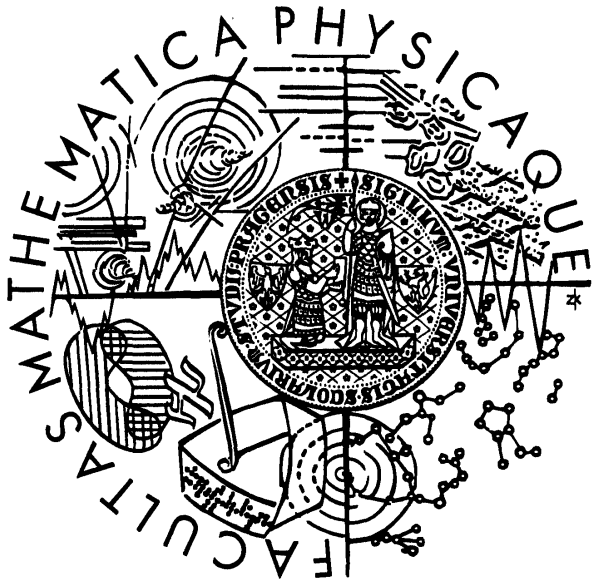
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Norbert Hanuska

Google hacking

Katedra softwarového inženýrství

Vedúci bakalárskej práce: Doc. RNDr. Pavel Pyrih, CSc.,

Katedra matematické analýzy

Štúdijný program: Informatika, Obecná informatika

2008

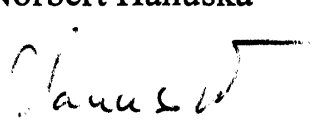
Poďakovanie

Moje poďakovanie patrí predovšetkým vedúcemu práce Doc. RNDr. Pavlovi Pyrihovi, CSc. za odborné vedenie, venovaný čas a pripomienky k vypracovaniu tejto práce a tiež RNDr. Tomášovi Pochovi za jeho konzultáciu a cenné rady.

Prehlasujem, že som svoju bakalársku prácu vypracoval samostatne a použil k tomu len literatúru, ktorú uvádzam v zozname priloženom k bakalárskej práci. Súhlasím so zapožičiavaním práce a jej zverejňovaním.

V Prahe dňa 25. 5. 2009

Norbert Hanuska



OBSAH

Názov práce: Google hacking

Autor: Norbert Hanuska

Katedra (ústav): Katedra softwarového inženýrství

Vedúci bakalárskej práce: Doc. RNDr. Pavel Pyrih, CSc., Katedra matematické analýzy

e-mail vedúceho bakalárskej práce: Pavel.Pyrih@mff.cuni.cz

Abstrakt: Cieľom bakalárskej práce je podať prehľad o možnostiach využitia vyhľadávača Google ako hackovacieho nástroja a to ako teoreticky, tak aj demonštratívne pomocou konkrétnych príkladov. Rozoberajú sa možnosti pokročilého vyhľadávania pomocou vyhľadávača, vysvetľujú sa princípy tvorby efektívnych dotazov. Práca sa zameriava najmä na popis jednotlivých metód Google hackingu, teda získavania citlivých alebo inak zaujímavých dát. Taktiež sa venuje ochrane proti takýmto typom útokov a na základe dostupných informácií sa snaží o zhodnotenie možných rizík. V neposlednom rade je priestor venovaný aj vyhľadávaciemu enginu Googlu, metóde ohodnocovania výsledkov vyhľadávania a určovaniu ich relevancie. Ďalej sú analyzované alternatívne spôsoby zneužitia vyhľadávača Google prostredníctvom Google bômb. V krátkosti sú spomenuté aj konkrétne využitia potenciálu Googlu na reálnych prípadoch.

Kľúčové slová: hacking, Google, vyhľadávanie, PageRank, Google bomba

Title: Google hacking

Author: Norbert Hanuska

Department: Department of software engineering

Supervisor: Doc. RNDr. Pavel Pyrih, CSc., Department of mathematical analysis

Supervisor's e-mail address: Pavel.Pyrih@mff.cuni.cz

Abstract: The aim of the bachelor thesis is to theoretically and practically with examples demonstrate possibilities of using Google search as a hacking tool. It analyzes possibilities of advanced searching by the Google and it's pros and cons that can be used for malicious purposes. Next, thesis also explains principles of building effective Google queries. Orientation is mainly to review methods of Google hacking, to serve possible solutions of protection and it tries to evaluate and summarize possible risks based on the available information. It also tries to explain Google search engine, method of ranking the results and determination of relevancy of each result. In additon, thesis tries to analyze alternative ways of using Google for malicious purposes by Google bombs. There are also mentioned real case studies of Google hacking.

Key words: hacking, Google, searching, PageRank, Google bomb

1 ÚVOD

Google ako jeden z najpopulárnejších prostriedkov na vyhľadávanie informácii na Internete sa stal pojmom. Kedykoľvek nás napadne nejaká otázka, obrátime sa s ňou najčastejšie práve na Google. A keď vieme klásť tie správne otázky, je vysoká pravdepodobnosť, že sa k vytúženej odpovedi skutočne dopátrame.

Nie každý si však uvedomuje, akou silou Google oplýva a že sa dá zneužiť aj k dopátraniu sa k veľmi citlivým informáciám, akými sú mená, heslá, osobné údaje, konfiguračné súbory, alebo informácie týkajúce sa bezpečnosti webov a ich zraniteľnosti.

Vo svojej práci sa pokúšam upozorniť na toto nebezpečenstvo, podať prehľad účinných vyhľadávacích stratégií a samozrejme sa zamerať aj na možnosti obrany pred takýmto typom útokov, nazývaných aj Google hacking. Dobré schopnosti v Googli sú však výhodné aj mimo oblasť informačnej bezpečnosti, preto sa v práci venujem aj základom vyhľadávania pomocou Googlu.

Ako množstvo našich vedomostí, správ a prejavu smeruje na web, vyhľadávacie enginy sa stále viac stávajú akýmisi strážcami, sprostredkovateľmi kyberpriestoru. A čo viac, jeden vyhľadávací engine – Google – teraz ovláda majoritu vyhľadávacích dotazov. Google smeruje stovky miliónov užívateľov k určitému obsahu a nie inému, k určitým zdrojom a nie iným. Z tohto pohľadu som sa snažil neopomenúť základy fungovania vyhľadávacieho enginu Googlu a jeho algoritmus PageRank, ktorých pochopenie pomáha k lepšiemu preniknutiu do podstaty práce.

Snažil som sa spomenúť aj témy, ktoré podľa prísnej definície do Google hackingu nepatria, no podľa môjho názoru s ním súvisia. Takými sú napríklad Google bomby, na ktoré som sa viacej zameral z algoritmického hľadiska a snažil som sa prísť na niektoré myšlienky ukryté za ich zneškodňovaním.

V prípade Google hackingu je potrebné zadávať dotazy veľmi konkrétne a premyslene, aby sme maximalizovali efektivitu a relevanciu našich výsledkov. K tomu slúžia najmä pokročilé vyhľadávacie techniky ponúkané Googlom, a tiež znalosť chodu rôznych softwarových produktov. Informácie a dáta získané prostredníctvom Google hackingu môžu byť následne zneužívané a útočníkovi nápomocné v budúcnosti, či už sa jedná o rôzne exploity, štatistiky webových serverov, alebo osobné údaje využiteľné sociotechnikmi. Zameral som sa aj na využitie automatizačných techník pre získavanie dát z Googlu a ich následnú analýzu. Na záver práce sú spomenuté aj konkrétne prípady využitia Googlu ako hackovacieho nástroja.

2 VYHĽADÁVANIE NA INTERNETE

Na Internete sa nachádza ohromné množstvo dát, odhady počtu stránok sa pohybujú v miliardách až desiatkach miliárd. Každý deň vznikajú milióny nových, ďalšie sa menia, iné mažú. Na Internete je teda neobvykle rušno. Aby v takomto extrémnom množstve dát nevládol zmätok a dokázali sme sa v ňom orientovať, musíme informácie nejakým spôsobom získavať, vyhľadávať. Nájdenie konkrétnej informácie v mori stránok zabezpečuje práve proces zvaný vyhľadávanie a stránky, ktoré ho sprostredkúvajú, sa nazývajú vyhľadávače [1].

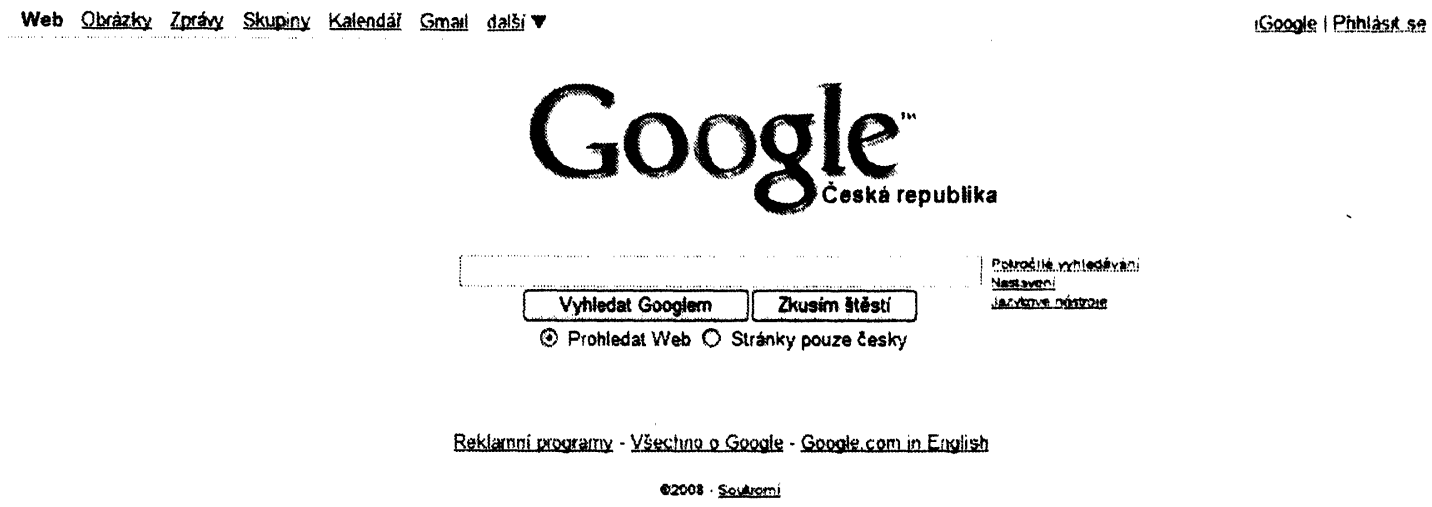
2.1 História a súčasnosť Googlu

Za vznikom celosvetovo úspešného vyhľadávača s názvom Google stoja dvaja postgraduálny študenti, Larry Page a Sergey Brin. Google vznikal ako ich spoločné dielo pri príprave dizertačnej práce na prestížnej americkej univerzite Stanford. Larry Page dostal zdanlivo šialenú myšlienku, že stiahne do svojho počítača celý web. Svojmu konzultantovi tvrdil, že to zvládne v priebehu jedného týždňa. Za jeden rok sa mu toho naozaj kúsok stiahnuť podarilo. Spolu uviedli do prevádzky prvú verziu vyhľadávača, ešte pod názvom BackRub a prevádzkovali ho v rámci univerzitnej siete Stanfordu. Postupne sa ich projekt rozrastal a keď v roku 1997 hľadali nový názov pre svoj vyhľadávač, ujal sa „Google" odvodený od slova „googol". Slovo „googol" bolo vymyslené Miltonom Sirottom, 9-ročným synovcom matematika Edwarda Kasnera a je matematickým výrazom pre číslo 1 so sto nulami. Použitie tohto slova reflektuje vôľu zakladateľov indexovať, prehľadávať a radiť pre ľudí nepreberné množstvo dát. Larry Page si k 15. septembru 1997 zaregistroval doménu *google.com* a neskôr, k 27. septembru 1998 aj firmu Google Inc., aby napokon 21.septembra 1999 bol Google oficiálne spustený [2].

