované SIP protokolem.

Vyžadováno	**Pole v hlavičce SIP**
Povinně	Call-ID, CSeq, Max-Forwards, To, Via
Volitelně	Authorization, Contact, Content-Disposition, Content-Encoding, Content-Language, Content-Length, Content-Type, Date, MIME Version, Record-Route, Route, Timestamp, User-Agent
Zakázano	Accept, Accept-Encoding, Accept-Language, Alert-Info, Allow, Authentication-Info, Call-Info, Error-Info, Expires, In-Reply-To, Min-Expires, Organization, Priority, Proxy-Authenticate, Proxy-Authorization, Proxy-Require, Reply-To, Require, Retry-After, Server, Subject, Supported, Unsupported, Warning, WWW-Authenticate

Techniky užité pro tento druh útoků jsou například:

- Payload tempering – manipulace s obsahem SIP zprávy
- injektování nežádoucího obsahu – SQL injection, forged content injection, ...
- přetečení
  - Buffer–overflow
  - Overflow–space
  - Overflow–null

  - Integer–overflow
- vymazání části SIP hlavičky
- použití nepodporovaného kódování – standardně je očekáváno ASCII
- využití zakázaných polí SIP hlavičky – viz. tabulka 2.4
- změna polí SIP hlaviček
- prolomení autentizace
- restart koncového zařízení
- upgrade firmware na koncovém zařízení

Techniky lze využít pro podvodná volání, přerušení služby i útoku na podpůrnou infrastrukturu [20]. Podobný efekt jako tento typ útoku může způsobit i legitimní aplikace, pokud dojde k chybě v implementaci, špatné konfiguraci serveru či zařízení na straně klienta.

Nejčastěji bývají SIP servery napadány za účelem volání na placené linky nebo do zahraničí. Mezi dále využívané metody podvodných hovorů patří:

- Zneužití nemonitorovaných zařízení – útok na zařízení nebo účet, u nějž se nepočítalo s danou funkcionalitou (fax, konferenční telefon, atd.).
- Nalezení PIN kódu pro odchozí nebo placené hovory.
- Manuální přepojení na odchozí PSTN linku – přepojení volání legitimním uživatelem
- Dial-through call – využívá možnosti zavolat na IP PBX a zvolit požadované číslo. Ústředna následně spojí toto číslo (placená linka) s externím číslem volajícího.
- Automatický IRSF – varianta předchozího útoku, cílem jsou placené linky v zahraničí. V další variantě útočník generuje provoz z interní sítě poskytovatele (malware, traffic generator).
- Wangiri – z japonštiny "jednou (zazvonit) a dost". Útočník provádí velice krátké hovory z čísla placené linky na legitimní čísla. Uživatel vidí zmeškaný hovor a volá zpět.
- Call pumping – útočník generuje mnoho krátkých hovorů na bezplatné linky call center nebo naopak málo hovorů, které ale dlouze udržují spojení s IVR (pomocí posílání DTMF kódů, audio nahrávek). Příjmem útočníka je podíl na provozu bezplatných linek.
- Smartphone fraud – podvodná volání prostřednictvím malware v chytrých telefonech.

Uvedené typy patří mezi nejčastější zástupce skupiny zmanipulovaných hovorů. Objem škod v této oblasti překonává více než dvojnásobně i podvody s kreditními kartami a dosahuje za rok 2008 částky 4,96 miliard USD. Mezi nejčastější země, kam hovory míří, patří: Kuba, Somálsko, Sierra Leone, Zimbabwe a Lotyšsko [13].

Obdobná situace je i v České republice, kde bývají často postihnuti jak malí provozovatelé (hotely, penziony, malé firmy, atd.), tak i velké společnosti s nedostatečně zabezpečenou pobočkovou ústřednou.

Tyto podvody bývají mnohdy prováděny s dalším typem útoků označovaných termínem spoofing. Princip spočívá v úpravě hlavičky SIP zprávy (Message tempering) za účelem odcizení identity uživatele či přístupu k neoprávněným operacím (volání placených čísel). Metody využívané při spoofingu:

- registration hijacking – odcizení registrace legitimního uživatele
- session hijacking – odcizení probíhající relace (hovoru)
- maskování čísla volajícího
- využití anonymizačních služeb
- jednorázové hovory – Skype, Google voice, předplacené SIM karty, atd.

Tématicky lze do této skupiny zařadit i specifické útoky spojené se sledováním a monitoringem komunikace. Pro účinný monitoring je nutné získat přístup k vnitřní síti poskytovatele VoIP služeb, dané postupy jsou ale nad rámec této práce (WiFi sniffing, zásah do síťové infrastruktury, MITM metody, malware, ...).

Pokud je útočník schopen zasahovat do komunikace, může ji efektivně sledovat a případně i zasahovat do jejího průběhu. Získaná data mohou být využita samostatně i v rámci dalších útoků.

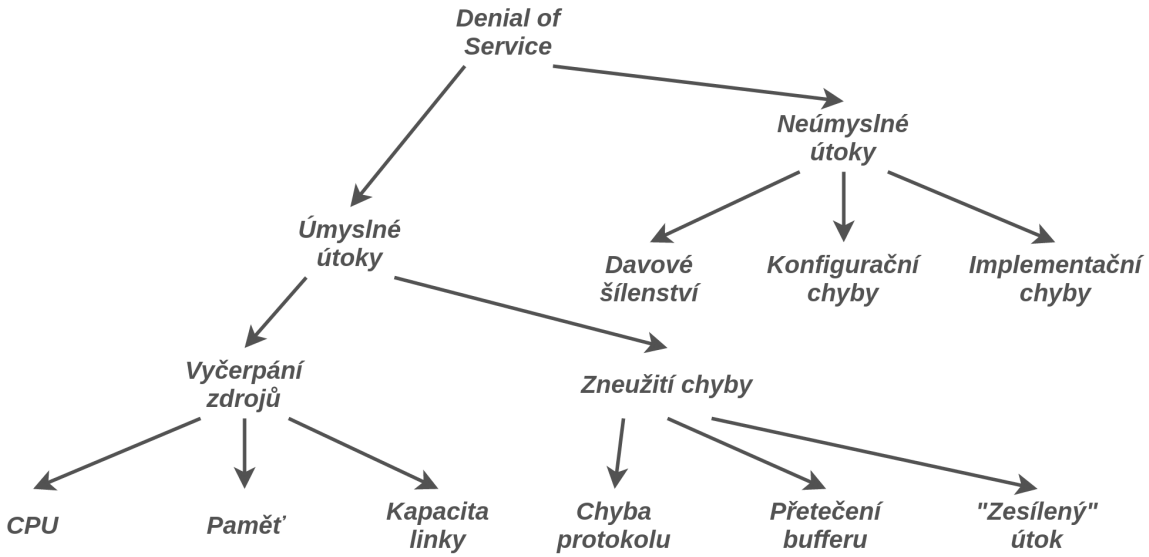
- TFTP Configuration sniffing – získání konfigurace z provisioning serveru
- monitoring aktivních SIP Extensions – pasivní identifikace účtů
- sledování vzorů v komunikaci
- odposlech hovorů a jejich analýza – lze např. odposlechnout DTMF kódy (např. PIN)

Omezení služby a blokování přístupu k službě

Útoky omezující nebo zamezující přístup k službě tvoří další velkou skupinu útočných metod. Jak už je z názvu jasné, cílem je zamezit skupině uživatelů v použití VoIP. Účinek útoku může být omezen na jednotlivé uživatele, ale i na proxy servery a tedy celou VoIP infrastrukturu. V rámci IP telefonie se lze setkat i s pojmem TDoS, který je ale významově stejný s DoS. Částečné nebo úplné zamezení přístupu ke službě může být také doprovodným faktorem některého z již dříve popsaných útoků (např.: Call pumping). Existují i případy použití DoS k maskování jiného útoku.

V praxi dochází k DoS i neúmyslně díky konfiguračním chybám či chybné implementaci SIP serveru. Podobný průběh jako DoS mají i případy "davového šílenství", během nichž je server přetížen nadměrným využíváním služby. K těmto situacím dochází během přírodních katastrof, neštěstí, reklamních akcí, před významnými svátky či jiného podmětu vyvolávajícího hromadnou potřebu komunikace. Zmíněný stav se většině případů rychle vrátí k normálu, jakmile se objem komunikace sníží [13, 32].

Obecně lze případy úmyslných DoS útoků rozdělit do dvou kategorií (viz. Obr. 2.2). Softwarové útoky míří na podstatu SIP protokolu a snaží se způsobit pád serverové aplikace, ukončit cizí hovor, přesměrovat hovor, atd. [42]. Tématicky lze software útoky zařadit



Obr. 2.2: Dělení DoS útoků.

i do kategorie manipulace se SIP zprávami zmíněné v předcházejícím bloku. Příkladem jsou:

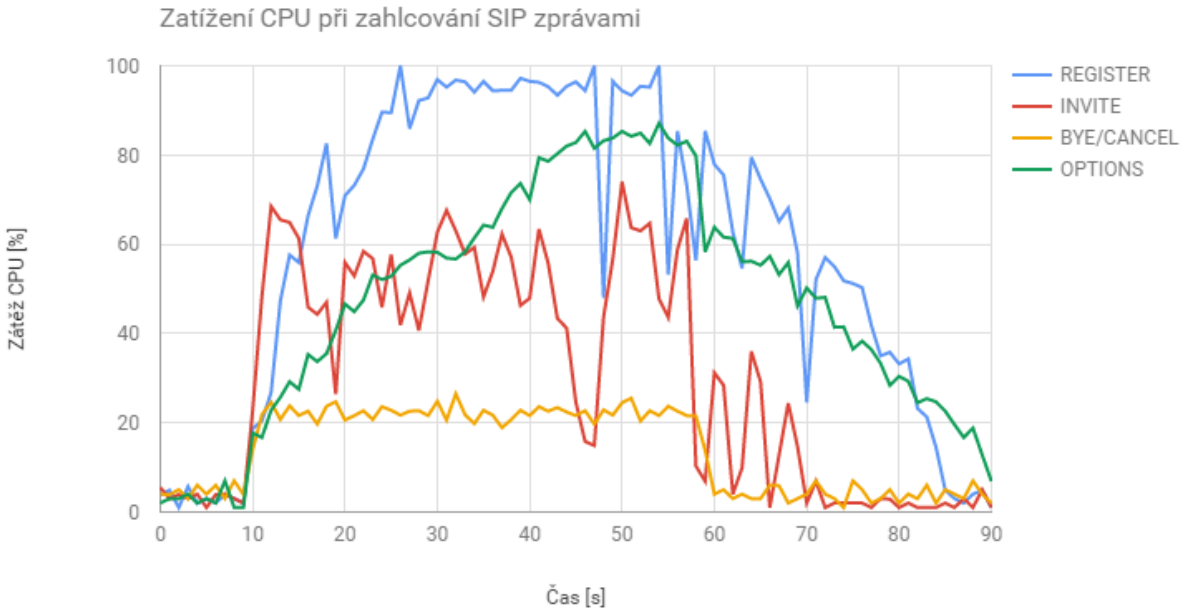
- Session teardown – přerušení relace legitimního uživatele (BYE a CANCEL útok)
- SIP proxy fuzzing – zaslání nekorektní SIP zprávy
- Overflow attacks

Hardwarové útoky se snaží o vyčerpání dostupných zdrojů serveru a řadí se mezi hlavní metody využívané k DoS. Je to způsobeno jednoduchým provedením a poměrně vysokou efektivitou. Obecně se zaměřují na jeden ze tří základních prostředků ke zpracování požadavků, mohou ale cílit i na více zdrojů zároveň.

- CPU
- Paměť
- Síťové připojení

Veškeré SIP zprávy jsou zpracovávány procesorem SIP proxy. Pokud ho zatížíme vysokým objemem SIP zpráv, lze poměrně efektivně vyčerpat dostupnou kapacitu pro jejich zpracování. Zatížení lze zvýšit i vyšší složitosti zpracování SIP zpráv. K vytížení procesoru pak dochází i pro nízké datové toky SIP zpráv.

SIP proxy se snaží obsloužit i částečně poškozené SIP zprávy, ovšem za cenu náročnějšího zpracování na procesoru. Výsledkem je opět účinnější efekt DoS útoku. Vliv jednotlivých SIP zpráv byl zkoumám LIPTEL týmem [Saf12] a dopad na vytížení procesoru je zobrazen na grafu 2.3. Útok probíhá od 10s do 60s, ve stejném množství 250 SIP zpráv za sekundu.



Obr. 2.3: Vliv použité SIP zprávy na zatížení CPU při DoS útoku [Saf12].

Zvýšenou zátěž procesoru způsobuje i šifrování SIP zpráv, obzvláště s neplatnými certifikáty. Server musí vynaložit dodatečné prostředky pro práci se zašifrovanou zprávou a ověřením certifikátů [32].

Při zpracování SIP zpráv dochází k ukládání dílčích informací na serveru. Množství uložených informací závisí na módu, ve kterém server pracuje (viz. kap. 2.1.2). Jednoduchým příkladem je TCP SYN flood nebo zasílání vysoce fragmentovaných paketů, z nichž útočník část záměrně vynechá.

Poslední typ útoků má za cíl zahltit síťovou linku, kterou je server připojen do sítě. Následně dochází k zahazování legitimních SIP požadavků, které nelze v záplavě škodlivých paketů rozlišit. Pro útok se využívá protokol UDP s maximální velikostí paketů. Existují samozřejmě i kombinace zmíněných útoků. Zahlcením serveru SIP zprávami INVITE přenášenými po UDP lze efektivně vytěžovat výpočetní prostředky i konektivitu serveru.

Uvedené DoS útoky značně závisí na konfigurace a dostupných kapacitách serveru. Některé z nich lze omezit naddimenzováním serverových prostředků nebo použitím dynamického alokování dalších prostředků. Ale ani tato protiopatření nemusí být efektivní vůči distribuovanému DoS útoku.

DDoS útoky bývají vedeny z více zdrojů, a jejich eliminace není jednoduchá. Zdrojem útoku je jednoduchá aplikace – bot, který se autonomně replikuje a je ovládán útočníkem k provedení specifických úloh. Jednotliví boti jsou sdružení do botnetu, ze kterých následně útočník zasílá miliony požadavků na cílovou službu (C&C botnet). Útok může přicházet i z více botnetů zároveň a trvat i několik dnů.

Amplifikační útoky dosahují podobného efektu jako DDoS tím, že útočník rozesílá legitimní požadavky s podvrženou adresou zdroje (cíl útoku) na reálné servery. Ty se snaží požadavek obsloužit, zreplikují jej a odešlou odpověď zpět. Vzápětí začne cíl zahlcovat řada odpovědí na útočníkem generované podvržené zprávy. Útočník nepotřebuje k tomuto druhu útoku infiltrovat žádná zařízení ani přístup k botnetu. Příkladem jsou:

- Smurf attack
- Fraggle attack

Specifické útoky

Doposud zmíněné útoky byly určené přímo proti jednotlivým SIP klientům či SIP proxy. Existují ale i další metody k dosažení podobných výsledků nebo útoky nepatří do žádné z dříve zmíněných kategorií. Použité metody bývají komplexní, vyžadují specifické podmínky a bývají použitelné jen pro úzký profil situací.

Častým zástupcem bývá SPIT, neboli nevyžádaný obsah šířený přes IP telefonii. Útočník využívá generátor hovorů pro přehrávání reklamních či jiných sdělení po vyzvednutí hovoru. Hovory mohou být generovány z interní i externí části IP infrastruktury.

VoIP infrastruktura může být ovlivněna i nepřímým útokem na některou z podpůrných služeb. Vzhledem k závislostem na ostatních technologiích a kombinaci s dalšími systémy (tarifikace hovorů, webová rozhraní, ...) se útočníkům otevírá řada možností:

- **DHCP** – DHCP Exhaustion
- **DNS** – DNS Cache Poisoning, DNS DoS flood
- **Provisioning server** – obsahuje konfigurace pro používaná zařízení
- **Gateway**
- **Webové rozhraní** – SIP proxy, billing systém,
- **Databáze** – využívána některými SIP proxy

Za vysoké částky se na černém trhu obchoduje s zero–day (nebo také 0–day) zranitelnostmi. Obsahují zranitelnost dané implementace či serveru, která ale ještě nebyla použita pro útok a nehrozí velké riziko snadného odhalení útoku (nejsou známy signatury útoky ani jeho průběh). Závisí však na konkrétní povaze zranitelnosti. Tyto hrozby patří mezi velká bezpečnostní rizika a byly důvodem vývoje behaviorálních analyzátorů provozu.

Nejkomplikovanější ale také i nejjednodušší formy útoků spoléhají na velkou slabinu v zabezpečení, kterou přináší lidský faktor. Social engineering, voice phishing (nebo i vishing) umožňuje získat přístup i do velmi dobře chráněných systémů a k citlivým datům. Vyžaduje však přípravu, znalost vnitřní organizace společnosti i herecký talent. Platí ovšem, že není třeba heslo prolomit, pokud ho interní uživatel ochotně sdělí.

Mnohdy bývají využity i zákeřné servery, tvářící se jako ty oficiální. Účelem bývá sběr přihlašovacích údajů, atd. Uživatelé jsou na tyto servery směrováni pomocí metod social engineeringu. Citlivé informace o zabezpečení systému lze získat i metodou dumpster

diving. Mnoho firem neřeší problematiku skartace citlivých dat a v odpadcích lze objevit velmi cenné informace využitelné k útokům.

Pro úspěšný útok je nutné i dobré načasování. Mnoho útoků s podvodnými hovory na placené linky proběhlo ve večerních hodinách či o svátcích, kdy nebyly jednotlivé systémy pod dozorem a útok mohl nerušeně probíhat delší dobu.

2.2.2 Zhodnocení rizik pro VoIP služby používající SIP protokol

Pro ohodnocení rizik jednotlivých druhů útoků musíme počítat s několika faktory. Jedním z nich je cíl útoku. Ačkoli je jakýkoliv úspěšný útok uvnitř VoIP infrastruktury kritický, záleží vždy také na dopadu útoku pro ostatní uživatele. Mezi odepřením služby pro několik uživatelů a všemi uživateli je značný rozdíl. Stejně tak představují větší hrozbu útoky mířící přímo na zranitelnosti protokolu SIP, než útoky proti konkrétním serverům, klientům či softwaru. Slabiny SIP protokolu ohrožují celou VoIP infrastrukturu nehledě na vlastnosti ústředny.

Jednoduchost provedení útoku či množství potřebných informací jsou dalšími faktory pro analýzu rizik VoIP útoků. Množství útočných nástrojů má v dnešní době implementováno grafické rozhraní a umožňuje spustit řadu variabilních útoků bez potřeby hlubších znalostí nejen funkcionality SIP protokolu, ale i cíle útoku a infrastruktury napadané sítě. Právě nízké nároky na znalosti útočníka a široká škála generovaných útoků činí tyto nástroje velice nebezpečné [20].

Situace nezlepší ani použití SSL/TLS zabezpečení. Odstraní se sice některé zranitelnosti (např. odposlech SIP zpráv), jiné ale nadále zůstávají. Šifrování provozu neochrání cílový SIP server před flood útoky nebo zmanipulovanými SIP zprávami [Saf12]. Útočník navíc nepotřebuje mít informace o použitých zabezpečeních a některé útoky šifrování nijak neovlivňuje. SIP servery se snaží i s poškozenými SIP zprávami jednat jako s těmi legitimními, což opět přispívá k vyšší zátěži serveru [13].

Každý úspěšný útok poškozuje dobré jméno poskytovatele a může mít na poskytovatele velký dopad. Nemusí se vždy jednat pouze o problém ztráty části zákazníků či poškození dobrého jména, mnohdy dochází k velkým finančním újmám díky laxnímu zabezpečení a ignorování základních bezpečnostních pravidel. Bezpečnost jakéhokoliv systému je komplexním problémem a vždy hrozí riziko napadení systému. Jediným východiskem je neustálé vylepšování stávajících metod obrany, školení personálu, kvalitní bezpečnostní monitoring a restriktivní politiky minimalizující případný dopad.

3 AKTUÁLNÍ STAV V OBLASTI DETEKCE ÚTOKŮ

Předchozí kapitola obsahuje výčet základních zranitelností SIP protokolu. Jak uvádím i ve zhodnocení rizik, každý z těchto útoků představuje pro VoIP infrastrukturu rozdílnou míru rizika. Komplexní systémy detekce útoků vyžadují řadu informací z různých zdrojů. Pro zajištění efektivního a zároveň i maximálního pokrytí potenciálních hrozeb se dále v dizertační práci zaměřím pouze na útoky cílené vůči SIP proxy serveru. Ačkoliv tím část útoků nelze detekovat, pokryji tak největší skupinu útoků, které zároveň také nejvíce ohrožují VoIP infrastrukturu samotnou.

S rozvojem různých útočných technik roste zároveň také počet způsobů jejich následné detekce. Samozřejmě se zvyšuje i počet metod pro blokovaní či zmírnění dopadu jednotlivých útoků. Nicméně neustálý vývoj v oblasti VoIP, bezpečnosti i útoků přináší stále nové přístupy na obou stranách. Některé způsoby detekce vycházejí z předchůdců SIP protokolu a komunikace prostřednictvím počítačových sítí obecně, některé pak přinášejí vlastní, zcela nové metody. V kapitole uvádím souhrn vybraných postupů i zhodnocení jejich vhodnosti pro použití v rámci detekce útoků na protokol SIP. Zaměřuji se hlavně na detekci útoků pro SIP servery, které jsou nejčastějším a nejpravděpodobnějším cílem útoků.

3.1 Detekování útoků na aplikační úrovni

Nejzákladnějším druhem je detekování na aplikační úrovni. Rozdíl mezi implementacemi jednotlivých serverových SIP aplikací je propastný a úroveň software se velice liší. Některé aplikace mají přímo implementovány bezpečnostní prvky proti určitým útokům, jiné nedokáží správně pracovat ani se SIP protokolem samým. Informace o nestandardním chování či výjimkách se běžně ukládají do logů aplikací, úroveň podrobnosti nastavuje správce a nemusí být dostatečná. V opačném případě dochází k podrobnému logování a podstatné informace mohou v množství dat zaniknout.

Jakékoli logování vytváří dodatečnou zátěž k napadené aplikaci a případný dopad útoku zhoršit a zvýšit rychlost vyřazení serveru. Existují aplikace umožňující reagovat na informace v log souborech a například i zablokovat určitý provoz. Log soubory ale poslouží většinou až následně po útoku pro získání dodatečných informací. Díky rozdílné povaze útoků i úrovni implementace aplikací se v log souborech nemusí užitečné informace o útoku nacházet.

Současný trend rozvoje datových sítí pro mobilní zařízení zvýšil počet implementací SIP klientů i pro širokou škálu těchto zařízení. Některé publikace se zabývají i ochranou proti útokům přímo na mobilních zařízeních, především díky rapidnímu nárůstu výkonu mobilních zařízení [32]. Problematika je ale podobná jako u serverových SIP ústředen. Existuje velká řada platforem pro tato zařízení, každá s řadou SIP klientů s rozdílnou úrovní implementace a podporou vlastností SIPu.

Technologicky je tedy možné útoky na aplikační úrovni nejen detekovat, ale i blokovat. Z praktického hlediska je výsledek silně závislý na úrovni konkrétní implementace pobočkové ústředny. Nelze také ignorovat fakt, že případná detekce zhoršuje dopad vybraných útoků a SIP server či SIP klient naopak může činit zranitelnějším. Z těchto důvodů je nutné citlivé nastavení použitých detektorů na aplikační úrovni pro sběr dat o útocích a jejich klasifikaci.

3.1.1 Honeypoty

Speciální typem detekce útoků na aplikační úrovni jsou honeypoty. Honeypot aplikace se snaží co možná nejvěrněji emulovat chování a funkcionalitu produkčního SIP serveru, často s bezpečnostní slabinou. Účelem je nalákat případné útočníky k útoku na takto emulovaný server. Honeypot následně monitoruje a ukládá informace o chování útočníka, aby bylo možné analyzovat metodu útoku či použité nástroje.

Výsledná data o útocích lze následně využít pro další analýzy a zvýšení zabezpečení existující infrastruktury. Honeypoty obsahují cenná data o útocích, které nelze získat jiným způsobem. Výsledná data bývají velmi popisná a informace je nutné před analýzou roztřídit a zpracovat. Pro účely sběru informací o útocích na VoIP infrastrukturu tvoří honeypot aplikace nenahraditelnou část architektury a základní prvek i mnou navrženého systému.

Dionaea

Praktické testy různých honeypotů odhalily jejich silné a slabé stránky [Saf05, Saf10]. Nejvhodnější se ukázala aplikace Dionaea, která dokáže emulovat celou řadu serverových služeb a protokolů, mezi něž patří i VoIP ústředna využívající SIP protokol. Aktivita útočníka je monitorována a fragmenty ze SIP zpráv uloženy do SQLite databáze. Kromě plné podpory SIP metod RFC 3261 dokáže Dionaea dále simulovat uživatelské účty a koncová zařízení. Hovory na simulované účty mohou být obslouženy prostřednictvím připravených nahrávek a aplikace automaticky pořídí zvukový záznam komunikace s útočníkem.

Ačkoli existují nástroje pro vyhodnocení dat z Dionaea (např. DionaeaFR), neobsahují klasifikaci SIP útoků ani podrobné informace o průběhu útoku. Z hlediska analýzy VoIP útoků jsou tyto nástroje nedostatečné.

3.2 Detekování útoků na serverové úrovni

Detekce útoků na serverové úrovni sdílí některé vlastnosti detekce na aplikační úrovni, resp. ji nadále rozšiřuje. Specializovaná řešení pro monitoring stavu serveru (jakými jsou např. Zabbix či Nagios) slouží ke sledování různých parametrů (zatížení CPU, obsazená paměť, vytížení linek, ...). Omezení ale zůstává stejné jako u aplikační úrovně. Každé další zatížení a detailnější mechanismus detekce více zatěžuje cílový server.

Existuje řada aplikací pro zvýšení bezpečnosti proti útokům na SIP servery, přímo na serveru by však k jejich detekci či potlačování docházet nemělo. Informace zachycené na úrovni serveru lze využít v rámci okamžitého informování správců či získání dodatečných informací až po případném bezpečnostním incidentu.

3.3 Detekování útoků na síťové úrovni

K detekci útoků dnes nejčastěji dochází na síťové úrovni. Během dlouhé existence počítačových sítí byly vytvořeny různé nástroje pro monitoring a inspekci provozu v počítačové síti, stejně jako i metody blokování a potlačování nežádoucího provozu. V současnosti jsou k dispozici nákladné síťové sondy a analyzátory, obsahující sofistikované a komplexní metody nejen pro detekci, ale také účinné blokování známých útoků. Na druhou stranu existuje celá řada kvalitních open-source nástrojů, jejichž spojením lze dosáhnout velmi dobré úrovně detekce. Navíc je možné techniky detekce upravit či libovolně kombinovat.

Jednou z nejjednodušších metod je zachycení síťového provozu a následná analýza těchto dat. Existuje řada nástrojů i metod pro přesměrování celého či jen části provozu na síťové kolektory (např. port mirroring). Na těchto kolektorech lze nalézt detailní informace o veškerém provozu, sloužící k detailní analýze útoků. Následnou analýzu samozřejmě značně omezuje použití šifrování a vypovídající hodnota dat značně klesá. Příkladem open-source aplikací jsou Tcpdump nebo OpenFlow.

Pro analýzu toků v síti existuje nástroj NetFlow. Některé směrovací prvky a firewally umí exportovat informace o tocích v formátu NetFlow. Informace o tocích jsou ale agregované a dochází tak ke ztrátě části informací o útoku. Využití v případě VoIP spočívá v korelaci s ostatními metodami nebo jako doplnění již existujících informací.

Podobně jako u monitoringu toků umožňuje sledovat zatížení a další provozní parametry prvků v síti protokol SNMP. Funguje na bázi klient-server a zasílá informace o stavu síťových prvků na monitorovací server. Vyhodnocení zachycených dat provádí další aplikace. Pro účely detekce a klasifikace VoIP útoků představuje jednu z doplňkových metod.

Jako IDS se označují systémy určené přímo pro detekci útoků, průniků či anomálií v síti (pokud tyto systémy umožňují i blokování rozpoznaných incidentů, označují se jako IPS). Pro detekci bývá použit přístup hledání konkrétních signatur či anomálií v síťovém provozu. IDS/IPS systémy se pak k síti připojují přes zrcadlící porty nebo speciální konektory, určené k monitoringu sítě. K dispozici jsou také již předpřipravená detekční pravidla pro tyto systémy, jejich rozšířením či úpravou je možné detekovat a klasifikovat nejen SIP útoky. Výhodou je snadné nasazení a provoz. Systémy ale mohou odhalit jen ty útoky, které odpovídají konkrétním signaturám či vykazující specifickou anomalitu vůči legitimnímu provozu. Zástupci open-source aplikací tohoto druhu jsou Snort, Bro či Surricata.

3.4 Vybrané metody detekce útoků proti SIP protokolu

Kromě již zmíněných metod a aplikací pro detekci a klasifikaci útoků jsou dostupné i mnohé další koncepty či implementace. Problematika zranitelnosti SIP protokolu je známá a zabývá se jí mnoho publikací a odborných článků. Následuje shrnutí a popis vybraných metod a postupů popsaných v odborné literatuře.

3.4.1 Metody využívající IDS systém

Rozšíření již existujícího IDS pro detekci útoků otevírá jednu z cest pro implementaci vlastního algoritmu detekce útoků. Díky modulárnímu návrhu a licenčním podmínkám projektu Snort je možné připojit k stávajícímu systému vlastní moduly. Rozšíření nevytváří nové detekční jádro systému, ale předpřipravují data zpracovávaná tímto jádrem – tzv. preprocesor [48, 30]. Aplikace Snort obsahuje i vlastní preprocessor pro SIP protokol od verze 2.9.1 (rok 2011).

Zmíněné preprocesory míří na zpracování stávajícího provozu a detekování následných anomálií. Projekt SPADE provádí statistickou analýzu provozu, kterou ukládá do podoby pravděpodobnostních tabulek. Nejvyšší váhu přisuzuje nejčastějším výskytům paketů, s klesajícím počtem výskytů klesá i váha dalších paketů. Úroveň anomality provozu vyjadřuje hodnota pravděpodobnosti a relativní anomalita (od 0 do 1) [48]. Tímto způsobem probíhá detekce odchylky v síťovém provozu.

Druhý z preprocesorů sestavuje informace o legitimním provozu, které porovnává s aktuálními daty. Na rozdíl od projetu SPADE probíhá sestavení informací o legitimním provozu jenom v trénovacím režimu. Zachycený provoz je profilován a profil uložen do databáze. Jakmile je preprocesor v detekčním režimu, porovnává aktuální data o provozu s uloženým profilem [30]. Dle autorů vede delší doba profilování v trénovacím režimu ke snížení statistické chyby detekce. Počítají ale pouze se situací, kdy je detekován výhradně korektní SIP provoz. Pokud se během trénovacího režimu vyskytne útok, systém ho bude nadále považovat za korektní.

Jiný přístup spoléhá na metodu Holt-Winter určenou k modelování časových řad s charakteristickými trendy a periodicitou. Díky tomu je tento model užíván například k predikci síťového provozu. V rámci Snort preprocesoru sbírá metoda informace o předchozím chování sítě a predikuje následující provoz. Pokud tento provoz neodpovídá předpovídanému, proběhne generování informace o odchylce [17]. Rozšíření Snort pro detekci DDoS pomocí multidimenzionálních GMM nalezneme v [6].

U každého z přístupů dochází ke ztrátám informace o odchylce a jejím zdroji. Detekční schopnosti systémů nejsou dostatečně podloženy daty a jejich účinnost není ověřená na referenční sadě dat.

3.4.2 Statistické vyhodnocení provozu

Statistickou analýzou SIP provozu se zabývá řada publikací. Rozdílné jsou především použité metody, sledované parametry SIP zpráv i vlastnosti SIP protokolu. První přístup se zaměřuje na sledování parametru Session-Expires.

Jsou sestaveny hypotézy pro rozdělení odpovídajících systémům bez útoků (nulová hypotéza) a pod útokem. Proběhne ověření normality naměřených hodnot a otestování na podobnost s nulovou hypotézou pomocí Anderson-Darlingova testu. Detekční systém je testován na referenční sadě dat a dosahuje statisticky významných úspěchů (interval spolehlivosti $< 5\%$) [28]. Detekční mechanismus se ale bohužel váže na sledování pouze jediného parametru a detekuje jeden specifický typ útoku charakterizovaný právě manipulací s hodnotou Session-Expires.

Víceúrovňové zpracování vlastností SIP zpráv uvádí článek o profilaci VoIP provozu [38]. Autoři se zaměřují na statistické zpracování charakteristik SIP provozu s omezenou sadou sledovaných parametrů, která se liší dle úrovně. Rozdělení do úrovní (server, entita, uživatel) umožňuje udržet vyrovnanou rychlost profilování, paměťové a výpočetní nároky, úroveň bezpečnosti i sofistikovanost metody.

Výsledkem je analýza chování SIP provozu z hlediska vyváženosti požadavků a odpovědí (request/response), charakteristika chování registrací uživatelů a hovorů. Vyhodnocení umožňuje diagnostikovat nejen výkonnostní problémy SIP architektury, ale také detekovat útoky [38].

Níže uvedené metody aplikované pro detekci útoků pomocí statistických metod spoléhají v rozhodování na výpočet Hellingerovy vzdálenosti, která určuje kvantifikovanou podobnost dvou distribučních rozložení. Pro výpočet se užívá vzorce (3.1).

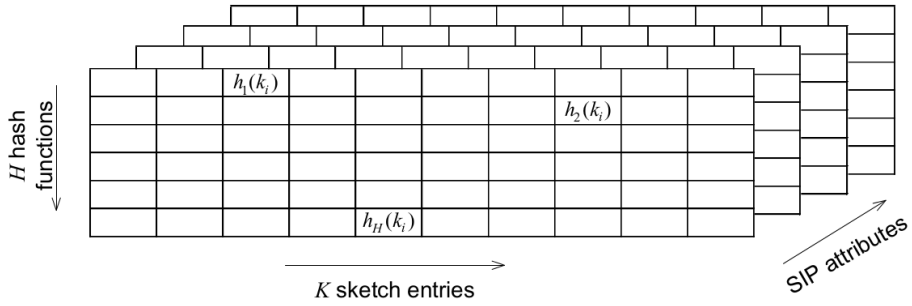
$$H^2(P_{HD}, Q_{HD}) = \frac{1}{2}\sum_{i=1}^{n}(\sqrt{p_i} - \sqrt{q_i})^2 \tag{3.1}$$

Hellingerova vzdálenost H^2 porovnává distribuční rozložení dat získaných během trénovací doby (hodnoty p) a rozložení získané během kratšího testovacího období (hodnoty q). HD byla k výpočtu použita pro své nízké výpočetní nároky, výsledným hodnotám vzdálenosti v intervalu $(0, 1)$ a s ohledem na analyzované parametry sledované SIP komunikace.

