Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Optimalizácia prehľadávania subdomén

Bakalárska práca

2023
Samuel Revúcky

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Optimalizácia prehľadávania subdomén

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	doc. RNDr. Martin Stanek, PhD.

Bratislava, 2023
Samuel Revúcky

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Samuel Revúcky
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Optimalizácia prehľadávania subdomén
Optimizing subdomain brute-forcing

Anotácia: V práci budeme predpokladať, že doménové mená v DNS sú korelované, teda existencia konkrétnych mien môže zvýšiť pravdepodobnosť, že v doméne sú aj niektoré ďalšie mená. Cieľom práce je preskúmať tento predpoklad a navrhnúť vhodný model pre reprezentáciu korelácií doménových mien. Model bude naplnený pomocou voľne dostupných zdrojov. Následne práca otestuje tento model pri adaptívnom prehľadávaní subdomén a porovná výsledky so štandardnými slovníkovými technikami.

Vedúci: doc. RNDr. Martin Stanek, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Dátum zadania: 30.10.2022

Dátum schválenia: 31.10.2022 doc. RNDr. Dana Pardubská, CSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Chcel by som sa týmto poďakovať môjmu školiteľovi Martinovi Stanekovi za pomoc, rady a veľkú ochotu počas celého písania tejto práce.

Abstrakt

V tejto bakalárskej práci sa snažíme pomocou vzťahov medzi doménovými menami optimalizovať prehľadávanie subdomén slovníkovým útokom. Navrhujeme model na reprezentáciu týchto vzťahov, naplnený dátami z CT logov. Algoritmus, ktorý na prehľadávanie implementujeme, využíva model na adaptívne vyberanie mien zo slovníka. Experimenty s enumeráciou subdomén ukazujú, že náš algoritmus dokáže byť účinnejší ako slovníkový útok, ak má k dispozícii približne 500 a viac DNS dotazov.

Kľúčové slová: DNS, subdoména, slovníkový útok

Abstract

In this thesis we study the impact of using relations between domain names in the dictionary subdomain enumeration. We propose a model to represent these relations, filled with data from CT logs. The algorithm we implement for the enumeration uses this model to adaptively pick names from a provided dictionary. By experimentally testing our algorithm we discover that it can be beneficial to use it instead of the standard dictionary technique, when there are around 500 or more DNS queries available.

Keywords: DNS, subdomain, dictionary attack

Obsah

Úvod

Ľudstvo sa každým dňom stáva viac závislé na informačných systémoch, či už sa jedná o verejnú infraštruktúru, alebo osobné zariadenia, čo zároveň zvyšuje zraniteľnosť voči kybernetickým útokom. Tie môžu mať veľa podôb, ako sú napríklad vírusy, červy, trójske kone, phishingové útoky a denial-of-service útoky. Útoky môžu mať za následok stratu citlivých údajov, finančných prostriedkov, obchodných tajomstiev, poškodenie reputácie a pod. Preto zohráva kybernetická bezpečnosť kľúčovú úlohu v našich životoch.

Správanie útočníkov všeobecne nazývame TTP (taktiky, techniky a procedúry). Z pohľadu ochrany je užitočné poznanie a skúmanie TTP a to z viacerých dôvodov. Porozumenie útočným metódam umožňuje bezpečnostným pracovníkom identifikovať a analyzovať potenciálne hrozby, alebo vykonať potrebné opatrenia pri prebiehajúcich útokoch. Taktiež pomáha organizáciám vytvárať a implementovať účinnejšie mechanizmy na ochranu svojej infraštruktúry. V neposlednom rade môže viesť k odhaleniu zraniteľností vo vlastných systémoch a ich odstráneniu skôr, ako by mohli byť využité útočníkmi.

Skúmanie TTP viedlo k vytvoreniu verejne dostupných modelov dokumentujúcich TTP, ako napríklad **MITRE ATT&CK** [10]. Ten pozostáva z dvoch hlavných častí, enterprise (útoky na spoločnosti) a mobile (útoky na mobilné zariadenia). Obe časti obsahujú viacero schém venovaných konkrétnym softvérom, ako je napríklad Windows. V každej z nich sú zdokumentované taktiky a jednotlivé techniky útokov. Ďalším príkladom je projekt **Unified Kill Chain**, ktorý vznikol spojením informácií z viacerých modelov za cieľom poskytnúť jednotnú a kvalitnejšiu schému útokov [8].

Zatiaľ čo práca nie je zameraná na priame zvyšovanie kybernetickej bezpečnosti, jej téma s ňou úzko súvisí. V práci sa venujeme problematike enumerácie subdomén. Najskôr predstavme pojmy doména a subdoména. Väčšina serverov v sieti internet má meno, podľa ktorého si ich ľudia môžu zapamätať. Toto meno nazývame doména. Domény sa skladajú z viacerých slov (úrovní), oddelených bodkou. Niektoré z týchto úrovní sa niekedy tiež zvyknú jednotlivo nazývať domény, čo môže čitateľovi spôsobiť určité nejasnosti. Členenie domén opíšeme na nasledujúcom príklade. Majme doménu **one.example.com**. Najvyššou úrovňou (top-level domain) je v tomto prípade **com**.

Ďalšou úrovňou je hlavná doména (second level/root domain), ktorá sa často myslí pod pojmom doména, v tomto prípade **example**. Zvyčajne reprezentuje meno organizácie, ktorá si doménu zaregistruje. Na poslednej úrovni máme **one**, túto nazývame subdoména. Subdomény zvyčajne reprezentujú rôzne aplikácie, servery a systémy v infraštruktúre organizácie. Samotné subdomény môžu mať viacero úrovní. Technicky aj **example** je subdoménou domény **com**, ale v tejto práci budeme pod pojmom subdoména myslieť len tretiu (a nižšiu) úroveň domén.

Na priraďovanie IP adries ku doménovým menám slúži systém DNS. Vďaka tomu sa dá existencia subdomény jednoducho overiť otázkou na DNS server, či k nej existuje IP adresa.

Ako dokumentujú vyššie spomenuté modely, prvou fázou kybernetických útokov je prieskum cieľa (reconnaissance). Enumerácia (prehľadávanie) subdomén je jednou z techník, ktorá sa pri tomto prieskume zvyčajne vykonáva. Úlohou enumerácie subdomén je nájsť čo najväčšie množstvo subdomén cieľovej domény. To môže priniesť útočníkovi cenné informácie o infraštruktúre cieľa, ktoré môže neskôr využiť. Naopak pre bezpečnostných pracovníkov je enumerácia užitočnou testovacou metódou, pretože môže poukázať napríklad na zabudnuté aplikácie, ktoré nespĺňajú bezpečnostné požiadavky a ktoré by inak boli potenciálnym slabým miestom infraštruktúry organizácie.

Kapitola 1

Enumerácia subdomén

Najprv vysvetlíme vzťah medzi pojmami doménové meno a subdoména. Doménovým menom myslíme reťazec, ktorý reprezentuje doménu, vrátane predpony reprezentujúcej subdoménu, ak nejakú má. Pojmom subdoména myslíme buď celú doménu, alebo len samotnú predponu, v závislosti od kontextu.

Existujú rôzne techniky na enumeráciu subdomén. Rozdeľujeme ich na dve kategórie: pasívne a aktívne.

Pasívne techniky získavajú subdomény bez priameho kontaktu s infraštruktúrou cieľovej domény. Využívané sú údaje z verejných zdrojov. Tieto techniky na rozdiel od aktívnych nevytvárajú žiadnu aktivitu, ktorú by obranné mechanizmy cieľa mohli vyhodnotiť ako podozrivú. Príklady takýchto zdrojov:

- **Certificate Transparency** [3, 5] – Internetový bezpečnostný štandard na monitorovanie digitálnych certifikátov. Vytvára systém verejných logov, v ktorých sa zaznamenávajú všetky certifikáty vydané dôveryhodnými verejnými certifikačnými autoritami. Existuje viacero databáz, napríklad `crt.sh`, ktoré monitorujú tieto logy a pomocou vyhľadávania v týchto databázach sa dajú z certifikátov získať doménové mená.

- **Webové vyhľadávače** – Vyhľadávanie pomocou webových vyhľadávačov sa dá využiť na získavanie doménových mien. Vyhľadávače reagujú na špeciálne príkazy vo vyhľadávanom reťazci. Príkazom `site:example.com` vyhľadávaču prikážeme, aby vyhľadával iba stránky, ktorých doménové meno obsahuje `example.com`.

- **Pasívne DNS** [4] – Systém zavedený na zvýšenie obranyschopnosti proti škodlivým stránkam. Je to archív, do ktorého DNS servery ukladajú spojenia typu *doménové meno – IP adresa* zo všetkých DNS dotazov, ktoré spracujú. Týmto spôsobom ostávajú zaznamenané všetky IP adresy, ku ktorým bolo kedy nejaké doménové meno naviazané. Pomocou prechádzania týchto archívov vieme získať informácie o existencii subdomén.