V dnešnej dobe je Google jednotkou medzi vyhľadávačmi s najväčším podielom na trhu (celosvetovo približne 62% všetkých vyhľadávaní [3]), konkurovať sa mu snažia hlavne Yahoo a Microsoft. Za svoj úspech vďačí najmä jednoduchosti, pokročilým možnostiam zadávania dotazov a prepracovaným systémom generovania výsledkov. Práve systém hodnotenia webových stránok, nazývaný aj PageRank, ktorý vracia odkazy na určité stránky častejšie a s vyššou prioritou než na tie ostatné, je jedno z najväčších obchodných tajomstiev Googlu. Prvotnú formulu PageRanku zostavil jeden zo zakladateľov spoločnosti, Larry Page [2].

2.2 Webové rozhranie Googlu

Webové rozhranie Googlu je pre užívateľa veľmi jednoduché a prehľadné, no napriek tomu neobyčajne silné. Dnes sa už nad hlavnou stránkou Googlu takmer nikto nepozastaví a nájdeme ju na adrese *www.google.com* (poprípade *www.google.cz*, stránky „českého" Googlu – s českým jazykovým rozhraním).



Obr. 1. Hlavná stránka Google Česká republika

Prvoradým prostriedkom na interakciu s užívateľom je vstupné textové pole, do ktorého sa vkladajú jednotlivé vyhľadávacie dotazy. Nad vstupným textovým poľom sa objavujú aj odkazy otvárajúce ďalšie sekcie vyhľadávania. Základná vyhľadávacia funkcionalita je rovnaká, ale v niektorých sekciách sa vyskytujú odlišné schopnosti, napríklad sa príjmajú iné vyhľadávacie operátory. Po spracovaní dotazu sa zobrazí stránka výsledkov. Výsledky sú zobrazené podľa relevantnosti, teda Google nám ponúkne na prvých miestach tie odkazy, o ktorých predpokladá, že sú pre nás primárne. Pri každej položke na stránke výsledkov vypíše Google názov webu, zhrnutie o webe na niekoľko málo riadkov, URL stránky obsahujúcu zhodu a jej veľkosť, odkaz *Archív (Cache)*, ktorý zobrazí stránku tak, ako vypadala keď ju Google prehľadal naposledy a odkaz na stránky s podobným obsahom.

Za zmienku taktiež stojí odkaz *Pokročilé vyhledávání (Advanced Search)*. Z rozhrania stránky, ktorá sa nám zobrazí po kliknutí na odkaz je možné vykonávať pokročilé vyhľadávania s možnosťami, ktoré by nám boli za normálnych okolností prístupné iba prostredníctvom modifikácie URL adresy, alebo niektorých pokročilých operátorov.

2.3 Vytváranie dotazov

Základom k plnému využitiu sily Googlu je ovládnuť proces vytváranaia a zadávania správnych dotazov. Pre čo najefektívnejšie využitie Googlu sa využíva technika zvaná zužovanie (narrowing), čiže redukcia výsledkov hľadania. Predpokladom takéhoto efektívneho hľadania je zvládnutie syntaxe Google dotazov, tým ďalším faktorom je určitá prax, skúsenosť a cit pre redukciu pri hľadaní.

2.3.1 Základné pravidlá pri vyhľadávaní pomocou Googlu

Podľa oficiálnej dokumentácie Googlu sa v dotazoch nerozlišuje veľkosť písmen. Teda nie je podstatné, či napíšeme dotaz spôsobom `gOogLE`, `GooGLe`, alebo `google`, vždy by to pre Google malo byť rovnaké slovo a vrátiť nám rovnaké výsledky. Ale moje nedávne pozorovanie bolo, že po vyhľadaní slov `Hanuska` a `hanuska` nielenže Google vrátil trochu odlišné výsledky, ale aj počty nájdených stránok sa líšili (apríl 2009). Bral som do úvahy aj normálnu fluktuáciu a vplyv datacentra – to ale zostávalo nezmenené. Vyjadrenia zástupcov Googlu k tejto problematike boli koncom roku 2008 v tom zmysle, že niekedy v budúcnosti Google môže brať do úvahy "case sensitivitu" dotazov, ak to

bude považovať za relevantné a zvyšujúce kvalitu výsledkov. Vyzerá to teda, že Google na tom skutočne pracuje. Taktiež stojí za spomenutie a bolo prakticky overené, že pri dotaze s diakritikou a bez môžeme očakávať mierne odlišné výsledky.

Zástupné znaky Googlu (wildcardy) sú niečo iné, než je väčšina programátorov bežne zvyknutá. Tí obvykle chápu zástupné znaky ako symbolickú reprezentáciu akéhokoľvek jediného znaku (napríklad v Unixe otáznik), alebo sériu znakov, ktorú predstavuje hviezdička. Kdežto v Googli zástupný znak hviezdička (*) reprezentuje v dotaze jedno, alebo niekoľko celých slov [5]. Napríklad hľadanie výrazu `kurzy * varenia` zobrazí výsledky pre frázy typu

```
"kurzy vegetariánskeho varenia"
```

a pod. Bohužiaľ vyhľadávanie zástupného znaku funguje iba pre celé slová alebo frázy, Google nepodporuje vyhľadávanie, v ktorom hviezdička označuje časť, alebo pokračovanie slova a teda pri dotaze typu

```
brit* awards
```

nedostaneme výsledok vyhľadávania

```
british awards
```

Google pri hľadaní ignoruje veľmi bežné anglické slová, znaky, číslice. Napríklad *who*, *where*, *what*, *the*, *a* alebo *an*. Zaujímavé však je, že logika pre vylúčenie určitého slova môže byť pri rôznych výrazoch rôzna. Napríklad ak hľadáme výraz `what the car in`, Google ignoruje slová *what*, *the* a *in*. Ak ich však hľadáme jednotlivo, Google ich prijme ako platné termíny a hľadanie termínu `the` vedie na vyše 20 miliárd výsledkov (máj 2009, v júni 2008 to bolo 14 miliárd).

Najjednoduchší dotaz sa skladá z jediného slova alebo kombinácie slov, napr. `google`, `google hacking`, `FBI hacker mitnick`. O niečo zložitejšie je hľadanie fráz. Fráza je slovné spojenie uzatvorené v úvodzovkách, ktorým hovoríme Googlu aby hľadal dané slovné spojenie so všetkými termínmi a v presnom poradí, ako je to uvedené v úvodzovkách. V týchto prípadoch Google nevylučuje bežné slová, ktoré inak ignoruje (and, the, a..). Príklady fráz: `"vacation hawaii"`, `"luxury hotels mauii"`.

Google ignoruje v dotazoch bodku, takže frázy môžeme tiež písať technikou:

```
vacation.hawaii
```

2.4 Booleovské operátory

Frázy sú však stále pomerne elementárna forma vyhľadávania. Pre efektívnejšie vyhľadávanie a konkrétnejšie výsledky vracajúce sa z dotazov potrebujeme poznať a vedieť používať aj booleovské operátory AND, OR a NOT.

Najbežnejším je operátor AND. S jeho pomocou môžeme do dotazu začleniť viacero termínov. Napríklad výraz `johnny AND long` nám vráti výsledky, kde sa vyskytuje súčasne slovo johnny aj long. Treba však poznamenať, že pre vyhľadávací engine Googlu je operátor AND nadbytočný, pretože automaticky hľadá všetky termíny, ktoré sme zaradili do dotazu (až na spomínané výnimky).

Užitočný je symbol plus (+), ktorým si vynútime, aby sa do vyhľadávania zaradilo slovo, ktoré za ním nasleduje. Čiže ak chceme v danom výraze hľadať aj slová ako *what*, *the*, *in*, napíšeme ich so znamienkom plus, za ktorým však nesmie byť medzera.

Napríklad:

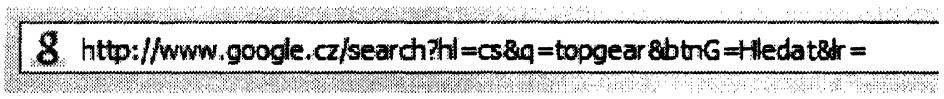
```
+the car +in
```

Ďalším bežne používaným operátorom je NOT, ktorý je funkčným opakom operátoru AND, to znamená, že vylučuje slovo z hľadania. Jeho ekvivalentom je znamienko mínus. Pri zadaní dotazu `bill -gates` sa teda z výsledkov odstránia stránky, ktoré pojednávajú o Billovi Gatesovi.

Medzi bežné operátory patrí aj Booleovský operátor OR. Môžeme ho reprezentovať aj zvislicou (|) a jeho význam je vyhľadanie buď jedného termínu, alebo druhého. Operátor má rovnakú váhy ako OR, AND, ako aj pokročilé operátory (o nich ešte budem hovoriť). Na predídenie rôznym zmätkom je vhodné použiť zátvorky, ktorým Google dobre rozumie.

2.5 Adresy URL Googlu

Každý dotaz Googlu sa dá reprezentovať pomocou URL, ktorá ukazuje na stránku výsledkov. Stránka výsledkov nie je statická, ale dynamicky sa meniaca v závislosti na tom, čo hľadáme. Odoslaním vyhľadávacieho dotazu prostredníctvom webového rozhrania sa teda dostaneme na stránku výsledkov, ktorá sa dá reprezentovať URL adresou. Predstavme si dotaz `topgear`. Ako náhle tento dotaz zadáme, Google nás presmeruje na adresu podobnú adrese na obrázku 2.



Obr. 2. URL adresa konkrétneho hľadania vygenerovaná Googlom

Ak túto adresu zadáme do poľa *Adresa* nášho prehliadača, Google znovu spracuje hľadanie `topgear` a zobrazí stránku výsledkov. Táto URL adresa sa dá veľmi ľahko modifikovať a teda aj výsledky vyhľadávania pri jej opätovnom spracovaní. Pozrime sa na jej syntax podrobnejšie. Prvá časť URL *www.google.cz/search* je umiestnenie vyhľadávacieho skriptu Googlu [4]. Otáznik ďalej znamená, že sa skriptu budú predávať určité parametre, čiže voľby, aby Google skutočne niečo urobil. Parametre sa oddeľujú znakom ampersand (&) a sú zložené z premenných, za ktorými nasleduje znak rovná sa (=) a hodnota, na ktorú sa má premenná nastaviť. Premenná *q* (query) vyjadruje odoslaný dotaz, *hl* zase jazyk rozhrania. Ak potrebujeme pridať do vyhľadávania nejaké ďalšie parametre (premenné), jednoducho ich môžeme dopísať priamo do URL v ľubovoľnom poradí, to isté platí aj pre odstraňovanie parametrov. Základná syntax teda vyzerá nasledovne:

```
www.google.com/search?premenna1=hodnota&premenna2=hodnota
```

Charakteristiky jednotlivých premenných som sa už zľahka dotkol. Popri premenných *q* a *hl* patrí medzi frekventovane používané aj premenná *lr*, ktorá obmedzuje výsledky vyhľadávania iba na stránky vytvorené v konkrétnom jazyku (*lr=lang_dk* vráti stránky iba v dánštine). Občas sa pletie s premennou *restrict*, ktorá dáva možnosť obmedziť výsledky hľadania na jednu zem alebo niekoľko zemí určených názvom domény najvyššej úrovne

(napríklad .cz) a/alebo zemepisným umiestnením IP adresy serveru [4]. Teda po zaradení parametra *restrict* do našej URL s hodnotou napríklad countryDK, Google nám vráti stránky, o ktorých si myslí, že sú fyzicky umiestnené v Dánsku. Tento predpoklad o polohe daného webu v určitom zemepisnom regióne si môžeme ľahko overiť nástrojmi *host* a *whois*. Často veľmi nápomocný je aj parameter *as_qdr*, ktorý umožňuje odfiltrovať výsledky staršie ako určité časové obdobie. Pre získanie stránok mladších ako tri mesiace teda použijeme *as_qdr=m3*.

Zoznam väčšiny bežne používaných parametrov môžeme bez väčších problémov nájsť na Internete, napríklad na adrese *http://www.joostdevalk.nl/wp-content/uploads/2007/07/google-url-parameters. pdf*.

2.6 Pokročilé operátory

Okrem základných vyhľadávacích techník Google ponúka aj špeciálne prvky nazývané pokročilé operátory, ktoré môžeme chápať ako rozširujúcu syntax slúžiacu na zužovanie výsledkov nášho hľadania. Bez použitia pokročilých operátorov v našich dotazoch Google hľadá uvedené termíny vo všetkých oblastiach webovej stránky, teda v titulku, texte, URL a podobne. Práve pokročilé operátory dovoľujú užívateľovi zamerať svoje hľadanie len na určité špecifické časti stránok a hľadať špecifický druh informácií.

