Netfox Detective - Identifikace aplikačních protokolů pomocí algoritmů strojového učení

Netfox Detective AppIdent

Technická zpráva FIT VUT v Brně

Jan Pluskal, Ondrej Lichtner, Ondřej Ryšavý

Technická zpráva č. FIT-TR-2017-05
Fakulta informačních technologií, Vysoké učení technické v Brně

Naposledy změněno: 21. července 2020

Netfox Detective - Identifikace aplikačních protokolů pomocí algoritmů strojového učení

Jan Pluskal, Ondrej Lichtner, Ondřej Ryšavý

Vysoké učení technické v Brně, email: ipluskal@fit.vutbr.cz

Abstrakt Klasifikace síťového provozu je naprosto nezbytná pro monitorování sítě, bezpečnostní analýzu a digitální forenzní vědu. Bez přesné klasifikace síťového provozu jsou výpočetní nároky na analýzu všech toků IP obrovské. Klasifikace může také snížit počet toků, které je nutné analyzovat a seřadit dle priority, aby vyšetřovatel analyzoval první forenzně nejdůležitější tok. Tato technická zpráva představuje automatickou metodu eliminace příznaků založenou na korelační matici příznaků. Dále porovnáváme dva algoritmy převzaté z literatury, které nabízí vysokou přesnost a přijatelný výkon s naším algoritmem - Vylepšenou identifikací statistického protokolu (ESPI). Každý z těchto algoritmů je používán s podmnožinou příznaků, které mu nejlépe vyhovují. Posuzujeme tyto algoritmy z hlediska jejich schopnosti identifikovat protokoly aplikační vrstvy a dodatečně samotné aplikace. Experimenty ukazují, že klasifikátor založený na metodě Náhodného lesa přináší nejslibnější výsledky, zatímco náš algoritmus poskytuje zajímavý kompromis mezi vyššími výkony a mírně nižší přesností.

1 Úvod

Klasifikace síťového provozu je užitečnou technikou pro monitorování sítě, bezpečnostní analýzu a digitální forenzní vědy. V digitální forenzní vědě mohou být typy souborů identifikovány příponou souborů nebo vyhledáváním takzvaných magických čísel na začátku souborů; známé soubory mohou být také identifikovány databázemi hashovaných hodnot. Identifikace typů souborů a filtrování známých souborů pomáhá snížit množství dat, které je třeba analyzovat.

Aplikovat totéž na síťový provoz je však složitější vzhledem k tomu, že každý přenos dat obsahuje specifické a dočasné charakteristiky, které mohou záviset na stavu sítě, využití sítě a umístění koncových bodů komunikace. Zachycení provozu mezi dvěma zařízeními umístěnými v hlavní síti se bude lišit charakteristikami, jako třeba kabelově připojené zařízení, mobilním zařízením s LTE či připojeným přes satelit. Správná klasifikace síťové komunikace pomáhá automatizovaným analyzátorům určit parser aplikačního protokolu, který se použije pro extrakci informací přenášených klasifikovanými IP toky[1], a naopak, pomáhá urychlit analýzu tím, že sníží počet neklasifikovaných IP toků.

[1] IP tok je paketová sekvence identifikovaná párem stejných zdrojových a cílových IP adres, portů transportní vrstvy a typem transportního protokolu.

Tradiční metody klasifikace provozu určují aplikace na základě použitých TCP nebo UDP portů. Tímto je dosaženo pouze omezené přesnosti (60-80 %), protože mnoho aplikací používá náhodné nebo nestandardní porty [3, 25], např. peer-to-peer aplikace, multimediální streamingové aplikace, počítačové hry nebo tunelovaný provoz. Pokročilá klasifikace provozu využívá metod strojového učení s učitelem (ML) založené na analýze payloadu, statistických metodách a hybridních přístupech [18, 20, 27, 28, 31].

Každá technika má své výhody a nevýhody. Například analýza šifrované komunikace založená na analýze payloadu je nepřijatelně nepřesná. Statistické a hybridní přístupy ukazují, že se nemusíme spoléhat pouze na obsah paketů [5, 14, 22], ale je možné kombinovat strukturální a behaviorální příznaky ke zvýšení přesnosti klasifikace [17].

Metody strojového učení bez učitele můžou klasifikovat neznámý síťový provoz [10] do neznačených klastrů podle jejich blízké podobnosti. Na základě odborných znalostí vyšetřovatele a inspekce několika vzorků z klastru můžeme klasifikovat celý klastr.

Mnoho výzkumných pracovníků provádělo obsáhlé výzkumy ML přístupů ke klasifikaci provozu. Většina výzkumů představuje metody, které mají za cíl klasifikovat síťový provoz za účelem identifikace použitého protokolu aplikační vrstvy jako nástroje pro inteligentní síťové filtrování nebo sledování pro zajištění bezpečnosti. Přestože klasifikace provozu pro síťové forenzní vědy vychází ze stejných myšlenek, existují určité rozdíly:

- Síťová forenzní analýza může být provedena offline na zachycených datech. V tomto případě je přesnost důležitější než rychlost. Proto lze využít kombinaci několika metod nebo aplikací s pomalejšími, ale za to mnohem přesnějšími metodami.
- Vyšetřovatel může kompenzovat nesprávné výsledky provedením manuální kontroly a úprav. Například některé metody vrátí vektor pravděpodobností, který lze prozkoumat za účelem zvážení odlišných výsledků.
- Klasifikace musí být deterministická jak vyžadují forenzní zásady, aby byly všechny výsledky ověřitelné.
- Metody klasifikace mohou být vyladěny vyšetřovatelem a mohou být opakovány za užití různých sad parametrů pro zvýšení citlivosti na úkor klesající specificity, viz sekce 3.1 pro více podrobností.

Zatímco klasifikace provozu je předmětem intenzivního výzkumu pro použití v oblasti monitorování sítí či bezpečnostní analýzy, došlo k výrazně menšímu výzkumu klasifikace provozu pro síťové forenzní vědy.

Tato technická zpráva vznikla v rámci bezpečnostní výzvy ministerstva vnitra, projektu Tarzan – Integrovaná platforma pro zpracování digitálních dat z bezpečnostních incidentů[2], VI20172020062. Projekt je zaměřen na detekci a analýzu nových forem kybernetické kriminality v prostředí Internetu věcí, mobilních a komunikačních aplikací. Cílem projektu je výzkum nových metod založených na dolování dat, strojovém učení, vizuální analýze a vytvoření funkčního vzorku

[2] http://www.fit.vutbr.cz/~ipluskal/grants.php?id=1063

integrující tyto metody pro efektivní vyšetřování incidentů. Technická zpráva popisuje modul pro identifikaci aplikačních protokolů, který je přímou součástí nástroje Netfox Detective 2.0, jenž byl vyvinut taktéž v rámci tohoto projektu [29]. Oba výstupy jsou součástí hlavního výstupu – Integrovaná platforma pro zpracování digitálních dat z bezpečnostních incidentů. Modul AppIdent tvoří proof-of-concept řešení identifikace apliakčních protokolů pomocí metod strojového učení jenž jsou implementovatelné v prostředí forenzní platformy.

1.1 Současný stav

Algoritmy strojového učení pro klasifikaci síťového provozu jsou zkoumány již od roku 1990. Nejběžnější algoritmy zahrnují support vector machine [14], aloritmus rozhodovacího stromu [22] (Decision tree), a pravděpodobnostní [5] (Probabilistics) nebo statistické metody [17, 20] (Statistical), což jsou příklady metod učení s učitelem. V případě učení bez učitele se u přístupu shlukování, jmenovitě algoritmus K-means [10], seskupí provoz podle jeho významných vlastností. Pokud je *vektor příznaků* řádně vybrán, pak mohou ML metody převyšovat přesnost 90% [27].

