VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Identifikácia mobilných aplikácií pomocou otlačkov TLS

Přenos dat, počítačové sítě a protokoly

Bc. Marián Kapišinský

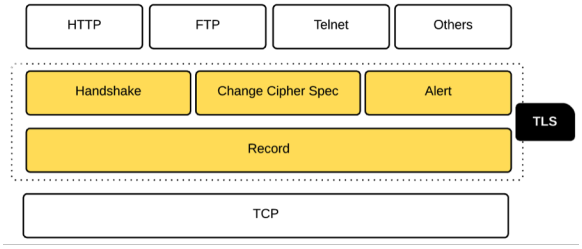
25. apríla 2021

Obsah

1 Úvod

Táto správa k projektu z predmetu *Přenos dat, počítačové sítě a protokoly* sa zaoberá identifikáciu mobilných aplikácií pomocou otlačkov TLS. Protokol TLS (Transport Layer Security) [1, 2] je protokol postavený nad protokolom TCP, teda medzi transportnou a aplikačnou vrstou TCP/IP modelu (Obr. 1.1), a jeho úlohou je zabezpečenie súkromia a integrity dát medzi dvomi komunikujúcimi aplikáciami. Skladá sa z dvoch častí a to z časti *TLS Handshake*, ktorá zabezpečuje dohodnutie parametrov protokolu, ako napr. verzia protokolu, kryptografické parametre pre výmenu zdieľaného kľúča, šifrovacích algoritmov, autentizáciu, atď, a časti *TLS Record*, ktorý zapúzdruje protokoly vyšších vrstiev a prenáša zašifrované dáta. Po nadviazaní TCP spojenia si teda obe strany oznámia podporované TLS parametre správami *Client Hello* a *Server Hello*. Keď sa obe strany dohodnú na parametroch, dáta aplikácie sú v zašifrovanej podobe prenášané pomocou TLS Record. Fungovanie protokolu TLS je zobrazené na Obr. 1.2.

Obr. 1.1: Transport Layer Security v TCP/IP modeli



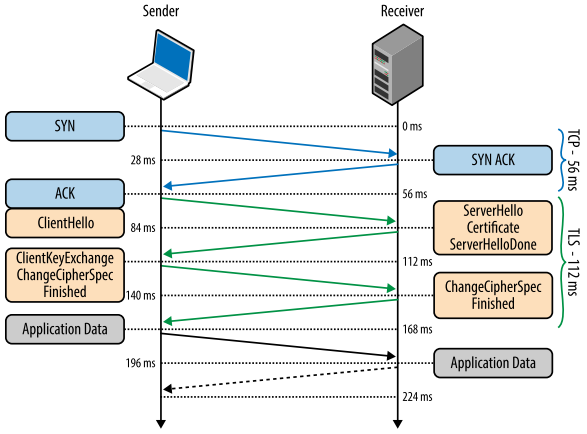
zdroj: https://medium.facilelogin.com/nuts-and-bolts-of-transport-layer-security-tls-2c5af298c4be

Z pohľadu tvorby otlačkov TLS sú podstatné správy Client Hello a Server Hello. Otlačok *JA3* [2] je MD5 hash vytvorený z políčok Client Hello – *TLS Handshake Version*,*Cipher Suite*, *Extensions*, *Supported Groups*, a *Elliptic Curve point format*. Otlačok *JA3S* [2] je MD5 hash vytvorený z polí Server Hello – *TLS Handshake Version*,*Cipher Suite*, *Extensions*. Okrem týchto dvoch otlačkov sa pre identifikáciu môže použiť aj *Server Name Indication*. Otlačky sa uložia do databázy vo vhodnom formáte a následne sa použijú na klasifikáciu otlačkov vytvorených z novej, neznámej, dátovej sady.

Úlohou projektu teda je

1. vytvorenie dátových sád pre 10 zvolených mobilných aplikácií,
2. získanie relevantných informácií z TLS Client Hello a Server Hello paketov,
3. vytvorenie otlačkov a ich uloženie do databáze,

Obr. 1.2: Nadviazanie TLS spojenia



zdroj: https://hpbn.co/transport-layer-security-tls/

4. vykonanie experimentovej na novej testovacej sade dát a zhodnotenie ich výsledkov.

Časť 2 obsahuje prehľad zvolených aplikácií a popis dátových sád použitých v tomto projekte. Časť 3 popisuje získanie a spracovanie dát, a tvorbu samotnej databázy. Časť 4 popisuje experimenty a diskutuje ich výsledky.

