VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

IDENTIFIKACE MOBILNÍCH APLIKACÍ V ŠIFROVANÉM PROVOZU

IDENTIFICATION OF MOBILE APPLICATIONS IN ENCRYPTED TRAFFIC

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. DANIEL SNÁŠEL**
AUTHOR

VEDOUCÍ PRÁCE **Ing. PETR MATOUŠEK, Ph.D., M.A.**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2021/2022

Zadání diplomové práce

Student: **Snášel Daniel, Bc.**
Program: Informační technologie a umělá inteligence
Specializace: Kybernetická bezpečnost
Název: **Identifikace mobilních aplikací v šifrovaném provozu**
Identification of Mobile Applications in Encrypted Traffic
Kategorie: Počítačové sítě
Zadání:

1. Seznamte se s metodami pro detekci aplikací v šifrovaném provozu pomocí otisků TLS. Prostudujte dostupné metody pro výpočet otisků TLS. Zaměřte se na charakteristiky získávané ze síťových dat.
2. Pro vybrané mobilní aplikace vytvořte trénovací datovou sadu.
3. Implementujte nástroj pro extrakci charakteristik síťového provozu pro vytváření otisků.
4. Navrhněte způsob pro vytváření otisků za použití např. váhování charakteristik, sledování statistických údajů o komunikaci, apod.
5. Navrhněte a implementujte nástroj pro detekci mobilních aplikací. Ověřte přesnost detekce a použitelnost různých charakteristik.
6. Vyzkoušejte navrženou metodu na reálném provozu. Vyhodnoťte výsledky detekce.
7. Zhodnoťte využití navržené metody a způsob jejího nasazení v praxi.

Literatura:

- Anderson, Blake & McGrew, David. (2020). Accurate TLS Fingerprinting using Destination Context and Knowledge Bases.
- Blake Anderson and David McGrew. 2016. Identifying Encrypted Malware Traffic with Contextual Flow Data. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security* (*AISec '16*). Association for Computing Machinery, New York, NY, USA, 35-46.
- van Ede, T., et al.: FlowPrint: Semi-Supervised Mobile-App Fingerprinting on Encrypted Network Traffic. In: NDSS. The Internet Society, 2020.
- Jiyuan Liu, et al: "Maldetect: a structure of encrypted malware traffic detection," *Computers, Materials & Continua*, vol. 60, no.2, pp. 721-739, 2019.

Při obhajobě semestrální části projektu je požadováno:

- Body 1-4.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Matoušek Petr, Ing., Ph.D., M.A.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 18. května 2022
Datum schválení: 11. října 2021

Abstrakt

Práce se zaměřuje na identifikaci mobilních aplikací v šifrovaném provozu na základě TLS otisků. Cílem práce bylo vytvoření architektury pro získávání vybraných atributů z ustanovení spojení TLS, vytvoření TLS otisků a jejich porovnávání. Byl zde kladen důraz na přesnost jednotlivých metrik, kvalitu vybraných atributů a na určení porovnávacího prahu T, který byl ve výsledku určen na 75 %. Bylo zvoleno celkem deset atributů z ustanovení TLS spojení, jako jsou IP adresa, *Cipher Suites*, *Server Name Indication*, velikosti prvních deseti paketů a další. Pro porovnávání jednotlivých atributů bylo zvoleno přesné, podřetězcové a indexové porovnávání. Celková podobnost dvou TLS otisků se následně vypočítá jako vážený součet shod jednotlivých atributů. Výsledná architektura umožňuje porovnat TLS otisky aplikací z vytvořeného datasetu s nově vytvořenými otisky ze zašifrované komunikace, a tak identifikovat aplikace. Dále umožňuje manuální nebo automatické naučení nových aplikací z porovnávaného souboru, případaně aktualizaci známých TLS otisků aplikací v datasetu.

Abstract

The work focuses on the identification of mobile applications in encrypted traffic based on TLS fingerprints. The aim of the work was to create an architecture for obtaining selected attributes from TLS connection handshake, to create TLS fingerprints and their comparison. Emphasis was placed on the accuracy of individual metrics, the quality of selected attributes and on the determination of the threshold T comparison, which was ultimately set at 75 %. A total of ten attributes were selected from the TLS connection handshake, such as IP address, Cipher Suite, Server Name Indication, the size of the first ten packets and more. Accurate, substring and index comparisons were chosen to compare individual attributes. The total similarity of the two TLS fingerprints is then calculated as the weighted sum of the matches of the individual attributes. The resulting architecture allows you to compare TLS application fingerprints from the created dataset with newly created fingerprints from encrypted communication, and thus identify the applications. It also allows manual or automatic learning of new applications from the compared file, or updating of known TLS fingerprints of applications in the dataset.

Klíčová slova

TLS otisk, identifikace aplikací, TLS, TCP, JA3, JA3s, ustanovení TLS spojení, mobilní aplikace, šifrovaný provoz

Keywords

TLS fingerprinting, identification of applications, TLS, TCP, JA3, JA3s, TLS handshake, mobile applications, encrypted traffic

Citace

SNÁŠEL, Daniel. *Identifikace mobilních aplikací v šifrovaném provozu*. Brno, 2022. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Petr Matoušek, Ph.D., M.A.

Identifikace mobilních aplikací v šifrovaném provozu

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Petra Matouška, Ph.D., M.A. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Daniel Snášel
14. května 2022

Poděkování

Rád bych poděkoval vedoucímu práce Ing. Petru Matoučkovi, Ph.D, M.A, za pomoc při výběru tématu, cenné rady, poskytnutí materiálů a odborné vedení diplomové práce.

Obsah

Kapitola 1

Úvod

Od doby, kdy byly počítače poprvé vzájemně propojeny přes síť, uběhlo již mnoho let. V průběhu několika let se počet vzájemně propojených počítačů v celosvětové síti internet zvýšil několikanásobně. Se zvyšujícím se počtem zařízení se vyvinula i hrozba ve formě sledování a odposlouchávání uživatele. Sledování síťové komunikace mezi klientským zařízením a serverem není užitečné jen pro útočníky, ale také pro síťové administrátory, kteří tak získávají větší přehled o aktivních aplikacích v síti. Nevhodné aplikace mohou následně omezit nebo úplně zakázat, případně na základě těchto informací nastavit přísnější pravidla pro *firewall*. Pomocí vhodných nástrojů lze ručně nebo automatizovaně rozpoznat *malware* na síti a ihned vyřešit bezpečností incident. Příkladem nástrojů vhodných pro detekci škodlivého obsahu na síti jsou IDS (Intrusion detection system) nebo DPI (Deep Packet Inspection) [3]. Pomocí protokolu TLS lze zamezit útočníkovi ve čtení a pochopení odposlouchávané síťové komunikace, ale i značně omezit rozeznávání škodlivé a legitimní síťové komunikace [2].

Protokol TLS (*Transport Layer Secure*) [13] nebo jeho předchůdce SSL (*Secure Sockets Layer*) slouží k šifrování síťové komunikace. TLS protokol chrání aplikační data, která jsou odeslána mezi dvěma body v síti, před útočníky. Šifrovací algoritmy v protokolu nedovolují útočníkům bez znalosti klíče číst či modifikovat přenášená data. Pro identifikaci aplikace v zašifrovaném síťovém provozu lze použít *otisky* aplikací. Standardní metody, jako je IDS pro rozeznávání síťové komunikace, zde nemusí fungovat, protože většina z nich je založena na zkoumání obsahu zasílaných paketů [4].

Termín otisk aplikace byl převzatý z daktyloskopie, kde se zkoumají otisky prstů. Otisk prstu je unikátní identifikátor pro danou osobu, od níž byl získán. Stejně jako je otisk prstu unikátní pro člověka, tak by otisk aplikace měl být co nejvíce unikátní pro aplikace. Míra unikátnosti otisku se odvíjí od procesu získávání až po jeho zpracování a uložení do databáze.

Jednou z možností vytváření otisku aplikace je algoritmus *JA3* a *JA3s* [22]. Algoritmus vytváří otisk aplikace pomocí dvou TLS paketů *Client Hello* a *Server Hello*. Detailnější popis postupu získávání otisku viz sekce 2.3.

Mobilní prostředí se neustále vyvíjí, tj. aplikace se pravidelně stahují, instalují, aktualizují a odinstalovávají. Při aktualizaci může dojít ke změně nastavení síťové komunikace pro danou aplikaci. To má za následek modifikaci atributů, ze kterých se vytváří otisk aplikace, a tak dojde i ke změně otisku aplikace. Pro algoritmus JA3 a JA3s je modifikace jednoho zkoumaného atributu fatální. Následně není schopen správně identifikovat známou aplikaci, protože poskytuje možnost pouze exaktního porovnání dvou MD5 heší [11].

Cílem práce je blíže seznámit čtenáře s problematikou identifikace mobilních aplikací v šifrovaném síťovém provozu, navázání TLS spojení a vytváření aplikačních otisků. Dále

pak vybrat vhodné atributy a metriky pro vytváření otisku aplikace a přiřadit k nim vhodné váhy, které budou nabízet co největší míru unikátnosti pro jednotlivé aplikace. Hlavním požadavkem je automatické zpracovávání otisků a vysoká přesnost detekce aplikací.

Text práce je rozdělen do čtyř částí. První část je teoretická, v ní bude vysvětlena problematika navazování síťového spojení mezi dvěma body v síti. V kapitole 2 je kladen důraz na vysvětlení ustanovení šifrované komunikace. V následující sekci je detailně vysvětlen algoritmus JA3 a JA3s. V kapitole 3 jsou popsána aktuální řešení pro identifikace aplikací, případně pro identifikace škodlivého obsahu na síti.

Druhá část se zaměřuje na extrakci paketů ze síťové komunikace a na výběr atributů z extrahovaných paketů. Ze série paketů, které si vyměňují mezi sebou klient a server, byly vybrány pakety *Client Hello* a *Server Certificate*. Kapitola 4 se detailněji zaměřuje na popis vybraných atributů a jejich následné váhové ohodnocení. Dále se zaměřuje na dvě statistické metriky. První je metrika časová, která se zaměřuje na počet přijatých a odeslaných paketů a na jejich velikost. Druhá metrika je zaměřena na zkoumání vlastností prvních deseti paketů a jejich poměru mezi odeslanými a přijatými pakety.

Třetí část práce se zaměřuje na implementaci celé topologie, a to od získávání záznamů komunikace až po vytvoření TLS otisků uložených v databázi. Všechna získaná data byla nahrána na webový portál *IEEE Dataport* [25]. Nejprve bude rozebrána topologie pro získávání záznamu mobilní aplikace a jejich výčet, viz kapitola 5. Výsledná implementace architektury byla řádně otestována. Následující podkapitoly se budou zabývat extrakcí atributů a jejich způsobem porovnání. Poslední část kapitoly 5 se bude zaobírat způsobem uložení otisků, vytvořením nových otisků a jejich aktualizací.

Poslední část práce se bude zaměřovat na experimenty nad testovací sadou. V kapitole 6 bude pomocí experimentů určen porovnávací práh, který je důležitý pro porovnávání dvou otisků a pro celkovou identifikaci. Dále bude kapitola obsahovat experimenty s vybranými atributy. Poslední kapitola 7 bude obsahovat porovnání nově vytvořenou metodu s metodou JA3 a JA3s.

Kapitola 2

Šifrování aplikačního provozu

Tato kapitola je věnována obecnému navázání komunikace od klienta k serveru. Nejprve je zde rozebrán postup navázání nešifrované komunikace a její vlastnosti. Dále kapitola obsahuje postup navázání šifrované komunikace dle dnešních standardů a její vlastnosti. Na konci kapitoly je popis jedné z možností vytváření otisku mobilní aplikace, která využívá šifrovanou komunikaci.

2.1 Navázání nešifrované komunikace

Tato podkapitola se bude zabývat navázáním komunikace na čtvrté vrstvě ISO/OSI modelu [29] pro protokoly TCP (Transmission Control Protocol) a UDP (User Datagram Protocol).

2.1.1 Navázaní komunikace TCP

Pro navázání komunikace nad protokolem TCP je zapotřebí provést třífázovou synchronizaci, tzv. *three-way handshake*. Jedná se o proces synchronizace a ustanovení spojení mezi dvěma koncovými body.

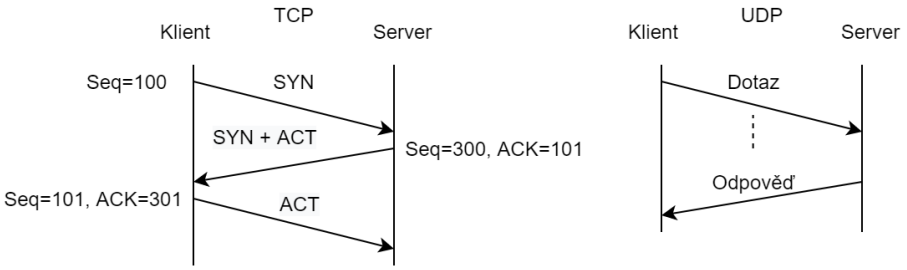
Na obrázku lze vidět, že klient zahajuje komunikaci zasláním paketu s příznakem *SYN*, dále paket obsahuje zdrojový a cílový port a další atributy. Důležitou položkou je sekvenční číslo na obrázku označené jako *Seq*. Toto číslo je náhodně vygenerované a slouží k řízení toku paketů na síti, potvrzení přijetí paketů atd. Server odpoví klientovi paketem, v němž potvrdí klientovo sekvenční číslo pomocí příznaku *ACK* a číslem, ve kterém je sečteno klientovo sekvenční číslo a velikost zaslaných dat v poli *Acknowledgement Number* [21]. Paket dále obsahuje sekvenční číslo serveru a příznak *SYN*.

Klient po přijetí paketu od serveru odešle paket s příznakem *ACK*, jímž potvrdí sekvenční číslo serveru. Po úspěšném navázání spojení už klient/server zasílají svá data. Pokud už nemá server nebo klient žádná data k odeslání, je potřeba provést ukončení spojení. Navázání spojení trvá přibližně 100 ms, doba je závislá na vzdálenosti dvou bodů a rychlosti připojení.

Protokol TCP je spolehlivý, protože řeší výpadky a znovu zaslání ztracených paketů [17]. Při ztrátě jednoho či více paketů dojde k jejich znovu zaslání, a to například pomocí algoritmu Go-Back-N [26].

2.1.2 Navázání komunikace UDP

Protokol UDP nenavazuje spojení, ale rovnou odesílá data, neřeší ztracené a duplicitní pakety, a proto není spolehlivý. O spolehlivé doručení dat se stará vyšší vrstva ISO/OSI modelu a programátor. Protokol UDP je výrazně rychlejší než protokol TCP, protože nečeká na potvrzení přijatých paketů.



Obrázek 2.1: Navázaní spojení TCP/UDP

2.2 Navázání šifrované komunikace

Při navazování TCP/UDP komunikace není zaručeno, že si zaslaný obsah nepřečte žádný útočník, a proto je zapotřebí komunikaci šifrovat. Navázání zašifrované komunikace u TCP vychází z navazování nešifrované komunikace, která je rozšířena o zprávy typu *Client Hello*, *Server Hello* a další [23]. Pomocí těchto zpráv si klient a server vymění informace o podporovaných algoritmech a domluví se na šifrovacím algoritmu. Celý proces navazování šifrovaného spojení se nazývá *TLS handshake*. Po ukončení navázání šifrovaného spojení se už zasílají aplikační data, jež jsou zašifrovaná, viz obrázek 2.2.

2.2.1 Kroky TLS handshake

TLS handshake je tvořen sérií zpráv (datagramů), které si mezi sebou vyměňují klient a server. *Handshake* je složen z několika kroků, podobně jako tomu bylo v případě TCP *three-way handshake*.

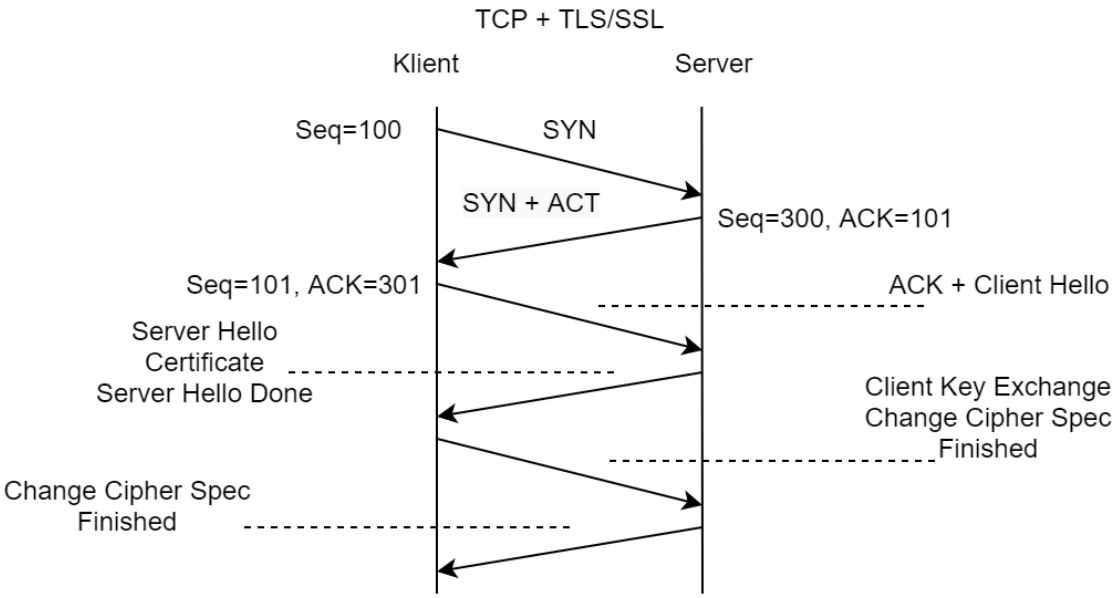
Přesné kroky se budou lišit v závislosti na druhu použitého algoritmu výměny klíčů a šifrovacích sadách podporovaných oběma stranami. Nejčastěji se pro výměny klíčů používají algoritmy RSA a Diffie-Hellman (dále už jen DH). Algoritmus DH využívá exponenciální výpočty pro získání sdíleného *premaster* klíče. Sever i klient postupně poskytují parametry potřebné k výpočtu sdíleného klíče. DH jsou rozepsány zde:

1. Klient zahájí handshake odesláním zprávy *Client Hello* na server. Zpráva bude obsahovat verzi TLS, kterou klient podporuje, podporované šifrovací sady (dále už jen cipher-suites) a řetězec náhodných bajtů známý jako *Random Client*.

2. Server zprávu od klienta přijme a odpoví mu zprávou *Server Hello*, jež bude obsahovat certifikát TLS serveru, serverem vybranou šifrovací sadu a *server random*, náhodný řetězec bajtů, který generuje server.

3. Server použije svůj soukromý klíč k zašifrování *Random Client*, *Random Server* a jeho parametru DH. Takto zašifrovaná data slouží jako digitální podpis serveru. Zároveň

tím server potvrzuje, že se jeho soukromý klíč shoduje s veřejným klíčem obsaženým v certifikátu. Alternativou DH je algoritmus RSA.



Obrázek 2.2: Navázání šifrovaného spojení

2.3 Vytváření otisků

Pro identifikaci mobilní aplikace ze zašifrovaného síťového provozu je zapotřebí nasbírat dostatečné množství síťové komunikace (dále už jen „dataset“). Ze získaného datasetu lze otisky aplikací vytvořit několika způsoby. Příkladem takového algoritmu je JA3 a JA3s[1].

U atributu *Cipher-suite-set* je nejprve potřebné odstranit nebo nahradit konstantou tzv. „Grease Value“[2] (Generate Random Extensions And Sustain Extensibility) [18]. Následně se všechny hodnoty převedou do celočíselného datového typu a nakonec je vypočítán otisk jako MD5 heš [11].

JA3 algoritmus [22] uvažuje pro výpočet samotného otisku následující atributy z paketu *Client Hello*[3] v přesně daném pořadí: verze protokolu TLS, podpora šifer, podpora rozšíření protokolu, podporované typy eliptických křivek a formáty bodů eliptických křivek.

Příklad výpočtu JA3

Pomocí překladu jednotlivých atributů do jejich dekadické reprezentace lze získat například následující hodnotu:

1) 769,47–53–5–10–49161–49162–49171–49172–50–56–19–4,0–10–11,23–24–25,0

Pokud v paketu *Client Hello* nejsou žádná rozšíření TLS, pole zůstanou prázdná.

[1] https://engineering.salesforce.com/tls-fingerprinting-with-ja3-and-ja3s-247362855967
[2] Algoritmus JA3 a JA3s tyto hodnoty pouze odstraňuje.
[3] TLS - https://hpbn.co/transport-layer-security-tls/

2) 769,4–5–10–9–100–98–3–6–19–18–99, , ,

Jelikož je tento řetězec příliš dlouhý pro ověřování inteligentními IDS systémy, používá se MD5 heš. Heš je vypočten ze získaných hodnot v minulém kroku.

1) ada70206e40642a3e4461f35503241d5
2) de350869b8c85de67a350c8d186f11e6

JA3s je metoda pro vytvoření otisku pro serverovou stranu komunikace. Pro výpočet samotného otisku se používají atributy z paketu *Server Hello* v přesně daném pořadí: verze protokolu TLS, podpora šifer a podpora rozšíření protokolu.

Příklad výpočtu JA3s

Nejprve je zapotřebí provést překlad atributů do dekadické reprezentace.

769,47,65281-0-11-35-5-16

Stejně jako v případě JA3 se nyní vypočte MD5 heš.

836ce314215654b5b1f85f97c73e506f

JA3 a JA3s jsou metody pro vytváření otisků z TLS komunikace. Pro identifikaci aplikace se využívá spojení těchto dvou otisků.

2.4 Shrnutí

Pro ustanovení šifrované komunikace nad protokolem TCP je nezbytné přidat režii ve formě *TLS handshake*, přičemž si obě strany vymění podporované algoritmy, formát bodů eliptických křivek a další. Po ustanovení spojení probíhá veškeré zasílání dat mezi klientem a serverem šifrovaně. Výhodou přidání šifrování je, že útočník není schopen odposlouchávat komunikaci. Nevýhodou je přidání režie a prodloužení ustanovení spojení z přibližně 100 ms až na 300 ms, viz obrázek 2.2.

Algoritmy JA3 a JA3s umí na základě dat z *Client Hello* a *Server Hello* vytvořit otisk aplikace, který se následně může využít pro identifikaci aplikace v zašifrovaném síťovém provozu. Spolehlivost algoritmu není stoprocentní, protože při aktualizaci aplikace může dojít ke změně otisku. U některých aplikací může docházet ke kolizím, protože různé aplikace můžou mít stejné otisky. Výsledek JA3 (JA3s) je MD5 heš dovolující pouze porovnání na přesnou shodu.

Další existující metody na rozpoznávání aplikací či detekci škodlivých programů budou rozebrány v následující kapitole 3.

Diplomová práce se zaměřuje na vylepšení a rozšíření JA3 algoritmu, viz kapitola 4. Nová metoda přidává možnost váhování jednotlivých extrahovaných parametrů. Tato možnost odstraňuje nevýhody porovnání pouze na přesnou shodu, jak tomu bylo v případě čisté metody JA3.

Kapitola 3

Existující řešení

V následující kapitole budou rozebrána čtyři aktuální řešení dané problematiky. Nejprve zde bude přístup založený na váhovaném Bayesově klasifikátoru, který jednotlivé aplikace spojuje do shluku dle jejich vlastností. Následně se pokusí shlukům přiřadit konkrétní existující aplikaci.

Druhé řešení si dává za úkol odstranit problém s aktualizací aplikace. Po procesu aktualizace nebo jejím přeinstalování, při němž se změní posloupnost zasílání paketů, certifikát, způsob šifrování atd. Řešení je založeno na metodě *Semi-Supervised*, která má na začátku databázi známých aplikací s vytvořenými otisky. Pokud aplikace změní například způsob šifrování, tak metoda přidá starý a nový otisk do jednoho shluku.

Poslední dvě řešení se zaměřují na detekci škodlivých programů (dále už jen „malware“) ze zašifrované komunikace. Třetí metoda zkoumá hlavně prvních osm paketů síťové komunikace, z nichž získává potřebná data pro *Online Random Forest* klasifikátor. Poslední metoda využívá kontextové informace z ustanovení šifrovaného spojení a následného datového toku, který je zaznamenáván až po dobu pěti minut.

3.1 Metoda Accurate TLS Fingerprinting

Metoda Accurate TLS Fingerprinting [7] se zabývá identifikací síťového provozu pomocí TLS otisků a naivního Bayesova klasifikátoru s váhovými vektory. Pro vytvoření otisku aplikace využívá převážně paket *Client Hello*, ze kterého získává IP adresu, port, jméno serveru (pokud je k dispozici), rozšíření a jejich typy. Z paketu jsou abstrahovány všechny typy, ale pouze některé hodnoty z rozšíření. Vynechaná rozšíření se vztahují k aktuální relaci, například *server name* a *key_share*. Mezi ponechaná data patří například *supported groups* a *compress certificate*. Získané hodnoty jsou normalizované například pomocí nahrazení *Grease Values* za hodnotu `0a0a`. Vytvořené otisky se následně ukládají do znalostní báze.

Znalostní báze je neustále aktualizována. Pokud se objeví nový TLS otisk, tak se jej metoda pokusí přiřadit k již existujícím otiskům, a tím vytvořit nebo zvětšit shluk, který bude obsahovat velmi podobné otisky. Pro zařazení aplikací do shluku se využívá naivní Bayesovský klasifikátor a váhovaný Bayesovský klasifikátor. Jednotlivým shlukům jsou přiřazeny odpovídající aplikace.

3.2 Systém Flowprint: Semi-Supervised

Flowprint [28] si dává za úkol odstranit nedostatek ostatních řešení, která pro detekci či rozpoznání potřebují předchozí znalost aplikací. Řešení se zaměřuje hlavně na mobilní aplikace, a to na zařízení s operačním systémem Android a iOS. Mobilní prostředí se však neustále vyvíjí, tj. aplikace se pravidelně instalují, aktualizují a odinstalovávají z mobilního zařízení. Z toho důvodu je pro stávající přístupy ke snímání TLS otisků nemožné pokrýt všechny aplikace, jež se mohou objevit v síti.

Řešení Flowprint nepředpokládá předchozí znalosti o aplikacích spuštěných v síti, ale zaměřuje se na generování otisků aplikací, které lze následně použít pro rozpoznání známé aplikace nebo izolaci dříve neviděné aplikace. Z otisků lze následně vytvořit seznam aplikací, jež budou v síti povoleny nebo zakázány.

Systém Flowprint využívá metodu *Semi-Supervised* neboli metodu učení založenou na znalosti některých vstupných aplikací. Metoda z datového toku extrahuje IP, port, časové razítko toku a TLS certifikát. Pomocí těchto hodnot vytváří shluky pro jednotlivé aplikace. Následně získávají data ze síťového toku, jako je počet paketů a jejich směr.

Flowprint automaticky nachází časové korelace mezi jednotlivými rysy souvisejícími se síťovým provozem, které následně využívá pro generování TLS otisků. K získaným hodnotám dále přidává informace z hlaviček jednotlivých síťových vrstev. Tento přístup umožňuje detekovat dříve neviděné aplikace nebo nové verze známých aplikací.