Pokročilý operátor je vlastne iba časť dotazu, ktorý predávame Googlu bežne. Avšak ich syntax je dosť striktná a musí sa dodržiavať. Základná syntax je:

```
operátor:hľadaný_termín
```

Dôležité je poznamenať, že medzi operátorom, dvojbodkou a termínom nemôžu byť žiadne medzery. V opačnom prípade by Google by pochopil syntakticky nesprávny pokročilý operátor ako ďalší hľadaný termín. Hľadaný termín má rovnakú syntax ako termíny v bežne zadávanom dotaze, čo znamená, že spolu s pokročilým operátorom môžeme použiť jeden vyhľadávací termín, frázu, tak isto ako aj Booleovské operátory.

V jednom dotaze je možné kombinovať aj viacero pokročilých operátorov, niektoré sa však dajú kombinovať lepšie než iné, niektoré dokonca vôbec. Taktiež nie všetky operátory sa dajú použiť kdekoľvek. Niektoré sa dajú použiť iba v sekcii Web, niektoré zase iba v sekcii Skupiny (Groups). Dotaz

```
intitle:"index of" "top gear".
```

vracia stránky, ktoré majú v titulku frázu `index of` a okrem toho sa niekde (v titulku, v texte, v URL...) vyskytuje fráza `top gear`. Google interpretuje medzeru v dotaze ako koniec hľadaného termínu pre pokročilý operátor a pokračuje spracovaním ďalšej časti dotazu. Teda operátor *intitle* sa vzťahuje iba na frázu `index of` a už nie na `top gear`.

V nasledujúcom texte sa pokúsim v skratke podať akýsi sumár najčastejšie používaných pokročilých operátorov v technikách Google hackingu.

Intitle, allintitle

Prostredníctvom operátoru *intitle* obmedzujeme hľadanie na titulok stránky. Ten sa dá vyjadriť ako text umiestnený vo vnútri tagu TITLE v HTML dokumente.

```
intitle:"index of" "top gear"
```

vráti stránky s frázou `index of` v titulku a `top gear` niekde v stránke.

Allintitle hľadá stránky, kde sa každý zo zadaných termínov za operátorom objavuje v titulku stránky. Kombinácia operátoru *allintitle* s inými pokročilými operátormi však nemusí byť úplne bezproblémová, preto sa odporúča v čo najväčšej miere obmedziť jeho použitie.

Intext

Pátra po reťazci v texte stránky. Je vhodný, ak vieme že hľadaný text sa má nachádzať iba v texte stránky, poprípade ak niektoré výrazy sú príliš bežné v odkazoch alebo URL adresách.

Site

Operátor *site* umožňuje pátrať len po tých stránkach, ktoré hosťujú na konkrétnom serveri, alebo pochádzajú z nejakej konkrétnej domény najvyššej úrovne. Parametre pre operátor musia končiť platným názvom domény najvyššej úrovne, inak nedostaneme žiadne výsledky.

Ak teda máme predstavu aké informácie chceme hľadať a vieme, že sa nachádzajú napríklad na stránkach MFF Karlovej univerzity, jedna z možností je dotaz:

```
site:mff.cuni.cz "harmonogram akademickeho roku"
```

Inurl

Obmedzuje hľadanie na URL stránky. Operátor *inurl* je vo veľkej miere používaný, pretože umožňuje vyhľadávať podadresáre, čo nám operátor site zabezpečiť nedokázal.

```
site:apple.store.com inurl:webobjects
```

Filetype

Google dokáže hľadať aj oveľa viac než len webové stránky. Pomocou operátora *filetype* môžeme vyhľadávať súbory rôznych typov, konkrétne teda operátor *filetype* hľadá stránky končiace na danú príponu súboru, ktorá je súčasťou URL. Hlavné typy súborov, ktoré Google hľadá sú html, pdf, ppt, xls, doc, txt a xml, ans, txt, ps. Vo svojej databáze má však informácie o omnoho väčšom množstve prípon.

Link

Operátor *link* umožňuje vyhľadávať stránky, ktoré majú odkazy na iné stránky. V argumente sa nezadáva hľadaný termín, ale URL alebo názov serveru. V rámci techník zužovania vyhľadávania pamätajme na to, že dlhé URL sú konkrétnejšie, a preto môžu vracať menej výsledkov.

Numrange

Operátor *numrange* dokáže vyhľadávať čísla v zadanom rozsahu, vyžaduje na to dva parametre, dolnú a hornú hranicu intervalu oddelených pomlčkou. Ak hľadáme číslo 36, bude na to stačiť dotaz v tvare `numrange:35-37`

Existuje aj skrátená verzia operátoru, v ktorej stačí namiesto *numrange* zadať obidve čísla oddelené dvomi bodkami - `35..37`. Operátor je veľmi silný, vďaka nemu sa ľahko darilo vyhľadávať čísla kreditných kariet (napr. zadaním dotazu `visa 4356000000000000..4356999999999999`), ale od určitého momentu Google na takýto dotaz začal vracať chybovú hlášku o rozpoznanom „automatickom dotaze spywaru".

Cache

Operátor *cache* slúži na zobrazenie archivovanej verzie stránky. Google ukladá časti stránok, ktoré predtým preliezol, takže k nim máme prístup na stránke výsledkov cez odkaz *Archív* (Podrobnejšie sa problematike Google archívu budeme venovať v kapitole 3). Ak by sme chceli rovno prejsť na archivovanú verziu nejakej stránky, stačí v dotaze Googlu použiť operátor *cache*.

V skratke ešte spomeniem operátor ***related***, ktorý nájde stránky podobné zadanej stránke a operátor ***info***, ktorý zobrazí súhrnné informácie o webe a zobrazí odkazy na iné hľadania Googlu. Podrobný prehľad ďalších pokročilých operátorov je možné na Internete veľmi ľahko nájsť (napríklad na *http://www.googleguide.com/advanced_operators.html*).

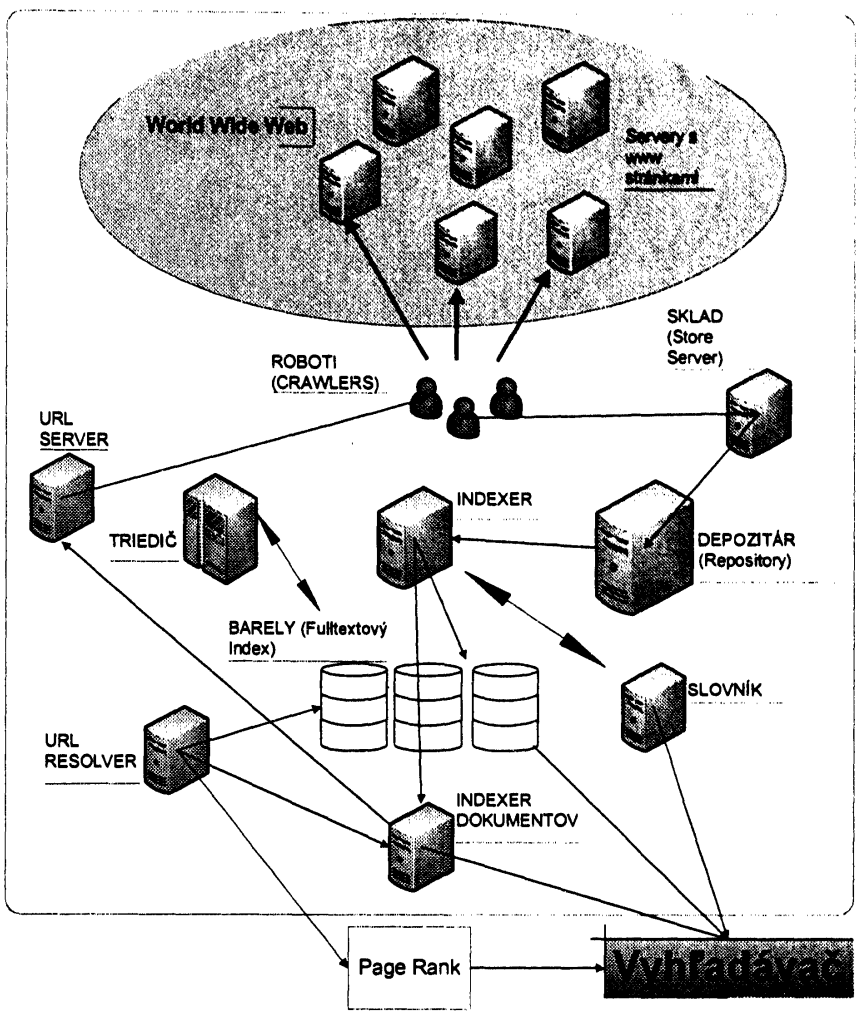
2.7 Princíp fungovania Googlu

O technológiách, ktoré používa súčasný vyhľadávač Google, je známe oveľa menej než v čase, keď šlo o akademický projekt, pretože Google ako komerčná firma už neposkytuje také množstvo informácií. Isté je, že celý vyhľadávač je rozdelený do niekoľkých datacentier, rozmiestnených po celom svete. Datacentrá sa skladajú z veľkého počtu bežných, „low-costových" PC a zodpovedajú dotazy nezávisle. Dotazy sa medzi ne prerozdeľujú pomocou rotácie DNS záznamov – DNS servery Google na každý dotaz vracajú vstupnú IP adresu iného datacentra, ktoré vyberá na základe geografickej polohy užívateľa a vyťaženosti jednotlivých centier.

Google beží na distribuovaných sieťach tisícov počítačov, a preto je schopný uskutočniť rýchle paralelné spracovanie. Čiže veľké množstvo kalkulácií je možné vykonať simultánne, čo má za následok signifikantné zrýchlenie spracovania [24].

2.7.1 Architektúra Googlu

Poďme si priblížiť základnú architektúru vyhľadávača Google, jeho hlavné časti. Pôvodnú štruktúru vyhľadávača som sa pokúsil zachytiť podľa originálneho dokumentu [26] od Sergeya Brina a Larryho Pagea, je zobrazená na obrázku 3.



Obr. 3. Štruktúra Google

Google neustále vysiela po sieti niekoľko špeciálnych programov, zvaných roboti alebo crawleri. Ich úlohou je sťahovať stránky, ktoré mu určí *URL server* (zoznam URL adries), do hlavnej databáze vyhľadávača – *skladového serveru (store server)*. Crawlerov beží niekoľko paralelne, každý naraz udržuje stovky otvorených spojení k webserverom, aby nebol zdržiavaný čakaním na ich odpovede a priebežne tak nepretržite prechádzajú miliardy stránok, ktoré dnes na Internete existujú. V skladovom serveri sa stránky komprimujú a uložia do *depozitára (repository)*. Každá stránka dostane unikátne identifikačné číslo, ktoré Google nazýva *docID*. O ich zaindexovanie sa stará tzv. *indexer* a *triedič (sorter)*. Indexer je teda akýmsi srdcom vyhľadávača, pretože obstaráva radu dôležitých činností. Každý dokument je prevedený do súboru slovných spojení zvaných hity. Hity obsahujú záznam o slove, pozíciu v dokumente, približnú veľkosť fontu a jeho rez (kapitálky, kurzíva apod.). Následne indexer odosiela tieto hity do tzv. *barelov* („zásobníkov"), ktoré tak tvoria čiastočne utriedený index. Súčasne je každé slovo prevedené na ďalšie unikátne identifikačné číslo – *wordID*. Pritom indexer spolupracuje so zdieľaným slovníkom. Akonáhle sa slovo prevedie na wordID, jeho výskyt sa zapíše do zoznamu v bareloch. Indexer zaisťuje aj ďalšiu dôležitú funkciu. Extrahuje z indexovaných stránok všetky odkazy a zaznamenáva nielen samotnú adresu v odkaze, ale tiež „anchor" text, teda text, ktorý je uvedený v odkaze. V tej chvíli vstupuje do procesu ďalší program – *URL resolver*, ktorý číta text v odkaze a prevádza relatívne adresy URL na absolútne a radí ich podľa docID. Následne tieto odkazy páruje so stránkami v už spracovanom registre. Vytvára databázu odkazov a z tejto databáze potom čerpá ďalší program pre výpočet PageRanku všetkých dokumentov. URL resolver okrem iného slúži aj ako zdroj dát pre URL server. Vyhľadávač Google uchováva index všetkých svojich internetových dát na už spomínaných data centrách (oddelené servery) a aktualizácia indexu sa postupne objavuje tak, ako sa jeden server po druhom aktualizuje [28].