Trénovací interval trvá po dobu $n \times \Delta t$, testovací data jsou získávána po dobu Δt. V následujícím kroku začíná trénovací sada v periodě $n + 1$ a obsahuje data z předchozího testovacího období. Nové testovací období přidává navazující data s periodou Δt. Tímto způsobem se postupně upravují trénovací data o nové prvky, přičemž stará data jsou zahazována. Pro detekci flood útoků musí hodnota HD přesáhnout stanovenou hranici. Autoři za tuto hranici považují desetinásobek HD z předchozího kroku [35].

Tento koncept zpřesňuje použití sketch tabulky, pomocí níž lze detekovat i víceatributové útoky (tedy útoky pomocí více druhů SIP zpráv). Jako sketch se označuje metoda

náhodné agregace dat uložené v poli o H řádcích, kde H představuje různé hashovací funkce, a K sloupcích, které slouží pro ukládání jednotlivých prvků. Konkrétní pole v tabulce pak obsahuje informaci ve formě klíč–hodnota. Klíčem je SIP adresa zdroje, hodnotou je počet SIP zpráv.



Obr. 3.1: Metoda více sketch tabulek [24].

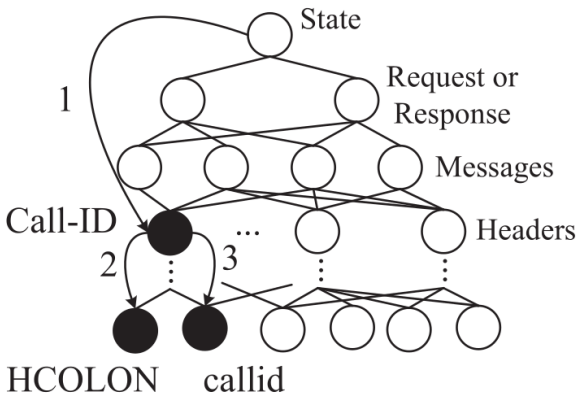
Pro každou novou SIP adresu se generuje nový záznam v každém řádku tabulky. Pokud již existuje klíč se stejnou hodnotou SIP adresy, hodnota ve všech polích s daným klíčem se navýší o novou hodnotu. K výpočtu indexu buňky v tabulce se využívají zvolené hashovací funkce. O útoku se rozhoduje na základě vyhodnocení informací z každého řádku sketch tabulky. Z výsledků je vytvořena sada dat, která je zkoumána pomocí metody HD [24, 33]. Jedna sketch tabulka odpovídá jednomu druhu SIP zpráv. Pro sledování více typů je nutné navýšení počtu sketch tabulek (3.1).

Aby útok neovlivňoval předchozí data v trénovací sadě, doporučují autoři zastavení aktualizace informací trénovací sady po dobu trvání útoku. Protože pomocí této metody nelze detekovat malé a postupně rostoucí počty útočných zpráv, lze metody výpočtu rozšířit i o techniku wavelet transformací [16]. Algoritmus je následně schopen detekce i velmi nízkých útočných toků, toků postupně rostoucích i útoků s distribuovaným charakterem.

3.4.3 Detekování anomálií pomocí stavového stromu

Metoda detekce útoků pomocí stavového diagramu nebo stromu se snaží vyřešit několik problémů spojených s detekcí SIP útoků. Těmi jsou velká variabilita útoků a efektivnost detekce (prohledávání signatur, výpočetně složitých algoritmů) jednotlivých útoků. Navrhované algoritmus využívá hierarchických korelací mezi jednotlivými stavy SIP komunikace. Výsledný rozhodovací strom tedy pracuje nejen s aktuální informací obsaženou v SIP zprávě, ale zároveň také s historií zpráv stejného zdroje.

Úrovně stromu zodpovídají za zpracování detailnějších informací ze SIP zpráv a snižují tak výsledné výpočetní zatížení. Použití stromové struktury přináší také logaritmickou rychlost prohledávání. Výsledkem je nejen detekce zahlcujících útoků a poškozených SIP



Obr. 3.2: Stavový rozhodovací strom [20].

zpráv (záměrně či nezáměrně). Díky monitorování stavů lze odhalit i další útoky spojené s nevyvážeností stavů SIP komunikace a poměrů zaslaných požadavků i odpovědí [20].

3.4.4 Metody detekce neuronovými sítěmi

Klasifikační systémy spoléhající na neuronové sítě se využívají především pro detekci anomálních stavů komunikačního systému. Odhalují, zda je na systém útočeno, nebo se jedná o normální stav. Velice často se neuronových síti využívá pro detekci útoků zahlcením SIP zprávami: DoS a DDoS.

Detekci známých i neznámých DDoS útoků se věnuje Saied. Článek popisuje využití dat z IDS systému SNORT a udávaná úspěšnost klasifikace dosahuje 98% [5]. Vysoké úspěšnosti v detekci průniku do sítě dosahuje s MLP i Moradi [45] K rozlišení DDoS útoků využívá Akilandeswari [19] neuronovou síť s aktivační funkcí Radial Basis. Stejné sítě využívá k detekci anomálního provozu i Karimazad [21]. Detailní popis odhalení DDos od náhlých špiček v komunikaci uvádí [2]. Problematiku distribuovaného zahlcení a výsledný efekt detailně rozebírá [11]. Porovnání algoritmů strojového učení pro detekci zahlcení obsahuje článek [29].

Detekci průniku do počítačové sítě pomocí modelů neuronové sítě s LVQ lze nalézt v [34, 27]. Implementaci IPS systému Sunshine pro detekci podvodných hovorů a zneužití UC popisuje Dirk Hoffstadt [12]. Využívá navíc různé detekční mechanismy a systém Sunshine je nasazen v reálném provozu.

Většina detektorů založených na neuronových sítích využívá dostupných sad se síťovým provozem a útoky generují v laboratorním prostředí. Přesnost detekce může být zkreslená a při reálném nasazení nemusí dosahovat výkonu z laboratorního prostředí. Zároveň se díky nedostatku dat o SIP provozu omezují jen na detekci normálního stavu systému a probíhajícího útoku.

3.4.5 Metody kombinující více algoritmů či systémů

Poslední skupina obsahuje metody využívající fúze klasifikátorů a různých detekčních systémů. Výsledky kombinace a návrh metody zlepšení finální přesnosti pomocí stromových modelů obsahuje [46]. Uvedená metoda je testována na různých datových sadách s dostatečnou škálou různých klasifikátorů.

Výsledky klasifikací SIP provozu klasifikačními algoritmy nástroje Weka popisuje Mehta [18]. Článek se zabývá i redukcí vstupního vektoru, která výrazně ovlivňuje přesnost některých algoritmů.

Spojením různých detekčních systémů lze výrazně zlepšit detekční schopnosti klasifikačních algoritmů. Fúzi vstupních dat z detektorů Netflow a IDS Snort popisuje Wang [37]. Výsledný typ útoku je určen MLP sítí. Podobný návrh přináší i Sharma [14], který detekci průniku provádí naučenou MLP sítí, ale vstupní vektor získává 3vrstvou kombinací několika detektorů. Kromě Netflow a IDS spoléhá i na údaje z firewallu, antivirového programu, interních logů OS aj.

4 STROJOVÉ UČENÍ

Oblast strojového učení je částí počítačových věd, zaměřená na dovednost učení bez nutného explicitního naprogramování. Vychází z metod pro rozpoznávání vzorů a umělé inteligence. K odhalování útoků využíváme jejich vlastností pro učení z klasifikovaných sad a provádění predikcí. Strojové učení úzce souvisí s problematikou výpočetní statistiky, optimalizačních problémů, data miningu atd.

Algoritmů strojového učení lze rozdělit do několika oblastí v závislosti na použitém principu učení:

- Učení bez učitele
- Učení s učitelem
- Zpětnovazebné učení

Pro učení bez učitele (Unsupervised learning) je typické, že učící data neobsahují požadovaný výstup. Algoritmy z této skupiny se snaží nalézt podobnosti mezi vstupními daty a jedince (instance) se společnými znaky přiřazovat do stejných skupin. Cílem může být odhalení skrytých závislostí mezi daty nebo určení podstatných atributů v datech (feature selection).

Během učení s učitelem (Supervised learning) získá algoritmus nejen vstupní data, ale i požadované výstupy. Algoritmus se následně snaží o nalezení obecného pravidla mapující vstupy na definované výstupy.

Při zpětnovazebném učení (Reinforcement learning) dochází k dynamické interakci algoritmu s prostředím, v němž plní algoritmus zadaný úkol. Samozřejmě existují překryvy a kombinace uvedených přístupů.

Protože se v této dizertační práci snažím o vytvoření obecného klasifikátoru útoků na SIP, zaměřím se dále na oblast učení s učitelem. Ani ostatní oblasti ale nezůstávají úplně opomenuty, v kapitole 7.4 se věnuji analýze vstupních dat pomocí shlukujících (Clustering) algoritmů. Zpětnovazebné algoritmy nejsou v této práci blíže rozebírány.

4.1 Učení s učitelem

Algoritmy využívající k učení již ohodnocených dat lze dále rozdělit do několika skupin podle podstaty využívaného principu, například:

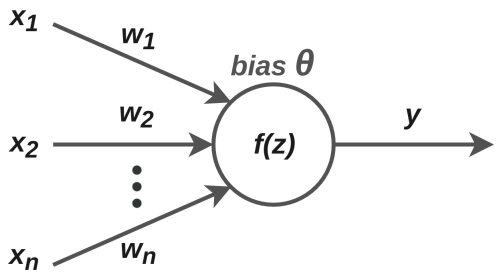
- Algoritmus k–nejbližších sousedů
- Bayesovské sítě
- Neuronové sítě
- Rozhodovací stromy
- Rozhodovací tabulky
- SVM

Algoritmy používané v této práci popisují následující části.

4.1.1 Neuronové sítě

Neuronové sítě jsou algoritmy které se inspirují biologickými neurony v mozcích savců. Koncept vychází z přenosu signálu mezi neurony přes nervová vlákna (axony a dendrity) a jejich spojení (synapse). Úkolem synapsí je zesílení či zeslabení přenášeného signálu.

Umělá neuronová sítě vychází z popsaného biologického vzoru a její model ve zjednodušené formě simuluje procesy probíhající v mozku. Každý neuron má definované vstupy (x) a výstupy (y). Vstupy neuronu zpracovává aktivační funkce (α), operující nad všemi vstupy. Variantou tohoto řešení je neuron s prahovou hodnotou (bias θ), která dále ovlivňuje výstup neuronu. Jednotlivé vstupy do neuronu mají definované váhy (w), které simulují funkci synapsí a slouží jako paměť neuronové sítě. Schéma neuronu zobrazuje obr.4.1



Obr. 4.1: Schéma neuronu.

Původní návrh perceptronu využíval skokovou funkci a zjednodušený model spoléhající na vážený součet vah. Z pohledu komplexních úloh je vhodnější využití spojité funkce, která se charakteristikou blíží skokové funkci. Výstup spojitého neuronu využívající aktivační funkci sigmoid se nachází na intervalu $(0; 1)$ a je definován rovnicemi:

$$z = \sum_{i=1}^{N} w_i x_i - \theta \tag{4.1}$$

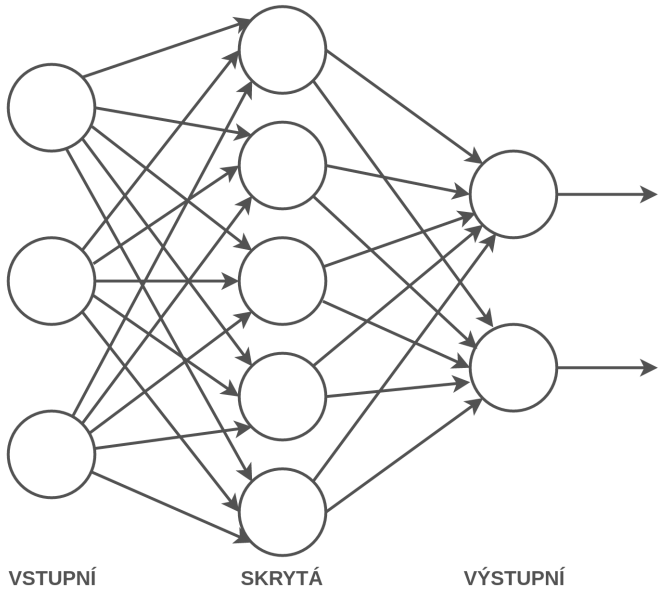
$$y = f(z) = \frac{1}{1 + e^{-cz}} \tag{4.2}$$

Vícevrstvé neuronové sítě

Podobně jako v lidském mozku, dochází i v neuronových sítích k propojení mnoha neuronů navzájem. Neuronová síť je potom schopna mapovat složité funkce a provádět generalizaci. Díky tomu je možné zpracovat i vstupy, se kterými se síť ještě nesetkala.

Každá neuronová síť se skládá z několika vrstev a neurony v jedné vrstvě jsou spojeny se všemi neurony vrstvy následující. Tyto vrstvy se rozdělují na tři skupiny. První vrstva se nazývá vstupní a předává hodnoty zpracovávaného vektoru dále do sítě. Skrytá vrstva přijímá aktivace neuronů vstupní vrstvy, zesílené či zeslabené vahami jednotlivých

spojení. Proběhne výpočet vnitřního potenciálu a šíření do dalších vrstev. Tento princip šíření signálu se označuje jako dopředný (feed–forward). Poslední vrstva je výstupní a slouží pro identifikaci výsledných tříd. Strukturu (topologii) jednoduché neuronové sítě se třemi neurony vstupní vrstvy, pěti neurony ve skryté vrstvě a dvěma výstupními neurony ilustruje obr. 4.2. Pro popis této topologie používám dále zápis: [3,5,2].



Obr. 4.2: Vrstvy neuronové sítě – struktura [3,5,2].

V praxi se využívají i sítě s vyšším počtem skrytých (vnitřních) vrstev, přičemž princip šíření signálu zůstává nezměněn. Ačkoli není počet vnitřních vrstev omezen, nesetkáváme se často se sítěmi s vysokým počtem skrytých vrstev (kromě tzv. deep–learning neural networks). Počet skrytých vrstev v praxi se omezuje na jednu, případně dvě skryté vrstvy. Více vrstev způsobuje problematické učení neuronových sítí a v praxi se s nimi až na výjimky nesetkáváme.

V literatuře se vyskytuje různé označením těchto sítí. Autoři používají i další zkratky jako například: DFF (Deep Feed Forward), BP-NN (BackPropagation neural network), MFF (Multilayer Feed Forward), MLP-FF (MLP Feed Forward), aj. Všechny tyto názvy odkazují na model MLP a tento název budu primárně v této práci používat.

Učení a metoda backpropagation

I když existuje více způsobů učení neuronových sítí, největšího úspěchu dosáhl algoritmus backpropagation představený již v 70. letech. Dnes je tento algoritmus základním kamenem pro učení neuronových sítí [10]. Cílem backpropagation je nalezení minima chybové funkce E, definované rovnicí 4.3 (dále odkazována jako QC).

$$E = \frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{m}(y_j - o_j)_i^2 \tag{4.3}$$

kde y_j představuje výstup neuronu j, o_j očekávaný výstup stejného neuronu, p je celkový počet prvků trénovací množiny a m množství neuronů výstupní vrstvy [55]. Při backpropagaci dochází ke zpětnému šíření signálu od výstupní vrstvy a dochází k postupným úpravám vah tak, aby docházelo k minimalizaci chyby na výstupu. Změnu váhy (4.4) ovlivňují i další parametry jako koeficient učení η a momentu μ určujícího vliv změny vah z předchozího kroku w_{ij}.

$$\Delta w_{ij} = -\eta\frac{\partial E}{\partial w_{ij}} + \mu\Delta w'_{ij} \tag{4.4}$$

Uvedená rovnice, resp. její první člen hledá směr a velikost růstu vektoru váhy z neuronu i pro neuron j v dané vrstvě, který je násoben parametrem η bránícímu oscilacím a tím i divergenci funkce. Parciální derivace je násobena -1 pro změnu směru chybové funkce k hledání minima. Druhý člen přenáší informaci o změně vah z předchozího kroku. Pokud uvedenou rovnici aplikujeme na celou sadu trénovací množiny, označuje se jako Gradient descent. V případě využití částí trénovací množiny (a zrychlení procesu učení) hovoříme o tzv. Stochastic gradient descent.

K výpočtu chyby dochází nejprve ve vrstvách vyšších, na základě výsledků pak může být vypočtena chyba neuronů v nižších vrstvách. Nielsen [10] definuje následující 4 základní rovnice backpropagation. Vektor chyb ve vrstvě L získáme dosazením do maticové rovnice 4.5

$$\delta^L = \nabla_y E \odot \sigma'(z^L) \tag{4.5}$$

$$\nabla_y E = (y^L - o) \tag{4.6}$$

Vektor chyb δ^L získáme jako Hadamard product vektoru gradientů $\nabla_y E$ a derivace vektoru aktivační funkce sigmoid $\sigma'(z^L)$. Chybu neuronu ve vrstvě můžeme poté vyjádřit pomocí chyby vrstvy následující jako:

$$\delta^l = ((w^{l+1})^T\delta^{l+1}) \odot \sigma'(z^l) \tag{4.7}$$

Chybu získáme pomocí transponované matice všech vah do následující vrstvy $(w^{l+1})^T$ a vektor chyb neuronů následující vrstvy δ^{l+1}. Chybu neuronu j ve vrstvě l lze tedy přímo vyjádřit rovnicí 4.8, která je zároveň chybou pro prahovou funkci.

$$\frac{\partial E}{\partial \theta_j^l} = \delta_j^l \tag{4.8}$$

Změnu vah v síti vyjádříme pomocí rovnice 4.9.

$$\frac{\partial E}{\partial w_{ij}^l} = y_j^{l-1}\delta_i^l \tag{4.9}$$

Kde y_i^{l-1} označuje výstup neuronu j v předcházející vrstvě $l-1$ a δ_i^l chyba neuronu i v aktuální vrstvě l.

Vylepšení backpropagace

V práci využívám pro neuronové sítě i další chybovou funkci, označovanou jako CEC – tzv. Cross-entropy (4.10).

$$E_{CEC} = -\frac{1}{n}\sum_x [o \ln y + (1-o)\ln(1-y)] \tag{4.10}$$

Uvedená rovnice udává chybu neuronové sítě pro n prvků trénovací množiny, kde x a o jsou vstupy a požadované výstupy neuronové sítě, y pak výstup pro daný vstup x. Výhoda uvedené funkce spočívá ve větší odezvě na saturované neurony. V případě použití kvadratické funkce dochází ke zpomalení učení, pokud je neuron silně excitován nebo silně potlačen. Cross-entropy funkce tento problém ve výstupní vrstvě řeší [10]. Chyba sítě E dosahuje vysokých hodnot pro saturované neurony a díky jiné derivaci nenastává při aktualizaci vah problém nízké hodnoty aktualizace vah jako u chybové funkce QC (více v [Saf01, Saf03]).

K podobnému stavu však může docházet i uvnitř neuronové sítě, což lze ovlivnit volbou vhodné inicializační funkce vah. Ty jsou na počátku učení většinou náhodně inicializovány a postupně upravovány procesem backpropagace. Účinek použité inicializace vah blíže rozebírám v experimentální části 8.1.2.

Značného vylepšení výsledků neuronových sítí dosáhneme i použitím předzpracování vstupních dat. Obvykle bývá preprocessing využit pro normalizaci vstupních dat. Příkladem uvádím lineární normalizaci 4.11 (také min-max normalizace), která vychází z hodnot zjištěných na učící sadě.

$$f_i^N = \frac{f_i - min(F_i)}{max(F_i) - min(F_i)} \tag{4.11}$$

Stanovením hranic pro jednotlivé prvky vstupního vektoru probíhá normalizace všech vstupů pro neuronové sítě i po naučení neuronové sítě. Výsledná hodnota parametru leží v intervalu $< -1; 1 >$.

Další typ předzpracování byl vytvořen přímo pro problematiku klasifikace útoků na SIP. Její princip spočívá v použití pseudobinárního zápisu jednotlivých parametrů. Určil

jsem hraniční hodnoty, které vychází z Fibonacciho posloupnosti. Pokud je hodnota parametru vyšší nebo rovna hodnotě hraniční hodnoty, je tato hodnota nastavena na 1, v opačném případě je rovna 0. Speciální případ tvoří stav, kdy je hodnota parametru přímo 0, což značí, že daný parametr nebyl během útoku detekován. Pro tyto případy existuje zvláštní hraniční hodnota, která označuje, že hodnota parametru je rovna právě 0 (a je tedy nastavena na 1, v ostatních případech 0).

Tímto způsobem jsem na základě min-max vyhodnocení učící sady navrhl odpovídají hraniční hodnoty pro každý parametr vstupního vektoru. Výsledkem je expanze vstupního vektoru neuronové sítě z 10 parametrů na 130 parametrů. Zlepšení výkonu neuronové sítě přináší možnost rozhodování sítě pomocí více hraničních hodnot, než je tomu v případě jednoduchých normalizací bez změny počtu vstupních parametrů. Použitím uvedeného pseudobinárního zápisu ale dochází ke ztrátě části informace, což může ovlivnit tvorbu učících sad a je nutné provést kontrolu sad i po provedení uvedené normalizace.

4.1.2 Rozhodovací stromy

Rozhodovací stromy patří k nejznámějším zástupcům algoritmů strojového učení. Skládají se z uzlů a listů, přičemž každý z uzlů představuje rozhodnutí na základě konkrétní podmínky. Touto podmínkou bývá většinou ta vlastnost vstupního vektoru, která rozdělí vstupní data na dvě či více podmnožiny s nejmenší chybou. Rozhodovací stromy rozdělují trénovací data do podmnožin, až dojde k dominanci jedinců z jediné třídy. Cílem rozhodovacího stromu je rozdělení vstupního vektoru do tříd (listů) pomocí uzlů, které představují sérii rozhodnutí. Při rozhodování se postupuje metodou specializace od shora dolů, tedy od prvního uzlu (kořenu) k listům [4].

Při trénování rozhodovacích stromů se tedy snažíme o nalezení vhodné struktury rozhodovacího stromu, která reprezentuje trénovací data. Existuje mnoho způsobů tvorby rozhodovacích stromů i možných vylepšení jejich rozhodování. Obecně se považují za vhodnější stromy s jednodušší strukturou. Následující typy rozhodovacích stromů byly použity v této práci.

HoeffdingTree

Řadíme mezi inkrementální rozhodovací stromy kategorie VFDT, využívající k rychlému učení stromu podvzorkování vstupních dat. Jsou určeny pro analýzu velkých sad nebo datových toků, které se mění v čase [4].

J48

Využívá algoritmus tvorby rozhodovacího stromu C4.5 vyvinutý Rossem Quinlanem, pro vytváření prořezaných i neprořezaných stromů. Je založen na konceptu informační entropie a částečně se podobá algoritmu ID3, který je jeho předchůdcem. J48 značí open–source implementaci algoritmu C4.5 v jazyce JAVA [59].

REPTree

Tvoří rozhodovací stromy na základě informačního zisku či rozptylu. K prořezání vytvořeného stromu využívá backfitting. Část funkcionality je podobná s algoritmem C4.5.

RandomTree

RandomTree algoritmus vytváří rozhodovací strom na základě K náhodně vybraných atributů. Neprovádí prořezání výsledného stromu. Umožňuje určit odhad posteriorní pravděpodobnosti výsledné třídy pomocí backfittingu.

4.1.3 Další vybrané algoritmy

Zde uvádím výčet algoritmů využitých pro klasifikaci útoků, které využívají různých principů pro rozdělení do tříd, avšak nespoléhají na kombinaci více algoritmů dohromady.

DecisionTable

DecisionTable představuje jednoduchý algoritmus založený na indukci z rozhodovacích tabulek. Využívá mapování majoritní třídy na pravidla, které se ukládají ve formě schéma (sady vlastností) do rozhodovací tabulky [58].

JRip

Algoritmus je optimalizovanou verzí IREP a spoléhá na učení pomocí sady pravidel. Metoda výběru těchto pravidel spočívá v opakovaném inkrementálním prořezáváním vedoucím ke snižování vytvářené chyby klasifikace [56].

OneR

Vytváří jednoduchý klasifikátor založený na sadě pravidel využívající k rozhodování pouze jeden parametr. Pomocí minimalizace chyby tohoto atributu na predikovaný výsledek vytváří výslednou sadu pravidel. [60].

PART

PART generuje rozhodovací pravidla pomocí metody rozděl a panuj. Využívá částečně C4.5 rozhodovací stromy a nejlepší listy převádí na výsledná pravidla [53].

IBk

Algoritmus klasifikuje data hledáním k-nejbližších sousedů. Snaží na základě podobnosti porovnávat nové instance s instancemi, se kterými se již setkal.

KStar

KStar je podobně jako IBk založen na porovnávání instancí, ale pro klasifikaci využívá entropii namísto podobnosti instancí [57].

Bayesovské klasifikátory

- **LWL** – Využívá naivní bayesovský klasifikátor pro učení lokálních vah jednotlivých instancí.
- **NaiveBayes** – Implementace naivního bayesovského klasifikátoru.
- **BayesNet** – Orientovaný acyklický graf, jehož hrany jsou pravděpodobnostní závislosti instancí.

4.1.4 Algoritmy kombinující klasifikátory

Skupina těchto algoritmů využívá pro svůj běh některého z dříve popsaných algoritmů. Kombinují ale výsledky z více naučených algoritmů dohromady za účelem zvýšení přesnosti celkové klasifikace. Tento postup vychází z principu, kdy skupina jednoduchých klasifikátorů s přesností přesahující náhodný klasifikátor dokáže překonat kombinací jejich výsledků komplexní algoritmus s vysokou přesností [52].

- **SimpleLogistic**
- **LogitBoost**
- **Bagging J48**
- **AdaBoostM1 J48**
- **RandomCommittee**

4.2 Učení bez učitele

V této práci využívám dále sadu algoritmů, které rozdělují data do tříd na základě shlukování bez znalosti výsledné třídy. Důvodem je ověření, zda vstupní data obsahují požadované třídy i nalezení a identifikace klíčových parametrů vstupního vektoru.

4.2.1 SOM

Samoorganizační mapy nebo také Kohonenovy mapy jsou typem neuronový sítí, které jsou tvořeny pouze dvěma vrstvami. První je vrstva vstupů a opět obsahuje počet neuronů odpovídající atributům (vlastnostem) vstupního vektoru. Druhou vrstvu tvoří síť vzájemně propojených neuronů, uspořádaných obvykle do trojúhelníkové, čtvercové nebo hexagonální konfigurace. Každý z neuronů v této vrstvě je propojen se svými sousedícími

neurony (v případě čtvercové konfigurace jsou to 4 neurony) a výjimku tvoří pouze hraniční neurony. Díky těmto se spojením se SOM výrazně odlišují od vrstev v MLP. Vstupní vrstva je plně propojena s výstupní vrstvou.

Při inicializaci sítě jsou vygenerovány náhodné váhy jednotlivých spojení a trénovacími daty je určen vítězný neuron Kohonenovy vrstvy podle vzdálenosti d_j vektoru vah rovnicí 4.12.

$$d_j = \sum_{i=0}^{n-1}(x_i(t) - w_{ij}(t))^2 \tag{4.12}$$

Přičemž x_i představuje vstupní vektor a w_{ij} váhu spojení od vstupního neuronu i k vítěznému neuronu j. Vítězný neuron následně upraví svůj vektor vah dle rovnice 4.13, aby se více blížil požadovanému vstupu. Zároveň však ovlivní i váhy ve svém okolí (tzv. sousedství) a sníží tak vzájemnou vzdálenost mezi nimi. Vzdálenost výstupní vrstvy reprezentuje míru podobnosti mezi neurony.

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t) \times (x_i(t) - w_{ij}(t)) \tag{4.13}$$

Váha v následujícím kroku $w_{ij}(t+1)$ se získá aktualizací současné váhy $w_{ij}(t)$ o hodnotu vzdálenosti váhy v závislosti na velikosti učícího koeficientu $\eta(t)$.

4.2.2 Clustering

Kromě SOM lze využít i řadu dalších algoritmů pro shlukování podobných dat. Protože se v práci vyskytují jen okrajově, uvádím pouze jejich stručný výčet.

- **Canopy** – Využívá pro shlukování canopy algoritmus, založený na odlišnostech instancí.
- **EM** – Přiřadí každé instanci pravděpodobností rozdělení, na základě nichž jsou pak instance shlukovány (simple expectation maximisation).
- **HierarchicalClusterer** – Provádí seřazení podle různých metod výpočtu vzdálenosti mezi instancemi.
- **FarthestFirst** – Jednoduché aproximační shlukování, vyhledává specifikovaný počet center shluků.
- **SimpleKMeans** – Podobně jako FarthestFirst, shlukuje instance s nejmenší vzdáleností od centra shluku na bázi K–means.
- **DensityBased** – Využívá distribuční rozložení a hustotu pro nalezení shluků instancí.

5 CÍLE DIZERTAČNÍ PRÁCE

Kapitoly 2.2 a 3 uvádí problém zabezpečení protokolu SIP a možných metod detekce útoků. Protokol SIP obsahuje několik závažných bezpečnostní nedostatků, pocházejících z jádra vlastního protokolu. Je jimi textová podoba protokolu, podobnost s dalšími protokoly, laxní definice SIP, široká řada doplňujících rozšíření, atd. Tento problém komplikují i různé úrovně implementace SIP řešení nejen pro serverovou část, ale i na straně klientů. Open-source podstata protokolu přináší značné výhody a stojí za masivním úspěchem tohoto protokolu. Přináší však mnohé negativní dopady zmíněné v dřívějších kapitolách.

Problematika zranitelnosti protokolu SIP je rozsáhlou oblastí a řeší ji řada odborných publikací. Širokému spektru slabin protokolu SIP, a s nimi spojených typů útoků, odpovídá i obdobné množství metod a technik, které SIP útoky detekují nebo blokují. Implementace těchto mechanismů a postupů na straně serverových SIP aplikací se značně liší a úroveň zabezpečení nemůže pokrýt všechny typy útoků. Stejná roztříštěnost jako u implementací SIP protokolu je charakteristická i pro detekci SIP útoků. Mnoho metod popisuje konkrétní postupy a metody detekce jednotlivých druhů útoků. Chybí ale sjednocující metodika, obsahující alespoň vybrané metody pokrývající většinu potenciálních SIP útoků.

Cílem této disertační práce je návrh a implementace nového distribuovaného systému pro sběr informací o SIP útocích. Využití těchto dat spočívá především v možnosti dalšího výzkumu. Systém umožní automatické vyhodnocení útoků získaných prostřednictvím sond pomocí prvků umělé inteligence, strojového učení nebo statistických metod. Výsledné informace, analýzy a klasifikace útoků mohou sloužit jako podklad ke zvýšení zabezpečení sítí s VoIP službou, k nalezení zařízení s vadnou konfigurací či problematickým SIP provozem a pro výzkumné účely.

Provedená rešerše odhaluje také značnou slabinu existujících způsobů detekce a složitost využití těchto technik v praxi. Systém sond pro sběr dat tuto situaci řeší pomocí snadného nasazení a jednoduché konfigurace sondy v síti. Tyto sondy budou spoléhat na existující open-source nástroje a budou dostupné v různých variantách. Umožní tedy zprovoznění sondy jak formou aplikace na již existujícím hardwarovém řešení v síti, tak i jako kompletní předpřipravený hardware.

Dalším nezbytným prvkem řešení je použití zabezpečené komunikace sond a centrálního serveru, obsahujícího také monitoring stavu jednotlivých sond. Funkcionalita navrhovaných bude experimentálně ověřena pomocí referenční sady reálných SIP útoků. Cíle tak, jak byly schváleny studijní komisí, sestávají z následujících bodů:

- **Vytvoření distribuovaného systému sond** pro sběr informací o útocích na SIP infrastrukturu. Sondy budou k dispozici v softwarové i hardwarové podobě, předpřipravené k použití v síti. Jednotlivé sondy jsou monitorovány a využívají zabezpečenou komunikaci pro výměnu dat o útocích.

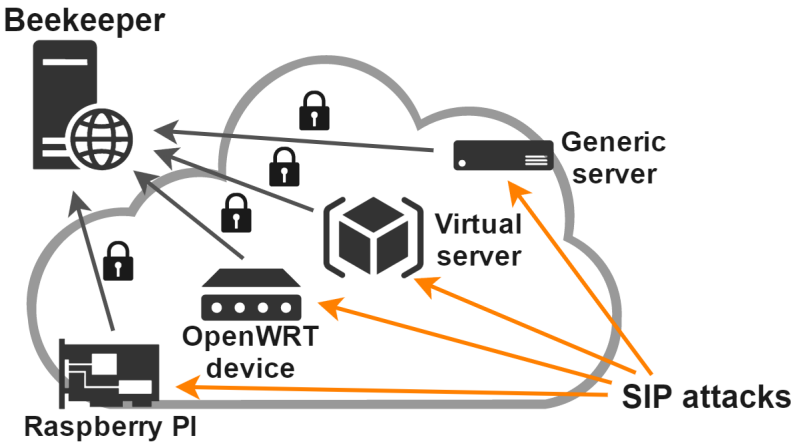
- **Návrh nového přístupu automatické klasifikace** používající k detekci prvků umělé inteligence a statistických metod. Server bude obsahovat minimálně 3 druhy různých klasifikačních metod.
- **Experimentální ověření funkcionality** na referenčních datech reálných SIP útoků a porovnání s existujícími metodami.