Průzkum klasifikačních metod založený na ML, jehož autorem je Nguyen [28] a další, obsahující aktuálnější výsledky od Namdev [27], poskytují obecný úvod do problematiky identifikace protokolů. Metody klasifikace šifrované komunikace byly revidovány v [31]. Nedávno, Al Khater a Overill [1] navrhli prozkoumat použití různých algoritmů strojového učení ke zdokonalení metod klasifikace provozu pro digitální forenzní vědy. Foroushani a Nur Zincir-Heywood [11] prokázali možnost identifikovat aplikační protokol za použití vysokoúrovňových charakteristik i ze šifrované komunikace několika síťových služeb. Dai [6] a Miskovic a spol. [24] popsali metodu pro získání otisku mobilních aplikací z jejich komunikace. Erman a spol. [8] prozkoumal klasifikační přístup založený na průtoku a navrhl klasifikační kombinovanou metodu učení s učitelem a bez (semi-supervised learning), která může identifikovat jak známé, tak neznámé aplikace.

1.2 Přínos a osnova technické zprávy

V této technické zprávě poskytujeme několik klíčových příspěvků do oblasti síťové forenzní vědy. Prvním úkolem bylo vytvořit vhodnou množinu dat, která by nám poskytla prostředky pro spolehlivé získání základní pravdy pro naše experimenty. Obvykle dostupné datové sady používají informace odvozené od *l7-filter* [30] nebo *nmap* [2], a proto poskytují pouze určitou aproximaci této informace. Shang a Huang [30] ukazují, že v těchto případech může být přesnost metod vždy 1 (tedy klasifikace neobsahuje žádné negativně pozitivní výsledky). Ale recall se pohybuje okolo $0.67 - 0.87$. To znamená, že $13 - 32\%$ vzorků nebylo označeno a výzkumníci je museli vyloučit z datové sady, protože jim chyběla označení [2, 14]. Zbývající datová sada je proto již klasifikovatelná pomocí technik DPI a méně relevantní k nalezení lepších metod klasifikace. V ostatních případech vědci nezmiňují žádné informace o datech použitých pro experimenty

nebo jsou popisy nejasné a nereprodukovatelné [30], či nepopisují, jak získání základní pravdy docílili [5].

Z těchto důvodů v sekci 2 popisujeme, jak jsme vytvořili anotovaný dataset použitý pro naše experimenty během jednoho týdne v laboratorním prostředí s osmi počítači. Dataset obsahoval zhruba 20 GB zachycených dat. Tato data byla automaticky označena úplnými informacemi o aplikaci, která komunikovala. Dataset je veřejně dostupný a je k dispozici ostatním výzkumníkům[3].

Představujeme *Enhanced Statistical Identification Protocol* (ESPI) metodu, což je klasifikátor založený na ML, používající statistické metody, které jsme vytvořili. Na základě výsledků získaných ze souvisejících studií jsme pro srovnání zvolili dva další klasifikátory, zejména Bayesian Network a Náhodný les. V sekci 3 popisujeme všechny tři metody, jejich použití pro vytvoření klasifikátorů schopných identifikovat protokoly aplikační vrstvy a obvykle i aplikace, které tyto protokoly využívaly. To je důležité s ohledem na to, že identifikace aplikací nám poskytuje více informací o síťovém provozu ve srovnání s tím, jaké informace lze získat pouze z identifikovaných aplikačních vrstev. Například HTTPS může být použito nejenom k zabezpečnému prohlížení webu, ale také i k vytvoření šifrovaného tunelu. Nástroj schopný rozeznávat aplikace, jako Google Drive, iTunes, a OneDrive, ze síťového provozu místo pouhé informace o aplikačním protokolu, tedy HTTPS, může být užitečný pro různé aplikační domény. Ve forenzní analýze může identifikace aplikací více redukovat množství dat, která musejí být analyzována ve srovnání s konvenčními přístupy.

Před aplikací metody ML na naše datové sady je nutné definovat vhodné příznaky, na jejichž základě se bude klasifikátor rozhodovat. V sekci 3.1 popisujeme, jak jsme začali s vektorem příznaků založených na diskriminátorech [26] navržených Moorem. Z tohoto vektoru příznaků jsme vybrali pouze ty invariantní příznaky síťové linky, tj. rychlost linky, latence a jitter. Navíc jsme modifikovali zbývající příznaky pro použití s aplikačními konverzacemi (L7 konverzace) a aproximacemi aplikačních zpráv. To vede ke snížení šumu při extrakci příznaků a dosažení přesnějších dat oproti základním paketovým přístupům. Vzhledem k tomu, že výsledný vektor příznaků je stále dost velká a složitá, vyvinuli jsme také konfigurovatelnou automatickou metodu eliminace příznaků na základě korelační matice, která je popsána v sekci 3.1. Zmenšení velikosti vektoru příznaků na základě ortogonality příznaků může zlepšit rychlost i přesnost použitých algoritmů strojového učení a zabraňuje přeučení.

V sekci 4 detailně popisujeme experimentální prostředí, zdroje dat, porovnáváme všechny tři algoritmy a revidujeme výsledky. Konečně, článek uzavírá sekce 5 obsahující shrnutí výsledků a navržení dalších směrů pro budoucí výzkum.

2 Sběr dat a předzpracování

Klasifikační metody síťového provozu obvykle používají pouze zachycená data síťového provozu jako vstup, nejčastěji ve formátu souboru PCAP. Zachycená

[3] http://nes.fit.vutbr.cz/AppIdent/

komunikace je pak rozdělena na sadu konverzací na úrovni L4, reprezentovaných jedním nebo dvěma IP toky (jednosměrná nebo obousměrná komunikace). Pro naše experimenty jsme připravili anotovaný dataset zachycený nástrojem Microsoft Network Monitor, který poskytuje informaci o aplikaci, která na síti komunikovala pro téměř všechny konverzace. Soubor dat představuje běžný síťový provoz osmi uživatelských stanic s operačním systémem Windows. Konečný dataset má následující charakteristiky[4]:

- Velikost PCAP souboru: $19.5GB$
- PCAP Formát: *Microsoft NetMon 2.x*
- Délka zachycení: $119h$
- Počet paketů: $276,161,38$
- Počet L7 konverzací: $269,459$
- Počet aplikačních protokolů: 58
- Počet komunikujících aplikací: 93

Před použitím zachyceného souboru vytvořeného pro naše experimenty jsme přidali další kroky předzpracování z naší předchozí práce, které byly použity ke zlepšení postupů k extrakci dat [23]. Konečný krok předzpracování použil jedno kolo experimentů metodou klasifikace ESPI. Z těchto prvních výsledků jsme manuálně vytvořili druhou instanci stejné datové sady, tentokrát již obsahující anotace o používaných aplikačních protokolech, které jsme založili na naší manuální, hierarchicky klastrované analýze výsledků[5].