Triedič pretrieďuje index do spätného indexu – hity namiesto podľa docID radí podľa wordID (ID slova, ktoré sa používa v hite) a zároveň vytvára zoznam použitých wordID a ich početností, z ktorých je vytváraný nový slovník.

Dátové štruktúry

Google sa snaží pre dosiahnutie čo najrýchlejšej odozvy používať čo najefektívnejšie dátové štruktúry a teda formáty uložených dát sú optimalizované tak, aby boli prístupy na disk čo najriedkejšie.

Depozitár (repository) obsahuje kompletný HTML kód každej zaindexovanej webovej stránky. Stránky sú uložené zkomprimované jedna za druhou, pričom pred každou je uložené jej docID, dĺžka a URL. K prístupu k repository nie sú nutné žiadne ďalšie dáta, takže v prípade potreby je možné z jeho kópie rekonštruovať všetky ostatné štruktúry.

V *indexe dokumentov* sú uložené informácie o každom dokumente – stránke. Používa štruktúru ISAM (Indexed Sequential Acess Method), v ktorej sú všetky záznamy pevnej dĺžky uložené sekvenčne za sebou, pričom hašovacie tabuľky pre urýchlenie prístupov podľa docID sú uložené mimo hlavné dáta. Každá položka obsahuje aktuálny stav dokumentu, ukazateľ do repository, kontrolný súčet súboru a rôzne štatistiky. Ak bol dokument rozparsovaný, je v položke tiež odkaz do súboru docinfo, v ktorom je zanesené URL dokumentu a jeho titulok. V opačnom prípade odkazuje do URL listu, ktorý obsahuje iba URL. Motiváciou tohto riešenia bola kompaktnosť dátovej štruktúry a schopnosť načítať záznam v jednom prístupe na disk. Taktiež existuje súbor pre prevod URL na docID – zotriedený zoznam hashov URL a odpovedajúcich docID. Pre prevod sa spočíta hash URL a v tabuľke sa vyhľadá príslušné docID. Pomocou zlievania je možné nájsť docID pre viacej URL naraz – túto techniku používa URL resolver.

Slovník je implementovaný ako sekvenčný zoznam slov oddelených nulami a hašovacou tabuľkou s ukazateľmi do nej. V prototype vyhľadávača slovník obsahoval 14 miliónov slov a zmestil sa do 256MB operačnej pamäte. Slovník je zdrojom problémov pri paralelizácii indexovania – pri bežnej prevádzke musia byť indexeri schopné pridávať do slovníku nové slová. Tento problém bol vyriešený jeho zafixovaním – nálezy nových slov sa zapisujú od zvláštneho súboru, ktorý na konci spracuje posledný indexer. Vzhľadom k veľkosti slovníku je tento dodatočný súbor relatívne malý a jeho spracovanie teda nezaberie veľa času.

Zoznamy hitov obsahujú údaje o výskyte slov v dokumentoch a ako už bolo spomenuté, vrátane informácií o pozícii, veľkosti písma a kapitalizácii. Pretože tvoria väčšinu obsahu indexu aj spätného indexu, je dôležité ich ukladať čo možno najefektívnejšie (Google pre uloženie hitu používa dva bajty). Hity sa rozlišujú na obyčajné a dôležité, pričom do dôležitých sa počítajú slová obsiahnuté v URL, titulku, texte odkazov a meta tagoch.

Vo vlastných indexoch sa pred zoznam hitov ukladá jeho dĺžka. Aby sa ušetrilo miesto, je dĺžka skombinovaná s wordID (resp. docID v spätnom indexe). Ak je hitov viacej, obsahuje pole pre dĺžku escape kód a samotná dĺžka je uložená v nasledujúcich dvoch bajtoch. Index je rozdelený do skupiny kontajnerov – *barelov,* pričom každý kontajner pokrýva určitú časť wordID. Ak dokument obsahuje slová, ktorých wordID spadajú do daného barelu, je do neho pridané jeho docID nasledované príslušnými wordID

a ich hitlistu. To síce vyžaduje o niečo viacej priestoru kvôli duplikovaným docID, ale zato významne zjednodušuje náročnosť úlohy pre triedič (sorter).

Spätný index sa skladá z rovnakých kontajnerov ako index normálny, kontajnery sú len pretriedené podľa wordID. Pre každé platné wordID je potom do slovníka doplnený odkaz do kontajneru s odpovedajúcim zoznamom dokumentov, ktoré slovo obsahujú. Dôležité je, v akom poradí je uvádzaný docID v tomto zozname. Jednoduchým riešením je radiť ich podľa docID – to umožňovalo ľahké zlučovanie zoznamov pri ich spracovaní viacslovných dotazov. Ďalšou možnosťou je radiť ich podľa PageRanku dokumentov. Potom sú odpovede na jednoslovné dotazy trivialitou a je pravdepodobné, že aj najlepšie odpovede na viacslovné dotazy budú blízko začiatku. Google používa kompromisné riešenie – má dve sady kontajnerov. „Krátka" obsahuje iba hity v titulkoch a odkazoch, „dlhá" je úplná. Pri prehľadávaní sa potom najprv preverí prvá sada a až pri príliš malom počte nájdených výsledkov sa skúma druhá.

Teraz sa už dostávame k samotnému vyhľadávaniu a vracaniu výsledkov. Ohodnocovanie výsledkov nezahŕňa iba PageRank, ale aj mnohé ďalšie kritériá a hodnotenie v Googli bolo navrhnuté tak, aby žiadny faktor nemal príliš veľký vplyv na výsledok. Ak je vyhodnocovaný jednoslovný dotaz, skúma sa zoznam hitov pre dané slovo. Google má pri každom hite uložený aj druh (titulok, text odkazu, URL, obyčajný text malým písmom,...) a každému druhu je potom priradená určitá váha. Na tieto váhy sa potom dá nazerať ako na vektor. Rovnako tak je ohodnotený počet hitov pre každý druh – ohodnotenie na začiatku rastie lineárne s počtom, ale potom sa závislosť „narovnáva", takže ak počet výskytov prekročí určitú medzu, ohodnotenie ďalej nerastie. Skalárny súčin vektoru váh a vektoru ohodnotení počtu výskytov tvorí ohodnotenie relevancie dokumentu, ktorého kombinácia s PageRankom určí konečné poradie dokumentu vo výsledku.

Pre viacslovné dotazy je situácia o niečo komplikovanejšia – musí sa prechádzať niekoľko zoznamov hitov naraz, aby bolo možné ohodnocovať výsledky na základe vzdialeností jednotlivých výskytov. Pre každú nájdenú skupinu hitov je spočítaná vzdialenosť výskytu všetkých nájdených slov v texte dokumentu alebo odkazu, ktorej priradené hodnotenie. Počty výskytov sa potom nepočítajú len pre rôzne druhy hitov, ale aj pre každú dvojicu druh - vzdialenosť. Obidva tieto údaje sú prevedené na príslušné ohodnotenia a ich skalárny súčin tvorí ohodnotenie relevancie dokumentu.

2.7.2 PageRank

Ako som už spomínal, je množstvo rôznych faktorov, ktoré ovplyvňujú hodnotenie a radenie stránok vo výsledkoch vyhľadávania Googlu. Predpokladá sa, že ich môže byť až 200, čo je podporené diskusiami kvalifikovaných odborníkov. PageRank, vzorček, ktorý vytvoril Larry Page, je iba jeden z nich, aj keď často považovaný za veľmi významný. Asi každí by dokázal na pár kritérií hodnotenia stránok prísť aj sám: výskyt slov dotazu na stránke (<title>, <h1>, v URL, v texte...), počet ich výskytov, vzdialenosť medzi jednotlivými slovami, reputácia stránky, „čerstvosť" stránky apod. Vyčerpávajúci a podľa môjho názoru aj pomerne kvalitný zoznam som objavil na webovej adrese: *http://www.vaughns-1-pagers.com/internet/google-ranking-factors.htm.*

Rád by som sa však na tomto mieste hlbšie zameral na pomerne zaujímavé kritériu a to práve na algoritmus PageRank. Algoritmus PageRank je iteratívny algoritmus na kalkuláciu ohodnotenia webových stránok založený na spätných odkazoch, ktoré na danú

stránku vedú. PageRank je teda číslo, ktoré algoritmus priradí každej web stránke, presnejšie každej URL, vyjadrujúce jej relevantnosť.

Za formulkou pre PageRank stojí v podstate jednoduchá úvaha:

- Ku každej webovej stránke existuje určitý počet iných webových stránok, ktoré na ňu odkazujú.
- Každá z týchto webových stránok má sama nejaký PageRank
- Odkaz zo stránky B na stránku A je ako pri bežnom hlasovaní považovaný za hlas od B pre A.
- Okrem toho PageRank stránky B dodá hlasu pre A určitú váhu. Čím vyšší je PageRank stránky B, tým lepšie.
- Ako ďalší faktor vstupuje do hry celkový počet všetkých dokumentov (stránok) na stránke B. Čím ich je menej, tým lepšie pre PageRank stránky A.
- Súčet všetkých PageRankov je rovný počtu dokumentov (stránok) na webe. PageRank teda reprezentuje významnosť dokumentu vo vnútri celého evidovaného Internetu.

Brin a Page danú myšlienku na počiatku matematicky zapísali približne takto [23]:

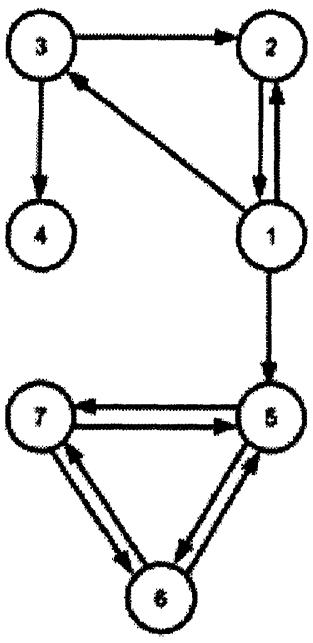
$$r(u) = \sum_{v \in B_u} \frac{r(v)}{N_v}$$

kde u značí stránku, ktorej PageRank $r(u)$ počítame, Bu je sada stránok odkazujúcich na u a N_v je celkový počet odkazov vychádzajúcich zo stránky v. Problém s danou rovnicou je v tom, že $r(v)$, PageRanky stránok odkazujúcich na stránku u sú na začiatku neznáme. Na vyriešenie tohto problému Brin a Page použili iteratívnu procedúru. Takže na začiatku majú všetky stránky rovnaký PageRank (napríklad $1/n$, kde n je celkový počet stránok v Google indexe webu). Teraz pri snahe spočítať $r(u)$ pre každú stránku môžeme použiť iteratívnu formulu:

$$r_{k+1}(u) = \sum_{v \in B_u} \frac{r_k(v)}{N_v}$$

Proces je inicializovaný hodnotou $r_0(u) = 1/n$ pre každú stránku u, $r_{k+1}(u)$ označuje hodnotu PageRanku stránky u v k+1 iterácii a celý proces je opakovaný s cieľom, aby hodnota PageRanku konvergovala k nejakej stálej hodnote. Aplikácia tejto rovnosti na malý web zobrazený na obrázku 4 dáva pre niekoľko iterácií hodnoty PageRanku zobrazené v tabuľke 1.