3.3 Systém MalDetect

Tato metoda se využívá na detekci malwaru [20], který se šíří pomocí celosvětové sítě internet v zašifrované podobě. Metoda je založena na principu odchytávání síťové komunikace mezi napadenou stanicí a komunikující stranou za pomoci knihovny *Libpcap*.

MalDetect extrahuje atributy přibližně z osmi paketů (počet se liší v různých tocích), a to na začátku každého toku paketů po síti, a proto je možné detekovat provoz malwaru dříve, než chování malwaru nabude praktických dopadů.

Metoda rozděluje extrakci a charakteristiku spojení do tří kategorií. První kategorie obsahuje data z charakteristik síťového toku, jako je počet a velikost odeslaných paketů od klienta na server. Následně zjišťuje počet a velikost paketů přijatých klientem od serveru. Zachycené pakety rozřadí do vektoru obsahujícího 11 pozic. Pokud má paket velikost 0-150 bytů, tak je přiřazen na první pozici ve vektoru. V případě velikosti paketu více než 1 500 bytů je paket zařazen na poslední index tohoto vektoru.

Druhá kategorie obsahuje hodnoty získané z *Client Hello* a *Server Hello* paketů. Z těchto paketů získává verzi TLS, podporované šifrovací algoritmy, vybrané metody komprese, nabízená a vybraná rozšíření.

Poslední kategorie získává data ze paketu *Server Certification*. Z tohoto paketu extrahuje číslo a verzi jednoho nebo více certifikátů, protože v paketu může být přenášeno více certifikátů. Dalšími získanými hodnotami jsou počet rozšíření certifikátů a doba, po kterou je certifikát validní, velikost veřejného klíče a použitý algoritmus pro jeho podpis.

MalDetect nejprve využíval trénování offline klasifikátoru se statistickými metodami [6], ale když dorazí velké množství nových vzorků, je potřeba tento klasifikátor přetrénovat. Tento způsob nebyl ideální, protože byl časově neefektivní a musel se po krátké době znovu přetrénovat. Druhé řešení zahrnovalo jiný klasifikátor, a to *Online Random Forest*. Tento klasifikátor umožňuje ladění parametrů, potřebných pro výpočet, za běhu programu a zbavit se tak procesu rekvalifikace. MalDetect je naprogramován v jazyce C++.

3.4 Metoda založená na kontextu zasílaných dat

Poslední metoda pro zkoumání hrozeb v zašifrované komunikaci využívá strojové učení s učitelem [5]. Z ustanovení šifrovaného spojení získává kontextové informace z hlaviček HTTP a TLS. Metoda je složena z několika funkcí, které extrahují hodnoty ze zašifrované komunikace. První funkce je založena na metadatech, jako jsou sekvence délky paketů a jejich čas přijetí. Sekvence jsou následně modelovány jako Markovy řetězce. Každých 50 ms záznamu je rozděleno do pole o velikosti deseti prvků.

Druhá funkce představuje data získaná z paketů odeslaných od klienta na server. Z paketů extrahuje seznam nabízených šifrovacích algoritmů, seznam rozšíření a délku veřejného klíče klienta. Třetí funkce shromažďuje názvy komunikujících serverů.

Poslední funkce zaznamenává TLS pakety a jejich HTTP hlavičky se stejnou zdrojovou IP adresou do jednoho 5 minutového datového toku. Následně ze zaznamenaných paketů získává hodnoty z pozic *Content-Type, User-Agent, AcceptLanguage a Server*

Všechny hodnoty jsou normalizovány a následně zaslány do klasifikátoru, který vyhodnotí, jestli se jedná o síťový provoz obsahující *malware* nebo ne.

Metoda využívá klasifikátor logické regrese, který je vysoce přesný a má snadno interpretovatelné parametry, na rozdíl od některých klasifikátorů poskytujících binární nebo procentuální výstupy.

Klasifikátor byl natrénován na datové sadě o velikosti přes jeden milion unikátních datových toků a byl testován na reálném provozu, kde se ukázala jeho přesnost, a to 99.978 % při 0 % chybovosti vstupních dat.

3.5 Získané znalosti

Z výše popsaných metod a algoritmu JA3/JA3s byly získány znalosti o využívaných atributech při identifikaci aplikace nebo *malware*. Pouze metoda JA3 využívá přesné porovnání otisku aplikace s vytvořeným otiskem ve znalostní bázi. Výhoda exaktního porovnávání je v její rychlosti, ale postupem času ztrácí přesnost. U každé hašovací funkce může vzniknout kolize, která může velmi ovlivnit přesnost detekce. Z tohoto důvodu bude v nově navržené metodě možnost porovnávat získané atributy postupně bez nutnosti generování a kontroly duplicity u heš hodnoty.

Některé metody využívají přidělování vah pro jednotlivé atributy, například váhovaný Bayesovský klasifikátor nebo metoda založená na získávání kontextových dat. Při zkoumání jednotlivých atributů z ustanovení síťového provozu byly k jednotlivým atributům přiděleny váhy, viz kapitola 4.

Další důležitou znalostí jsou metody pro tvoření otisků a jejich následné klasifikování či shlukovaní. Extrahované atributy lze ukládat do znalostí databáze odděleně, jelikož každá hodnota má předem definovaný sloupec. Další způsob je atributy spojovat do dvojic či řetězců. Kvůli přehlednosti uložených dat byl zvolen první způsob ukládání dat. Data je nutné před jejich uložení nebo porovnáním normalizovat, aby byl výsledek přesnější, například *Grease Values* jsou nahrazeny konstantou.

Různé metody využívají různé váhy a atributy pro identifikaci aplikace, případně pro jejich shlukování. Všechny metody ale využívají data z *TLS handshake*, přesněji z paketu *Client Hello*. Vybrané atributy z ustanovení TLS spojení budou rozebrány v následující kapitole, a to včetně přidělených vah. Druhá část kapitoly bude zaměřena na časovou korelaci statistických dat.

Metoda	Využívané atributy	Klasifikátor	Udávaná přesnost	Výhody/nevýhody
Accurate TLS [7]	IP, port, SNI, rozšíření a jejich typy	Váhovaný Bayesův klasifikátor	99.90 % F1 = 0.9	+ Vysoká přesnost + Počítačové aplikace - Nutný vstupní dataset - Časem ztrácí přesnost
Flowprint [28]	IP, port, TLS certifikát, čas a doba trvání datového toku, počet přenesených paketů a jejich směr	Semi-Supervised	89.20 %	+ Android a IOS + Detekce aktualizovaných aplikací + Malý vstupní dataset - Dlouhý učící se čas - Nedokáže detekovat malware
MalDetect [20]	Počet přijatý/deslaných paketů a jejich velikost, TLS verze, TLS šifrovací alg. ...	Online Random Forest	99.11 %	+ Detekce malware + Vysoká přesnost - Nutný vstupní dataset
Kontextová metoda [5]	5 min záznam síťového toku, HTTP hlavičky, velikost paketů...	Klasifikátor logické regrese	99.97 %	+ Detekce malware + Dobře interpretovatelné výsledky - Nutný vstupní dataset - Relativně pomalá
JA3 JA3+JA3s JA3+JA3s+ SNI [22]	Verze protokolu TLS, podporovanou šifrovací sadu, podporované rozšíření, IPv4 ...	Exaktní porovnání dvou MD5 hash	18.18 % 69.70 % 78.79 %	+ Rychlá metoda + Snadno implementovatelná - Exaktní provnání - Nutný vstupní dataset

Tabulka 3.1: Srovnání existujících řešení identifikac síťového provozu.

Porovnání jednotlivých metod, viz tabulka 3.1. Tabulka obsahuje celkem pět sloupců, první sloupec obsahuje název metody. Ve druhém sloupci se vyskytují atributy, které využívá data metoda. Následující sloupce obsahují využívaný klasifikátor, přesnost udávanou ve vědeckém článku a jejich výhody případně nevýhody.

Kapitola 4

Výběr charakteristických hodnot pro detekci

Tato kapitola bude věnována především analýze síťové komunikace mezi klientem a serverem. Nejprve zde budou rozebrány vybrané atributy paketů *Client Hello* a *Server Certificate*. Následně budou rozebrána statistická data. Všechny níže vypsané atributy jsou ohodnoceny váhou dle jejich významnosti pro identifikaci. V tabulce 4.1 se nacházejí aplikace včetně jejich verzí, na kterých byly prováděny experimenty.

Název aplikace	**Verze aplikace**
Bazoš	2.9.0
Benzina	4.2.1
BlaBlaCar	5.80.0
Booxy	4.5.5
ČHMÚ	1.8
Kaufland	2.20.0
Lidl	14.36.0
Mapy.cz	8.5.0
Shazam	11.21.0-210409
Waze	4.73.0.3

Tabulka 4.1: Zkoumané aplikace a jejich verze.

4.1 Informační entropie

Informační nebo též shannonovská entropie [8] je střední hodnota množství informace připadající na jeden symbol generovaný stochastickým zdrojem dat. Shannon definoval informace následovně:

> informace je míra množství neurčitosti nebo nejistoty o nějakém náhodném ději odstraněná realizací tohoto děje.

Informace je zpráva, která snižuje neznalost (neurčitost) o stavu systému. Systém má n možných stavů, kde p_i, i = 1, 2, .., n je pravděpodobnost jejich výskytu, přičemž musí

platit vztah:

$$\sum_{i=1}^{n} p_i = 1, p_i \geq 0 \tag{4.1}$$

Poté množství informace nutné k odstranění neurčitosti o stavu systému (tzv. Shannonova entropie) je dáno následujícím vzorcem: **Shannonův vzorec**

$$H = -\sum_{i=1}^{n} p_i \cdot ln\ p_i \tag{4.2}$$

Pro výpočet neurčitosti jednotlivých extrahovaných atributů byla použita právě informační entropie [27]. Na základě vypočtené informační entropie byla následně odvozena váha jednotlivých atributů a zjištěna korelace mezi některými atributy. Pomocí této korelace lze snížit počet extrahovaných atributů, a tak zvýšit efektivnost otisků a jejich porovnání.

4.2 Ustanovení TLS spojení

Při navazování TLS spojení si klient a server vymění sekvenci paketů, v nichž se domluví na šifrovacím algoritmu, šifrovacích klíčích obou stran a dalších podrobnostech spojení. Některé atributy z těchto paketů se dají použít pro identifikaci komunikující aplikace. Příklady vhodných atributů pro identifikaci aplikací, viz podkapitola vytváření otisků 2.3.

4.2.1 Zpráva *Client Hello*

Důležité atributy paketu *Client Hello* jsou *IP adresa serveru*, *Port*, *Verze TLS protokolu*, *Cipher Suite*, *Extensions*, *název serveru* a další. Například algoritmus JA3 využívá jen verze TLS protokolu, *Cipher Suite*, *Extensions*, eliptické křivky a jejich formát.

IP adresa serveru se získává z IP datagramu. Adresa cíle je velmi důležitá pro identifikaci síťového provozu, protože má vysokou variabilitu. Celkový počet IPv4 adres je 4 294 967 296 (2^{32}), po odečtení privátních rozsahů je počet všech veřejných IPv4 adres 3 706 452 992 [19]. Síťové aplikace využívají jednu až několik desítek IP adres, na které zasílají nebo z nich získávají data. Pokud aplikace komunikuje pouze s jednou IP adresou, je tato adresa velmi dobrým atributem pro identifikaci.

Pro aplikace z vytvořeného datasetu platí, že stačí kombinace IP a portu pro detekci aplikace. Obecně toto tvrzení nemusí fungovat, protože jedna IP adresa může poskytovat více služeb, případně se IP adresy serveru mohou měnit využitím principu *load balancing*. Tento princip označuje efektivní distribuci příchozího síťového provozu mezi skupinu serverů poskytujících službu [10].

Většina aplikací komunikuje s více IP adresami v případě, že aplikace obsahuje dodatečné reklamy, čímž se počet komunikujících adres zvýší, viz tabulka 4.2. Tabulka obsahuje aplikace a IP adresy, z nichž byly odstraněny adresy poskytující dodatečné reklamy. Všechny aplikace ze zkoumaného datasetu komunikují s unikátními IP adresami serveru. Toto pravidlo nemusí obecně platit pro všechny aplikace. Celkový počet zaznamenaných unikátních IP je 46. Do tohoto počtu nebyly zahrnuty reklamní adresy a adresy nepatřící přímo k samotné aplikaci. Odstranění těchto adres bylo možné díky znalosti mobilních aplikací a názvu komunikujících serverů. U každého paketu *Client server* bylo zkoumáno pole s názvem serveru, který byl porovnáván s názvem aplikace. Pokud toto pole obsahovalo

podřetězec s názvem aplikace nebo podřetězec, jenž byl ručně vyhodnocen jako unikátní pro danou aplikaci, tak z toho paketu byla získána IP adresa.

Z paketu byla získávána a porovnávána pouze IPv4 adresa. Pro IPv6 by se získávání a porovnávání provádělo obdobně.

Informační entropie pro IP adresu je přibližně 5.585 informačního bitu. Takto vysoká hodnota entropie značí, že tento atribut nese vysokou míru neurčitosti. Z toho důvodu je IPv4 adresa serveru jeden z nejdůležitějších atributů pro identifikaci aplikace.

Na základě vysoké hodnoty informační entropie byla pro IP adresu přiřazena jedna z nejvyšších vah, a to konkrétně 15. Pro úspěšnou identifikaci je potřeba získat vážený součet, který je roven 75 % a více. Přesný výpočet podobnosti dvou otisků bude vysvětlen později. Největší nevýhodou pro tento atribut je možnost, že jedna IP adresa může být přiřazena více aplikacím, například pokud aplikace využívají *cloud* pro své hostování, případně aplikace může komunikovat se sítí pro doručování obsahu. Jedná se o distribuovanou síť serverů umožňující efektivní doručování webového obsahu uživatelům (dále už jen „CDN“). Sítě CDN za účelem minimalizace latence uchovávají obsah uložený v mezipaměti na hraničních serverech, které jsou v lokalitách POP (Point of Presence) blízko koncovým uživatelům. Například aplikace Kaufland využívá CDN síť pro doručovaní svého obsahu, a proto využívá IP adresy serveru podle toho, kde se daný uživatel nachází. Z těchto důvodu není možné využít IP adresu jako definitivní identifikátor aplikací. Celkový přehled komunikujících IP adres a jejich zařazení do Cloud/CDN sítí, viz příloha F.

Aplikace	**IP adresy komunikujících serverů.**	**Počet IP**	**Počet Cloud IP**	**CDN IP**
Bazoš	88.86.119.246	1	0	Ne
Benzina	91.231.171.211, 91.231.171.212	2	0	Ne
BlaBlaCar	34.117.9.118, 35.244.237.205, 34.95.68.122, ...	4	4	Ano Google
Booxy	185.183.8.127	1	0	Ne
ČHMU	104.21.89.95, 172.67.139.185	2	2	Ano Cloudflare
Kaufland	52.236.157.206, 35.181.18.61, 52.157.251.163, ...	6	6	Ano Akamai
Lidl	51.105.123.133, 51.105.175.147, 40.127.239.139, ...	10	10	Ano Akamai
Mapy	77.75.77.28, 77.75.79.155, 77.75.76.182, ...	7	0	Ano Seznam
Shazam	35.241.30.194, 34.102.169.70, 34.120.233.61, ...	4	4	Ano Google a Fasty
Waze	35.190.40.98, 130.211.16.132, 107.178.245.42, ...	13	13	Ano Google

Tabulka 4.2: Aplikace a komunikující IPv4 adresy.

Aplikace	Názvy komunikujících serverů (SNI)
Bazoš	www.bazos.cz
Benzina	lms.benzina-platby.cz, zps.benzina-platby.cz
BlaBlaCar	edge.blablacar.cz, auth.blablacar.cz, t.blablacar.com, cdn.blablacar.com
Booxy	ms.cbdb.cz, www.cbdb.cz, partner.cbdb.cz
ČHMU	data.pocasi-data.cz
Kaufland	app.kaufland.net, kauflandstiftungandcokg.sc.omtrdc.net, sync.kaufland.de, media.kaufland.com
Lidl	accounts.lidl.com, segments.lidlplus.com, media.lidl-flyer.com, appgateway.lidlplus.com, lidlplusprod.blob.core.windows.net
Mapy	vectmap.mapy.cz, mapserver.mapy.cz, api.mapy.cz, pro.mapy.cz
Shazem	beacon.shazam.com, amp.shazam.com, config.shazam.com, cdn.shazam.com, images.shazam.com
Waze	rt.waze.com, cresg.waze.com, ads-resources-legacy.waze.com, rtproxy-row.waze.com, tts.waze.com, ads-recources.waze.com, inbox-row.waze.com, gabi.waze.com, sdk.waze.com, row-advil.waze.com, social-row.waze.com

Tabulka 4.3: Aplikace a jména serverů komunikujících s aplikacemi.

Název serveru je společně s IP adresou atributem obsahujícím velmi důležitou informaci pro identifikaci aplikace. Každá IP adresa má v jeden okamžik přiděleno několik názvů serverů (dále už jen „SNI“), viz tabulka 4.3. SNI může ale nemusí obsahovat přímo jméno aplikace, například pro aplikaci *Shazam* platí, že v SNI se nachází řetězec shodující se s názvem aplikace. Pro aplikaci *Booxy* toto neplatí, protože získává data ze serveru *www.cbdb.cz*. SNI lze získat z rozšíření paketu *Client Hello* z pole s názvem *server name*, které obsahuje řetězec v čitelné podobě.

Pro tento atribut byla zvolena vyšší váha 15, a to kvůli vysokému informačnímu významu pro identifikaci aplikace. Vypočtená hodnota informační entropie je pro SNI přibližně rovna 5.322 informačního bitu. Pokud SNI obsahuje název aplikace, tak je tento atribut jednou z klíčových hodnot pro identifikaci. Atribut je velmi unikátní pro rozlišení velkého počtu aplikací, případně pro spojení několika aplikací do jedné třídy.

Port může nabývat 65 536 celočíselných hodnot. Pro šifrovanou komunikaci se většinou využívá port 443, ale existují i jiná čísla portu pro specifické aplikace, jako je IMAP, která pro šifrování pomocí TLS využívá port 993 pro šifrování e-mailové komunikace [7]. Při identifikaci mobilní aplikace nese port jen nízkou informaci. Experimenty ukázaly, že mobilní aplikace z vytvořeného datasetu navazují TLS komunikaci vždy na portu 443, a proto i vypočítaná entropie je rovna 0. Váha pro atribut port je tedy nulová, protože se pro standardní komunikaci nepředpokládá jiná hodnota než 443.

Šifrovací algoritmy se nacházejí v paketu *Client Hello*, který nese informaci o podporovaných šifrovacích algoritmech. Pole obsahující tyto algoritmy nabývá standardizovaných

hodnot z intervalu od 0 do 65 535 (0x0 až 0xffff) [23]. Hodnoty standardizuje organizace IANA[1].

Z pole *šifrovací algoritmy* je potřeba odstranit (nebo zaměnit za konstantu) tzv. *Grease Value* [9], jež mohou výpočet otisku aplikace nepříznivě ovlivnit. Přehled všech hodnot je v tabulce 4.4.

Grease Values	
0x0a0a, 0x1a1a, 0x2a2a, 0x3a3a	0x4a4a, 0x5a5a, 0x6a6a, 0x7a7a
0x8a8a, 0x9a9a, 0xaaaa, 0xbaba	0xcaca, 0xdada, 0xeaea, 0xfafa

Tabulka 4.4: Hodnoty *Grease Values* v hexadecimální podobě.

Tabulka 4.5 obsahuje jména aplikací z datasetu, *Cipher Suites* a jejich počet. Aplikace *Booxy*, *ČHMU* a *Waze* obsahovaly několik *Grease Value* hodnot.

Tyto hodnoty byly nahrazeny za konstantu, čímž byl redukován počet řetězců pro danou aplikaci. Například pro aplikaci *Booxy* existovalo celkem deset řetězců se šifrovacími algoritmy, ve kterých se nacházely *Grease Value*, přesněji *0x1a1a, 0x2a2a, 0x5a5a, 0x8a8a, 0xaaaa, 0xdada a 0xfafa*. Všechny vyjmenované hodnoty byly nahrazeny za konstantu *0x0a0a*, a tím došlo ke zredukovaní z deseti na tři řetězce: 0xc02b, 0xc02f, 0xc00a, 0xc009, 0xc013, 0xc014, 0x9c, 0x2f, 0x35

Zajímavé z hlediska identifikace nejsou jen samotné hodnoty, ale i jejich pořadí a řetězec hodnot jako takový. Z tohoto důvodu je lepší *Grease Value* nahrazovat pomocí konstanty, než je mazat.

Pro zkoumání hodnot byla použita frekvenční analýza, která ukazuje počet jednotlivých hodnot a jejich výskyt, viz tabulka 4.6. Hodnoty *0x1301, 0x1302, 0x1303* se vyskytují pouze ve třech řetězcích, ale ve dvou rozdílných aplikacích. Hodnoty *0x2f, 0x35, 0x9c, 0xc014* a další hodnoty se nacházely ve všech řetězcích. Hodnoty, které se vyskytují méně často, jsou vhodnější pro porovnání.

Tabulka neobsahuje jen počet výskytů dané hodnoty, ale i pravděpodobnost jejich výskytu. Pravděpodobnost výskytu byla spočítána ze všech *Cipher Suite* hodnot včetně nahrazovací konstanty. Následně byla z těchto pravděpodobností vypočítána informační entropie, která nabývá přibližné hodnoty 3.906 informačního bitu pro jednotlivé hodnoty. Informační entropie je přibližně rovná hodnotě 1.574 informačního bitu pro celé řetězce. Váha pro tento řetězec šifrovacích algoritmů byla nastavena na osm, pro jednotlivé hodnoty byla váha nastavena na nula.

Verze TLS nabývá několika hodnot, a to SSLv3, TLS 1.0, TLS 1.1, TLS 1.2 a TLS 1.3. Většina aplikací v dnešní době podporuje TLS 1.1 a TLS 1.2. V hexadecimální soustavě má SSL 3.0 hodnotu 0x0300. TLS 1.0 je roven hodnotě 0x301. TLS 1.1, TLS 1.2 a TLS 1.3 jsou rovny hodnotám 0x0302, 0x0303 a 0x0304.

Veškeré zkoumané aplikace navázaly šifrované spojení dle verze TLS 1.2 [13]. Změnou TLS verze se dají rozlišit i verze aplikace. Dnes je nejvíce využíván standart TLS 1.2, ale v blízké budoucnosti lze očekávat přechod na verzi TLS 1.3 [23].

Informační entropie pro atribut verze TLS je stejně jako u atributu port rovná 0. Z tohoto důvodu nebyl tento atribut zařazen do otisku aplikace.

[1] https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml#tls-parameters-4

Aplikace	**Šifrovací algoritmy**	**Počet hodnot**
Bazoš	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	12
Benzina	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	12
BlaBlaCar	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	12
Booxy	**0x0a0a**, 0x1301, 0x1302, 0x1303, 0xc02b, 0xc02f, 0xc02c, 0xc030, 0xcca9, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	42
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc009, 0xc00a, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
ČHMÚ	**0x0a0a**, 0x1301, 0x1302, 0x1303, 0xc02b, 0xc02f, 0xc02c, 0xc030, 0xcca9, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	28
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
Kaufland	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc009, 0xc00a, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	26
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
Lidl	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc009, 0xc00a, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	26
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
Mapy	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc009, 0xc00a, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	26
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
Shazam	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	21
	0xc02b, 0xc02f, 0xc00a, 0xc009, 0xc013, 0xc014, 0x9c, 0x2f, 0x35	
Waze	**0x0a0a**, 0x1301, 0x1302, 0x1303, 0xc02b, 0xc02f, 0xc02c, 0xc030, 0xcca9, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	42
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc009, 0xc00a, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	
	0xc02b, 0xc02c, 0xcca9, 0xc02f, 0xc030, 0xcca8, 0xc013, 0xc014, 0x9c, 0x9d, 0x2f, 0x35	

Tabulka 4.5: Šifrovací algoritmy s konstantou **0x0a0a** místo *Grease Values* pro zkoumané aplikace.

Šifrovací algoritmy	Výskyt
0x1301	3/19
0x1302	3/19
0x1303	3/19
0x0a0a	**3/19**
0xc009	6/19
0xc00a	6/19
0x009d	18/19
0xc02c	18/19
0xc030	18/19
0xcca8	18/19
0xcca9	18/19
0x2f	19/19
0x35	19/19
0x9c	19/19
0xc013	19/19
0xc014	19/19
0xc02b	19/19
0xc02f	19/19

Šifrovací algoritmy	Pravděpodobnost výskytu
0x1301	3/247
0x1302	3/247
0x1303	3/247
0x0a0a	3/247
0xc009	6/247
0xc00a	6/247
0x009d	18/247
0xc02c	18/247
0xc030	18/247
0xcca8	18/247
0xcca9	18/247
0x2f	19/247
0x35	19/247
0x9c	19/247
0xc013	19/247
0xc014	19/247
0xc02b	19/247
0xc02f	19/247

Tabulka 4.6: Frekvenční analýza a pravděpodobnost výskytu pro hodnoty šifrovacích algoritmů.

Parametr List of Extensions neboli seznam rozšíření přenášený v *Client Hello* paketu. Rozšíření mohou ale nemusí být obsažena v odeslaném paketu, mezi rozšíření patří například SNI, eliptické křivky a jejich formáty. Algoritmus JA3 a JA3s využívá pouze typ těchto rozšíření. Atribut typ může nabývat celkově 2^{16} celočíselných hodnot. Počet ani pořadí rozšíření v paketu nemusí být stejné a liší se v závislosti na aplikaci. Nejčastější hodnoty jsou například 0x17, 0x23, 0xff01 [16] nebo 0, která odpovídá SNI. Jednotlivé zaznamenané hodnoty jsou v tabulce 4.7. Tabulka je tvořena celkem čtyřmi sloupci. První sloupec obsahuje jména aplikací. Ve druhém sloupci jsou typy rozšíření a následně ve třetím sloupci je jejich počet. Poslední sloupec vyjadřuje počet unikátních hodnot.

Pro tento atribut byla vypočítaná informační entropie jak pro jednotlivé prvky, tak pro celé řetězce. Pro jednotlivé prvky entropie nabývá přibližně hodnoty 3.502 informačního bitu a pro celé řetězce je hodnota přibližně rovná 2.012 informačního bitu. Váhové ohodnocení pro jednotlivé hodnoty je rovno nule, pro celé řetězce je váhové ohodnocení rovno devíti.

Váhové ohodnocení pro parametr *List of Extensions* z paketu *Client Hello* je nižší kvůli nízkému počtu unikátních hodnot. Paket může zahrnovat proměnný počet těchto typů, a proto je to vhodný parametr pro porovnávání a identifikaci.