Tyto kroky předzpracování mohou zvýšit přesnost klasifikace komunikace snížením šumu v extrahovaných prvcích způsobeným jedním nebo více následujícími faktory:

- Důležité informace o řízení relace TCP pravděpodobně chybí, tj. synchronizační nebo finalizační packety nejsou přítomné.
- Při dlouhých konverzacích TCP mohou sekvenční čísla přetéct, což lze nesprávně interpretovat a rozdělit konverzaci na dvě, nebo spojit nesouvisející IP toky do jediné konverzace.
- Spojení zachycených souborů z více snímacích sond musí řešit problémy s případnou duplikací paketů a řádným uspořádáním paketů patřící ke stejné konverzaci.
- Některé IP pakety mohou chybět nebo být duplikovány, např. v případě protokolu TCP retransmisí.
- Konečně, je důležité správně spárovat přidružené IP toky do obousměrné konverzace.

Ukázali jsme [23], že ostatní zkoumané nástroje pro síťové forenzní zpracování neřeší tyto otázky efektivně, což nás vede k předpokladu, že přijetí navrhovaných

[4] Pro více informací viz https://github.com/pluskal/AppIdent/blob/gh-pages/Pages/captureStats.pdf. Dataset je dostupný na http://nes.fit.vutbr.cz/AppIdent

[5] Specifické kroky lze naleznout v provedené implementaci.

dodatečných kroků by také bylo prospěšné v kontextu oblasti klasifikace síťového provozu. K řešení těchto problémů jsme použili příznaků nástroje Netfox Detective[6] (nástroje, který jsme vyvinuli pro užití v těchto specifických případech) ke zpracování zachycených PCAP souborů.

2.1 Konverzace na aplikační úrovni a aproximace aplikačních zpráv

Kromě řešení základních problémů při zpracování L4 konverzací, Netfox Detektiv také umožňuje zpracovávat náš datový soubor pro segregaci L7 konverzací a aproximovat individuální aplikační zprávy. To může pomoci zvýšit přesnost klasifikace pomocí identifikace vzorců komunikace daných aplikací. To také eliminuje zbytky fragmentace síťových paketů na internetové vrstvě a TCP retransmisí na vrstvě transportní. Fragmentace paketů a segmentace TCP jsou nezávislé na aplikačních komunikačních schématech, a tak můžou negativně ovlivnit klasifikaci.

Aplikační zpráva je identifikována ve znovu sestaveném proudu dat v závislosti na transportním protokolu podle následujících pravidel:

- Pokud proud využívá přenosový protokol UDP, pak celá užitečná zátěž každého UDP datagramu se považuje za jednu aplikační zprávu.
- Pro transportní protokol TCP jsou segmenty rozděleny na aplikační zprávy založené na paketech s příznaky PSH, RST nebo FIN, nebo na základě časových limitů.

Tato pravidla jsou jednoduše implementovatelná a pro většinu aplikací poskytují přesnou aproximaci aplikačních zpráv.

3 Metody klasifikace

Použití ML algoritmů pro klasifikaci síťového provozu není nový koncept v oblasti síťové forenzní vědy, avšak obvykle jde o případ identifikace použitého aplikačního protokolu [28, 31]. V našem výzkumu jsme rozšířili tento přístup také o schopnost identifikovat aplikaci, která síťový provoz vytvořila. Takto lze získat více informací, které může vyšetřovatel použít ke snazší a přesnější analýze.

V této sekci přezkoumáme obvykle používaný vektor příznaků [17, 20, 26], aby lépe vyhovovaly našim potřebám, a představíme metodu eliminace příznaků založenou na korelaci příznaků za účelem zlepšení přesnosti vytvořených klasifikátorů. Nakonec popíšeme ESPI – klasifikátor založený na statistických metodách, který jsme vytvořili, a na dalších dvou metodách klasifikace, které, na základě souvisejících článků, poskytují slibné výsledky pro identifikace provozu.

[6] https://github.com/nesfit/NetfoxDetective

3.1 Vektor příznaků

Kvalita vektoru příznaků přímo ovlivňuje přesnost klasifikace [33]. Běžně používané příznaky provozu klasifikace se vztahují k významným aspektům paketové komunikace a síťové architektury. Čísla portů, typ přenosového protokoly, počáteční sekvence bajtů payloadu, výskyty vzoru, délky zprávy a načasování zpráv jsou často používány. Ze souvisejících článků [17, 20, 26] jsme identifikovali seznam možných příznaků sestávající z 92 položek, které jsou invariantní k charakteristikám síťové komunikace[7].

ML algoritmy dosahují nejlepšího výkonu, pokud jsou vybrané funkce ortogonální, tj. pokud mezi nimi neexistuje žádná korelace [16]. Existuje několik přístupů, jak vypočítat korelaci mezi příznaky, např. Pearson, Spearman, Kendall korelační vzorce [34] nebo kovarianční matice [15]. V tomto článku jsme se rozhodli použít metodu kovarianční matice pro její snadnou implementaci.

Kovarianční matice ukazuje korelační hodnoty mezi každou dvojicí příznaků. Na základě této matice jsme navrhli jednoduchý automatický postup pro eliminaci příznaků pomocí dvou parametrizovaných kroků. Nejprve je kovarianční matice vypočtena na základě zvoleného poměru tréninkových vzorků k ověřovacím (t/v). Následně, na základě maximální povolené korelační hodnoty, hledáme páry prvků s vyššími korelačními hodnotami a odstraníme příznak, která je, v průměru, více korelovaná se všemi ostatními příznaky. Vytvořený vektor příznaků je pak použit zvolenou klasifikační metodou.

Dle našich měření více než 80% dvojic příznaků, které jsme původně navrhli, ukázaly korelační hodnoty $0,5$ nebo vyšší. V tabulce1 ukazujeme dva různé poměry t/v a jak by zvolený vektor příznaků vypadal na základě různých hodnot přijaté korelační hodnoty až do výše $0,5$. Většina zobrazených příznaků popisuje charakteristiky toků spíše než jednotlivých paketů, což potvrzuje náš předpoklad, že z hlediska lepších výsledků pro šifrovaný nebo méně strukturovaný provoz je vhodné se méně spoléhat na otisk nebo nějaký specifický vzor v obsahu paketů.

3.2 Vylepšená identifikace statistického protokolu

Identifikace statistického protokolu (SPID) [17] byla původně vyvinuta Erikem Hjelmvikem za účelem identifikace aplikačních protokolů pro nástroj NetworkMiner [8]. Fáze učení metody vytváří databázi protokolových otisků, které lze pak použít k identifikaci aplikačních protokolů. Vektor příznaků používaný v SPID se nazývá naměřený atribut protokolu a každá položka představuje jiný druh informace. Některé položky mohou představovat skalární hodnoty, např. velikost dat payloadu, počet paketů v relaci nebo číslo portu. Dalšími položkami mohou být kompozitní hodnoty, například násobek sestávající ze směru paketů, uspořádání paketů, velikost paketu, frekvence bytové hodnoty. Původní implementace SPID používá asi 35 různých naměřených atributů protokolu a extrahuje informace z

[7] Tento seznam lze nalézt v odkazu implementace https://github.com/pluskal/AppIdent/blob/gh-pages/Pages/allFeatures.pdf