6 NÁVRH A IMPLEMENTACE ŘEŠENÍ

Předchozí teoretické kapitoly uvádí výčet různých útoků i metod využívaných pro jejich detekci. Ačkoli existuje řada dostupných implementací SIP serverů i SIP honeypotů, vlastní data o útocích jsou vzácná. Existují sice signatury známých útoků, chybí však veřejně dostupné datové sady obsahující SIP provoz či zachycené útoky.

Primárním cílem této práce je výzkum a vývoj systému pro autonomní klasifikaci SIP útoků. To by ovšem nebylo možné bez vytvoření odpovídající infrastruktury pro sběr a práci s útočnými daty. Celý koncept je založen na sondách sbírajících informace o útocích. Nasazení sond v různých sítích oddělených nejen logicky na úrovni topologie, ale i geograficky, umožňuje získat variabilní data o SIP útocích.

Zároveň s množstvím nasazených sond roste i potřeba automatického a zároveň i autonomního zpracování útočných informací. Celý koncept systému je zobrazen na obrázku 6.1. Skládá se ze dvou hlavních komponent, detekčních sond a centrálního serveru Beekeeper pro sběr dat a jejich klasifikaci.



Obr. 6.1: Koncept distribuované sítě honeypotových sond.

Uvedený koncept byl publikován a konzultován na řadě konferencí, například: [Saf05, Saf06, Saf08]. Prvotní návrh počítal pouze s nasazením sond s honeypot aplikací, postupně se ale systém rozšířil i na možnost využití plnohodnotných SIP proxy.

Celý systém je vyvíjen tak, aby byl modulární a jednoduše rozšiřitelný. Díky tomu je možné přidávat další klasifikační algoritmy a metody, stejně tak i další zdroje analyzovaných dat. Server pracuje se surovými daty a agregace probíhá až následně. Díky tomu je možné navrhnout i nové způsoby agregace. Zmíněný koncept byl v plné míře implementován a je aktivně využíván (viz. [Saf01]).

6.1 Detekční sondy

Základem vlastního detekčního systému jsou detekční sondy, sbírající data o útocích na SIP proxy server. Jak bylo uvedeno výše, data o útocích nebývají často zveřejňována a pokud tomu tak je, bývají často ořezána a zbavena podstatných parametrů. Veřejně dostupná data nemusí být kompletní a přístupná po delší dobu.

Data o útocích mohou být snadno generována v laboratorních podmínkách za pomoci řady nástrojů jako sipp, juno, inviteflood, sipvicious, atd. Cílem práce je ale navržení klasifikátoru pro reálné útoky, které útočníci aktivně využívají. Z těchto důvodů jsem přistoupil k nasazení vlastního systému detekčních sond, který je schopen poskytovat reálná data o SIP útocích dlouhodobě, a pokud možno i ve formě umožňující opakované zpracování dat v budoucnosti pomocí nových algoritmů.

Jak ilustruje již obr. 6.1, existuje několik druhů sond. Použití rozdílných druhů sond je způsobeno postupným vývojem a specifickými potřebami prostředí, v němž budou cílové sondy nasazeny. S rozdílným nasazením se současně mění i koncept jednotlivých sond a použitého software. Různé detekční metody a cílové SIP servery zároveň zvyšují variabilitu zachycených útoků.

6.1.1 Prvotní návrh sondy

Původní koncept sondy vycházel ze zkušeností s účinností SIP útoků a metod jejich detekce [Saf11, Saf09]. Součástí byly nejenom různé honeypot aplikace, ale také IDS Snort. Ačkoliv Snort disponuje značnými možnostmi detekce na bázi signatur, vlastní zpracování údajů o útocích představovalo širší možnosti výzkumu i vývoje klasifikátoru.

Nejlepší výsledky poskytla aplikace Dionaea, což je honeypot umožňující emulaci více druhů služeb, mezi něž patří i SIP PBX. Na základě informací z tohoto honeypotu jsem získal první vzorky SIP útoků, jejichž zpracováním jsem začal budovat systém Beekeeper, který by sloužil pro klasifikaci jednotlivých útoků.

Díky nasazení a testování těchto prvních prototypů sond se vyprofilovalo několik požadavků nezbytných pro následující generace. Mezi nejpodstatnější se řadí následující:

- **Open-source** – Použitý software by měl vycházet z open-source nástrojů. Nejedná se jenom o operační systém, ale i o nezbytné součásti software.
- **Modularita** – Software využívaný na sondách by měl být pokud možno nahraditelný a rozšiřitelný. Umožní to výslednou variabilitu nasazení i použití sond.
- **Jednoduchost** – Pro zajištění dostatečné variability dat o útocích musí být nasazení sondy jednoduché, intuitivní a do značné míry autonomní.
- **Zabezpečení** – Sonda nesmí představovat bezpečnostní hrozbu pro koncové sítě, v nichž bude nasazena.
- **Náklady** – Vzhledem k předpokládanému vysokému počtu nasazených sond nesmí cena detekční sondy znamenat vynaložení vysokých nákladů.

- **Výkon** – Detekční sondy musí být schopné emulovat chování SIP proxy a zároveň ukládat informace o přijatých SIP zprávách.

6.1.2 Aktuální druhy detekčních sond

Požadavky z návrhu splňují všechny dále uvedené typy sond. Ačkoliv se postupně vyvíjelo a měnilo softwarové vybavení jednotlivých sond, mají společných několik vlastností. Obecně pro všechny sondy platí využití návnady (honeypot) a monitoring probíhající komunikace útočníka s návnadou.

Každá z uvedených sond filtruje příchozí provoz pomocí firewallu. Komunikace může být povolena na libovolných portech, ve výchozím stavu jsou však otevřeny pouze tyto porty:

- **5060, 5061** – Výchozí port používaný SIP proxy (TCP i UDP).
- **22** – SSH z důvěryhodného subnetu.
- **68** – Port pro komunikaci s DHCP serverem.
- **ICMP request** – Zařízení odpovídá na ICMP request zprávy (ping).

Provoz na jiných portech je zahazován. Přístup k SSH je povolen jen z konkrétního subnetu a striktně povolen pouze s odpovídajícím klíčem. Autentizace pomocí hesla není povolena. Uvedená konfigurace firewallu se může lišit v závislosti na typu sondy.

Jednotlivé sondy by měly být v cílové síti umístěny tak, aby byly dostupné z internetu. Ideálním umístěním je demilitarizovaná zóna v nasazované síti. Možné je umístění uvnitř sítě s omezenou dostupností (např. pouze v rámci interní sítě), což ale výrazně snižuje množství zachycených útoků a tedy i efektivitu návnady.

Protože jsou jednotlivé sondy určeny přímo jako cíl pro útočníky, nelze vyloučit možnost prolomení zabezpečení. Sondy by tedy měly být nasazovány pouze do vyhrazených segmentů sítě se specifickými pravidly filtrování provozu, aby při případném nasazení nedošlo ke škodám v koncové síti.

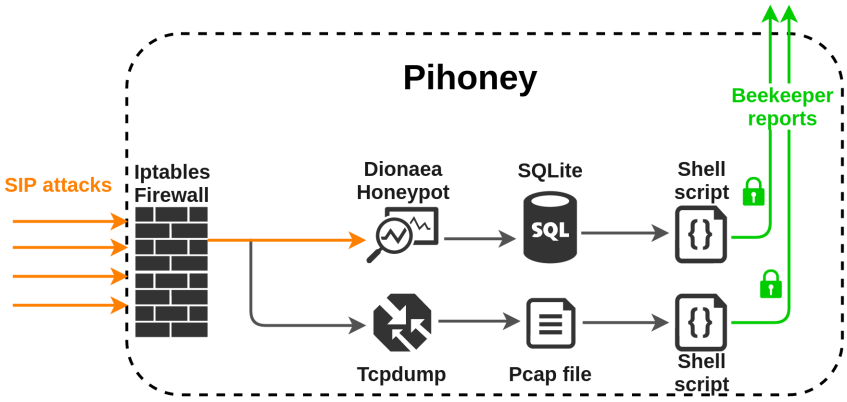
Shodným znakem pro všechny sondy je, že neprovádí vyhodnocení daného provozu. Důvodem je omezený výkon jednotlivých zařízení, ale i nutnost centrálního skladování zachycených dat ze všech sond v jejich nezměněné podobě. Pokud by probíhala agregace již na sondách, docházelo by ke ztrátám informace, které nelze následně na serveru odstranit.

Následuje přehled aktuálně dostupných typů detekčních sond, vyvinutých pro zachycení signatur útoků pomocí SIP protokolu.

Raspberry Pi sonda – Pihoney

Značný rozvoj v oblasti miniaturizace jednodeskových počítačů umožnil nasazení prvotního prototypu detekční sondy na mikropočítači Raspberry Pi. Díky tomu se povedlo také značně eliminovat cenu za jednu sondu, která dosahovala částky 2000 Kč. Raspberry Pi má ARM procesor taktovaný na 700 MHz a disponuje 512 MB RAM, díky čemuž má

dostatek výkonu pro potřeby monitoringu SIP komunikace. Data jsou ukládána na 16GB SDHC kartu.



Obr. 6.2: Schéma sondy pro Raspberry Pi.

První generace využívala Raspberry Pi model B s operačním systémem Raspbian (Debian pro Raspberry Pi). Díky tomu bylo možné využít stávajícího prototypu a provést migraci na OS Raspbian. S vývojem platformy jsem využil i následné modely jednodeskových zařízení, dosahujících vyššího výkonu a uváděné na trh v dalších letech po představení prvního modelu:

- Raspberry Pi 2 model B
- Raspberry Pi 3

Sonda spoléhá na honeypot Dionaea, který ukládá informace o provozu na honeypotu do SQLite databáze. Data o útocích jsou zpracovány skriptem *sendToBeekeeper.sh* a v pravidelných intervalech zasílána zabezpečeným kanálem (TLS) ke klasifikaci na server Beekeeper, viz. obr.6.2.

V dalších generacích byla přidána podpora monitoringu provozu pomocí packet snifferu. Tcpdump zachytává pouze provoz na portech vyhrazených pro SIP protokol v hodinových monitorovacích oknech. Informace z tohoto hodinového okna jsou uloženy do *pcap* souboru. Tento soubor je následně zpracován pomocí python aplikace *parser.py*, která vyextrahuje potřebné informace o SIP provozu podobně jako skript *sendToBeekeeper.sh*. Po zpracování je *pcap* soubor přesunut k archivaci a skript *tcpduplоad.sh* zašle report na server Beekeeper.

Ačkoli jsou veškerá data o útocích ukládána na serveru Beekeeper, každá sonda zároveň archivuje záznamy o provozu za poslední den a ponechává je dočasně uložené pro případ nedostupnosti či ztráty dat na serveru. V době nasazení prvních generací sond patřil projekt Dionaea k aktivně vyvíjeným a tvořil jádro detekčního systému. Tento projekt lze nadále využívat, nicméně již není aktivně vyvíjen.

Ukončení vývoje honeypotu Dionaea bylo jedním z impulzů pro jeho nahrazení jiným honeypotem či přímo reálným SIP serverem. Aby bylo možné nadále sledovat SIP provoz, implementoval jsem podporu pro analýzu z *pcap* souborů generovaných prostřednictvím aplikace *tcpdump*.

Vybrané softwarové požadavky na systém:

- Dionaea
- Tcpdump
- Python 3.2 a vyšší
- SQLite 3.3.6 a vyšší
- IPtables, Cron, Curl, ... – typicky již součástí OS

Nasazení sond na Raspberry Pi umožnilo zajistit testovací provoz několika sond rozmístěných v různých sítích a na několika místech v České republice. Mezi největší výhody tohoto hardwaru patří cenová dostupnost v kombinaci s dostatečným výkonem. Díky shodnému operačnímu systému s prototypem sondy bylo možné jednoduše zmigrovat existující skripty a programy. Problematice sond s Raspberry Pi se důkladněji věnuji např. v [Saf04].

Hlavní nevýhodou hardware Raspberry Pi byly problémy s diskovým úložištěm, které využívá SDHC kartu. Při reálném nasazení docházelo k poškození karet častými zápisy a čtením, na což není karta navržena. Životnost SDHC karet dosahovala od 3 do 19 měsíců. Tento problém lze částečně řešit omezením počtu zápisu a čtení na disk (například pomocí RAMFS).

Úspěch Raspberry Pi nastartoval rozvoj v oblasti jednodeskových PC a v současnosti existuje celá řada dalších podobných zařízení, například: Banana Pi, Carambola, Ordroid, BeagleBone, Arduino Uno, PandaBoard, Intel Galileo, Intel NUC, ...

Některá z těchto PC byla testována i v rámci skupiny LIPTEL. Jako nejslibnější varianta pro potřeby detekčních sond se ale ukázala platforma pro embedded zařízení OpenWRT, umožňující provozovat sondy na široké škále jednodeskových zařízení.

OpenWRT sondy

OpenWRT je linuxová distribuce zaměřená na vestavěné jednodeskové systémy (Embedded devices), konkrétně WiFi routery. Univerzálnost této distribuce s širokými možnostmi nasazení a centrální správy (provisioning) z ní ale dělá vhodný nástroj použitelný pro detekční sondy.

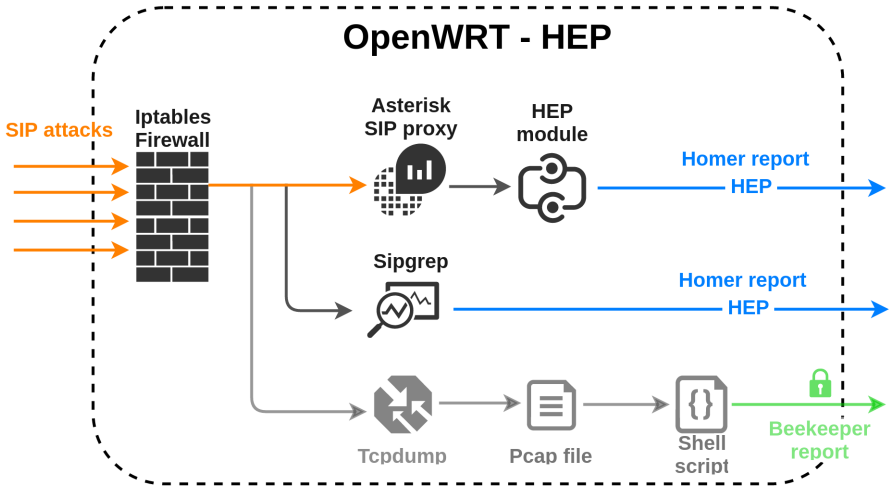
Výsledný obraz lze využívat i na zařízeních s velmi omezeným výkonem (již uvedené routery), ale i na výkoných jednodeskových PC (např. Intel NUC) nebo ve virtualizačních prostředích. OpenWRT obsahuje vlastní balíkovací systém, usnadňující instalaci potřebného softwaru.

Protože byl systém navržen pro potřeby routerů, bylo nutné do systému dokompilovat jednotlivé komponenty jako honeypot Dionaea a další. Kompilace přitom neprobíhá přímo

na daném zařízení, ale formou tzv. cross-compilation na jiném stroji, kde je vygenerován i obraz pro nasazení na cílovou architekturu.

Prvním úspěchem bylo přenesení existujících funkcionality navržené pro OS Debian i na OpenWRT. Ačkoli došlo k jistým úpravám, systém zůstává víceméně shodný se schématem 6.2.

Během nasazení první generace sond probíhalo i testování dalších metod detekce. Rozvoj detekční sondy byl způsoben mimo jiné i ukončením vývoje Dionaea. Účelem testů bylo nahrazení této kritické komponenty za reálný SIP proxy server a vytvoření plnohodnotného honeypotu.



Obr. 6.3: Schéma sondy OpenWRT se SIP proxy.

Změnou honeypot aplikace zmizela možnost získání dat přímo z cílové aplikace. Nabízí se ale možnost využití modulu pro export do monitorovacího systému Homer5 ve formátu HEP. Homer5 poskytuje exportní modul pro nejpoužívanější implementace SIP serverů:

- **Asterisk**
- **Kamailio**
- **OpenSIPS**
- **FreeSwitch**

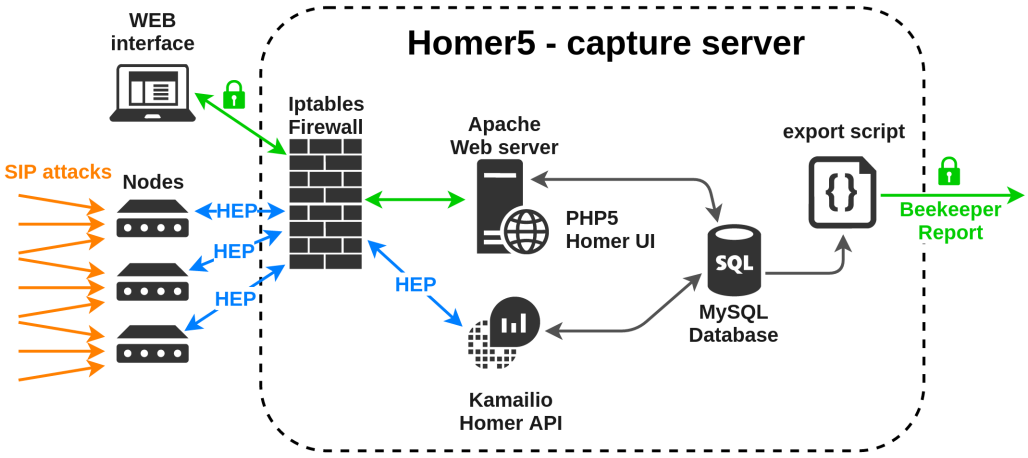
Pokud by bylo nutné využít některou z dalších implementací SIP proxy, nabízí se také možnost zachycení SIP provozu pomocí aplikací Sipgrep, SNgrep, CaptAgent. Pro testovací účely a oveření funkcionality zůstala stále zachována i varianta monitoringu provozu pomocí aplikace tcpdump, viz. obr.6.3.

Rozšíření honeypotu na reálný SIP server zvyšuje věrohodnost pro útočníky a zároveň umožňuje i vytváření komplexního chování SIP proxy. Z hlediska detekčních sond se snižuje závislost na použité technologii díky univerzální možnosti monitoringu SIP provozu. Problém představuje použití šifrované komunikace, kdy je možné monitorovat probíhající komunikaci interními moduly nasazenými v SIP ústřednách.

S použitím sledování provozu se nabízí možnost klasifikace provozu produkčně využívaných SIP serverů. Existují zde sice určitá omezení a komplikace, nicméně lze využít výše navrženou architekturu. Vyhledávání útoků v reálném SIP provozu je ale nad rámec této dizertační práce a není zde dále řešeno.

Využitím exportních modulů ale vzniká problém se zpracováním dat ve formátu HEP. Interně dochází na sondě k zrcadlení SIP provozu, který je převeden do formátu HEP a zaslán na odpovídající capture server. Na rozdíl od předchozích řešení tak dochází k téměř real-time přenosu informací na daný capture server. V případě DoS útoku či zvýšeného provozu na více sondách by mohlo docházet k zahazování části provozu a následně i ztrátě dat o průběhu útoků. Zvýšený provoz by také mohl způsobit DoS na straně serveru Beekeeper a znemožnit tak analýzu dat i ze sond, které na HEP nespoléhají.

Z uvedených důvodů tedy Beekeeper neobsahuje přímou podporu pro HEP protokol. Během testování HEP protokolu byl ale implementován vlastní capture server, který umožňuje zpracování HEP exportů z jednotlivých sond. Ke zpracování exportů využívám existujícího programu přímo od tvůrce HEP protokolu - capture server Homer5. Tato serverová aplikace slouží pro základní vyhodnocení SIP provozu a jeho statistik. Systém Beekeeper zůstává orientován pouze na analýzu zachycených útoků a není přímo ovlivnitelný útoky na sondy.



Obr. 6.4: Schéma funkcionality serveru Homer5.

Průběh zpracování HEP dat nástrojem Homer5 zobrazuje obr. 6.4. Server funguje jako prostředník mezi sondami a systémem Beekeeper. Provoz na Homer5 je filtrován pomocí firewallu a přenášená data mohou být šifrována, ve výchozím stavu ale není šifrování vyžadováno. Protokol HEP je navržen i se základní podporou autentizace sond, ta však ještě není plně funkční a server Homer5 přijímá data v HEP formátu z jakéhokoliv zdroje. Toto chování představuje potenciální bezpečnostní slabinu.

Informace o SIP provozu jsou na serveru Homer5 vyextrahovány z HEP dat (využívá se Kamailio) a uloženy do MySQL databáze. MySQL databázi využívá k zobrazení statis-

tik webové rozhraní HOMER UI (implementováno v PHP5, využívá web server Apache, náhled na obr. 6.5). Podobně jako v případě Dionaea jsem připravil skript, který extrahuje informace o útocích a zasílá je pravidelně (každých 10 minut) zabezpečeným kanálem na server Beekeeper.

Obr. 6.5: Homer5 - ukázka webového rozhraní.

Vzhledem k vlastnostem capture serveru Homer5 je vhodné jeho umístění v koncové síti společně s detekčními sondami. Umožní administrátorovi koncové sítě přehledně zobrazit průběh komunikace a základní statistiky provozu na jednotlivých sondách. Zároveň hrozí menší riziko zneužití než v případě použití jednoho serveru pro všechny sondy exportující data v HEP formátu.

Mezi hlavní výhody sond na OpenWRT platformě patří jejich univerzálnost a snadné nasazení v koncové síti. Každá ze sond je již připravena k nasazení a stačí ji pouze připojit do počítačové sítě. Sonda očekává přidělení IP adresy z DHCP serveru. Jediným krokem nutným ze strany administrátora je nastavení hesla pro účet root. Pokud je to nezbytné, lze každou ze sond dále konfigurovat pomocí CLI nebo prostřednictvím webového rozhraní LuCI. Konfiguraci jednotlivých zařízení lze také řešit pomocí funkce provisioning, kdy si jednotlivé sondy samy stahují konfigurace ze serveru k tomu určenému.

Mezi nevýhody patří složitá příprava každého obrazu, kdy je potřeba vytvořit obraz pro každou architekturu zvlášť. Omezení představuje i platforma OpenWRT sama o sobě. Jak již bylo uvedeno, původním cílem byla distribuce pro jednoduché routery. Kompilace dalšího softwaru tak nemusí být vždy možná díky vzájemně konfliktním závislostem různých aplikací.

Virtuální obrazy a obecný server

Poslední možností provozu sond představuje nasazení ve virtualizovaném prostředí nebo na již běžících strojích. Pro virtuální infrastrukturu jsou připravené obrazy založené na OS Debian i OpenWRT a funkčností odpovídající těmto řešením. Při vytváření obrazu pomocí upraveného build systému OpenWRT získáme pro 64-bitovou architekturu následující obrazy:

- beesip-honey_1.3.2-x86_64.img (331 MiB)
- beesip-honey_1.3.2-x86_64.img.gz (22 MiB)
- beesip-honey_1.3.2-x86_64.squashfs (19 MiB)
- beesip-honey_1.3.2-x86_64.sysupgrade (331 MiB)
- beesip-honey_1.3.2-x86_64.vmdk (22 MiB)
- beesip-honey_1.3.2-x86_64.vmlinuz (2.2 MiB)

Obraz beesip-honey_1.3.2-x86_64.vmdk lze přímo nasadit ve virtualizačním prostředí VMware. Ostatní obrazy lze bitově zkopírovat na cílová zařízení nebo provést upgrade starší verze firmware pomocí nástroje sysupgrade a odpovídajícího obrazu.

Software pro provoz sond je k dispozici jako open-source a je volně dostupný. Stejně tak jsou dostupné i připravené obrazy pro různé architektury. Detekční software může být nasazen také na již běžící servery, při splnění závislostí jednotlivých součástí. Je však nutné brát ohled na to, že zařízení bude pod útoky a může dojít ke kompromitaci již běžících služeb.

6.2 Beekeeper

Jádro dizertační práce tvoří informační systém Beekeeper. Tento systém byl od počátku vyvíjen jako datový sklad a autonomní klasifikátor SIP útoků. Postupně se však k systému přidávaly i další funkcionality. Hlavních funkce systému Beekeeper:

- datový sklad
- agregátor a čistič dat
- klasifikátor útoků
- monitoring a správa sond
- analyzátor klasifikovaných dat
- zdroj dat pro další expertní systémy

Základním kamenem úspěšné klasifikace jsou kvalitní data, která jsou pro Beekeeper získávána prostřednictvím sítě sond. Ačkoli existuje řada různých technik detekce útoků (viz. kapitola 3), mnohé se orientují pouze na detekci konkrétních typů útoků.

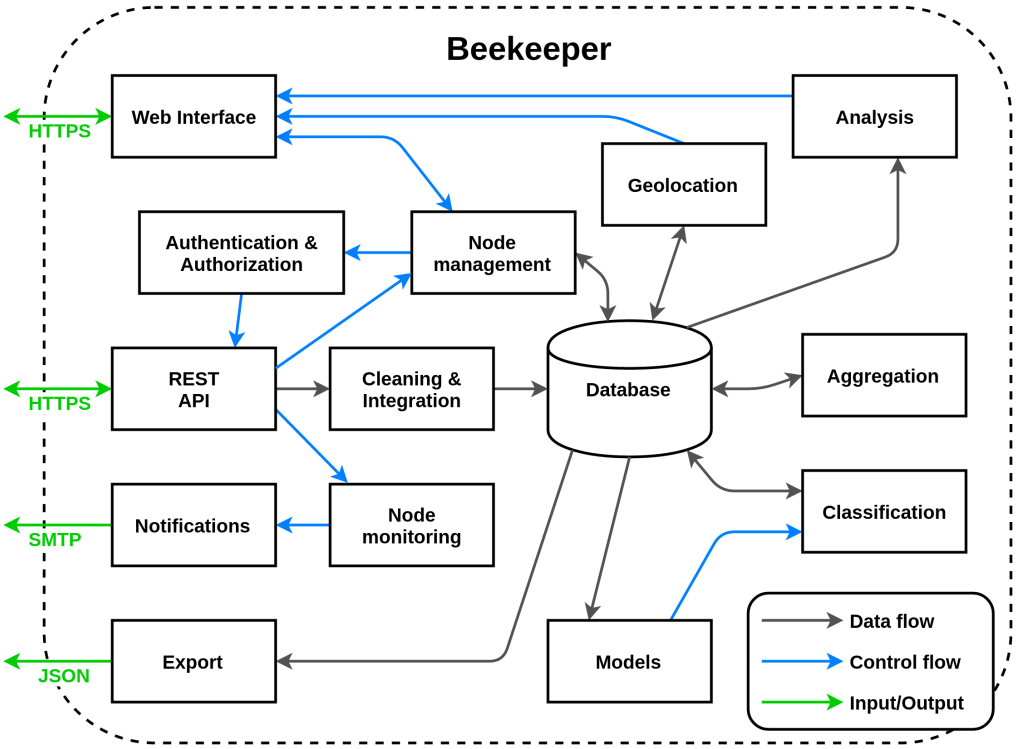
Klasifikátor navrhovaný pro systém Beekeeper byl vyvíjen pro obecnou klasifikaci škodlivého SIP provozu. Navržený systém umožňuje dosahovat vysoké přesnosti klasifikace útoků nezávisle na jeho typu. Mnoho systémů spoléhá při klasifikaci útoků na definované signatury, prahové hodnoty komunikace či anomality vůči normálnímu stavu.

Při použití metod strojového učení není nutné přesně definovat signatury jednotlivých typů útoků, stačí získat odpovídající a reprezentativní skupinu těchto útoků. Sítí honeypotových sond je možné tato data získat a v budoucnosti vytvářet i aktualizované a přesnější učící sady.

Možné zvýšení přesnosti výsledků klasifikace se nachází i ve fúzi několika různých klasifikátorů. Jednotlivé klasifikátory nemusí dosahovat vynikajících výsledků klasifikace a mohou chybovat v rozdílných případech. Bylo dokázáno, že řada jednoduchých klasifikátorů s vysokou chybou dosahuje lepších výsledků než jeden komplexní klasifikátor [52]. To samé obecně platí i při použití rozdílných klasifikačních metod. Výsledné spojení jednotlivých klasifikací je následně přesnější než nejlepší klasifikátor.

6.2.1 Architektura

Architektura informačního systému Beekeeper vychází z návrhového vzoru MVC. Pro jednodušší zobrazení funkcionality celého systému jsou jednotlivé funkce rozděleny do modulů, jak zobrazuje obr. 6.6. Systém umožňuje jednotlivé moduly přidávat, upravovat a odebírat bez ovlivnění funkcionality systému jako celku (pokud na sobě nejsou moduly přímo závislé).



Obr. 6.6: Modulární architektura systému Beekeeper.

Vstupy a výstupy systému

Základním vstupem do systému jsou data zasílaná z jednotlivých detekčních sond. Veškerá komunikace sond s Beekeeper probíhá přes REST rozhraní a šifrovaný kanál protokolu HTTPS (využívá se TLS 1.2). Pro testovací účely je dostupná i komunikace pomocí nezabezpečeného HTTP. V závislosti na typu sondy a použitém detekčním mechanismu se v HTTP POST požadavku přenáší data pomocí specifických parametrů a adekvátních datových souborů (CSV nebo JSON). Server zpětně informuje sondy o zpracování požadavku.

Webové rozhraní se využívá pro interakci uživatele se systémem. Využívá standardních metod používaných pro webová rozhraní (HTML, CSS a JavaScript). Výstupem ze systému jsou emailové zprávy zasílané notifikačním modulem. Beekeeper zároveň exportuje informace o útocích ve formátu IDEA (JSON) do jiných expertních systémů k následnému zpracování.

Aggregation

Úkolem agregačního modulu je zpracování surových dat ze sond. Ta jsou sice již uložena a naformátována v odpovídající struktuře v databázi, pro jejich zpracování je ale nezbytná agregace. V aktuální verzi probíhá agregování pomocí posuvného pěti minutového okna.

SIP zprávy se nejprve rozdělí podle zdroje útoku (IP adresa útočníka). Následně probíhá agregace využívající časové známky o přijetí jednotlivých paketů. Všechny pakety, které byly na sondu doručeny během pěti minut od posledního přijatého paketu z daného zdroje, se agregují do jedné struktury.

Pokud je časový interval delší než pět minut, je současná skupina uzavřena a vytvořena nová, obsahující pouze aktuálně zpracovávaný paket. Struktura agregované zprávy (obr. 6.13) již téměř odpovídá vstupu pro klasifikační algoritmy (popsáno dále v kap. 6.2.4), obsahuje navíc pouze další informační atributy jako geolokace, atd.

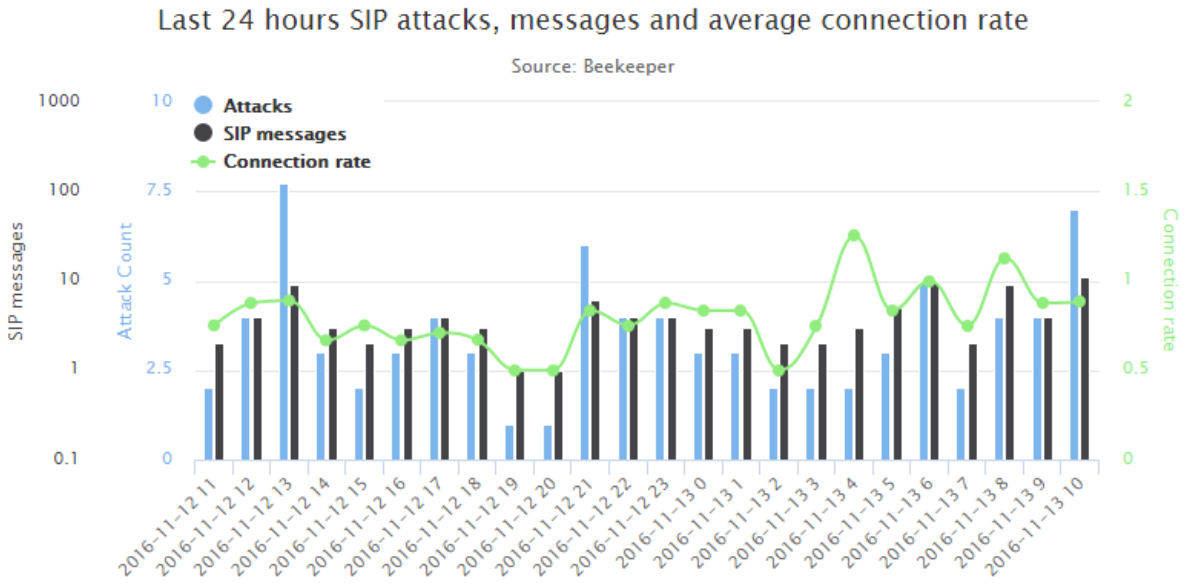
Analysis

Modul pro analýzu zodpovídá za vytváření základních analýz o průběhu útoků. Jsou jimi například informace o počtu útoků detekovaných od posledního uploadu nebo stav za posledních 24 hodin (rozděleno do jednotlivých hodin, viz. obr. 6.7).

Výsledná data z analýzy jsou zobrazována dynamicky přes webové rozhraní, pomocí něhož je také možné měnit jednotlivé parametry analýzy. Současně tento systém využívá i dat získaných pomocí geolokačního modulu.

Authentication & Authorization

Úkolem modulu je provádět autentizaci a autorizaci jednotlivých sond během nahrávání detekovaných informací. Každá nová sonda, která se pokusí o nahrání dat na server je



Obr. 6.7: Přehled útoků za 24 hodin, Beekeeper.

zařazena na seznam čekajících. Administrátor musí následně povolit příjem dat z této sondy ve webovém rozhraní.

Sonda musí pro upload vždy používat specifický autentizační kód, který je uložen zároveň na sondě i na serveru.

Classification

Klasifikační modul je základem systému Beekeeper. Pracuje s jednotnou strukturou dat, která vzniká po agregaci útoků. Agregované útoky jsou zpracovány dostupnými klasifikačními algoritmy. Výsledky detekce různými algoritmy jsou uloženy do databáze.

Důkladněji je klasifikační modul rozebrán v samostatné kapitole 6.3.

Cleaning & Integration

Jakmile jsou přijata data ze sondy, probíhá jejich kontrola. Testuje se, zda byla zaslána korektní data a převádí se do definované struktury, která je uložena do databáze. V této formě jsou data předána ke zpracování agregačním modulem.