Aplikace	Typy rozšíření	Počet	PU
Bazoš	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa	9	0
Benzina	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa	9	0
BlaBlaCar	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa	9	0
Booxy	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa, 0xaaaa, 0x12, 0x33, 0x2d, 0x2b, 0x1b, 0x4469, 0x2a2a, 0x15, 0x8a8a, 0xfafa, 0x9a9a, 0xcaca, 0x4a4a, 0xbaba,0xeaea, 0xdada, 0x6a6a, 0x1a1a, 0x3a3a, 0x29, 0xa0a, 0x5a5a	32	8
ČHMU	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa, 0xdada, 0x12, 0x33, 0x2d, 0x2b, 0x1b, 0x4469, 0x3a3a, 0x15, 0xbaba, 0x2a2a, 0x5a5a	21	0
Kaufland	0x0, 0x17, 0xff01, 0xa, 0xb, 0x5, 0xd, 0x23, 0x10	9	0
Lidl	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa	9	0
Mapy	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa, 0x15	10	0
Shazem	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa, 0x15	10	0
Waze	0xff01, 0x0, 0x17, 0x23, 0xd, 0x5, 0x10, 0xb, 0xa, 0x15, 0xaaaa, 0x12, 0x33, 0x2d, 0x2b, 0x1b, 0x3a3a, 0xbaba, 0x5a5a, 0xfafa, 0x1a1a,	21	0

Tabulka 4.7: *List of Extension* získaný z paketu *Client Hello*. PU znamená počet unikátních hodnot.

Parametr podporované skupiny (support groups) obsahuje informaci o podporovaných typech eliptických křivek pro asymetrické šifrování. Toto rozšíření bylo původně pojmenováno *elliptic curves*, ale bylo přejmenováno na *supported groups*, aby bylo obecnější pro jiné typy kryptografie. Příklad pro podporované skupiny: 00 0a 00 0a 00 08 00 1d 00 17 00 18 00 19, kde:

- 00 0a - přiřazená hodnota pro rozšíření *supported groups*.
- 00 0a - deset bytů pro rozšíření.
- 00 08 - osm bytů dat v seznamu křivek.
- 00 1d - přiřazená hodnota pro křivku „x25519“ [1].
- 00 17 - přiřazená hodnota pro křivku „secp256r1“.
- 00 18 - přiřazená hodnota pro křivku „secp384r1“.
- 00 19 - přiřazená hodnota pro křivku „secp521r1“ [12].

Získané podporované skupiny jsou v tabulce 4.8. Tabulka je tvořena čtyřmi sloupci. V prvním sloupci se nachází jména zkoumaných aplikací. Druhý sloupec obsahuje podporované skupiny a ve třetím sloupci jsou jejich počty. Poslední sloupec vyjadřuje počet unikátních hodnot.

Stejně jako u předešlého atributu i zde byla vypočítaná entropie pro jednotlivé hodnoty i pro celé řetězce. Pro jednotlivé hodnoty byla přibližně vypočtena hodnota 1.585 a pro celé řetězce byla získána hodnota 0.

Atributu *supported groups* byla přiřazena nulová váha z důvodu nízkého počtu unikátních hodnot. Podporované skupiny se nacházejí v každém paketu *Client Hello*, ale vždy se nenacházejí ve stejném počtu [15]. Z tohoto důvodu byl tento atribut vynechán z porovnání.

Aplikace	Podporované skupiny	Počet	Počet unikátních hodnot
Bazoš	0x001d, 0x0017, 0x0018	3	0
Benzina	0x001d, 0x0017, 0x0018	3	0
BlaBlaCar	0x001d, 0x0017, 0x0018	3	0
Booxy	0x001d, 0x0017, 0x0018, 0xdada, 0x9a9a, 0x4a4a, 0xeaea, 0xbaba, 0x6a6a, 0xfafa, 0xcaca, 0x1a1a	12	6
ČHMU	0x001d, 0x0017, 0x0018, 0xaaaa, 0x6a6a, 0x9a9a	6	0
Kaufland	0x001d, 0x0017, 0x0018	3	0
Lidl	0x001d, 0x0017, 0x0018	3	0
Mapy	0x001d, 0x0017, 0x0018	3	0
Shazem	0x001d, 0x0017, 0x0018	3	0
Waze	0x001d, 0x0017, 0x0018, 0x6a6a, 0xaaaa, 0xdada	6	0

Tabulka 4.8: Podporované skupiny získané z paketu Client Hello.

Parametr Formát bodů eliptických křivek Během ustanovování TLS spojení si klient a server vyměňují informace o vybraných bodech v komprimované nebo nekomprimované formě [14]. Příklad pro formát eliptických křivek:

- 00 0b - přiřazená hodnota pro rozšíření *formát bodů eliptických křivek*.
- 00 02 - 2 byty pro toto rozšíření.
- 01 - 1 byte pro seznam podporovaných formátů.
- 00 - přiřazená hodnota pro nekomprimovanou komunikaci.

Experimentálním testováním atributu formátu bodů eliptických křivek bylo zjištěno, že tento atribut nabývá pouze jedné hodnoty, a to 0. Z tohoto důvodu byl atribut vynechán z porovnání.

4.2.2 Certifikát serveru

Z paketu s názvem *Server Certificate* [24] byly vybrány následující atributy, které jsou vhodné pro identifikaci mobilní aplikace, jako je doménové jméno (dále už jen DNS) a *Common name* z certifikátu. Tento paket se odesílá ihned po zprávě *Client Hello*, který si požádal o certifikát.

Struktura certifikátu podle standardu X.509 v3 je následující:

- *Version* - verze certifikátu (celočíselná hodnota) nabývající hodnot 0-2.
- *Serial Number* - sériové číslo certifikátu (celočíselná hodnota).
- *Signature Algorithm* - podpisový algoritmus (řetězec) obsahuje identifikátor pro kryptografický algoritmus používaný CA k podpisu certifikátu.
- *Issuer* - pole obsahuje entitu, která vydala a podepsala certifikát.

- *Validity* - obsahuje datum začátku a konce platnosti certifikátu.
- *Subject* - vlastník veřejného klíče.
- *Subject Publick key info* - obsahuje informace o veřejném klíči vlastníka, a to algoritmus vytvoření veřejného klíče a veřejný klíč.
- *Issuer Unique Identifier* - unikátní identifikátor vydavatele (volitelný).
- *Subject Unique Identifier* - unikátní identifikátor vlastníka (volitelný).
- a další.

Pole *Issuer* a *Subject* jsou struktury obsahující informaci o názvu entity a o její adrese. Název entity je označen jako *Common Name* (CN). Adresa obsahuje pole jako *Country Name* (C), *Organization* (O), *Organizational Unit* (OU), *State Or Province Name* (S) a *Locality* (L). Například pro entitu *Issuer* můžou pole vypadat následujícím způsobem:

CN=RapidSSL RSA CA, OU=www.digicert.com, O=DigiCert Inc, C=US.

Příklad pro entitu *subject*:

CN=*.benzina-platby.cz.

Parametr Common name (dále už jen CN) představuje název serveru chráněného pomocí TLS certifikátu. Certifikát je platný pouze v případě, že CN odpovídá názvu hostitele. Nejčastěji je atribut v paketu reprezentován *commonName* polem ve specifikaci certifikátu X.509. CN je umístěný v hlavičce certifikátu přijatého od serveru, přesněji je součástí jména vydavatele (*issure name*. Experimentálně bylo zjištěno, že se v certifikátu nemusí vždy vyskytovat doménové jméno serveru. Může se zde vyskytovat jméno *Application Service Provider* (dále už jen „ASP“). Poskytovatel aplikačních služeb (ASP) poskytuje aplikace a související služby přes internet. CN tedy může obsahovat název poskytované služby, ASP nebo případně jakoukoliv hodnotu, viz tabulka 4.9.

Pro atribut CN byla vypočtena informační entropie, která nabývá přibližné hodnoty 5.087 informačního bitu. Kvůli vysoké hodnotě informačního bitu a unikátnosti CN ve vytvořeném datasetu byla pro atribut přiřazena váha s hodnotou 15.

Aplikace	Common name	Počet CN
Bazoš	*.bazos.cz	1
Benzina	*.benzina-platby.cz	1
BlaBlaCar	*.blablacar.cz, cdn.blablacar.com	2
Booxy	ms.cbdb.cz, cbdb.cz	2
ČHMU	pocasi-data.cz, sni.cloudflaressl.com	2
Kaufland	app.kaufland.net, www.app.kaufland.net, sync.kaufland.de, media.kaufland.com	4
Lidl	accounts.lidl.com, segments.lidlplus.com, appgateway.lidlplus.com, *.lidl-flyer.com	4
Mapy	*.mapy.cz, vectmap.mapy.cz	2
Shazam	beacon.shazam.com, config.shazamid.com, beta-amp.shazam.com	3
Waze	rtproxy-l7-am-gcp.waze.com, cres.waze.com, ads-resources-legacy.waze.com, rtproxy-row.waze.com, ads-resources.waze.com, inbox-row.waze.com, rtproxy-l7-row-gcp.waze.com, *.waze.com, ctiles-row.waze.com, social-row.waze.com, sdk.waze.com, tts.waze.com	12

Tabulka 4.9: *Common Name* získáné z paketu *Server Certificate*.

DNS (Domain Name System) je obsažen v paketu *Server Certificate*, ve kterém se nachází rozšíření s názvem *GeneralName*. Zmíněné rozšíření obsahuje doménové jméno aplikačního serveru, s nímž daná aplikace komunikuje. Například pro aplikaci Benzina toto pole nabývá hodnoty *.benzina-platby.cz.

Na rozdíl od *Common name* parametr DNS obsahuje doménové jméno serveru, ale tato hodnota nemusí být vždy k dispozici. Například pro aplikaci Booxy je *Common name* shodné s *DNS*, viz tabulka 4.10.

Entropie pro atribut DNS nabývá přibližné hodnoty 5.170 informačního bitu. Pro atribut DNS byla zvolena vysoká váha, protože hodnota informačního bitu je nad hodnotou pět. Navíc DNS je cenným atributem pro přesnou identifikaci aplikace.

Mezi atributem *CN* a *DNS* byla nalezena silná korelace, protože při změně prvního atributu se změní druhý, případně naopak. Detailnější korelační analýzou bylo zjištěno, že není potřeba do vytvářeného TLS otisku vkládat obě hodnoty, ale pouze jeden jejich podřetězec. Například pro aplikaci BlaBlaCar není potřeba ukládat atribut *CN*, který nabývá hodnot *.blablacar.cz a cdn.blablacar.com. Atribut *DNS* nabývá hodnot blablacar.cz, cdn.blablacar.com. U těchto dvou atributů si lze povšimnout, že rozdíl mezi atributem *CN* a atributem *DNS* je relativně malý.

Příklad podřetězce pro aplikaci BlaBlaCar byl extrahován pomocí rozdělení celého řetězce na podřetězce pomocí tečky, která byla použita jako oddělovač, detailnější popis algoritmu viz 1. Výsledný podřetězec je následně *blabacar*, který je jak podřetězcem pro atribut *CN*, tak pro atribut *DNS*. Zároveň je tento podřetězec unikátní v porovnání podřetězců jiných aplikací.

Doménová jména pro aplikace Benzina, Lidl a Waze obsahují nejen znak tečky, ale i znak pomlčky. Pomlčka je druhý znak, podle kterého lze doménové jména rozdělit. Pro aplikaci Benzina by dělení řetězce vypadalo následovně: DNS **.benzina-platby.cz* by se nejprve rozdělilo podle znaku tečky a podřetězec by byl *benzina-platby*. Výsledný podřetězec

Algoritmus 1: Získání podřetězců

```
1 def get_substring(data: str)→ str:
      # Například data = 'cdn.blablacar.com'
      # Rozdělí řetězec podle tečky.
2     list_of_substring = data.split(".")
      # list_of_substring = ['cdn', 'blablacar', 'com']
      # Vezme předposlední prvek z pole.
3     result = list_of_substring[-2]
      # result = 'blablacar'
4     return result
```

po dělení i podle znaku pomlčky by vypadal jako název aplikace, a to *benzina*. V práci bylo vybráno dělení pouze podle znaku tečky.

Aplikace	**Doménové jméno**	**Počet DNS**
Bazoš	*.bazos.cz	1
Benzina	*.benzina-platby.cz	1
BlaBlaCar	blablacar.cz, cdn.blablacar.com	2
Booxy	ms.cbdb.cz, cbdb.cz	2
ČHMU	sni.cloudflaressl.com	2
Kaufland	app.kaufland.net, www.app.kaufland.net, sync.kaufland.de, media.kaufland.com	4
Lidl	accounts.lidl.com, segments.lidlplus.com, appgateway.lidlplus.com, *.lidl-flyer.com	4
Mapy	*.mapy.cz, vectmap.mapy.cz	2
Shazam	beacon.shazam.com, amp.shazam.com	2
Waze	rtproxy-l7-am-gcp.waze.com, cres.waze.com, ads-resources-legacy.waze.com, rtproxy-row.waze.com, ads-resources.waze.com, inbox-row.waze.com, rtproxy-l7-row-gcp.waze.com, *.waze.com, ctiles-row.waze.com, social-row.waze.com, sdk.waze.com, tts.waze.com	12

Tabulka 4.10: Doménová jména získaná z paketu *Server Cerificate*.

Velikost paketů může sloužit jako upřesňující charakteristika. Při experimentování byla zjištěna určitá unikátnost, viz tabulka 4.11. V tabulce se nacházejí velikosti paketů, kterých může být více v závislosti na počtu komunikujících serverů. Velikosti paketů se při experimentech neměnily, proto lze využít tuto metriku pro identifikaci aplikace.

V případě vystavení nového certifikátu bude uložená hodnota neplatná, z tohoto důvodu se konkrétní hodnota nebude přímo porovnávat. Níže bude popsán robustnější způsob využití velikosti paketů při porovnávání dvou otisků.

Aplikace	Velikosti paketů [B]
Bazoš	1028
Benzina	1268
BlaBlaCar	1042, 858
Booxy	413
ČHMÚ	1155
Kaufland	1503, 1504
Lidl	548, 1354, 1513
Mapy	1003, 978, 1009
Shazam	787, 1281, 1014, 1291
Waze	399

Tabulka 4.11: Velikosti paketů se *Server Certificate* pro jednotlivé aplikace.

4.3 Statistická data

V této podkapitole budou zkoumány statistické vlastnosti šifrované komunikace. Nejprve zde bude rozebrán počet a velikost paketů pro komunikaci o délce pěti vteřin, následně pro prvních deset paketů, kde bude kladen důraz na průměrné velikosti paketů a počet odeslaných paketů klientem na server a opačně.

Pro jednotlivé aplikace z datasetu platí, že se liší v počtu a velikosti zasílaných a přijatých paketů. Pokud aplikace poskytuje audio-vizuální obsah, tak bude z pravidla zasílat více paketů než aplikace poskytující textová data. Velikost paketů v síťovém toku se také může lišit v závislosti na typu aplikace. Z těchto důvodů je možné použít statistická data na rozlišení a případně i na detekci aplikací.

Nejprve bylo zapotřebí nasbírat vhodná data pro analýzu, viz podkapitola 5.1. Tato data byla následně analyzována pomocí nástroje TShark[2] a vlastních skriptů.

4.3.1 Navázání spojení z hlediska času

Nejprve bude rozebrán začátek komunikace, který se nachází v prvních pěti vteřinách. Během těchto prvních vteřin se ustanoví šifrované spojení mezi klientem a servem. Nejprve se zašle *Client Hello* a *Server Hello*. Následně server zašle svůj certifikát v paketu obsahujícím *Certification, Certification Status, Server Key Exchange a Server Hello Done* a klient odpoví paketem s obsahem *Client Key Exchange, Change Cipher Spec a Encrypted Handshake Message*. Celé ustanovení proběhne během 160-300 ms. Následně začíná zašifrovaná komunikace mezi klientem a serverem. V prvních několika paketech zašifrované komunikace se posílají přihlašovací údaje, synchronizační zprávy a důležité informace pro aplikaci, jako je například časová zóna a GPS souřadnice.

Pro analýzu bylo vybráno celkem deset mobilních aplikací[3] viz tabulka 4.12. Tabulka je rozdělena na dvě části *Klient* a *Server*, protože při spojení obou stran dohromady by se mohla ztratit důležitá informace. První část neboli část *Klient* obsahuje informace o paketech zaslaných od klienta na server. Část *Server* obsahuje pakety zaslané ze serveru na klienta. Obě části obsahují několik sloupců, a to sloupce *počet* obsahující informace o průměrném počtu zaslaných paketů během pěti vteřin. Atribut *velikost* označuje průměrnou velikost zaslaného paketu. *Min* a *max* pak označují nejmenší/největší velikost paketů v ko-

[2] TShark - https://tshark.dev/

[3] Konkrétní verze jednotlivých aplikací, viz tabulka 5.1.

munikaci. Poslední sloupce vyjadřují poměr mezi odeslanými a přijatými pakety/velikosti paketů. Pro každou aplikaci byly zkoumány minimálně tři a maximálně pět datových toků. Všechny velikosti jsou v tabulce uvedeny v bytech.

Z tabulky vyplývá, že velikost a počet paketů zaslaných během prvních pěti vteřin nejsou dostatečně unikátními parametry pro jednoznačnou identifikaci aplikace, ale mohou výrazně přispět ke správné identifikaci. Počet se může měnit i v závislosti na používání aplikace. Vhodným příkladem jsou aplikace s multimediálním obsahem, kde si uživatel může vyhledávat informace, pročítat textový obsah nebo zobrazovat audiovizuální data, která aplikace poskytuje. V případě vyhledávání a čtení textových informací bude aplikace zasílat méně dat, než když uživatel začne stahovat audiovizuální tvorbu. Minimální a maximální průměr velikosti paketů určuje interval, který zpřesňuje identifikaci aplikace. Na základě těchto dat je možné zařadit jednotlivé aplikace do několika tříd dle počtu a velikosti zaslaných paketů za určitý časový úsek. Na základě těchto měření byla přiřazena nižší váha pro rozpoznávání konkrétní aplikace.

Velikost i počet paketů se mezi aplikacemi liší. Průměrná velikost paketů není přímo závislá na počtu zaslaných paketů. Počet zaslaných paketů je přímo úměrný počtu odeslaných paketů od klienta. Celý záznam paketů pro jednotlivé aplikace je uveden v příloze C.

Klient					
Průměrné hodnoty paketů odeslaných na server					
Aplikace	**Počet**	**Velikost [B]**	**Min [B]**	**Max [B]**	**Poměr odeslaných a přijatých paketů**
Bazoš	42.8	188.2	176.7	197.6	42.8/116.3
Benzina	19	625.2	625.2	625.2	19/31.5
BlaBlaCar	26.5	317.7	312.9	322.4	26.5/39
Booxy	53	253	252.2	253.8	53/88.5
ČHMU	14.5	204.7	197.1	212.2	14.5/40
Kaufland	29.5	396.2	385.7	406.8	29.5/75.5
Lidl	70	592.1	545.8	638.5	70/106.5
Mapy	94	415.2	394.7	435.6	94/134.5
Shazam	52.5	305.7	273.8	337.7	52.5/104.5
Waze	109.5	331.4	318.4	344.5	109.5/1537

Server					
Průměrné hodnoty paketů přijatých ze serveru					
Aplikace	**Počet**	**Velikost [B]**	**Min [B]**	**Max [B]**	**Poměr odeslaných a přijatých bytů**
Bazoš	116.3	1088.3	1011.3	1162.3	188.2/1088.3
Benzina	31.5	783.3	773.3	793.2	625.2/783.3
BlaBlaCar	39	614.6	587.7	641.4	317.7/614.6
Booxy	88.5	629.4	543.6	715.3	253/629.4
ČHMU	40	780.7	546.2	1015.1	204.7/780.7
Kaufland	75.5	983.8	816.6	1150.9	396.2/983.8
Lidl	106.5	976.5	907.7	1045.2	592.1/976.5
Mapy	134.5	1250.4	617.2	1883.7	415.2/1250.4
Shazam	104.5	914.2	863.8	964.6	305.7/914.2
Waze	1537	1337.3	1325.9	1348.7	331.4/1337.3

Tabulka 4.12: Začátek ustanovení komunikace v závislosti na čase. Tabulka obsahuje zprůměrované hodnoty ze všech běhů aplikací.

4.3.2 Prvních deset paketů komunikace

Ve druhé tabulce jsou počty zaslaných a odeslaných paketů omezeny na celkový počet deseti paketů, viz tabulka 4.13. Prvních několik paketů (přibližně pět) obsahuje *Client Hello*, *Server Hello* a pakety pro výměnu certifikátů, klíčů a nastavení šifrování. Následující zbytek paketů, jak již bylo zmíněno, obsahuje aplikační data, jako jsou přihlašovací údaje, synchronizační/aplikační data atd. Na základě metod Flowprint, Maldetect a kontextové metody bylo zjištěno, že prvních deset paketů je dostačujících pro identifikaci aplikace. Pomocí experimentů byl počet ověřen a byl klasifikován jako dostačující pro identifikaci aplikace. Tyto pakety se skládají z 50 % z paketů navazujících TLS spojení a přibližně stejného počtu paketů obsahujících zašifrovaný obsah. Následující pakety už by mohly být ovlivněny uživatelem a nejsou pro identifikaci aplikace významné.

Klient					
Průměrné hodnoty paketů odeslaných na server					
Aplikace	**Počet**	**Velikost [B]**	**Min [B]**	**Max [B]**	**Poměr odeslaných a přijatých paketů**
Bazoš	4.3	265.8	212.5	304.5	4.3/5.8
Benzina	4	625.2	610.5	610.5	4/6
BlaBlaCar	4.5	203.9	192.8	215	4.5/5.5
Booxy	4	233.4	200	266.8	4/6
ČHMU	4	191.8	191.3	192.5	4/6
Kaufland	4	236.5	224.7	248.4	4/6
Lidl	4.5	307.1	301.2	313	4.5/5.5
Mapy	4.5	250.5	250.2	250.8	4.5/5.5
Shazam	4.5	295.6	244.8	346.6	4.5/5.5
Waze	3.5	401.5	370.3	432.7	3.5/6.5

Server					
Průměrné hodnoty paketů přijatých ze serveru					
Aplikace	**Počet**	**Velikost [B]**	**Min [B]**	**Max [B]**	**Poměr odeslaných a přijatých bytů**
Bazoš	5.8	770.3	624.2	974.2	265.8/770.3
Benzina	6	944.3	944	944.7	625.2/944.3
BlaBlaCar	5.5	774.4	624.6	924.2	203.9/774.4
Booxy	6	566.8	449.3	684.3	233.4/566.8
ČHMU	6	893.8	730.7	1057	191.8/893.8
Kaufland	6	1172	998.6	1345.4	236.5/1172
Lidl	5.5	867.5	810.2	924.8	307.1/867.5
Mapy	5.5	649.1	584.2	714	250.5/649.1
Shazam	5.5	751.8	618	885.6	295.6/751.8
Waze	6.5	1103.7	1065.8	1141.6	401.5/1103.7

Tabulka 4.13: Prvních deset paketů komunikace. Tabulka obsahuje zprůměrované hodnoty ze všech běhů aplikací.

Počet odeslaných paketů od klienta je nižší než počet odeslaných ze serveru. Tato skutečnost je dána tím, že klient posílá několik málo paketů, které mohou obsahovat důležité informace pro server. Ten mu odpoví potřebnými daty, jichž bývá zpravidla více. Stejně tak je tomu s průměrnou velikostí paketů.

Z tabulky vyplývá, že poměr odeslaných a přijatých paketů u zkoumaných aplikací se často opakuje, například u aplikací *Lidl, Mapy, BlaBlaCar* a *Shazam* je poměr roven $\frac{4.5}{5.5}$. Poměr přenesených bytů je zajímavější atribut, protože při zkoumání zde nedošlo k úplnému přikrytí hodnot. Z minimální a maximální velikosti paketu lze vytvořit interval, do něhož by s velkou pravděpodobností měla spadnout průměrná velikost odeslaných dat. Intervaly musí být dva, a to jak pro klientskou, tak pro serverovou stranu. Z tohoto důvodu byla udělena metrice střední váha pro identifikaci aplikace.

Z obou tabulek vyplývá, že samotné velikosti a počty paketů nestačí pro přesnou identifikaci jednotlivých aplikací, ale mohou poskytnout důležité informace pro rozhodování. Oproti tabulce 4.12 je tabulka s omezeným počtem paketů více stabilní, protože prvních

deset paketů není ovlivněno užíváním aplikace uživatelem. Celý záznam paketů se nachází v příloze B.

4.3.3 Velikost paketů při ustanovení TLS spojení

Paket *Client Hello* je menší než součet *Server Hello* a zaslaný certifikát. Stejný poměr platí i pro celkový počet odeslaných bytů, při ustanovení TLS spojení je menší než počet přijatých bytů. Například pro aplikaci *Booxy* platí, že součet velikosti paketů pro klienta je přibližně osmkrát menší než součet pro server. Výpočet hodnoty pro aplikaci *Booxy* se nachází v tabulace 4.14,

Z tabulky vyplývá, že počet odeslaných paketů se liší mezi klientskou a serverovou stranou, ale u stejné aplikace zůstává relativně stejný. Velikost jednotlivých paketů mají aplikace mezi sebou rozdílné, a proto se dá použít pro identifikaci aplikací. Z tohoto důvodu byla velikosti paketů při ustanovení TLS spojení udělena vyšší váha.

Směr	Paket	Velikost [B]
Klient->Server	Client Hello	241
Server->Klient	Server Hello	1514
Server->Klient	Server Certificate, ...	1501
Klient->Server	Client Key Exchange, ...	159
Server>Klient	Change Cipher Spec, Encryp ...	117

Součet pro klienta:	400
Součet pro server:	3132

Tabulka 4.14: Příklad velikosti paketů při ustanovení TLS komunikace.

Tabulka 4.15 obsahuje velikosti jednotlivých paketů z ustanovení TLS spojení v bytech pro všechny zkoumané aplikace. Sloupce s označením CH obsahují velikost paketu *Client Hello*. SH znamená *Server Hello*, sloupec s označením SC obsahuje velikost paketu *Server Certificate*. CKE je označení pro *Client Key Exchange* paket. Následující sloupec je určen pro *Change Cipher Spec*. Další dva sloupce obsahují součty velikosti paketů pro klientskou a serverovou stranu. Poslední sloupec obsahuje poměr mezi klientskou a serverovou stranou. Jednotlivá ustanovení TLS spojení pro všechny zkoumané aplikace jsou v příloze A. Z tabulek vyplývá, že velikost odeslaných paketů je několikrát menší než počet přijatých paketů.

V příloze A.1 lze vidět nízkou variabilitu uvnitř třídy neboli výsledky měření pro jednu aplikaci jsou relativně neměnné. Například pro aplikaci *Bazoš* jsou hodnoty pro jednotlivá spuštění stejné. Výjimkou je pak aplikace *Mapy*, u níž se při třetím spuštění lišil poměr zaslaných a poměr odeslaných velikostí paketů. Důvodem byla větší velikost paketu obsahující *Server Certificate*. U jednotlivých naměřených hodnot nedochází k překrytí všech hodnot jedné aplikace s hodnotami z ostatních aplikací. Z tohoto důvodu je tato metrika vhodná k identifikaci aplikace.