Aktívne techniky získavajú subdomény aktívnym posielanín dotazov priamo do infraštruktúry cieľa, alebo verejným DNS resolverom. Niekoľko príkladov:

- **Použitie hrubej sily** (Brute–forcing) – Postupne sa generujú všetky možné reťazce z danej množiny určenej napr. maximálnou dĺžkou a znakovou sadou. Každý z nich sa pripojí k doméne ako subdoména a existencia vzniknutého doménového mena sa overí v DNS. Tento proces má však exponenciálnu časovú zložitosť vzhľadom na dĺžku hľadaných mien.
- **Slovníkový útok** – Podobne ako v predošlom príklade, subdomény sa získavajú dotazmi do DNS, ale v tomto prípade sa k doméne nepripájajú všetky možné reťazce, ale len mená z nejakého zoznamu – slovníka. Takto sa nemusí podariť získať všetky subdomény cieľovej domény, ale zníži sa časová náročnosť a vznikne možnosť výberu mien, ktoré bude prehľadávanie skúšať.
- **Enumerácia PTR záznamov** – Systém DNS ponúka možnosť reverzných DNS dotazov, ktoré na základe informácií uložených v PTR záznamoch vrátia ku zadanej IP adrese prislúchajúce doménové meno. Pri tomto type enumerácie vezmeme rozsah IP adries pridelený organizácii, ktorej subdomény chceme enumerovať a robíme pre každú IP adresu reverzný DNS dotaz. Nevýhodou tejto techniky je, že nie každá doména má PTR záznam.

Existuje mnoho nástrojov na enumeráciu subdomén, ktoré implementujú vyššie spomenuté metódy. Príklady takýchto nástrojov:

- Sublist3r [1] – zameraný na enumeráciu pomocou pasívnych techník
- SubBrute [11] – zameraný na enumeráciu pomocou slovníkovej techniky
- Amass [6] – kombinuje viacero prístupov

Cieľom práce je optimalizovať enumeráciu slovníkovým útokom. Predpokladáme, že doménové mená vzájomne korelujú, čiže existencia, alebo absencia konkrétnej domény ovplyvňuje pravdepodobnosť existencie ďalších. Pôvod tejto korelácie nie je záujmom tejto práce, ale pre lepšie pochopenie čitateľa, môže pochádzať napríklad zo zvykov a konvencií medzi správcami webových serverov pri pomenovaní služieb, alebo môže pomenovanie byť implikované účelom služby a pod. Chceme namodelovať tieto vzťahy a navrhnúť algoritmus na enumeráciu, ktorý ich bude využívať. Ak sa predpoklad ukáže byť pravdivý, očakávame, že náš algoritmus bude schopný nájsť pri rovnakom počte DNS dotazov viac subdomén ako obyčajný slovníkový útok. Pokiaľ je nám známe, takýto postup enumerácie nevyužíva zatiaľ žiadny verejne dostupný nástroj.

Kapitola 2

Model vzťahov doménových mien

Vzťahy doménových mien modelujeme na základe informácií z SSL/TLS certifikátov získaných z CT logov, preto najprv podrobnejšie popíšeme systém Certificate Transparency (CT).

2.1 SSL/TLS certifikáty

Secure Sockets Layer (SSL) a jeho následník Transport Layer Security (TLS) sú protokoly, ktoré zabezpečujú šifrovanú komunikáciu medzi dvomi zariadeniami. Digitálne certifikáty slúžia na distribúciu verejných kľúčov serverov, čo umožňuje overenie autenticity serverov. Certifikáty sú vydávané dôveryhodnými certifikačnými autoritami (CA), ktoré vydaním ručia za to, že verejný kľúč v certifikáte skutočne patrí subjektu, na ktorý je certifikát vydaný.

Certifikáty zvyčajne obsahujú nasledujúce informácie [2]:

- Doménove meno subjektu (Common Name – CN)
- Názov organizácie subjektu
- Názov vydávajúcej CA
- Alternatívne mená subjektu (Subject Alternative Name – SAN)
- Dátum vydania
- Dátum expirácie
- Verejný kľúč
- Digitálny podpis vydávajúcej CA

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            03:9a:66:cf:74:1e:81:be:da:22:0e:25:24:0e:c2:de:ce:15
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: (CA ID: 183267)
            commonName                = R3
            organizationName          = Let's Encrypt
            countryName               = US
        Validity (Expired)
            Not Before: Nov 17 16:02:09 2022 GMT
            Not After : Feb 15 16:02:08 2023 GMT
        Subject:
            commonName                = itzroks-060000vub3-h080i9.us-east.containers.appdomain.cloud
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:c6:ba:94:7f:ff:39:a2:b0:60:b6:52:30:8c:bc:
                    2b:98:10:32:f2:46:5f:17:ab:03:de:34:0e:cd:b8:
                    d0:ed:39:71:37:c3:cf:e1:3b:51:09:ec:34:b0:35:
                    8c:c4:df:ad:ea:a9:62:bb:df:5d:2f:14:78:e1:e6:
                    b2:ad:cf:65:25:05:da:dd:a7:1a:4e:21:73:5e:be:
                    89:e6:49:5b:4f:9b:68:08:2f:29:fe:b1:cc:a3:ac:
                    5e:51:6c:d2:d5:66:1b:d7:c4:a5:3d:af:16:43:83:
                    c4:6e:76:f2:db:bb:1e:c6:dd:80:4e:b0:95:cc:14:
                    54:fc:b4:19:78:23:26:85:0e:33:35:5a:1b:b3:dc:
                    e8:76:15:c9:cb:cb:f4:1f:15:9a:b9:c2:7c:15:90:
                    f9:29:3e:e1:62:08:62:0f:47:ee:d7:0a:f0:41:b2:
                    e0:9e:71:2f:d2:0e:1f:67:72:2e:53:83:c4:60:98:
                    4f:56:81:da:e0:1e:a4:71:82:a5:84:f5:cd:87:a1:
                    8a:f8:c4:88:32:65:a0:63:5b:7f:7a:88:76:12:77:
                    e1:0b:d9:54:d8:29:db:5e:b7:55:f8:0b:84:6f:45:
                    fe:53:1b:99:6c:52:60:45:e6:02:3d:2f:ef:78:5d:
                    87:79:65:fb:f5:e5:c2:37:6a:88:b7:48:54:03:d4:
                    f5:1b
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Subject Key Identifier:
                BB:39:38:1F:7E:8F:AC:3B:F6:F9:89:A0:73:2B:09:4B:E1:65:70:1C
            X509v3 Authority Key Identifier:
                keyid:14:2E:B3:17:B7:58:56:CB:AE:50:09:40:E6:1F:AF:9D:8B:14:C2:C6

            Authority Information Access:
                OCSP - URI:http://r3.o.lencr.org
                CA Issuers - URI:http://r3.i.lencr.org/