2 Tvorba Dátových Sád

Pre vytvorenie databázy TLS otlačkov je potrebá sada TLS Client Hello a TLS Server Hello paketov. Na to som využil Android Virtual Device (AVD), konkrétne Google Pixel 4, API 30 (Android 11.0), x86_64, Android Debugging Bridge (adb) pre inštaláciu aplikácií, a Wireshark s pluginom androiddump pre odchytanie paketov zo sieťového rozhrania AVD. Postupne som pre každú aplikáciu vytvoril príslušný *pcap* súbor zachytávajúci 5 behov aplikácie v časovom okne približne 500 sekúnd. Ďalej som vytvoril *pcap* súbor zachytávajúci rôzny počet behov všetých aplikácií v náhodnou poradií v časovom okne približne 700 sekúnd. Zo všetkých súborov som následne vyextrahoval všetky relevantné informácie pomocou programu `tshark` a uložil do vlastných súborov. Tabuľka 2.1 obsahuje prehľad zvolených aplikácií, ich verzie a počet zachytených TLS Client Hello a TLS Server Hello paketov. Posledný riadok tabuľky zobrazuje počet zachytených paketov pre testovaciu dátovú sadu. Na extrahovanie dát z *pcap* súborov som použil skript *extract.sh* (viď Dodatok A).

```
$ tshark -r <app>.pcap -T fields -E separator=";" -e ip.src \
-e ip.dst -e tcp.srcport -e tcp.dstport -e tls.handshake.type \
-e tls.handshake.version -e tls.handshake.ciphersuite \
-e tls.handshake.extension.type -e tls.handshake.extensions_server_name \
-e tls.handshake.extensions_supported_group \
-e tls.handshake.extensions_ec_point_format \
-R "tls.handshake.type==1 or tls.handshake.type==2" -2 > <app>-tlss.csv
```

Tabuľka 2.1: Prehľad zvolených aplikácií. ich verzí a počtu zachytených paketov

Aplikácia	**Verzia**	**Počet paketov**
Discord	68.0	168
Ebay	6.14.1.1	256
Flashscore	3.3.1	836
Ideme Vlakom	1.1.4	118
LinkedIn	4.1.558	67
Netflix	7.98.0 build 7 35414	340
Reddit	2021.14.0	560
Twitch	10.5.0.2	448
Twitter	8.88.0	224
Windy	12.0.0	610
*	-	1034

3 Tvorba Databázy Otlačkov TLS

Ako prvý krok spracovania dát som zo získaných dátových sád vyextrahoval všetky kombinácie <*ip.src;ip.dst;SNI*>, aby som vedel podľa získaných SNI vytvoriť zoznam klúčových slov pre každú aplikáciu. Pre túto úlohu som použil skript *get-sni-csv.py* (viď Dodatok A), získaný výstup som prešiel manuálne a určil príslušné kľúčové slová. Následne som opäť použil tento skript, ale s prepínačom *-k*, ktorý filtruje výstup podľa špecifikovaných kľúčových slov, čím som získal všetky rôzne SNI pre každú aplikáciu. Tie budú neskôr použité na filtráciu TLS Handshake-ov podľa SNI, tak ako to robí [2] a to z dôvodu odfiltrovania paketov, ktoré nepatria priamo aplikacií, ale rôznym API alebo serverom poskytujúcim reklamy danej aplikácií. Tabuľka 3.1 ukazuje získané kľúčové slová pre každú aplikáciu.

```
$ python3 get-sni-csv.py --csv <app>-tlss.csv
$ python3 get-sni-csv.py --csv <app>-tlss.csv -k <keyword1,keyword2,...>
```