Aplikace	CH	SH	SC	CKE	CCS	Σ Klient	Σ Server	Poměr
Bazoš	243	1514	1514	828	159	1071	3187	2.98
Benzina	252	1514	1268	192	117	444	2899	6.53
BlaBlaCar	248	1484	858	159	358	407	2700	6.63
Booxy	241	1514	1501	159	117	400	3132	7.83
ČHMU	238	1514	1321	147	312	385	3147	8.17
Kaufland	225	1514	1514	1514	334	1739	3362	1.93
Lidl	249	1514	548	216	348	465	2410	5.18
Mapy	246	1294	1003	159	324	405	2621	6.47
Shazam	248	1484	1014	159	358	407	2856	7.02
Waze	242	1484	1484	213	159	455	3127	6.87

Tabulka 4.15: Velikosti paketů při ustavení TLS spojení. CH - *Client Hello*, SH - *Server Hello*, NS - *New Session*, SC - *Server Certificate*, CCS - *Change Cipher set.*

Pro jednotlivé typy paketů z tabulky 4.15 byla vypočtena informační entropie. Nejvyšší hodnotu informačního bitu má *Client Hello* paket, naopak nejnižší hodnotu informační bitu má paket obsahující *Change Cipher Spec.* Přehled všech hodnot viz tabulka 4.16.

Název atributu	**Informační entropie [informační bit]**
Velikost paketu *Client Hello*	3.585
Velikost paketu *Server Hello*	3.418
Velikost paketu *Server Certificate*	3.459
Velikost paketu *Change Ciper set.*	2.126
Velikost paketu *New Session*	2.948
Minimální velikost ustanovení spojení ze strany klienta	3.585
Maximální velikost ustanovení spojení ze strany klienta	3.252
Minimální velikost ustanovení spojení ze strany serveru	3.342
Maximální velikost ustanovení spojení ze strany serveru	3.189

Tabulka 4.16: Informační entropie pro jednotlivé typy TLS paketů.

4.4 Vybrané atributy a metriky

Výše zmíněným atributům nebo metrikám byla přidělena váhová ohodnocení odvozená od vypočtené informační entropie, která nabývá určitých hodnot. Na základě hodnoty informačního bitu byly jednotlivé atributy rozděleny do tří skupin. Pokud hodnota informačního bitu přesáhla hodnotu pět, tak byl daný atribut umístěn do skupiny s vysokým významem. Atributy nebo metriky s hodnotou informačního bitu mezi 3–5 byly zařazeny do skupiny se

středním významem. Do skupiny s nízkým významem jsou zařazeny atributy s hodnotou informačního bitu mezi 1–3. Ostatní atributy nebyly zařazeny do žádné skupiny, protože jejich hodnota pro detekci aplikace nebyla významná. Kompletní přehled všech atributů, jejich hodnot informačního bitu, přiřazené skupiny a odvozené váhy se nachází v tabulce 4.17.

Mezi nejzajímavější atributy neboli atributy s největší hodnotou informační entropie patří IP adresa komunikujícího serveru, název serveru (SNI), *Common Name* a *DNS*.

Mezi vyjmenovanými atributy byla zjištěna silná kladná korelace pomocí korelační analýzy, která zdůrazňuje propojení dvojice atributů. Například při změně IP adresy se změní i SNI, stejné tvrzení platí pro *CN* a *DNS*.

Korelace mezi IP adresou a SNI umožňuje odstranit jeden člen z dvojice atributů a druhému zvýšit váhové ohodnocení. Druhou možností je neukládat celý řetězec hodnoty atributu *CN* a *DNS*, ale ukládat pouze jejich společný podřetězec. Jak již bylo zmíněno u atributu DNS, lze pro aplikaci BlaBlaCar uschovávat pouze podřetězec *blablacar*, který je podřetězcem pro *CN* a *DNS* atributů pro tuto aplikaci a zároveň je unikátní oproti podřetězcům ostatních aplikací.

Entropie nebyla vypočítána jen pro atributy, ale také i pro jednotlivé metriky. Nejvyšší informační hodnotu entropie má metrika, která se zaměřuje na prvních deset TLS paketů. Přehled všech atributů a vypočtených entropií se nachází v tabulce 4.16. Atributy a metriky s vysokou hodnotou informačního bitu jsou nejvíce vhodné pro přesnou identifikaci aplikace a mají tak největší vliv na celkové rozhodnutí.

Atributy se střední váhou nejsou vhodné jako unikátní hodnoty pro identifikaci aplikace, ale mohou velmi přispět pro správné zařazení aplikace do třídy a případně zpřesnit proces identifikace. Mezi tyto atributy se řadí řetězce hodnot pro šifrovací algoritmy, řetězce hodnot pro rozšíření TLS a velikost přijatých/odeslaných paketů při ustanovování TLS spojení.

Atributy označené nízkou hodnotou informačního bitu jsou vhodné jen pro dodatečné zpřesnění identifikace aplikace. Zástupci těchto atributů jsou velikosti *Change Cipher Spec.* a *New Session* paketů.

Zbylé extrahované atributy mají hodnotu informačního bitu rovnou nule nebo relativně vysoké hodnoty informačního bitu, ale při porovnání těchto atributů nastávalo velké množství falešně úspěšných porovnávání. Z tohoto důvodu byly odstraněny z porovnávání vytvořených TLS otisků.

Hodnota informačního bitu se rovná nule znamená, a to znamená, že všechny extrahované hodnoty daného atributu byly stejné, a tudíž nevhodné k porovnávání, například atribut port, který nabýval vždy hodnoty 443. Příklady jednotlivých hodnot se nacházejí v tabulce 4.18.

Název atributu	**Extrahované hodnota**
Port	443
TLS verze	0x0303
Podporované skupiny	0x001d, 0x0017, 0x0018 (0x0a0a)
Formát bodů eliptických křivek	0

Tabulka 4.18: Atributy s hodnotou informačního atributu rovnou nule.

Název atributu	**Informační entropie [informační bit]**	**Přiřazená skupina**	**Odvozená váha**
IP adresa	5.585	Vysoká	15
SNI	5.322	Vysoká	15
Port	0	X	0
Šifrovací algoritmy pouze hodnoty	3.906	X	0
Šifrovací algoritmy celé řetězce	1.574	Střední	8
Verze TLS	0	X	0
List of Extensions pouze hodnoty	3.502	X	0
List of Extensions celé řetezce	2.012	Střední	9
Podporované skupiny pouze hodnoty	1.585	X	0
Podporované skupiny celé řetězce	0	X	0
Formát bodů eliptických křivek	0	X	0
Common Name	5.087	Vysoká	15
DNS	5.170	Vysoká	15
Prvních 10 paketů exaktní porovnání velikostí	6.267	Vysoká	18
Prvních 10 paketů hodnotové porovnání velikostí paketů	5.198	Vysoká	
Velikost paketu *Client Hello*	3.585	Střední	1.2
Velikost paketu *Server Hello*	3.418	Střední	1.2
Velikost paketu *Server Certificate*	3.459	Střední	1.2
Velikost paketu *Change Cipher Spec.*	2.126	Nízká	0.7
Velikost paketu *New Session*	2.948	Nízká	0.7
Minimální velikost ustanovení spojení ze strany klienta	3.585	Střední	0
Maximální velikost ustanovení spojení ze strany klienta	3.252	Střední	
Minimální velikost ustanovení spojení ze strany serveru	3.342	Střední	0
Maximální velikost ustanovení spojení ze strany serveru	3.189	Střední	

Tabulka 4.17: Informační entropie a rozdělení vah pro jednotlivé atributy/metriky.

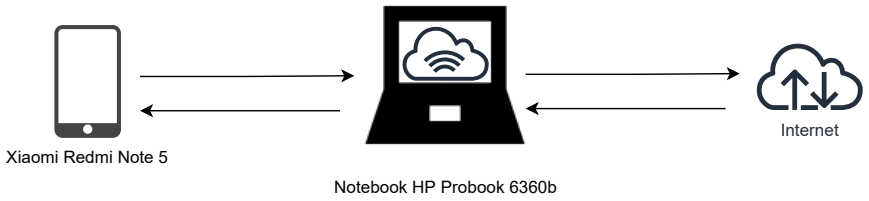
Kapitola 5

Architektura detekčního systému

Kapitola bude věnována návrhu a implementaci architektury pro získání otisků aplikací ze zašifrované komunikace. Nejprve zde bude rozebraná topologie pro získávání záznamů mobilních aplikací. Následně zde bude popsán postup získávání jednotlivých záznamu z mobilního zařízení pomocí programu *Wireshark*. Ve druhé části kapitoly bude popsána technika převodu jednotlivých záznamů na dataset pomocí programovacího jazyka Python3 a knihovny PyShark[1]. Ve třetí části kapitoly bude detailně rozebrán postup pro porovnání TLS otisků. Další část kapitoly bude popisovat skript, který na základě jednotlivých atributů v TLS otisku dokáže identifikovat známé aplikace. Poslední část kapitoly bude zaměřena na aktualizaci a na přidání nového otisku z neznámé síťové komunikace.

5.1 Vytváření datasetu

Pro vytvoření datasetu byl Notebook HP Probook 6360b připojen pomocí síťového kabelu UTP CAT5e do sítě KolejNet VUT Brno. Na notebooku byl po dobu zaznamenávání síťového provozu vytvořen *Hotspot*, ke kterému byl připojen telefon Xiaomi Redmi Note 5 se 4 GB operační pamětí a 64 GB vnitřní pamětí s operačním systémem Android 9 PKQ1.180904.001 s nástavbou MIUI global 12.0.2 (stabilní), viz obrázek topologie 5.1. Na notebooku byl spuštěn program Wireshark, který umožňuje získávat síťovou komunikaci z konkrétního síťového rozhraní. Na rozhraní byl připojen pouze zmíněný mobil pro čistější sběr dat.



Obrázek 5.1: Topologie pro zachycení komunikující aplikace.

Pro každou aplikaci byly vytvořeny tři záznamy (soubory s koncovkou .pcapng) pomocí programu Wireshark. Soubory byly následně pojmenovány pomocí jména aplikace, podřetězce doménového jména a čísla záznamu (X nabývá hodnot od nula do dvou). Jednotlivé

[1] Knihovna PyShark - http://kiminewt.github.io/pyshark/

Název aplikace	Verze aplikace	Názvy souborů
Bazoš	2.9.0	bazos_bazos_X.pcapng
Benzina	4.2.1	benzina_benzina_X.pcapng
BlaBlaCar	5.80.0	blablacar_blablacar_X.pcapng
Booxy	4.5.5	booxy_cbdb_X.pcapng
ČHMÚ	1.8	chmu_data.pocasi_X.pcapng
Kaufland	2.20.0	kaufland_kaufland_X.pcapng
Lidl	14.36.0	lidl_lidl_X.pcapng
Mapy.cz	8.5.0	mapy_mapy_X.pcapng
Shazam	11.21.0-210409	shazam_shazam_X.pcapng
Waze	4.73.0.3	waze_waze_X.pcapng

Tabulka 5.1: Dataset obsahuje celkem 10 aplikací, které byly na telefonu již nainstalovány.

Název aplikace	Doba trvání záznamu [s]	Velikost [B]	Počet paketů	Počet TLS paketů
Bazoš	1 105	58 546	70 317	14 121
Benzina	74	3 138	4 249	1 188
BlaBlaCar	393	11 850	19 950	8 035
Booxy	919	24 188	39 602	11 742
ČHMÚ	109	5 079	7 888	710
Kaufland	79	11 452	14 235	2 595
Lidl	33	1 878	3 257	952
Mapy.cz	35	2 971	4 247	1 633
Shazam	125	63 714	66 274	5 988
Waze	51	13 520	16475	8 142

Tabulka 5.2: Statistická data k jednotlivým aplikacím.

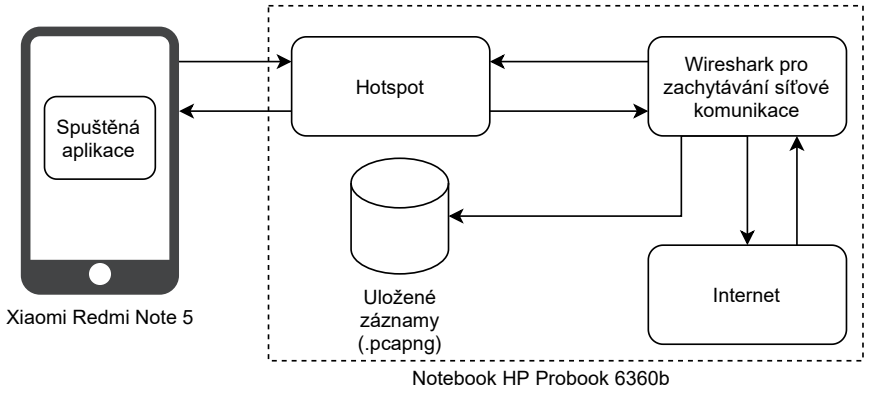
části byly odděleny pomocí podtržítka. Například aplikace Kaufland, která se nachází v tabulce 5.1, má verzi 2.20.0 a zaznamenané soubory s názvem kaufland_kaufland_X.pcapng.

Tabulka 5.2 obsahuje celková statistická data .pcap souborů. Jednotlivé sloupce obsahují součty jednotlivých parametrů ze všech souborů pro danou aplikaci.

Aplikace *ČHMÚ* má nejnižší počet přijatých a odeslaných TLS paketů. Aplikace většinu svých dat zasílá pomocí nešifrovaného protokolu UDP. Zašifrovanou komunikaci využívá jen pro synchronizační, autentizační, lokalizační a další osobní data. Nejvyšší počet vyměněných paketů má aplikace *Bazoš*, která při experimentálním měření zasílá vysoké procento TLS paketů.

5.1.1 Vytvoření záznamu aplikace

Pro vytvoření jednoho záznamu bylo zapotřebí nejprve restartovat mobilní telefon, následně telefon připojit k notebooku pomocí bezdrátového připojení. Po připojení začal telefon komunikovat přes notebook s celosvětovou sítí internet. Před vytvořením záznamu pro konkrétní aplikaci byl telefon ponechán pět až deset minut bez jakékoliv interakce od uživatele.



Obrázek 5.2: Digram získávání záznamu pro jednu aplikaci.

Po uplynutí tohoto intervalu bylo v programu *Wireshark* zapnuto zaznamenávání síťové komunikace pro rozhraní komunikující s mobilním zařízením. Následně byla na telefonu spuštěna konkrétní aplikace, v níž byla provedena interakce dle typu aplikace. Například pro aplikaci *Booxy* byly vyhledány knížky a zaznamenán pokrok čtení pro jednotlivé vyhledané knihy.

5.1.2 Převod záznamů na dataset

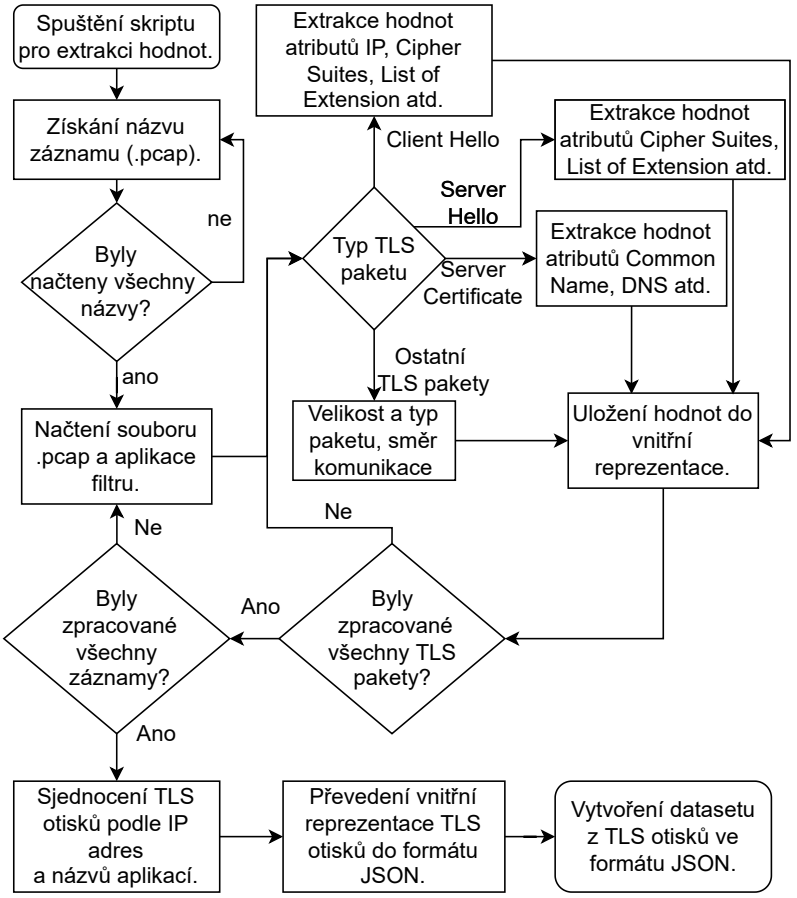
Pro převod záznamů na dataset bylo zapotřebí vytvořit skript, který ze získaných záznamů získá pouze TLS pakety. Skript byl napsán v programovacím jazyce Python3 s využitím knihovny PyShark. Po spuštění skript vytvoří seznam aplikací podle pojmenování záznamů ve složce *pcaps* a přiřadí k nim cestu k souboru se záznamem konkrétní aplikace.

Po sestavení slovníku, kde klíč je název dané aplikace a hodnoty jsou cesty k jednotlivým .pcap souborům, se soubory začnou postupně zpracovávat. Na každý soubor je použit filtr a filtrovací řetězec při nahrávání do paměti. Filtrovací řetězec obsahuje typy TLS paketů a seznam IP adres dle konkrétní aplikace. Filtr společně s filtrovacím řetězcem následně zaručí, že se do paměti načtou pouze specifické TLS pakety.

```
class Fingerprint:
    def __init__(self, name: str, files: List[str]):
        self.name: str = name # Název aplikace
        self.files: List[str] = files # Pcap soubory
        self.ciphersuite: List[List[int]] # Parametr Cipher Suites
        self.dst_ips: List[str] # Parametr IP adresa
        self.extension: List[List[int]] # Parametr List of extensions
        self.server_name: List[List[str]] # Parametr SNI
        self.support_groups: List[List[int]] # Parametr Support Groups
        self.CN: List[str] # Parametr Common Name
        self.DNS: List[str] # Parametr DNS
        # Parametr velikosti prvních deset paketů
        self.len_ten_packets: List[List[int]]
        # Parametr velikost ustanovení TLS spojení
        self.len_handshake: Dict[str, int]
        self.JA3: List[str] # Algoritmus JA3 (MD5 heš)
        self.JA3s: List[str] # ALgoitsmus JA3s (MD5 heš)
```

Výpis 5.1: Datová struktura vnitřní reprezentace TLS otisku.

Další funkcí skriptu je zpracovaní jednotlivých paketů. Na základě prvního zpracovaného paketu se vytvoří objekt pro reprezentaci TLS otisku aplikace 5.1, do kterého se budou vkládat extrahované atributy. Pro každý datový tok je vytvořena jedna vnitřní reprezentace TLS otisku, to znamená, že pokud klientská aplikace komunikuje s několika servery, tak pro každý server je vytvořen nový otisk. Takto vytvořený otisk zná jen cestu k aktuálnímu souboru a IP adresu komunikujícího serveru. Podle typu zpracovaného TLS paketu se určí, jaké atributy z daného paketu budou extrahovány. Například z paketu *Client Hello* se extrahuje cílová IP adresa, cílový port, verze TLS, *Cipher Suites*, *List of Extension*, doménové jméno serveru, *Support Groups* a další, viz kapitola 4. Z paketu *Server Certificate* se získávají atributy *Common name* a *DNS*. Z každého paketu je mimo jiné získána i jeho velikost, které se také ukládá do vnitřní reprezentace TLS otisku.



Obrázek 5.3: Vývojový diagram pro převod záznamů na dataset.

Pro možnost srovnání aktuálního řešení identifikace aplikací ze zašifrované síťové komunikace s jiným řešením byl přidán do procesu extrakce algoritmus JA3 a JA3s. Výsledný MD5 heš z algoritmus JA3 a MD5 heš z algoritmu JA3s jsou uloženy do vnitřní reprezentace TLS otisku.

Po extrakci všech atributů a velikostí z jednotlivých paketů se vnitřní reprezentace spojí do jedné vnitřní reprezentace objektu s názvem dané aplikace. Některé atributy jsou pouze překopírované a uložené do pole, například IP adresa komunikujících serverů. Atributy obsahující hodnoty *Grease Value* jsou před uložením do jednoho pole ještě předzpracovány. Předzpracování probíhá tak, že všechny *Grease Value* jsou nahrazeny konstantou 0x0a0a,

aby nedošlo ke zkreslení informací při porovnávání. Z velikostí jsou následně vypočtené minimální a maximální velikosti TLS paketů v bytech, které se také uloží do výsledné reprezentace TLS otisku.

Po vytvoření TLS otisků s názvem příslušné aplikace se tento proces opakuje i pro ostatní aplikace. Po ukončení fáze extrahování atributů a spojování vnitřních reprezentací otisků následuje část vytvoření datasetu. V programovacím jazyce Python se dají získat atributy a jejich hodnoty z objektu ve formátu JSON pomocí jednoduché operace. Výsledný dataset se skládá právě z těchto řetězců JSON a je uložen do stejné složky, odkud byl spuštěn skript na extrahování atributů a vytvoření datasetu ze síťové komunikace. Toto rozhodnutí dovoluje porovnávat otisky aplikací po jednotlivých hodnotách a ne pouze exaktní porovnání, jak je tomu v případě algoritmu JA3 a JA3s.

Ukázka TLS otisku pro aplikaci Bazoš:


```
{"name": "bazos", "files": ["bazos_bazos_1_.pcapng", ...], "dst_ips":
["88.86.119.246"], "ciphersuite": [[49195, 49196, 52393, 49199, 49200,
52392, 49171, 49172, 156, 157, 47, 53]], "extension": [[65281, 0, 23, 35,
13, 5, 16, 11, 10]], "server_name": ["www.bazos.cz"], "len_ten_packets":
[[159, 243, 243, 243, 340, 828, 1514, 1514, 1514, 1514], ...],
"len_handshake": "1": [243], "2": [1514], "4": [340], "11": [1514, 1028],
"16": [159], "CN": ["*.bazos.cz"], "dns": ["*.bazos.cz"],
"JA3": ["6f5e62edfa5933b1332ddf8b9fb3ef9d"],
"JA3s": ["5deff56d75697caa937c30eae4aeaf84"]}
```


5.2 Porovnání jednotlivých atributů

Pro identifikace aplikací ze zaznamenané šifrované komunikace je potřeba porovnat nově vytvořené TLS otisky aplikací s TLS otisky v datasetu. Samotný TLS otisk se skládá z několika atributů, které se porovnávají způsoby uvedenými dále i s příklady porovnání.

5.2.1 Parametr IP adresa

Jako první atribut se porovnává IP adresa komunikujícího serveru. Nově vytvořený otisk obsahuje jednu adresu, na rozdíl od otisku aplikace v datasetu, kde pro jeden otisk aplikace je zpravidla více IP. Při porovnání IP adres se provádí exaktní porovnání IP adresy se seznamem IP adres v datasetu, příklad porovnání viz tabulka 5.3. Tabulka obsahuje tři reálné příklady porovnání IP adres.

Jako první se porovnává IP adresa 91.231.171.211, která náleží známé aplikaci Lidl. IP adresy se porovnávají exaktně, proto se s IP adresou neprovádí předzpracování a rovnou se začne vyhledávat v datasetu. Výsledná shoda je jedna, protože aplikace byla známá a v datasetu se povedlo najít stejnou IP adresu. Druhá porovnávaná IP adresa náleží aplikaci Facebook, která se nenachází v datasetu, a proto bylo porovnávaní neúspěšné. Třetí IP adresa náleží reklamní společnosti. Reklamní společnosti byly při vytváření otisku aplikací odstraněny, takže porovnávání bylo neúspěšné.

Extrahovaná IP adresa z neznámé komunikace	**Porovnávaná IP adresa**	**IP adresy v datasetu**	**Výsledná shoda**
91.231.171.211	91.231.171.211	[**91.231.171.211**, 91.231.171.212]	1
157.240.30.18	157.240.30.18	[91.231.171.211, 91.231.171.212]	0
40.70.161.102	40.70.161.102	185.183.8.127	0

Tabulka 5.3: Příklad porovnání pro parametr IP adresa.

5.2.2 Parametr SNI

Druhým porovnávaným atributem je SNI, který není porovnáván exaktním způsobem, jako tomu je v případě IP adresy, ale porovnává se pouze jeho podřetězec. Podřetězec je získán pomocí rozdělení řetězce na několik podřetězců, a to použitím tečky jako oddělovače. Ze získaných podřetězců je vybrán ten, který obsahuje doménové jméno serveru. Výsledné doménové jméno se nakonec porovná s hodnotou SNI uloženou v datasetu. Získané doménové jméno se postupně přikládá ke každé hodnotě SNI v datasetu. Pokud se celý podřetězec nachází v řetězci SNI otisku aplikace v datasetu, tak je porovnávání úspěšné. Příklad porovnání SNI se nachází v tabulce 5.4. Tabulka obsahuje tři porovnání SNI, před samotným porovnáním se provádí předzpracování pro získání podřetězce, který se následně vyhledává ve všech SNI v datasetu. Výsledná shoda je jedna, pokud se daný podřetězec nachází v některém otisku, případně nula pokud ne.

Extrahovaný SNI z neznámé komunikace	**Porovnávaný podřetězec**	**SNI v datasetu**	**Výsledná shoda**
segments.lidlplus.com	**lidlplus**	[accounts.lidl.com, segments.**lidlplus**.com, appgateway.**lidlplus**.com, ...]	1
corpapp.thumbr.io	thumbr	data.pocasi-data.cz	0
ms.cbdb.cz	cbdb	ms.cbdb.cz	0

Tabulka 5.4: Příklad porovnání pro parametr SNI.

5.2.3 Parametr Cipher Suite

Třetím atributem vhodným k porovnání jsou *šifrovací sady* z *Client Hello* paketu. Šifrovací sady mohou obsahovat hodnoty *Grease Value*. Pokud je taková hodnota nalezna, tak je nahrazena konstantou. Počet hodnot a samotné hodnoty šifrovacích sad záleží na vlastnostech aplikace. Výsledné hodnoty jsou spojeny do jednoho řetězce, který se následně porovnává se všemi řetězci stejného atributu v datasetu, kde se hledá identický řetězec. Příklad porovnání šifrovacích sad se nachází v tabulce 5.5.

V tabulce lze vidět, že šifrovací sady se extrahují z paketu jako celočíselné hodnoty, které se porovnávají všechny najednou. Při experimentálním porovnávání po hodnotách bylo zjištěno velké množství falešných detekcí. Nevýhoda je v tom, že šifrovací sady nemusí být unikátní. Například u posledního porovnává parametr šifrovací sady známá aplikace *Bazoš* se známou aplikaci *Kaufland*. Výsledné porovnání je jedna, ale jedná se o falešně pozitivní porovnání.

Extrahované šifrovací sady z neznámé komunikace	**Šifrovací sady v datasetu**	**Výsledná shoda**
[49195, 49196, 52393, 49199, 49200, 52392, 49161, 49162, 49171, 49172, 156, 157, 47, 53]	[[49195, 49196, 52393, 49199, 49200, 52392, 49161, 49162, 49171, 49172, 156, 157, 47, 53], [49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53], ...],	1
[49195, 49196, 52393, 49199, 49200, 52392, 49161, 49162, 49171, 49172, 156, 157, 47, 53]	[49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53]	0
[49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53]	[49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53]	1

Tabulka 5.5: Příklad porovnání pro parametr šifrovací sady.