            X509v3 Subject Alternative Name:
                DNS:*.itzroks-060000vub3-h080i9-4b4a324f027aea19c5cbc0c3275c4656-0000.us-east.containers.appdomain.cloud
                DNS:itzroks-060000vub3-h080i9-4b4a324f027aea19c5cbc0c3275c4656-0000.us-east.containers.appdomain.cloud
                DNS:itzroks-060000vub3-h080i9.us-east.containers.appdomain.cloud
            X509v3 Certificate Policies:
                Policy: 2.23.140.1.2.1
                Policy: 1.3.6.1.4.1.44947.1.1.1
                  CPS: http://cps.letsencrypt.org
```

Obr. 2.1: Ukážka certifikátu

2.2 Certificate Transparency

CT je technológia vyvinutá na zvýšenie transparentnosti a zodpovednosti pri vydávaní a spravovaní SSL/TLS certifikátov. Jej cieľom je zabrániť vydávaniu podvodných alebo závadných certifikátov, ktoré môžu byť použité napríklad na man-in-the-middle útoky, odchytávanie a modifikáciu bezpečných komunikácií a kradnutie citlivých informácií. CT to dosahuje poskytovaním nezmeniteľného a verejne dostupného záznamu certifikátov prostredníctvom systému verejných logov, v ktorých sú zaznamenávané všetky certifikáty vydané dôveryhodnými certifikačnými autoritami (CA).

CT vyžaduje, aby CA verejne zaznamenávali každý vydaný certifikát. CA tak vytvárajú chronologický zoznam všetkých certifikátov. Tento zoznam môže byť overený kýmkoľvek. Prevádzkovatelia webových lokalít, prehliadače, alebo iné subjekty môžu monitorovaním týchto logov odhaliť falošné certifikáty a odmietnuť komunikáciu so servermi prezentujúcimi sa takýmito certifikátmi.

CA prevádzkujú svoje vlastné logy na zaznamenávanie vydávaných certifikátov. CA sú zodpovedné za udržiavanie svojich logov. Okrem individuálnych logov CA existujú aj kolektívne logy, udržiavané tretími stranami, ktoré agregujú informácie z viacerých logov. Kolektívne logy poskytujú komplexnejší pohľad na systém certifikátov a umožňujú sledovať certifikáty vydané viacerými certifikačnými autoritami. Jeden certifikát sa zvyčajne vyskytuje vo viacerých kolektívnych CT logoch.

Príklady spoločností, ktoré vedú kolektívne CT logy [9], aj s menami logov:

- Google (Xenon, Argon)
- DigiCert (Nessie, Yeti)
- CloudFlare (Nimbus)
- Sectigo (Mammoth)

2.3 Zber dát

`Crt.sh` je webová stránka poskytujúca databázu certifikátov z najrelevantnejších CT logov. Bola vytvorená za účelom zjednodušenia monitorovania a analýzy certifikátov. Na `crt.sh` je možné vyhľadávať doménové mená, IP adresy a tiež informácie o certifikátoch.

Kvôli týmto vlastnostiam sme ako zdroj dát pre náš model zvolili práve túto stránku. Dáta sú verejne dostupné a je ich dostatočné množstvo. Stránka poskytuje dva spôsoby, ako z jej databázy získavať informácie. Prvým sú API requesty a druhým je priamy prístup do PostgreSQL databázy. Vybrali sme druhú možnosť, pretože funkcionalita PostreSQL nám už bola známa z predošlého štúdia. Pre túto prácu z celej databázy postačovala tabuľka ***certificate***, obsahujúca stĺpce *id*, *issuer_ca_id* a *certificate*. V tabuľke sú certifikáty usporiadané podľa stĺpca *id*, čo zhruba korešponduje s poradím podľa dátumu platnosti certifikátov.

Do SAN môžu patriť doménové mená, IP adresy, emailové adresy a ďalšie [2]. Umožňuje, aby jedným certifikátom mohla byť zabezpečená dôveryhodnosť viacerých domén. Vďaka tejto vlastnosti môžu prevádzkovatelia serverov do jedného certifikátu spolu s hlavnou doménou zahrnúť aj viacero jej subdomén. Do časti CN sa potom zvyčajne

udáva meno hlavnej domény. Intuitívne sa dá predpokladať, že subdomény jednej hlavnej domény majú medzi sebou vzťah. S cieľom využiť tento predpoklad pre náš model sme z certifikátov extrahovali CN a SAN.

Pôvodným cieľom bolo získať dáta zo všetkých certifikátov vydaných v roku 2022. V čase zberu bolo už vydaných takých certifikátov viac ako osem miliárd. Toto množstvo značne prevyšovalo dotazovacie kapacity databázy. Preto sme sa rozhodli pre náhodný výber zhruba desiatich miliónov certifikátov vydaných v roku 2022. Našli sme identifikátor prvého certifikátu z roku 2022 a identifikátor posledného, ktorý bol v tom čase v tabuľke, keďže zber dát sme vykonali ešte v 2022. Z tohoto intervalu identifikátorov sme pomocou knižničnej pythonovskej funkcie `random.randrange()` náhodne vyberali certifikáty. Pre optimalizáciu rýchlosti sme z databázy žiadali certifikáty po skupinách o veľkosti 2700 certifikátov, nasledujúcich v tabuľke po sebe. Prvý certifikát v skupine bol vždy ten, ktorý vybrala funkcia `randrange()`. Prekryvom týchto skupín sme zabránili tým, že sme funkcii dali ako parameter `step` veľkosť skupiny. Dáta sme uložili do textového súboru tak, že pre každý certifikát sa do nového riadku vypísalo CN a následne všetky mená zo SAN, čiže výsledný súbor mal toľko riadkov, koľko certifikátov bolo spracovaných.

Čistenie datasetu

Po analýze získaného datasetu sa ukázalo, že obsahoval veľké množstvo pre nás nezaujímavých dát, ako napríklad:

- Certifikáty vydané firmám poskytujúcim webhosting, ktorých SAN obsahujú doménové mená vzájomne rôznych hlavných domén.
- Domény vytvorené poskytovateľmi cloudových služieb, alebo generátormi webových stránok, ktoré pozostávajú z náhodne vygenerovaných reťazcov.
- Certifikáty obsahujúce **wildcards**. Také domény majú tvar `*.example.com`. Certifikát vydaný na takúto doménu by zabezpečoval dôveryhodnosť pre doménu example.com a neohraničené množstvo jej prvoúrovňových subdomén.

Pri prvej predstave modelu sme chceli, aby algoritmus na hľadanie subdomén mal možnosť využívať všetky mená v datasete. Na takýto prístup sme potrebovali dataset očistiť od všetkých nezaujímavých mien. Počas pokusov o vyčistenie sa ale dataset zmenšil na menej ako 10% svojej pôvodnej veľkosti, pričom ešte stále obsahoval mnoho nežiadúcich mien a čistiaci algoritmus bol už príliš orientovaný na tento konkrétny dataset. Preto sme zmenili prístup k celému modelu, opíšeme ho v časti 2.4. Algoritmus na čistenie sme vďaka tomu mohli zjednodušiť a zovšeobecniť tak, aby sa prípadne

Algoritmus 1 Čistiaci algoritmus

Vstup: dataset.txt

Výstup: dataset_vyčistený.txt

```
for all riadok in dataset do
    CN = riadok[0]
    SAN = riadok[1:]
    lcs = longest_common_suffix(SAN)
    if CN už spracované or lcs nemá tvar doména.top_level_doména then
        continue
    mená = {}
    for all doména in SAN do
        odrež lcs od doména
        if doména in mená or doména = '' then
            continue
        mená.add(doména)
    if mená.size() ≥ 2 then
        zapíš mená do nového riadku do výsledného súboru
```

neskôr mohol rovnako dobre použiť na iný dataset. Pseudokód čistiaceho algoritmu je uvedený v algoritme 1.

Z datasetu o veľkosti 9 681 964 certifikátov sme takto dospeli k veľkosti 922 493 certifikátov, s priemerným počtom približne 4,5 mien v SAN a štandardnou odchýlkou 4,74.

2.4 Model

Model sme sa rozhodli naviazať na nejaký konkrétny slovník, s ktorým by sa algoritmus na enumeráciu v experimentálnej časti porovnával. Namiesto uvažovania celého datasetu sme uvažovali len mená, ktoré sa zároveň vyskytujú v danom slovníku. Tento prienik sa vykonal až po vyčistení datasetu. Týmto sa odstránila akákoľvek diskriminácia jednotlivých prístupov k enumerácii vzhľadom na potenciálne rôznu sadu mien pri experimentoch. Je vhodné pripomenúť, že dataset tak, ako sme ho opísali v predošlej podkapitole, nebol očistený od úplne všetkých nežiaducich mien. Obsahoval mená ako napríklad: `*.2dd8edce6aafaba5d61908ce.mcizs, *, *.mcizs, *.mcizs.mesh`. To ale nebol problém, keďže tie sa po prieniku datasetu so slovníkom nepoužili.

Slovník sme vyberali z tých, ktoré sú používané nástrojmi na enumeráciu Amass a Subbrute. Zvolili sme slovník z repozitára Amassu subdomains-top1mil-110000.txt (v

tabuľke ako amass_2) [7]. Obsahuje 8,1% dvojúrovňových a 4,4% viacúrovňových mien. Napriek nášmu cieľu zameriavať sa hlavne na jednoúrovňové subdomény, rozhodli sme sa nemeniť slovník, pretože dvojúrovňové mená, ktoré obsahuje, majú potenciál byť v DNS a zároveň množstvo viacúrovňových je akceptovateľné. Relatívny prienik slovníku s datasetom je iba 17,9%. Zjavne ale je už frekvenčne utriedený a zo skúmaných slovníkov má najväčší absolútny prienik s datasetom.

slovník	**počet mien**	**veľkosť prieniku s datasetom**
names.txt (Subbrute)	101010	13884
amass_1	100000	18495
amass_2	114606	20516

Tabuľka 2.1: Porovnanie slovníkov

Za reprezentáciu modelu sme zvolili maticu $A_{n\times n}$, kde n je počet mien v slovníku. Prvok $a_{i,j}$ kde $i, j \in \{0, \ldots, n-1\}$ je pravdepodobnosť, že mená i a j sú spolu subdoménami jednej domény, počítaná ako relatívna početnosť spoločných výskytov v datasete. Nech $c_{i,j}$ je počet spoločných výskytov mien i a j v datasete, potom

$$a_{i,j} = \frac{c_{i,j}}{\sum\limits_{0\leq x<n} c_{i,x}}.$$

Platí $\sum\limits_{0\leq j<n} a_{i,j} = 1$ a $a_{i,i} = 0$ pre všetky $i \in \{0, ..., n-1\}$. Situácia, že by sa nejaké meno nikde nevyskytovalo spoločne s nejakým ďalším nenastane, lebo pri čistení sme také certifikáty odstránili.

Kvôli riedkosti a pamäťovej náročnosti matice sme ju neimplementovali ako dvojrozmerné pole, ale pomocou asociatívneho poľa, kde kľúče sú mená zo slovníka a hodnoty sú opäť asociatívne polia, ktorých kľúče sú mená zo slovníka a hodnoty sú prvky matice A prislúchajúce danej dvojici kľúčov (mien). Model je implementovaný v Pythone, kde ako asociatívne pole používame dátovú štruktúru dictionary. Ukážka modelu v Pythone:

```
A = { 'www': { 'mail': a_{www,mail}, 'ftp': a_{www,ftp}, ... },
      'mail': { 'www': a_{mail,www}, 'ftp': a_{mail,ftp}, ... },
      .
      .
      .
    }
```

Kapitola 3

Algoritmus na enumeráciu subdomén

Cieľom algoritmu je nájsť čo najviac subdomén s použitím čo najmenšieho počtu DNS dotazov. Pri hľadaní využíva mená z poskytnutého slovníka a vzťahy medzi týmito menami z modelu. Pseudokód konečnej verzie algoritmu na enumeráciu je uvedený v algoritme 2.

Princíp algoritmu je, že vyberá mená zo slovníka a následne ich overuje v DNS. Na rozdiel od obyčajného slovníkového útoku, mená nevyberá v poradí, v akom sú v slovníku, ale uprednostňuje jednotlivé mená na základe vzťahov vyplývajúcich z modelu. Na to mu slúži vektor p, ktorý každému menu v slovníku priraďuje pravdepodobnosť, že toto meno bude subdoména. Algoritmus pre každý nasledujúci DNS dotaz vyberá meno s najvyššou aktuálnou pravdepodobnosťou. Aby využil model čo najlepšie, algoritmus upravuje hodnoty vo vektore p po každom DNS dotaze v závislosti od toho, či meno z dotazu bolo subdoménou. V prípade ak bolo, zvýhodní tie mená, ktoré majú podľa modelu šancu byť tiež subdoménami. Preto k hodnotám v p prislúchajúcim týmto menám algoritmus pripočíta pravdepodobnosti z modelu. Naopak, ak meno z dotazu nie je subdoménou, príslušné mená z modelu znevýhodní. Algoritmus vtedy zvýši hodnoty v p všetkým okrem týchto mien.

Algoritmus používa aj dva vstupné parametre s_1 a s_2 reprezentujúce mieru závislosti na dátach z modelu. Predtým než pripočíta pravdepodobnosti z modelu k p, vynásobí ich hodnotou s_1 v prípade, že meno bolo subdoménou a s_2 v opačnom prípade.

3.1 Vývoj algoritmu

Konečná verzia je veľmi podobná počiatočnému návrhu algoritmu. Počas vývoja sme ale vykonali, resp. testovali nasledujúce úpravy, resp. varianty:

- Prvá verzia algoritmu mala iba jeden parameter miery závislosti na dátach z modelu spoločný pre obe vetvy podmienky, teda $s_1 = s_2 = s$. V tabuľke 3.1 je porovnanie tejto verzie so slovníkovým útokom. Keďže výsledky tejto verzie neboli

Algoritmus 2 Na enumeráciu subdomén

Vstup: zoznam mien d, názov cieľovej domény *domain*, počet DNS dotazov q, vektor pravdepodobností p, model A, parametre s_1, s_2, platí $|d| = |p|$

Výstup: počet nájdených subdomén *result*