Během agregace dochází k odstranění nadbytečných informací. Bez zachování původních dat by již nebylo možné ztracená data zpětně rekonstruovat. Původní data jsou důležitá především pro přípravu nových sad (trénovací, validační, testovací) i pro vývoj nových metodik agregace a klasifikace.

Jedinou daní za uložení zdrojových dat o útocích jsou zvýšené nároky na úložný prostor. Tato nadbytečná data je však možné pravidelně archivovat a ze systému odstranit.

Export

Výsledná data o útocích na sondy nejsou dostupná jen prostřednictvím aplikace Beekeeper. Zapojením Beekeeperu do dalších expertních systémů se zvýšila hodnota i další využitelnost těchto dat. Mezi tyto systémy patří bezpečnostní projekty vyvíjené sdružením CESNET:

- **Mentat**[1] – Distribuovaný modulární systém navržený pro monitoring sítí větších velikostí.
- **Warden**[2] – Systém pro sdílení informací o detekovaných bezpečnostních událostech.
- **SABU**[3] – Systém pro inteligentní analýzu a efektivní předávání informací typu bezpečnostní událost a incident.

Klasifikovaná data jsou do zmíněných systémů exportována pomocí tohoto modulu ve formátu odpovídajícímu nárokům cílového systému. Lze určit pouze konkrétní sondy, pro něž budou data exportována.

Geolocation

Geolokační modul zprostředkovává lokalizaci zdrojové IP adresy použité pro útok. V závislosti na dostupných informacích je určen stát, kraj, město a GPS souřadnice. Informace o státu je přístupná pro většinu vyhledávaných IP adres, další upřesňující údaje už nemusí být vždy dostupné.

Modul spoléhá na data z již vytvořených databází, namísto aktivní detekce lokality útočníka. Dochází tedy k určitému časovému posuvu od doby zaregistrování IP adresy a detekce útoku. Informace o původu útoku může být i snadno podvržena (především u DoS útoků). Pro potřeby analýzy jsou však tato omezení akceptovatelná a nepředstavují výrazný problém.

Lokalizace IP adresy probíhá nezávisle na dalších modulech a je k jednotlivým útokům přidávána podle přístupnosti této informace. Důvodem je závislost na externí službě, která má omezení počtu lokalizovaných IP adres a nemusí být neustále dostupná.

Models

Obsluhuje různé klasifikační algoritmy a jejich konfigurace. Slouží nejenom pro spouštění klasifikace pomocí těchto algoritmů, ale také jako jejich přehled a nástroj pro správu. Umožňuje správci určit, které algoritmy a s jakou konfigurací budou použity.

[1] https://mentat.cesnet.cz
[2] https://warden.cesnet.cz
[3] https://sabu.cesnet.cz

Node management

Modul slouží ke správě dostupných sond. Obsahuje výčet aktivních sond v systému a časové známky poslední komunikace se systémem Beekeeper. Umožňuje autorizovat nové sondy, deaktivovat autorizované sondy. Zároveň je možné spravovat parametry specifické pro konkrétní sondu (např. autentizační klíč).

Node monitoring

Pracuje v součinnosti s předchozím modulem. Modul sleduje komunikace sond se systémem, uchovává informace o proběhnuté komunikaci. Detekuje nové sondy, které se pokouší navázat kontakt s Beekeeperem a udržuje jejich seznam.

Hlavní náplní je ale sledování chyb v komunikaci. Pokud k takovéto chybě dojde, předává informace do notifikačního modulu. Zároveň tento modul sleduje i odchylky v pravidelných výměnách informací se sondami. V případě více než 3 hodinové odmlky sondy je notifikován administrátor.

Notification

Systém Beekeeper pracuje autonomně a vyžaduje interakci s administrátorem pouze v několika případech. Pokud dojde ke stavu, kdy je nutný zásah ze strany administrátora, zašle tento modul informaci o stavu systému pomocí emailové zprávy na adresu administrátora. Jak jsem již uvedl, častým zdrojem těchto notifikací je modul pro sledování komunikace sond.

REST API

REST rozhraní obsluhuje požadavky ze sond. Využívá a předává informace mezi různými moduly Beekeeperu. Pro sondy jsou připraveny metody s URL, které obsuluhují pouze požadavky typu GET a POST, nutné pro komunikaci sond se systémem.

Ačkoli Beekeeper udržuje v databázi i surová data ze sond, je nutné podotknout, že tato data jsou již do jisté míry upravena díky zásahům z integračního modulu (Cleaning & Integration module). Pro uchování dat ve zdrojové podobě (ve formátu odeslaném ze sond) provádí systém také ukládání nahraných souborů a jejich každodenní archivaci. V případě potřeby je možné data rekonstruovat (chyby či selhání některého z modulů manipulujících s daty) z těchto záložních souborů. Tato vlastnost slouží hlavně k vývoji a testování nových metod zpracování dat či jejich agregace a v produkčním nasazení ji není nutné využívat.

Web interface

Webové rozhraní slouží pro interakci uživatele se systémem. Umožňuje snadný přístup k datům, správu systému i jednotlivých sond, jednoduché vyhledání informací o útocích

a jejich vizualizaci.

6.2.2 Implementace systému Beekeeper

Systém Beekeeper je implementován jako webová aplikace v jazyce JAVA 8, přičemž využívá framework Spring pro tvorbu moderních MVC webových aplikací a REST služeb. Jednotlivé webové stránky jsou napsány v HTML5 a využívají JavaScript knihoven JQuery a HighCharts. Kompletní aplikace je distribuována pomocí jednoho *war* archivu, jenž obsahuje všechny komponenty aplikace.

Data jsou ukládána do MySQL databáze pomocí JDBC konektoru. Před nasazením Beekeeperu je tedy nutné zprovoznění databázového serveru s odpovídající databází a přístupem. V budoucnu je možné snadno nahradit MySQL databázi za jinou technologii, vhodnější pro ukládání velkého množství dat (například MongoDB, NoSQL, Hadoop).

Výsledný *war* archiv lze nasadit v různých aplikačních kontejnerech, doporučuji však webový server a servlet kontejner Tomcat8. Ověřená konfigurace byla otestována s těmito komponentami:

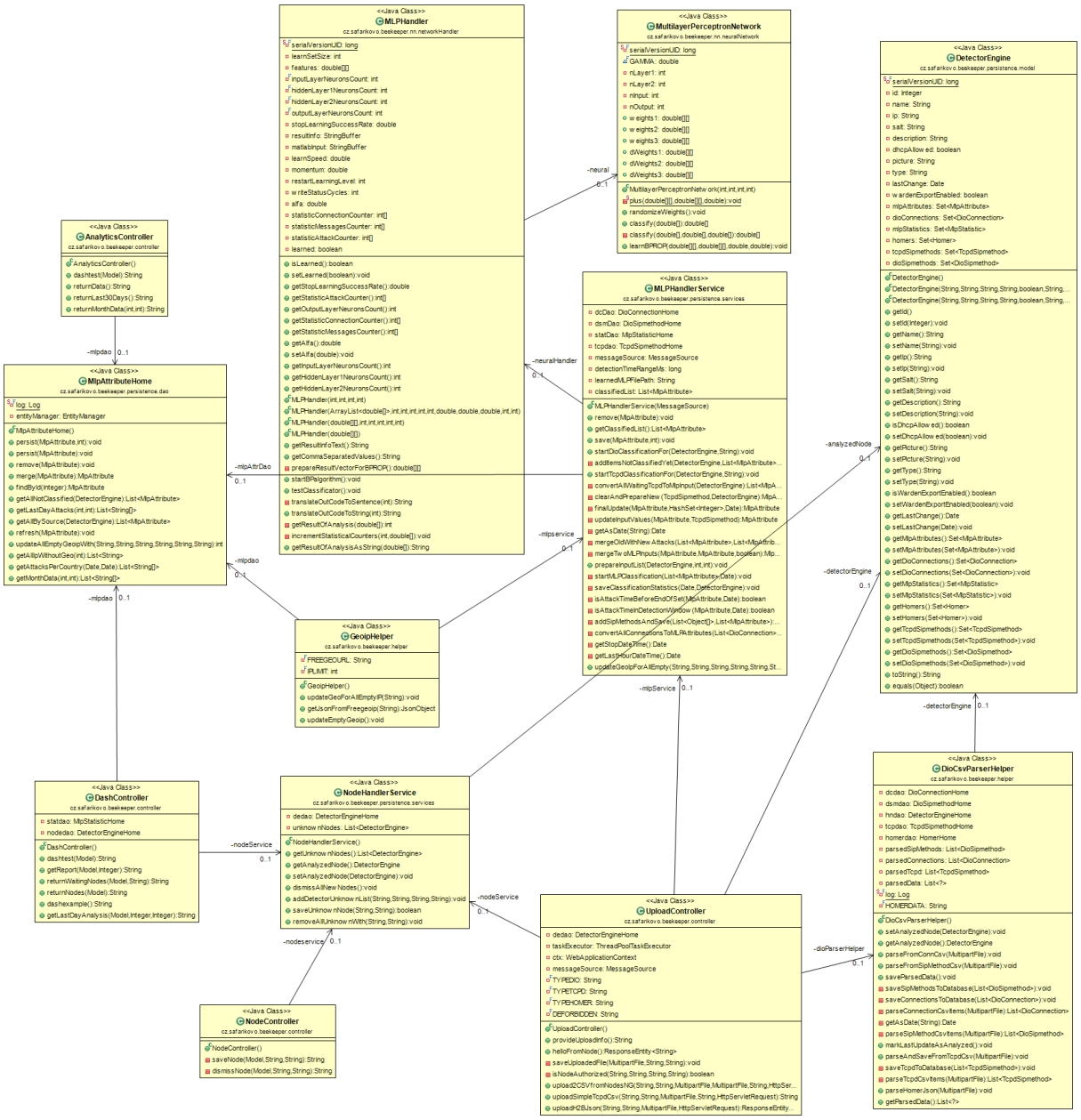
- **Apache2** verze 2.4.10+ (webová proxy)
- **Tomcat** verze 8.0.15+ (webový server a kontejner)
- **MySQL server** verze 5.5+ (databáze)
- **Mentat-client** verze 0.6.81+ (klient pro exporty do dalších systémů)

Detailní postup pro nasazení aplikace je uveden v dokumentaci a webových stránkách repozitáře (více E) se zdrojovými kódy pro aplikaci Beekeeper.

Systém Beekeeper byl vyvíjen postupně a mnohé funkce byly přidávány až v dalších verzích. Původní systém pracoval pouze s jednoduchou sondou na bázi Pihoney s honeypotem Dionaea (viz. sekce sondy Pihoney v kap. 6.1.2). Průběh implementace funkcí systému shrnuje stručný výčet verzí systému Beekeeper (některé vlastnosti a funkce byly pro zjednodušení vynechány).

- **Prototyp** – Testovací prototyp (až do verze 6). Produkčně nasazená verze prováděla agregaci dat ze sondy, klasifikaci útoků, sledovala statistiky průběhu klasifikace a obsahovala jednoduché WUI.
- **0.0.7** – Přidána podpora zpracování tcpdump dat, přidáno rozhraní pro analýzu.
- **0.0.8** – Úpravy exportního modulu, opravy modulu pro integraci a agregování útoků (především spojování a časové známky).
- **0.0.9** – Dodává monitoring komunikace sond se systémem, přidání notifikačního a geolokačního modulu (s odpovídajícím rozhraním).
- **0.0.10** – Opravná verze, řešící řadu problémů ve zpracování dat ze sond, čištění a refactoring kódu. Opraveno a přidáno logování kritických akcí systému.
- **0.0.11** – Podpora pro data z Homer5, zavádí důkladné zálohování dat.
- **0.0.12** – Současná verze systému (plný přechod na HTTPS).

Obrázek 6.8 zobrazuje závislosti vybraných komponent pracující s klasifikátorem na bázi MLP a jejich celkového zapojení v systému Beekeeper.



Obr. 6.8: Třídní diagram BKPR pro MLP.

Protože se práce zabývá klasifikací SIP útoků, věnuji se dále pouze popisu dat a implementaci komponent úzce svázaných s touto problematikou.

6.2.3 Struktura importovaných dat

Jak již bylo uvedeno, systém Beekeeper zpracovává data z různých druhů sond. Jednotlivé sondy a detekční mechanismy na nich použité používají různé metody uložení dat. Díky

tomu se liší i struktura dat exportovaná z těchto sond.

Je důležité podotknout, že ze sond se exportují informace v neagregované formě a pokud je to možné i v neupravené podobě. Bylo by možné data již předzpracovat na každé sondě, nicméně by tím došlo ke ztrátě původní informace o útoku. Během vývoje a testovací fáze došlo několikrát k selhání sondy a ztrátě dat na ní uložených.

Využitím modelu, který předpokládá kompletní zpracování dat na serveru, můžeme ovlivnit, jak bude se zdrojovými daty nakládáno (zda budou archivována nebo pouze agregována a zahazována). Zároveň zůstává koncept sondy jednoduchý a nenáročný na zdroje. DoS útoky jsou schopny během krátké doby vygenerovat gigabyty dat, které by byly uloženy na sondě. Nevýhodou je nutnost přenášet data sítí na Beekeeper. Centralizovaný server, na rozdíl od sond, disponuje dostatečným výkonem a zdroji pro zpracování útočných dat.

Data sondy získají pomocí nejméně jednoho ze tří typů detekčních mechanismů, kterými jsou:

- honeypot Dionaea
- monitoring provozu Tcpdump
- Homer5 data (HEP)
  - monitoring provozu Sipgrep, Sngrep
  - exportní moduly pro PBX

Dionaea data

Data z honeypotu Dionaea jsou exportována ve formě dvou csv souborů. První z nich obsahuje záznamy o navázaných SIP relací s honeypotem (obr. 6.9). Konkrétně to jsou: identifikátor relace, použitý transportní protokol, časová známka navázání relace, rodičovské záznamy (odkazy na jiné záznamy), ip adresa a port útočníka.

```
196889,udp,SipSession,"2016-10-03 09:45:22",196889,,94.23.8.146,5071
196890,udp,SipCall,"2016-10-03 09:45:22",196889,196889,94.23.8.146,5071
196891,udp,SipSession,"2016-10-03 09:49:08",196891,,89.163.204.25,5076
196892,udp,SipCall,"2016-10-03 09:49:09",196891,196891,89.163.204.25,5076
196893,udp,SipSession,"2016-10-03 09:57:57",196893,,158.69.244.172,5121
```

Obr. 6.9: Ukázka exportu dat z Dionaea - connections.

Druhou část tvoří další csv soubor s informacemi o využité SIP metodě. Z každé SIP zprávy jsou vyextrahovány parametry do tohoto formátu: interní identifikátor, odkaz na SIP relaci, SIP metoda, Call_id a identifikátor klienta. Příkladem je obr. 6.10 níže.

Uvedený výčet parametrů závisí na honeypotu Dionaea. Pro nezávislost na použitém honeypotu lze využít dat získaných pomocí sledování provozu.

```
182884,196888,INVITE,3fa33a49d621716722a3357fb1008ebe,sipcli/v1.8,46
182885,196890,INVITE,ece959944a187beac42765ec581f9186,sipcli/v1.8,46
182886,196892,INVITE,87f5f8f62d1efdbcf339e668c2cb82d7,sipcli/v1.8,46
182887,196893,OPTIONS,157428148635632297991066,friendly-scanner,0
```

Obr. 6.10: Ukázka exportu dat z Dionaea - SIP methods.

Tcpdump data

Aplikace tcpdump monitoruje provoz na sondě v pevně stanovených časových intervalech. Po uplynutí tohoto intervalu dojde k vygenerování pcap souboru se záznamem komunikace ve stanoveném intervalu. *Pcap* soubor je zpracován pomocí python parseru, který vygeneruje csv soubor s podobnými informacemi jako je tomu i v případě Dionaea.

Exportovaná data lze s ohledem na prováděnou klasifikaci značně zjednodušit. Pro zachování informací odpovídajících exportu z honeypotu Dionaea probíhá export následujících hodnot: časová známka přijetí zprávy, použitý transportní protokol, IP adresa a port útočníka, SIP metoda.

```
2016-10-03 09:45:22,UDP,94.23.8.146,5071,INVITE
2016-10-03 09:49:08,UDP,89.163.204.25,5076,INVITE
2016-10-03 09:57:57,UDP,158.69.244.172,5121,OPTIONS
2016-10-03 10:04:00,UDP,89.163.204.25,5074,INVITE
```

Obr. 6.11: Ukázka exportu dat z Tcpdump.

Výchozí pcap soubory zůstávají dočasně archivovány na sondě a je možná jejich další transformace, např. do podobné formy jako Homer data.

Homer data

Nejširší možnosti získání dat o útocích představují data získaná z exportů aplikace Homer5. Jak již bylo uvedeno v kapitole o sondách, Homer5 funguje jako prostředník mezi samotným exportem dat ze sondy a Beekeeperem. Pomocí HEP protokolu se přenáší kompletní SIP zpráva, která je zpracována na serveru Kamailio s modulem Homer5.

SIP zpráva je uložena do databáze MySQL, z níž je exportována pomocí skriptu h2b ve formátu JSON do systému Beekeeper. Atributy použité v JSON datech jsou víceméně samovysvětlující a odpovídají polím v SIP zprávě (viz. příloha A.1). Konkrétní příklad je zobrazen na obr. 6.12. Vzhledem k obsáhlosti zprávy jsou delší části kráceny, citlivé hodnoty byly anonymizovány.

```
{"id": "37031", "date": "2016-10-03 06:36:00", "micro_ts": "1475469360638770",
"method": "REGISTER", "reply_reason": "", "ruri": "sip:1.2.3.4:5060",
"ruri_user": "", "ruri_domain": "1.2.3.4", "from_user": "1003",
"from_domain": "1.2.3.4", "from_tag": "b888479605a9bbe04b68...44",
"to_user": "1003", "to_domain": "1.2.3.4", "to_tag": "", "pid_user": "",
"contact_user": "1003", "auth_user": "", "callid": "f9f2...9b","callid_aleg": "",
"via_1": "SIP/2.0/UDP213.202.233.49:37088;branch=z9h...2e9b0",
"via_1_branch": "z9hG4b...b0", "cseq": "1 REGISTER", "content_type": "",
"auth": "", "user_agent": "", "source_ip": "213.202.233.49",
"source_port": "37088", "destination_ip": "1.2.3.4", "destination_port": "5060",
"contact_ip": "213.202.233.49", "contact_port": "37088", "originator_ip": "",
"originator_port": "0", "correlation_id": "f9f265478b7a64dcb2ff9881d2bec19b",
"type": "1", "node": "homer01:101", "windowEndTime": "2016-10-03 06:43:00"}
```


Obr. 6.12: Ukázka exportu JSON dat z Homer5.

6.2.4 Agregovaná data a vstupy pro klasifikaci

Jakmile jsou přijata data z autorizované sondy, probíhá jejich validace, převedení do odpovídající struktury a uložení do databáze. Každá sonda je následně informována o výsledku akce. Z takto připravených dat probíhá agregace do entity *mlp_attribute*.

Uchovávaná data jsou podobná jako v případě exportovaných dat. Jedná se o identifikátory, IP adresu a port útočníka, transportní protokol, počet spojení, počty jednotlivých SIP zpráv (register, invite, ack, bye, cancel, options, subscribe), množství zpráv na spojení, časová známka začátku útoku, délka trvání útoku, klasifikace, informace o lokalitě útočníka (stát, region, město, časová známka geolokace).

```
| 178670 | 8 | 209.126.122.X | 5124 | udp  | 1 | 0 | 0 | 0 | 0 | 0
| 1 | 0 | 1 | 196903 | 2016-10-03 10:45:32 | 0 | opt_test | US
| United States | MO | Missouri | St Louis | 2016-10-03 11:00:24
```

Obr. 6.13: Příklad dat z tabulky mlp_attribute.

Průběh agregace je již popsán v části agregačního modulu (kap. 6.2.1). Z agregovaných zpráv se vytváří vstupy pro klasifikaci, je však nutné odstranit některé nadbytečné atributy, které by na klasifikaci neměly mít vliv nebo by mohly negativně ovlivnit její přesnost.

Pro klasifikaci se používá pouze 10 parametrů, které charakterizují různé typy útoků. Data určená pro klasifikaci (vstupní vektor) zobrazuje tabulka 6.1. Jeden řádek představuje agregovaná data z jednoho zdroje útoku pro konkrétní časové okno.

Tab. 6.1: Příklady vstupních vektorů pro klasifikaci útoků.

Delta	ConnC	RC	IC	AC	BC	CC	OC	SC	ConnRate
125	2	0	0	0	0	0	2	0	1.00
28	192	0	156	0	0	0	0	0	0.81
298	2	110	220	44	40	40	0	0	227.00
165	3	26059	0	177	0	0	0	0	8745.00
21	10	25	0	0	0	0	0	0	2.50

Význam parametrů uvedených v tabulce:

- **Delta** – Délka trvání útoku v sekundách.
- **ConnC** – Počet SIP spojení v agregačním okně.
- **RC** – Počet doručených REGISTER zpráv.
- **IC** – Počet doručených INVITE zpráv.
- **AC** – Počet doručených ACK zpráv.
- **BC** – Počet doručených BYE zpráv.
- **CC** – Počet doručených CANCEL zpráv.
- **OC** – Počet doručených OPTIONS zpráv.
- **SC** – Počet doručených SUBSCRIBE zpráv.
- **ConnRate** – Poměr celkového počtu doručených k počtu SIP spojení

Uvedenou strukturu vstupních parametrů využívají všechny dále zmíněné klasifikátory a tvoří základní kámen pro další práci s daty.

6.3 Klasifikační modul

Po dokončení agregace útoků je možné uložená data v systému Beekeeper vyhodnotit pomocí různých klasifikátorů. Druhy těchto klasifikátorů a jejich množství závisí na konkrétním nastavení specifickém pro každou běžící instanci Beekeeperu (v závislosti na licenčních podmínkách). V základu je možné využívat klasifikaci pomocí neuronových sítí. Dostupné jsou tyto generace neuronové sítě:

- **MLPv1** – původní jednoduchá implementace MLP neuronové sítě
- **ANNv2** – přepsaná implementace MLP, univerzální a široce konfigurovatelná síť

Zmíněné neuronové sítě byly implementovány v jazyce JAVA, jsou dostupné pod licencí GPL 3.0 a jejich použití není omezeno (stejně jako Beekeeper a na něj vázaný software). Beekeeper však neprovádí učení těchto sítí či přípravu datových sad. Odladění datových sad a naučení sítí je nutné provést mimo prostředí Beekeeperu. Naučené neuronové sítě lze exportovat (resp. jejich konfiguraci) do formátu JSON. Systém Beekeeper dokáže nahrát exportované sítě a využívat je ke klasifikaci útoků.

Kromě využití neuronových sítí umožňuje Beekeeper napojení na API aplikace Weka. Díky tomu je možné pracovat s algoritmy připravenými pro tuto aplikaci. Vyjma již dostupných algoritmů disponuje Weka i online repozitářem s dalšími algoritmy. Pro účely výzkumu je tedy k dispozici široká základna algoritmů. Použití těchto algoritmů může být vázáno dalšími podmínkami či licencí, která nemusí umožňovat jejich komerční nebo výzkumné použití.

Z těchto důvodů lze Weka algoritmů využít, ale jejich použití je omezeno účelem nasazení Beekeeperu. Zároveň je nutné nastavit konfiguraci těchto algoritmů stejně jako v případě neuronových sítí.

7 METODIKA PRÁCE A NÁVRH KLASIFIKAČNÍCH MODELŮ

Kapitola navazuje na sekci popisující implementaci systému Beekeeper a jednotlivých sond, aby blíže rozvedla metody využité pro klasifikaci SIP útoků. Obsahuje definice tříd SIP útoků, přípravu a vlastnosti datových sad. Analytické vyhodnocení slouží k pochopení dále prezentovaných výsledků.

7.1 Definice tříd útoků

Veškerá data použitá v této práci jsou získána pomocí uvedené sítě honeypotů a reflektují současný stav útoků používaných v IP telefonii. Ačkoli existuje celá řada různých typů útoků (viz. kapitola 2.2.1), mnoho z nich se v praxi nevyužívá, není zachyceno nebo chybí dostatečný počet vzorků.

Důvodem může být problematická dostupnost či složitost nástrojů pro SIP útoky, podobný efekt útoku či nízká možnost zisku z daného útoku. Obecně platí, že se v rámci SIP útoků setkáme s několika typickými zástupci útoků a další typy se vyskytují velice sporadicky.

Dalším důvodem může být odhalení honeypotu, který lze identifikovat na základě detailních odchylek v chování vůči produkčním ústřednám. Tyto jemné detaily jsou zároveň útočníky využívány i pro přesné určení provozované verze SIP ústředny. Komplexní útoky, složené z několika rozdílných fází útoku, bývají mířené proti konkrétním cílům (firmy, hotely, atd.) a jejich zachycení na navrženém systému detekčních sond je nepravděpodobné.

Díky tomu je preferován postup nasazení sond s tzv. plnohodnotným honeypotem spoléhajícím na nasazení reálné ústředny (openWRT sonda s ústřednou Asterisk). Postupným testováním a dostupností dostatečného počtu zástupců byly stanoveny tyto třídy SIP útoků:

- **Options test** – Testování dostupnosti PBX pomocí OPTIONS zpráv.
- **Options scanning** – Dlouhodobé skenování PBX pomocí OPTIONS zpráv.
- **Call test** – Pokusy o sestavení hovoru přes PBX.
- **Unknown protocol** – Komunikace na monitorovaném portu, nebyla však detekována SIP zpráva dle RFC 3261.
- **Register and call** – Pokus o registraci na PBX a následné hovory přes PBX.
- **Register test** – Testování registrace na PBX.
- **Register flood** – Zahlcení PBX pomocí REGISTER zpráv.
- **Register attempt** – Pokus o registraci legitimním uživatelem.

Uvedené třídy vychází z teoretických poznatků o útocích, dělení používaném v komerčních nástrojích a odborné literatuře. Jejich přesné definice prošly řadou úprav s ohledem na získaná data. Zmíněné třídy jsou relevantní pro všechny výstupy uvedené v této dizertační

práci. Výsledky dosažené s předchozími sadami jsou uvedeny například v [Saf03, Saf06]. S rostoucím množstvím dat o útocích je možné tyto třídy dále upravovat a rozšiřovat dle aktuálního stavu na poli útoků.

7.2 Data o útocích

Pro potřeby dizertační práce byla sestavena sada dat (dále sada 8) o útocích zachycená systém Beekeeper za období necelých 2 let. Dohromady jednotlivé sondy monitorovaly provoz po dobu 1689 dní. Monitoring neprobíhal kontinuálně během zmíněných let, ale docházelo ke změnám nasazení sond v různých sítích.

Nejkratší doba, po kterou byl honeypot nasazen byla 69 dní, nejdelší pak 296 dní (přičemž tato sonda nadále monitoruje provoz). Rozdílný je i počet SIP zpráv získaných pomocí různých metod – sadu 8 tvoří např. 322 506 SIP zpráv z Dionaea a 3 725 942 SIP zpráv získaných pomocí Tcpdump. Změna v nastavení a nasazení sondy je vhodná i pro zachování dostatečné variability a nezávislosti dat.

Tabulka 7.1 uvádí základní informace o složení útoků v sadě 8 (počet absolutních i unikátních signatur) a porovnání s jednou z předchozích verzí.

Tab. 7.1: Vlastnosti unikátních signatur - porovnání sady 8 a sady 6.

Třída útoku	**Sada 8**		**Sada 6**	
	Počet útoků	**Unikátní**	**Počet útoků**	**Unikátní**
Options test	49 334 (26,73%)	29 (0,83%)	20 209 (23,86%)	13 (2,12%)
Options scanning	234 (0,13%)	235 (6,76%)	42 (0,05%)	8 (1,29%)
Call test	123 559 (66,94%)	1 876 (53,95%)	54 938 (64,85%)	116 (18,83%)
Unknown protocol	7 933 (4,30%)	79 (2,27%)	7 801 (9,21%)	83 (13,47%)
Register and call	171 (0,09%)	210 (6,04%)	126 (0,15%)	63 (10,23%)
Register test	1 041 (0,56%)	438 (12,60%)	763 (0,10%)	86 (13,96%)
Register flood	571 (0,31%)	560 (16,11%)	308 (0,36%)	232 (37,66%)
Register attempt	1 745 (0,95%)	50 (1,44%)	527 (0,62%)	63 (15,91%)
Celkem	**184 590**	**3 477**	**84 717**	**1 360**

Z uvedené tabulky je patrná nevyváženost dat, ovlivňující i tvorbu učících, validačních a testovacích sad. Problematické jsou především třídy *Options test*, *Options scanning* a *Unknown protocol*. Není to dáno pouze nízkým počtem unikátních signatur, ale i nízkou variabilitou mezi možnými signaturami.

Ze sady 8 jsem pomocí metod pro oversampling a undersampling (více [31]) vytvořil vyvážené a vzájemně nezávislé sady pro potřeby klasifikátorů. Tyto sady byly navrženy tak, aby primárně využívaly dostupných signatur a oversampling se využíval co nejméně.

Výsledkem jsou sady obsahující celkem 480 signatur, což představuje 14,8% všech unikátních signatur ze sady 8. Obvyklé rozdělení mezi učící a testovací sadu je dle [1, 3, 10] doporučováno v poměru 70:30 nebo 80:20. Vzhledem k přítomnosti validační sady jsem zvolil poměr přibližně 66–16–16. Sady jsou záměrně navrženy tak, aby zůstaly vzájemně nezávislé i po preprocessingu používaného u ANNv2 (viz. kapitola 4.1.1).

- **ls008** – učící sada, obsahuje 320 signatur (60 vzorků pro každou třídu útoku)
- **vs008** – validační sada, 80 signatur (10 vzorků na třídu)
- **ts008** – testovací sada, 80 signatur (10 vzorků na třídu)

Základní statistické údaje o učící sadě zobrazuje následující tabulka 7.2. Vyhodnocení zbývajících sad je uvedeno v příloze D.2.

Tab. 7.2: Vlastnosti atributů učící sady (ls008).

	Delta	**CoC**^{0}	**RC**	**IC**	**AC**	**BC**	**CC**	**OC**	**SC**	**CRate**^{0}
x_{min}	0	1	0	0	0	0	0	0	0	0
x_{max}	43869	556	246720	277	347	10	88	70	4	123361
$\bar{x}$	527,909	12,262	1462,509	7,397	3,637	0,063	1,950	2,075	0,013	663,052
σ	3186,451	42,730	14199,815	27,754	25,037	0,723	7,853	6,261	0,224	7013,182

7.3 Definice pojmů pro vyhodnocení klasifikátorů

Pro vyhodnocení výsledků klasifikace lze využít řady různých parametrů popisujících vlastnosti daného klasifikátoru. Tyto parametry mohou být v literatuře definovány různě, uvádím zde tedy popis a metody výpočtu těchto parametrů použitých pro porovnání klasifikátorů v této práci (vycházím mimo jiné z [23, 31, 40]). Vzhledem k podobnosti českých ekvivalentů a zachování přehlednosti budu nadále využívat především anglické termíny.

Ačkoli níže uvedené parametry slouží pro vyhodnocení binárních klasifikátorů (tj. zda daný vzorek patří/nepatří do jediné třídy), lze je využívat i pro vícetřídní klasifikátory. K evaluaci se používá se základní sada termínů (viz. tab. 7.3). Tyto termíny můžeme vztahovat na klasifikaci obecně, ale i ke konkrétní třídě.

- **True positive (TP)** – třída útoku (C_{real}) byla korektně klasifikována do odpovídající třídy (C_{pred}), například pro detekci třídy c_1: $C_{real} = c_1, C_{pred} = c_1$
- **False positive (FP)** – útok z jiné třídy byl klasifikován jako útok z právě detekované třídy, například pro detekci třídy c_1: $C_{real} = c_5, C_{pred} = c_1$
- **True negative (TN)** – útok byl korektně klasifikován do své skupiny, například pro detekci třídy c_1 : $C_{real} = c_3, C_{pred} = c_3$
- **False negative (FN)** – nesprávná klasifikace útoku do jiné třídy, například pro detekci třídy c_1 : $C_{real} = c_1, C_{pred} = c_5$

V případě použití pro více tříd útoků dochází k překryvům mezi jednotlivými třídami. Např. FP pro jednu třídu automaticky tvoří FN klasifikace na jiné třídě, do níž byl útok klasifikován (a vice versa).

Tab. 7.3: Confusion matrix – Metriky vyhodnocení klasifikace.

		Predikovaná třída		
		Poz. pred.	**Neg. pred.**	
Reálná třída	**Ano**	TP True positive	FN False negative	P_C
	Ne	FP False positive	TN True negative	N_C

Výsledné hodnoty TP, FP, FN, TN je tedy nutné zjistit pro každou třídu a následně interpretovat pro klasifikátor jako celek. Pro všechny třídy útoků C stanovíme i poměry, abychom mohli určit hodnoty chyb pro konkrétní klasifikátor:

$$TP_{rate} = \frac{TP}{P_C} \tag{7.1}$$

$$FP_{rate} = \frac{FP}{N_C} \tag{7.2}$$

$$TN_{rate} = \frac{TN}{N_C} \tag{7.3}$$

$$FN_{rate} = \frac{FN}{P_C} \tag{7.4}$$

Vlastní přesnost (dále *accuracy*, ACC) a chybovost (*Error rate*) daného klasifikátoru lze tedy vyjádřit následovně:

$$ACC = \frac{TP + TN}{P_C + N_C} \tag{7.5}$$

nebo také

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{7.6}$$

$$E_{rate} = 1 - ACC \tag{7.7}$$

Uvedené pojmy poskytují pouze základní informace o výkonu klasifikátoru, proto se zavádí další ukazatele výkonu klasifikace.

Preciznost klasifikace (*precision*, dále také *positive predicted value* či *negative predicted value*) označuje počet správně klasifikovaných jedinců dané třídy vzhledem ke všem jedincům s takto predikovanou třídou. Jinými slovy, kolik jedinců z predikované třídy do této třídy skutečně patří.

$$PRE = \frac{TP}{TP + FP} \tag{7.8}$$

Druhým důležitým faktorem je senzitivita klasifikace (*recall*, ale také *sensitivity*, *hit rate*, ...). Značí, kolik jedinců z dané třídy útoku bylo korektně klasifikováno.

$$REC = \frac{TP}{TP + FN} \tag{7.9}$$

Je patrné, že parametry *precision* a *recall* jsou navzájem v inverzním vztahu, podobně jako *accuracy* a *error rate*. V případě klasifikátoru s více třídami dochází k překryvům těchto hodnot mezi jednotlivými třídami. Zároveň je důležité si uvést, že parametr *precision* je závislý na distribučním rozložení klasifikované třídy útoku, u parametru *recall* tomu tak není. Každý z těchto parametrů má svá omezení, ale obecně jsou efektivními ukazateli výkonu klasifikace.