5.2.4 Parametr List of Extension

Pro atributy *List of Extension* se provádí stejný postup pro extrakci hodnot i pro výsledné porovnání jako v případě parametr *Cipher Suite*. Příklad porovnání atributu se nachází v tabulce 5.6.

Extrahovaný parametr List of Extension z neznámé komunikace	Typy rozšíření v datasetu	Výsledná shoda
[65281, 0, 23, 35, 13, 5, 16, 11, 10]	[[65281, 0, 23, 35, 13, 5, 16, 11, 10], [0, 23, 65281, 10, 11, 35, 16, 5, 13]]	1
[65281, 0, 23, 35, 13, 5, 16, 11, 10]	[65281, 0, 23, 35, 13, 5, 16, 11, 10]	1
[0, 23, 65281, 10, 11, 35, 16, 5, 13]	[65281, 0, 23, 35, 13, 5, 16, 11, 10]	0

Tabulka 5.6: Příklad porovnání pro parametr List of Extension.

5.2.5 Parametry Common Name a DNS

Dalšími atributy vhodnými pro identifikace aplikací jsou *CN* a *DNS*. Mezi těmito atributy byla nalezena kladná korelace. Z tohoto důvodu je možné využít pouze jeden z atributů. Nejprve se extrahuje hodnota z pole *Server Certificate* a následně se provede získání podřetězce stejným způsobem jako v případě doménového jména. Po extrakci podřetězců se zkontroluje, jestli jsou podřetězce stejné nebo rozlišené, a nakonec se vyhledává nový podřetězec *CN* v každém z atributů *CN* v datasetu. Stejný princip je pak u podřetězce a atributu *DNS*. Pokud jsou extrahované podřetězce stejné, tak se porovnává pouze jeden podřetězec.

Extrahované DNS z neznámé komunikace	**Porovnávaný podřetězec**	**DNS v datasetu**	**Výsledná shoda**
app.kaufland.net	**kaufland**	[app.**kaufland**.net, sync.**kaufland**.de, *.sc.omtrdc.net, ...]	1
*.benzina-platby.cz	**benzina-platby**	*.**benzina-platby**.cz	1
*.places.adobe.com	adobe	*.benzina-platby.cz	0

Tabulka 5.8: Příklad porovnání pro parametr DNS.

Extrahované Common Name z neznámé komunikace	**Porovnávaný podřetězec**	**Common Name v datasetu**	**Výsledná shoda**
accounts.lidl.com	**lidl**	[accounts.**lidl**.com, segments.**lidl**plus.com, appgateway.lidlplus.com, ...]	1
corpapp.thumbr.io	thumbr	ms.cbdb.cz	0
ms.cbdb.cz	**cbdb**	ms.**cbdb**.cz	1

Tabulka 5.7: Příklad porovnání pro atribut Common Name.

5.2.6 Parametr velikost prvních deseti paketů

Následující hodnoty pro porovnání jsou odvozeny z výše vyjmenovaných metrik. První metrika je velikost prvních deseti TLS paketů. U těchto hodnot dochází k indexovému porovnání všech hodnot. Indexové porovnání je mnohem pomalejší než exaktní porovnání, protože při exaktním porovnání je možné porovnávat pouze hodnoty heší, a tím porovnání zrychlit. Nevýhodou při porovnání dvou haší je, že pokud se liší velikost alespoň jednoho z prvních deseti paketů, tak porovnání nebude úspěšné, protože vypočítané heši budou rozdílné. Indexové porovnání umožňuje porovnávat hodnoty po jednotlivých indexech, a tím dosáhnout výsledku 0–1. Nejprve se provede porovnání prvního prvku z pole prvních deseti paketů u neznámé aplikace s prvním prvkem prvního pole pro otisk aplikace z datasetu. Následně se provede porovnání druhých prvků a tak dále. Pokud mezi poli nebyla nalezna stoprocentní shoda, je výsledek uložen do proměnné a začne se porovnávat od začátku s další polem či prvním polem další aplikace.

Porovnání tří hodnot lze vidět v tabulce 5.9. Jak lze vidět při prvním porovnání, aplikace generují velké množství velmi podobných datových toků. Z tohoto důvodu je potřeba udržet velké množství dat pro přesnou detekci. První porovnaná aplikace byla známá, a proto je výsledek porovnání jedna. Sloupec obsahující velikosti prvních deseti paketů aplikace z datasetu nabývá pro první řádek několik podobných hodnot, roto se s velkou pravděpodobností jedná o datový tok stejné aplikace.

Velikost extrahovaných prvních deseti paketů z neznámé komunikace	**Velikosti prvních deseti paketů v datasetu**	**Výsledná shoda**
[117, 117, 159, 159, 241, 241, 1501, 1501, 1514, 1514]	[[117, 117, 159, 159, 241, 241, 241, 413, 1514, 1514], [117, 117, 159, 159, 241, 241, 413, 413, 1514, 1514], [117, 117, 159, 159, 241, 243, 413, 425, 1514, 1514], ...]	1
[252, 252, 252, 252, 1282, 1282, 1282, 1506, 1506, 1506]	[[117, 192, 213, 252, 284, 1268, 1514], [117, 192, 213, 213, 252, 284, 284, 284, 1268, 1514]]	0.1
[159, 246, 246, 246, 324, 1003, 1003, 1294, 1294, 1294]	[[147, 210, 238, 312, 571, 1321, 1514, 1514, 1514], [147, 238, 312, 571, 1320, 1514, 1514, 1514, 1514], [147, 147, 238, 238, 312, 312, 1439, 1440, 1514, 1514]]	0

Tabulka 5.9: Příklad porovnání pro parametr velikosti prvních deseti paketů.

5.2.7 Parametr maximální a minimální velikosti paketů

Z důvodu požadavku vysoké přesnosti na tuto metriku byly přidány ještě dvě sumy všech velikostí deseti paketů pro jeden běh aplikace. První suma je pro největší součet velikostí a druhá je pro nejnižší součet. Při porovnání se následně vypočítá suma prvních deseti TLS paketů pro neznámou aplikaci a následně se porovná, jestli náleží mezi minimálním a maximálním součtem některé aplikace z datasetu. Porovnání na minimální a maximální součet se nachází v tabulce 5.10.

Výsledek prvního a posledního je roven jedné, protože se jedná o známé aplikace, které mají otisk v datasetu. V případě druhého porovnání se porovnává neznámá aplikace Adobe se známou aplikací Bazoš. Tento atribut můžu obsahovat i velké množství falešně pozitivních porovnání, protože u některých aplikací může být rozsah velmi široký. Například v prvním řádku tabulky se vyskytuje rozsah známé aplikace Lidl, která má rozsah intervalu <2 851, 10 038>. Do tohoto rozsahu ale spadají porovnávané aplikace na druhém i třetím řádku. Z důvodu vysokého počtu falešně pozitivních porovnání byl tento atribut z porovnání všech atributů pro identifikaci aplikace vynechán.

Součet velikostí prvních deseti paketů z neznámé komunikace	**Minimální součet prvních deseti paketů v datasetu**	**Maximální součet prvních deseti paketů v datasetu**	**Výsledná shoda**
3 811	2 851	10 038	1
4 224	5 770	8 697	0
7 064	5 770	8 697	1

Tabulka 5.10: Příklad porovnání pro parametr minimálního a maximálního součtu velikosti prvních deseti paketů.

Předposlední hodnoty pro porovnání jsou maximální a minimální součet velikosti paketů odeslaných ze strany klienta na server a přijatých paketů ze serveru. Ze zašifrované komunikace se extrahují pouze pakety z ustanovení TLS spojení a následně se rozdělí do dvou skupin. První skupina obsahuje jen TLS pakety odeslané z klienta na server (*Client Hello* a *Change Cipher set.*) a druhá skupiny obsahuje pakety přijaté klientem (*Server*

Hello, *Server Certificate* a *New Session*). Příklad porovnání pro velikost paketů odeslaných z klienta na server se nachází v tabulce 5.11.

Tabulka obsahuje celkem tři porovnání známých i neznámých aplikací. Na prvním řádku se porovnává známá aplikace Kaufland s otiskem stejném aplikace v datasetu. Na druhém řádku se vykonává porovnání známé aplikace Booxy s jejím otiskem v databázi. Aplikace Booxy zasílala vždy stejné velkosti paketů od klienta na server při ustanovování TLS spojení. Z tohoto důvodu je v datasetu stejná minimální i maximální velikost. Poslední řádek obsahuje dříve neviděnou aplikaci Facebook, která odesílá pakety s větší velikostí než aplikace Bazoš, proto je výsledek roven nule. Extrahované hodnoty pro aplikaci Kaufland jsou rovny hodnotě 449 a pro aplikaci Booxy jsou rovny 400. Obě aplikace nabývají hodnoty z rozsahu aplikace Bazoš (poslední řádek v tabulace), a tím způsobují falešně pozitivní porovnání. Experimenty s tímto atributem ukázaly, že existují překryvy jednotlivých hodnot, a proto je generováno velké množství falešně pozitivních porovnání. Z tohoto důvodu byl parametr z celkového porovnání vynechán. Experimenty se nachází v kapitoly 6.

Extrahovaná velikost odeslaných paketů od klienta z neznámé komunikace	**Minimální velikost odeslaných paketů v datasetu**	**Maximální velikost odeslaných paketů v datasetu**	**Výsledná shoda**
449	410	503	1
400	400	400	1
775	410	503	0

Tabulka 5.11: Příklad porovnání pro parametr minimálního a maximálního součtu velikosti odeslaných paketů.

5.2.8 Parametr velikost ustanovení TLS spojení

Porovnání velikosti paketů zaslané v opačném směru lze vidět v tabulce 5.12. V případě prvního i druhého řádku dochází k testování hodnot od známých aplikací. První řádek obsahuje známou aplikaci Lidl a druhý řádek aplikaci Booxy. Výsledek porovnávání je u obou aplikací roven jedné. Kdyby došlo k porovnání aplikace Booxy s aplikací Lidl nebo naopak, tak by se výsledek nezměnil. Poslední řádek obsahuje dříve neviděnou aplikaci Facebook, testuje součet velikostí vůči aplikaci Bazoš. Z důvodu vyššího součtu velikostí paketů je výsledek roven nule.

Velikost odeslaných paketů ze serveru na klienta Neznámá aplikace	**Minimální velikost přijatých paketů v datasetu**	**Maximální velikost odeslaných paketů v datasetu**	**Výsledná shoda**
2386	1471	3368	1
2547	1834	3342	1
907	2882	3368	0

Tabulka 5.12: Příklad porovnání pro parametr minimálního a maximálního součtu velikosti přijatých paketů.

Poslední porovnávané hodnoty jsou velikosti paketů pro ustanovení TLS spojení. Každý extrahovaný typ paketu má určitou velikost, například *Client Hello* paket bude mít menší

velikost než *Server Certificate* paket. Z tohoto důvodu se porovnávají vždy stejné typy paketů. Samotné porovnání probíhá následně tak, že se extrahuje velikost například *Client Hello* paketu pro neznámou aplikaci a následně se stejná velikost daného typu paketu hledá v datasetu. Výsledek porovnání může být 0–1 vynásobený s přiřazenou váhou.

Příklad porovnání jednotlivých typů paketů z ustanovení TLS spojení se nachází v tabulce 5.13. První řádek tabulce porovnává již viděnou aplikaci se záznamem v datasetu, z tohoto důvodu je výsledek porovnání roven jedné. Druhý řádek porovnává ještě neviděnou aplikaci Facebook s aplikací Bazoš. Tyto dvě aplikace mají rozdílné velikosti paketů, jen paket *New Session* má stejnou velikost. Z tohoto důvodu je výsledek porovnání roven 0.2. Na posledním řádku dochází k porovnání známé aplikace Bazoš s otiskem aplikace Bazoš v datasetu. Jedná se sice o stejnou aplikaci, ale velikost *Server Hello* paketů je rozlišená. Z tohoto důvodu je výsledek porovnání roven 0.8 nikoliv jedné.

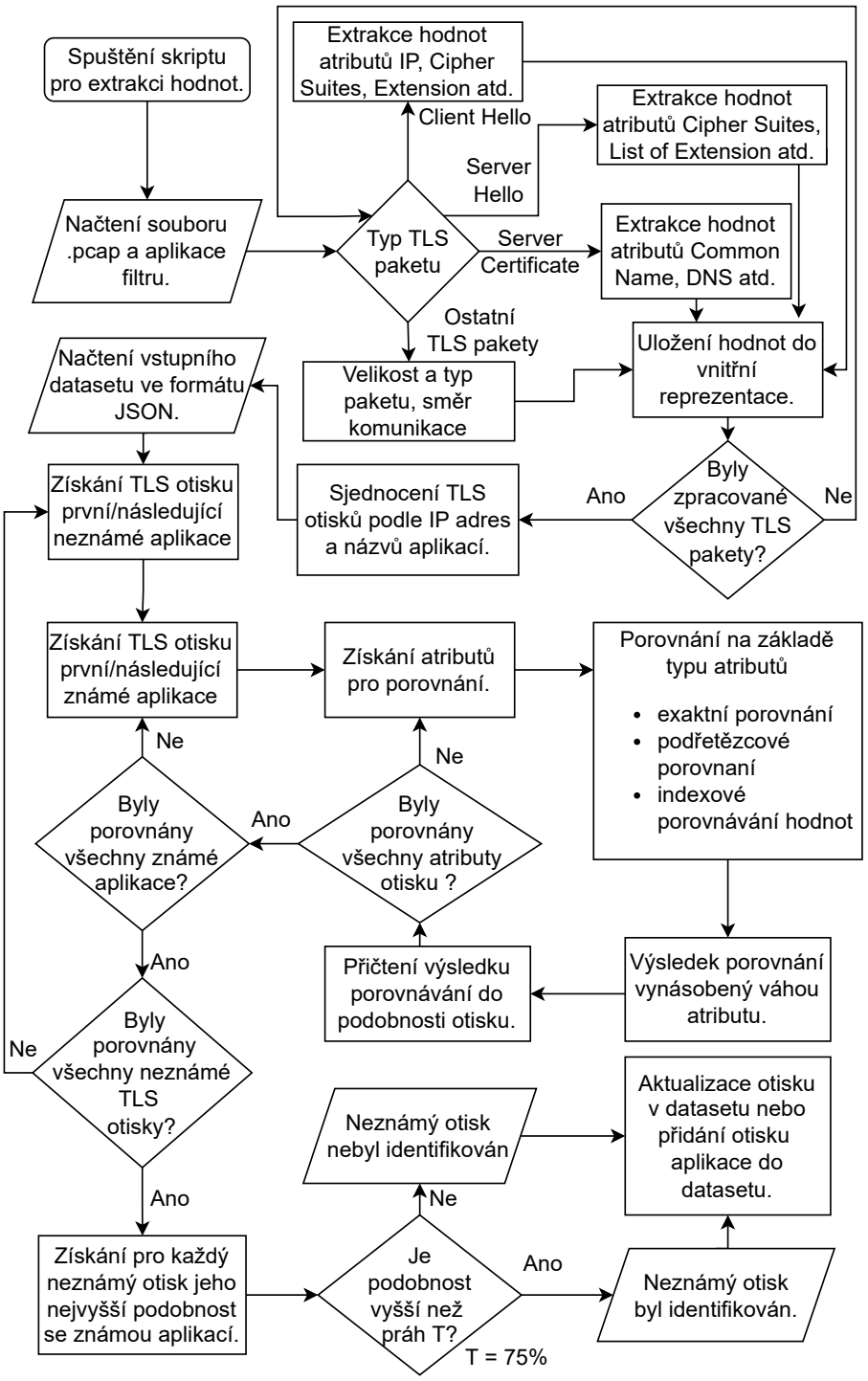
Velikost paketů pro ustanovení TLS spojení	**Velikost paketů pro ustanovení TLS spojení v datasetu**	**Výsledná shoda**
CH: 249, SH: 1514, NS: 324, SC: 548, CCS: 192	CH: [249, 252, 254, 298, 583], SH: [1514,1354, 1506], NS: [348, 340, 356, 324], SC: [548, 1513, 1514, 1266, 580, 1106, 1506, 1282], CCS: [216, 117, 192, 159]	1
CH: 583, SH: 222, NS: 340, SC: 345, CCS: 192	CH: [243], SH: [1514], NS: [340], SC: [1514], CCS: [159]	0.2
CH: 243, SH: 1506, NS: 340, SC: 1514, CCS: 159	CH: [243], SH: [1514], NS: [340], SC: [1514], CCS: [159]	0.8

Tabulka 5.13: Příklad porovnání pro velikost ustanovení TLS paketů. CH - Client Hello, SH - Server Hello, NS - New Session, SC - Server Certificate, CCS - Change Cipher set.

5.3 Porovnání TLS otisků a identifikace aplikací

Pro porovnávání TLS otisků byl vytvořen druhý skript a stejně jako první skript je naprogramován v jazyce Python3 s využitím knihovny PyShark. Skript pro správnou funkčnost potřebuje na vstupu testovací soubor .pcap a dataset ve formátu JSON. Celý popis algoritmu pro extrakci atributů se nachází v podkapitole 5.1.2. Z důvodu kompatibility s algoritmem JA3 a JA3s jsou ještě vypočteny nové MD5 heši hodnoty. Vývojový diagram příslušného algoritmu se nachází zde 5.4.

Po extrahování a dopočtení všech hodnot jsou otisky postupně porovnávané vůči všem otiskům TLS ve vytvořeném datasetu. Postup porovnání jednotlivých atributů je popsán výše. Každý atribut neznámého otisku je porovnán se stejným atributem u všech otisků



Obrázek 5.4: Vývojový diagram pro porovnání neznámých TLS otisku s TLS otisky v datasetu.

uložených v datasetu. Výsledek porovnání závisí na podobnosti či existenci hodnoty v daném atributu, výsledek porovnání může nabývat hodnot 0–1. Tento výsledek se dále násobí váhou daného atributu. Vynásobená hodnota se uložení do paměti a začne se porovnávat hodnota s dalším otiskem v datasetu, případně se začne porovnávat další atribut, jestliže porovnávaný otisk v datasetu byl poslední. Podobnost mezi dvěma aplikacemi se vypočítá jako vážený součet shody jednotlivých atributů s_i vynásobených přidělenou váhou w_i. Celkový počet atributů je označen písmenem n, index jednotlivých atributů je označen písmenem i.

$$Podobnost = \sum_{i=0}^{n} s_i \cdot w_i, Podobnost \in < 0; 100 > \tag{5.1}$$

Například u parametr IP adresa může shoda s_i nabývat hodnot 0–1 a přidělená váha je 15. Z toho plyne, že pokud se IP adrese nachází v konkrétním otisku, tak mezivýsledek bude roven 15. Celkový přehled všech parametrů se nachází v tabulce 5.14.

Po dokončení porovnání všech atributů TLS otisku se pro každý otisk aplikace vybere z testovacího souboru známá aplikace s největší mírou podobnosti. Pro úspěšné porovnání musí být podobnost mezi otisky nad prahem T. Za předpokladu podobnosti nad práh T, kdy podobnost nebude 100 %, tak lze tento uložený TLS otisk v datasetu aktualizovat a rozšířit otisk o další hodnoty. Pokud porovnání bylo úspěšné, tak je aplikace identifikována jako známá aplikace z datasetu s nejvyšší podobností.

V případě, kdy bude podobnost pod prahem T, tak otisk aplikace z testovacího souboru bude vyhodnocen jako nedetekovaný. To se může stát ve dvou případech. V prvním případě se porovnávala dříve neviděná aplikace s TLS otisky v datasetu. V tomto případě se jedná o správné vyhodnocení, protože dříve neviděná aplikace by neměla být detekována jako známá aplikace. Skript podporuje funkci přidání daného TLS otisku pro naučení se nové aplikace. Při následném spuštění testovacího souboru je tento otisk následně detekován a identifikován.

Ve druhém případě došlo k porovnání známé aplikace s TLS otisky v datasetu, ale podobnost mezi nimi byla nižší než práh T. Pokud byla aplikace detekována správně, ale podobnost byla nižší, lze otisk příslušné aplikace v datasetu rozšířit o nové hodnoty, a tím zpřesnit detekci a identifikaci dané aplikace.

Aktualizace otisku aplikace v datasetu je nutná, protože s každou další aktualizací dané aplikace TLS otisk stárne a snižuje se tak jeho podobnost s aktuálními otisky extrahovanými z nově zaznamenané komunikace.

Parametr	Typ porovnání	Množina hodnot pro shodu parametrů
IP adresa	přesné porovnání	{0, 1}
SNI	porovnání na podřetězce	{0, 1}
Cipher suite	přesné porovnání	{0, 1}
List of Extensions	přesné porovnání	{0, 1}
DNS	porovnání na podřetězce	{0, 1}
CN	porovnání na podřetězce	{0, 1}
Velikost prvních deseti paketů	indexové porovnávání	{0; 0,1; ... 0,9; 1}
Velikost ustanovení TLS spojení	Přesné porovnání pro jednotlivé typy TLS paketů	{0; 0,2; 0,4; 0,6; 0,8; 1}

Tabulka 5.14: Parametry a jejich způsob porovnání včetně množiny hodnot pro jejich shodu.

5.3.1 Porovnání známé aplikace

Tabulka 5.15 obsahuje porovnání otisku s *Common Name* app.kaufland.net z testovacího souboru s otiskem aplikace Kaufland z datasetu. Výsledkem porovnání je podobnost rovna 96.4 %. Při nastaveném prahu T = 75 % byla tato neznámá aplikace identifikována jako aplikace Kaufland. Jediný parametr, u kterého nedošlo ke stoprocentní shodě, je velikost prvních deseti paketů. U tohoto parametru dochází k indexovému porovnání jednotlivých velikostí paketů, pro třetí a čtvrtý index bylo porovnání neúspěšné, a proto je shoda tohoto atributu pouze 0.8. Shoda byla následně vynásobena váhou 15 a následně přičtena k celkovému výsledku. Tabulka obsahuje jen dva řádky pro algoritmus JA3 a JA3s. Na základě těchto dvou algoritmů byla aplikace Kaufland identifikována jako aplikace Kaufland a Lidl, protože tyto aplikace mají stejné JA3 i JA3s otisky.

Parametr	Neznámá aplikace	Známá aplikace	Shoda	Váha
Název aplikace	app.kaufland.net	Kaufland	X	X
IP adresa	['52.236.157.206']	['52.236.157.206', ..., '77.87.187.36']	1	15
Cipher suites	[[49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53]]	[[49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53], ...]	1	8
List of Extension	[[0, 23, 65281, 10, 11, 5, 13]]	[[0, 23, 65281, 10, 11, 5, 13], ...]	1	9
SNI	['app.kaufland.net']	['app.kaufland.net', ...]	1	15
Common Name	['app.kaufland.net']	['app.kaufland.net', ...]	1	15
DNS	['app.kaufland.net']	['app.kaufland.net', ...]	1	15
Velikost prvních deseti paketů	[[117, 224, 225, 257, 1506, 1506, 1506, 1506, 1506, 1506]]	[[117, 224, 244, 244, 1506, 1506, 1506, 1506, 1506, 1506], ...]	0,8	18
Velikost ustanovení TLS spojení	{1: [257], 2: [1506], 11: [1506], 16: [224], 20: [117]}	{1: [257, 225, 269, ...], 2: [1514, 1506, ...], 11: [1514, 1506, ...], 16: [224, 159, 192], 20: [117], 4: [324, 356]}	1	5
JA3	['6c0f0a346dcd84...']	['6c0f0a346dcd84...', ...]	1	X
JA3s	['f6dfdd25d1522e...']	['f6dfdd25d1522e...', ...]	1	X

Podobnost	96.4 %

Tabulka 5.15: Proces porovnání známé aplikace Kaufland s otiskem aplikace Kaufland z datasetu.

5.3.2 Porovnání neznámé aplikace

Tabulka 5.16 obsahuje porovnání otisku s atributem *Common Name* *.facebook.com z testovacího souboru s otiskem aplikace Bazoš z datasetu. Výsledná podobnost je rovna 10 %. Při nastaveném prahu T = 75 % nebyla aplikace správně detekována. U parametru *List of Extension* došlo k falešně pozitivní shodě. U ostatních parametrů nebyla nalezena žádná významná shoda. Z tohoto důvodu je výsledná podobnost pod prahem T. Pro možnost srovnání s algoritmem JA3 a JA3s byla tabulka rozšířena o dva řádky. Na základě algoritmu JA3 byla neznámá aplikace Facebook identifikována jako aplikace Kaufland, Lidl, Waze. Pro algoritmus JA3s neobsahoval dataset shodný MD5 heš, proto tento algoritmus, případně spojení algoritmů JA3 a JA3s, správně nedetekoval aplikaci.

Parametr	Neznámá aplikace	Známá aplikace	Shoda	Váha
Název aplikace	.facebook.com	Bazoš	0	X
IP adresa	['69.171.250.15']	['88.86.119.246']	0	15
Cipher suites	[[49195, 49196, 52393, 49199, 49200, 52392, 49161, ... 53]]	[[49195, 49196, 52393, 49199, 49200, 52392, 49171, 49172, 156, 157, 47, 53]]	0	8
List of Extension	[[65281, 0, 23, 35, 13, 5, 16, 11, 10], ...]	[[65281, 0, 23, 35, 13, 5, 16, 11, 10]]	1	9
SNI	['graph.facebook.com']	['www.bazos.cz']	0	15
Common Name	['*.facebook.com']	['*.bazos.cz']	0	15
DNS	['*.facebook.com']	['*.bazos.cz']	0	15
Velikost prvních deseti paketů	[[192, 192, 250, 250, 340, 340, 465, 466, 1446, 1446]]	[[159, 243, 243, 243, 340, 828,1514, 1514, 1514, 1514], ...]	0	18
Velikost ustanovení TLS spojení	{1: [250], 2: [1446], 4: [340], 11: [466], 16: [192]}	{1: [243], 2: [1514], 4: [340], 11: [1514, 1028], 16: [159]}	0,2	5
JA3	['d8c87b9bfde3889..., '6ec2896feff57469..., 'ee26b1f1aec16d60...']	['6f5e62edfa5933...]	0	X
JA3s	['6e15a5bf660856f...', '79d63d20e65217c...']	['5deff56d75697c..']	0	X

Podobnost	9 %

Tabulka 5.16: Proces porovnání pro dříve neviděnou aplikaci Facebook a známou aplikaci Bazoš z datasetu.

5.4 Aktualizace datasetu

Identifikace na základě TLS otisků je ideální pouze v neměnném prostředí, ale toho ve skutečnosti nelze dosáhnout. Telefonní aplikace se neustále aktualizují, případě si uživatelé mohou nainstalovat nové, které by nemusely být detekovatelné. Z tohoto důvodu je potřeba základní dataset rozšířit o nové TLS otisky, případně aktualizovat stávající. Tato sekce se bude zaměřovat právě na aktualizaci a přidaní nových otisků do již existujícího datasetu.