```
result ← 0
mask ← [0, 0, ...], |mask| = |p|
n ← 0
loop
    i ← argmax(p)
    if mask[i] = 1 then
        continue
    mask[i] ← 1
    n ← n + 1
    w ← d[i]
    if 'w.domain' je v DNS then
        p ← p + s_1 * a_{w,.}        ▷ a_{w,.} je w-ty riadok matice A reprezentujúcej model
        result ← result + 1
    else
        n' ← počet neskúšaných mien
        b ← [0, 0, ...], |b| = |p|
        for j = 1, ..., |p| do
            if a_{i,j} = 0 then
                b_j ← 1/n'
        p ← p + s_2 * b
    p ← p / Σ p
    if n = q then return result
```

uspokojivé, skúsili sme pridať druhý parameter miery závislosti. To nám umožnilo používať v algoritme rôznu mieru závislosti podľa výsledku DNS dotazu. Po otestovaní niekoľkých kombinácií hodnôt parametrov s_1 a s_2 na 6 doménach s 1000 DNS dotazmi sa ukázal potenciál na uspokojivé výsledky.

- Algoritmus sme testovali na dvoch verziách slovníka. Prvou z nich bol vyššie spomenutý subdomains-top1mil-110000.txt a druhou bola jeho verzia frekvenčne utriedená podľa nášho modelu. Samotná sada mien, ktorú algoritmus dostal bola rovnaká v oboch prípadoch, ale hodnoty v iniciálnom vektore p sa líšili. V prvom prípade bola na začiatku každá hodnota v p rovná $1/n$, kde n bol počet mien v slovníku a v druhom bol p naplnený pravdepodobnosťami počítanými z rela-

doména: harvard.edu, počet dotazov: 1000

algoritmus	**citlivosť s**	**počet nájdených subdomén**
slovníkový útok	-	94
náš algoritmus s $s_1 = s_2 = s$	0,50	76
	0,75	83
	1,00	74
	1,50	93
	1,75	84
	2,00	95
	2,50	100

Tabuľka 3.1: Porovnanie slovníkového útoku s našim algoritmom využívajúcim len jeden parameter citlivosti

tívnej početnosti mien v modeli. V tabuľke 3.2 je porovnanie druhého prípadu so slovníkovým útokom. Algoritmus sme spustili pre rôzne s_1 a s_2 na šiestich doménach po 1000 DNS dotazov. Keďže výsledky boli neuspokojivé, v ďalších experimentoch sme už slovník utriedený podľa nášho datasetu nepoužívali.

$\mathbf{s_1}$	$\mathbf{s_2}$	**rozdiel v %**
0,50	3,00	−15,98
0,75	4,50	−14,77
1,00	6,00	−12,09
1,25	7,50	−9,75
1,50	9,00	−10,68

Tabuľka 3.2: Priemerné výsledky algoritmu s iniciálnym vektorom p naplneným dátami z modelu oproti slovníkovému útoku

- V snahe o zrýchlenie algoritmu sme skúšali metódu batchovania. Ideou bolo neaktualizovať vektor p pri každej iterácii, ale len pri každej tretej. Táto metóda zmenšila časovú náročnosť algoritmu, ale zhoršila výsledky. V tabuľke 3.3 je porovnanie so slovníkovým útokom. Na získanie týchto dát sme spustili algoritmus na šiestich doménach s 1000 DNS dotazmi. Keďže algoritmus aj tak väčšinu času čakal na odpovede z DNS, nepovažovali sme ušetrený čas hodný horších výsledkov a tento variant sme zavrhli.

s_1	s_2	rozdiel v %
0,50	3,00	−3,74
0,75	4,50	−6,28
1,00	6,00	−8,75
1,25	7,50	−5,82
1,50	9,00	−1,14

Tabuľka 3.3: Priemerné výsledky algoritmu s batchovaním oproti slovníkovému útoku

3.2 Optimalizácia parametrov

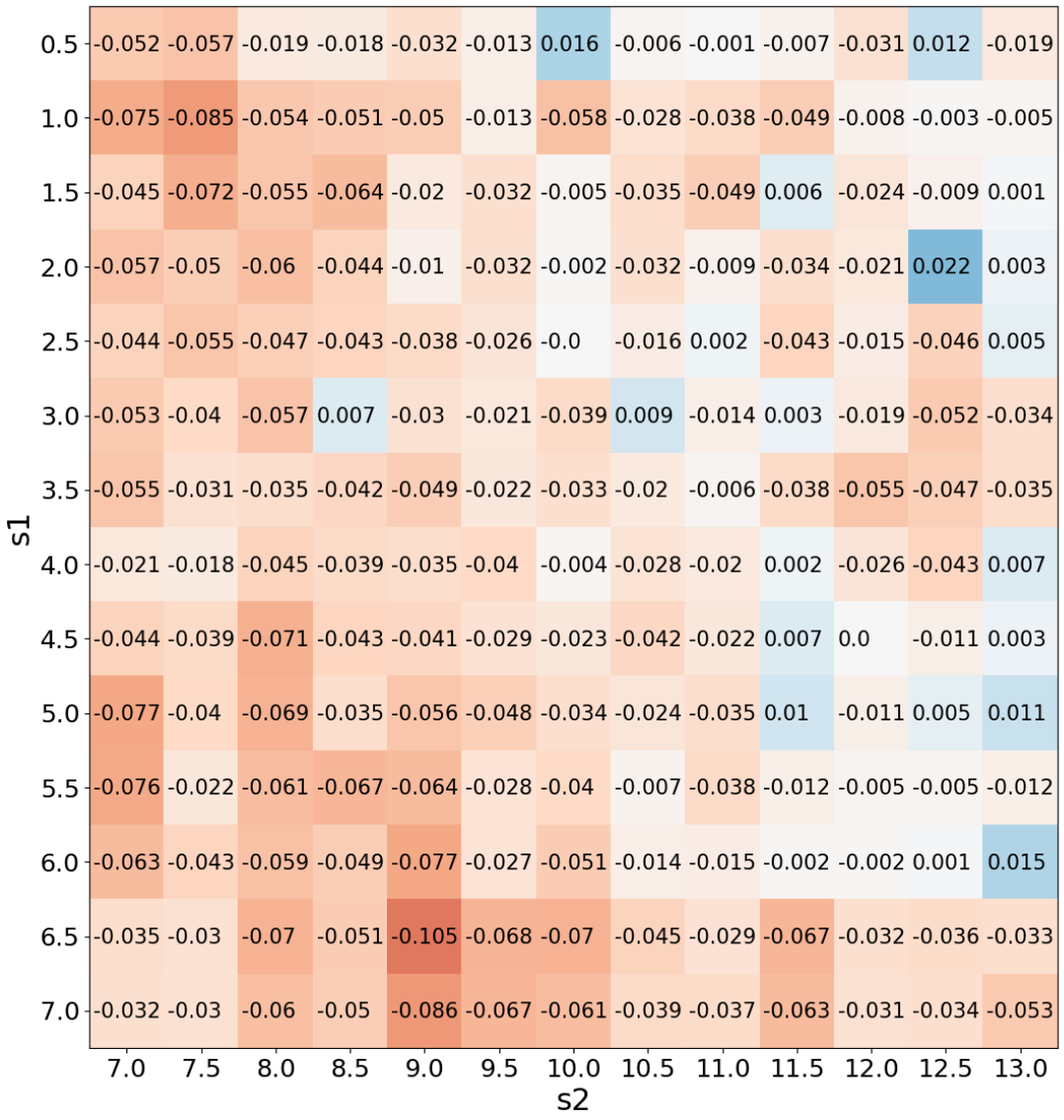
Parametre s_1 a s_2 sme optimalizovali tak, že sme spustili algoritmus na enumeráciu za týchto podmienok:

- Vybrali sme 50 domén, na ktorých sme algoritmus spúšťali.
- Znížili sme počet DNS dotazov na 200, aby sme kompenzovali vysoké množstvo vykonávaných behov algoritmu.
- Algoritmus sa vykonal pre každú usporiadanú dvojicu hodnôt (s_1, s_2), kde $s_1 \in \{0{,}5; 1{,}0; \ldots; 7{,}0\}$ a $s_2 \in \{7{,}0; 7{,}5; \ldots; 13{,}0\}$.

Nech V je množina všetkých usporiadaných dvojíc (s_1, s_2) a D je množina všetkých domén, s ktorými algoritmus pracoval. Nech $v \in V$ a $d \in D$, potom $s_{v,d}$ je výsledok slovníkového útoku pre dvojicu v a doménu d a $a_{v,d}$ je výsledok nášho algoritmu. Na vyhodnotenie dát sme použili vzorec

$$r_v = \frac{\sum_{d \in D} (a_{v,d} - s_{v,d})/s_{v,d}}{|D|}.$$

Premenná r_v vyjadruje priemer relatívneho rozdielu počtu nájdených subdomén medzi slovníkovým útokom a našim algoritmom naprieč všetkými testovanými doménami. Ak je výsledok kladný, náš algoritmus s danými hodnotami (s_1, s_2) prekonal slovníkový útok a analogicky v prípade záporného výsledku bol náš algoritmus horší. Výsledky prikladáme v podobe matice na obrázku 3.1.
Na použitie v ďalších experimentoch sme vybrali hodnoty $s_1 = 2{,}0$ a $s_2 = 12{,}5$, pretože dosiahli najlepší výsledok. Výsledky napovedajú o tom, že každá zmena hodnoty jedného z parametrov má nepredvídateľný dopad na počet nájdených subdomén. Pravdepodobne je to spôsobené tým, že aj jedna zmena hodnôt vo vektore pravdepodobností p dokáže značne ovplyvniť množinu mien, ktoré algoritmus vyskúša. Taktiež by možno



Obr. 3.1: Výsledky optimalizácie parametrov

výsledky mali menej náhodný charakter, ak by sa vykonala optimalizácia na vyššom počte domén a s viacej DNS dotazmi.

3.3 Implementácia

Algoritmus sme implementovali v jazyku Python. Prikladáme ho v časti Príloha B. Použili sme knižnice `numpy` na prácu s vektormi, `dns.resolver` na prácu s DNS a `openpyxl` na ukladanie výsledkov do excelovského zošita. Ďalej sme použili `sys`, `os`, `time` a `argparse` na zlepšenie používateľského rozhrania a výpis dát. Aby sme predišli zahlteniu predvoleného DNS resolvera dotazmi, použili sme päť resolverov, ktoré sme obmieňali každých desať dotazov. Konkrétne to boli resolvery `8.8.8.8`, `8.8.4.4` (Google), `1.1.1.1`, `1.0.0.1` (Cloudflare), `9.9.9.9` (Quad9). Dotazy do DNS sme posielali

sekvenčne. V skutočnosti je súčasťou programu na enumeráciu aj vytvorenie modelu z datasetu, pretože sme považovali za jednoduchšie vytvoriť model pri každom behu programu, ako ho ukladať do súboru. Pre každú doménu program najprv vykoná enumeráciu slovníkom a následne našim algoritmom. Keďže veľa mien sa opakuje v oboch prípadoch, použili sme asociatívne pole na ukladanie výsledkov dotazov. Tým sme zamedzili opakovaným DNS dotazom na rovnaké subdomény, čím sme skrátili trvanie nášho algoritmu bez ovplyvnenia výsledkov. V tabuľke 3.4 uvádzame približné výkonové parametre programu na enumeráciu, pri 1000 dotazoch na doménu. Pripomíname, že veľkosť datasetu je 922 493 riadkov, s priemerným počtom 4,5 mien v riadku a veľkosť slovníka je 114 606 mien. Dĺžka behu algoritmu sa môže líšiť v závislosti od dĺžky čakania na DNS, čo závisí od vyťaženia serverov a kvality internetového pripojenia. Je nutné poznamenať, že optimalizovanie výkonu algoritmu nebolo cieľom tejto práce. Všetky procesy sa vykonávali na zariadení vybavenom procesorom Intel Core i7-6500U s taktovaciou frekvenciou 2,5 GHz a pamäťou veľkosti 8GB s rýchlosťou 2133 MHz.

	sekundy
vygenerovanie modelu	8
slovníkový útok	70
náš algoritmus	40

Tabuľka 3.4: Približné výkonové parametre programu na enumeráciu

Kapitola 4

Experimenty

Cieľom experimentov v tejto práci je porovnať rôzne metódy enumerácie a vyhodnotiť využiteľnosť korelácie domén na efektívnejšiu enumeráciu subdomén. Porovnávame dva prístupy. Prvým je slovníkový útok opísaný v kapitole 1, ktorý pri enumerovaní využíva slovník z časti 2.4. Druhým je algoritmus z kapitoly 3 s nasledujúcimi parametrami: má rovnakú sadu mien, ako slovníkový útok, na vstupe dostane iniciálny vektor pravdepodobností, v ktorom sú hodnoty $1/n$, kde n je počet mien v slovníku, parametre s_1 a s_2 majú hodnoty 2,0 a 12,5, ktoré sa ukázali byť optimálne spomedzi testovaných hodnôt.

Pre každú enumerovanú doménu vykoná 1000 DNS dotazov, čo platí aj pre slovníkový útok. Oba sa spúšťajú na 150 doménach. Medzi tieto domény sme zahrnuli hlavne univerzity (britské a americké), IT spoločnosti ako sú napríklad Google a Facebook a tiež výrobcov informačných technológií ako napríklad Apple a Samsung. Ďalej sme vybrali farmaceutické a finančné spoločnosti, firmy z automobilového priemyslu a výrobcov pohonných hmôt, módneho priemyslu, stravovacie reťazce, sociálne siete, noviny. Zvyšné sme doplnili z rebríčkov populárnych domén. Celý zoznam je v časti Príloha C.

Než rozoberieme výsledky experimentov, zavedieme si pojem hustota domény. Hustotou budeme myslieť počet subdomén nejakej domény. Čím viac subdomén bude doména mať, tým bude hustejšia. Ak budeme hustote priraďovať číselné hodnoty, budeme tým myslieť absolútny počet subdomén. Na vyhodnotenie výsledkov sme sa rozhodli rozdeliť domény do skupín podľa hustoty. Aby sme sa neobmedzovali na dáta z našich experimentov používajúcich 1000 DNS dotazov, pre účely tohoto rozdelenia sme vykonali pre každú doménu slovníkový útok s 10000 DNS dotazmi. Domény sme rozdelili do štyroch skupín, viď tabuľku 4.1.

Štvrtá skupina je veľmi špecifická, pretože obsahuje veľmi husté domény. Konkrétne sú to msn.com, etoro.com, vk.com, yelp.com, medium.com, udemy.com, etsy.com, qualtrics.com, wordpress.org, telegram.org, github.com, bitly.com, zoopla.co.uk, zendesk.com, slack.com, quora.com. Pri vyššie spomenutom slovníkovom útoku s 10000 dotazmi mali

	hustota	**počet domén**
skupina 1	$[0, 100)$	65
skupina 2	$[100, 300)$	52
skupina 3	$[300, 1500)$	17
skupina 4	$[1500, \infty)$	16

Tabuľka 4.1: Rozdelenie domén

tieto domény hustotu viac ako 9000. Na zistenie príčiny sme vykonali experiment, kde sme pre každú použili príkaz `dig <náhodný reťazec>.doména.xyz`, náhodný reťazec je dlhý 20 znakov, obsahuje veľké a malé písmená a číslice, aby sme vytvorili doménové meno s minimálnou šancou na skutočnú existenciu. V každom zo 16 prípadov sme dostali kladnú odpoveď. Ukážku niektorých z nich prikladáme na obrázku 4.1. To nás priviedlo k záveru, že tieto domény majú v DNS systéme wildcard záznamy, čo znamená, že daná doména sa úspešne priradí ku svojej IP adrese s hocijakou subdoménovou predponou, aj keď spolu tvoria meno neexistujúcej domény. Toto podporuje aj fakt, že všetky tieto domény majú wildcard certifikáty. U každej z nich sme ale zistili, že pre niektoré mená zo slovníka sa vzniknutá subdoména nenachádza v DNS. Zároveň u niektorých domén, ako napríklad github.com, bitly.com, quora.com, sa táto množina mien zvykne v horizonte niekoľkých hodín meniť. Príčinu tohto javu sme nezistili.

Domény v tretej skupine majú vysokú hustotu, ale nepoužívajú wildcard záznamy. Veľkosť takejto infraštruktúry vytvára veľkú réžiu, subdomény treba spravovať a zabezpečovať pre ne certifikáty. Nie všetky subdomény potrebujú certifikáty, aj tak to však vedie ku zvýšeniu prácnosti. Tu vidíme dôvod priepastného rozdielu v hustote domén tretej a štvrtej skupiny. Myslíme si, že niekde okolo hodnoty 1500 je hranica, kedy si firmy radšej vytvoria wildcard DNS záznam.

4.1 Výsledky

Na zhodnotenie výsledkov sme použili podobný vzorec, ako pri optimalizovaní parametrov s_1 a s_2. Vyjadruje priemer relatívneho rozdielu hustoty medzi slovníkovým útokom a našim algoritmom naprieč doménami v skupine vyjadrený v percentách. Nech $i \in \{1, 2, 3, 4\}$ je index skupiny, D_i je množina domén v danej skupine, $d \in D_i$, s_d je hustota domény d podľa slovníkového útoku a a_d je hustota podľa nášho algoritmu, potom výsledok pre danú skupinu je

$$v_i = \frac{\sum_{d \in D_i} (a_d - s_d)/s_d}{|D_i|} * 100.$$