Parametry PRE a REC slouží pro detailnější pochopení chyb způsobených klasifikátorem. Pro porovnání se v odborné literatuře využívají i následující parametry označované *F-measure* (7.10) a *G-mean* (7.12).

$$F_m = \frac{(1 + \beta^2) \times REC \times PRE}{\beta^2 \times REC + PRE} \tag{7.10}$$

Atribut β slouží pro nastavení důležitosti parametru REC (obvykle $\beta = 1$). Hodnota *F–measure* kombinuje parametry *precision* a *recall* do jediné hodnoty, vyjadřující množství efektivity klasifikace vzhledem k vážené důležitosti těchto parametrů. Poskytuje tak vhodnější hodnotu pro efektivitu klasifikátoru než jeho přesnost (*accuracy*).

Specifičnost (specificity) definuje podíl správně identifikovaných vzorků jiných tříd ke všem útokům v těchto třídách a využívá se pro výpočet parametru *G–mean*, který hodnotí stupeň odvozené systematické chyby pro poměry pozitivních a negativních predikcí (rovnice 7.12).

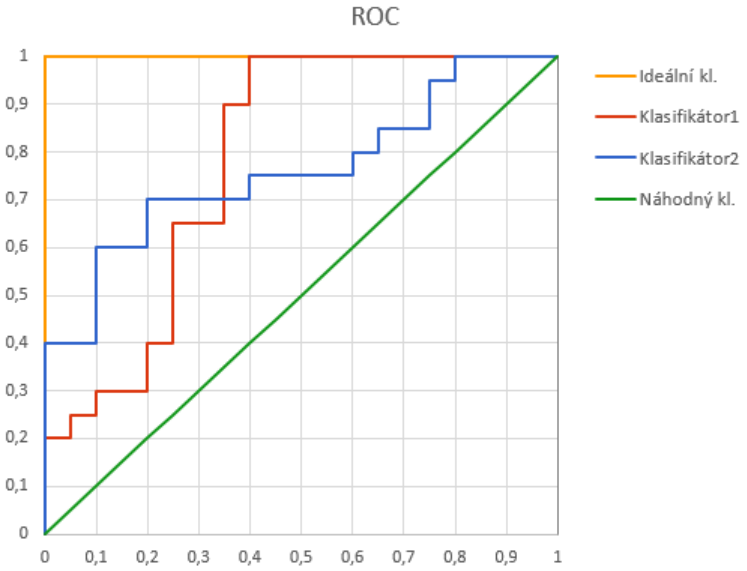
$$SPE = \frac{TN}{TN + FP} \tag{7.11}$$

$$G_m = \sqrt{PRE \times SPE} \tag{7.12}$$

Uvedené sady parametrů (hlavně F_m, G_m) značně vylepšují informace o klasifikátoru ve srovnání s *accuracy*. Pro srovnání výkonu klasifikátorů se často využívá i ROC křivek.

7.3.1 Receiver Operating Characteristics Curves

ROC křivky zobrazují formou grafu vztah mezi TP_{rate} a FP_{rate} (rovnice 7.1, 7.2). Poskytují tak vizuální reprezentaci relativního vztahu mezi páry benefitů (TP) a cen (FP) binárního klasifikátoru. Příklady ROC křivek ilustruje graf 7.1.



Obr. 7.1: ROC – příklady křivek.

Čím více se body či křivka blíží bodu $0; 1$ (levý horní roh) v ROC grafu, tím více se daný klasifikátor blíží ideálnímu klasifikátoru (značen oranžově v grafu 7.1). Každý hard–decision klasifikátor je schopný generovat body v ROC prostoru. Soft–decision klasifikátory mohou generovat křivky v ROC prostoru. Platí, že každý hard-decision klasifikátor lze převést na soft–decision klasifikátor.

Pro číselné vyjádření efektivity klasifikátoru se využívá obsahu pod křivkou (AUC. Vzhledem k tvaru křivky může klasifikátor s nižší AUC dosahovat lepších výsledků klasifikace v části ROC prostoru než klasifikátor s vyšší AUC, závisí ale na tvaru těchto křivek (viz klasifikátor1 a klasifikátor2 na obr. 7.1).

Nevýhodou ROC křivek je jejich silná vazba na parametr FP a tedy závislost na distribučním rozložení sady pro hodnocení. Z těchto důvodů vznikly další podobné grafy, jako Precision-Recall curves, DET curves či Cost curves.

ROC pro vícetřídní klasifikátory

ROC křivky lze využít i pro vícetřídní klasifikátory [47, 41, 51], přičemž výsledkem je nárůst dimenzionality zobrazovaného ROC prostoru. Namísto zobrazení vztahu mezi párem $TP - FP$ je nutné vytvořit matici všech benefitů (pro c tříd) a jejich cen $(c^2 - c)$. ROC graf tedy zobrazuje plochu (respektive hyperplochu) v závislosti na počtu klasifikovaných tříd. Případně lze vytvořit separátní ROC grafy s křivkami pro jednotlivé třídy.

Podobně jako lze u ROC křivky vyjádřit hodnotu AUC, lze pro hyperplochu nalézt tzv. VUC, která vyjadřuje objem pod hyperplochou. S rostoucím počtem tříd se zvyšuje i náročnost vyhodnocení. Vysoká dimenzionalita řešení komplikuje interpretaci výsledků,

protože se při vyšším počtu klasifikovaných tříd hodnoty VUC blíží výsledkům náhodné klasifikace.

Dle Landgrebe [39] můžeme výše popsaný postup použít pro nízké počty tříd ($|C| \in < 3, 6 >$). Existují i metody pro zpracování vyššího počtu tříd, nicméně v rámci této dizertační práce se pro porovnání klasifikátorů řídím hodnotami parametrů F_m, G_m, apod.

7.4 Clustering vstupních parametrů

Vstupní parametry pro klasifikátory odpovídají popisu uvedeném v tabulce 6.1 a byly testovány řadou algoritmů k nalezení shluků (clustering), tedy skupin podobných vstupních vektorů. Testování probíhalo jak v prostředí MatLab, tak i nástroje Weka. Vstupem pro tyto algoritmy byla testovací sada bez uvedené klasifikace útoků, příp. počet hledaných shluků a další nezbytné parametry pro konkrétní algoritmy.

Tab. 7.4: Porovnání clustering algoritmů.

Algoritmus	**Počet skupin**	**Error rate**
Canopy	3[1]	86,56%
EM	4[1]	68,75%
HierarchicalClusterer	10[1]	85,63%
FarthestFirst	54-62[2]	85,63%
SimpleKMeans - MD	8[2]	50,63%
SimpleKMeans - ED	11-12[2]	60,93%
DensityBased (EM)	4[1]	69,38%
DensityBased (SimpleKMeans - MD)	8[2]	58,13%
DensityBased (SimpleKMeans - ED)	8[2]	66,25%

Potvrdila se hypotéza, že dané algoritmy nejsou schopny autonomně rozeznávat ve vstupních datech definované skupiny útoků. Chybovost 87,5% odpovídá jediné korektně identifikované skupině útoků. V praxi to znamená, že detekované skupiny obsahují jednoho až čtyři jedince a jedna skupina všechny zbývající. Obecně tyto algoritmy odpovídají přesností náhodnému klasifikátoru.

Část algoritmů (označena v tabulce 7.4 indexem 2) vyžaduje pro svůj běh definici počtu hledaných skupin. Chybovost těchto algoritmů je již nižší proti předchozí skupině, stále však nedosahují kvalit vhodných pro nasazení.

Nejlepší přesnosti dosáhl algoritmus KMeans využívající pro nalezení shluků Manhattan distance. I v tomto případě však chybovost dosahuje 50,63%. Velice dobrých výsledků dosahuje především na třídách útoků register attempt a register flood. Vysoká úspěšnost

[0] Kráceno: CoC – ConnC, CRate – ConnRate

[1] Autonomně detekovaný počet skupin

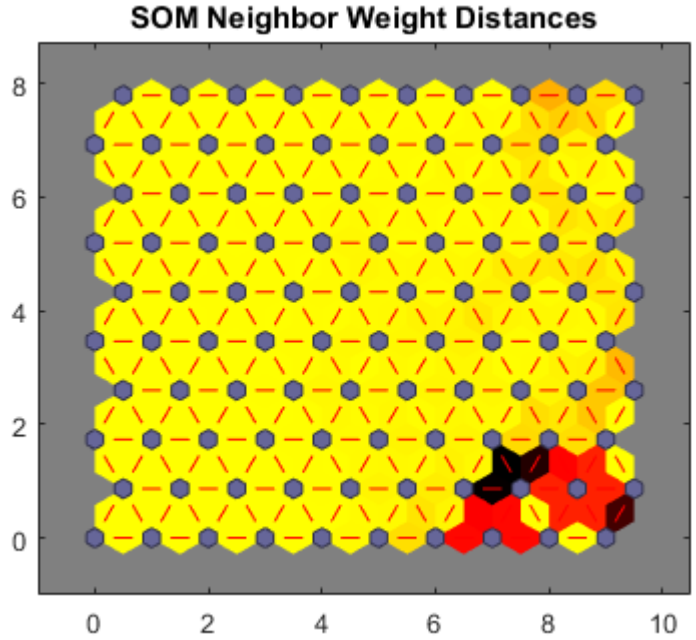
[2] Definovaný počet skupin, pro který dosáhl daný algoritmus nejnižší chybovosti

je ale vykoupena vysokým množství false positive klasifikací (31,79% pro RA a 14,29% pro RF útoky). Výsledná matice záměn (Confusion matrix) zobrazuje tabulka 7.5.

Tab. 7.5: Matice záměn pro SimpleKMeans – MD clustering.

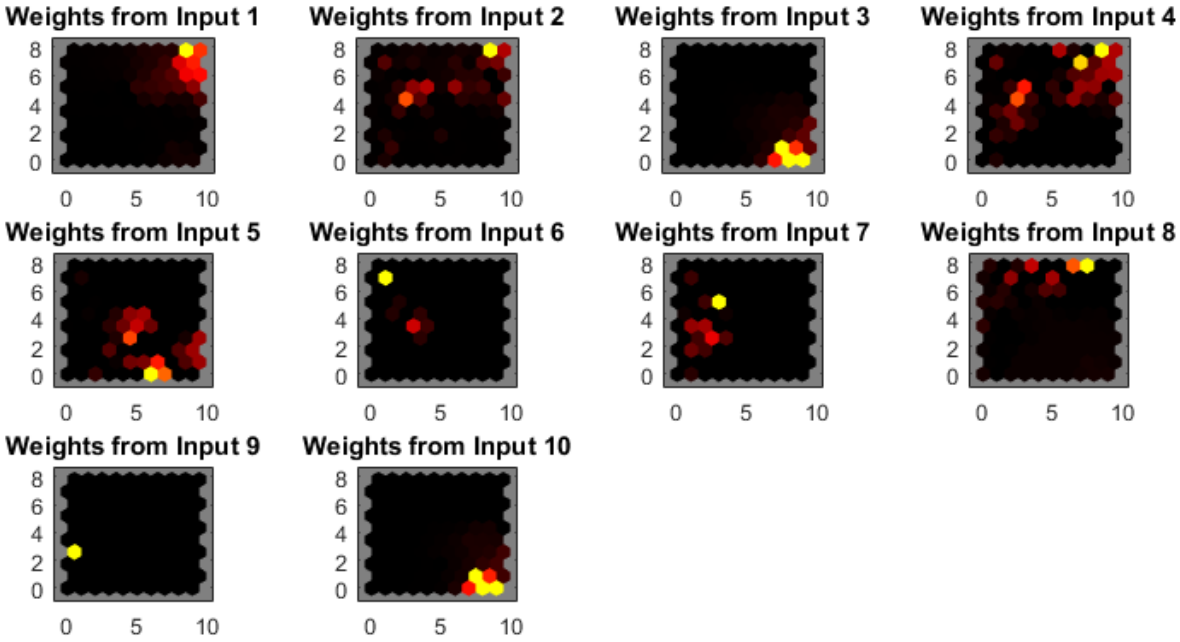
Predikovaná třída útoku									
RT	**OT**	**RA**	**CT**	**X**	**RC**	**OS**	**RF**		
0	**20**	0	0	0	0	0	20	**OT**	**Reálná třída**
0	18	0	0	0	0	**10**	12	**OS**	
0	0	26	**10**	4	0	0	0	**CT**	
4	0	36	0	0	0	0	0	**UKW**	
0	0	5	4	1	**30**	0	0	**RC**	
10	0	22	0	0	0	0	8	**RT**	
2	0	0	0	0	0	0	**38**	**RF**	
0	0	**40**	0	0	0	0	0	**RA**	

Mezi varianty neuronových sítí pro clustering dat patří samoorganizační mapy (SOM). I zde se potvrdila hypotéza o problematickém shlukování. Výsledné váhy mezi sousedními neurony ilustruje obr. 7.2. Žlutá barva značí malou vzdálenost mezi neurony (tj. výstupy aktivované podobnými vstupy), tmavší odstíny představují vzdálenosti větší. Naprostá většina vstupního vektoru se nachází ve žlutém poli, pouze několik vstupů se nachází v pravé dolní oblasti grafu, kde jsou vzdálenosti mezi výstupními neurony větší. Opět se tedy dostáváme na úroveň majoritního klasifikátoru s úspěšností 12,5%.



Obr. 7.2: SOM – váhy mezi výstupními neurony (ls008).

SOM umožňuje také analýzu vlivu jednotlivých vstupů na výsledné aktivace. Je tedy možné zobrazit redundantní vstupy či vstupy s podobným vlivem. Podobně jako v případě dřívější učící sady (založené na sadě 6, ls006), zobrazuje graf pro analýzu vstupního vektoru velice podobný vliv parametrů 3 a 10 (tj. počet REGISTER zpráv a výsledný poměr přijatých zpráv). Způsobeno to bude pravděpodobně silnou závislostí těchto parametrů v rámci poloviny útočných tříd.



Obr. 7.3: SOM – analýza vstupního vektoru pro učící sadu ls008.

Nepotvrdila se závislost parametrů 6,7 a 9 (tj. počet zpráv BYE, CANCEL, SUBSCRIBE), která byla patrná na předchozí verzi učící sady ls006 (závislosti vstupních vektorů zobrazeny v příloze, obr. D.1). Tento stav může být způsoben rozdílnými zástupci v učících sadách, složením a variabilitou hodnot vstupních parametrů. Výsledky vlivu jednotlivých parametrů zobrazuje obr. 7.3, podrobné výsledky analýzy sady ls006 uvádím v samostatném článku [Saf02].

8 EXPERIMENTÁLNÍ OVĚŘENÍ

Úspěšnost klasifikace pomocí metod machine learningu silně ovlivňuje volba parametrů nutných k běhu každého algoritmu. Pro většinu modelů jsou stanoveny výchozí hodnoty jednotlivých parametrů, ty je však nutné ověřit empiricky a upravit pro používaná vstupní data.

Veškerých výsledků v této kapitole bylo dosaženo pomocí datových sad verze 8. Pro upřesnění uvádím, že učící sada slouží k natrénování daného algoritmu, validační sada se využívá pro vyhodnocení výkonu modelu a nastavení potřebných parametrů, testovací sada slouží k závěrečné evaluaci vítězného klasifikátoru.

8.1 Optimální parametry pro ANNv2

Jak již uvádím v implementační části MLP (kap. 6.3), neuronová síť ANNv2 je vůči MLPv1 univerzálnější a disponuje rozsáhlejší sadou modifikovatelných parametrů. Tato část popisuje nastavení hodnot jednotlivých parametrů. Z hlediska ANNv2 se jedná o stanovení vhodné struktury, velikosti sady pro postupné učení (MBS), učící koeficient η, momentum μ, vlivu L2 regularizace λ, typu chybové funkce, preprocessingu vstupních dat, inicializace neuronové sítě, maximálního počtu učících epoch (kompletních průchodů učící sadou) a omezení minimální chyby klasifikace.

Tyto parametry se vzájemně ovlivňují a změna jednoho z nich může způsobit nevyhnutelné úpravy ve zbývajících parametrech. Zároveň může existovat i více optimálních řešení s různými kombinacemi parametrů. Porovnání efektivity změny každého parametru komplikuje i náhodné stanovení vnitřních vah mezi neurony a biasů neuronů.

Výhodou náhodného stanovení těchto hodnot pro každý běh učícího algoritmu je odolnost proti případnému opakování systematických chyb vzniklých během inicializace neuronové sítě. Cílem této části je nalezení optimálních parametrů pro naučení neuronové sítě, které se blíží ideálnímu klasifikátoru, a zároveň dosahujících nízké náročnosti na systémové prostředky.

Učení neuronové sítě závisí na uvedených parametrech, některé z nich se ale učení nemění a lze je tedy považovat za konstanty. Těmito parametry jsou: datové sady verze 8 (učící, validační a testovací), omezení maximálního počtu epoch a minimální chyby klasifikace (není-li to explicitně uvedeno jinak).

- Učící, validační a testovací sada – odpovídají popisu uvedeném v kap. 7.2 a jsou vždy použity pro všechny dále uváděné výsledky.
- Omezení počtu epoch – maximum je 250 epoch.
- Omezení minimální chyby klasifikace – chybovost musí být menší než 2%
- Preprocessing – navýšení počtu vstupů.

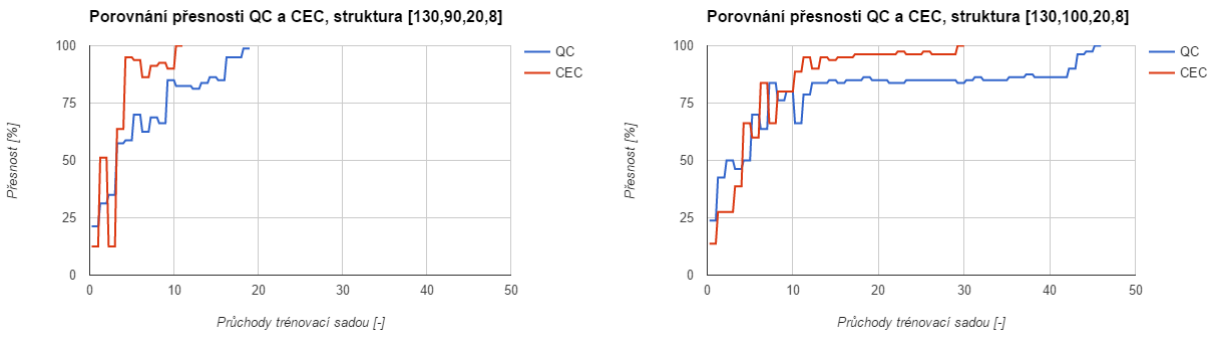
Prvotní nastavení jednotlivých parametrů vychází ze zkušeností s předchozí verzí datové sady. Počáteční experimenty s učící sadou prokázaly vysoké hodnoty úspěšnosti klasifikace, které dosahovaly až 100%. S ohledem na vyšší náročnost učení na takto vysoké hodnoty přesnosti byla z praktických důvodů vyžadována chybovost menší než 2%, která znatelně zvýšila počet úspěšně naučených klasifikátorů a usnadnila nalezení optimálních parametrů. Závěrečné učení modelu probíhalo s podmínkou nulové chybovosti.

8.1.1 Chybová funkce

Nielsen [10] uvádí klíčovou výhodu chybové funkce Cross–entropy, jíž je její rychlá reakce na saturované výstupní neurony. Bylo však nezbytné ověřit, zda je tento jev patrný a přínos CEC dostatečně efektivní i s dostupnou učící sadou.

- **QC** – značí použití Quadrature cost
- **CEC** – značí použití Cross–entropy cost

Následující grafy zobrazují rozdíly mezi použitou chybovou funkcí:



Obr. 8.1: Vliv chybové funkce na přesnost klasifikace.

Na obou grafech vidíme nárůst přesnosti v průběhu učení neuronové sítě (jednotlivé epochy backpropagace). Zobrazeny jsou výsledky jak pro nejvhodnější nalezenou strukturu s chybovou funkcí QC – [130, 100, 20, 8], tak i pro nejvhodnější strukturu pro CEC – [130, 90, 20, 8]. Každý z grafů zobrazuje vždy nejlepší dosažený průběh učení. Ostatní parametry se blíží optimálnímu nastavení s danou chybovou funkcí.

Tab. 8.1: Vliv chybové funkce na učení neuronové sítě.

Struktura	PPE	NPE	Chybová funkce
[130,90,20,8]	31,333	11	*Cross–entropy*[1]
[130,90,20,8]	46,033	19	Quadrature
[130,100,20,8]	57,367	30	Cross–entropy
[130,100,20,8]	53,000	46	*Quadrature*[1]

[1]Nejlepší dosažený PPE pro danou chybovou funkci

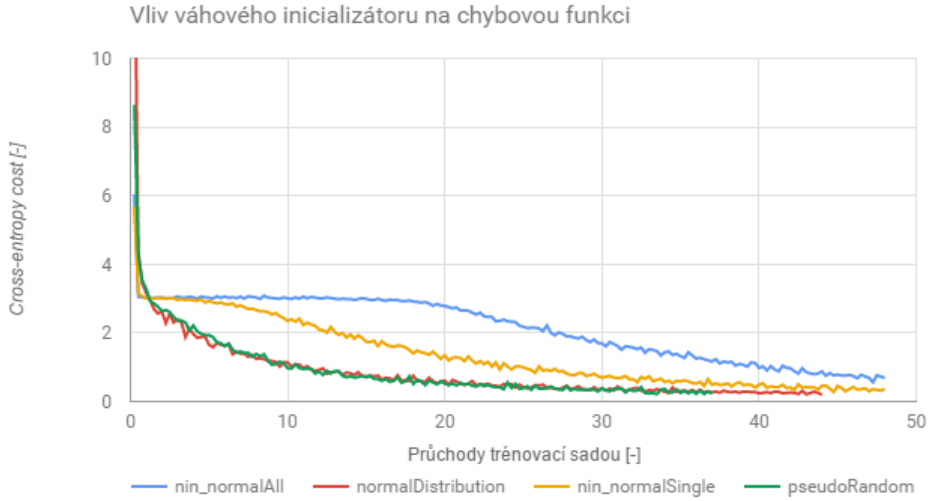
Vyhodnocení vychází z 30ti iterací testování 10 nejvhodnějších struktur (5 nejlepších pro každou chybovou funkci). Kompletní výsledky jsou obsaženy v příloze D.3, vybrané hodnoty prezentuje tabulka 8.1. V obou případech dochází k rychlejšímu učení s chybovou funkcí CEC (viz. nejnižší počet epoch – NPE), ačkoli chybová funkce QC dosahuje nižšího průměrného počtu epoch (PPE) nutných k naučení na struktuře [130,100,20,8]. Důvodem je již zmíněné zobrazení jen nejlepšího průběhu učení pro každou chybovou funkci.

Provedené experimenty potvrdily vlastnosti chybové funkce CEC a bude používána pro následující nastavení zbývajících parametrů (pokud to není uvedeno jinak).

8.1.2 Metody inicializace vah neuronové sítě

Počáteční úspěšnost a rychlost učení závisí na metodě inicializace vah. Správná volba metody pro inicializaci značně ovlivňuje rychlost učení. Preferovanou metodou inicializace vah je dle [10] použití náhodných hodnot z normálního rozložení se směrodatnou odchylkou závislou na počtu vstupních vah do neuronů. Testovány byly metody náhodné inicializace generující tato rozdělení dat:

- **Normal** – normální rozdělení: průměr $\bar{x} = 0$, směrodatná odchylka $\sigma = 1$
- **Nin_normalAll** – normální rozdělení: průměr $\bar{x} = 0$, směrodatná odchylka $\sigma = 1/\sqrt{130}$
- **Nin_normalSingle** – normální rozdělení: průměr $\bar{x} = 0$, směrodatná odchylka $\sigma = 1/\sqrt{n_{in}}$
- **PseudoRandom** – pseudonáhodně generované hodnoty uniformě rozložené na intervalu <0,1): průměr $\bar{x} \approx 0$, směrodatná odchylka $\sigma \approx 1$



Obr. 8.2: Vliv inicializace vah na chybovou funkci CEC.

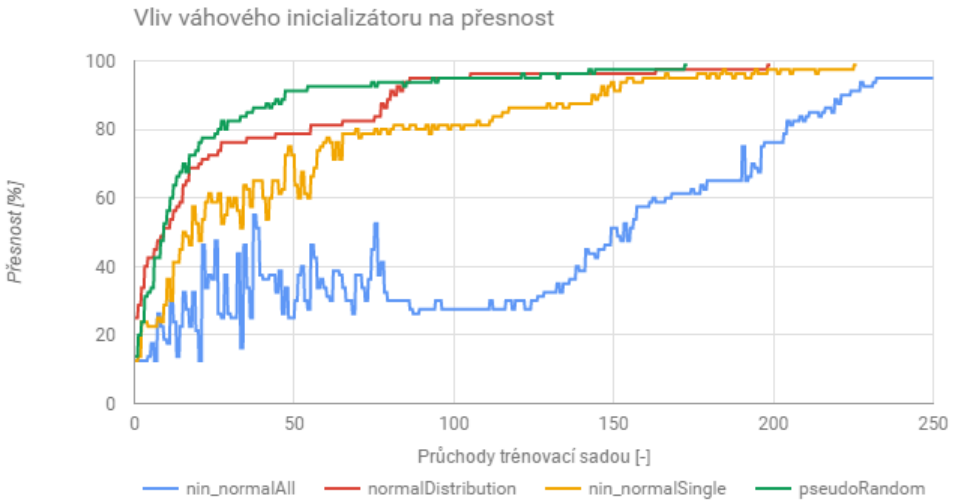
Pseudonáhodně generované hodnoty se blíží normálnímu rozložení, nicméně dosahují větší odchylky od ideálních hodnot, než je tomu při použití generátoru náhodných čísel

z normálního rozložení. Průběh chybové funkce v závislosti na použitém inicializátoru je nezávislý na použité chybové funkci, výsledný vliv na CEC ilustruje graf 8.2.

Z grafu je viditelné problematické učení v počátečních epochách pro inicializátory založené na počtu vah vstupujících do neuronů (Nin_normalSingle a Nin_normalAll). Zpomalení je pravděpodobně způsobené silnou saturací vnitřních vah mezi neurony. Propagování chyby ve vícevrstvých sítích je pomalejší a náprava špatně inicializovaných dat je náročnější.

Nežádoucí saturace vah neuronové sítě je dobře patrná pro inicializátor Nin_normalAll. Inicializátor Nin_normalSingle využívá podobného principu, nicméně se směrodatná odchylka generovaných dat blíží 1 rychleji, a proto není saturace vah tak výrazná. Díky snižující mu se počtu neuronů směrem k výstupní vrstvě dochází k rychlejšímu propagování chyby hlouběji do sítě a tím i upravení chybných hodnot vah. Postupně ale chyba neuronové sítě konverguje k nule bez ohledu na použitou metodu inicializace.

Průběhy chybovosti pro inicializátory Normal a PseudoRandom se svým průběhem blíží exponenciální křivce a rychle reagují na chyby sítě v případě použití QC i CEC.



Obr. 8.3: Vliv inicializace vah na přesnost neuronové sítě (QC).

Zpomalení učení ilustruje i vývoj přesnosti klasifikace v průběhu učení sítě. Graf 8.3 zobrazuje průběh pro neuronovou síť s chybovou funkcí QC, která navíc nepoužívá zcela optimalizované parametry učení (ještě více je zpomalen průběh učení). Díky tomu lze pozorovat rozdílné chování přesnosti sítě pro problematické inicializátory. V epochách, během nichž nedochází ke snižování chybovosti sítě, přesnost sítě osciluje. Jakmile začne docházet ke snižování chyby učení, oscilace ustanou a přesnost sítě se zvyšuje. Průběh vývoje chyby v neuronové síti s chybovou funkcí QC je uveden v příloze D.2, vývoj přesnosti pro chybovou funkci CEC v D.3.

Rozdíl mezi pseudonáhodným a normálním inicializátorem není výrazný a projeví se až při opakovaném experimentování s učením neuronových sítí. Opět bylo testováno 30

iterací 10 různých struktur a výsledky shrnuje tabulka 8.1. Kompletní výsledky uvádím v příloze D.4.

Tab. 8.2: Vliv inicializace vah na učení neuronové sítě.

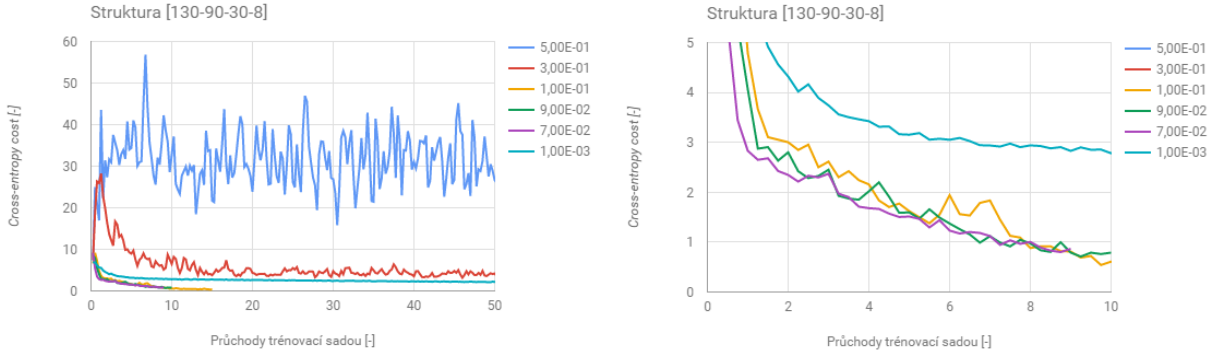
Struktura	PPE	Inicializace vah	Chybová funkce
[130, 90, 20, 8]	93,567	Normální	Cross–entropy
[130, 100, 20, 8]	106,633		
[130, 90, 20, 8]	41,300	Pseudonáhodné	
[130, 100, 20, 8]	31,333		
[130, 90, 20, 8]	134,467	Normální	Quadrature
[130, 100, 20, 8]	99,867		
[130, 90, 20, 8]	46,033	Pseudonáhodné	
[130, 100, 20, 8]	53,000		

Nejlepších výsledků na učící sadě ls008 dosáhneme použitím pseudonáhodné inicializace vah. Průměrný počet epoch při použití normálního inicializátoru je přibližně dvojnásobný než při použití pseudonáhodného přístupu. Ještě více se projeví vliv chybové funkce, která opět přibližně dvojnásobně snižuje počet PPE při využití CEC.

I když lze totožnými metodami inicializovat i hodnoty bias, provádím během inicializace jejich nastavení na 0. Všechny biasy v neuronové síti se nastaví pouze za pomoci učícího algoritmu.

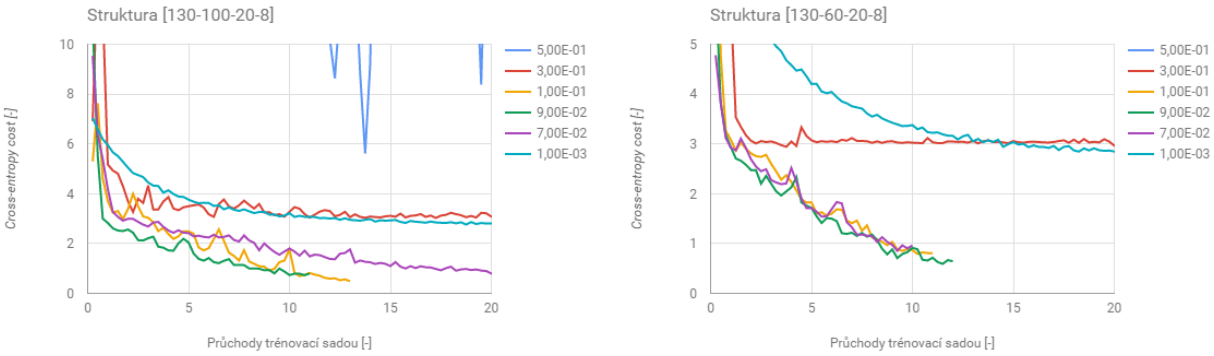
8.1.3 Koeficient učení neuronové sítě

Učící koeficient patří mezi nejdůležitější parametr pro učení neuronové sítě. Jeho hodnota je závislá na druhu použité chybové funkce a musí být, podobně jako v předchozích případech, odladěna pro konkrétní datovou sadu. Pro nalezení vhodného koeficientu učení byly použity 3 struktury (jedna neutrální a po jedné nejvhodnější pro každou chybovou funkci).



Obr. 8.4: Vliv η na chybu CEC funkce – struktura [130, 90, 30, 8].

Obrázek 8.4 zobrazuje chybu neuronové sítě při použití CEC v průběhu učení se strukturou [130, 90, 30, 8]. Levý graf znázorňuje průběhy ve větším měřítku, pravý se v menším měřítku zaměřuje na zobrazení tří nejlepších průběhů stejného grafu. Pro tuto strukturu se křivky chyby velice blíží a nelze jednoduše stanovit, která je nejvhodnější.

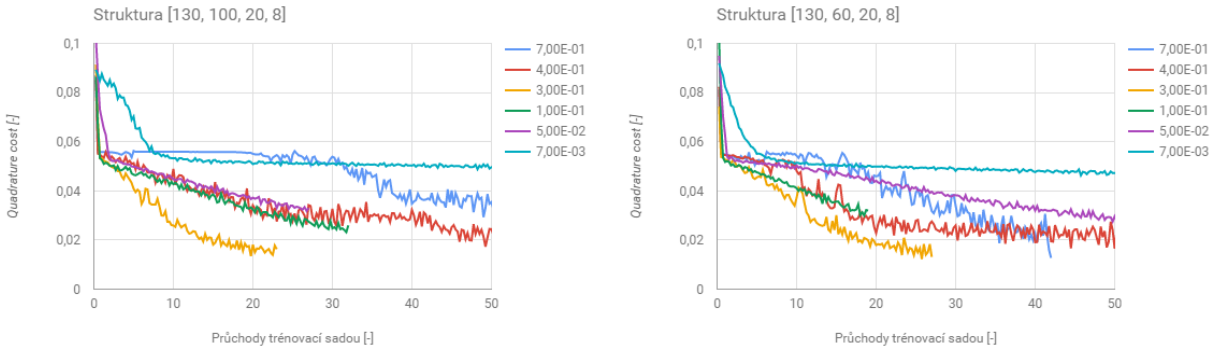


Obr. 8.5: Vliv η na chybu CEC funkce.