Knihovna Mat.-fyz. fakulty
Informatické oddělení
Malostranské náměstí 25
118 00 Praha 1



Obr. 4

Iterácia 0	*Iterácia 1*	*Iterácia 2*	*Rank v it. 2*
$r_0(u1) = 0.143$	$r_1(u1) = 0.143$	$r_2(u1) = 0.119$	4
$r_0(u2) = 0.143$	$r_1(u2) = 0.119$	$r_2(u2) = 0.063$	5
$r_0(u3) = 0.143$	$r_1(u3) = 0.048$	$r_2(u3) = 0.040$	6
$r_0(u4) = 0.143$	$r_1(u4) = 0.024$	$r_2(u4) = 0.019$	7
$r_0(u5) = 0.143$	$r_1(u5) = 0.190$	$r_2(u5) = 0.212$	1
$r_0(u6) = 0.143$	$r_1(u6) = 0.167$	$r_2(u6) = 0.195$	3
$r_0(u7) = 0.143$	$r_1(u7) = 0.179$	$r_2(u8) = 0.204$	2

Tabuľka 1

V spomenutých rovnostiach sa kalkulujú PageRanky jednej stránky za jednotku času. S použitím matíc sa však veľmi efektívne dá nahradiť $\sum$ a v každej iterácii počítať PageRank vektor rozmerov *1 x n* a udržiavať hodnoty PageRankov pre všetky stránky v indexe. S týmto súvisí zavedenie *n x n* matice hyperlinkov *H* a *1 x n* riadkového vektoru $\boldsymbol{\pi}^T$. Matica *H* má na indexe H_{ij} hodnotu $1/N_v$ ak existuje odkaz z vrcholu *i* do vrcholu *j* a 0 inak. *H* má tak pre graf na obrázku 4 nasledovný tvar:

$$\mathbf{H} = \begin{pmatrix} 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 \end{pmatrix}$$

Nenulové prvky riadku *i* sa viažu k stránkam, *na ktoré* odkazuje stránka *i*, kdežto prvky stĺpca *i* sa vzťahujú k stránkam, ktoré odkazujú *na* stránku *i*. V tejto chvíli zavádzame riadkový vektor $\boldsymbol{\pi}^{(k)T}$, ktorý je vektorom PageRankov v k-tej iterácii. S použitím tejto maticovej notácie, vyššie uvedená rovnosť (pre výpočet PR) môže byť kompaktnejšie vyjadrená ako

$$\boldsymbol{\pi}^{(k+1)T} = \boldsymbol{\pi}^{(k)T}\,\mathbf{H}.$$

Maticová rovnosť poskytuje viaceré pozorovania. Napríklad každá iterácia výpočtu zahŕňa jedno násobenie vektora s maticou, v konečnom dôsledku teda potrebujeme $O(n^2)$ výpočtov, kde *n* je veľkosť štvorcovej matice *H*. Ďalej si môžeme všimnúť, že matica *H* je riedka (veľký počet jej prvkov je nulových), pretože webové stánky odkazujú iba na obmedzený počet iných stránok. Tento fakt do značnej miery uľahčuje jej úschovu pre veľké *n*, pretože sa ukladá iba nenulová schéma a v konečnom dôsledku ani výpočtov nebude $O(n^2)$, ale iba *O(# nenulových prvkov H)*. *H* ďalej môžeme považovať za stochastickú pravdepodobnostú maticu prechodu pre Markovov reťazec [23]. Vrchol siete,

z ktorého nevychádzajú žiadne odkazy smerom k iným stránkam, vytvára nulové riadky v matici.

Iteratívny proces, ktorý som sa snažil predstaviť vyššie, však v sebe ukrýva nejednu otázku či dokonca problém, najmä spojené s konvergenciou celého procesu. Napríklad, vyskytuje sa tu tzv. „rank sink" problém – vrcholy, ktoré akumulujú stále viacej a viacej PageRanku v každej iterácii bez toho, aby PageRank zdieľali ďalej. V našom príklade podgraf *u5, u6, u7* vytvára „rank sink". Boli spomenuté už aj vrcholy, z ktorých nevychádzajú žiadne odkazy smerom k iným stránkam a majú za následok nulové riadky v matici *H*. Preto sa do základného modelu museli pridať určité úpravy, prispôsobenia.

Pri popise týchto prispôsobení Brin a Page použili model náhodného surfera. Uvažovali o PageRanku ako o modeli chovania, kde surfer náhodne kliká na odkazy bez ohľadu na obsah. Náhodný surfer teda navštevuje stránku s určitou pravdepodobnosťou, ktorá je odvodená z PageRanku. Pravdepodobnosť, že náhodný surfer klikne na odkaz je výhradne daná počtom odkazov na stránke. To je dôvod, prečo PageRank stránky nieje kompletne predaný ďalej na stránky na ktoré odkazuje, ale je rozdelený na počet odkazov na stránke. Avšak tento model obsahuje problém, že surfer je v pasci, akonáhle sa preklikne na nejaký vrchol so žiadnymi vychádzajúcimi odkazmi. A na webe je takýchto vrcholov dosť – pdf súbory, obrázky, tabuľky... Ako riešenie bolo navrhnuté nahradiť riadkové vektory $\boldsymbol{0}^T$ v matici *H* s $1/n\boldsymbol{e}^T$ kde $\boldsymbol{e}^T$ je riadkový vektor pozostávajúci zo samých jednotiek. Výsledkom je, že náhodný surfer po navštívení stránky, z ktorej nevedie žiaden odkaz von, sa preklikne náhodne na akúkoľvek inú stránku. Tým sa mení substochastická matica *H* na stochastickú maticu *S*, ktorá by v našom prípade vyzerala asi takto:

$$\mathbf{S} = \begin{pmatrix} 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 \end{pmatrix}$$

Túto úpravu môžeme matematicky zapísať $\mathbf{S} = \mathbf{H} + \mathbf{a}(1/n\mathbf{e}^T)$, kde $a_i = 1$ ak stránka *i* je vrchol so žiadnymi doprednými odkazmi, 0 inak. *S* je teda kombináciou originálnej matice hyperlinkov *H* a matice s hodnosťou $1 - 1/n\boldsymbol{a}\boldsymbol{e}^T$.

Avšak ani toto vylepšenie nám samo o sebe negarantuje požadovanú konvergenciu výsledkov. Preto existuje ďalšie, ktoré vytvorí maticu stochastickú a navyše primitívnu – teda ireducibilnú a aperiodickú. Princíp je nasledovný. Surfer niekedy nesleduje hyperlinkovú štruktúru webu, ale vloží nejakú novú URL destináciou do vyhľadávača. Keď sa to stane, náhodný surfer sa „teleportuje" do novej destinácii, kde sa opäť preklikáva cez hyperlinkovú štruktúru novej stránky až do ďalšej teleportácie atď. Na modelovanie tejto aktivity matematicky, Brin a Page vymysleli novú maticu **G**,

$$\mathbf{G} = \alpha\mathbf{S} + (1-\alpha)1/n\mathbf{e}\mathbf{e}^T$$

kde α je skalár medzi 0 a 1. *G* sa nazýva Google matica a parameter α, občas nazývaný aj faktor útlmu, je parameter, ktorý kontroluje veľkosť času, ktorý náhodný surfer strávi nasledovaním hyperlinkov oproti teleportácii na novú stránku. V prípade, že α = 0.6, znamená to, že 60% času náhodný surfer nasleduje hyperlinkovú štruktúru webu a zostávajúcich 40% času sa teleportuje na novú stránku. Teleportácia je náhodná, pretože matica teleportácie $\mathbf{E} = 1/n\mathbf{e}\mathbf{e}^T$ je uniformná, čo znamená, že je rovnako pravdepodobné, že surfer pri teleportácii skočí na ľubovoľnú stránku. Google iniciálne nastavuje α = 0.85 – aspoň to tak bolo v dobe, keď Brin a Page začínali s prevádzkou Googlu.

O matici G možno taktiež urobiť rozmanité pozorovania – G je stochastická, ireducibilná, aperiodická, primitívna a hustá. Dá sa vyjadriť aj pomocou matice *H:*

$$\mathbf{G} = \alpha\mathbf{S} + (1-\alpha)1/n\mathbf{e}\mathbf{e}^T = \alpha(\mathbf{H} + 1/n\mathbf{a}\mathbf{e}^T) + (1-\alpha)1/n\mathbf{e}\mathbf{e}^T = \alpha\mathbf{H} + (\alpha\mathbf{a} + (1-\alpha)\mathbf{e})1/n\mathbf{e}^T .$$

Konečne sa dostávame k finálnemu výsledku – Google počíta po všetkých úpravách PageRank metódou:

$$\pi^{(k+1)T} = \pi^{(k)T}\mathbf{G}.$$

V našom príklade webu z obrázku 4 by matica *G* vyzerala takto:

$$\mathbf{G} = \begin{pmatrix} 0.021 & 0.305 & 0.305 & 0.021 & 0.305 & 0.021 & 0.021 \\ 0.871 & 0.021 & 0.021 & 0.021 & 0.021 & 0.021 & 0.021 \\ 0.021 & 0.446 & 0.021 & 0.446 & 0.021 & 0.021 & 0.021 \\ 0.143 & 0.143 & 0.143 & 0.143 & 0.143 & 0.143 & 0.143 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.021 & 0.446 & 0.446 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.446 & 0.021 & 0.446 \\ 0.021 & 0.021 & 0.021 & 0.021 & 0.446 & 0.446 & 0.021 \end{pmatrix}$$

a prislúchajúci PageRank vektor :

$$\pi^T = (0.093, 0.077, 0.054, 0.050, 0.254, 0.236, 0.236)$$

čo znamená, že stránky v našom mini-webe by podľa relevantnosti boli zoradené (u5 u6 u7 u1 u2 u3 u4), teda stránka *u5* je podľa PageRank definície dôležitosti najdôležitejšia a *u4* najmenej dôležitá. Na podobné výpočty je veľmi šikovný program Matlab, ktorý som použil aj ja pri svojich skúšobných výpočtoch. Pre zaujímavosť, Cleve Moler, zakladateľ Matlabu, v roku 2002 prehlásil PageRank za „Svetovo najväčšieho kalkulátora matíc". Vtedy Google používal svoju metódu pre riedku maticu rádu 2.7 miliardy, dnes je tento rád vyše 8.7 miliardy. A ak nás zaujíma exaktná podoba dnešného výpočtu PageRanku, čaká nás sklamanie – Google ju nezverejňuje a pravdepodobne tak skoro ani nezverejní.

Rád by som spomenul aj kocept modelu „inteligentného" surfera, ako alternatívu, alebo skôr vylepšenie k spomenutému „náhodnému surferovi" [25]. V tomto modeli Matthewa Richardsona a Pedra Dominga už surfer nekliká na odkazy náhodne, ale sleduje iba odkazy, ktoré majú niečo spoločné s pôvodnou témou stránky. Pre „inteligentného" surfera sú teda relevantné iba stránky, ktoré obsahujú slovo, ktoré pôvodne hľadal. To však znamená, že pre každé slovo, ktoré sa na stránke objaví, je potrebé spočítať vlastný PageRank založený na odkazoch medzi stránkami, ktoré dané slovo obsahujú.

Ak by sa PageRank počítal týmto spôsobom, prinieslo by to rad problémov. A to najmä v prípadoch, kedy sa vyhľadávané slovo nevyskytuje často. Preto, aby sa dalo

špecifické slovo zahrnúť do výpočtu, musí byť nielen na danej stránke, ale tiež na stránkach, ktoré na ňu odkazujú. To ale znamená, že výsledky budú založené na malom výseku webu a možu byť opomenuté relevantné stránky.

Problém je aj vo vypočítavaní takého PageRanku. Ak by sme uvažovali 100 000 výrazov, bol by potrebný čas pre vypočítanie tématicky zameraného PageRanku oproti originálnemu 100 až 200 násobne väčší. Teda ak sa pôvodný PageRank počíta 5 hodín, tematický by sa počítal najmenej 3 týždne.