Tabuľka 3.1: Prehľad kľúčových slov pre aplikácie

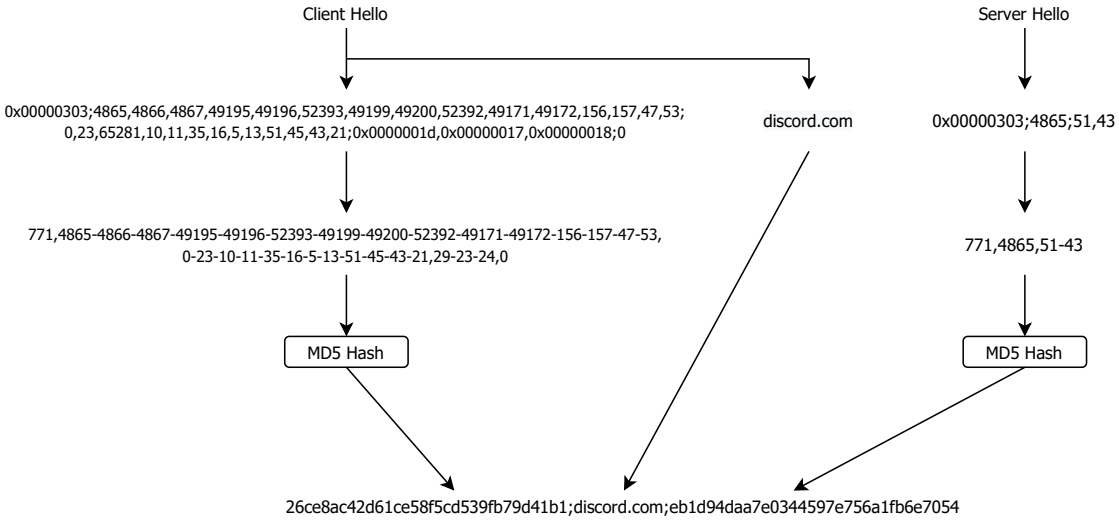
Aplikácia	**Kľúčové slová**
Discord	discord
Ebay	ebay
Flashscore	t.flashscore.sk, livesport.cz
Ideme Vlakom	slovakrail.sk, zssk.sk
LinkedIn	linkedin.com, licdn.com
Netflix	netflix.com, nflx
Reddit	reddit, redd.it
Twitch	twitch, ttvnw
Twitter	twitter.com, twimg.com, t.co
Windy	windyapp.co

Ďalším krokom je tvorba databázy otlačkov pre každú aplikáciu. Z dátovej sady aplikácie som najprv pre každú odpovedajúcu dvojicu Client Hello a Server Hello vytvoril JA3 a JA3S otlačky a spolu s SNI ich uložil do súboru CSV ako záznam vo formáte <*JA3;SNI;JA3S*>. Hodnoty SNI som filtroval podľa získaných SNI z predchádzajúceho kroku. Čo je potrebné spomenúť sú tzv. hodnoty *GREASE* definované v RFC 8701 [3]. Ide o randomizované hodnoty v poliach *Cipher Suite*, *Extensions*, *Supported Groups* a podľa [2], aj spolu s hodnotou 65281, je vhodné tieto hodnoty pred samotnou tvorbou otlačku odstrániť. Zníži sa tak počet unikátnych otlačkov, a takisto sa môže zvýšiť presnosť identifikácie aplikácie. Okrem tohto, je potrebné previesť hexadecimálne hodnoty to dekadického tvaru. Nakoniec, som spojil všetky

databázy jednotlivých aplikácií do jednej, kde záznam má formát *<APP;JA3;SNI;JA3S>*. Na túto úlohu som použil skripty *tlss2ja3.py* a *create_ db.sh* (viď Dodatok A). Príklad tvorby záznamu databázy otlačkov pre aplikáciu je uvedený na Obr. 3.1.

```
$ python3 tlss2ja3.py --csv <app>-tlss.csv --sni <app>-sni-filtered.csv
$ ./create_db.sh <app1-db.csv, app2-db.csv, ...>
```

Obr. 3.1: Tvorba otlačku TLS



4 Experimenty

Pred samotnými experimentami je potrebné zo všetkých TLS Handshake-ov z testovacej dátovej sady vytvoriť TLS otlačky rovnakým spôsobom ako pri tvorení databázy. Tých vzniklo presne **510**. Pre každý som manuálne určil aplikáciu, ktorej prislúchajú a následne som vykonal štyri experimenty metódou presného porovnania ("exact match") a to podľa:

1. JA3+JA3S+SNI
2. (JA3+JA3S+SNI)+(JA3+JA3S)
3. (JA3+JA3S+SNI)+(JA3+JA3S)+JA3
4. (JA3+JA3S+SNI)+(JA3S+SNI)

Pre zjednodušenie sú aplikácie ďalej značené písmenami A (Discord), B (Ebay), C (Flashscore), D (Ideme Vlakom), E (LinkedIn), F (Netflix), G (Reddit), H (Twitch), I (Twitter), J (Windy) a X (nezmána aplikácia).

Experiment 1. V prvom experimente som vyskúšal použitie len kombinácie JA3, JA3S a SNI, s výsledkom **209** správne pozitívnych, **0** nesprávne pozitívnych, **232** správne negatívnych a **69** nesprávne negatívnych identifikácií. Presnosť (angl. accuracy) bola *86.47*%, precíznosť (angl. precision) *100*% a senzitivita (angl. recall) *75.18*% (viď Tabuľka 4.1). Obr. 4.1 zobrazuje príslušnú konfúznu maticu.