V grafech 8.5 lze pozorovat dominanci koeficientu učení $0,09$. Velice podobných výsledků můžeme dosáhnout i s hodnotou $0,1$ učícího koeficientu. Ideální hodnota koeficientu se nachází na intervalu $< 0,09; 0,1 >$, ovlivňuje ji ale i volba zbývajících parametrů. Výsledky budou závislé na konkrétních konfiguracích a náhodné inicializaci sítě na počátku učení.

Z experimentů vychází, že pro 68 nejlepších výsledků s různými konfiguracemi dosahuje koeficient $0,1$ lepších výsledků v 34 případech, koeficient $0,09$ v 33 případech (v 1 případě byly výsledky shodné). Ačkoli jsou rozdíly zanedbatelné, nižší koeficient učení dosahuje lepších výsledků s jednodušími strukturami.

Pro kvadratickou chybovou funkci QC byl rozdíl mezi koeficienty výraznější a lze tak stanovit hodnotu $\eta = 0,3$ jako nejvhodnější. Průběhy chyby dle funkce QC během učení zobrazují grafy na obr. 8.6.



Obr. 8.6: Vliv η na chybu QC funkce.

8.1.4 Momentum učení neuronové sítě

Parametr momentum μ řídí důležitost změn vah z předchozího kroku učení. Díky vysoké úspěšnosti modelu lze porovnávat dopad změn tohoto koeficientu na rychlost učení neuronové sítě. Hodnota tohoto parametru je závislá především na použité hodnotě učícího faktoru.

Tabulka 8.3 shrnuje výsledky experimentování s hodnotami v rozmezí $< 0; 1 >$ pro neuronovou síť s chybovou funkcí CEC. Každé snížení hodnoty způsobilo zvýšení počtu epoch nutných k naučení neuronové sítě a nejlepších hodnot bylo dosaženo s hodnotou $\mu = 1,0$.

Tab. 8.3: Statistické hodnoty pro různé μ.

μ	PPE_μ	σ
1,0	38,833	11,896
0,9	149,370	20,574
0,8	160,617	25,416

Kompletní výsledky 900 experimentů shrnuje tabulka D.5 v příloze této práce.

8.1.5 Velikost fragmentu učící sady

Učení neuronové sítě využívá metodu SGD. Pro stochastické určování gradientu je potřeba určit velikost fragmentů učící sady, na kterých bude následně probíhat učení. Velikost fragmentu je závislé na hodnotě učícího koeficientu a momentu. Pokud fragment pokryje celou učící sadu, jedná se technicky o metodu Gradient descent.

Vhodné nastavení parametru bylo experimentálně hledáno pro chybovou funkci CEC během 30ti iteracích učení pro sadu 10 struktur. Kompletní výsledky obsahuje tabulka D.6 uvedená v příloze, výsledky shrnuje tabulka 8.4.

Tab. 8.4: Výsledný PPE pro hodnoty μ.

MBS	**Sm. odchylka PPE**	**Průměr PPE**
320	0,000	250,000
160	3,300	248,847
80	8,537	39,990
64	11,896	38,833
40	8,683	56,233
32	8,582	65,407
20	18,136	101,717

Nejrychleji probíhá učení pro fragmenty o velikosti 64 a 80 prvků, které tedy obsahují $\frac{1}{5}$ či $\frac{1}{4}$ prvků učící sady. Nejnižší PPE byl dosažen pro $MBS = 64$ jedinců.

8.1.6 Nastavení L2 regularizace

Posledním parametrem ovlivňujícím učení neuronové sítě je λ určující velikost L2 regularizace při úpravě vah v síti. Použití regularizace bylo experimentálně otestováno, výsledky však potvrdily, že ji není nutné využít. Jejím účelem je potlačení overfittingu, ke kterému ale díky rychlému učení a použití metody early–stoping nedochází v takové míře, aby bylo nutné tuto chybu potlačovat L2 regularizací.

Maximální efektivity dosáhla L2 regularizace při hodnotě $\lambda = 0,001$. Výsledky experimentů shrnuje tabulka D.7.

8.1.7 Struktura neuronové sítě

Optimální struktura neuronové sítě musí dosahovat vysoké přesnosti, rychlého učení a počet neuronů ve vrstvách by se měl postupně snižovat směrem k výstupní vrstvě [55]. Ačkoli vyhodnocení dat na naučené neuronové síti není tak výpočetně náročné jako učení, preferuji ty struktury, které dosahují lepších nebo podobných výsledků s menším počtem neuronů.

Nejedná se jen o praktické využití Occamovy břitvy, ale o důraz na schopnost generalizace neuronové sítě. Předchozí výzkum prokázal schopnost sítě zvyšovat svoji přesnost, konvergující k maximu pro danou učící sadu při rostoucím počtu neuronů skrytých vrstev (mimo jiné například [Saf06]). Nejlepších výsledků na testovací sadě dosáhly tyto struktury:

- [130-88-8]
- [130-108-8]
- [130-60-50-8]
- [130-80-20-8]
- [130-80-50-8]
- [130-90-50-8]
- [130-100-40-8]
- [130-100-50-8]
- [130-100-60-8]
- [130-110-30-8]
- [130-110-60-8]
- [130-120-20-8]
- [130-120-50-8]
- [130-120-60-8]

8.2 Optimální parametry pro MLPv1

První generace neuronové sítě MLP využívá pouze parametrů η a μ. Možnosti ovlivnění učení se tak hodně limitují. Zároveň není možné během učení sledovat další parametry, jako například vývoj chyby v průběhu učení. Účel této verze spočíval v ověření funkčnosti a pokročilá funkcionalita a univerzální přístup byl implementován až v následující generaci ANNv2, jejíž nastavení bylo popsáno dříve.

I přesto, že možnosti MLPv1 jsou ve srovnání s ANN omezené, stále dosahuje dostatečně přesné klasifikace a lze ji v praxi využívat. Dále se věnuji nastavení dostupných parametrů.

8.2.1 Momentum

Neuronová síť MLPv1 umožňuje nastavení momentu učení. To se ale uplatňuje až při učení pomocí MBS o velikosti větší než 1 a v případě MLPv1 probíhá plně stochastické učení s velikostí $MBS = 1$. Díky tomu nedochází ke korektnímu zaznamenání vlivu změn vah z předchozího kroku učení. Výsledky shrnuje tabulka 8.5, obsáhlejší výsledky uvádím v příloze D.8.

Tab. 8.5: MLP - Přesnost v závislosti na momentu.

μ	**Max ACC**	**Průměr ACC**	**Sm. odchylka ACC**
1,0	82,00%	75,00%	0,0305
0,8	80,00%	74,61%	0,0237
0,7	80,00%	75,27%	0,0278
0,5	83,00%	75,49%	0,0291
0,0	81,00%	77,07%	0,0190

Výsledky učení potvrzují domněnku o neuplatnění parametru momentum při učení. Dále tedy jeho vliv na učení zanedbám.

8.2.2 Učící koeficient

Vzhledem k omezení sítě lze učení ovlivnit pouze za pomoci učícího koeficientu η. Neuronová síť MLPv1 interně využívá podobné parametry jako ANNv2, ovšem bez možnosti jejich ovlivnění. Shodné je například použití kvadratické chybové funkce. Díky tomu lze částečně vycházet z výsledků pro ANNv2.

Nejlepší nalezená hodnota η ale odhaluje rozdílnost implementací MLPv1 a ANNv2. První generace neuronové sítě MLP dosahuje nejlepších výsledků s $\eta_{MLPv1} = 0,01$, druhá generace s chybovou funkcí QC při $\eta_{ANNv2} = 0,3$. Pokud porovnáme nejvhodnější struktury, dochází již ke shodě a mezi nejlepšími pěti strukturami nalezneme pro oba algoritmy následující konfigurace:

- [130,80,40,8]
- [130,90,20,8]
- [130,110,60,8]
- [130,110,70,8]
- [130,110,100,8]

Ověřená přesnost na validační sadě dosáhla v případě MLPv1 hodnoty 85%. Tohoto výsledku však bylo dosaženo se sadou bez předzpracování dat, které MLPv1 neobsahuje. Závislost na tomto preprocessingu bude vysvětlena dále v kapitole 8.4.1.

Tab. 8.6: Závislost přesnosti MLP na učícím koeficientu.

η	**Průměr ACC**	σ_{ACC}
0,700	64,25%	0,0699
0,500	69,63%	0,0433
0,400	71,43%	0,0424
0,300	73,11%	0,0368
0,200	75,09%	0,0361
0,100	77,76%	0,0301
0,010	84,95%	0,0406
0,001	73,57%	0,1359

8.3 Nastavení algoritmů v aplikaci Weka

Nástroj Weka je souborem algoritmů strojového učení pro data mining. K výsledkům klasifikací se přistupuje jednotným způsobem a zobrazeny jsou relevantní data jako dosažená přesnost, počet FP a FN, hodnoty F-measure, MCC, atd. Nevýhodou jednotného výstupu je ztráta dat o učení algoritmu. Různé algoritmy mohou využívat odlišných metod a tak nemusí být sledování jednotných parametrů možné či relevantní a není tedy možné replikovat odpovídajícím způsobem experimenty provedené s neuronovými sítěmi.

Konkrétní odladění klasifikátoru závisí také na znalosti jeho přesné funkcionality. Každý z algoritmů obsahuje přednastavené výchozí hodnoty parametrů učení vhodné pro obecnou klasifikaci. V práci vycházím z těchto hodnot a specifické nastavení uvádím pro každý algoritmus zvlášť. Vybrána byla sada nejvhodnějších algoritmů, dle dosažené přesnosti.

Z hlediska algoritmů je nutné striktně oddělovat metody založené na kolektivním vědomí, fúze výsledků, boostingu, baggingu apod., neboť tyto algoritmy založené na kombinaci více klasifikátorů ve většině případů dominují svou přesností nad algoritmy využívající pouze jediný algoritmus.

Níže uvádím konfigurace používaných algoritmů. Porovnání algoritmů obsahuje navazující kapitola 8.4.

8.3.1 Přehled konfigurací vybraných algoritmů

Pokračuji přehledem konfigurací s nimiž ostatní algoritmy dosáhly nejvyšších hodnot úspěšnosti klasifikace. Uvádím pouze nejvhodnější nastavení pro každý algoritmus. Vzhledem k omezenému výstupu aplikace Weka nelze detailně sledovat vliv změn, stejně jako v případě MLPv1. Vhodnost algoritmu posuzuji dle přesnosti na testovací sadě.

REPTree

```
REPTree -M 2 -V 0.001 -N 8 -S 1 -L -1 -I 0.0
```

RandomTree

```
RandomTree -K 2 -M 2.0 -V 0.001 -S 1 -batch-size 80
```

J48

```
J48 -C 0.3 -M 2 -A -batch-size 80
```

HoeffdingTree

```
HoeffdingTree -L 2 -S 1 -E 1.0E-7 -H 0.05 -M 0.01 -G 200.0 -N 0.0
```

PART

```
PART -M 2 -C 0.2 -Q 1 -batch-size 80
```

OneR

```
OneR -B 8 -batch-size 80
```

JRip

```
JRip -F 3 -N 2.0 -O 3 -S 1 -batch-size 80
```

DecisionTable

```
DecisionTable -X 10 -E auc -S "weka.attributeSelection.BestFirst -D 1
-N 5" -batch-size 64
```

LWL

Interně využívá rozhodovací strom J48.

```
LWL -U 0 -K -1 -A "weka.core.neighboursearch.LinearNNSearch
-A \"weka.core.EuclideanDistance -R first-last\""
-W weka.classifiers.trees.J48 -- -C 0.25 -M 2
```

KStar

```
KStar -B 20 -M n -batch-size 80
```

IBk

```
IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree
-A \"weka.core.EuclideanDistance -R first-last\"
-C \"weka.core.neighboursearch.balltrees.TopDownConstructor
-S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren
-N 40\"" -batch-size 80
```

NaiveBayes

```
NaiveBayes -batch-size 80
```

BayesNet

```
BayesNet -batch-size 80
-Q weka.classifiers.bayes.net.search.local.SimulatedAnnealing --
-A 10.0 -U 10000 -D 0.999 -R 1 -S BAYES
-E weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
```

MultilayerPerceptron

Implementace neuronové sítě MLP v prostředí Weka.

```
MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a
```

8.3.2 Přehled konfigurací využívající kombinaci algoritmů

Následuje přehled vybraných algoritmů využívající některý z princip kombinace výsledků více algoritmů do jediného výstupu.

LogitBoost

Využívá kombinaci rozhodovacích stromů DecisionStump.

```
LogitBoost -P 100 -L -1.7976931348623157E308 -H 1.0 -Z 3.0 -O 1 -E 1
-S 1 -I 10 -W weka.classifiers.trees.DecisionStump -batch-size
```

SimpleLogistic

```
SimpleLogistic -I 0 -M 500 -H 50 -W 0.0
```

AdaBoostM1

Využívá kombinaci rozhodovacích stromů J48.

```
AdaBoostM1 -P 100 -S 1 -I 10 -W weka.classifiers.trees.J48
-batch-size 80 -- -C 0.25 -M 2
```

Bagging

Využívá kombinaci rozhodovacích stromů J48.

```
Bagging -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.J48
-batch-size 80 -- -C 0.25 -M 2
```

RandomCommittee

Využívá kombinaci algoritmů RandomTree.

```
RandomCommittee -S 1 -num-slots 1 -I 10
-W weka.classifiers.trees.RandomTree -- -K 0 -M 1.0 -V 0.001 -S 1
```

8.4 Výsledné porovnání úspěšnosti klasifikace

Pro všechny klasifikátory byla nalezena optimální konfigurace a výsledky učení byly ověřeny pomocí validační sady vs008. Klasifikátory s nejvyšším výkonem byly použity pro závěrečnou evaluaci na testovací sadě a nejlepší výsledky shrnuje tabulka 8.7. Kompletní výsledky pro všechny algoritmy uvádím v krácené formě v příloze D.9, podrobný rozbor je obsažen na přiloženém médiu.

Tab. 8.7: Porovnání vybraných klasifikátorů na testovací sadě (ts008).

ALG	**ACC**	$\mathbf{E}_{rate}$	$\mathbf{F}_m$	$\mathbf{G}_m$	**MCC**
REPTree	97,50%	2,50%	0,9747	0,9875	0,9727
J48	98,75%	1,25%	0,9875	0,9933	0,9861
PART	98,75%	1,25%	0,9875	0,9933	0,9861
Bagging J48	98,75%	1,25%	0,9875	0,9933	0,9861
AdaBoostM1 J48	98,75%	1,25%	0,9875	0,9933	0,9861
RandomCommittee	100,00%	0,00%	1,0000	1,0000	1,0000
MLPweka	53,75%	46,25%	0,4716	0,6231	0,4394
MLPv1	86,25%	13,75%	0,8587	0,9341	0,8514
ANNv2	100,00%	0,00%	1,0000	1,0000	1,0000

Výsledky v tabulce 8.7 jsou rozděleny do tří skupin, z nichž každá obsahuje tři nejvhodnější klasifikátory. První z nich obsahuje výsledky pro nejlepší jednoduché klasifikátory.

Mezi nejlepšími jsou pouze rozhodovací stromy s dosaženou přesností až 98, 75%. Chybovost 1, 25% způsobila nesprávná klasifikace jediné položky testovací sady, kdy byl špatně klasifikován útok z kategorie *Register test* jako *Register flood*.

Obdobná chyba často nastávala během validace, kdy docházelo k chybné klasifikaci útoku *Options scan* do skupiny *Options test*. Tyto chybné klasifikace způsobuje vlastní signatura útoku, která se vyskytuje na hranici mezi jednotlivými třídami. Klasifikátor pak může takovýto útok zařadit do nesprávné třídy.

Odstranění problému s klasifikací hraničních signatur je možné s využitím tzv. soft-decision klasifikace a následné fúze více různých klasifikátorů. Problematika je blíže objasněna v kapitole věnované fúzi.

Druhá skupina uvedená v tabulce 8.7 obsahuje kombinace algoritmů. Výsledky na validační sadě dosahovaly 100% úspěšnosti, ale při ověření dochází opět k chybné klasifikaci jediné položky. I když tyto algoritmy kombinují více různých běhů jednoho algoritmu, nedokáží úplně potlačit chybu hraničních signatur. Výjimkou je seskupení *RandomComittee*, které dosáhlo bezchybné přesnosti.

Poslední skupinu tvoří neuronové sítě. Ty sice tématicky spadají do první kategorie, ale pro porovnání rozdílů mezi implementacemi jsou umístěny do separátní kategorie. Zde již výsledky značně kolísají, což je způsobeno hlavně rozdílnými optimalizacemi prováděnými na neuronové síti. Nejvýrazněji však přesnost neuronových sítí ovlivňuje předzpracování dat.

8.4.1 Předzpracování analyzovaných dat

Pro zpřesnění klasifikace využívají neuronové sítě úpravu vstupních vektorů. Dostupné metody předzpracování závisí na algoritmu i typu vstupních dat. Primárním úkolem je normalizace vstupních dat, ale vyskytují se i další úpravy vstupního vektoru.

Přestože možnosti předzpracování jsou různé, první generace MLPv1 neprovádí žádné předzpracování. Druhá generace ANNv2 obsahuje řadu metod vysvětlených blíže v teoretické části (4.1.1). Implementace MLP v prostředí Weka může například provádět převod nominálních vstupů do binární podoby, apod. V tabulkách 8.7 a D.9 proto respektuji dostupné možnosti předzpracování pro dané algoritmy a používám vždy pouze to nastavení, které přináší nejlepší výsledky.

Tab. 8.8: Výkon neuronových sítí – bez předzpracování.

ALG	**ACC**	$\mathbf{E}_{rate}$	$\mathbf{F}_m$	$\mathbf{G}_m$	**MCC**
MLPweka	53,75%	46,25%	0,4716	0,6231	0,4394
MLPv1	86,25%	13,75%	0,8587	0,9341	0,8514
ANNv2	86,25%	13,75%	0,8581	0,9214	0,8433

Pro porovnání jednotlivých implementací je nutné stanovit stejné podmínky bez ovlivnění předzpracováním. Výsledky neuronových sítí bez předzpracování uvádím v tabulce

8.8. I přes snahu odladit MLP algoritmus v prostředí Weka dosahovala jeho úspěšnost pouze 53, 75%. Neuronové sítě MLPv1 a ANNv2 dosahovaly stejné úspěšnosti a rozdíly mezi nimi jsou minimální, nezávisle na různé struktuře, nastavení učících parametrů i chybové funkci. Obě neuronové sítě dosáhly na validační sadě nižší úspěšnosti (v řádu procent) než pro testovací sadu.

Pokud upravíme vstupní data tak, aby odpovídala výstupu po předzpracování na ANNv2, dosáhneme zvýšení výkonu všech uvedených neuronových sítí. Navýšení přesnosti je rapidní a umožňuje dosahovat vynikající klasifikace (viz tabulka 8.9). Zlepšení je pravděpodobně způsobeno navýšením počtu možných rozhodovacích hladin ve vstupní vrstvě a také vyšším počtem neuronů ve skrytých vrstvách.

Tab. 8.9: Výkon neuronových sítí – s předzpracováním.

ALG	**ACC**	$\mathbf{E}_{rate}$	$\mathbf{F}_m$	$\mathbf{G}_m$	**MCC**
MLPweka	96,25%	3,75%	0,9617	0,9800	0,9580
MLPv1	96,25%	3,75%	0,9617	0,9800	0,9580
ANNv2	100,00%	0,00%	1,0000	1,0000	1,0000

Obdobné navýšení přesnosti bylo pozorováno i pro MLP implementaci v prostředí Matlab. S použitím upravené vstupní sady se navýšila přesnost o téměř 45%, což odpovídá zlepšení algoritmu MLP v prostředí Weka. Výsledky ale negativně ovlivnily další okolnosti, které vyloučily tuto implementaci ze závěrečného srovnání.

Bezchybné klasifikace na validační i testovací sadě dosáhla pouze ANNv2, přičemž mezi nejvhodnějšími strukturami se vyskytovaly i sítě obsahující jednu skrytou vrstvu. Porovnání neuronových sítí provedené na sadě všech unikátních signatur zobrazuje tabulka 8.10. Nejlepšího výsledku dosáhla ANNv2 při použití dvou skrytých vrstev.

Tab. 8.10: Výsledky MLP na sadě všech unikátních signatur.

ALG	**Struktura**	**ACC**	$\mathbf{E}_{rate}$	$\mathbf{F}_m$	$\mathbf{G}_m$	**MCC**
ANNv2	[130,88,8]	96,78%	3,22%	0,9034	0,9262	0,9054
MLPv1	[130,80,40,8]	97,33%	2,67%	0,9149	0,9353	0,9178
MLPweka	[130,69,8]	97,44%	2,56%	0,9170	0,9387	0,9199
ANNv2	[130,60,50,8]	97,64%	2,36%	0,9346	0,9511	0,9348

Výsledky jasně zobrazují vysokou podobnost úspěšnosti mezi klasifikátory a pozitivní vliv předzpracování na výkon neuronových sítích. Ostatní klasifikátory dosahují podobných výsledků, nepřekonávají ale výsledek ANNv2 (například RandomCommittee dosahuje na sadě unikátních signatur úspěšnosti 96, 17%).

8.4.2 Fúze klasifikátorů

Fúze klasifikátorů přináší metody pro další potenciální vylepšení přesnosti klasifikace. Obecně lze zvýšení dosáhnout, pokud jsou alespoň částečně splněny tyto podmínky:

- různé druhy senzorů pro získání dat
- různé druhy signatur pro stejné entity
- různé zachycení signatury konkrétní entity
- dostupnost odlišných vstupních vektorů

V případě analýzy VoIP útoků nejsou tyto podmínky splněny. Dochází například k zachycení stejného útoku pomocí různých detektorů, nicméně na různých sondách. Díky tomu vznikají v datech nežádoucí závislosti, které nelze fúzí odstranit [54, 44].

Z výsledků dat plyne, že nejlepší klasifikátory dosahují velice vysoké přesnosti na testovací sadě. Praktickým ověřením na kompletní sadě unikátních signatur se tato přesnost potvrdila. Nízká chybovost nabízí pouze malý prostor pro možné vylepšení klasifikace. Dalším negativním faktorem je závislost všech klasifikátorů na stejné učící sadě. Stejná učící data a vstupní vektory mohou vést ke stejných chybám v klasifikaci.

Pokud se zaměříme blíže na konkrétní chyby na sadě unikátních signatur, zjistíme že se ve vysoké míře opakují i pro zbylé klasifikátory. Nejnižší chybovosti dosahuje dvouvrstvá ANNv2. Z 82 chyb v klasifikaci se $98,78\%$ z těchto chyb vyskytuje i u dalších klasifikátorů. Velice podobné je to i s dalšími klasifikátory (např. shoda $93,26\%$ u MLP v prostředí Weka). Rozhodování o výsledné klasifikaci při využití fúze využívá dle Jaina [44] obvykle některou z těchto metod:

- majority voting
- weighted majority voting
- behavior knowledge space
- weighted voting
- Dempster-Shafer theory of evidence

Shoda v chybě mezi různými klasifikátory ovlivňuje právě stanovení výsledné klasifikace. U soft–decision algoritmů lze pozorovat i velmi vysoké hodnoty posteriorní pravděpodobnosti chybných klasifikací. Weka MLP uvádí pravděpodobnost správnosti klasifikace vyšší než 95% u 31 chybných klasifikací, které tvoří $34,83\%$ všech chybných klasifikací. Daný algoritmus je tedy silně přesvědčen o správnosti rozhodnutí a může tak vahou tohoto rozhodnutí změnit rozhodnutí při fúzi na stranu chybné klasifikace, zvláště když dochází k vícečetnému výskytu chyb.

Co je ale důvodem $2,36\%$ chybných klasifikací? Problematické signatury patřící do některé z těchto kategorií:

- hraniční signatury
- fuzzing signatur
- neznámé signatury

V případě hraničních signatur dochází ke klasifikaci signatury na rozhraní dvou tříd a útok může být nesprávně klasifikován do komplementární třídy. Hranice mezi třídami bývá označována jako oblast nejistoty. Nízká oblast nejistoty je obvyklým indikátorem nízké generalizace a potencionálního efektu overfitting. Důraz na její eliminaci není vhodný, zvláště pokud je útok dle posteriorní pravděpodobnosti tohoto algoritmu správně klasifikován. Definice tříd v této práci se může pro některé útoky podobat a rozlišujícím parametrem mohou být různé hodnoty jen jednoho parametru.

Jako fuzzing se označují metody úmyslného zanesení nesmyslných hodnot do signatury útoku. Výsledkem může být selhání systémů pro detekci či chybná klasifikace. Využitím metod uvedených v této dizertační práci nelze vliv fuzzingu efektivně potlačit.

Posledním zástupcem problematických klasifikací jsou neznámé signatury. Jde o signatury nových útoků, které nebyly známy v době vytvoření učící sady (zero–day zranitelnosti). Pro vytvoření kvalitní učící sady muselo dojít i k vynechání těch signatur útoků, které nebyly dostupné v dostatečné míře nebo se vyskytovaly jen sporadicky.

Vliv těchto chybných klasifikací nelze nikdy úplně eliminovat a stávající přesnost klasifikačních algoritmů lze považovat za blížící se ideálnímu klasifikátoru. Z výše uvedených důvodů by bylo použití fúze klasifikátorů kontraproduktivní a nepřinášelo podstatné zvýšení výkonu. Možnosti rozvoje přesnosti Beekeeper se nachází především v dalších oblastech tohoto systému.

9 PŘÍNOSY DIZERTAČNÍ PRÁCE A ZÁVĚR

V rámci výzkumu pro dizertační práci byly dosaženo stanovených cílů a výsledky lze považovat za přínosné pro oblast autonomní detekce útoků. Lze je rozdělit dle toho, zda se jedná o přínos teoretický či praktický. Teoretickými přínosy jsou:

- Provedení rešeršního průzkumu a shrnutí problematiky zabezpečení VoIP provozu využívajícím SIP protokol s důrazem na potenciální slabiny VoIP infrastruktury.
- Rozborem existujících metod detekce hrozeb byly identifikovány charakteristické postupy detekce hrozeb, jejich silné a slabé stránky.
- Určení a nasazení metod pro sběr a zpracování útočných signatur.
- Stanovení a experimentální ověření metod tvorby datových sad pro potřeby detekce VoIP hrozeb.
- Byly navrženy modely neuronové sítě umožnující univerzální detekce hrozeb.
- Experimentálně byla ověřena možnost využití modelů neuronových sítí pro autonomní klasifikaci útoků.
- Návrh a experimentální ověření metod pro zpřesnění výsledků klasifikace neuronových sítí.
- Implementované algoritmy byly porovnány s dalšími dostupnými klasifikačními metodami.

Mezi praktické přínosy práce patří:

- Návrh a implementace variabilních sond umožňujících monitoring hrozeb a snadné nasazení v cílové síti.
- Návrh a implementace expertního informačního systému Beekeeper pro autonomní analýzu VoIP útoků.
- Návrh a implementace modulárních systémů obohacujících výsledky o dodatečné informace a zvyšující tak užitečnost a využitelnost dat.
- Napojení Beekeeper na další expertní systémy.
- Implementace neuronových sítí s důrazem na konfigurovatelnost a univerzálnost modelu.
- Vznikly jednoznačně definované struktury popisující SIP útoky i zachycené SIP zprávy.
- Byla vytvořena řada vzorových a prakticky využitelný sad pro detekci hrozeb.
- Vzniklo unikátní datové úložiště pro uložení detekovaných útoků.

Tato dizertační práce přináší nový návrh modelu klasifikace útoků pomocí neuronových sítí a vybraných algoritmů strojového učení. V provedených experimentech byla potvrzena aplikovatelnost tohoto řešení a v průběhu několikaletého výzkumu byly nalezeny a identifikovány optimální algoritmy a jejich konfigurace. Cílem bylo vytvoření expertního centralizovaného systému pro analýzu, klasifikaci a skladování dat o útocích na VoIP.

Důvodem pro výzkum je především rostoucí obliba a počet nasazení VoIP ústředen.

S tím je spojen i nárůst útoků a pokusů o zneužití VoIP služeb. Zvýšený zájem o nástroje v oblasti bezpečnosti je patrný i ze zapojení popisovaného systému Beekeeper do dalších výzkumných i komerčních projektů. V současnosti již existuje řada nástrojů pro detekci hrozeb a možných algoritmů jejich detekce. Tyto nástroje jsou podrobně popsány v kapitole 3, s uvedením předností i slabin zmíněných metod. Příležitost pro řešení uvedené v této dizertační práci spočívá v nalezení univerzálního přístupu pro klasifikování dat o útocích i metod získání těchto dat.

Systém Beekeeper byl od počátku vyvíjen jako klasifikátor VoIP útoků a proto je část teorie věnována stručnému základu komunikace pomocí SIP protokolu. V teto části se zaměřuji především na ty oblasti SIP protokolu, které bývají útočníky zneužívány či přímo souvisí s útoky na VoIP. Blíže rozebrána je i bezpečnost protokolu SIP a výchozí bezpečnostní opatření a doporučení definovaná ve standardu RFC 3261. Kapitola plynule přechází k přehledu známých zranitelností a útoků používaných ve VoIP infrastrukturách. Přehled bezpečnosti uzavírá kapitola věnovaná aktuálním technikám a teoretickým přístupům detekce SIP útoků.

Kapitola popisující klasifikační algoritmy uvádí čtenáře do problematiky strojového učení. Obsahuje rozdělení a přehled vlastností jednotlivých algoritmů. Detailně zpracovává zejména problematiku neuronových sítí, které tvoří jádro detekčního mechanismu navrhované v rámci dizertační práce. Identifikovány a popsány jsou i další metody strojového učení využívané pro detekci.

Cíle dizertační práce definuji v kapitole 5 ve formě, v jaké byly schváleny komisí během rigorózních zkoušek. V navazující části se již věnuji jádru práce. Nejprve uvádím rozbor a vlastnosti centralizovaného systému pro sběr dat o útocích na SIP. Jedná se o popis návrhu a implementace různých typů detekčních sond i multifunkčního centrálního prvku Beekeeper. Tento expertní informační systém zastává řadu funkcí a váže dohromady jak provedený teoretický výzkum, tak i veškeré prakticky nasazené technologie do jediného komplexního řešení.

Další kapitola provází čtenáře implementací jádra systému Beekeeper, které tvoří klasifikátor založený na neuronových sítích. Tento klasifikátor je doplněn předchozí verzí založenou také na neuronových sítích a napojením na API nástroje Weka, poskytujícího rozmanité algoritmy strojového učení pro data mining. Kapitola obsahuje i metodiku vyhodnocení a porovnání klasifikátorů z hlediska výkonu na testovacích datech.

Experimentální část se zaměřuje na nalezení optimalizovaného algoritmu řešícího problém klasifikace s maximální přesností na dostupných datech. Zároveň obsahuje i porovnání jednotlivých algoritmů mezi sebou. Nejlepších výsledků na dostupných datech dosáhla již dříve uvedená implementace pomocí neuronových sítí.

Mezi schválené cíle dizertační práce patří vytvoření sond pro sběr informací o útocích na SIP infrastrukturu. V práci předkládám koncept na bázi obrazů připravených pro instalaci ve virtualizované infrastruktuře, tak i hardwarová zařízení podporující různé architektury. Uvedené detekční sondy byly nasazeny v praxi a jsou aktivně využívány pro

sběr dat. Systém Beekeeper umožňuje pasivní monitoring těchto sond a komunikace se sondami využívá šifrovaného kanálu.

Dále systém Beekeeper provádí autonomní zpracování a klasifikaci útoků zachycených pomocí uvedené sítě sond. Detekce je založena na experimentálně ověřené klasifikaci neuronovými sítěmi, přičemž umožňuje i libovolné nasazení dalších algoritmů. Dostupné jsou vždy výsledky vybraných klasifikací.

Veškeré algoritmy uvedené v experimentální části byly testované na reálných datech získaných během více než dvouletého monitorování SIP útoků. Byla tak vytvořena odpovídající datová základna umožňující další výzkum v oblasti a přístup ke strukturovaným útočným datům. Cíle dizertační práce jsou tedy splněny ve všech bodech.

Potřebnost a aktuálnost popsané problematiky potvrzuje i zapojení Beekeeperu v dalších expertních systémech pro sdílení informacích o detekovaných bezpečnostních incidentech Warden a SABU, systému pro monitoring infrastruktur Mentat. Zároveň proběhlo jednání o nasazení v komerčním produktu společnosti InveaTech (nyní Flowmon). Beekeeper je nasazován i v prostředí monitoringu a vyhodnocení SIP komunikace v projektu TAČR: Bezpečnost mobilních zařízení a komunikace.

Široká využitelnost spočívá v modulárním designu a možných rozšířeních stávajícího stavu o další funkcionalitu. Systém není omezen pouze na klasifikaci útoků cílených na SIP protokol, ale poskytuje pracovní prostředí pro obecné zpracování dat z široké škály sond, nezávisle na jejich technologii. Vzhledem k rozvoji v oblasti IoT lze předpokládat další využití tohoto systému pro analýzu dat získaných z těchto zařízení. Celý zdrojový kód aplikace Beekeeper je otevřený a veřejně dostupný pod licencí GPL. Systém byl vyvíjen v průběhu několika let a výsledky byly publikovány ve vědeckých žurnálech a konzultovány na řadě odborných konferencí.

Budoucí rozvoj systému Beekeeper pro analýzu VoIP komunikace spočívá v rozšíření množství detekovaných incidentů a rozvoji nových učících dat. Zároveň bude probíhat i výzkum v oblasti klasifikačních algoritmů a jejich dalšího zpřesnění. Díky velkému množství zachycených útoků poskytuje Beekeeper již nyní unikátní a kvalitní zdroj informací pro další výzkum a vývoj v oblasti útoků na VoIP pomocí SIP protokolu.