2.7.3 Vplyv IP adresy na výsledky vyhľadávania

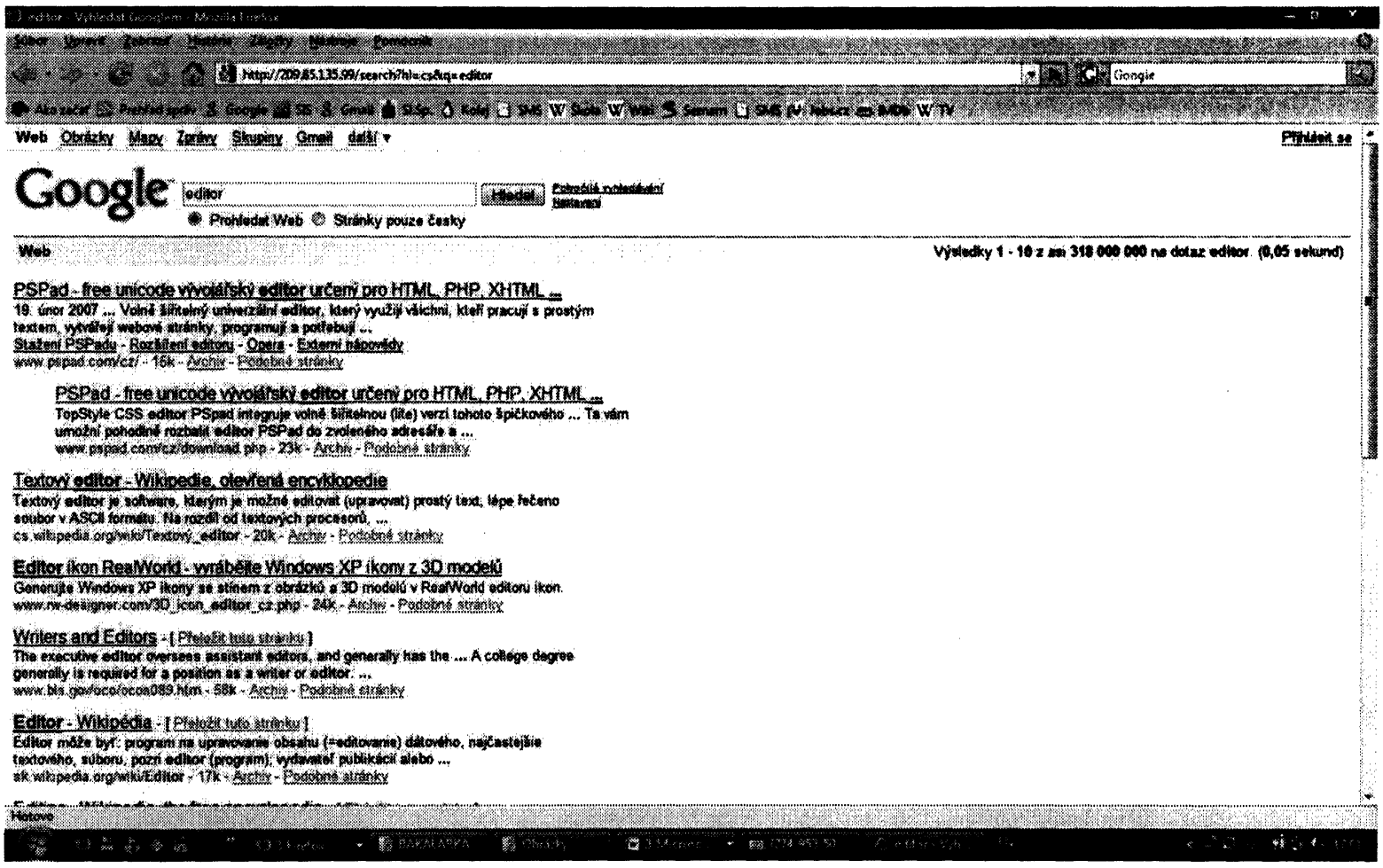
V mnohých diskusiách sa objavujú dohady o kritériách, podľa ktorých Google predkladá výsledky vyhľadávania. Koniec koncov o jednom z najdôležitejších, PageRanku, som sa už rozpísal vyššie. Ale aj ďalšie sú veľmi zaujímavé a keďže súvisia s charakterom mojej práce, pokúsil som sa urobiť malý výskum na túto tému. Zaujímať ma bude najmä IP adresa, z ktorej sa vyhľadávací dotaz posiela serveru Googlu. Zatiaľ čo simulovanie vyhľadávania s rôzne nastaveným preferovaným jazykom v prehliadači (?hl=cs, ?hl=sk) je pomerne jednoduché, vyhľadávanie ovplyvnené IP adresou už také triviálne nie je z dôvodu ťažšieho nastavenia proxy, cez ktorú vyhľadávanie vo vyhľadávači realizujeme a predovšetkým preto, že funkčné proxy servery sa relatívne horšie hľadajú a priraďujú ku konkrétnej zemi.

Pri diskusiách o tom, či vyhľadávač Google prekladá výsledky podľa IP adresy, alebo nie, sa často argumentuje tým, že výsledky sú ovplyvnené geografickou polohou dotazujúceho a že výsledok vyhľadávania je ovplyvnený tým, či dotaz prišiel napríklad z Českej republiky, alebo Belgicka. Táto teória sa zdá byť potvrdená v prípade, že jednoducho do vyhľadávača zadáme adresu *www.google.com* a bez akýchkoľvek špecifikácií začneme klásť vyhľadávacie dotazy.

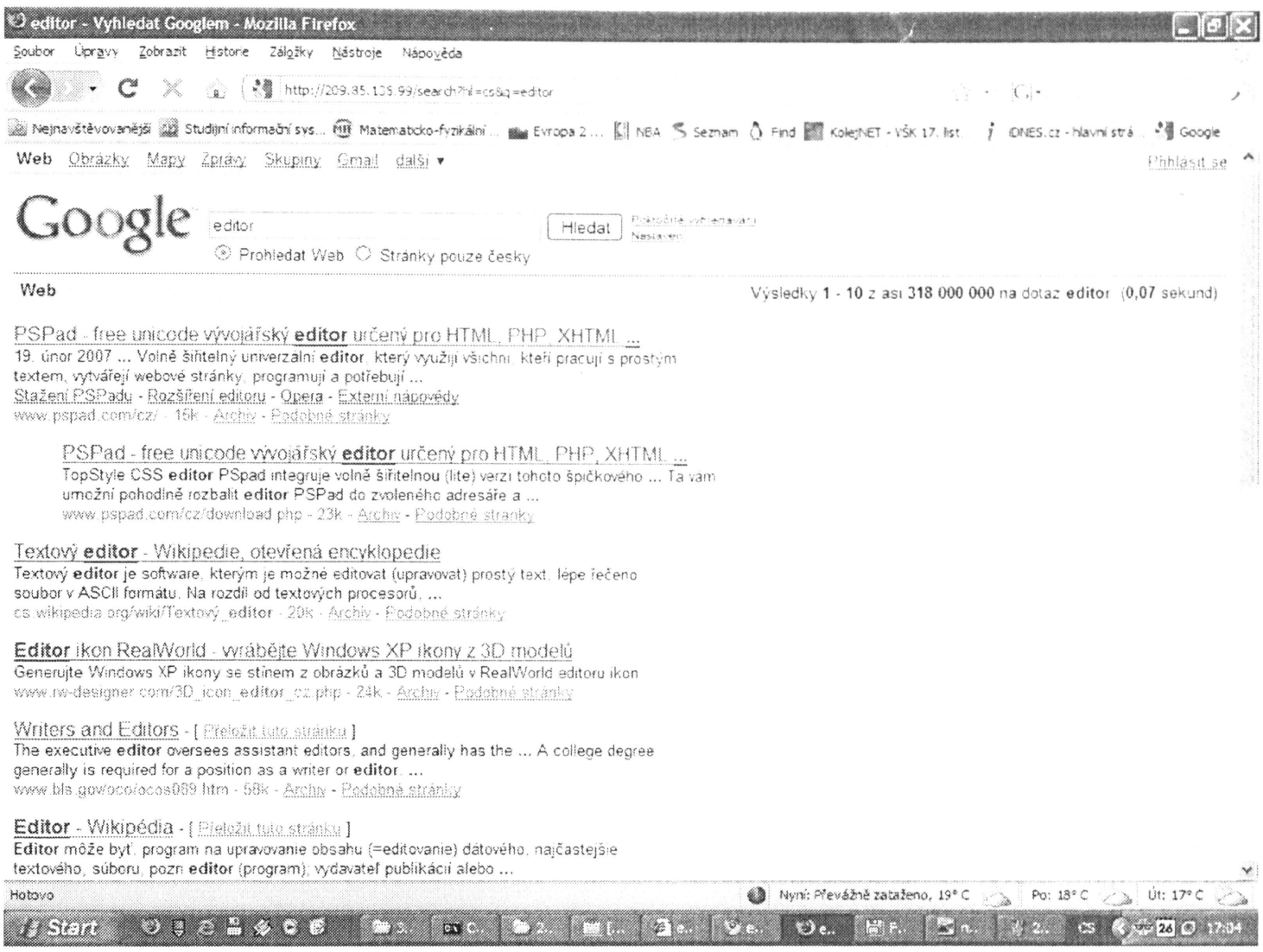
Na overenie faktu, či IP adresa počítača a geografická poloha, z ktorej odchádza dotaz na server Googlu má skutočne vplyv na vrátené výsledky som sa rozhodol urobiť sadu testov. Pri skúmaní jednotlivých faktorov, ktoré by mohli mať v tejto súvislosti na dané výsledky vplyv mi ako zaujímavé a podstatné vyšli faktory dva – datacentrum, na ktoré sa Google pripája a jazyk rozhrania, v ktorom sa zobrazuje stránka Googlu. Je zrejmé, že nastavenie jazyka, v ktorom sa má vyhľadávanie previesť, má značný vplyv na výsledky, takže ďalej som tomuto faktoru nevenoval väčšiu pozornosť. Rozhodol som sa zamerať na inú vec – aby som mohol overiť, či IP adresa má na vrátenie výsledkov skutočne relevantný vplyv a je zahrnutá do rankovacieho modulu Googlu pri zachovaní ostatných kritérií, musíme sa pripájať na to isté datacentrum Googlu, s nastaveným tým istým jazykom vyhľadávania, identickým jazykom rozhrania Googlu a prirodzene použitím rovnakého vyhľadávača.

Testy boli postupne prevedené z lokalít Praha - Brno, Praha - Zvolen a Praha - Košice s odstupom niekoľkých dní až týždňov, ale každá dvojica prirodzene v približne rovnaký čas (odchýlka v ráde desiatok sekúnd). Všetky vyhľadávania boli prevedené priamym vložením URL do adresného riadku prehliadača. Dotaz (?q=editor) bol kladený vždy datacentru s identickou IP adresou (209.85.135.147 resp. 209.85.135.99 - náhodne vybrané datacentrá), jazyk rozhrania bol nastavený na češtinu (?hl=cs), jazyk hľadania výsledkov nebol nijak obmedzený a použitý bol vyhľadávač MS Internet Explorer alebo Mozzila Firefox v defaultnom nastavení.

Dňa 14.4.2009 som sa nachádzal na Slovensku, konkrétne vo Zvolene, teda dotaz Googlu som položil z tejto lokality. Stojí za to spomenúť, že pripojenie k Internetu bolo realizované modemom a teda IP adresa bola počítaču priradená poskytovateľom dynamicky. Súčasne bol rovnaký dotaz `editor` položený aj z Prahy. Výsledky som porovnal. Za všetky testy by som rád ukázal aspoň dva obrázky z merania uskutočneného 26.4.2009 približne o 17:00. V tento moment bol použitý vyhľadávač Firefox pre obidve merania, jedno bolo prevedené z Košíc, druhé z Prahy. Stabilne sa obidve stanice, resp. meracie miesta pripájali na datacentrum na IP adrese 209.85.130.99, ktoré sme priamo predurčili. Ako vidno z obrázkov 5 a 6, vrátené výsledky sú totožné (resp. na obrázku je vidieť prvých niekoľko výsledkov) a počet nájdených výsledkov je 318 000 v obidvoch prípadoch. Prvý obrázok ukazuje výsledky dostupné v Košiciach, druhý výsledky z Prahy.



Obr. 5 Výsledky na dotaz editor z Košíc



Obr. 6 Výsledky na dotaz editor z Prahy

Ďalšou záležitosťou, na ktorú som za zameral a ktorá ako vidno z obrázku 7 má vplyv na výsledky, je, či sa pripájame na stránky *google.cz*, *google.ca*, *google.bi*, teda rôzne jazykové verzie stránok. Aj keď sú ostatné nastavenia identické, výsledky vyhľadávania prevedené z týchto stránok sú rôzne. Ďalšou zaujímavosťou, ktorá potvrdzuje teóriu o vplyve datacentra na vrátené výsledky je, že v ten istý moment som položil dotaz rôznym datacentrám z toho istého počítača a v aspoň jednom prípade sa niektoré vrátené výsledky odlišovali od ostatných. Hoci rozdiely neboli na prvý pohľad príliš výrazné, zmena pozície niektorej stránky vo výsledkoch či úplne nová stránka v prvej desiatke to jasne potvrdzovali.

Ďalšia testovacia séria meraní prebiehala štýlom „obnovenia stránky", kedy sme v náhodných intervaloch prevádzali refresh stránky s výsledkami a porovnávali sme ich s predchádzajúcim stavom. V tomto prípade sme však neovplyvňovali, do ktorého datacentra sa budeme dotazovať. Priebežne sme kontrolovali pingom, ktoré datacentrá sa nám hlásia. Vždy, keď došlo k zmene výsledkov, bola zistená aj zmena datacentra, ktoré odpovedalo adrese *www.google.com*.