[8] http://www.netresec.com/?page=NetworkMiner

Korelace	Příznak pro poměr 0,1 t/v	Příznak pro poměr 0,2 t/v
	BytePairsReoccuringDownFlow	
	DirectionChanges	
	First3BytesEqualDownFlow	First3BytesEqualDownFlow
	FirstBitPositionUpFlow	FirstBitPositionUpFlow
	FirstPayloadSize	
	MinInterArrivalTimeDownFlow	
	MinInterArrivalTimePacketsUpAnd DownFlow	MinInterArrivalTimePacketsUpAnd DownFlow
	MinPacketLengthDownFlow	MinPacketLengthDownFlow
	NumberOfBytesDownFlow	
	NumberOfPacketsUpFlow	
	PacketLengthDistributionDownFlow	PacketLengthDistributionDownFlow
	PacketLengthDistributionUpFlow	
		ThirdQuartileInterArrivalTimeUp
		ByteFrequencyUpFlow
		MaxSegmentSizeDown
		MaxSegmentSizeUp
		MinInterArrivalTimePacketsUpFlow
		NumberOfBytesUpFlow
		ThirdQuartileInterArrivalTimeDown
<0.25	PUSHPacketsDown	PUSHPacketsDown
	ThirdQuartileInterArrivalTimeDown	
		NumberOfBytesUpFlow
<0.3		FirstPayloadSize
	ByteFrequencyUpFlow	
	MinPacketLengthUpFlow	MinPacketLengthUpFlow
	NumberOfPacketsPerTimeUp	
		DirectionChanges
		BytePairsReoccuringDownFlow
<0.4		MeanPacketLengthUpFlow
<0.5	MeanPacketLengthUpFlow	

Tabulka 1: Vyčíslení příznaků zbývajících po vyloučení příznaků na zkušebním data vzorku 0.1 a 0.2 pro ověření poměru t/v. Korelační sloupec zobrazuje maximální přípustnou korelační hodnotu příznaků uvedených na tomto a vyšším řádku. Tyto vektory příznaků byly použity pro klasifikaci s klasifikátory Bayesian a Náhodný les. Optimální výběr funkcí má velký vliv na přesnost klasifikace, jak ukazují experimenty, viz tabulka 2 popis experimentu, obrázek 1 pro grafické porovnání a detaily výsledků na obrázku 3.

prvních několika paketů IP toků k dosažení lepší rychlosti ve srovnání s jinými klasifikačními metodami, které analyzují celý tok IP po jeho ukončení. Pro výpočet vzdálenosti analyzovaných dat ke známým protokolovým otiskům používá Kullback-Leibler (KL) divergence, přičemž nejlepším odpovídajícím protokolovým otiskem je ten, který má nejmenší součet KL-odchylek pro všechny atributy. Později, Kohnen a spol. [20] vyvinuli novou verzi SPID algoritmu přidáním podpory pro UDP a zaměřili se na streamingové protokoly, které užívají jinou sadu naměřených atributů protokolu.

V našem výzkumu jsme tuto koncepci dále upravili a vytvořili klasifikátor založený na SPID, který nazýváme Vylepšená statistická identifikace protokolu – Enhanced Statistical Probability Identification (ESPI). Zaprvé, v kontextu forenzního vyšetřování, máme větší zájem o přesnost identifikace než o rychlost (i když rychlejší identifikace je stále důležitá), takže budeme k analýze využívat úplné konverzace namísto pouhých prvních pár paketů. Dále, jak bylo zmíněno dříve, nesnažíme se pouze identifikovat aplikační protokoly, ale aplikace samotné, takže místo toho analyzujeme aproximované aplikační zprávy místo jednotlivých paketů. Kromě těchto změn používá ESPI jiný vektor příznaků (92 funkcí, které jsme vybrali, jak je popsáno v 3.1) a odlišnou metodu pro kalkulaci vzdálenosti měřených hodnot k otiskům protokolu učení. Každý příznak je spojen s:

- funkcí f, která vyhodnotí odchylku měřené hodnoty k otiskové hodnotě,
- funkcí g, která vrátí hodnotu normalizované funkce ze skutečně naměřené hodnoty,
- funkcí w, která vrátí váhu funkce pro otisk protokolu.

Odchylka ze zjištěných otisků je založena na Eukleidovské metrice [7] vážených odchylek pro jednotlivé funkce, jak je vidět v rovnici 1, kde $x_1, ..., x_n$ představuje hodnoty pro protokol, $c_1, ..., c_n$ představuje normalizované hodnoty v protokolovém otisku a $w_i(c)$ představuje váhu funkce i-th v otisku protokolu c.

$$d_{x,c} = \sqrt{\sum_{i=0}^{n} \left(w_i(c) \cdot f_i(g_i(x_i), c_i)\right)^2} \tag{1}$$

Použitím tohoto vzorce počítáme rozdíly d_{x,c^j} pro každý protokolový otisk c^j a identifikujeme protokol nebo aplikaci k jako $d_{x,c^k} = min(d_{x,c^1} \dots d_{x,c^m})$.

Ve srovnání s jinými metodami ML ESPI netrpí přeučením ve vztahu k použití souvisejících funkcí, protože ESPI přiřazuje váhy k jednotlivým funkcím. Tato vlastnost dělá ESPI snadno použitelnou pro klasifikaci nových protokolů a zavedení nových příznaků, které jsou pro tyto protokoly jedinečné, ale mohou být v korelaci s dalšími příznaky jiných protokolů.

3.3 Klasifikátor Bayesovské sítě

Algoritmus, který řídí Bayesovský síťový klasifikátor [13] vychází z Bayesovy věty, která definuje pravděpodobnost události s ohledem na podmínky týkající se výskytu události. Klasifikátor je složený z Bayesovských důvěrných sítí,

které jsou postaveny během fáze učení. Bayesovská síť je řízený acyklický graf a soubor podmíněných pravděpodobnostních tabulek. Uzly představují proměnné příznaků a hrany představují podmíněné závislosti. Pravděpodobnostní tabulky poskytují pravděpodobné funkce pro uzly. Aplikační protokol je identifikován vyhledáním uzlu (nebo souboru uzlů), který má pro dané hodnoty vstupní funkce nejvyšší pravděpodobnost. Výhodou tohoto klasifikátoru je, že také vypočítá pravděpodobnost konverzace patřící do identifikované třídy. Tyto informace mohou být užitečné pro vyšetřovatele, aby rozhodl, zda tuto konverzaci dále analyzuje, nebo ne.

3.4 Klasifikátor Náhodný les

Náhodný les je, v kontextu této práce, souborová metoda sestavující více rozhodovacích stromů C4.5 v *trénovací fázi*, které jsou použity pro klasifikaci v *ověřovací fázi*, kde je režim částečných výsledků vybrán jako výsledná třída [4]. To činí Náhodný les náchylnou k přeučení [12]. Klasifikátory Náhodného lesa jsou parametrizovány více proměnnými, např. počet lesů, spojení a trénink v poměru k ověření. Optimální hodnoty použité pro tyto parametry lze nalézt pomocí křížové validace a výpočtem out-of-bag-error (OOB) za účelem odhadu výkonu kombinace určitých parametrů. Protože algoritmus počítá OOB chybu, není třeba mít samostatná ověřovací data. Proto lze algoritmus učit na celé datové množině. Naše pokusy nicméně prokázaly, že tento přístup je výpočetně velmi nákladný.

4 Experimenty a výsledky

V této sekci představujeme návrh a výsledky našich experimentů se všemi třemi klasifikačními metodami popsanými v sekci 3. Experimenty jsme navrhli při sledování třech cílů. Prvním cílem bylo porovnat výsledky získané pomocí strojového učení a statistických metod, které sdílejí stejný vektor základních příznaků, ale mají zásadně odlišné přístupy ke klasifikaci. Druhým cílem bylo sledovat velikost tréninkové sady a jak eliminační poměr příznaků ovlivňuje přesnost identifikace aplikačního protokolu a klasifikace aplikací. Konečně, třetím cílem bylo dokázat, že pomocí klasifikátorů aplikací můžeme klasifikovat síťovou komunikaci založenou na síťových aplikací, které ji vygenerovaly.