```
;; QUESTION SECTION:
;58aBfFTsq77rROYEUaZo.msn.com.  IN      A

;; ANSWER SECTION:
58aBfFTsq77rROYEUaZo.msn.com. 0 IN      CNAME   www-msn-com.a-0003.a-msedge.net.
www-msn-com.a-0003.a-msedge.net. 0 IN   CNAME   a-0003.a-msedge.net.
a-0003.a-msedge.net.    0       IN      A       204.79.197.203


;; QUESTION SECTION:
;eUts51flZYvuGJEpjGVV.etoro.com.        IN      A

;; ANSWER SECTION:
eUts51flZYvuGJEpjGVV.etoro.com. 0 IN    A       13.79.184.58


;; QUESTION SECTION:
;y3iUhq5oNDviCCnjjeU5.vk.com.   IN      A

;; ANSWER SECTION:
y3iUhq5oNDviCCnjjeU5.vk.com. 0  IN      A       87.240.137.164
y3iUhq5oNDviCCnjjeU5.vk.com. 0  IN      A       87.240.132.67
y3iUhq5oNDviCCnjjeU5.vk.com. 0  IN      A       87.240.132.78
y3iUhq5oNDviCCnjjeU5.vk.com. 0  IN      A       93.186.225.194
y3iUhq5oNDviCCnjjeU5.vk.com. 0  IN      A       87.240.129.133
y3iUhq5oNDviCCnjjeU5.vk.com. 0  IN      A       87.240.132.72
```

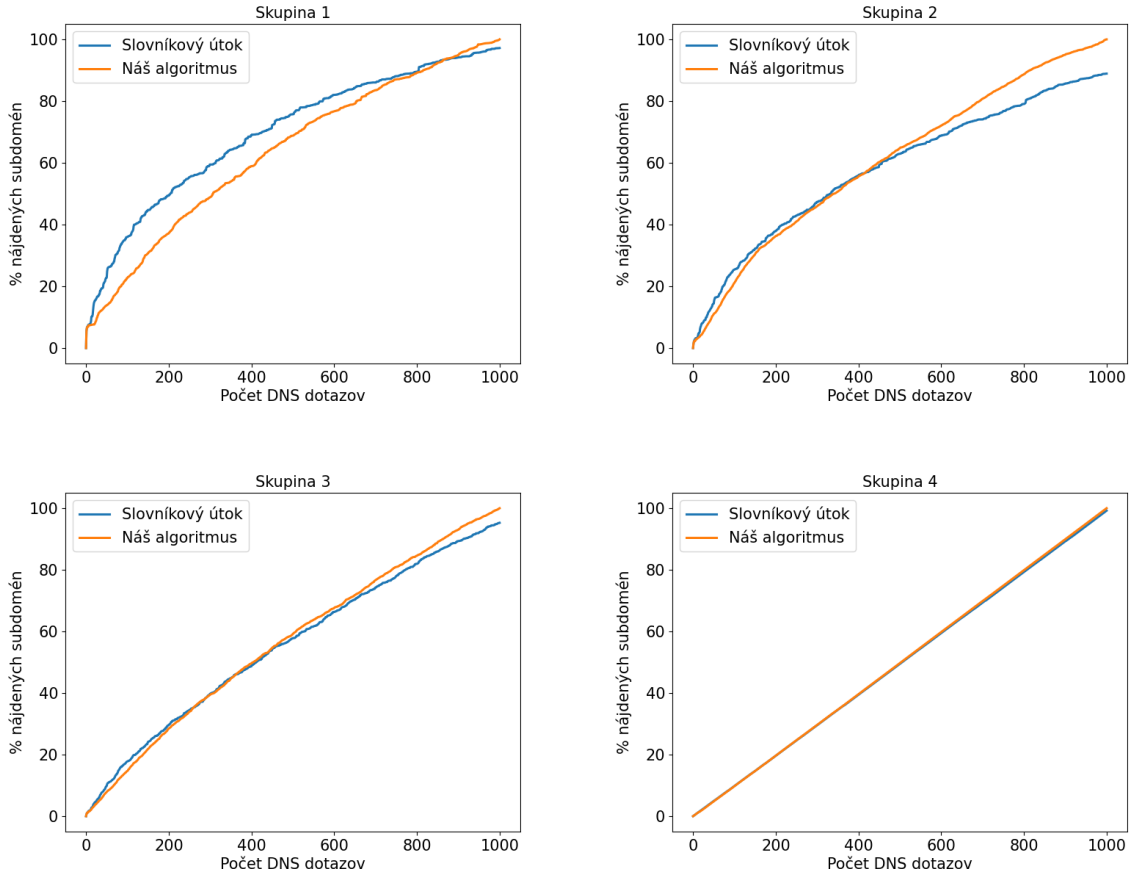
Obr. 4.1: Príklad odpovede na náhodnú subdoménu v doméne zo skupiny 4

V tabuľke 4.2 uvádzame výsledky experimentov jednotlivých skupín domén. Ďalej na obrázku 4.2 uvádzame grafy znázorňujúce vývoj hustoty v priebehu algoritmu. V nich je na horizontálnej osi časový údaj reprezentovaný počtom už vykonaných DNS dotazov a na vertikálnej osi je percentuálny podiel už nájdených subdomén, počítaný z počtu subdomén, ktoré našiel náš algoritmus po všetkých 1000 dotazoch. Dáta sú spriemerované cez všetky domény z danej skupiny.

	v_i
skupina 1	4,24
skupina 2	25,64
skupina 3	7,52
skupina 4	0,80

Tabuľka 4.2: Výsledky experimentov

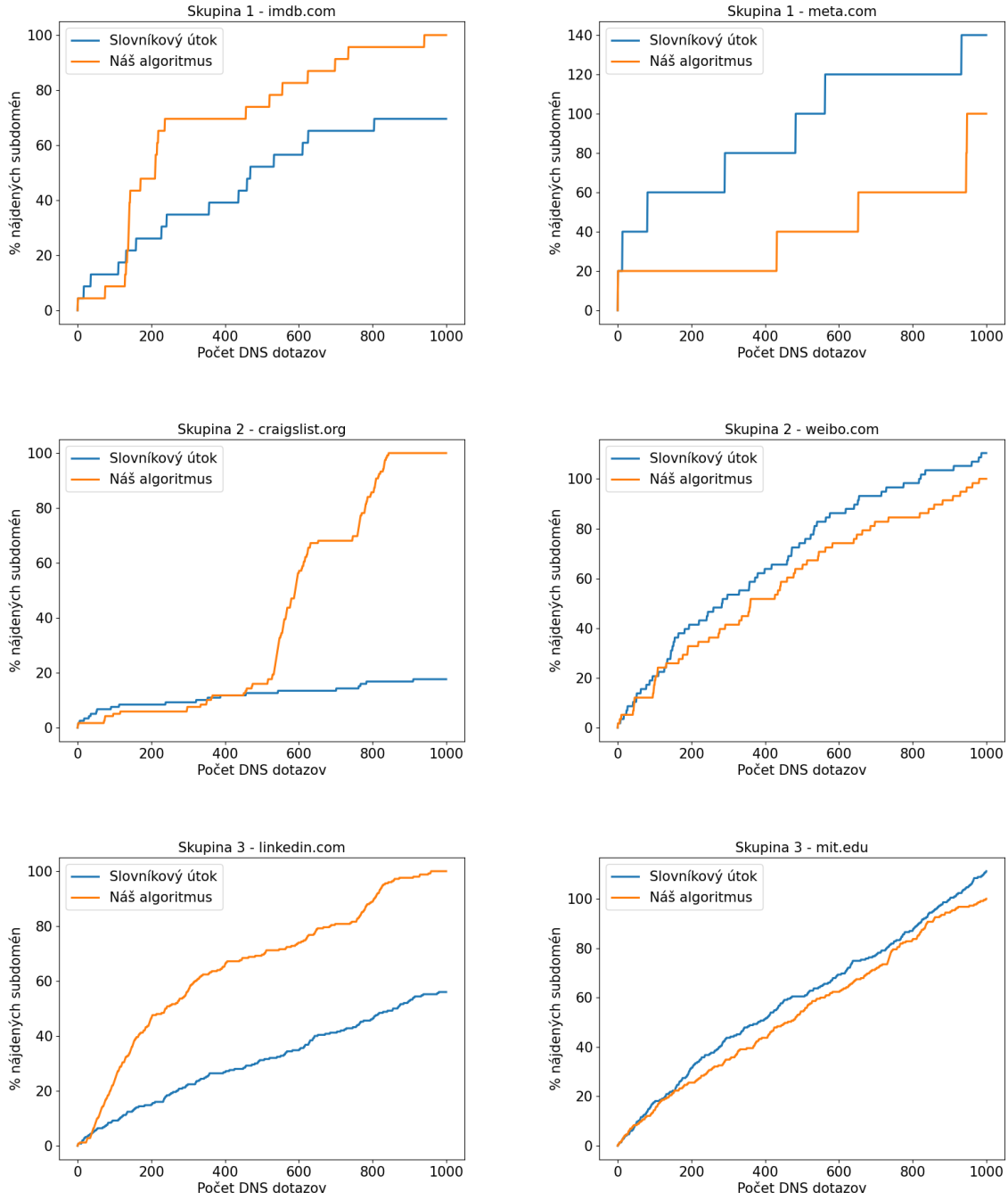
Z grafov je vidno, že slovníkový útok zo začiatku predbieha náš algoritmus. To je spôsobené tým, že použitý slovník je pravdepodobne kvalitne utriedený podľa popularity subdoménových mien v reálnom svete. Zároveň má náš algoritmus tendenciu sa na začiatku zameriavať na mená s vyšším indexom v slovníku než tie, ktoré skúša slovníkový útok. Až neskôr, keď začne skúšať mená zo začiatku slovníka, predbehne slovníkový útok a krivky sa pretnú. Až na štvrtú skupinu majú krivky len jeden zjavný priesečník, aj keď v prípadoch druhej a tretej skupiny sa krivky po dobu asi 200 dotazov držia veľmi blízko seba, než sa od seba výrazne vzdialia. Pri štvrtej skupine malo každé slovníkovým útokom skúšané meno počas našich experimentov viac ako 90% šancu byť subdoménou kvôli vysokej hustote domén v skupine. Napriek tomu dosiahol náš



Obr. 4.2: Vývoj hustoty v priebehu algoritmu

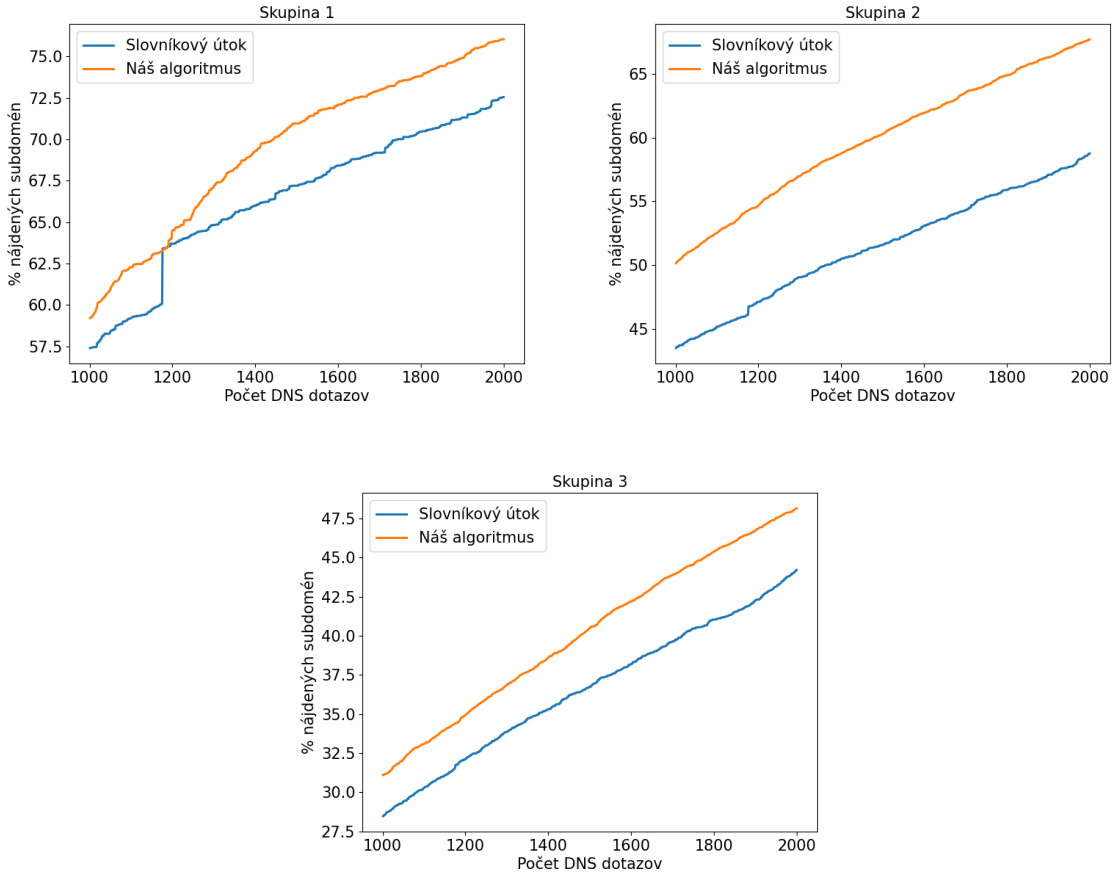
algoritmus v priemere zhruba rovnaké výsledky. Aj vďaka tomu, že väčšina mien s indexmi od 1000 do 10000 boli tiež subdoménami. Krivky v grafe sú kvôli vysokej hustote takmer identické. V prípade takýchto hustých domén sa zdá, že náš algoritmus neprináša žiadne výhody oproti slovníkovému útoku. Podobne v prípade prvej skupiny je náš algoritmus len zanedbateľne lepší od slovníkového útoku. Tieto domény sú pre neho príliš riedke, čo naznačuje aj fakt, že krivky v grafe sa pretínajú až tesne pred koncom behu. Náš algoritmus tu zo začiatku mrhá príliš veľa dotazmi skúšaním mien s vysokými indexmi v slovníku. Výnimočne dobré výsledky dosiahol algoritmus na doménach z druhej skupiny. Na základe toho môžeme predpokladať, že domény s hustotou z intervalu prislúchajúcemu tejto skupine sú ideálne na využitie nášho algoritmu. Poskytujú dostatočné množstvo subdomén na to, aby sa využili vlastnosti algoritmu a zároveň nie sú tak husté, aby bolo jedno, aký prístup k enumerácii použijeme. Toto sa ale pravdepodobne ťažšie deteguje pred vykonaním samotnej enumerácie.

Ďalej na obrázku 4.3 uvádzame grafy najlepších a najhorších domén v skupinách jedna až tri podľa výsledkov zo vzorca $(a_d - s_d)/s_d$, kde a_d je hustota podľa nášho algoritmu a s_d je hustota podľa slovníkového útoku. Pri štvrtej skupine nemá zmysel rozoberať



Obr. 4.3: Vývoj hustoty v priebehu algoritmu jednotlivých domén, najlepšia zo skupiny vľavo, najhoršia vpravo

jednotlivé prípady, pretože všetky vyzerajú takmer identicky. Grafy zobrazujú hodnoty rovnako ako tie na obrázku 4.2, teda percentá na vertikálnej osi sú počítané z hustoty, ktorú dosiahol náš algoritmus. To znamená, že ak slovníkový útok dosiahol lepšiu hustotu, bude mať v grafe viac ako 100%. Vidno, že v skupinách existujú extrémy, ktoré sa výrazne líšia od priemeru. Najmä v skupinách dva a tri rozdiel medzi hustotami z nášho algoritmu a zo slovníkového útoku je výrazný v dobrom prípade. Zároveň v zlom



Obr. 4.4: Vývoj hustoty po tisícom DNS dotaze