5.4.1 Aktualizace stávajících otisků aplikací

Při vytváření datasetu byly nejprve získány záznamy aplikací, ze kterých se následně extrahovaly atributy. Z extrahovaných atributů vznikly TLS otisky. Celý postup od získání záznamů až po vytvoření otisků byl popsán již výše.

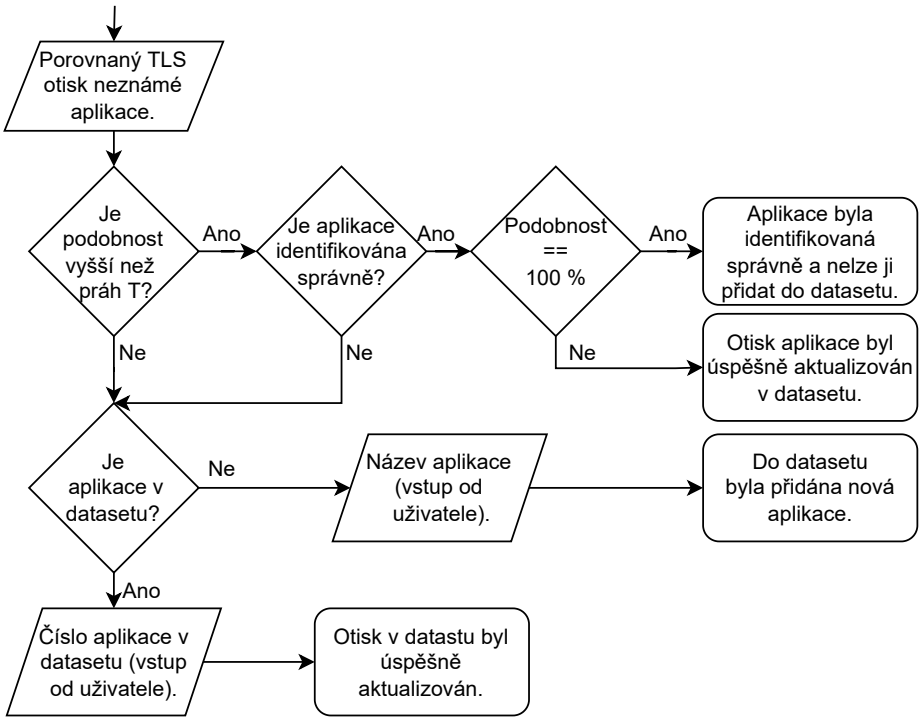
Při experimentování s datasetem byly zjištěny dvě zásadní nedokonalosti, jež mohou negativně ovlivňovat přesnost detekcí.

První nedokonalost je stárnutí samotného datasetu. Jak již bylo zmíněno, aplikace se neustále aktualizují, pokud aktualizace způsobí změnu i v síťové komunikace, tak tato změna může přispět ke zhoršující přesnosti detekce. Z tohoto důvodu je nutné udržovat jednotlivé otisky aplikací neustále aktuální.

Druhá nedokonalost je nemožnost zachytit všechna možná chování aplikace. Při vytváření záznamu byla daná aplikace spuštěna a bylo s ní zacházeno dle typu aplikace. Při zachytávání reálné zašifrované komunikace a následné identifikace aplikace bylo zjištěno, že některé aplikace nebyly zapnuty korektně nebo se nepodařilo ustanovit TLS spojení.

Při nedokončeném ustanovení TLS komunikace se detekce provádí pouze z její části, například pouze z paketu *Client Hello*. Tato skutečnost velmi zásadně ovlivňuje detekci a identifikaci daného spojení.

Aktualizace otisku se může provádět dvěma způsoby, a to manuálně nebo automaticky. Nejprve zde bude rozebrán manuální způsob pro aktualizaci otisků. Jako první krok je potřeba spustit program pro detekci aplikací ze zašifrovaného síťového provozu, který na základě datasetu a prahu T rozdělí aplikace do dvou skupin.

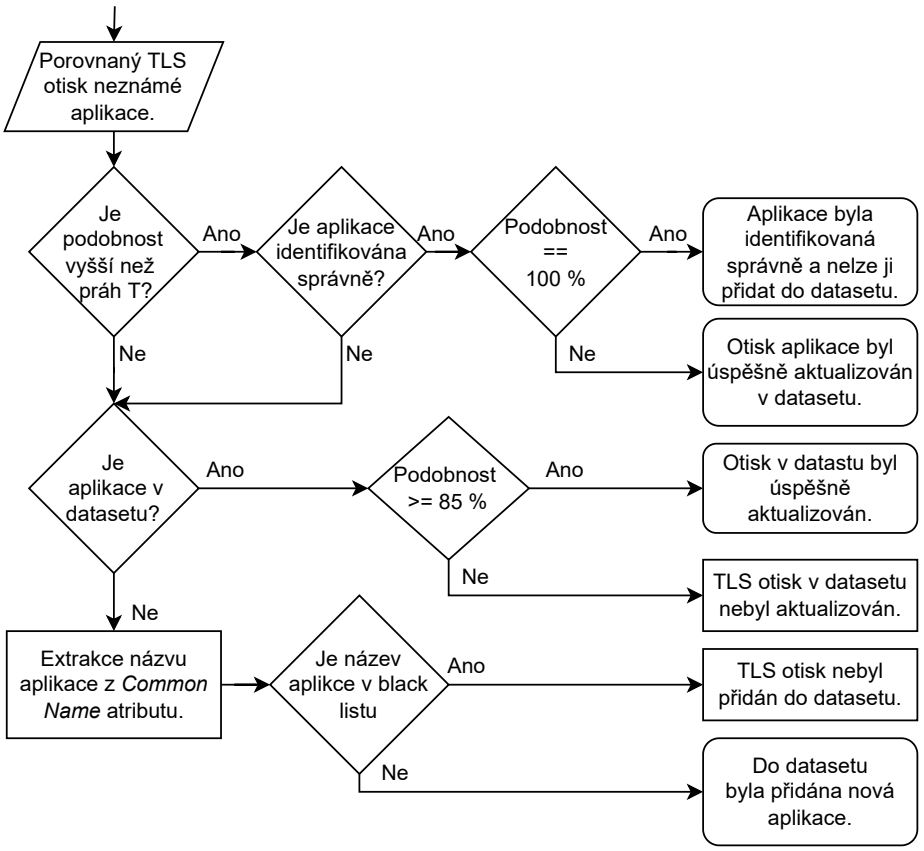


Obrázek 5.5: Vývojový diagram pro manuální aktualizaci datasetu.

První skupina jsou detekované aplikace, u kterých můžu být přesnost detekce od prahu T po 100 %. Ostatní aplikace se budou nacházet ve skupině pro nedetekované aplikace. Podle rozdělení aplikací do skupin se liší způsob jejich aktualizace. Pokud byla daná aplikace rozeznána na více jak práh T, ale na méně než 100 %, tak se jedná o aktualizaci, která pomáhá zpřesnit detekci dané aplikace. Během této aktualizace dochází jen ke kontrole položek, kde nebyla nalezená 100% shoda. Následně je hodnota atributů přidána do již vytvořeného otisku.

Pro aktualizaci TLS otisku z druhé skupiny je nejprve potřeba ručně zadat jméno aplikace, k níž daný otisk patří. Následně se provede kontrola všech atributů a metrik, kde se zkontroluje procento shody a duplicita daných hodnot. Následně se do otisku v datasetu vloží rozdílné hodnoty. Při dalším spuštění aplikace se provede detekce s již novým datasetem, který zajistí přesnější detekci známých či neznámých aplikací.

Pro automatické aktualizování správně identifikované aplikace bylo zapotřebí přidat nový práh. Tento práh zaručuje přidání hodnot do konkrétního TLS otisku v datasetu. Z tohoto důvodu je práh nastaven na 85 % podobnosti.



Obrázek 5.6: Vývojový diagram pro automatickou aktualizaci datasetu.

5.4.2 Přidání nové aplikace do datasetu

Pro přidání nové aplikace bylo potřeba naimplementovat postup vytváření nového otisku ze zašifrované neznámé komunikace a také metody dovolující změnit název aplikace v existujícím datasetu. Nejprve se ze zašifrované komunikace extrahují atributy a z těch se vytvoří otisky, které se následně porovnávají a rozdělí se do dvou skupin.

Pokud je aplikace ve druhé skupině, tedy ve skupině aplikací, které se nepovedlo identifikovat, tak se může otisk přidat jako nová aplikace. Nejprve je tedy potřeba vybrat otisk a následně zadat jméno aplikace. Po zadání proběhne algoritmus, kde se do stávajícího datasetu vloží nová aplikace s nově vytvořeným otiskem.

Při dalším spuštění programu nad stejným záznamem síťové komunikace je aplikace identifikována. V případě, kdy se spustí jiný záznam komunikace, který obsahuje nově na-

učenou aplikaci, může dojít k její identifikace, ale taky nemusí. To je z toho důvodu, že nová aplikace se učila z jednoho nebo malého počtu otisků, proto bude potřeba tento otisk rozšířit pomocí jeho aktualizace.

Pro automatické přidávání nových aplikací bylo zapotřebí vytvořit soubor. Tento soubor bude obsahovat *Common Name* nebo jejich podřetězec, se kterým se bude porovnávat nový název aplikace. Tímto způsobem lze do značné míry omezit přidávání reklamních a jiných méně podstatných TLS otisků.

Kapitola 6

Experimenty

V následující kapitole budou rozebrány jednotlivé experimenty, které výrazně pomohly k určení porovnávacího prahu T, vyloučené/spojení některých atributů a zpřesnění vah pro jednotlivé atributy. V první části kapitoly budou kompletní experimenty a výpočty pro určení prahu T. Další část bude obsahovat experimenty pro možnost sjednocení nebo odstranění některých atributů. Poslední část se zaměřuje na experimenty pro stanovení a upřesnění vah jednotlivých atributů.

6.1 Ustanovení porovnávacího prahu

Porovnávací práh (dále už jen „práh T“) slouží k detekci neznámých aplikací ze zaznamenané šifrované komunikace. Pro každý otisk je nejprve vypočítána podobnost na základě vzorce 5.1. Výsledná podobnost je porovnávána s prahem T, pokud je podobnost vyšší než práh T, tak aplikace byla správně identifikována. V opačném případě aplikace nebyla rozpoznána.

Pro ustanovení prahu T bylo potřeba provést několik experimentů. Nejprve bylo potřeba definovat metriku dovolující porovnat různé prahy T. Pro porovnání jednotlivých prahů byla použita chybová matice, která udává v kolika případech se aplikace detekovala správně, případně v kolika případech byla detekce chybná. Práh T byl následně zvolen na hodnotu 75 %, kde docházelo k nejmenšímu počtu falešně pozitivních detekcí a k nejmenšímu počtu nedotkávaných aplikací.

		Predikovaná hodnota	
	Klasifikace	Pozitivní	Negativní
Skutečná hodnota	Pozitivní	True positive (správná detekce)	False negative (nedetekovaná aplikace)
	Negativní	False positive (nesprávná detekce)	True negative (správně nedetekovaná aplikace)

Tabulka 6.1: Chybová matice (matice záměn) neboli *Confusion matrix* pro práh T.

Hodnota prahu T byla nastavena nejprve na 60 % a následně se zvyšovala po pěti procentech až k hodnotě 90 %. Poslední hodnota prahu je nastavena na 100 %. Hodnota prahu T rovna 60 % znamená, že pokud získaný otisk z testovacího souboru bude na více

Práh	**60 %**	**65 %**	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**100 %**
True positive (Správná detekce)	10	10	**10**	**10**	9	9	5	2
False positive (Nesprávná detekce)	0	0	**0**	**0**	0	0	0	0
False negative (Nedetekovaná aplikace)	0	0	**0**	**0**	1	1	5	8
True negative (Správně nedetekovaná aplikace)	18	18	**18**	**18**	18	18	18	18

Tabulka 6.2: Výsledek prvního testovací souboru pro nastavení hodnoty prahu T.

jak 60 % podobný otisku nějaké známé aplikace, tak bude aplikace detekována. V opačném případě bude aplikace nedetekována. Pro každý práh byla vypočítána chybová matice.

První testovací soubor obsahoval několik aplikací, ze kterých se podařilo vytvořit celkem 28 otisků aplikací. Pro některé aplikace se podařilo získat i více než jeden otisk. Tato skutečnost je dána tím, že aplikace komunikuje s několika servery a pro každou komunikaci je následně vytvářen otisk. Testovací soubor tedy obsahoval známé aplikace *Bazoš*, *BlaBlaCar*, *Kaufland*, *Lidl* a aplikaci *Mapy*. Dále soubor obsahoval několik neznámých aplikací, jako je například aplikace *Facebook*, a také obsahoval několik reklamních serverů.

Pro jednotlivé prahy T byla vytvořena tabulka 6.2, v níž lze vidět správnost detekcí. Pro práh T 60 % bylo správně detekovaných deset otisků, které patřily výše vyjmenovaným aplikacím. Zbylých osmnáct otisků bylo správně detekováno jako neznámé aplikace.

První testovací soubor s prahem 75 %			
Detekované aplikace			
CN	**Podobnost**	**Aplikace**	**Klasifikace**
account.kaufland.com	77.33 %	kaufland	TP
app.kaufland.net	93.33 %	kaufland	TP
*.sc.omtrdc.net	86.67 %	kaufland	TP
sync.kaufland.de	93.33 %	kaufland	TP
*.lidl-flyer.com	78.67 %	lidl	TP
*.bazos.cz	90.67 %	bazos	TP
vectmap.mapy.cz	86.67 %	mapy	TP
*.mapy.cz	100 %	mapy	TP
blablacar.com	88.89 %	blablacar	TP
cdn.blablacar.com	78.67 %	blablacar	TP

Tabulka 6.3: Přehled všech detekovaných otisků z prvního testovaného souboru s prahem T = 75 %.

Z tabulky vyplývá, že pro hodnotu prahu 60 % až 75 % došlo k nejlepšímu počtu detekcí známých i neznámých aplikací, dokonce všechny detekce byly správně klasifikovány. Při zvýšení hodnoty prahu na 80 % došlo ke špatné detekci jednoho otisku. Pro aplikaci *Kaufland* byly vytvořeny celkem čtyři otisky a pouze jeden z nich měl podobnost s exis-

tujícím otiskem pod 80 %, přesněji měl podobnost 77.33 %. Při detailnější analýze bylo zjištěno, že parametr IP adresa byl porovnán s nulovou shodou z důvodu neznámé IP adresy. Dále parametr velikosti prvních deseti paketů byl porovnán se shodou pouze 60%. Pro větší podobnost mezi TLS otisky u aplikace *Kaufland* by bylo potřeba se daný otisk naučit, a tím zlepšit budoucí detekci této aplikace.

Celkový přehled aplikací a detekovaných/nedetekovaných aplikací s prahem T = 75 % se nachází v tabulkách 6.3 a 6.4. Kompletní přehled aplikací a jejich klasifikací ve všech testech při prahu 75 % se nachází v příloze D.

První testovací soubor s prahem 75 %		
Nedetekované aplikace		
CN	**Podobnost**	**Klasifikace**
firebaseinstallations. googleapis.com	65.33 %	TN
*.facebook.com	60.44 %	TN
assets.adobedtm.com	60.67 %	TN
*.demdex.net	64 %	TN
*.lokalise.co	50.67 %	TN
crashlytics.com	61.33 %	TN
*.google-analytics.com	56 %	TN
firebaseremoteconfig. googleapis.com	62 %	TN
thumbr.io	57.33 %	TN
*.places.adobe.com	62.67 %	TN
app-ks.seznam.cz	65.33 %	TN
*.cloudfront.net	61.33 %	TN
*.privacy-center.org	62.67 %	TN
y.ssl.fastly.net	57.33 %	TN
crashlyticsreports-pa. googleapis.com	65.33 %	TN
app.adjust.com	62 %	TN
store. steampowered.com	66.67 %	TN
a248.e.akamai.net	64 %	TN

Tabulka 6.4: Přehled všech nedetekovaných otisků z prvního testovaného souboru s prahem T = 75 %.

Na základě provedených testů pro nalezení správného prahu T byl vybrán práh T = 75 %. Při této hodnotě prahu T byly všechny známé i neznámé aplikace klasifikovány správně.

Zbylé chybové matice se nacházejí v tabulkách níže, kde, stejně jako pro první testovací soubor, byly nastaveny hodnoty prahu T od 60 % do 100 %.

Druhý testovací soubor								
Práh	**60 %**	**65 %**	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**100 %**
True positive (Správná detekce)	0	0	0	0	0	0	0	0
False positive (Nesprávná detekce)	0	0	0	0	0	0	0	0
False negative (Nedetekovaná aplikace)	0	0	0	0	0	0	0	0
True negative (Správně nedetekovaná aplikace)	34	34	34	34	34	34	34	34

Tabulka 6.5: Chybová matice z druhého testovacího souboru pro stanovení prahu T.

Druhý testovací soubor 6.5 neobsahoval žádnou dříve viděnou aplikaci. Největší shodu měla aplikace s atributem *Common Name firebaseremoteconfig.googleapis.com*, která měl shodu rovnu 35.40 %.

Třetí testovací soubor 6.6 obsahoval šest známých otisků aplikací z celkového počtu 29 otisků. Pro práh T = 85 % byl jeden známý TLS otisk špatně identifikován. Jedná se o otisk aplikace Lidl s atributem CN roven **.lidl-flyer.com*, u kterého byla vypočítána podobnost 82 %, protože parametr velikosti prvních deseti paketů by neúspěšně porovnán se stejným parametrem otisku aplikace Lidl uloženého v datasetu. Výsledná shoda těchto parametrů je rovna nule. Pro vylepšení podobnosti TLS otisků je zapotřebí rozšířit otisk aplikace z datasetu o hodnoty nového otisku, a to například pomocí učícího algoritmu pro vytváření a aktualizování TLS otisků v datasetu. Zbylé otisky známých aplikací měly podobnost větší než 90 %.

Třetí testovací soubor								
Práh	**60 %**	**65 %**	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**100 %**
True positive (Správná detekce)	6	6	6	6	4	4	3	0
False positive (Nesprávná detekce)	0	0	0	0	0	0	0	0
False negative (Nedetekovaná aplikace)	0	0	0	0	0	2	3	6
True negative (Správně nedetekovaná aplikace)	23	23	23	23	23	23	23	23

Tabulka 6.6: Chybová matice z třetího testovacího souboru pro stanovení prahu T.

Čtvrtý testovací soubor 6.7 zahrnoval celkem 41 TLS otisků aplikací a z toho devět známých otisků jedné aplikace. Pro osm otisků známé aplikace byla vypočtena podobnost rovná 100 %, pouze u jednoho otisku byla podobnost rovna 84.70 %. Při detailnější analýze bylo zjištěno, že parametr velikosti prvních deseti prvků byl shodný pouze na 40 %. Z tohoto

důvodu byla výsledná podobnost rovna pouhých 84.70 %. Při nastavení prahu T = 75 % by byla tato aplikace správně detekována.

Čtvrtý testovací soubor								
Práh	**60 %**	**65 %**	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**100 %**
True positive (Správná detekce)	9	9	9	9	9	9	8	8
False positive (Nesprávná detekce)	0	0	0	0	0	0	0	0
False negative (Nedetekovaná aplikace)	0	0	0	0	0	0	1	1
True negative (Správně nedetekovaná aplikace)	33	33	33	33	33	33	33	33

Tabulka 6.7: Chybová matice z čtvrtého testovacího souboru pro stanovení prahu T.

V pátém testovací souboru 6.8 se nacházelo celkem 33 otisků aplikací, z toho pět otisků bylo známých aplikací. Aplikace s nejmenší podobností byla aplikace Kaufland s 79.20 %. Detailnější analýza ukázala, že parametr *Cipher Suite* nebyl úspěšně porovnán se stejným parametrem aplikace Kaufland. Výsledná shoda pro tento atribut je rovna nule, ale celková podobnost dvou otisků je rovna 79.20 %. Podobnost dvou otisků se vypočítá pomocí vážený součtu shod jednotlivých parametrů. Parametr velikosti prvních deseti paketů byl porovnán se shodou 40 % a parametr velikosti paketů pro ustanovení TLS spojení byl porovnán se shodou 80 %. Pro stoprocentní podobnost otisku aplikace Kaufland je zapotřebí rozšířit její otisk v databázi o hodnoty parametrů, u nichž nebyla nalezena stoprocentní shoda. Zbylé otisky známých aplikací byly porovnány s podobností nad 90 %.

Pátý testovací soubor								
Práh	**60 %**	**65 %**	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**100 %**
True positive (Správná detekce)	5	5	5	5	4	4	4	2
False positive (Nesprávná detekce)	0	0	0	0	0	0	0	0
False negative (Nedetekovaná aplikace)	0	0	0	0	1	1	1	3
True negative (Správně nedetekovaná aplikace)	27	27	27	27	27	27	27	27

Tabulka 6.8: Chybová matice z pátého testovacího souboru pro stanovení prahu T.

Poslední soubor obsahoval celkem 28 otisků aplikací, z toho bylo deset otisků známých aplikací. Aplikace Kaufland měla podobnost rovnu 87.20 %, protože u parametru velikosti prvních deseti paketů a velikosti ustanovení TLS spojení byla nalezna pouze částečná shoda.

Šestý testovací soubor								
Práh	**60 %**	**65 %**	**70 %**	**75 %**	**80 %**	**85 %**	**90 %**	**100 %**
True positive (Správná detekce)	10	10	10	10	10	10	9	8
False positive (Nesprávná detekce)	0	0	0	0	0	0	0	0
False negative (Nedetekovaná aplikace)	0	0	0	0	0	0	1	2
True negative (Správně nedetekovaná aplikace)	18	18	18	18	18	18	18	18

Tabulka 6.9: Chybová matice z šestého testovacího souboru pro stanovení prahu T.

Výsledné chybové matice pro jednotlivé experimenty ukazují, že při nastavení prahu mezi 60 až 75 % dochází ke správnému rozdělení aplikací na detekované a nedetekované aplikace. Při nastavení prahu T na 60 % a méně může dojít v budoucnu k falešně pozitivní detekci nějaké neznámé aplikace. Při nastavení prahu na více jak 75 % dochází k nedetekování některých známých aplikací, a tím se snižuje i celková přesnost algoritmů. Kvůli tomu byl porovnávací práh T nastaven na hodnotu 75 %, kdy nedochází k falešně pozitivním detekcím, ani k nedetekování známých aplikací. Při reálném nasazení navržené aplikace může dojít k aktualizaci jednotlivých otisků, a tak může dojít ke správné detekci aplikací i u otisků s nižší hodnotou podobnosti.

6.2 Změna porovnání pro jednotlivé atributy

Váhy pro jednotlivé atributy byly odvozeny na základě informační entropie a hodnoty informačního bitu. Čím vyšší měl atribut hodnotu informačního bitu, tím vyšší váhu získal a byl zařazen do jedné ze tří kategorií. Následné experimenty ukázaly, že se u všech atributů nedá spoléhat pouze na hodnotu informačního bitu. Například pro hodnoty, které se porovnávaly na maximální a minimální velikost, byla hodnota informačního bitu mezi 3–4, a to z důvodu počítání informační entropie pouze z minimální a maximální hodnoty součtu různých běhů aplikace. Pro aplikaci Lidl byly takové hodnoty 388 bytů pro minimální součet velikostí paketů odeslaných klientem na server při ustanovení spojení TLS a 775 bytů pro maximální součet velikostí. Při intervalovém porovnávání tohoto parametru docházelo k velkému počtu falešně pozitivních detekcí. Například aplikace Kaufland měla minimální součet 410 bytů a 503 bytů pro maximální součet. Lze vidět, že interval aplikace Kaufland je podintervalem intervalu aplikace Lidl. Z tohoto důvodu nelze pouze na základě hodnoty informačního bitu odvozovat význam parametru pro identifikaci aplikací.

6.2.1 Intervalové porovnávání

Tabulka 6.10 obsahuje minimální a maximální součet velikostí paketů odeslaných od klienta na server během ustanovení spojení TLS. Tabulka obsahuje všech deset testovacích aplikací včetně aplikace Lidl, která má nejmenší i největší součet paketů odeslaných od klienta na server, a to při porovnávání součtů extrahovaných velikostí paketů odeslaných z klient-

ského zařízení s intervalem v tabulce. Bylo pozorováno velké množství falešně pozitivních shod, protože intervaly se navzájem překrývají. Například při porovnání jakékoliv hodnoty z intervalu pro aplikaci Benzina by nastala falešná shoda u aplikací Kaufland, Lidl. Mapy, Shazam a Waze.

Velikosti paketů odeslaných z klienta na server		
Název aplikace	**Minimální součet [B]**	**Maximální součet [B]**
Bazoš	402	402
Benzina	444	476
BlaBlaCar	405	407
Booxy	400	400
ČHMU	718	718
Kaufland	410	503
Lidl	**388**	**775**
Mapy	401	742
Shazam	404	742
Waze	401	742

Tabulka 6.10: Minimální a maximální velikost paketů odeslaných z klienta na server.

V tabulce 6.11 se nachází minimální a maximální součet velikostí paketů odeslaných serverem na klientské zařízení ve fázi ustanovení TLS spojení. Aplikace Shazam má nejmenší součet odeslaných paketů, naopak aplikace Mapy má nejvyšší součet odeslaných paketů ze serveru. Navíc obě aplikace mají relativně velké intervaly, proto známé nebo neznámé aplikace padnou s vysokou pravděpodobností právě do tohoto intervalu a způsobí mnoho falešně pozitivních shod. Z důvodu pozorování překryvu intervalu byly parametry minimální a maximální počet odeslaných paketů z klientského zařízení na server nebo opačně vyloučeny z klasifikace.

Velikosti paketů odeslaných ze serveru na klienta		
Název aplikace	**Minimální součet [B]**	**Maximální součet [B]**
Bazoš	2882	3368
Benzina	1598	2899
BlaBlaCar	2700	3326
Booxy	2044	3132
ČHMU	1843	3266
Kaufland	1834	3342
Lidl	1471	3368
Mapy	2596	3855
Shazam	1358	3193
Waze	2064	3326

Tabulka 6.11: Minimální a maximální velikost paketů odeslaných ze serveru na klienta.

Poslední tabulka 6.12 pro intervalové porovnání je zaměřena na minimální a maximální velikost prvních deseti paketů. V tabulce se nachází i aplikace Mapy, která má největší interval ze všech aplikací z tabulky. Ostatní aplikace jsou součástí právě tohoto intervalu a vzájemně se překrývají. Z tohoto důvodu dochází při intervalovém porovnávání k falešně pozitivním shodám.

Velikosti prvních deseti paketů		
Název aplikace	**Minimální součet [B]**	**Maximální součet [B]**
Bazoš	5770	8697
Benzina	3343	4621
BlaBlaCar	3733	6582
Booxy	4716	8543
ČHMU	4313	7301
Kaufland	3554	9907
Lidl	2851	10038
Mapy	**2997**	**10164**
Shazam	3047	7198
Waze	3727	9904

Tabulka 6.12: Minimální a maximální velikosti prvních deseti TLS paketů.

Analýza parametru minimální a maximální velikosti paketů ukázala, že se minimální a maximální intervaly aplikací překrývají, a tím způsobují falešně pozitivní výsledky detekcí. Z tohoto důvodu bylo porovnání na maximální a minimální hodnoty vyloučeno z klasifikace.