Obr. 7. Rôzne výsledky vyhľadávania

Môj záver z výskumu je nasledovný. V sade testov, ktoré boli urobené z rôznych miest v Česku a na Slovensku vyplýva, že pri použití identického predurčeného datacentra (IP adresa 209.85.135.147 resp. 209.85.135.99), na ktoré sa stanice pripájali a identických jazykových nastaveniach, boli vždy páry výsledkov vrátené v rovnaké dni a rovnaké časy ekvivalentné, hoci dotazy pochádzali z miest s rôznou geografickou polohou a rôznou IP adresou (Praha – Košice, Praha – Brno a Praha – Zvolen).

Geografická poloha a IP adresa však zohráva úlohu pri výbere datacentra, na ktoré sa Google pripája a výbere mutácie jazykovej verzie stránky (google.cz, google.sk). Tým pádom sa výsledky môžu diametrálne líšiť. Navyše ak je nastavený aj rôzny jazyk rozhrania, na vrátené výsledkov to má signifikantný vplyv. Použitie rôznych vyhľadávačov, konkrétne MS Internet Exlorer, Firefox, poprípade Google Chrome sa neprejavil na celkových výsledkoch vracaných Googlom, avšak nie je možné to vylúčiť.

Pochopiteľne závery, ktoré som urobil, sú založené na malom počte pozorovaní a ako formálny dôkaz by moja obhajoba výsledkov neuspela, avšak na základe prezentovaných meraní nemožno túto teóriu ani jednoznačne vyvrátiť.

Prevedené merania však transparentne preukázali, že výsledky hľadania prostredníctvom Google sú závislé na tom, ktoré datacentrum nám odpovedá. Súčasne bolo preukázané, že poloha a teda aj naša IP adresa značne ovplyvňujú, ktoré datacentrum nám bude pridelené a teda aké výsledky vyhľadávania nám budú distribuované. Ďalším poznatkom môže byť, že Google za poslednú dobu zdá sa vytriedil výsledky a ponúka ich v niektorých prípadoch možno menej, zato o to viac relevantné.

3 HACKING GOOGLOM

Na to, aby bola obrana proti Google hackerom pátrajúcich po rôznych citlivých informáciách účinná, musíme poznať zbrane ich arzenálu. V tejto kapitole sa pokúsim priblížiť vybrané techniky a metódy Google hackingu, ich motiváciu a tiež spôsob hľadania a využitia určitého druhu informácií. Na začiatok uvádzam tabuľku s prehľadom jednotlivých techník (tabuľka 2), popisu a využitím najzaujímavejších z nich sa budem ďalej venovať.

Technika	Podrobnejšie informácie
Využitie Google archívu	anonymita
Výpisy adresárov a pátranie po súboroch	konfiguračné súbory systémové logy
Získavanie informácií o sieťach a systémoch – profil webového serveru	pomocou výpisov adresárov hlásenia o chybách prihlasovacie portály štandardné stránky sieťové zariadenia webové utility
Databázy a Google	hlásenia o chybách prihlasovacie portály
Pátranie po užívateľských menách, heslách, osobných dátach a dôverných informáciách	
Lokalizácia exploitov a hľadanie cieľov	
Google a automatizovaný zber informácií	automatizované získavanie dát zo zdroja parsovanie, obrusovanie a ďalšie spracovanie získaných dát
Google služby a ich možnosti	AJAX Search API Google Kalendár Google vlastný vyhľadávač
Získavanie „inak“ zaujímavých dát	video, hudba, software...

Tabuľka 2. Techniky Google hackingu

3.1 Anonymita s Google archívom

Povaha aktivity hackerov zväčša vyžaduje určitý stupeň anonymity, preto sa neraz stáva ich prioritou číslo jedna. Obecne všetka komunikácia medzi našim systémom a vzdialeným serverom, či už sa jedná o prezeranie stránok alebo hľadanie citlivých dokumentov, môže byť zaznamenávaná. K účelu zabezpečenia anonymity pri prezeraní webovej stránky cieľa, na ktorý sa útočník zameriava, sa využívajú najmä proxy serveri, avšak do určitej miery k tomu môže dobre poslúžiť aj Google.

Google oplýva z tohto pohľadu veľmi užitočnou vlastnosťou – archívom (cache). Ako náhle prejde nejakú stránku alebo dokument, skoro vždy bude k dispozícii kópia tejto stránky alebo dokumentu. Samozrejme nevýhodou tohto prístupu je fakt, že sa ku kópii môže dostať hypotetický útočník, aj keď pôvodný zdroj už dávno neexistuje. Ďalšou tienistou stránkou archívu je to, že hackeri môžu prehľadať celý náš web bez toho, aby odoslali čo len jediný paket na náš webový server. Následkom toho je, že server nemôže nič zapísať do súborov protokolov (logov) a teda prípadné odcudzenie citlivých dát nebude zaznamenané.

Googlom archivované stránky sú prístupné cez odkaz *Archív (cache)* na stránke výsledkov vyhľadávania, alebo ich môžeme vyhľadať pomocou pokročilého operátora *cache*. Pri prezeraní archivovanej verzie stránky sa zobrazí akési záhlavie - úvodný text stránky Google pre archivovaný dokument. Tu stojí za povšimnutie upozornenie, že archivovaná stránka sa môže odkazovať na obrázky, ktoré už nie sú dostupné. To okrem iného znamená, že ak si prezeráme nejakú archívnu kópiu webovej stránky, začneme vlastne sťahovať obrázky priamo zo samotného serveru, na ktorom je umiestená pôvodná stránka. Ak sme sa pokúšali o anonymitu tým, že prezeráme archívnu kópiu Googlu, nielenže sme ju nedosiahli, ale dokonca náš prehliadač informoval webový server o tom, že sa pokúšame prezerať archivovanú verziu stránky. Existuje však spôsob, ako načítať pôvodnú stránku bez toho, aby sme komunikovali s externým serverom a aby naša komunikácia prebiehala iba so serverom Googlu – pripojením parametra *&strip=1* na koniec URL adresy, ktorá odkazuje na archivovanú verziu. Tým zaistíme iba zobrazenie archivovaného textu bez všetkých externých odkazov, podobne ako po kliknutí na odkaz *„archivovaný text“* v úvodnom texte archivovanej verzie stránky.

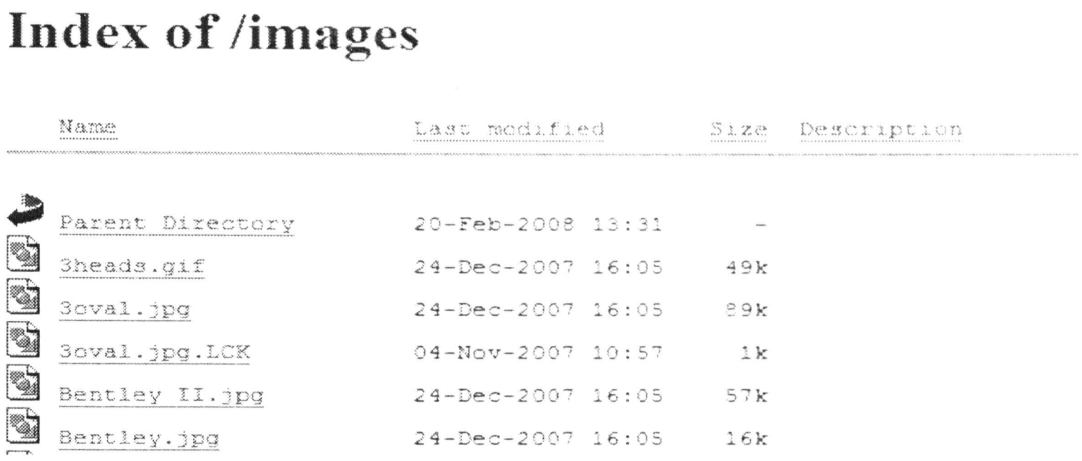
Ak by sme na prezeranie testovanej stránky použili anonymný proxy server, dostal by webový server iba IP adresu tohto proxy servera a nie našu skutočnú IP adresu, pretože proxy server predstavuje akúsi medzi stanicu pri putovaní dát, ktoré užívateľ odosiela a príjma. Pri pátraní po proxy serveroch vie byť Google opäť nápomocný. Dotazom

```
inurl:"nph-proxy.cgi" "Start browsing through this CGI-based proxy"
```

je možné naraziť na pár použiteľných CGI proxy serverov, ktoré fungujú na webovom serveri, navonok sa chovajú ako klasický HTTP proxy a dobre poslúžia ako čiastočná ochrana. Defaultná stránka CGI proxy býva zväčša uložená v súbore s názvom *nph-proxy* a obsahuje text *„Start browsing through this CGI-based proxy by entering a URL below“*. Tieto informácie viedli k zostaveniu vyššie uvedeného dotazu a zistenie podobných informácií býva zárodkom aj ďalších hľadaní podobného charakteru.

3.2 Výpisy adresárov a pátranie po súboroch

Výpis adresára je určitý druh stránky, ktorá obsahuje zoznam súborov a adresárov nachádzajúcich sa na danom serveri. Na takýchto stránkach je možné prechádzať z adresára do adresára jednoduchým kliknutím na ich odkaz a tiež sťahovať a prezerať súbory v nich obsiahnuté.



Index of /images

Name	Last modified	Size	Description
Parent Directory	20-Feb-2008 13:31	-	
3heads.gif	24-Dec-2007 16:05	49k	
3oval.jpg	24-Dec-2007 16:05	89k	
3oval.jpg.LCK	04-Nov-2007 10:57	1k	
Bentley II.jpg	24-Dec-2007 16:05	57k	
Bentley.jpg	24-Dec-2007 16:05	16k	

Obr. 8. Ukážka stránky s výpisom adresára

Trpia však rôznymi nedostatkami. V prvom rade nie sú bezpečné. Nijak nebránia užívateľovi v sťahovaní súborov, alebo v prístupe k iným adresárom. Typicky obsahujú titulok a niektoré pätičku (príklad vidíme na obrázku 9), v ktorej sa zobrazuje verzia webového servera, čo je pre útočníka cenná informácia ak sa chce dozvedieť nejaké detaily o web serveri pri vytváraní jeho profilu . Tiež sa často zobrazujú omylom, keď chýba alebo je neplatný úvodný súbor stránky (*index.html* a pod.)

uc anim22.gif	24-Dec-2007 16:12	25k
uc anim22.gif.LCK	29-Oct-2007 19:05	1k

Apache/1.3.37 Server at www.internetplayerscafe.com Port 80

Obr. 9. Pätička s verziou webového serveru

Vypátrať pomocou Googlu výpisy adresárov nie je vôbec zložité, väčšina takýchto stránok začína reťazcom „Index of", ktorý sa nachádza aj v ich titulkoch. Dotaz `intitle:"Index of"` však vráti aj veľké množstvo stránok, ktoré sú pre nás irelevantné. V tejto chvíli príde na rad zužovanie výsledkov a naskytne sa priestor na variácie dotazov. Už dotazy ako

```
intitle:"index of" "parent directory"
```

```
intitle:"index of" +name +size
```

nám zabezpečia oveľa rozumnejšie výsledky.