rozdiel nie je veľmi výrazný. Toto určite prispelo k dobrým priemerným výsledkom týchto skupín. V prvej skupine sú rozdiely zhruba podobné, čo sa zhoduje s tým, že v priemere bol v prvej skupine náš algoritmus približne rovnako účinný ako slovníkový útok. To, že náš algoritmus potrebuje určité množstvo DNS dotazov na to, aby predbehol slovníkový útok, potvrdzujú grafy domén imdb.com a craigslist.org.

Zaujímalo nás tiež, ako sa budú grafy vyvíjať za hranicou 1000 dotazov. Pre skupiny jeden až tri sme algoritmus spustili s počtom 2000 DNS dotazov. Grafy prikladáme na obrázku 4.4. Percentá na vertikálnej osi sú tentokrát počítané z hustoty, ktorú dosiahol slovníkový útok s 10000 dotazmi a zobrazujeme len dáta z intervalu $[1000, 2000]$ dotazov. Vo všeobecnosti sa dá povedať, že efektivita nášho algoritmu je v tomto intervale zhruba rovnaká ako slovníkového útoku. Krivky rastú približne rovnakým tempom, až na jeden úsek tesne pred hranicou 1200 dotazov v grafe prvej skupiny. Tam krivka slovníkového útoku rýchlo narastie až na úroveň nášho algoritmu, ten sa neskôr ale opäť začne postupne vzďaľovať.

Záver

Myslíme si, že experimenty potvrdili existenciu korelácie medzi doménovými menami aj jej využiteľnosť pri enumerácii. Videli sme to najmä pri optimalizácii hodnôt parametrov. Rôzne hodnoty dokázali výrazne ovplyvniť výsledok algoritmu. Špeciálne, lepšie výsledky dosahoval algoritmus vtedy, keď hodnoty s_2 boli výrazne vyššie ako s_1. To znamená, že ak nejaká subdoména nebola v DNS, tak znevýhodnenie mien, ktoré sa s ňou vyskytovali v modeli malo priaznivejší účinok ako ich zvýhodnenie v prípade, keď sa v DNS nachádzala. To môže byť spôsobené len vzorkou certifikátov, z ktorých sme model vytvárali, alebo to môže byť všeobecný jav. Do budúcna by stálo za preskúmanie vytvorenie modelov z iných vzoriek certifikátov, alebo získať dáta do modelu z iných zdrojov a následne testovať algoritmus na týchto modeloch. Ďalším vylepšením algoritmu by mohlo byť pokračovanie v optimalizovaní parametrov s_1 a s_2, či už so zvýšeným počtom DNS dotazov, skúšaných domén, alebo kombináciou oboch. Dá sa povedať, že náš algoritmus môže byť užitočnejší ako slovníkový útok v prípadoch, keď je k dispozícii dostatočne vysoký počet DNS dotazov. Dáta z našich experimentov naznačujú, že tento počet je zhruba 500. Naopak sa príliš nehodí v situáciach, kde je prioritou čas enumerácie, keďže popri čakaní na odpovede z DNS spotrebováva citeľné množstvo času aj na výpočty nad vektorom pravdepodobností určujúcim poradie skúšaných mien.

Literatúra

[1] aboul3la. Sublist3r: Fast subdomains enumeration tool for penetration testers, GitHub. [Citované 2022–12–11] Dostupné z `https://github.com/aboul3la/Sublist3r`.

[2] Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, Stephen Farrell, and David Cooper. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, Máj 2008.

[3] Certificate Transparency. How CT Works. [Citované 2023–3–22] Dostupné z `https://certificate.transparency.dev/howctworks/`.

[4] Florian Weimer. Passive DNS Replication. Technical report, Apríl 2005.

[5] Ben Laurie, Adam Langley, Emilia Kasper, Eran Messeri, and Rob Stradling. Certificate Transparency Version 2.0. RFC 9162, December 2021.

[6] OWASP Amass Project. amass: In-depth Attack Surface Mapping and Asset Discovery, GitHub. [Citované 2023–2–26] Dostupné z `https://github.com/OWASP/Amass`.

[7] OWASP Amass Project. amass/examples/wordlists, GitHub. [Citované 2023–2–26] Dostupné z `https://github.com/OWASP/Amass/tree/master/examples/wordlists`.

[8] Paul Pols. The Unified Kill Chain. White paper, Február 2010.

[9] Sectigo Limited. crt.sh: Monitored Logs. [Citované 2023–2–22] Dostupné z `https://crt.sh/monitored-logs`.

[10] The MITRE Corporation. MITRE ATT&CK. [Citované 2023–4–4] Dostupné z `https://attack.mitre.org/`.

[11] TheRook. subbrute: A DNS meta-query spider that enumerates DNS records and subdomains., GitHub. [Citované 2022–12–11] Dostupné z `https://github.com/TheRook/subbrute`.

Príloha A: obsah elektronickej prílohy

V elektronickej prílohe priloženej k práci sa nachádzajú zdrojové kódy, súbory s výsledkami experimentov a súbory s datasetmi, slovníkom a zoznamom domén. Obsah je zverejnený aj na stránke `https://github.com/samuelrevucky/bc_subdomain_enumeration`.
Algoritmus na získavanie certifikátov je v súbore `crt_pg.py` a algoritmus na čistenie získaného datasetu v súbore `crt_filter.py`. Vyčistený aj nevyčistený dataset sa nachádza v priečinku `datasets`.

Návod na prípravu prostredia pre spustenie enumerácie je v súbore `README`. Algoritmus na enumeráciu sa nachádza v súbore `alg.py`.