Použili jsme nástroj *Netfox Detective* jako middleware pro analýzu a zpracování zachyceného provozu do konverzací a aplikačních zpráv. Algoritmus vylučování příznaků a klasifikační metody byly všechny implementované jako moduly nástroje Netfox Detective pro snadnou integraci se vstupními daty. Samostatná aplikace byla použita k automatizaci experimentů s různými parametry. Metoda ESPI byla implementována kompletně v naší režii. Klasifikátory Náhodného lesa a Bayesovský síťový klasifikátor byly implementovány za užití knihovny Accord.NET[9] obsahující algoritmy strojového učení.

[9] http://accord-framework.net/

4.1 Schéma experimentů

Jak již bylo zmíněno, používali jsme nástroj Netfox Detective pro analýzu a zpracování zachyceného provozu a extrakci úplné sady hodnot funkcí pro výsledné konverzace (vektory funkcí). Vektory všech prvků byly anotovány (pomocí základní pravdy získané z původního souboru se zachycenou komunikací) se značením závislým na úrovni klasifikace:

1. Identifikace aplikace – zastoupena jako entice *typu transportního protokolu*, *cílového portu relační vrstvy* nebo *manuálně udělené označení*, a *informace o aplikačním procesu*, např. *tcp_ http_ skypeexe*.
2. Identifikace aplikačního protokolu – zastoupena jako entice *typu transportního protokolu* a *portu cílové relační vrstvy* nebo *manuálně udělené označení*, např. *tcp_ http*.

Protože se jedná o časově náročnou operaci a je třeba ji provést pouze jednou, uložili jsme výsledky jako samostatný binární soubor. Odtud jsme použili specificky vytvořenou aplikaci pro automatizaci běhu stejného experimentu s různými hodnotami konfiguračních parametrů (metoda klasifikace, učení k poměru ověření a přijatá korelační hodnota pro vyloučení příznaků). Všechny experimenty probíhaly stejným postupem podle následujících kroků:

1. Zdroj dat byl rozdělen na dvě nesourodé datové sady podle parametru učení k poměru ověření. První datová sada byla použita pro učení, druhá pro ověření.
2. [Eliminace příznaků] Pro experimenty s použitím metod klasifikátoru Bayesovské sítě a klasifikátoru Náhodného lesa, data pro učení z kroku 1 byla užita spolu s algoritmem eliminace příznaků popsaného v sekci3.1. Všechny experimenty s metodou ESPI používaly akceptovanou korelační hodnotu 1, aby byly zahrnuty všechny příznaky, protože metoda ESPI nevyžaduje eliminaci příznaků (vysvětleno v sekci 3.2).
3. [Fáze učení] Za užití tréningových dat z kroku 1, byl vytvořen klasifikátor a naučen:
   (a) *ESPI* – Pro každou skupinu vektorů příznaku stejného označení byl vypočtený otisk aplikačního protokolu s užitím funkce g.
   (b) *Bayesovský síťový klasifikátor* – Pro každou skupinu vektorů příznaku se stejným označením byl naučen Bayesovský síťový klasifikátor.
   (c) *Náhodný les* – Byly nalezeny optimální parametry (jak již bylo zmíněno v sekci 3.4), které vedly k získání nejpřesnějšího modelu. K určení nejlepšího modelu byla použita křížová validace.
4. [Fáze verifikace] Klasifikátory z předchozích kroků jsou použity ke klasifikaci každé konverzace z ověřovací datové sady pomocí:
   (a) Vektoru vzdáleností nebo pravděpodobností pod vícero označením. Vektor příznaků je následně vyžádán a je vybráno označení s nejmenší vzdáleností nebo nejvyšší pravděpodobností. Toto je relevantní k:
      i. ESPI kde jsme vypočítali euklidovskou vzdálenost konverzace ke každé aplikaci nebo otisku aplikačního protokolu.

ii. Bayesovský síťový klasifikátor, kde každý Bayesovský klasifikátor nese pravděpodobnost současné konverzace patřící do třídy (aplikace nebo aplikační protokol) zastoupené tímto klasifikátorem.

(b) Jediné označení v případě klasifikátoru Náhodný les.

5. Označení bylo porovnáno s anotací a byly vypočteny statistické vlastnosti klasifikační metody.

4.2 Shrnutí výsledků

Pomocí automatizace jsme spustili mnoho experimentů s různými konfiguracemi parametrů s cílem nalézt konfigurace, které dosáhnou toho nejlepšího výsledku. Naše experimenty jsme uspořádali dle klasifikační metody a vybrali nejúspěšnější experimenty pro každou metodu s různými poměry trénovacích a testovacích dat pro lepší srovnání.

V tabulce2 jsme sepsali konfigurace experimentu, které jsou uvedeny v tomto shrnutí. Experimenty lze rozdělit do dvou Kategorií. Při pokusech B1, B2, B3, ESPI1, RF1 a RF2 byly použity klasifikátory pro identifikaci aplikačního protokolu, kde úplná datová sada obsahovala 58 tagů aplikačního protokolu. Na druhou stranu, při pokusech B4, B5, B6, ESPI2, RF3 a RF4 byly použity klasifikátory pro identifikaci aplikace, kdy úplný soubor dat obsahoval 93 tagů aplikací. Kompletní výsledky všech experimentů byly zveřejněny online[10]. Číselné hodnoty a tabulky v tomto souhrnu ukazují zkrácené výsledky těchto experimentů. Zkrácení bylo provedeno výběrem nejlepších experimentů v každé kategorii jako základní linie, ze které jsme vybrali nejvíce 20 přesně identifikovaných označení, které jsou zobrazeny pro všechny experimenty v kategorii.

Označení vrácená klasifikačními metodami byla porovnána se základní pravdou z původních zachycených dat a rozdělena do čtyř kategorií definovaných konfúzní maticí v sekci 3.

Pro vyhodnocení a porovnání různých metod jsme použili *F-skóre*, známý také jako *vyrovnané F-skóre* [16]. Toto jediné skóre představuje harmonický prostředek přesnosti a vyvolání a je vypočten jako rovnice 2, kde *přesnost* a *vyvolání* jsou vypočteny jako rovnice 3 a rovnice 4 s hodnotami získanými z konfúzních matric.

$$F = 2 \times \frac{presnost \times vyvolani}{presnost + vyvolani} \tag{2}$$

$$pesnost(M) = \frac{TP}{TP + FP} \tag{3}$$

$$vyvoln(M) = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{4}$$

[10] Kompletní výsledky pro klasifikaci protokolu lze nalézt na https://github.com/pluskal/AppIdent/blob/gh-pages/Pages/comparisonPA.pdf nebo pro *identifikaci aplikací* na https://github.com/pluskal/AppIdent/blob/gh-pages/Pages/comparisonPAAppTags.pdf.

Metoda	Označení experimentu	Poměr trénovacích a testovacích dat	Nejvyšší povolená korelace příznaků
Bayesovský klasifikátor	B1	0.1	0.3
	B2	0.2	0.5
	B3	0.5	0.5
	B4	0.1	0.2
	B5	0.2	0.25
	B6	0.5	0.25
ESPI	ESPI 1	0.7	1
	ESPI 2	0.2	1
Náhodný les	RF1	0.1	0.4
	RF2	0.2	0.4
	RF3	0.1	0.5
	RF4	0.2	0.5

Tabulka 2: Konfigurace klasifikačních metod. Sloupec *Označení experimentu* uvádí seznam identifikátorů, které se používají v jiných grafických znázorněních a tabulkách, aby mohly být výsledky přiřazeny ke konkrétní konfiguraci experimentu. *Poměr trénovacích a testovacích dat* je poměr velikostí trénovací množiny ku množině testovací. A *nejvyšší povolená korelace příznaků* specifikuje hodnotu použitou algoritmem eliminace příznaků.