LITERATURA

[1] ZYARAH, Abdullah M., Abhishek RAMESH, Cory MERKEL a Dhireesha KUDITHIPUDI. Optimized hardware framework of MLP with random hidden layers for classification applications. In: *Proceedings of SPIE: Machine Intelligence and Bio-inspired Computation: Theory and Applications X* [online]. 2016, s. 985007-1-11. DOI: 10.1117/12.2225498. Dostupné z: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2225498

[2] SACHDEVA, Monika, Krishan KUMAR a Gurvinder SINGH. A comprehensive approach to discriminate DDoS attacks from flash events. *Journal of Information Security and Applications* [online]. 2016, **26**, 8-22. DOI: 10.1016/j.jisa.2015.11.001. ISSN 22142126. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S2214212615000472

[3] CASTELLI, Mauro, Luca MANZONI a Ales POPOVIC. An Artificial Intelligence System to Predict Quality of Service in Banking Organizations. In: *Computational Intelligence and Neuroscience*. Hindawi Publishing Corporation, 2016.

[4] WITTEN, Ian. *Data mining: practical machine learning tools and techniques*. Boston, MA: Elsevier, 2016. ISBN 9780128042915.

[5] SAIED, Alan, Richard E. OVERILL a Tomasz RADZIK. Detection of known and unknown DDoS attacks using Artificial Neural Networks. *Neurocomputing* [online]. 2016, **172**, 385-393. DOI: 10.1016/j.neucom.2015.04.101. ISSN 09252312. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S092523121501053X

[6] CEPHELI, Özge, Saliha BÜYÜKÇORAK a Güneş KARABULUT KURT. Hybrid Intrusion Detection System for DDoS Attacks. *Journal of Electrical and Computer Engineering* [online]. 2016, **2016**, 1-8 . DOI: 10.1155/2016/1075648. ISSN 20900147. Dostupné z: http://www.hindawi.com/journals/jece/2016/1075648/

[7] CRUZ, Karen A. A., Jose de J. M. JUAREZ a Jose L. F. MUNOZ. Neural Net Gains Estimation Based on an Equivalent Model. In: *Computational Intelligence and Neuroscience*. Hindawi Publishing Corporation, 2016.

[8] JAGADEESAN, Lalita, Alan MC BRIDE, Vijay K. GURBANI a Jie YANG. Cognitive Security: Security Analytics and Autonomics for Virtualized Networks. In: *Proceedings of the Principles, Systems and Applications on IP Telecommunications - IPTComm '15* [online]. New York, New York, USA: ACM Press, 2015, s. 43-50. DOI: 10.1145/2843491.2843837. ISBN 9781450339490. Dostupné z: http://dl.acm.org/citation.cfm?doid=2843491.2843837

[9] BARKER, Sam. *Future Voice Strategies: mVoIP, Carrier OTT, WebRTC, HD Voice & Video Calling 2015-2020* [online]. Juniper, 2015.

[10] NIELSEN, Michael A. *Neural Networks and Deep Learning* [online]. Determination Press, 2015. Dostupné z: http://neuralnetworksanddeeplearning.com/

[11] DESHMUKH, Rashmi V. a Kailas K. DEVADKAR. Understanding DDoS Attack & its Effect in Cloud Environment. *Procedia Computer Science* [online]. 2015, **49**, 202-210 . DOI: 10.1016/j.procs.2015.04.245. ISSN 18770509. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S1877050915007541

[12] HOFFSTADT, Dirk, Erwin RATHGEB, Matthias LIEBIG, Ralf MEISTER, Yacine REBAHI a TRAN QUANG THANH. A comprehensive framework for detecting and preventing VoIP fraud and misuse. In: *2014 International Conference on Computing, Networking and Communications (ICNC)* [online]. IEEE, 2014, s. 807-813 . DOI: 10.1109/ICCNC.2014.6785441. ISBN 9781479923588. Dostupné z: http://ieeexplore.ieee.org/document/6785441/

[13] COLLIER, Mark D. a David ENDLER. *Hacking exposed: unified communications & VoIP security secrets & solutions*. Second edition. New York: McGraw-Hill Education, 2014. ISBN 0071798765.

[14] SHARMA, Ramnaresh a Manish SHRIVASTAVA. A Study of Various Intrusion Detection Model Based on Data Fusion, Neural Network and DS Theory. In: *Network*. 2. 2012.

[15] DAINOTTI, Alberto, Alistair KING, Ferdinando PAPALE a Antonio PESCAPE. Analysis of a "/0" Stealth Scan from a Botnet. In: *Proceedings of the 2012 ACM conference on Internet measurement conference*. New York: ACM, 2012. ISBN 978-1-4503-1705-4.

[16] TANG, Jin, Yu CHANG a Yong HAO. Detection and Prevention of SIP Flooding Attacks in Voice over IP networks. In: *PROCEEDINGS IEEE INFOCOM*. IEEE, 2012, s. 1161-1169. ISBN 9781467307734.

[17] SZMIT, Maciej, Anna SZMIT, Slawomir ADAMUS a Sebastian BUGALA. Implementation of Brutlag's algorithm in Anomaly Detection 3.0. In: *Proceedings of the Federated Conference on Computer Science and Information Systems*. Wrocław: Poland fedCSIS, 2012, s. 685-691. ISBN 978-83-60810-51-4.

[18] MEHTA, Anil, Neda HANTEHZADEH, Vijay K. GURBANI, Tin Kam HO a Flavia SANDER. On using multiple classifier systems for Session Initiation Protocol (SIP) anomaly detection. In: *2012 IEEE International Conference on Communications (ICC)* [online]. IEEE, 2012, s. 1101-1106 . DOI: 10.1109/ICC.2012.6364010. ISBN 9781457720536. Dostupné z: http://ieeexplore.ieee.org/document/6364010/

[19] AKILANDESWARI, V. a S. Mercy SHALINIE. Probabilistic Neural Network based attack traffic classification. In: *2012 Fourth International Conference on Advanced Computing (ICoAC)* [online]. IEEE, 2012, s. 1-8 . DOI: 10.1109/ICoAC.2012.6416848. ISBN 9781467355841. Dostupné z: http://ieeexplore.ieee.org/document/6416848/

[20] SEO, Dongwon, Heejo LEE a Ejovi NUWERE. SIPAD:SIP-VoIP Anomaly Detection using a Stateful Rule Tree. *Computer Communications*. IPC Science and Technology Press, 2012, **36**(5), 562-574. DOI: http://dx.doi.org/10.1016/j.comcom.2012.12.004. ISSN 0140-3664.

[21] KARIMAZAD, Reyhaneh a Ahmad FARAAHI. An anomaly-based method for DDoS attacks detection using RBF neural networks. In: *2011 International Conference on Network and Electronics Engineering*. 2011.

[22] LEIF MADSEN, Jim Van Meggelen. *Asterisk: the definitive guide*. 3rd ed. Sebastopol, CA: O'Reilly Media, 2011. ISBN 9780596517342.

[23] POWERS, David M. W. *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*. Bioinfo Publications, 2011, , 1-24.

[24] TANG, Jin a Yu CHENG. Quick Detection of Steathy SIP Flooding Attacks in VoIP Networks. In: *2011 ieee international conference on communications*. Ieee Press Books, 2011, s. 1-5. ISBN 9781612842325.

[25] REZAC, Filip, Miroslav VOZNAK, Karel TOMALA, Jan ROZHON a Jiri VYCHODIL. Security Analysis System for Detection Security Threats on a SIP VoIP Infratructure Elements. *Advances in Electrical and Electronic Engineering*. 2011, **2011**(9), 225-232. DOI: 10.15598/aeee.v9i5.546. ISSN 1804-3119.

[26] Zneužívání pobočkových ústředen a připojení VoIP telefonů. *Zpráva ČTÚ* [online]. 2011 . Dostupné z: http://www.ctu.cz/cs/download/ochrana_spotrebitele/ochrana_spotrebitele_zneuzivani-pripojeni-voip.pdf

[27] LI, Jin, Yong LIU a Lin GU. DDoS attack detection based on neural network. In: *2010 2nd International Symposium on Aware Computing* [online]. IEEE, 2010, s. 196-199 . DOI: 10.1109/ISAC.2010.5670479. ISBN 9781424483136. Dostupné z: http://ieeexplore.ieee.org/document/5670479/

[28] TANG, Jin, Yong HAO, Yu CHENG a Chi ZHOU. Detection of Resource-Drained Attacks on SIP-Based Wireless VoIP Networks. In: *2010 IEEE Global Telecommunications Conference GLOBECOM 2010* [online]. IEEE, 2010, s. 1-5 . DOI: 10.1109/GLOCOM.2010.5684028. ISBN 9781424456369. Dostupné z: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5684028

[29] AKBAR, M. Ali a Muddassar FAROOQ. Application of evolutionary algorithms in detection of SIP based flooding attacks. In: *Proceedings of the 11th Annual conference on Genetic and evolutionary computation - GECCO '09* [online]. New York, New York, USA: ACM Press, 2009, s. 1419- . DOI: 10.1145/1569901.1570092. ISBN 9781605583259. Dostupné z: http://portal.acm.org/citation.cfm?doid=1569901.1570092

[30] GÓMEZ, Julio. Design of a snort-based hybrid intrusion detection system. In: *International Work-Conference on Artificial Neural Networks*. Springer Berlin Heidelberg, 2009, s. 515-522. ISBN 978-3-642- 02480-1.

[31] HE, Haibo a Edwardo A. GARCIA. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*. IEEE, 2009, **21**(9), 1263-1284.

[32] SISALEM, Dorgham. *SIP security*. Chichester, U.K.: Wiley, 2009. ISBN 0470516364.

[33] TANG, Jin, Yu CHENG a Chi ZHOU. Sketch-Based SIP Flooding Detection Using Hellinger Distance. In: *IEEE Global Telecommunications Conference*. GLOBECOM, 2009, s. 3380-3385. ISBN 9781424441471.

[34] LU, Zheng a Taoxin PENG. The VoIP intrusion detection through a LVQ-based neural network. In: *Internet Technology and Secured Transaction: ICITST 2009*. IEEE, 2009, s. 1-6. ISBN 978-1-4244-5647-5.

[35] SENGAR, Hermant, Haining WANG, Duminda WIJESEKERA a Sushil JAJODIA. Detecting VoIP Floods Using the Hellinger Distance. *IEEE Transactions on Parallel and Distributed Systems* [online]. 2008, **19**(6), 794-805 . DOI: 10.1109/TPDS.2007.70786. ISSN 10459219. Dostupné z: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4359462

[36] RUSSELL, Travis. *Session Initiation Protocol (SIP): Controlling Convergent Networks*. New York: McGraw-Hill Education, 2008. ISBN 0071488529.

[37] WANG, Huiqiang, Xiaowu LIU, Jibao LAI a Ying LIANG. Network security situation awareness based on heterogeneous multi-sensor data fusion and neural network. In: *Second International Multi-Symposiums on Computer and Computational Sciences (IMSCCS 2007)* [online]. IEEE, 2007, s. 352-359. DOI: 10.1109/IMSCCS.2007.15. ISBN 0769530397. Dostupné z: http://ieeexplore.ieee.org/document/4392625/

[38] KANG, Hun Jeong, Zhi-Li ZHANG, Supranamaya RANJAN a Antonio NUCCI. SIP-based VoIP Traffic Behavior Profiling and Its Applications. In: *MineNet'07: proceedings of the Third Annual ACM Workshop on Mining*. New York, N.Y: Association for Computing Machinery, 2007, s. 39-44. ISBN 9781595937926.

[39] LANDGREBE, Thomas a R. DUIN. A simplified extension of the area under the ROC to the multiclass domain. In: *Seventeenth annual symposium of the pattern recognition association of South Africa*. 2006, s. 241-245.

[40] FAWCETT, Tom. An introduction to ROC analysis. *Pattern Recognition Letters* [online]. 2006, **27**(8), 861-874 . DOI: 10.1016/j.patrec.2005.10.010. ISSN 01678655. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S016786550500303X

[41] SUN, Yanmin, Mohamed S. KAMEL a Yang WANG. Boosting for learning multiple classes with imbalanced class distribution. In: *Sixth International Conference on Data Mining (ICDM'06)*. IEEE, 2006, s. 592-602. ISBN 0769527019.

[42] SISALEM, Dorgham, Jiri KUTHAN a Sven EHLERT. Denial of service attacks targeting a SIP VoIP infrastructure: attack scenarios and prevention mechanisms. *IEEE Network* [online]. 2006, **20**(5), 26-31 . DOI: 10.1109/MNET.2006.1705880. ISSN 08908044. Dostupné z: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1705880

[43] GENEIATAKIS, Dimitris, Tasos DAGIUKLAS, Georgios KAMBOURAKIS, Costas LAMBRINOUDAKIS, Stefanos GRITZALIS, Karlovassi EHLERT a Dorgham SISALEM. Survey of security vulnerabilities in session initiation protocol. *IEEE Communications Surveys & Tutorials* [online]. 2006, **8**(3), 68-81 . DOI: 10.1109/COMST.2006.253270. ISSN 1553877x. Dostupné z: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4020603

[44] JAIN, Anil, Karthik NANDAKUMAR a Arun ROSS. Score normalization in multimodal biometric systems. *Pattern Recognition* [online]. Elsevier, 2005, **38**(12), 2270-2285 . DOI: 10.1016/j.patcog.2005.01.012. ISSN 00313203. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S0031320305000592

[45] MORADI, Mehdi a Mohammad ZULKERNINE. A neural network based system for intrusion detection and classification of attacks. In: *Proceedings of the 2004 IEEE international conference on advances in intelligent systems-theory and applications*. IEEE, 2004, s. 148-04.

[46] DŽEROSKI, Saso a Bernard ŽENKO. Is Combining Classifiers with Stacking Better than Selecting the Best One? *Machine Learning* [online]. 2004, **54**(3), 255-273 . DOI: 10.1023/B:MACH.0000015881.36452.6e. ISSN 08856125. Dostupné z: http://link.springer.com/10.1023/B:MACH.0000015881.36452.6e

[47] FAWCETT, Tom. ROC graphs: Notes and practical considerations for researchers. *Machine learning*. 2004, **31**(1), 1-38.

[48] BILES, Simon. Detecting the Unknown with Snort and the Statistical Packet Anomaly Detection Engine (SPADE). *Tech. Rep*. Computer Security Online, 2003.

[49] SPITZNER, Lance. *Honeypots: Tracking Hackers*. Boston: Addison-Wesley, c2003. ISBN 0321108957.

[50] SIP: Session Initiation Protocol. *The Internet Engineering Task Force (IETF®)* [online]. 2002 . Dostupné z: http://www.ietf.org/rfc/rfc3261.txt

[51] HAND, David J. a Robert J. TILL. A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Machine Learning* [online]. 2001, **45**(2), 171-186 . DOI: 10.1023/A:1010920819831. ISSN 08856125. Dostupné z: http://link.springer.com/10.1023/A:1010920819831

[52] SCHWENK, Holger a Yoshua BENGIO. Boosting Neural Networks. *Neural Computation* [online]. 2000, **12**(8), 1869-1887 .

DOI: 10.1162/089976600300015178. ISSN 08997667. Dostupné z: http://www.mitpressjournals.org/doi/abs/10.1162/089976600300015178

[53] FRANK, Eibe a Ian H. WITTEN. Generating Accurate Rule Sets Without Global Optimization. In: *Fifteenth International Conference on Machine Learning*. 1998, s. 144-151.

[54] KITTLER, Josef, Mohamad HATEF, Robert P.W. DUIN a Jiri MATAS. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence*. IEEE, 1998, **20**(3), 226-239. ISSN 0162882898.

[55] VONDRÁK, Ivo. *Umělá inteligence a neuronové sítě*. Dot. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, 1998. ISBN 8070782595.

[56] COHEN, William W. Fast Effective Rule Induction. In: *Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995, s. 115-123.

[57] CLEARLY, John G. a Leonard E. TRIGG. K*: An instance-based learner using an entropic distance measure. In: *Proceedings of the 12th International Conference on Machine learning*. 5. 1995.

[58] KOHAVI, Ron. The power of decision tables. In: *European conference on machine learning*. Springer Berlin Heidelberg, 1995, s. 174-189.

[59] QUINLAN, Ross J. C4.5: Programs for Machine Learning. *Machine Learning* [online]. Morgan Kaufmann Publishers, 1994, **16**(3), 235-240 . DOI: 10.1007/BF00993309. ISSN 08856125. Dostupné z: http://link.springer.com/10.1007/BF00993309

[60] HOLTE, Robert C. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*. 1993, **11**, 63-91.

CITOVANÉ PŘÍSPĚVKY AUTORA

[Saf01] SAFARIK, Jakub a Jiri SLACHTA. Comparison of artificial intelligence classifiers for SIP attack data. In: *Proceedings of SPIE: Machine Intelligence and Bio-inspired Computation: Theory and Applications X* [online]. 9850. SPIE, 2016. DOI: 10.1117/12.2225292. ISBN 978-151060091-1. ISSN 0277786X. Dostupné z: `http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2225292`. Konferenční článek, Baltimore, USA (SJR 0,216).

[Saf02] SAFARIK, Jakub a Jiri SLACHTA. VoIP attacks detection engine based on neural network. In: *Proceedings of SPIE: Independent Component Analyses, Compressive Sampling, Large Data Analyses (LDA), Neural Networks, Biosystems, and Nanoengineering XIII* [print]. 9496. SPIE, 2015. DOI: 10.1117/12.2178182. ISBN 978-162841612-1. ISSN 0277786X. Dostupné z: `http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2178182`. Konferenční článek, Baltimore, USA (SJR 0,216).

[Saf03] SAFARIK, Jakub, Miroslav VOZNAK, Miralem MEHIC, Pavol PARTILA a Martin MIKULEC. Neural network classifier of attacks in IP telephony. In: *Proceedings of SPIE: Independent Component Analyses, Compressive Sampling, Wavelets, Neural Net, Biosystems, and Nanoengineering XII* [online]. 9118. SPIE, 2014. DOI: 10.1117/12.2050671. ISBN 978-162841055-6. ISSN 0277786X. Dostupné z: `http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2050671`. Konferenční článek, Baltimore, USA (SJR 0,220).

[Saf04] SAFARIK, Jakub, Miroslav VOZNAK, Filip REZAC, Pavol PARTILA a Karel TOMALA. Automatic analysis of attack data from distributed honeypot network. In: *Proceeding of SPIE: Mobile Multimedia/Image Processing, Security, and Applications* [online]. SPIE, 2013. DOI: 10.1117/12.2015514. ISBN 978-081949546-4. ISSN 0277786X. Dostupné z: `http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2015514`. Konferenční článek, Baltimore, USA (SJR 0,202).

[Saf05] SAFARIK, Jakub, Miroslav VOZNAK, Filip REZAC a Lukas MACURA. IP telephony server emulation for monitoring and analysis of malicious activity in VOIP network. *Komunikacie*. 2013, (Vol. 15, 2 A), 191-196. ISSN 13354205. Článek v časopise (SJR 0,209).

[Saf06] SAFARIK, Jakub, Pavol PARTILA, Filip REZAC, Lukas MACURA a Miroslav VOZNAK. Automatic classification of attacks on IP telephony. *Advances in Electrical and Electronic Engineering*. 2013, (Vol. 11, 6), 481-486. DOI: 10.15598/aeee.v11i6.899. ISSN 13361376. Článek v časopise (SJR 0,240).

[Saf07] VOZNAK, Miroslav a Jakub SAFARIK. DoS attacks targeting SIP server and improvements of robustness. *International Journal of Mathematics and Computers*

in Simulation. 2012-07, (Vol. 6, 1), 177-184. ISSN 19980159. Článek v časopise (SJR 0,226).

[Saf08] SAFARIK, Jakub, Miroslav VOZNAK a Filip REZAC. Security Evaluation of Multimedia Systems. In: *Proc. of Networking to Services*. 2012-05. ISBN 978-90-77559-21-5. Článek ve sborníku.

[Saf09] VOZNAK, Miroslav, Jakub SAFARIK, Lukas MACURA a Filip REZAC. Malicious Behavior in Voice over IP Infrastructure. In: *Recent researches in communications and computers proceedings of the 16th WSEAS International Conference on Communications (part of CSCC'12)*. WSEAS, 2012. ISBN 9781618041098. Konferenční článek, Kos, Řecko.

[Saf10] SAFARIK, Jakub, Miroslav VOZNAK, Filip REZAC a Lukas MACURA. Malicious traffic monitoring and its evaluation in VoIP infrastructure. In: *2012 35th International Conference on Telecommunications and Signal Processing (TSP)* [online]. IEEE, 2012, s. 259-262. DOI: 10.1109/TSP.2012.6256294. ISBN 9781467311182. Dostupné z: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6256294`. Konferenční článek, Praha, Česká republika.

[Saf11] SAFARIK, Jakub, Miroslav VOZNAK, Jan ROZHON a Karel TOMALA. Improvements of SIP Proxy Robustness against DoS Attacks. In: *The 13th International Conference on Research in Telecommunication Technologies*. 2011, s. 63-66. ISBN 978-80-214-4283-2. Konferenční článek, Těchov, Česká republika.

[Saf12] VOZNAK, Miroslav a Jakub SAFARIK. SIP Proxy Robustness against DoS Attacks. In: *Recent researches in mathematical methods in electrical engineering and computer science: Proceedings of the applied computing conference 2011*. WSEAS Press, 2011. ISBN 9781618040510. Konferenční článek, Angers, Francie.

PUBLIKAČNÍ ČINNOST AUTORA

K doložení svých vědecko–výzkumných aktivit přikládám i aktuální stav záznamů v relevantních vědeckých databázích ke dni odevzdání tohoto dokumentu.

- Celkový počet publikací včetně technických zpráv (2011-2016): **36**
- Publikace v bibliografické databázi ISI - Web of Knowledge: **12**
- Publikace v bibliografické databázi SCOPUS: **27**
- h-index podle ISI - Web of Knowledge: **1**
- h-index podle SCOPUS: **4**
- Publikace relevantní k problematice dizertační práce indexované na ISI – Web of Knowledge a Elsevier Scopus: **17**

Participace na řešení projektů během studia

- **TA ČR TF01000091** – Specifický výzkum: Bezpečnost mobilních zařízení a komunikace.
- **SP2016/170** – Specifický výzkum: Vytěžování informací z komunikačních sítí, jejich modelování a simulace.
- **SP2015/82** – Specifický výzkum: Vytěžování informací z komunikačních sítí, jejich modelování a simulace.
- **SP2014/72** – Specifický výzkum: Výzkum atmosférických vlivů na přenosy v rádiovém kanálu.
- **02540/2013/RRC SPP: MK 9333423** – Specifický výzkum: Zabezpečení komunikace na sítích GSM, UMTS, LTE
- **SP2013/94** – Specifický výzkum: Výzkum vlivu okolního prostředí na vlastnosti rádiového kanálu a vývoj nových přístupů k hodnocení kvality služeb (QoS) multimedií v sítích 4G.
- **SP2012/180** – Specifický výzkum: Změny podmínek šíření rádiových signálů vlivem počasí.
- **FRVS2011/1672** – Inovace laboratorních cvičení předmětu Voice over Internet Protocol v oblasti použití pobočkových ústředen a zabezpečené komunikace.
- Projekt společného „Vývojového, školícího a testovacího centra OpenScape“ s firmou SIEMENS/UNIFY/iXperta podpořený částkou 1 mil. Kč v období 2013-2015.
- Projekt spolupráce mezi fakultou FEI a akademií CZ.NIC zaměřený na pořádání kurzů v oblasti IT pro zaměstnance firem a studenty fakulty

A POLE V HLAVIČCE SIP ZPRÁV

Tab. A.1: Podporovaná pole v hlavičce dle RFC 3261.

Header field	**where**	**proxy**	**ACK**	**BYE**	**CAN**	**INV**	**OPT**	**REG**
Accept	R		-	o	-	o	m*	o
Accept	2xx		-	-	-	o	m*	o
Accept	415		-	c	-	c	c	c
Accept-Encoding	R		-	o	-	o	o	o
Accept-Encoding	2xx		-	-	-	o	m*	o
Accept-Encoding	415		-	c	-	c	c	c
Accept-Language	R		-	o	-	o	o	o
Accept-Language	2xx		-	-	-	o	m*	o
Accept-Language	415		-	c	-	c	c	c
Alert-Info	R	ar	-	-	-	o	-	-
Alert-Info	180	ar	-	-	-	o	-	-
Allow	R		-	o	-	o	o	o
Allow	2xx		-	o	-	m*	m*	o
Allow	r		-	o	-	o	o	o
Allow	405		-	m	-	m	m	m
Authentication-Info	2xx		-	o	-	o	o	o
Authorization	R		o	o	o	o	o	o
Call-ID	c	r	m	m	m	m	m	m
Call-Info		ar	-	-	-	o	o	o
Contact	R		o	-	-	m	o	o
Contact	1xx		-	-	-	o	-	-
Contact	2xx		-	-	-	m	o	o
Contact	3xx	d	-	o	-	o	o	o
Contact	485		-	o	-	o	o	o
Content-Disposition			o	o	-	o	o	o
Content-Encoding			o	o	-	o	o	o
Content-Language			o	o	-	o	o	o
Content-Length		ar	t	t	t	t	t	t
Content-Type			*	*	-	*	*	*
CSeq	c	r	m	m	m	m	m	m
Date		a	o	o	o	o	o	o
Error-Info	300-699	a	-	o	o	o	o	o
Expires			-	-	-	o	-	o
From	c	r	m	m	m	m	m	m

Pokračování na další straně

Tab. A.1 – Pokračování z předchozí strany

Header field	**where**	**proxy**	**ACK**	**BYE**	**CAN**	**INV**	**OPT**	**REG**
In-Reply-To	R		-	-	-	o	-	-
Max-Forwards	R	amr	m	m	m	m	m	m
Min-Expires	423		-	-	-	-	-	m
MIME-Version			o	o	-	o	o	o
Organization		ar	-	-	-	o	o	o
Priority	R	ar	-	-	-	o	-	-
Proxy-Authenticate	407	ar	-	m	-	m	m	m
Proxy-Authenticate	401	ar	-	o	o	o	o	o
Proxy-Authorization	R	dr	o	o	-	o	o	o
Proxy-Require	R	ar	-	o	-	o	o	o
Record-Route	R	ar	o	o	o	o	o	-
Record-Route	2xx,18x	mr	-	o	o	o	o	-
Reply-To			-	-	-	o	-	-
Require		ar	-	c	-	c	c	c
Retry-After	404,413		-	o	o	o	o	o
Retry-After	480,486		-	o	o	o	o	o
	500,503		-	o	o	o	o	o
	600,603		-	o	o	o	o	o
Route	R	adr	c	c	c	c	c	c
Server	r		-	o	o	o	o	o
Subject	R		-	-	-	o	-	-
Supported	R		-	o	o	m*	o	o
Supported	2xx		-	o	o	m*	m*	o
Timestamp			o	o	o	o	o	o
To	c(1)	r	m	m	m	m	m	m
Unsupported	420		-	m	-	m	m	m
User-Agent			o	o	o	o	o	o
Via	R	amr	m	m	m	m	m	m
Via	rc	dr	m	m	m	m	m	m
Warning		r	-	o	o	o	o	o
WWW-Authenticate	401	ar	-	m	-	m	m	m
WWW-Authenticate	407	ar	-	o	-	o	o	o

Vysvětlivky

Následuje vysvětlení jednotlivých znaků v tabulce dle RFC 3261 [50]. Sloupec *where* označuje, zda se pole vyskytuje v požadavku či odpovědi.

- **R**: Pole se může vyskytovat pouze v rámci požadavku (request).
- **r**: Pole se může vyskytovat pouze v rámci odpovědi (response).
- **2xx, 4xx, atd.**: Numerická hodnota kódu v odpovědi, pro kterou je pole využito.
- **c**: Pole je kopírováno z požadavku do odpovědi.
- Pokud není uvedeno nic, pole se může vyskytovat ve všech požadavcích a odpovědích.

Sloupec *proxy* označuje, které operace je SIP proxy oprávněna nad poli provádět.

- **a**: SIP proxy může přidat či spojit pole, pokud není přítomno.
- **m**: SIP proxy může modifikovat hodnotu pole.
- **d**: SIP proxy může vymazat hodnotu pole.
- **r**: SIP proxy musí být schopna přečíst hodnotu pole – nelze ji tedy šifrovat.

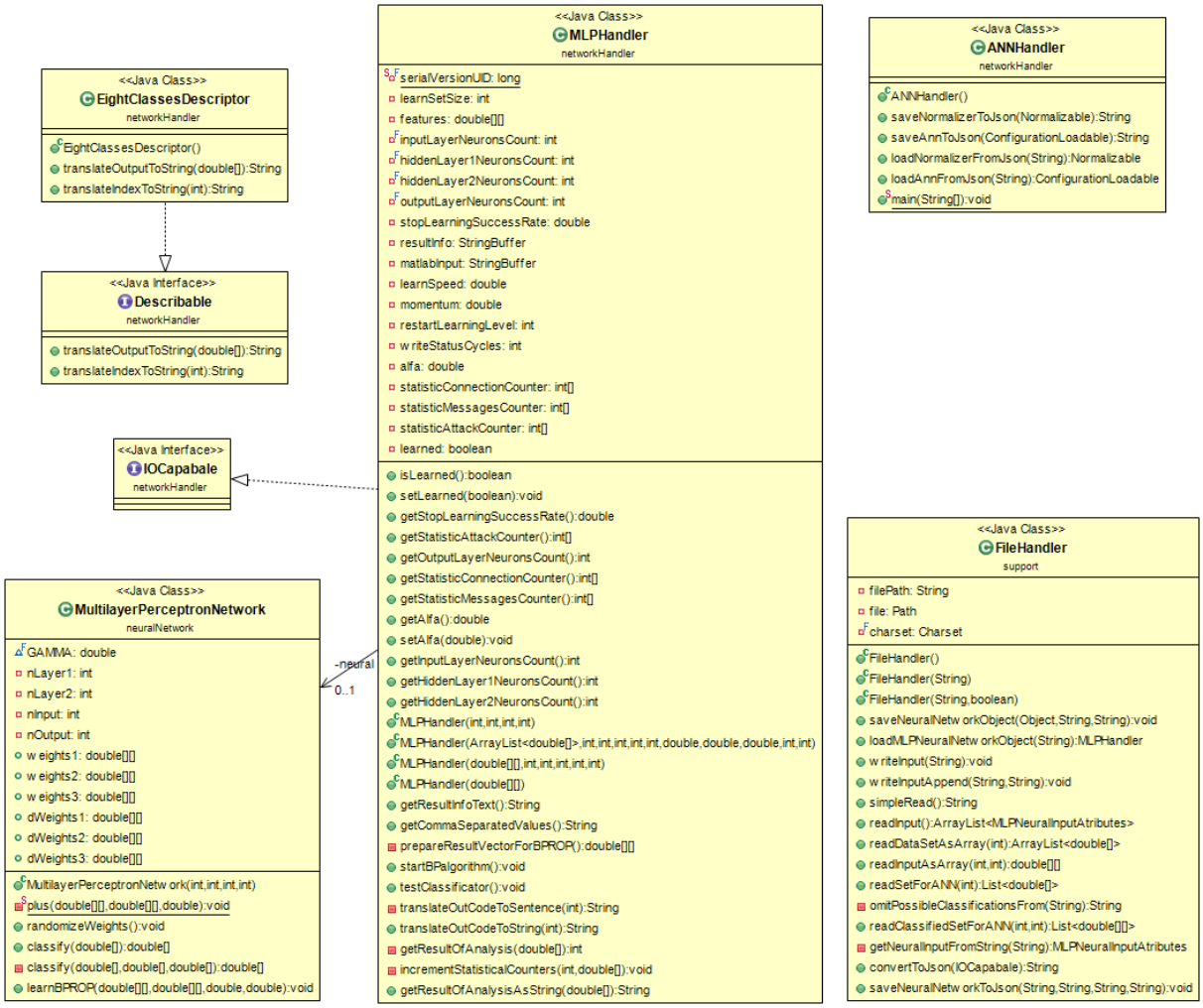
Posledních 6 sloupců se vztahuje k použitým SIP metodám.

- **c**: Podmíněno kontextem zprávy.
- **m**: Pole je povinné.
- **m***: Pole by mělo být zasláno, ale klient nebo server musí bý připraven SIP zprávu zpracovat i pokud chybí.
- **o**: Pole je volitelné.
- **t**: Pole by mělo být zasláno, ale klient nebo server musí bý připraven SIP zprávu zpracovat i pokud chybí. V případě streamového protokolu musí být zasláno (např.: TCP).
- **t***: Viz. *t*. Pole je povinné, pokud není tělo zprávy prázdné.
- **-**: Pole je nodostupné pro danou SIP zprávu.

Převzato z RFC 3261.