6.2.2 Indexové porovnávání

Parametr velikosti prvních deseti paketů nebyl vhodný pro intervalové porovnání na minimální a maximální součet velikostí. Tabulka 6.13 obsahuje celkem čtyři řádky, na nichž se porovnávají velikosti prvních deseti paketů s různou mírou shody. Při přesném porovnáváním by byl úspěšný pouze poslední záznam v tabulce, zde se nachází stroprocentní shoda.

Druhé možné porovnání je na existenci hodnot. Porovnaní funguje na principu získání hodnoty z pole neznámé aplikace a následné hledání v poli známé aplikace. Pokud je hodnota úspěšně nalezena ve druhém poli na libovolném indexu, tak je porovnání pro jednu hodnotu úspěšné. Za předpokladu nalezení všech hodnot z prvního pole v poli druhém dochází ke stoprocentní shodě. Při nalezení pouze pěti hodnot z deseti je shoda rovna 50 %, a to při porovnávání pouze na existenci hodnot z pole. Tímto způsobem byly úspěšně porovnané poslední dva řádky v tabulace. Neznámá aplikace na třetím řádku totiž obsahuje všechny hodnoty známé aplikace, proto by došlo k úspěšnému porovnání. Poslední řádek obsahuje identická pole, proto byl úspěšně porovnán se stoprocentní shodou.

Velikosti prvních deseti paketů		
Shoda	**Neznámá aplikace**	**Známá aplikace z datasetu**
70 %	[159, 159, 246, 246, 324, 324, 1003, 1003, 1294, 1294]	[159, 246, 246, 246, 324, 1003, 1003, 1294, 1294, 1294]
80 %	[117, 224, 244, 244, 1506, 1506, 1506, 1506, 1506, 1506]	[117, 224, 225, 257, 1506, 1506, 1506, 1506, 1506, 1506]
90 %	[159, 159, 246, 246, 324, 324, 1003, 1003, 1294, 1294]	[159, 159, 246, 246, 246, 324, 1003, 1003, 1294, 1294]
100 %	[117, 224, 225, 225, 225, 1514, 1514, 1514, 1514, 1514]	[117, 224, 225, 225, 225, 1514, 1514, 1514, 1514, 1514]

Tabulka 6.13: Velikosti prvních deseti paketů pro indexové porovnání.

Poslední možnost je použití indexového porovnání, kdy se porovnává nejen samotná hodnota, ale i její pozice. Indexové porovnání dovoluje vypočítat shodu porovnání od 0 po 100 %. Na rozdíl od porovnání na existenci hodnoty je zde důležitá nejen hodnota, ale i její pozice, která snižuje falešně pozitivní shodu atributů. Z tohoto důvodu je parametr velikosti prvních deseti paketů porovnáván indexově. Při porovnávání parametru velikosti prvních deseti paketů, může dojít k porovnání menšího množství paketů, například pokud je přerušeno navazování spojení. V takovém případě je indexově porovnáván pouze zaznamenaný počet. Příklad pole neznámé aplikace bude obsahovat pouze pět hodnot, dojde k porovnání pouze pěti hodnot. Výsledná shoda tohoto parametru je od 0–50 %.

Příklad pro porovnávání na existenci hodnot a indexové porovnání se nachází v tabulce 6.14.

Porovnávání na existenci hodnot		
První pole	**Druhé pole**	**Shoda**
[1, 2, 3, 4, 5]	[1, 6, 7, 3, 9]	2/5 = 40 %
[1, 2, 3, 4, 5]	[5, 4, 3, 2, 1]	5/5 = 100 %

Indexové porovnávání		
První pole	**Druhé pole**	**Shoda**
[1, 2, 3, 4, 5]	[5, 4, 3, 2, 1]	0 %
[1, 2, 3, 4, 5]	[1, 6, 7, 3, 9]	1/5 = 20 %
[1, 2, 3, 4, 5]	[1, 2, 3, 4, 5]	5/5 = 100 %

Tabulka 6.14: Ukázka porovnání na existenci hodnot a indexového porovnávání.

Kapitola 7

Srovnání s existujícím řešením

V této kapitole se srovnává existující metoda JA3 a JA3s s navrženou metodou pro extrakci síťové komunikace a vytváření TLS otisků. Nejprve bude rozebrán postup úpravy programu pro podporu algoritmů JA3 a JA3s. Následně bude program spuštěn a jeho výsledky analyzovány. Poslední část kapitoly se bude věnovat porovnání obou metod a jejich úspěšnosti při identifikování jednotlivých aplikací.

7.1 Podpora algoritmu JA3 a JA3s

Skript pro extrakci atributů ze síťové komunikace byl rozšířen o možnost získat potřebné atributy z paketu *Client Hello* a paketu *Server Hello*. Z paketu *Client Hello* bylo zapotřebí extrahovat atributy verze protokolu TLS, podporované šifry, podporované rozšíření protokolu, podporované typy eliptických křivek a formát bodů eliptických křivek. Extrahované hodnoty byly následně zpracovány podle JA3 algoritmu. Všechny atributy bylo potřeba vložit do jednoho řetězce s tím, že každý atribut bude oddělen čárkou. Po vytvoření řetězce je potřeba z něj vypočítat MD5 heš. Příklad extrahovaných hodnot a výpočtu algoritmu JA3 se nachází v sekci 2.3.

Z paketu *Server Hello* byly získány hodnoty pro atributy verze protokolu TLS, podporované šifrovací algoritmy a podporované rozšíření protokolu. Příklad extrahovaných hodnot a výpočtu algoritmu JA3s se nachází v sekci 2.3.

Po vypočtení JA3 a JA3s algoritmu byly pro každou aplikaci ponechány pouze unikátní MD5 heši. Duplicitní heši byly odstraněny.

7.1.1 Získané JA3 a JA3s otisky

Program pro extrakci TLS otisku ze zašifrované síťové komunikace byl spuštěn nad všemi zaznamenanými záznamy komunikace pro jednotlivé aplikace. Výsledný dataset byl rozšířen o nové atributy MD5 heš pro algoritmus JA3 a JA3s. Při detailní analýze vygenerovaných heší byla zjištěna jejich duplicita mezi aplikacemi.

Například heš *6f5e62edfa5933b1332ddf8b9fb3ef9d* byl vygenerován pomocí algoritmu JA3 celkem u devíti z deseti aplikací. Heš *9d9ce860f1b1cbef07b019450cb368d8* získaný výpočtem algoritmu JA3s byl vygenerován celkem pro čtyři aplikace z deseti. Přehled vygenerovaných heší a počet jejich duplicit v rámci deseti aplikací se nachází v tabulce 7.1. Heš, který byl vygenerován pro více jak jednu aplikaci, může způsobit nesprávné detekování aplikace. Z tabulky také vyplývá, že algoritmus JA3 generuje mnohem více duplicitních heší

Výsledný MD5 heš pro algoritmus JA3	Počet shodných hešů mezi aplikacemi	Výsledný MD5 heš pro algoritmus JA3s	Počet shodných hešů mezi aplikacemi
6f5e62edfa5933b1332ddf8b9fb3ef9d	9	9d9ce860f1b1cbef07b019450cb368d8	4
dbcb23ed726620bc44e6cebcb49eba84	4	eb1d94daa7e0344597e756a1fb6e7054	3
e9ec38c2b40ff3e300e9975dd7619902	4	5deff56d75697caa937c30eae4aeaf84	2
f79b6bad2ad0641e1921aef10262856b	3	e54965894d6b45ecb4323c7ea3d6c115	2
d8c87b9bfde38897979e41242626c2f3	2	f4febc55ea12b31ae17cfb7e614afda8	2
47fa89d5778a07c03b95ab3a2a9de52b	2	d7e12962b60127bdbe4f65f39221f9e8	2
775df03ff14c70439638df33df727f71	1	5b94af9bf6efc9dea416841602004fbb	2
6c0f0a346dcd84cb4b97a0d9382c53fd	1	00447ab319e9d94ba2b4c1248e155917	2
6ec2896feff5746955f700c0023f5804	1	b44baa8a20901c5663b3a9664ba8a767	2
eaabed81520b23ea8a800b36bd7e359e	1	1249fb68f48c0444718e4d3b48b27188	2
...	...	...	...

Tabulka 7.1: Přehled vygenerovaných MD5 heší a jejich počet duplicit mezi deseti zkoumanými aplikacemi.

než v případě algoritmu JA3s, proto je vhodnější použít pro porovnání JA3s heši, případně kombinace JA3 a JA3s haší.

7.2 Porovnání existujících řešení

Nově vytvořené řešení v rámci této práce bylo porovnáno s algoritmem JA3 a JA3s. Porovnání probíhalo nad prvním testovacím souborem, který obsahoval celkem 28 různých otisků aplikací. Deset otisků patřilo známým aplikacím, jako jsou aplikace *Kaufland*, *Lidl* a další. Ostatní otisky patřily reklamním serverům nebo dříve neviděným aplikacím. Výsledek porovnání se nachází v tabulce 7.2.

Souhrn	**Algoritmus JA3**	**Algoritmus JA3S**	**Algoritmus JA3 + JA3s**	**Algoritmus JA3 + JA3s + SNI**	**Navržená metoda T = 75%**
Vícenásobná detekce	5	1	1	0	0
Správna detekce	7	17	8	26	28
Špatná detekce	16	10	19	2	0
Přesnost	25 %	60,71 %	28,57 %	92,86 %	100 %

Tabulka 7.2: Porovnání nové detekční metody s algoritmy JA3 a JA3s.

Tabulka obsahuje celkem pět algoritmů, a to JA3, JA3s, kombinaci JA3 a JA3s, kombinace JA3 s JA3s a SNI a nově vytvořenou metodu. Algoritmus JA3 správně detekoval 7 z 28 otisků, 16 z 28 detekoval špatně a u 5 otisků nastala vícenásobná detekce. Vícenásobná detekce znamená, že heš neznámé aplikace byl shodný s více než jednou známou aplikací. Celková přesnost algoritmu JA3s, který svůj heš generuje z hodnot paketu *Server Hello* a správně detekoval 17 z 28 otisků, je více než dvojnásobná vůči algoritmu JA3.

Pod spojením algoritmů JA3 a JA3s si lze představit dvojité porovnání. Nejprve se provede klasické porovnání jako v případě algoritmu JA3 a pokud jsou heši shodné, tak se

Algoritmus JA3 MD5 heš	**Algoritmus JA3s MD5 heš**	**Název serveru (SNI)**	**Detekce**
6f5e62edfa5933b-1332ddf8b9fb3ef9d	00447ab319e9d94b-a2b4c1248e155917	pro.mapy.cz	Ano
6f5e62edfa5933b-1332ddf8b9fb3ef9d	00447ab319e9d94b-a2b4c1248e155917	pro.mapy.cz	
eaabed81520b23-ea8a800b36bd7e359e	d154fcfa5bb4f0748-e1dd1992c681104	account.kaufland.com	Ne
6c0f0a346dcd84-cb4b97a0d9382c53fd	d154fcfa5bb4f0748-e1dd1992c681104	account.kaufland.com	
6f5e62edfa5933b-1332ddf8b9fb3ef9d	9d9ce860f1b1cbef0-7b019450cb368d8	cdn.blablacar.com	Ne
d8c87b9bfde3889-7979e41242626c2f3	9d9ce860f1b1cbef0-7b019450cb368d8	cdn.blablacar.com	

Tabulka 7.3: Detekce algoritmů JA3 a JA3s se spojením s SNI.

pokračuje s porovnáním jejich heší získaných pomocí algoritmu JA3s. Z toho vyplývá, že se musí rovnat jak MD5 heš pro paket *Client Hello*, tak MD5 heš vypočítaný z paketu *Server Hello*. Spojení těchto algoritmů dosahuje přesnosti nad 70 %.

Spojení algoritmů JA3 a JA3s s názvem komunikujícího serveru (SNI) dosahuje přesnosti nad 90 %. Jen u otisků s CN account.kaufland.com a cdn.blablacar.com nebyl tento algoritmus úspěšný. U obou aplikací byla důvodem špatné detekce MD5 heš algoritmu JA3. V tabulce 7.3 lze vidět tři porovnání podle MD5 heší algoritmu JA3 a JA3s se spojených s SNI. První porovnání bylo úspěšné pro všechny tři hodnoty. Zbylá dvě porovnání byla neúspěšná z důvodu rozdílnosti MD5 heši algoritmu JA3, a proto je výsledná přesnost tohoto algoritmus 92.86 %.

Nejpřesnější je nově navržená metoda s prahem 75 %, která dokázala správně detekovat všechny otisky.

Kapitola 8

Závěr

Tato práce je zaměřena na problematiku identifikace mobilních aplikací v šifrovaném provozu. Cílem práce bylo navrhnout architekturu pro identifikaci mobilních aplikací pomocí TLS komunikace. Architektura se skládá ze získávání záznamů mobilní komunikace, extrakce jednotlivých atributů z paketů, jejich porovnání, poslední část architektury se zaměřuje na identifikaci dvou TLS otisků. Dále bylo potřeba navrhnout vhodné metriky a jejich porovnání. Jde zde popsána problematika šifrovaného síťového provozu a vytváření otisků aplikací. Byl popsán způsob extrakce jednotlivých atributů a jejich úprava před porovnáváním. Práce se rovněž zabývala problematikou porovnávání extrahovaných hodnot, implementací navrhnuté architektury a experimenty.

Pro výslednou práci bylo potřeba detailně nastudovat protokol TLS, aktuální řešení pro identifikaci aplikací ze šifrované komunikace a vytváření otisků aplikací pomocí algoritmů JA3 a JA3s.

Výsledná práce obsahuje návrh a implementaci architektury pro identifikaci mobilních aplikací na základě extrahovaných atributů z TLS paketů. Extrahované atributy jsou jako TLS otisky ve formátu JSON ukládány do datasetu. Důraz byl kladen na přesnost jednotlivých metrik, kvalitu vybraných atributů a na určení porovnávacího prahu T, který byl ve výsledku určen na 75 %.

Architektura umožňuje nejen porovnat otisky aplikací z vytvořeného datasetu s nově vytvořeným záznamem zašifrované komunikace a identifikovat aplikace, ale také možnost naučení se nových aplikací z porovnávaného souboru. Učení nových aplikací probíhá pomocí manuálního nebo automatického řešení. Architektura byla naprogramována pomocí programovacího jazyka Python 3 s rozšiřující knihovou PyShark a byla rozdělená do dvou skriptů. První skript extrahuje atribut TLS paketů za zašifrované síťové komunikace a otisky aplikací, které ukládá do datasetu. Druhý skript slouží k porovnání vytvořených otisků v datasetu s otisky ze zašifrované komunikace, které následně porovnává a identifikuje. Druhý skript ještě umožňuje aktualizovat známý otisk. případně vytvořit otisk pro novou aplikaci.

Výslednou architekturu lze integrovat například do systému detekce průniku (IDS), jako je nástroj Suricata, který dovoluje implementovat vlastní moduly. Administrátor by tento modul mohl nasadit v základní konfiguraci s malým datasetem a nechat algoritmus automaticky užít nové aplikace. Administrátor by tak mohl získat detailnější přehled o existenci jednotlivých aplikací v jeho síti. Na základě těchto znalostí by následně mohl lépe nastavit síťové zařízení firewall nebo například analyzovat zašifrovaný datový tok pro určení nejpoužívanějších aplikací v čase.

Do budoucna plánuji rozšířit architekturu o detekci aplikací nad ostatními operačními systémy, případně rozšířit jednotlivé metriky a počty atributů. Následně plánuji rozšířit

architekturu o vizualizaci dat pomocí webové stránky, kde budou veškerá data přehledně zobrazena a zařazena do skupin.

Literatura

[1] A. Langley, M. H. a Turner, S. *Elliptic Curves for Security.* RFC 8446, 8. 2018.

[2] Al Naami, K., Chandra, S., Mustafa, A., Khan, L., Lin, Z. et al. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In: *Proceedings of the 32nd Annual Conference on Computer Security Applications.* 2016, s. 177–188. ISBN 9781450347716.

[3] Alan, H. F. a Kaur, J. Can Android applications be identified using only TCP/IP headers of their launch time traffic? In: *Proceedings of the 9th ACM conference on security & privacy in wireless and mobile networks.* 2016, s. 61–66. ISBN 9781450342704.

[4] Alsmadi, I. M. a AlEroud, A. SDN-based real-time IDS/IPS alerting system. In: *Information Fusion for Cyber-Security Analytics.* Springer, 2017, s. 297–306. ISBN 978-3-319-44256-3.

[5] Anderson, B. a McGrew, D. Identifying Encrypted Malware Traffic with Contextual Flow Data. In: Guan, Z., ed. *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security.* New York, NY, USA: Association for Computing Machinery, 2016, s. 35–46. AISec '16. DOI: 10.1145/2996758.2996768. ISBN 9781450345736.

[6] Anderson, B. a McGrew, D. Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity. In: *Proceedings of the 23rd ACM SIGKDD International Conference on knowledge discovery and data mining.* 2017, s. 1723–1732. ISBN 9781450348874.

[7] Anderson, B. a McGrew, D. A. Accurate TLS Fingerprinting using Destination Context and Knowledge Bases. *CoRR.* 1. vyd. 2020, abs/2009.01939, č. 1.

[8] Bartl, E. Teorie informace. *Matematika–Fyzika–Informatika.* 4. 2015, sv. 24, č. 3, s. 219–228.

[9] Benjamin, D. *RFC 8701 Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility.* Technical report, Internet Engineering Task Force, 2020.

[10] Cardellini, V., Colajanni, M. a Yu, P. S. Dynamic load balancing on web-server systems. *IEEE Internet computing.* IEEE. 1999, sv. 3, č. 3, s. 28–39.

[11] Chan, X. a Liu, G. Discussion of one improved hash algorithm based on MD5 and SHA1. In: Citeseer. *Proceedings of the World Congress on Engineering and Computer Science.* 2007, sv. 2007.

[12] D. Brown, R. H. a Polk, T. *Elliptic Curve Cryptography Subject Public Key Information*. RFC 5480, 3. 2009.

[13] Dierks, T. a Rescorla, E. *The Transport Layer Security (TLS) Protocol Version 1.2*. RFC 5246 (Proposed Standard), 8. 2008.

[14] Driscoll, M. *The Illustrated TLS 1.2 Connection*. 2022. Dostupné z: https://tls.ulfheim.net/.

[15] Driscoll, M. *The Illustrated TLS 1.3 Connection*. 2022. Dostupné z: https://tls13.ulfheim.net/.

[16] Eastlake, D. *Transport Layer Security (TLS) Extensions: Extension Definitions*. RFC 6066, 1. 2011.

[17] G. Fairhurst, L. W. *Advice to link designers on link Automatic Repeat reQuest (ARQ)*. RFC 3366, 8. 2002.

[18] Gancheva, Z., Sattler, P. a Wüstrich, L. TLS Fingerprinting Techniques. *Network*. 1. vyd. 2020, sv. 15, č. 1.

[19] J. Weil, V. K. a Donley, C. *IANA-Reserved IPv4 Prefix for Shared Address Space*. RFC 6598, 4. 2012.

[20] Liu, J., Zeng, Y., Shi, J., Yang, Y., Wang, R. et al. MalDetect: A Structure of Encrypted Malware Traffic Detection. *Computers, Materials and Continua*. 2019, sv. 60, č. 2.

[21] M. West, S. M. *TCP/IP Field Behavior*. RFC 4413, 3. 2006.

[22] Matoušek, P., Burgetová, I., Ryšavý, O. a Victor, M. On Reliability of JA3 Hashes for Fingerprinting Mobile Applications. In: Springer. *International Conference on Digital Forensics and Cyber Crime*. 2020, s. 1–22. ISBN 978-3-030-68733-5.

[23] Rescorla, E. *The Transport Layer Security (TLS) Protocol Version 1.3*. RFC 8446, 8. 2018.

[24] S. Santesson, S. B. a Polk, W. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC 5280, 3. 2008.

[25] Snášel, D. *Encrypted communication dataset for ten mobile applications*. IEEE Dataport, 2022. DOI: 10.21227/nhqh-mm15. Dostupné z: https://dx.doi.org/10.21227/nhqh-mm15.

[26] Towsley, D. The stutter go back-N ARQ protocol. *IEEE transactions on Communications*. IEEE. 1979, sv. 27, č. 6, s. 869–875.

[27] Vais, V. *TEORETICKÁ INFORMÁTIKÁ 2. C Á ST*. 2015.

[28] van Ede, T., Bortolameotti, R., Continella, A., Ren, J., Dubois, D. et al. FlowPrint: Semi-Supervised Mobile-App Fingerprinting on Encrypted Network Traffic. In: *Network and Distributed System Security Symposium (NDSS)*. 27. vyd. Internet Society, 2. 2020. DOI: 10.14722 dss.2020.24412. ISBN 1-891562-61-4. Network and Distributed System Security Symposium.

[29] Zimmermann, H. OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications*. 1980, sv. 28, č. 4, s. 425–432. DOI: 10.1109/TCOM.1980.1094702.

Příloha A

Velikost TLS paketů

Tabulka A.1 obsahuje velikosti jednotlivých paketů z ustanovení TLS spojení v bytech pro všechny zkoumané aplikace. Sloupce s označením CH obsahují velikost paketu *Client Hello*. SH znamená *Server Hello*, sloupec s označením SC obsahuje velikost paketu *Server Certificate*. CKE je označení pro *Client Key Exchange* paket. Následující sloupec je určen pro *Change Cipher Spec*. Další dva sloupce obsahují součty velikosti paketů pro klientskou a serverovou stranu. Poslední sloupec obsahuje poměr mezi klientskou a serverovou stranou.

Aplikace	**CH**	**SH**	**SC**	**CKE**	**CCS**	**Klient**	**Server**	**Poměr**
Bazoš_0	243	1514	1514	828	159	1071	3187	2.98
Bazoš_1	243	1514	1514	828	159	1071	3187	2.98
Bazoš_2	243	1514	1514	828	159	1071	3187	2.98
Benzina_0	252	1514	1268	192	117	444	2899	6.53
Benzina_1	252	1514	1268	192	117	444	2899	6.53
Benzina_2	252	1514	1268	192	117	444	2899	6.53
BlaBlaCar_0	248	1484	858	159	358	407	2700	6.63
BlaBlaCar_1	248	1484	858	159	358	407	2700	6.63
BlaBlaCar_2	248	1484	858	159	358	407	2700	6.63
Booxy_0	241	1514	1501	159	117	400	3132	7.83
Booxy_1	241	1514	1501	159	117	400	3132	7.83
Booxy_2	241	1514	1501	159	117	400	3132	7.83
ČHMU_2	238	1514	1321	147	312	385	3147	8.17
ČHMU_1	238	1514	1320	147	312	385	3146	8.17
ČHMU_0	238	1514	1439	147	312	385	3265	8.48
Kaufland_0	225	1514	1514	1514	334	1739	3362	1.93
Kaufland_1	225	1514	1514	1514	334	1739	3362	1.93
Kaufland_2	225	1514	1514	1514	334	1739	3362	1.93
Lidl_0	249	1514	548	216	348	465	2410	5.18
Lidl_1	249	1514	548	216	348	465	2410	5.18
Lidl_2	249	1514	548	216	348	465	2410	5.18
Mapy_0	246	1294	1003	159	324	405	2621	6.47
Mapy_1	246	1294	1003	159	324	405	2621	6.47
Mapy_2	246	1294	2231	159	324	405	3849	9.50
Shazam_0	248	1484	1014	159	358	407	2856	7.02
Shazam_1	248	1484	787	159	358	407	2629	6.46

Shazam_2	248	1484	787	159	358	407	2629	6.46
Waze_0	242	1484	1484	213	159	455	3127	6.87
Waze_1	233	1484	1484	129	159	362	3127	8.64
Waze_2	251	1484	1484	101	159	352	3127	8.88

Tabulka A.1: Velikosti paketů při ustavení TLS spojení [B].

Příloha B

Prvních deset paketu komunikace

Následující tabulka obsahuje prvních deset paketů TLS komunikace a to včetně navázání spojení. Tabulka je rozdělena na dva běhy jedné aplikace a obsahuje několik sloupců. První a třetí sloupec ukazují pořadí zaslaných paketů. Druhý a čtvrtý sloupec vyobrazuje velikosti paketů, které klient odeslal na server. Třetí a poslední sloupec obsahuje velikost paketu, které byli odeslány ze serveru na klientskou stanici. Poslední dva sloupce tabulek pro jednotlivé aplikace pak obsahují součet velikostí paketů a také jejich průměrnou velikost.

Bazoš					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	243		1.	243	
2.		1514	2.		1514
3.		1514	3.		1514
4.		828	4.		828
5.	159		5.	159	
6.		340	6.		340
7.		135	7.	243	
8.	119		8.		135
9.	697		9.	205	
10.		117	10.		1514
Součet [B]	1218	4448	**Součet [B]**	850	5845
Průměr [B]	304,5	741,33	**Průměr [B]**	212,5	974,17

Benzina					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	252		1.	252	
2.		1514	2.		1514
3.		1268	3.		1268
4.	192		4.	192	
5.		117	5.		117
6.	484		6.	484	
7.		1514	7.		1514
8.		1155	8.		1151

9.		100	9.		100
10.	1514		10.	1514	
Součet [B]	2442	5668	**Součet [B]**	2442	5664
Průměr [B]	610,5	944,67	**Průměr [B]**	610,5	944

BlaBlaCar					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	248		1.	248	
2.		1484	2.		1484
3.		1042	3.		1042
4.	159		4.	159	
5.		358	5.		358
6.		135	6.		135
7.	205		7.	248	
8.		104	8.	205	
9.	104		9.		1484
10.	248		10.		1042
Součet [B]	964	3123	**Součet [B]**	860	5545
Průměr [B]	192,8	624,60	**Průměr [B]**	215	924,17

Booxy					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	241		1.	241	
2.	241		2.	241	
3.		1514	3.		1514
4.		413	4.		413
5.		413	5.		1514
6.	159		6.		413
7.		117	7.	159	
8.		135	8.	159	
9.	426		9.		117
10.		104	10.		135
Součet [B]	1067	2696	**Součet [B]**	800	4106
Průměr [B]	266,75	449,33	**Průměr [B]**	200	684,33

ČHMU					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	238		1.	238	
2.		1514	2.		1514
3.		1321	3.		1439
4.	147		4.	238	
5.		312	5.		1514
6.		123	6.		1440
7.	288		7.	147	
8.		92	8.	147	
9.	92		9.		312
10.		1022	10.		123
Součet [B]	765	4384	**Součet [B]**	770	6342
Průměr [B]	191,25	730,67	**Průměr [B]**	192,5	1057

Kaufland					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	225		1.	225	
2.		1514	2.		1514
3.		1514	3.		1514
4.		1514	4.		1514
5.		334	5.		334
6.	224		6.	225	
7.	225		7.	224	
8.		117	8.		1514
9.	225		9.		1514
10.	343		10.		1514
Součet [B]	1242	4993	**Součet [B]**	674	9418
Průměr [B]	248,4	998,60	**Průměr [B]**	224,67	1345,43

Lidl					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	249		1.	249	
2.		1514	2.		1514
3.		548	3.		548
4.	216		4.	216	
5.		348	5.		348
6.	535		6.	535	
7.		1514	7.		1514
8.		111	8.		127
9.	252		9.	252	
10.		1514	10.	254	
Součet [B]	1252	5549	**Součet [B]**	1506	4051
Průměr [B]	313	924,83	**Průměr [B]**	301,2	810,2

Mapy					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	246		1.	246	
2.		1294	2.		1294
3.		1003	3.		2231
4.	159		4.	159	
5.		324	5.		324
6.	205		6.	119	
7.		150	7.		141
8.	395		8.	479	
9.	246		9.		104
10.		150	10.		190
Součet [B]	1251	2921	**Součet [B]**	1003	4284
Průměr [B]	250,2	584,20	**Průměr [B]**	250,75	714

Shazam					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	248		1.	246	
2.		1484	2.		1484
3.		787	3.		1351
4.	159		4.	159	
5.		358	5.		358
6.	984		6.		135
7.		285	7.	205	
8.	97		8.		104
9.	245		9.	369	
10.		1514	10.		276
Součet [B]	1733	4428	**Součet [B]**	979	3708
Průměr [B]	346,6	885,6	**Průměr [B]**	244,75	618

Waze					
První běh aplikace			**Druhý běh aplikace**		
Pořadí	**Klient [B]**	**Server [B]**	**Pořadí**	**Klient [B]**	**Server [B]**
1.	242		1.	251	
2.		1484	2.		1484
3.		1484	3.		1484
4.		213	4.		101
5.	159		5.	159	
6.		358	6.		358
7.	710		7.	1088	
8.		1484	8.	233	
9.		1484	9.		1484
10.		1484	10.		1484
Součet [B]	1111	7991	**Součet [B]**	1731	6395
Průměr [B]	370,33	1141,57	**Průměr [B]**	432,75	1065,83

Tabulka B.1: Statistická data pro prvních deset paketů.