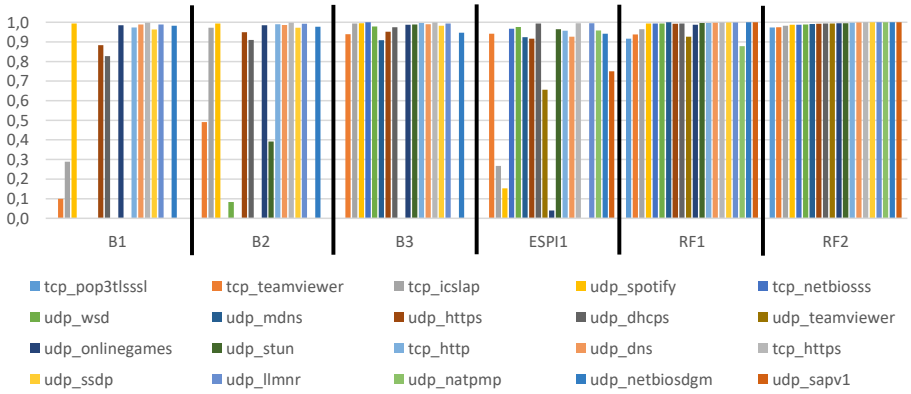
Základní pravda / **Výsledek klasifikace**	**Pozitivní**	**Negativní**	**Celkem**
Pozitivní	Skutečně pozitivní (TP)	Falešně pozitivní (FP)	P
Negativní	Falešně negativní (FN)	Skutečně pozitivní (TN)	N
Celkem	P‘	N‘	$P + N$

Tabulka 3: Konfúzní matice pro jediné označení (aplikace nebo aplikační protokol). Výsledky klasifikace jsou pozitivní, když klasifikátor odpoví, že konverzace může být označena tímto označením a negativní, když nemůže. Základní pravda je pozitivní, když je konverzace v datové sadě skutečně označena tímto označením a negativní, pokud tomu tak není.

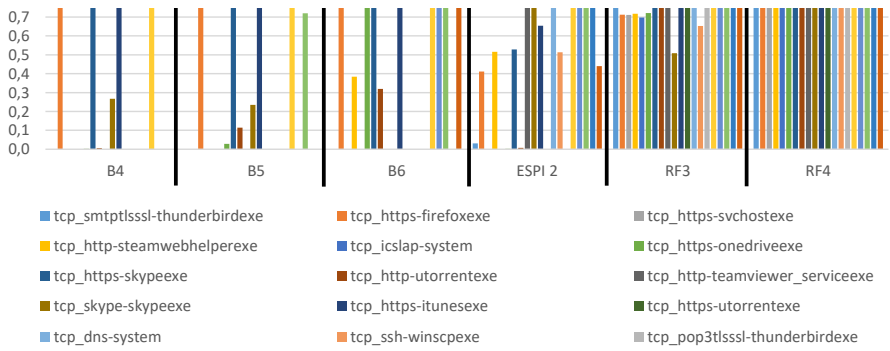
GreaterOrEqual F-Measure	B1	B2	B3	ESPI1	RF1	RF2	B4	B5	B6	ESPI2	RF3	RF4
0,0	58	58	58	58	58	58	93	93	93	93	93	93
0,1	21	19	23	33	47	51	22	25	36	43	83	83
0,2	16	18	23	31	45	47	22	23	34	40	77	77
0,3	14	18	22	29	41	45	20	22	34	37	74	75
0,4	14	16	22	29	40	43	19	22	30	36	68	70
0,5	14	14	22	28	37	41	19	22	29	31	63	63
0,6	13	14	22	26	36	39	16	20	27	27	54	58
0,7	12	13	21	24	34	37	15	17	26	22	45	47
0,8	11	12	19	21	32	36	13	13	26	20	38	41
0,9	8	12	18	17	26	31	7	12	15	17	25	28

Tabulka 4: Shrnutí výsledků klasifikačních metod. Čísla v buňkách zobrazují počet označení, která byla klasifikována pomocí F míry a jsou stejná nebo větší než hodnota ve sloupci F míry.

Obrázky 1 a 2 shrnují nejpřesnější výsledky klasifikace aplikací a aplikačního protokolu. Agregovaná statistika v tabulce 4 zahrnuje všechny třídy.



Obrázek 1: Výkon klasifikátorů aplikačních protokolů s užitím F míry. Názvy experimentů jsou vysvětlené v Table 2. Hodnoty F míry zobrazeny na ose y.



Obrázek 2: Výkon klasifikátorů aplikací s užitím F míry. Názvy experimentů jsou vysvětleny v Table 2. Hodnoty F míry zobrazeny na ose y.

Obrázek 1 zobrazuje identifikaci aplikačních protokolů. Můžeme pozorovat, že oba klasifikátory Náhodného lesa (RF1, RF2) jsou velmi přesné. Bayesovský klasifikátor (B3) také funguje velmi dobře, ale vyžaduje větší učící se soubor s učením v poměru k verifikaci 0,5 a více příznaků, viz tabulka 2.

Obrázek 2 poskytuje vizuální zobrazení výsledků pro klasifikaci aplikací. Můžeme pozorovat, že klasifikátory Náhodného lesa opět poskytují nejlepší výsledky.

Nicméně, v tomto případě Bayesovské klasifikátory překonal ESPI, což také ukazuje nejlepší kompromis mezi výkonností a přesností. Čas potřebný pro různé kroky experimentů, jako výběr příznaků, načítání dat, učení a zařazení je uvedeno na Obrázku 3 společně s porovnáním výkonu každé vyhodnocené metody.

AppProtocol	B1	B2	B3	ESPI1	RF1	RF2
Time [h]	1:01	1:08	1:13	0:50	2:41	13:21
tcp_pop3tlsssl	0,00	0,00	0,00	0,00	0,92	0,97
tcp_teamviewer	0,10	0,49	0,94	0,94	0,94	0,97
tcp_icslap	0,29	0,97	0,99	0,27	0,96	0,98
udp_spotify	0,99	0,99	1,00	0,15	0,99	0,99
tcp_netbiosss	0,00	0,00	1,00	0,97	0,99	0,99
udp_wsd	0,00	0,08	0,98	0,98	0,99	0,99
udp_mdns	0,00	0,00	0,91	0,92	1,00	0,99
udp_https	0,88	0,95	0,95	0,92	0,99	0,99
udp_dhcps	0,83	0,91	0,98	0,99	0,99	0,99
udp_teamviewer	0,00	0,00	0,00	0,66	0,93	0,99
udp_onlinegames	0,98	0,98	0,99	0,04	0,99	0,99
udp_stun	0,00	0,39	0,99	0,96	1,00	1,00
tcp_http	0,97	0,99	1,00	0,96	1,00	1,00
udp_dns	0,99	0,99	0,99	0,93	1,00	1,00
tcp_https	1,00	1,00	1,00	0,99	1,00	1,00
udp_ssdp	0,96	0,97	0,98	0,00	1,00	1,00
udp_llmnr	0,99	0,99	0,99	1,00	1,00	1,00
udp_natpmp	0,00	0,00	0,00	0,96	0,88	1,00
udp_netbiosdgm	0,98	0,98	0,95	0,94	1,00	1,00
udp_sapv1	0,00	0,00	0,00	0,75	1,00	1,00

(a) Klasifikátory aplikačních protokolů.