Experiment 2. V druhom experimente som vyskúšal použitie kombinácie JA3, JA3S a SNI a pri neúspechu kombináciu JA3 a JA3S, s výsledkom **231** správne pozitívnych, **40** nesprávne pozitívnych, **192** správne negatívnych a **47** nesprávne negatívnych identifikácií. Presnosť bola *82.94*%, precíznosť *85.24*% a senzitivita *83.09*% (viď Tabuľka 4.1). Obr. 4.2 zobrazuje príslušnú konfúznu maticu.

Experiment 3. V treťom experimente som vyskúšal použitie kombinácie JA3, JA3S a SNI, pri prvom neúspechu kombináciu JA3 a JA3S a na koniec iba otlačok JA3, s výsledkom **233** správne pozitívnych, **57** nesprávne pozitívnych, **175** správne negatívnych a **45** nesprávne negatívnych identifikácií. Presnosť bola *80*%, precíznosť *80.34*% a senzitivita *83.81*% (viď Tabuľka 4.1). Obr. 4.3 zobrazuje príslušnú konfúznu maticu.

Experiment 4. V štvrom experimente som vyskúšal použitie kombinácie JA3, JA3S a SNI a pri neúspechu kombináciu JA3S a SNI, s výsledkom **243** správne pozitívnych, **0** nesprávne pozitívnych, **232** správne negatívnych a **35** nesprávne negatívnych identifikácií. Presnosť bola *93.14%*, precíznosť *100%* a senzitivita *87.41%* (viď Tabuľka 4.1). Obr. 4.4 zobrazuje príslušnú konfúznu maticu.

V [2] najlepšie výsledky dosahovala metóda použitá v experimente 1. Mňa zaujímalo akých výsledkov dosiahnem ak postupnem skombinujem aj ostatné spôsoby identifikácie, teda akých výsledkov dosiahnem postupným pridaním JA3+JA3S a JA3. Z výsledných štatiských je možné konštatovať, že výsledky sa postupne zhoršovali, jedine hodnota senzivity narastala. Taktiež narastal počet nesprávne klasifikovaných otlačkov (nesprávne pozitívnych). V poslednom experimente som sa teda snažil maximalizovať počet správne klasifikovaných otlačkov. Keďže prvá metóda sa ukázala ako najlepšia, vyskúšal som pridanie JA3S+SNI. Podľa výsledných štastík sa táto metóda ukázala ako najlepšia s najväčším počtom správnych klasifikácií.

Tabuľka 4.1: Štatistiky

	Accuracy	**Precision**	**Recall**
Experiment 1	86.47%	100%	75.18%
Experiment 2	82.94%	85.24%	83.09%
Experiment 3	80%	80.34%	83.81%
Experiment 4	93.14%	100%	87.41%

Na experimenty som použil skript *test.py* a na ich vyhodnotenie *stat.py* (viď Dodatok A).

```
$ test.py --csv <csv> --db <db> {-1|-2|-3|-4}
$ stat.py predicted.csv real.txt
```

Obr. 4.1: JA3+JA3S+SNI

		Real										
		A	B	C	D	E	F	G	H	I	J	X
Predicted	A	25	0	0	0	0	0	0	0	0	0	0
	B	0	5	0	0	0	0	0	0	0	0	0
	C	0	0	29	0	0	0	0	0	0	0	0
	D	0	0	0	11	0	0	0	0	0	0	0
	E	0	0	0	0	3	0	0	0	0	0	0
	F	0	0	0	0	0	4	0	0	0	0	0
	G	0	0	0	0	0	0	35	0	0	0	0
	H	0	0	0	0	0	0	0	27	0	0	0
	I	0	0	0	0	0	0	0	0	22	0	0
	J	0	0	0	0	0	0	0	0	0	48	0
	X	0	13	0	0	9	22	3	9	12	1	232

Obr. 4.2: (JA3+JA3S+SNI)+(JA3+JA3S)

		Real										
		A	B	C	D	E	F	G	H	I	J	X
Predicted	A	25	0	0	0	0	0	0	0	0	0	38
	B	0	5	0	0	0	0	0	0	0	0	1
	C	0	0	29	0	0	0	0	0	0	0	0
	D	0	0	0	11	0	0	0	0	0	0	0
	E	0	0	0	0	4	0	0	0	0	0	0
	F	0	6	0	0	0	18	0	0	0	0	0
	G	0	0	0	0	0	0	36	1	0	0	0
	H	0	0	0	0	0	0	0	31	0	0	0
	I	0	0	0	0	0	0	0	0	24	0	0
	J	0	0	0	0	0	0	0	0	0	48	1
	X	0	7	0	0	8	8	2	4	10	1	192