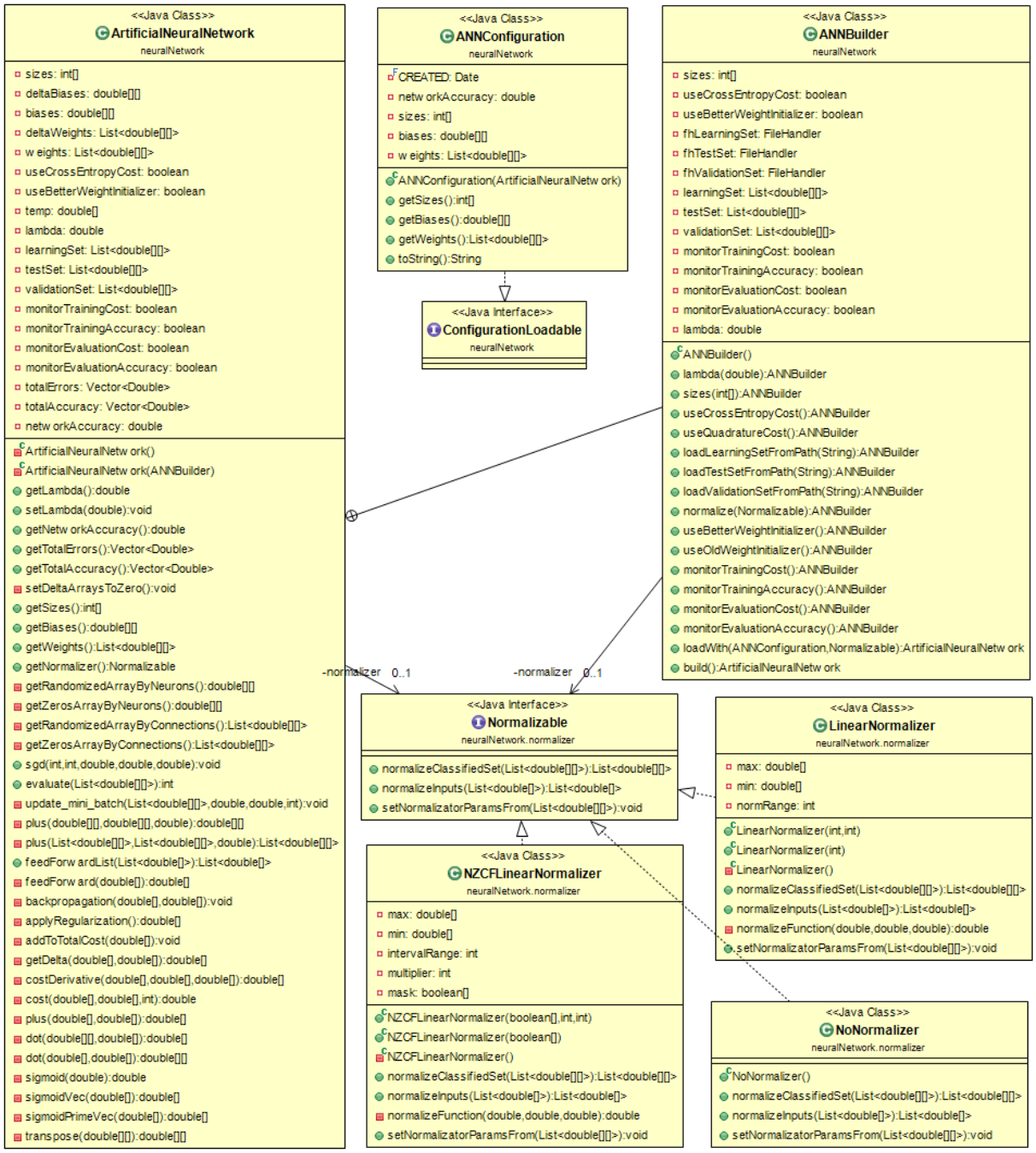
B VYBRANÉ TŘÍDNÍ DIAGRAMY

B.1 Multilayer perceptron



Obr. B.1: Třídní diagram pro MLP.

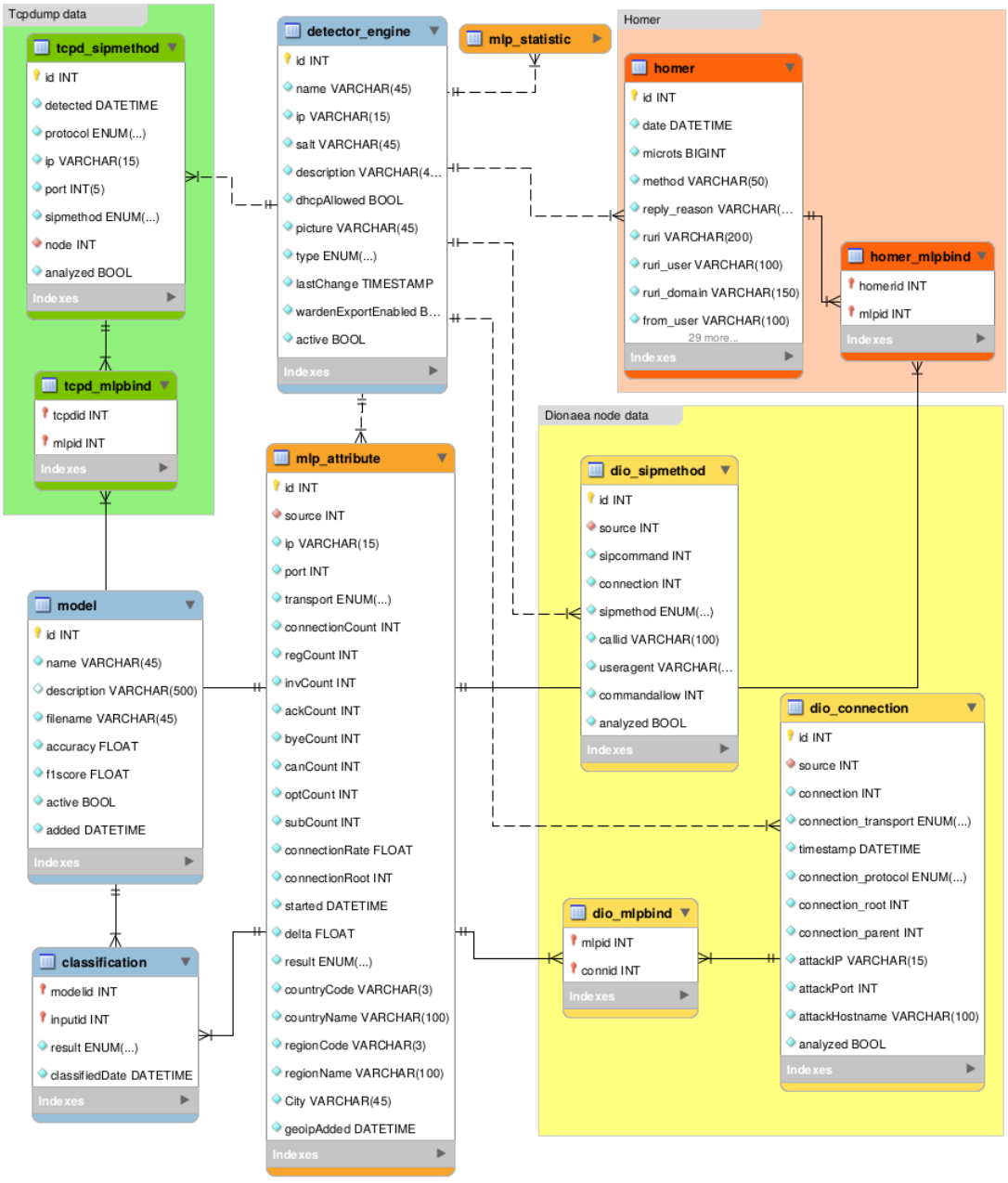
B.2 Artificial Neural Network



Obr. B.2: Třídní diagram pro ANN.

C IMPLEMENTACE SYSTÉMU BEEKEEPER

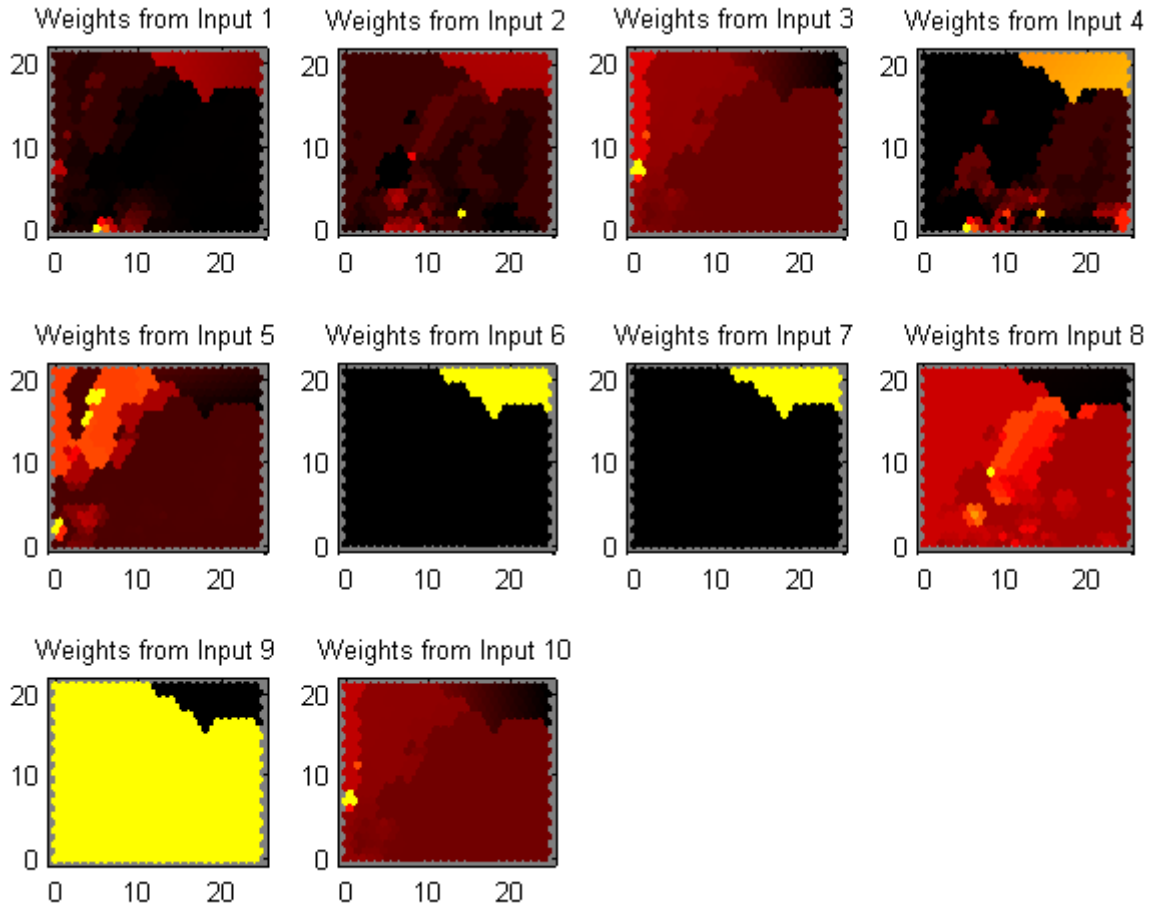
C.1 Struktura databáze



Obr. C.1: Struktura MySQL databáze pro Beekeeper.

D EXPERIMENTÁLNÍ OVĚŘENÍ DAT

D.1 Clustering datových sad



Obr. D.1: SOM – analýza vstupního vektoru pro učící sadu 6 (ls006).

D.2 Vlastnosti datových sad

Tab. D.1: Vlastnosti atributů učící sady (ls008).

	Delta	**ConnC**	**RC**	**IC**	**AC**	**BC**	**CC**	**OC**	**SC**	**ConnRate**
x_{min}	0	1	0	0	0	0	0	0	0	0
x_{max}	43869	556	246720	277	347	10	88	70	4	123361
$\bar{x}$	527,909	12,262	1462,509	7,397	3,637	0,063	1,950	2,075	0,013	663,052
σ	3186,451	42,731	14199,815	27,754	25,037	0,723	7,853	6,261	0,224	7013,182

Tab. D.2: Vlastnosti atributů validační sady (vs008).

	Delta	**ConnC**	**RC**	**IC**	**AC**	**BC**	**CC**	**OC**	**SC**	**ConnRate**
x_{min}	0	1	0	0	0	0	0	0	0	0
x_{max}	147126	214	123360	7860	373	0	44	38	0	61680,500
$\bar{x}$	2130,363	10,588	2208,725	103,588	5,825	0	2,388	1,675	0	1034,292
σ	16436,312	26,628	14049,944	878,309	41,961	0	7,779	5,118	0	6958,444

Tab. D.3: Vlastnosti atributů testovací sady (ts008).

	Delta	**ConnC**	**RC**	**IC**	**AC**	**BC**	**CC**	**OC**	**SC**	**ConnRate**
x_{min}	0	1	0	0	0	0	0	0	0	0
x_{max}	43869	556	246720	277	347	10	88	70	4	123361
$\bar{x}$	527,909	12,262	1462,509	7,397	3,637	0,063	1,950	2,075	0,013	663,052
σ	3186,451	42,730	14199,815	27,754	25,037	0,723	7,853	6,261	0,224	7013,182

D.3 Určení optimálních parametrů pro neuronovou síť ANN

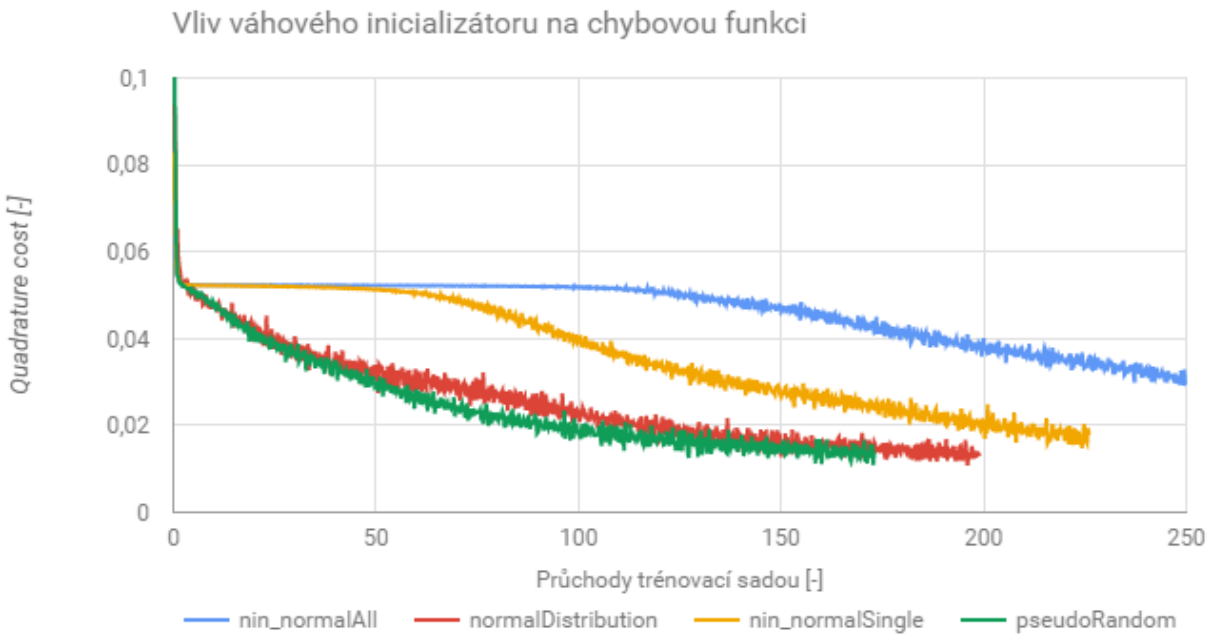
Tab. D.4: Vliv inicializace vah na učení neuronové sítě – kompletní.

Struktura	**PPE**	**Chybová funkce** **Inicializace vah**	**Sm. odchylka PPE** **Průměrné PPE**
[130, 80, 30, 8]	71,233		
[130, 50, 30, 8]	83,133		
[130, 70, 20, 8]	95,600		
[130, 90, 20, 8]	93,567		
[130, 80, 50, 8]	71,333	CEC	$\sigma = 13,794$
[130, 90, 30, 8]	78,067	Normální	$\bar{x} = 84,730$
[130, 100, 20, 8]	106,633		
[130, 60, 20, 8]	67,367		
[130, 40, 30, 8]	78,633		
[130, 120, 20, 8]	101,733		
[130, 80, 30, 8]	47,567		
[130, 50, 30, 8]	40,700		
[130, 70, 20, 8]	45,100		
[130, 90, 20, 8]	41,300		
[130, 80, 50, 8]	33,167	CEC	$\sigma = 9,395$
[130, 90, 30, 8]	36,233	Pseudonáhodná	$\bar{x} = 42,280$
[130, 100, 20, 8]	31,333		
[130, 60, 20, 8]	57,367		
[130, 40, 30, 8]	33,333		
[130, 120, 20, 8]	56,700		
[130, 80, 30, 8]	185,367		
[130, 50, 30, 8]	152,400		
[130, 70, 20, 8]	151,800		
[130, 90, 20, 8]	134,467		
[130, 80, 50, 8]	218,700	QC	$\sigma = 35,342$
[130, 90, 30, 8]	184,733	Normální	$\bar{x} = 156,573$
[130, 100, 20, 8]	99,867		
[130, 60, 20, 8]	136,567		
[130, 40, 30, 8]	178,767		
[130, 120, 20, 8]	123,067		

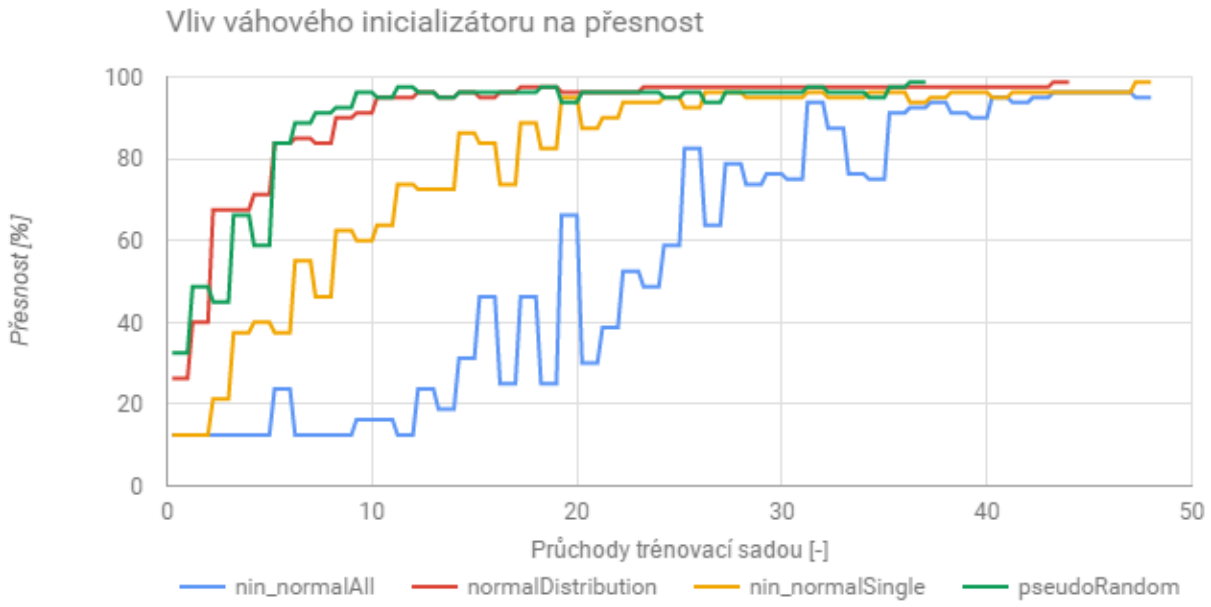
Pokračování na další straně

Tab. D.4 – Pokračování z předchozí strany

Struktura	**PPE**	**Chybová funkce** **Inicializace vah**	**Sm. odchylka PPE** **Průměrné PPE**
[130, 80, 30, 8]	104,700		
[130, 50, 30, 8]	94,267		
[130, 70, 20, 8]	67,067		
[130, 90, 20, 8]	46,033		
[130, 80, 50, 8]	225,033	QC	$\sigma = 52,910$
[130, 90, 30, 8]	93,500	Pseudonáhodná	$\bar{x} = 86,840$
[130, 100, 20, 8]	53,000		
[130, 60, 20, 8]	58,333		
[130, 40, 30, 8]	79,833		
[130, 120, 20, 8]	46,633		



Obr. D.2: Vliv inicializace vah na chybovou funkci QC.



Obr. D.3: Vliv inicializace vah na přesnost neuronové sítě (CEC).

Tab. D.5: Výsledný PPE pro hodnoty μ.

Struktura	**PPE**	μ	**Sm. odchylka PPE** **Průměrné PPE**
[130, 80, 30, 8]	35,533	1.0	$\sigma = 11,896$ $\bar{x} = 38,833$
[130, 50, 30, 8]	33,967		
[130, 70, 20, 8]	46,400		
[130, 90, 20, 8]	35,667		
[130, 80, 50, 8]	22,400		
[130, 90, 30, 8]	22,800		
[130, 100, 20, 8]	45,367		
[130, 60, 20, 8]	59,767		
[130, 40, 30, 8]	51,133		
[130, 120, 20, 8]	35,300		
[130, 80, 30, 8]	148,100	0.9	$\sigma = 20,574$ $\bar{x} = 149,370$
[130, 50, 30, 8]	151,467		
[130, 70, 20, 8]	162,900		
[130, 90, 20, 8]	170,700		
[130, 80, 50, 8]	105,067		
[130, 90, 30, 8]	137,100		
[130, 100, 20, 8]	159,833		
[130, 60, 20, 8]	158,833		
[130, 40, 30, 8]	170,500		
[130, 120, 20, 8]	129,200		
[130, 80, 30, 8]	132,600	0.8	$\sigma = 25,416$ $\bar{x} = 160,617$
[130, 50, 30, 8]	182,933		
[130, 70, 20, 8]	174,967		
[130, 90, 20, 8]	161,000		
[130, 80, 50, 8]	116,833		
[130, 90, 30, 8]	153,967		
[130, 100, 20, 8]	143,833		
[130, 60, 20, 8]	204,600		
[130, 40, 30, 8]	172,533		
[130, 120, 20, 8]	162,900		

Tab. D.6: Velikost fragmentu učící sady a vliv na učení neuronové sítě.

Struktura	**PPE**	**MBS**	**Sm. odchylka PPE Průměrné PPE**
[130, 80, 30, 8]	250,000	320	$\sigma = 0,000$ $\bar{x} = 250,000$
[130, 50, 30, 8]	250,000		
[130, 70, 20, 8]	250,000		
[130, 90, 20, 8]	250,000		
[130, 80, 50, 8]	250,000		
[130, 90, 30, 8]	250,000		
[130, 100, 20, 8]	250,000		
[130, 60, 20, 8]	250,000		
[130, 40, 30, 8]	250,000		
[130, 120, 20, 8]	250,000		
130, 80, 30, 8]	250,000	160	$\sigma = 3,300$ $\bar{x} = 248,847$
[130, 50, 30, 8]	250,000		
[130, 70, 20, 8]	248,967		
[130, 90, 20, 8]	250,000		
[130, 80, 50, 8]	250,000		
[130, 90, 30, 8]	250,000		
[130, 100, 20, 8]	250,000		
[130, 60, 20, 8]	250,000		
[130, 40, 30, 8]	239,500		
[130, 120, 20, 8]	250,000		
[130, 80, 30, 8]	43,900	80	$\sigma = 8,537$ $\bar{x} = 39,990$
[130, 50, 30, 8]	42,733		
[130, 70, 20, 8]	33,733		
[130, 90, 20, 8]	33,733		
[130, 80, 50, 8]	39,467		
[130, 90, 30, 8]	34,900		
[130, 100, 20, 8]	48,933		
[130, 60, 20, 8]	34,500		
[130, 40, 30, 8]	57,900		
[130, 120, 20, 8]	30,100		

Pokračování na další straně

Tab. D.6 – Pokračování z předchozí strany

Struktura	**PPE**	**MBS**	**Sm. odchylka PPE** **Průměrné PPE**
[130, 80, 30, 8]	35,533	64	$\sigma = 11,896$ $\bar{x} = 38,833$
[130, 50, 30, 8]	33,967		
[130, 70, 20, 8]	46,400		
[130, 90, 20, 8]	35,667		
[130, 80, 50, 8]	22,400		
[130, 90, 30, 8]	22,800		
[130, 100, 20, 8]	45,367		
[130, 60, 20, 8]	59,767		
[130, 40, 30, 8]	51,133		
[130, 120, 20, 8]	35,300		
[130, 80, 30, 8]	65,233	40	$\sigma = 8,683$ $\bar{x} = 56,233$
[130, 50, 30, 8]	45,767		
[130, 70, 20, 8]	57,100		
[130, 90, 20, 8]	57,433		
[130, 80, 50, 8]	50,233		
[130, 90, 30, 8]	62,667		
[130, 100, 20, 8]	53,967		
[130, 60, 20, 8]	53,433		
[130, 40, 30, 8]	72,167		
[130, 120, 20, 8]	44,333		
[130, 80, 30, 8]	73,933	32	$\sigma = 8,582$ $\bar{x} = 65,407$
[130, 50, 30, 8]	64,400		
[130, 70, 20, 8]	78,633		
[130, 90, 20, 8]	55,100		
[130, 80, 50, 8]	73,433		
[130, 90, 30, 8]	50,867		
[130, 100, 20, 8]	66,200		
[130, 60, 20, 8]	60,700		
[130, 40, 30, 8]	67,567		
[130, 120, 20, 8]	63,233		

Pokračování na další straně

Tab. D.6 – Pokračování z předchozí strany

Struktura	**PPE**	**MBS**	**Sm. odchylka PPE** **Průměrné PPE**
[130, 80, 30, 8]	87,633	20	$\sigma = 18,136$ $\bar{x} = 101,717$
[130, 50, 30, 8]	112,333		
[130, 70, 20, 8]	129,467		
[130, 90, 20, 8]	101,367		
[130, 80, 50, 8]	95,467		
[130, 90, 30, 8]	75,333		
[130, 100, 20, 8]	97,967		
[130, 60, 20, 8]	132,033		
[130, 40, 30, 8]	97,833		
[130, 120, 20, 8]	87,733		

Tab. D.7: Velikost L2 regularizace a vliv na učení neuronové sítě.

Struktura	**PPE**	λ	**Sm. odchylka PPE Průměrné PPE**
[130, 80, 30, 8]	35,533	0	$\sigma = 11,896$ $\bar{x} = 33,833$
[130, 50, 30, 8]	33,967		
[130, 70, 20, 8]	46,400		
[130, 90, 20, 8]	35,667		
[130, 80, 50, 8]	22,400		
[130, 90, 30, 8]	22,800		
[130, 100, 20, 8]	45,367		
[130, 60, 20, 8]	59,767		
[130, 40, 30, 8]	51,133		
[130, 120, 20, 8]	35,300		
[130, 80, 30, 8]	37,767	1,00E-01	$\sigma = 3,478$ $\bar{x} = 40,323$
[130, 50, 30, 8]	40,233		
[130, 70, 20, 8]	39,700		
[130, 90, 20, 8]	37,100		
[130, 80, 50, 8]	43,100		
[130, 90, 30, 8]	39,400		
[130, 100, 20, 8]	35,700		
[130, 60, 20, 8]	44,967		
[130, 40, 30, 8]	46,467		
[130, 120, 20, 8]	38,800		
[130, 80, 30, 8]	37,600	1,00E-02	$\sigma = 12,024$ $\bar{x} = 40,460$
[130, 50, 30, 8]	51,000		
[130, 70, 20, 8]	68,933		
[130, 90, 20, 8]	30,133		
[130, 80, 50, 8]	35,933		
[130, 90, 30, 8]	30,267		
[130, 100, 20, 8]	39,633		
[130, 60, 20, 8]	32,933		
[130, 40, 30, 8]	45,367		
[130, 120, 20, 8]	32,800		

Pokračování na další straně

Tab. D.7 – Pokračování z předchozí strany

Struktura	**PPE**	λ	**Sm. odchylka PPE** **Průměrné PPE**
[130, 80, 30, 8]	47,967	1,00E-03	$\sigma = 9,752$ $\bar{x} = 42,237$
[130, 50, 30, 8]	40,267		
[130, 70, 20, 8]	43,467		
[130, 90, 20, 8]	58,300		
[130, 80, 50, 8]	26,167		
[130, 90, 30, 8]	33,867		
[130, 100, 20, 8]	41,200		
[130, 60, 20, 8]	55,767		
[130, 40, 30, 8]	39,367		
[130, 120, 20, 8]	36,000		
[130, 80, 30, 8]	34,367	1,00E-05	$\sigma = 9,286$ $\bar{x} = 41,067$
[130, 50, 30, 8]	34,000		
[130, 70, 20, 8]	29,400		
[130, 90, 20, 8]	48,300		
[130, 80, 50, 8]	36,567		
[130, 90, 30, 8]	33,933		
[130, 100, 20, 8]	55,133		
[130, 60, 20, 8]	43,500		
[130, 40, 30, 8]	55,767		
[130, 120, 20, 8]	39,700		

D.4 Určení optimálních parametrů pro neuronovou síť MLP

Tab. D.8: Přesnost učení MLP v závislosti na momentu.

Přesnost (ACC)				
$\mu = 1$	$\mu = 0,8$	$\mu = 0,7$	$\mu = 0,5$	$\mu = 0,0$
80,0%	80,0%	74,0%	72,0%	74,0%
82,0%	78,0%	78,0%	74,0%	76,0%
71,0%	78,0%	78,0%	73,0%	78,0%
79,0%	78,0%	73,0%	76,0%	81,0%
74,0%	78,0%	77,0%	75,0%	77,0%
75,0%	78,0%	70,0%	71,0%	78,0%
70,0%	78,0%	76,0%	78,0%	75,0%
71,0%	78,0%	73,0%	72,0%	76,0%
79,0%	77,0%	81,0%	76,0%	76,0%
71,0%	77,0%	71,0%	73,0%	75,0%
73,0%	77,0%	78,0%	83,0%	73,0%
76,0%	77,0%	77,0%	77,0%	77,0%
77,0%	75,0%	75,0%	76,0%	79,0%
71,0%	75,0%	78,0%	73,0%	77,0%
75,0%	75,0%	73,0%	72,0%	78,0%
74,0%	75,0%	80,0%	78,0%	76,0%
72,0%	75,0%	74,0%	77,0%	80,0%
74,0%	75,0%	77,0%	76,0%	80,0%
72,0%	75,0%	74,0%	75,0%	79,0%
73,0%	74,0%	72,0%	78,0%	77,0%
79,0%	74,0%	73,0%	74,0%	75,0%
76,0%	74,0%	78,0%	76,0%	75,0%
73,0%	73,0%	72,0%	80,0%	78,0%
77,0%	73,0%	76,0%	79,0%	79,0%
76,0%	73,0%	77,0%	81,0%	78,0%
74,0%	73,0%	71,0%	75,0%	76,0%
80,0%	73,0%	72,0%	76,0%	77,0%
72,0%	73,0%	78,0%	70,0%	79,0%
72,0%	73,0%	73,0%	73,0%	75,0%
73,0%	73,0%	75,0%	80,0%	80,0%
75,0%	73,0%	76,0%	76,0%	75,0%
76,0%	73,0%	73,0%	77,0%	77,0%
79,0%	73,0%	76,0%	75,0%	74,0%
78,0%	72,0%	79,0%	72,0%	75,0%

D.4.1 Porovnání algoritmů na testovací sadě (ts008)

Tab. D.9: Porovnání klasifikace sady ts008 různých algoritmů.

ALG	**ACC**	**E**$_{rate}$	**TP**$_{rate}$	**FP**$_{rate}$	**PRE**	**REC**	**SPE**	**F**$_m$	**G**$_m$	**MCC**
NaiveBayes	45,00%	55,00%	0,4500	0,0786	0,6331	0,4500	0,9214	**0,4325**	**0,7378**	**0,4215**
OneR	53,75%	46,25%	0,5375	0,0661	0,5089	0,5375	0,9339	**0,5023**	**0,6255**	**0,3348**
DecisionTable	72,50%	27,50%	0,7250	0,0393	0,8379	0,7250	0,9607	**0,7412**	**0,8931**	**0,7295**
IBk	80,00%	20,00%	0,8000	0,0286	0,8009	0,8000	0,9714	**0,7899**	**0,8792**	**0,7686**
HoeffdingTree	88,75%	11,25%	0,8875	0,0161	0,9113	0,8875	0,9839	**0,8764**	**0,9458**	**0,8747**
KStar	90,00%	10,00%	0,9000	0,0143	0,9196	0,9000	0,9857	**0,8975**	**0,9512**	**0,8912**
RandomTree	92,50%	7,50%	0,9250	0,0107	0,9292	0,9250	0,9893	**0,9247**	**0,9579**	**0,9156**
JRip	95,00%	5,00%	0,9500	0,0071	0,9598	0,9500	0,9929	**0,9491**	**0,9756**	**0,9457**
BayesNet	95,00%	5,00%	0,9500	0,0071	0,9598	0,9500	0,9929	**0,9500**	**0,9756**	**0,9461**
LWL	96,25%	3,75%	0,9625	0,0054	0,9659	0,9625	0,9946	**0,9623**	**0,9798**	**0,9582**
REPTree	97,50%	2,50%	0,9750	0,0036	0,9792	0,9750	0,9964	**0,9747**	**0,9875**	**0,9727**
J48	98,75%	1,25%	0,9875	0,0018	0,9886	0,9875	0,9982	**0,9875**	**0,9933**	**0,9861**
PART	98,75%	1,25%	0,9875	0,0018	0,9886	0,9875	0,9982	**0,9875**	**0,9933**	**0,9861**
SimpleLogistic	96,25%	3,75%	0,9625	0,0054	0,9659	0,9625	0,9946	**0,9623**	**0,9798**	**0,9582**
LogitBoost	97,50%	2,50%	0,9750	0,0036	0,9761	0,9750	0,9964	**0,9750**	**0,9861**	**0,9718**
Bagging J48	98,75%	1,25%	0,9875	0,0018	0,9886	0,9875	0,9982	**0,9875**	**0,9933**	**0,9861**
AdaBoostM1 J48	98,75%	1,25%	0,9875	0,0018	0,9886	0,9875	0,9982	**0,9875**	**0,9933**	**0,9861**
RandomCommittee	100,00%	0,00%	1,0000	0,0000	1,0000	1,0000	1,0000	**1,0000**	**1,0000**	**1,0000**
MLPweka	53,75%	46,25%	0,5375	0,0661	0,4956	0,5375	0,9339	**0,4716**	**0,6231**	**0,4394**
MLPv1	86,25%	13,75%	0,8625	0,0196	0,8928	0,8625	0,9804	**0,8587**	**0,9341**	**0,8514**
ANNv2	100,00%	0,00%	1,0000	0,0000	1,0000	1,0000	1,0000	**1,0000**	**1,0000**	**1,0000**

E OBSAH PŘILOŽENÉ SD KARTY

Přehled adresářů a jejich obsahu:

- *Beekeeper* – zdrojové kódy systému Beekeeper
- *data* – data s útoky, datové sady pro učení klasifikátorů, analýza sad
- *results* – výsledky testování struktur neuronových sítí a jejich konfigurace, analýzy algoritmů strojového učení
- *thesis* – elektronická verze dizertační práce

Obrazy všech detekčních sond nejsou na přiloženém médiu vzhledem ke své velikosti k dispozici. Odkazy ke stažení lze nalézt na projektových stránkách `https://bitbucket.org/jsafarik/beekeeper/overview`. Projektové stránky dále obsahují kompletní zdrojový kód systému Beekeeper a podpůrných skriptů.