AppProtocol	B4	B5	B6	ESPI 2	RF3	RF4
Time [h]	0:53	1:03	2:00	1:11	20:13	23:20
tcp_smtptlsssl-thunderbirdexe	0,00	0,00	0,00	0,03	0,89	0,75
tcp_https-firefoxexe	0,88	0,93	0,91	0,41	0,71	0,77
tcp_https-svchostexe	0,00	0,00	0,00	0,00	0,71	0,77
tcp_http-steamwebhelperexe	0,00	0,00	0,38	0,52	0,72	0,79
tcp_icslap-system	0,00	0,00	0,00	0,00	0,70	0,81
tcp_https-onedriveexe	0,00	0,03	0,82	0,00	0,72	0,81
tcp_https-skypeexe	0,86	0,99	0,87	0,53	0,78	0,82
tcp_http-utorrentexe	0,01	0,11	0,32	0,01	0,84	0,83
tcp_http-teamviewer_serviceexe	0,00	0,00	0,00	0,87	0,88	0,86
tcp_skype-skypeexe	0,27	0,24	0,00	0,96	0,51	0,87
tcp_https-itunesexe	0,86	0,89	0,89	0,65	0,86	0,87
tcp_https-utorrentexe	0,00	0,00	0,00	0,00	0,92	0,89
tcp_dns-system	0,00	0,00	0,00	0,97	1,00	0,89
tcp_ssh-winscpexe	0,00	0,00	0,00	0,51	0,65	0,91
tcp_pop3tlsssl-thunderbirdexe	0,00	0,00	0,00	0,00	0,98	0,92
tcp_http-spotifyexe	0,93	0,91	0,93	0,90	0,93	0,93
tcp_tripe-spotifyexe	0,00	0,00	0,92	0,91	0,94	0,94
tcp_jabberssl-apsdaemonexe	0,00	0,72	0,81	0,91	0,94	0,95
tcp_jabber-pidginexe	0,00	0,00	0,00	0,97	0,94	0,97
tcp_netbiosss-system	0,00	0,00	0,90	0,44	0,98	0,99

(b) Klasifikátory aplikací.

Obrázek 3: Porovnání výkonnosti klasifikace experimentu pro top 20 se štítky založenými na nejúspěšnějším experimentu v kategorii. Názvy experimentů vysvětlené v Table 2. Políčka tabulky obsahují F míry. První řádek tabulky obsahuje celkový čas potřebný pro experiment.

5 Závěr

Tento příspěvek zkoumá různé aspekty aplikace metod strojového učení na problém s klasifikací síťového provozu při používání v síťové forenzní vědě. Konkrétně se zaměřujeme na identifikaci síťových aplikací vedle pouhých aplikačních protokolů. Z tohoto důvodu jsme vybrali příznaky, které charakterizují chování aplikace, jako je časování zpráv, délka obsahu, TCP flagy, namísto příznaků souvisejících s charakteristikami síťové linky. Kromě toho jsme také představili algoritmus pro eliminaci příznaků založený na korelaci s cílem dále zlepšit výsledky klasifikace. Tímto přístupem jsme vyvinuli náš klasifikační algoritmus založený na statistických principech s názvem *Vylepšená identifikace statistického protokolu - ESPI* a porovnali jsme jej s dalšími dvěma metodami klasifikace strojového učení v široké škále experimentů.

Předložené výsledky směřují k následujícím závěrům:

– Aplikace generující síťovou komunikaci lze s jistotou klasifikovat. Například služba NetBIOS nebo DNS byly identifikovány přesně, dále pak několik

běžných aplikací, které používají protokol HTTP(S), byly identifikovány s vysokou přesností. Stejně tak je možné rozlišit komunikační stopy aplikací OneDrive, Skype, iTunes, Spotify, Steam nebo μ Torrent, i když všechny používají stejný aplikační protokol (HTTPS).
- V našich experimentech získaly klasifikátory Náhodného lesa ty nejlepší výsledky, což souhlasí se závěry dalších výzkumných pracovníků experimentujících s různými ML algoritmy pro klasifikaci provozu [21, 32]. Při srovnání s metodou ESPI, kterou jsme vyvinuli, tato dosahuje lepších výsledků než Bayesovský síťový klasifikátor, zatímco je také mnohem rychlejší než obě metody - Bayesovská a metoda Náhodného lesa.
- Kromě klasifikace aplikačních protokolů jsme experimentovali s rozlišováním mezi aplikacemi používajícími stejný protokol. V tomto případě metoda Náhodného lesa vykazuje svou schopnost identifikovat většinu aplikací správně. Výsledky metody ESPI jsou ve srovnání s Bayesovským klasifikátorem přesnější a mnohem rychlejší než Bayesovská metoda a metoda Náhodného lesa.

Další výzkum se může odvíjet několika směry, zejména:

- Systematická analýza vektoru příznaků ke zlepšení přesnosti a robustnosti klasifikačních metod. Přesnost klasifikace je dána především kvalitou vybraných prvků. V prezentované práci jsme navrhli příznaky spíše na základě předchozích pozorování a intuice než uplatňování systematického přístupu.
- Zlepšit identifikaci aplikací ambivalentně pomocí stejného aplikačního protokolu, tj. odstraňování chyb, když *tcp_http_skypeexe* je klasifikován jako *tcp_http_firefoxexe*, nebo vice-versa, navrhujeme další výzkum hierarchických metod klasifikace. Příkladem toho může být hierarchické shlukování založené na otiscích protokolu ESPI. Vyšetřovatel by pak mohl odvodit skutečné třídy aplikací analýzou vizuálních klastrů. Tento přístup by se mohl rozšířit i na další úrovně, jako je např. úroveň jednotlivých aplikačních zpráv, což by mohlo zvýšit hodnotu pro síťovou forenzní vědu.
- Kombinování různých klasifikátorů [19] za účelem zvýšení důvěryhodnosti výsledků. I když se ke klasifikaci určitého druhu aplikačního provozu nejlépe hodí jedna metoda, kombinací několika klasifikátorů můžeme dosáhnout celkově přesnějších výsledků klasifikace.
- Výzkum klasifikačních metod kombinace učení s učitelem a bez učitele [9], které umožňují vytvářet modely z částečně označených dat. Protože síťová forenzní věda zahrnuje značné množství lidské práce, mohl by tohoto využít vyšetřovatel a navádět tak klasifikátor během fáze učení.
- Další experimenty na rozšíření klasifikačních modelů a k vyhodnocení vlastností jiných datových sad. Metody klasifikace v tomto článku vyžadují přesně vytvořené modely. Tvorba těchto modelů vyžaduje analýzu velkého počtu provozních vzorků. Experimentování s různými datovými sadami by poskytlo přesnější klasifikační modely a další údaje o vlastnostech jednotlivých klasifikačních metod.

Referenční implementace je dostupná pod licencí MIT naGitHub: https://pluskal.github.io/AppIdent/. Tato zahrnuje implementaci rámce

pro analýzu zachycených dat, extraktory příznaků, algoritmus jejich eliminace na základě korelace, všechny tři klasifikátory použité v této technické zprávě a samostatnou aplikaci implementující experimenty. Datový soubor je k dispozici na http://nes.fit.vutbr.cz/AppIdent. Tuto implementaci poskytujeme pro snadnou reprodukovatelnost našich experimentů a jako možnou benchmarking platformu pro testování dalších pokusů s ML aplikační identifikací.