```
usage: alg.py [-h] [-w] [-D] [-d] [-n] [-r]

Subdomain enumerating script. Uses dictionary attack and our adaptive
algorithm.

options:
  -h, --help  show this help message and exit
  -w          dictionary
  -D          dataset
  -d          domain list
  -n          number of DNS queries
  -r          output directory
```

Program `alg.py` predvolene vykonáva 1000 DNS dotazov a ako ostatné argumenty berie súbory, ktoré sa nachádzajú v prílohe. To je vyššie spomínaný dataset, slovník `top_mil_110000.txt` a zoznam domén `domains_150.txt`.

Príloha B

```
# p - probability vector
# a - correlation model
# n - number of DNS queries
# wordlist - list of dictionary words
# wordlist_indices - associative array of dictionary words and
#                    their indices in wordlist

import numpy as np

mask = np.zeros((len(p),))
n_ = len(p)
i = 0
while i < n and len(mask[mask == 0]) > 0:
    w = np.argmax(p)
    p[w] = 0
    if mask[w]:
        continue
    n_ -= 1
    w = wordlist[w]
    if dns_query(w + '.' + domain):
        mask[wordlist_indices[w]] = 1
        update = np.zeros((len(p),))
        if w in a:
            for word in list(a[w]):
                update[wordlist_indices[word]] = a[w][word]
        p = p + update * s1
    else:
        mask[wordlist_indices[w]] = -1
        update = np.full((len(p),), 1/n_)
        if w in a:
            for word in list(a[w]):
                update[wordlist_indices[word]] = 0
        p = p + update * s2
    p[wordlist_indices[w]] = 0
    p[p > 0] /= sum(p)
    i += 1
```

V tejto prílohe pripájame kód jadra nášho algoritmu. To vykonáva operácie s pravdepodobnostným vektorom a selektovanie mien na DNS dotazy zo slovníka. Kód vykonávajúci réžiu ako načítavanie zo súborov, zapisovanie výsledkov do súborov, vytváranie modelu a posielanie DNS dotazov nie je súčasťou prílohy. Taktiež neprikladáme algoritmy, ktoré sme používali na vyhodnocovanie výsledkov, vykresľovanie grafov, získavanie certifikátov a čistenie datasetu.

Príloha C

Tu uvádzame všetkých 150 domén, na ktorých sme vykonávali experimenty. Domény sú uvedené po skupinách spolu s počtami subdomén, ktoré našiel slovníkový útok s 10000 DNS dotazmi.

Skupina 1

pixabay.com	4	steamcommunity.com	5
honhai.com	12	apnews.com	13
exxon.com	13	ziprecruiter.com	14
bookingbuddy.com	15	meta.com	15
pexels.com	17	zillow.com	17
unitedhealthgroup.com	19	zara.com	19
flickr.com	21	android.com	24
loreal.com	28	steampowered.com	28
europa.eu	29	zomato.com	31
techcrunch.com	32	vimeo.com	33
stackoverflow.com	35	blogger.com	36
stellantis.com	39	jetbrains.com	40
tripadvisor.com	43	tiktok.com	44
volkswagen.com	44	coinbase.com	45
interactivebrokers.com	45	cargurus.com	46
instagram.com	46	soundcloud.com	46
allrecipes.com	48	cvshealth.com	51
hbo.com	52	imdb.com	53
wellsfargo.com	58	gnu.org	60
theguardian.com	62	costco.com	63
capitalone.com	64	dropbox.com	64
buzzfeed.com	65	huawei.com	66
office.com	66	xbox.com	68
youtube.com	70	reuters.com	71
sanofi.com	71	upwork.com	71
zulily.com	75	sony.com	76

zoominfo.com	77	dailymail.co.uk	79
zynga.com	82	hulu.com	84
mcdonalds.com	84	cam.ac.uk	86
hugoboss.com	87	mozilla.org	93
forbes.com	94	pinterest.com	94
bbc.co.uk	97	chase.com	97
foxnews.com	98		

Skupina 2

netflix.com	102	nytimes.com	102
weibo.com	102	airbnb.com	106
samsung.com	108	washingtonpost.com	108
twitch.tv	114	expedia.com	117
fidelity.com	119	ox.ac.uk	121
salesforce.com	121	wikimedia.org	124
amd.com	129	npr.org	130
shell.com	130	db.com	131
sap.com	131	toshiba.com	132
adidas.com	138	booking.com	139
ibm.com	142	intel.com	142
bankofamerica.com	145	twitter.com	145
un.org	146	cnn.com	148
walmart.com	154	nasa.gov	157
xinhuanet.com	162	webmd.com	164
porsche.com	168	astrazeneca.com	175
oracle.com	177	playstation.com	189
wiktionary.org	192	nike.com	193
columbia.edu	196	pfizer.com	197
zoho.com	197	craigslist.org	204
cnbc.com	210	wikipedia.com	212
americanexpress.com	218	hilton.com	226
wsj.com	240	apple.com	245
godaddy.com	262	yale.edu	262
ebay.com	267	dell.com	269
google.com	273	adp.com	276

Skupina 3

adobe.com	300	accenture.com	333
amazon.com	354	alibaba.com	374

linkedin.com	389	harvard.edu	399
microsoft.com	434	cisco.com	473
princeton.edu	477	facebook.com	491
yahoo.com	606	caltech.edu	645
baidu.com	751	berkeley.edu	918
zoom.us	1056	mit.edu	1146
stanford.edu	1456		

Skupina 4

msn.com	9507	etoro.com	9865
vk.com	9903	yelp.com	9903
medium.com	9917	udemy.com	9917
etsy.com	9936	qualtrics.com	9939
wordpress.org	9942	telegram.org	9957
github.com	9958	bitly.com	9960
zoopla.co.uk	9970	zendesk.com	9977
slack.com	9980	quora.com	9981