Obr. 4.3: (JA3+JA3S+SNI)+(JA3+JA3S)+JA3

		Real										
		A	B	C	D	E	F	G	H	I	J	X
Predicted	A	25	0	0	0	0	0	0	0	0	0	38
	B	0	5	0	0	0	0	0	0	0	0	1
	C	0	0	29	0	0	0	0	0	0	0	0
	D	0	0	0	11	2	0	0	0	0	0	3
	E	0	0	0	0	4	0	0	0	0	0	6
	F	0	8	0	0	0	18	0	0	0	0	0
	G	0	0	0	0	0	0	36	1	0	0	0
	H	0	0	0	0	0	0	0	31	0	0	2
	I	0	0	0	0	1	0	0	0	25	0	1
	J	0	0	0	0	0	0	0	0	0	49	6
	X	0	5	0	0	5	8	2	4	9	0	175

Obr. 4.4: (JA3+JA3S+SNI)+(JA3S+SNI)

		Real										
		A	B	C	D	E	F	G	H	I	J	X
Predicted	A	25	0	0	0	0	0	0	0	0	0	0
	B	0	17	0	0	0	0	0	0	0	0	0
	C	0	0	29	0	0	0	0	0	0	0	0
	D	0	0	0	11	0	0	0	0	0	0	0
	E	0	0	0	0	12	0	0	0	0	0	0
	F	0	0	0	0	0	15	0	0	0	0	0
	G	0	0	0	0	0	0	35	0	0	0	0
	H	0	0	0	0	0	0	0	27	0	0	0
	I	0	0	0	0	0	0	0	0	24	0	0
	J	0	0	0	0	0	0	0	0	0	48	0
	X	0	1	0	0	0	11	3	9	10	1	232

5 Záver

Identifikácia mobilných aplikácií pomocou otlačkov TLS použítim kombinácie JA3, JA3S a SNI je v celku presná pre niektoré aplikácie, ako napr. Discord, Flashscore, Ideme Vlakom, Reddit a Windy. Kombinovanie s JA3 a JA3S alebo len samotným JA3 jej presnosť výrazne zhoršuje. Avšak, v tomto projekte sa ju podarilo spresniť pridaním kombinácie JA3S a SNI, najmä pre aplikácie Ebay a LinkedIn.

Projekt bol vypracovaný na základe prezentácie, materiálov a nástrojov poskytnutých vedúcim projektu Ing. Petrom Matouškom, Ph.D., M.A..

Literatúra

[1] Dierks, T. *The Transport Layer Security (TLS) Protocol Version 1.2* [online]. Internet Engineering Task Force (IETF), august 2008. Dostupné z: https://tools.ietf.org/html/rfc5246.

[2] Matoušek, P., Burgetová, I., Ryšavý, O. a Victor, M. On Reliability of JA3 Hashes for Fingerprinting Mobile Applications. In: Goel, S., Gladyshev, P., Johnson, D., Pourzandi, M. a Majumdar, S., ed. *Digital Forensics and Cyber Crime*. Springer, Cham, 2021. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, sv. 351. Dostupné z: https://doi.org/10.1007/978-3-030-68734-2_1. ISBN 978-3-030-68734-2.

[3] Benjamin, D. *Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility* [online]. Internet Engineering Task Force (IETF), január 2020. Dostupné z: https://tools.ietf.org/html/rfc8701.

A Súbory k Projektu

Dodatok obsahujem zoznam súborov k projektu dostupných na Google Drive[1].

- apk/ – súbory .apk na inštaláciu aplikácií
- db/ – CSV súbory obsahujúce databázy otlačkov
- pcap/ – PCAP súbory
- predicted/ – experimenty a súbor s anotovanými dátami
- sni-filtered/ – súbory s unikátnymi SNI pre každú aplikáciu
- statistics/ – súbory so štatistikami experimentov
- tlss-csv/ – extrahované TLS informácií z PCAP súborov
- apps.txt – zoznam aplikácií, ich verzií a odkazmi na stiahnutie
- create_db.sh – bash skript na vytvorenie finálnej databázy
- extract.sh – bash skript na extrakciu TLS informácií z PCAP súborov
- get-sni-csv.py – python skript pre extrahovanie SNI z TLS paketov
- keywords.csv – zoznam klúčových slov
- stat.py – python skript na vypočítanie konfúznych matíc a ďaľších štatistík
- test.py – testovací python skript
- tlss2ja3.py – python skript pre tvorbu databázy TLS otlačkov

[1] https://drive.google.com/drive/folders/1-vyl9JvMuEdQHhx3OcQvJaO8ASNqpU44?usp=sharing