Odkazy

[1] Noora Al Khater a Richard E Overill. „Forensic Network Traffic Analysis“. In: *Proceedings of The Second International Conference on Digital Security and Forensics, Cape Town, South Africa*. 2015.

[2] Shane Alcock a Richard Nelson. „Libprotoident: traffic classification using lightweight packet inspection“. In: *WAND Network Research Group, Tech. Rep.* (2012).

[3] Tom Auld, Andrew W. Moore a Stephen F. Gull. „Bayesian Neural Networks for Internet Traffic Classification.“ In: *IEEE Transactions on Neural Networks* 18.1 (19. ún. 2008), s. 223–239. URL: http://dblp.uni-trier.de/db/journals/tnn/tnn18.html#AuldMG07.

[4] Leo Breiman. „Random forests“. In: *Machine Learning* 45.1 (2001), s. 5–32. ISSN: 08856125. DOI: 10.1023/A:1010933404324. arXiv: /dx.doi.org/10.1023{\%}2FA{\%}3A1010933404324 [http:].

[5] Elie Bursztein. „Probabilistic identification for hard to classify protocol“. In: *IFIP International Workshop on Information Security Theory and Practices*. Springer. 2008, s. 49–63.

[6] Shuaifu Dai et al. „NetworkProfiler: Towards automatic fingerprinting of Android apps“. In: *Proceedings - IEEE INFOCOM* (2013), s. 809–817. ISSN: 0743166X. DOI: 10.1109/INFCOM.2013.6566868.

[7] Michel Marie Deza a Elena Deza. „Encyclopedia of distances“. In: *Encyclopedia of Distances*. Springer, 2009, s. 1–583.

[8] Jeffrey Erman et al. „Offline/realtime traffic classification using semi-supervised learning“. In: *Performance Evaluation* 64.9-12 (2007), s. 1194–1213. ISSN: 01665316. DOI: 10.1016/j.peva.2007.06.014.

[9] Jeffrey Erman et al. „Offline/realtime traffic classification using semi-supervised learning“. In: *Performance Evaluation* 64.9 (2007), s. 1194–1213.

[10] Alessandro Finamore, Marco Mellia a Michela Meo. „Mining unclassified traffic using automatic clustering techniques“. In: *Lecture Notes in Computer Science*. Sv. 6613 LNCS. 2011, s. 150–163.

[11] Vahid Aghaei Foroushani a A Nur Zincir-Heywood. „Investigating application behavior in network traffic traces“. In: *Computational Intelligence for Security and Defense Applications (CISDA), 2013 IEEE Symposium on*. IEEE. 2013, s. 72–79.

[12] Jerome Friedman, Trevor Hastie a Robert Tibshirani. „The Elements of Statistical Learning: Data Mining, Inference, and Prediction“. In: *Springer Series in Statistics* (2009).

[13] Nir Friedman et al. „Bayesian Network Classifiers“. In: *Machine Learning* 29 (1997), s. 131–163. ISSN: 0885-6125. DOI: 10.1023/A:1007465528199.

[14] Gabriel Gómez Sena a Pablo Belzarena. „Early traffic classification using support vector machines“. In: *Proceedings of the 5th International Latin American Networking Conference*. ACM. 2009, s. 60–66.

[15] Isabelle Guyon a André Elisseeff. „An introduction to variable and feature selection“. In: *Journal of machine learning research* 3.Mar (2003), s. 1157–1182.

[16] Jiawei Han, Micheline Kamber a Jian Pei. *Data Mining: Concepts and Techniques*. 3rd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN: 0123814790, 9780123814791.

[17] Erik Hjelmvik. „The SPID algorithm-statistical protocol identification“. In: *Gävle, Sweden, October* (2008).

[18] Jawad Khalife, Amjad Hajjar a Jesus Diaz-Verdejo. „A multilevel taxonomy and requirements for an optimal traffic-classification model“. In: *International Journal of Network Management* 24.2 (2014), s. 101–120.

[19] Josef Kittler. „Combining classifiers: A theoretical framework“. In: *Pattern analysis and Applications* 1.1 (1998), s. 18–27.

[20] Christopher Köhnen et al. „Enhancements to Statistical Protocol IDentification (SPID) for Self-Organised QoS in LANs.“ In: *ICCCN*. 2010, s. 1–6.

[21] Jun Li et al. „Identifying skype traffic by random forest“. In: *2007 International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2007*. 2007, s. 2841–2844. ISBN: 1424413125. DOI: 10.1109/WICOM.2007.705.

[22] Yan Luo, Ke Xiang a Sanping Li. „Acceleration of decision tree searching for IP traffic classification“. In: *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*. ACM. 2008, s. 40–49.

[23] Petr Matoušek et al. „Advanced Techniques for Reconstruction of Incomplete Network Data“. Angl. In: *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering* 2015.157 (2015), s. 69–84. ISSN: 1867-8211. URL: http://www.fit.vutbr.cz/research/view_pub.php.cs?id=10864.

[24] Stanislav Miskovic et al. „AppPrint: Automatic fingerprinting of mobile applications in network traffic“. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Sv. 8995. Springer Verlag, 2015, s. 57–69.

[25] Andrew W Moore a Konstantina Papagiannaki. „Toward the accurate identification of network applications“. In: *Passive and Active Network*

Measurement 3431 (2005). Ed. ConstantinosEditor Dovrolis, s. 41–54. URL: http://www.springerlink.com/index/re7ej0uj7eep2htl.pdf.

[26] Andrew Moore, Denis Zuev a Michael Crogan. *Discriminators for use in flow-based classification*. Tech. zpr. 2013.

[27] Neeraj Namdev, Shikha Agrawal a Sanjay Silkari. „Recent advancement in machine learning based internet traffic classification“. In: *Procedia Computer Science* 60 (2015), s. 784–791.

[28] Thuy TT Nguyen a Grenville Armitage. „A survey of techniques for internet traffic classification using machine learning“. In: *IEEE Communications Surveys & Tutorials* 10.4 (2008), s. 56–76.

[29] Jan Pluskal. *Netfox Detective 2.0 - Nástroj pro síťovou forenzní analýzu*. czech. Tech. zpr. FIT-TR-2017-06, CZ, 2017, s. 16. URL: http://www.fit.vutbr.cz/research/view_pub.php?id=11567.

[30] Chaofan Shen a Leijun Huang. „On detection accuracy of L7-filter and OpenDPI“. In: *Networking and Distributed Computing (ICNDC), 2012 Third International Conference on*. IEEE. 2012, s. 119–123.

[31] Petr Velan et al. „A survey of methods for encrypted traffic classification and analysis“. In: *International Journal of Network Management* 25.5 (2015), s. 355–374.

[32] Yu Wang a Shun Zheng Yu. „Machine learned real-time traffic classifiers“. In: *Proceedings - 2008 2nd International Symposium on Intelligent Information Technology Application, IITA 2008*. Sv. 3. 2008, s. 449–454. ISBN: 9780769534978. DOI: 10.1109/IITA.2008.536.

[33] Liu Zhen a Liu Qiong. „A new feature selection method for internet traffic classification using ml“. In: *Physics Procedia* 33 (2012), s. 1338–1345.

[34] Ivan Žežula. „On multivariate Gaussian copulas“. In: *Journal of Statistical Planning and Inference* 139.11 (2009), s. 3942–3946.