Příloha C

Prvních pět vteřin komunikace

Následující tabulky obsahují statistická data pro prvních pět vteřin komunikace. Každá pod tabulka je rozdělena na dva běhy aplikace. Každý běh aplikace je rozdělen do sloupců s názvem *Klient* a *Server*. Sloupec *Klient* obsahuje velikosti a počet paketů zaslaných od klienta na server, pro sloupec *Server* jsou hodnoty totožné je pro případ odeslaných paketů ze serveru na klienta. Pro každou každý běh jednotlivé aplikace je zde spočítán součet velikosti paketů v bytech a jejich průměrná velikost. Poslední řádek obsahuje počet zaslaných paketů.

Bazoš					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	6826	106579	**Součet velikosti paketu [B]**	6717	82675
Průměrná velikost paketu [B]	189,61	1146,01	**Průměrná velikost paketu [B]**	197,56	1033,44
Počet paketů	36	93	**Počet paketů**	34	80

Benzina					
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
První běh aplikace			**Druhý běh aplikace**		
Součet velikosti paketu [B]	11878	23797	**Součet velikosti paketu [B]**	11878	25518
Průměrná velikost paketu [B]	625,16	793,23	**Průměrná velikost paketu [B]**	625,16	773,27
Počet paketů	19	30	**Počet paketů**	19	33

BlaBlaCar					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	8706	24096	**Součet velikosti paketu [B]**	8135	23733
Průměrná velikost paketu [B]	322,44	587,71	**Průměrná velikost paketu [B]**	312,88	641,43
Počet paketů	27	41	**Počet paketů**	26	37

Booxy					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	14877	71525	**Součet velikosti paketu [B]**	11930	41861
Průměrná velikost paketu [B]	252,15	715,25	**Průměrná velikost paketu [B]**	253,83	543,65
Počet paketů	59	100	**Počet paketů**	47	77

ČHMU					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	1910	7100	**Součet velikosti paketu [B]**	3942	68015
Průměrná velikost paketu [B]	212,22	546,15	**Průměrná velikost paketu [B]**	197,10	1015,15
Počet paketů	9	13	**Počet paketů**	20	67

Kaufland					
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
První běh aplikace			**Druhý běh aplikace**		
Součet velikosti paketu [B]	13016	125449	**Součet velikosti paketu [B]**	10415	34299
Průměrná velikost paketu [B]	406,75	1150,91	**Průměrná velikost paketu [B]**	385,74	816,64
Počet paketů	32	109	**Počet paketů**	27	42

Lidl					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	44209	140058	**Součet velikosti paketu [B]**	37669	71710
Průměrná velikost paketu [B]	545,79	1045,21	**Průměrná velikost paketu [B]**	638,46	907,72
Počet paketů	81	134	**Počet paketů**	59	79

Mapy					
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
První běh aplikace			**Druhý běh aplikace**		
Součet velikosti paketu [B]	39863	69739	**Součet velikosti paketu [B]**	37901	293853
Průměrná velikost paketu [B]	394,68	617,16	**Průměrná velikost paketu [B]**	435,64	1883,67
Počet paketů	101	113	**Počet paketů**	87	156

Shazam					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	49664	3393263	**Součet velikosti paketu [B]**	21701	739841
Průměrná velikost paketu [B]	318,36	1348,67	**Průměrná velikost paketu [B]**	344,46	1325,88
Počet paketů	156	2516	**Počet paketů**	63	558

Waze					
První běh aplikace			**Druhý běh aplikace**		
Směr komunikace	Klient	Server	**Směr komunikace**	Klient	Server
Součet velikosti paketu [B]	49664	3393263	**Součet velikosti paketu [B]**	21701	739841
Průměrná velikost paketu [B]	318,36	1348,67	**Průměrná velikost paketu [B]**	344,46	1325,88
Počet paketů	156	2516	**Počet paketů**	63	558

Tabulka C.1: Prvních pět vteřin komunikace včetně ustanovení spojení.

Příloha D

Experimenty pro určení prahu T

Pro stanovení prahu T bylo provedeno pět experimentů nad testovací sadou. Z každého souboru byli vytvořeny otisky, které se následně porovnávali s již vytvořeným datasetem. Na základě podobností mezi novými otisky s otisky v databázi se vytvořila chybová matice, která popisuje správnost detekovaných a nedetekovaných aplikací. Chybová matice byla vypočítána pro práh 60 % až 100 %. Z těchto matic lze následovně vyčíst, že nejlepší práh T je 75 %. Následující tabulky obsahují detekované i nedetekované aplikace z testovacích souborů při nastavení prahu právě na 75 %.

První testovací soubor s prahem 75 %			
Detekované aplikace			
Common Name	**Podobnost**	**Aplikace**	**Klasifikace**
account.kaufland.com	87.20%	kaufland	True positive
app.kaufland.net	96.40%	kaufland	True positive
*.sc.omtrdc.net	77.80%	kaufland	True positive
sync.kaufland.de	94%	kaufland	True positive
*.lidl-flyer.com	91%	lidl	True positive
*.bazos.cz	87.20%	bazos	True positive
vectmap.mapy.cz	86.10%	mapy	True positive
*.mapy.cz	100%	mapy	True positive
blablacar.com	100%	blablacar	True positive
cdn.blablacar.com	87.40%	blablacar	True positive

Nedetekované aplikace		
Common Name	**Podobnost**	**Klasifikace**
firebaseinstallations. googleapis.com	35.40%	True negative
*.facebook.com	16.0%	True negative
assets.adobedtm.com	15.50%	True negative
*.demdex.net	27.71%	True negative
*.lokalise.co	27.20%	True negative
crashlytics.com	26.30%	True negative
*.google-analytics.com	10.50%	True negative
firebaseremoteconfig.googleapis.com	30.90%	True negative

thumbr.io	30.80%	True negative
*.places.adobe.com	19.0%	True negative
app-ks.seznam.cz	43.20%	True negative
*.cloudfront.net	18.0%	True negative
*.privacy-center.org	19.0%	True negative
y.ssl.fastly.net	20.0%	True negative
crashlyticsreports-pa. googleapis.com	35.40%	True negative
app.adjust.com	23.70%	True negative
store. steampowered.com	22.0%	True negative
a248.e.akamai.net	20.0%	True negative

Tabulka D.1: Přehled všech hodnoty a detekcí pro první testovací soubor s prahem T = 75 %.

První testovací soubor s prahem 75 %			
Detekované aplikace			
CN	**Podobnost**	**Aplikace**	**Klasifikace**

Nedetekované aplikace		
CN	**Podobnost**	**Klasifikace**
android.clients.google.com	8%	True negative
firebaseremoteconfig.googleapis.com	35.40%	True negative
*.facebook.com	10.50%	True negative
*.spotifycdn.com	19%	True negative
crashlytics.com	30.80%	True negative
app.adjust.com	22.70%	True negative
*.spotify.com	35.40%	True negative
*.branch.io	19%	True negative
*.scdn.co	24.50%	True negative
*.scorecardresearch.com	18%	True negative
a248.e.akamai.net	27.20%	True negative
*.google-analytics.com	26.20%	True negative
config.teams.microsoft.com	9.83%	True negative
teams.events.data.microsoft.com	18%	True negative
chatsvcagg.teams.microsoft.com	19%	True negative
login.microsoftonline.com	17%	True negative
*.teams.cdn.office.net	25.50%	True negative
*.vo.msecnd.net	19%	True negative
*.appcenter.ms	30.80%	True negative
go.microsoft.com	30%	True negative
mamservice.manage.microsoft.com	10%	True negative
presence.teams.microsoft.com	18%	True negative
emea.ng.msg.teams.microsoft.com	17%	True negative

westeurope-prod-4. notifications.teams.microsoft.com	17%	True negative
config.office.net	28.20%	True negative
api.flightproxy.teams.microsoft.com	8.50%	True negative
whiteboard.microsoft.com	27%	True negative
codepush.teams.microsoft.com	17%	True negative
trouter2-azsc-ukwe-1-b. trouter.teams.microsoft.com	1%	True negative
static.whatsapp.net	10%	True negative
sni.cloudflaressl.com	56.20%	True negative
discord.gg	26.20%	True negative
discord.media	26.20%	True negative
accounts.google.com	10%	True negative

Tabulka D.2: Přehled všech hodnoty a detekcí pro druhý testovací soubor s prahem T = 75 %.

První testovací soubor s prahem 75 %			
Detekované aplikace			
CN	**Podobnost**	**Aplikace**	**Klasifikace**
accounts.lidl.com	85%	lidl	True positive
segments.lidlplus.com	92.80%	lidl	True positive
appgateway.lidlplus.com	89.20%	lidl	True positive
*.lidl-flyer.com	82%	lidl	True positive
lidlplusprod.blob .core.windows.net	93.50%	lidl	True positive
ms.cbdb.cz	86.80%	booxy	True positive

Nedetekované aplikace		
CN	**Podobnost**	**Klasifikace**
*.facebook.com	10.50%	True negative
*.api.kochava.com	30.80%	True negative
sweatco.in	19%	True negative
android.clients.google.com	10%	True negative
*.bugsnag.com	30.80%	True negative
codepush.appcenter.ms	18%	True negative
cognito-identity. eu-west-1.amazonaws.com	27.20%	True negative
sns.eu-west-1.amazonaws.com	17%	True negative
firebaseinstallations.googleapis.com	35.40%	True negative
appstore.huawei.com	10%	True negative
crashlytics.com	30.80%	True negative
metrics.dt.hicloud.com	10%	True negative
accounts.google.com	10%	True negative
*.eu16.force.com	19%	True negative
firebaseremoteconfig.googleapis.com	35.40%	True negative

crashlyticsreports-pa.googleapis.com	35.40%	True negative
*.prod-general.stocard-backend.com	23.21%	True negative
*.mixpanel.com	31.80%	True negative
*.stocard-backend.com	26.20%	True negative
z-p4-graph.facebook.com	0%	True negative
mqtt-p4.facebook.com	0%	True negative
scontent.xx.fbcdn.net	0%	True negative
scontent-prg1-1.xx.fbcdn.net	0%	True negative

Tabulka D.3: Přehled všech hodnoty a detekcí pro třetí testovací soubor s prahem T = 75 %.

První testovací soubor s prahem 75 %			
Detekované aplikace			
CN	**Podobnost**	**Aplikace**	**Klasifikace**
rtproxy-row.waze.com	100%	waze	True positive
cres.waze.com	100%	waze	True positive
inbox-row.waze.com	100%	waze	True positive
tts.waze.com	84.70%	waze	True positive
rtproxy-l7-row-gcp.waze.com	100%	waze	True positive
*.waze.com	100%	waze	True positive
social-row.waze.com	100%	waze	True positive
ctiles-row.waze.com	100%	waze	True positive
sdk.waze.com	100%	waze	True positive

Nedetekované aplikace		
CN	**Podobnost**	**Klasifikace**
mail.google.com	26.20%	True negative
notifications-pa.googleapis.com	22.07%	True negative
outlook.office365.com	9%	True negative
*.facebook.com	10.50%	True negative
*.bugsnag.com	30.80%	True negative
*.securedtouch.com	30.80%	True negative
*.wish.com	30.80%	True negative
siftscience.com	22.21%	True negative
*.riskified.com	23.21%	True negative
firebaseremoteconfig.googleapis.com	27.30%	True negative
static.securedtouch.com	10%	True negative
www.paypalobjects.com	19%	True negative
api.braintreegateway.com	17%	True negative
canary.contestimg.wish.com	18%	True negative
www.google.com	21.80%	True negative
wallet.google.com	10%	True negative
*.google-analytics.com	30.80%	True negative
ipws2.timetable.cz	33.50%	True negative

row-advil.waze.com	47%	True negative
login.microsoftonline.com	9%	True negative
*.g.doubleclick.net	17.20%	True negative
crashlytics.com	30.80%	True negative
*.hit.gemius.pl	18%	True negative
resources.crws.cz	17%	True negative
main.crws.cz	27%	True negative
tpc.googlesyndication.com	11%	True negative
www.gstatic.com	10%	True negative
lh3.googleusercontent.com	10%	True negative
www.googletagservices.com	10%	True negative
cm.g.doubleclick.net	10%	True negative
s0.2mdn.net	10%	True negative
*.search.spotxchange.com	3%	True negative
connectivitycheck.gstatic.com	30.80%	True negative

Tabulka D.4: Přehled všech hodnoty a detekcí pro čtvrtý testovací soubor s prahem T = 75 %.

První testovací soubor s prahem 75 %			
Detekované aplikace			
CN	**Podobnost**	**Aplikace**	**Klasifikace**
*.sc.omtrdc.net	79.20%	kaufland	True positive
*.bazos.cz	92.60%	bazos	True positive
vectmap.mapy.cz	89.70%	mapy	True positive
*.mapy.cz	100%	mapy	True positive
blablacar.com	100%	blablacar	True positive

Nedetekované aplikace		
CN	**Podobnost**	**Klasifikace**
data.mistat.intl.xiaomi.com	18.20%	True negative
*.facebook.com	10.50%	True negative
assets.adobedtm.com	20%	True negative
*.demdex.net	19%	True negative
mbs.internetbanka.cz	28.20%	True negative
firebaseremoteconfig.googleapis.com	33.60%	True negative
*.google-analytics.com	9.50%	True negative
img.alicdn.com	15.70%	True negative
*.aliexpress.com	21.70%	True negative
*.alicdn.com	18.20%	True negative
ru.aliexpress.com	21%	True negative
*.alibaba.com	17%	True negative
www.google.com	21.80%	True negative
userlocation.googleapis.com	10%	True negative
*.bugsnag.com	35.40%	True negative

*.wish.com	30.80%	True negative
*.securedtouch.com	30.80%	True negative
*.riskified.com	23.21%	True negative
siftscience.com	30.80%	True negative
main.cdn.wish.com	19%	True negative
static.securedtouch.com	10%	True negative
api.braintreegateway.com	17%	True negative
wallet.google.com	10%	True negative
*.cloudfront.net	18%	True negative
y.ssl.fastly.net	20%	True negative
connectivitycheck.gstatic.com	30.80%	True negative
app.adjust.com	23.70%	True negative

Tabulka D.5: Přehled všech hodnoty a detekcí pro pátý testovací soubor s prahem T = 75 %.

První testovací soubor s prahem 75 %			
Detekované aplikace			
CN	**Podobnost**	**Aplikace**	**Klasifikace**
accounts.lidl.com	100%	lidl	True positive
segments.lidlplus.com	100%	lidl	True positive
account.kaufland.com	100%	kaufland	True positive
app.kaufland.net	87.20%	kaufland	True positive
*.sc.omtrdc.net	94.60%	kaufland	True positive
sync.kaufland.de	100%	kaufland	True positive
*.lidl-flyer.com	100%	lidl	True positive
ms.cbdb.cz	100%	booxy	True positive
*.benzina-platby.cz	96%	benzina	True positive
	100%		
Nedetekované aplikace			
CN	**Podobnost**		**Klasifikace**
appstore.huawei.com	11%		True negative
crashlytics.com	23.60%		True negative
metrics.dt.hicloud.com	10.71%		True negative
android.clients.google.com	8.00%		True negative
app.adjust.com	23.70%		True negative
appgateway.lidlplus.com	24.40%		True negative
firebaseremoteconfig.googleapis.com	18%		True negative
*.device.marketingcloudapis.com	19.20%		True negative
data.mistat.intl.xiaomi.com	26.20%		True negative
*.eu16.force.com	17%		True negative
*.facebook.com	15.50%		True negative
assets.adobedtm.com	30.80%		True negative
*.demdex.net	27.20%		True negative
*.lokalise.co	31.80%		True negative

thumbr.io	19%	True negative
*.places.adobe.com	10%	True negative
app-measurement.com	27.90%	True negative
firebaseinstallations.googleapis.com	20.00%	True negative

Tabulka D.6: Přehled všech hodnoty a detekcí pro šestý testovací soubor s prahem T = 75 %.

Příloha E

Uživatelský manuál

Architektura pro identifikaci mobilních aplikací ze zašifrované síťové komunikace byla naimplementovaná v programovacím jazyce Python3 s využití knihovny PyShark. Architektura se skládá ze dvou skriptů, první skript slouží pro vytvoření základního datasetu pro druhý skript. Druhý skript porovnává TLS otisky z datasetu s otisky získanými ze zašifrované síťové komunikace a umožňuje přidat nebo aktualizovat otisky v datasetu.

Postup instalace

Potřebné rekvizity pro instalaci a spuštění skriptů:

- Interpret pro jazyk Python3 pro verzi 3.8+,
- nástroj TShark a
- adresář se skripty.

Nastavení virtuálního prostřední a instalace potřebných závislostí.

```
$ virtualenv -p python3 venv nebo python3 -m venv venv
$ source venv/bin/activate
$ pip install -r requirements.txt
```

Spuštění skriptu pro vytvoření datasetu

Skript pro vytvoření datasetu lze spustit pomocí příkazového řádku nebo terminálů.

```
$ python create_dataset.py --help # Vypíše nápovědu
```

Skript obsahuje celkem čtyři přepínače.

- -h nebo --help - Vypíše nápovědu pro spuštění skriptu.
- --input <název adresáře> - Adresář obsahující pcap soubory.
- --output <název datasetu> - Název nově vytvořeného datasetu.
- -v - Přepínač pro detailnější výpis kroků algoritmu.

Příklad spuštění skriptu pro vytvoření datasetu ze zaznamenané šifrované komunikace. Výsledný dataset ve formátu JSON je uložen ve stejném adresáři jako byl spuštěn skript.

```
$ python create_dataset.py --input pcaps --output dataset.json
```

Spuštění skriptu pro identifikace aplikací

Tento skript se spouští obdobným způsobem.

```
$ python identify_apk.py --help # Vypíše nápovědu
```

Skript obsahuje celkem deset následujících přepínačů.

- -h nebo --help - Vypíše nápovědu pro spuštění skriptu.
- --input-dataset <cesta k datasetu> - Dataset obsahující TLS otisku (výstup z prvního skiptu).
- --outpu-dataset <název datasetu> - Název aktualizovaného datasetu.
- --input-pcap <název pcap souboru> - Název souboru pcap se zašifrovanou TLS komunikací.
- --automatic-learning-mode - Aktualizace otisku a přidávání otisku do datasetu bude probíhat automaticky.
- --blocked-cn <název souboru> - Soubor se seznamem blokovaných Common Name.
- --treshold-t <float> - hodnota prahu T, v základu 0.75.
- -e - Vypočítá informační entropii pro jednotlivé atributy z datasetu a ukončí skript.
- -j - Vypíše výsledek pro algoritmus JA3 a JA3s.
- -v - Přepínač pro detailnější výpis kroků algoritmu.

Příklad spuštění skriptu pro identifikaci aplikací ze zašifrované síťově komunikace se vstupním a výstupním datasetem, zkoumaným souborem se síťovou komunikací a automatickým způsobem učení.

```
$ python identify_apk.py --input-dataset dataset.json --outpu-dataset
    dataset_update.json --automatic-learning-mode --input-pcap test/test_3.pcapng

Výstup skriptu:
----------------------------------------------------------------------
Detekované známe aplikace:
----------------------------------------------------------------------
1: accounts.lidl.com na 85.00 % detekovaná jako: lidl
2: segments.lidlplus.com na 92.80 % detekovaná jako: lidl
3: appgateway.lidlplus.com na 89.20 % detekovaná jako: lidl
4: *.lidl-flyer.com na 82.00 % detekovaná jako: lidl
5: lidlplusprod.blob.core.windows.net na 93.50 % detekovaná jako: lidl
6: ms.cbdb.cz na 90.80 % detekovaná jako: booxy
```

```
------------------------------------------------------------------
Neznámé detekované aplikace:
------------------------------------------------------------------
7: *.facebook.com na 19.00 % detekovaná jako: lidl
8: *.api.kochava.com na 30.80 % detekovaná jako: waze
9: sweatco.in na 19.00 % detekovaná jako: chmu
10: android.clients.google.com na 10.00 % detekovaná jako: mapy
11: *.bugsnag.com na 30.80 % detekovaná jako: waze
12: codepush.appcenter.ms na 18.00 % detekovaná jako: benzina
13: cognito-identity.eu-west-1.amazonaws.com na 27.20 % detekovaná jako: lidl
14: sns.eu-west-1.amazonaws.com na 17.00 % detekovaná jako: lidl
15: firebaseinstallations.googleapis.com na 35.40 % detekovaná jako: waze
16: appstore.huawei.com na 10.00 % detekovaná jako: chmu
17: crashlytics.com na 30.80 % detekovaná jako: blablacar
18: metrics.dt.hicloud.com na 10.00 % detekovaná jako: chmu
19: accounts.google.com na 10.00 % detekovaná jako: mapy
20: *.eu16.force.com na 19.00 % detekovaná jako: lidl
21: firebaseremoteconfig.googleapis.com na 35.40 % detekovaná jako: waze
22: crashlyticsreports-pa.googleapis.com na 35.40 % detekovaná jako: waze
23: *.prod-general.stocard-backend.com na 27.71 % detekovaná jako: lidl
24: *.mixpanel.com na 31.80 % detekovaná jako: waze
25: *.stocard-backend.com na 26.20 % detekovaná jako: shazam
26: z-p4-graph.facebook.com na 0.00 % detekovaná jako: bazos
27: mqtt-p4.facebook.com na 0.00 % detekovaná jako: bazos
28: scontent.xx.fbcdn.net na 0.00 % detekovaná jako: bazos
29: scontent-prg1-1.xx.fbcdn.net na 0.00 % detekovaná jako: bazos
------------------------------------------------------------------
Aplikace: ms.cbdb.cz, detekována jako: booxy na 90.80 % Byla úspěšně
    aktualizována.
Aplikace: accounts.lidl.com, detekována jako: lidl na 85.00 % Byla úspěšně
    aktualizována.
...
Byla vybrána aplikace: *.facebook.com, nejblíže aplikaci: lidl na 19.00 %
Byla přidána aplikace s názvem: facebook a CN: *.facebook.com
...
```

Adresářová struktura

/
- blocked_cn.txt - Blokované Common name pro automatické učení.
- create_dataset.py - Skript pro vytvoření datasetu.
- identify_apk.py - Skript pro identifikaci aplikací.
- Makefile - Soubor obsahující příklady spuštění skriptů.
- requirements.txt - Potřebné závislosti pro spuštění skriptů.
- src
  - apk_fingerprint.py - Vnitřní reprezentaci TLS otisku aplikace.
  - dataset_wrapper.py - Operace s TLS otisky v datasetu.
  - entropy.py - Operace pro výpočet informační entropie.
  - handshake.py - Soubor obsahující TLS typy a IP adresy.
  - ja3.py - Algoritmus pro JA3 a JA3s.
  - packets_extractor.py - Operace pro získání paketu ze záznamu.
- test - Adresář obsahující testovací pcap soubory.
  - test_0.pcapng
  - ...
  - test_5.pcapng
- pcaps - Adresář obsahující pcap soubory.
  - bazos_bazos_0_.pcapng
  - bazos_bazos_1_.pcapng
  - bazos_bazos_2_.pcapng
  - benzina_benzina_0_.pcapng
  - ...

Příloha F

Cloud a CDN sítě pro zkoumané aplikace

Aplikace	IP	Cloud	CDN
Bazoš	88.86.119.246	Ne	Ne
Benzina	91.231.171.211	Ne	Ne
	91.231.171.212	Ne	
Blablacar	34.117.9.118	Google Cloud	Ano Google
	35.244.237.205	Google Cloud	
	34.95.68.122	Google Cloud	
	35.190.32.124	Google Cloud	
Booxy	185.183.8.127	Ne	Ne
ČHMU	172.67.139.185	Cloudflare	Ano Cloudflare
	104.21.89.95	Cloudflare	
Kaufland	52.236.157.206	Azure	Ano Akamai
	35.181.18.61	Amazone	
	52.157.251.163	Azure	
	15.237.76.117	Amazone	
	23.9.15.213	Ne	
	77.87.187.36	Ne	
Lidl	51.105.123.133	Azure	Ano Akamai
	51.105.175.147	Azure	
	40.127.239.139	Azure	
	91.236.122.93	MyRacloud	
	40.68.232.16	Azure	
	51.104.185.154	Azure	
	20.54.8.177	Azure	
	20.82.218.32	Azure	
	40.114.188.208	Azure	
	51.105.123.85	Azure	

Aplikace	IP	Cloud	CDN
Mapy	77.75.77.28	Ne	Ano Seznam
	77.75.79.155	Ne	
	77.75.76.182	Ne	
	77.75.77.138	Ne	
	77.75.78.182	Ne	
	77.75.79.28	Ne	
	77.75.77.155	Ne	
	77.75.74.178	Ne	
Shazam	35.241.30.194	Google Cloud	Ano Google Fasty
	34.102.169.70	Google Cloud	
	34.120.233.61	Google Cloud	
	199.232.17.80	Fastly's edge cloud	
Waze	35.190.40.98	Google Cloud	Ano Google
	130.211.16.132	Google Cloud	
	107.178.245.42	Google Cloud	
	34.120.106.34	Google Cloud	
	130.211.20.72	Google Cloud	
	130.211.12.225	Google Cloud	
	35.190.76.168	Google Cloud	
	216.239.38.129	Google Cloud	
	34.120.141.74	Google Cloud	
	130.211.33.104	Google Cloud	
	34.120.172.68	Google Cloud	
	142.250.185.115	Google Cloud	
	35.186.206.199	Google Cloud	

Tabulka F.1: Cloud a CDN sítě pro IP adresy zkoumaných aplikací.