Pri pátraní po konkrétnom adresári môžeme dotaz ľubovoľne modifikovať a skúšať, či vráti relevantné výsledky (napr.: `intitle:Index.of.admin)`

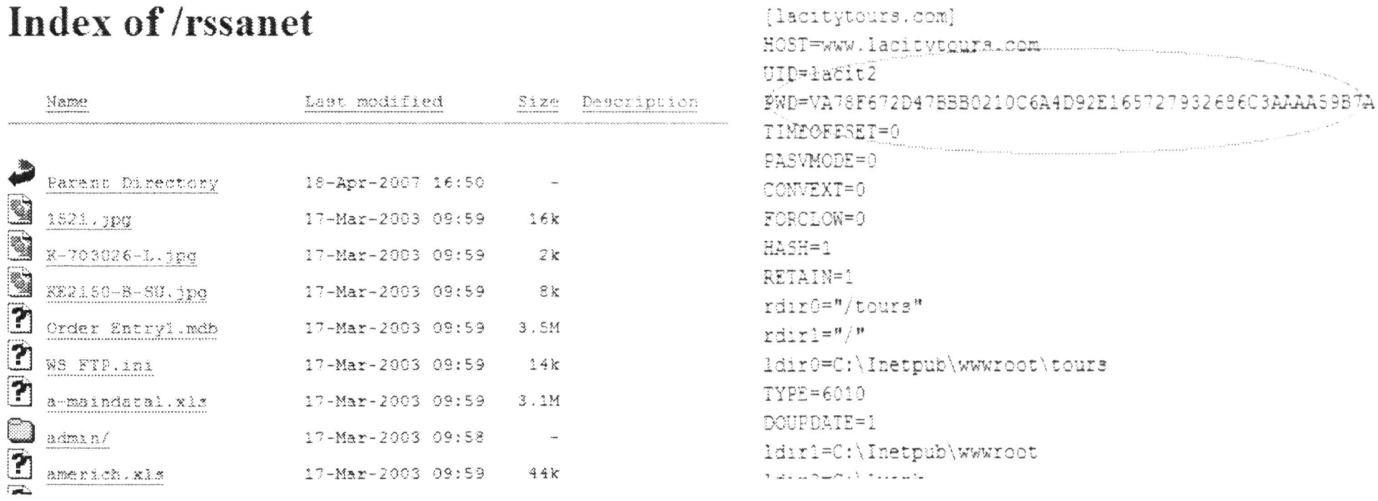
Vo výpisoch adresárov je tiež možné nájsť konkrétne súbory a to tak, že budeme hľadať „index of" v titulku a názov súboru v texte webovej stránky. Zoberme si ako príklad FTP klient WS_FTP, ktorý tak ako väčšina užívateľských programov umožňuje zapamätať

si heslá pre účty. Svoju konfiguráciu a informácie o užívateľských účtoch ukladá do súboru *ws_ftp.ini* a každý, kto získa prístup ku konfigurácii FTP klienta má prístup aj ku jeho zdrojom. Heslá ukladané v súbore *ws_ftp.ini* sú síce šifrované, ale ak má útočník k dispozícii konfiguračný súbor, môže použiť nástroje pre dešifrovanie hesla, alebo jednoducho nainštalovať program WS_FTP a spustiť ho s nájdenou konfiguráciou. K množstvu konfiguračných súborov klienta WS_FTP sa môžeme dopátrať napríklad pomocou dotazov:

```
intitle:"index of" "parent directory" "ws_ftp.ini"
```

```
filetype:ini ws_ftp pwd
```

Výsledky takéhoto snaženia sú zobrazené na obrázku 10, ktorý som získal približne v marci 2008 a je na ňom zobrazený ako výpis adresára so súborom *ws_ftp.ini*, tak aj obsah tohto súboru so zvýrazneným zašifrovaným heslom. Výpisy adresárov však nie sú zase až také bežné, ale dokážu poskytnúť priaznivé výsledky a dajú sa relatívne ľahko vypátrať.



Index of /rssanet

Name	Last modified	Size	Description
Parent Directory	18-Apr-2007 16:50	-	
1521.jpg	17-Mar-2003 09:59	16k	
R-703026-L.jpg	17-Mar-2003 09:59	2k	
XE2150-B-SU.jpg	17-Mar-2003 09:59	8k	
Order Entry1.mdb	17-Mar-2003 09:59	3.5M	
WS_FTP.ini	17-Mar-2003 09:59	14k	
a-maindata1.xls	17-Mar-2003 09:59	3.1M	
admin/	17-Mar-2003 09:58	-	
americh.xls	17-Mar-2003 09:59	44k	

```
[lacitytours.com]
HOST=www.lacitytours.com
UID=lacit2
PWD=VA78F672D47BBB0210C6A4D92E165727932696C3AAAA59B7A
TIMEOFFSET=0
PASVMODE=0
CONVEXT=0
FORCLOW=0
HASH=1
RETAIN=1
rdir0="/tours"
rdir1="/"
ldir0=C:\Inetpub\wwwroot\tours
TYPE=6010
DOUPDATE=1
ldir1=C:\Inetpub\wwwroot
```

Obr. 10. Konfiguračný súbor programu WS_FTP s vyznačeným zašifrovaným heslom

3.3 Informácie o sieťach a systémoch

Účinnému útoku na počítačový systém zväčša predchádza rozpoznanie cieľa, vytvorenie jeho akéhosi profilu. Tento proces zahŕňa okrem iného skenovanie počítačov, otestovanie fungujúcich služieb, typu operačného systému a verzie služieb programu. K tomuto účelu sa obvykle používa nejaký skener typu *Nmap*, no existuje ešte iná možnosť. Správcovia systémov občas inštalujú www servery, ktoré za behu generujú štatistiky o práci serveru, obsahujú zoznamy spustených procesov, alebo aj systémové záznamy (logy) [8]. Ak sa Googlu útočník spýta na napríklad na štatistiky programu *phpSystem* (alebo rôznych iných programov) dotazom

```
"Generated by phpSystem"
```

dostane sa mu do rúk množstvo podrobností o systéme. Ďalšie možnosti dotazov na štatistiky a informácie vytvárané populárnymi programami by mohli vyzerať podobne, príklady sú uvedené v tabuľke 3 [8].

Dotaz	*Typ informácií*
`"Generated by phpSystem"`	typ a verzia operačného systému, hrdwarová konfigurácia, prihlásení užívatelia, otvorené spojenia...
`"This summary was generated by wwwstat"`	štatistiky práce WWW serveru
`intitle:"Statistics of" "advanced web statistics"`	štatistiky práce WWW serveru, informácie o návštevníkoch
`intitle:"Multimon UPS status page"`	štatistiky práce UPS zariadení
`intitle:"Apache::Status" (inurl:server-status | inurl:status.html | inurl:apache.html)`	verzia serveru, typ operačného systému, zoznam procesov

Tabuľka 3. Programy vytvárajúce štatistiky o fungovaní systému

Predstavme si ďalej situáciu, že v nejakom bežne používanom programe sa vyskytne bezpečnostná diera. Hypotetický útočník chce nájsť nejaké počítače s týmto programom, aby sa mohol pokúsiť na ne zaútočiť (napríklad má k dispozícii účinný exploit). Keby sa spomínaná bezpečnostná diera týkala napríklad serveru Microsoft IIS verzie 5.0, výsledkom dotazu

`"Microsoft-IIS/5.0 Server at" intitle:index.of`

by boli odkazy na ***hľadané servery***.

Príklady ďalších dotazov na iné verzie serverov demonštruje tabuľka 4 [8]. Stojí za povšimnutie, že informácie takéhoto druhu väčšinou prezrádzajú stránky s výpisom adresárov, preto aj pátranie po nich využíva túto techniku. Dôvodom úspešného hľadania je fakt, že niektoré servery pridávajú do určitých dynamicky generovaných stránok titulky obsahujúce svoje meno a verziu.

Dotaz	**Server**
`„Apache/2.0 Server at" intitle:index.of`	Apache 2.0
`"Apache/* Server at" intitle:index.of`	ľubovoľná verzia Apache
`"Microsoft-IIS/* Server at" intitle:index.of`	ľubovoľná verzia Microsoft Internet Information Services
`"HP Apache-based Web Server/*" intitle:index.of`	ľubovoľná verzia serveru HP

Tabuľka 4. Dotazy na rôzne typy www serverov

Pri vytváraní profilu webového serveru sú pre útočníka nápomocné aj ***prihlasovacie portály*** umožňujúce prístup ku konkrétnym schopnostiam alebo funkciám webu až po tom, čo sa užívateľ prihlási. Ak má útočník po ruke účinný exploit proti konkrétnemu softwaru a tento software prevádzkuje prihlasovací portál, útočník je schopný pomocou vhodných dotazov Googlu vypátrať potenciálne vhodné ciele. Niekoľko dotazov pátrajúcich po prihlasovacích portáloch je zhrnutých v tabuľke 5.

Dotaz	Prihlasovací portál
`inurl:"login.asp"`	Užívateľ všeobecne
`inurl:"admin/login.asp"`	Administrátor všeobecne
`intitle:novell intitle:webaccess "copyright * - * novell"`	Novell Groupwise
`inurl:"exchange/logon.asp"`	MS Outlook Web Access

Tabuľka 5. Dotazy pátrajúce po prihlasovacích portáloch

3.4 Databázy a Google

V poslednej dobe sa intenzívne sústreďuje pozornosť na bezpečnosť webových databázových aplikácií, najmä na software komunikácie s databázou. Dokladajú to aj stále častejšie diskusie o téme injektáže SQL, alebo nedávny masový útok na Microsoft SQL databázy (apríl 2008). Útočník však obvykle nepoužije Google k tomu, aby sa vlámal do databázy alebo aby poškodil nejakú aplikáciu pracujúcu s databázou – hackeri Googlu skôr skladajú útržky informácií, ktoré unikli z potenciálne zraniteľných webov a použijú ich k výberu vhodného cieľa a neskoršiemu premyslenému útoku na daný cieľ.

O stránkach zvaných prihlasovacie portály som sa už v krátkosti zmienil v predchádzajúcej kapitole. Ich potenciál poskytovať cenné informácie je značný aj v prípade, že sa jedná o databázové prihlasovacie portály. Bez ohľadu na silu zabezpečenia, už len existencia prihlasovacieho portálu sprostredkováva útočníkovi letmý pohľad na typ software a hardware, na ktorý by sa dalo zamerať.

Pri tvorbe profilov všetkého druhu ako aj vo fáze zhromažďovania informácií sú využívané aj ***hlásenia o chybách***. Je z nich možné získať dáta o systéme, konfigurácii a štruktúre databázy. Opäť uvádzam pre ilustráciu niekoľko dotazov, nájdeme ich v tabuľke 6 [9].

Dotaz	Popis
`intitle:"Execution of this script not permitted"`	„Cgiwrap“ chybové hlásenie odhaľuje administrátorský e-mail, porty..
`intitle:"htsearch error" ht://Dig error`	Môže odhaliť administrátorský e-mail, adresárovú štruktúru..
`"Warning: mysql_connect(): Access denied for user: '*@*" "on line" - help -forum`	Odhaľuje prihlásenia k databáze, ktoré boli kôli nejakej príčine odmietnuté
`"access denied for user" "using password"`	SQL chybová správa môže odhaliť názvy súborov, cesty, mená funkcií...

Tabuľka 6. Dotazy pátrajúce po chybových správach databáz

Časté sú najmä odhaľované zraniteľnosti založené na databázach Oracle. Identifikované Googlom sú tucty stránok s iSQL*Plus interfaceom, prvý krok hackera s cieľom spustiť buffer overflow útok na Oracle databázu:

```
intitle:iSQL intitle:Release inurl:isqlplus
```

Detekované taktiež môžu byť stránky hostujúce fóra, aplikačné servery atd.

```
inurl:discoverer/viewer
```

```
inurl:f90servlet
```

3.5 Pátranie po užívateľských menách, heslách

Mechanizmy pre overovanie totožnosti obvykle chránia informácie pomocou užívateľského mena a hesla. Na sieti sú dostupné ako prístupové mená, tak aj heslá k rôznym službám.

Prístupové mená sú často podhodnocované, lebo tvoria tú menej dôležitú polovicu pre väčšinu systémov overovania totožnosti, no dajú sa využiť napríklad v sociálnom inžinierstve. Jedným zo spôsobov ich získavania je pátranie po prihlasovacích portáloch:

```
login | logon
```

Prirodzene ciest pre nájdenie užívateľských mien pomocou Googlu je mnoho a jednou z ďalších môžu byť štatistky programov, ktoré kontrolujú aktivity daného webu. Výstupné súbory napr. programu Webalizer sa dajú vypátrať pomocou dotazov ako:

```
intext:webalizer intext:"total usernames" intext:"Usage Statistics for"
```

Užívateľské mená sa však môžu vyskytovať aj na bežných stránkach a k výsledkom sa dá dopátrať dotazom:

```
username | userid | employee.ID | "your username is"
```

Takéto všeobecné techniky hľadania sú veľmi účinné, ak sú zamerané na nejaký konkrétny server a kombinujú sa teda ešte s operátorom *site*. Niekoľko ďalších príkladov funkčných dotazov je zhrnutých v tabuľke 7.

Dotaz	Popis
`inurl:admin inurl:userlist`	súbory so zoznamom užívateľov všeobecne
`inurl:admin filetype:asp inurl:userlist`	súbory so zoznamom užívateľov všeobecne
`filetype:reg intext:"internet account manager"`	MS Internet Account Manager môže prezradiť užívateľské mená
`filetype:reg reg HKEY_CURRENT_USER username`	exportované registre Windows môžu obsahovať užívateľské mená a ďalšie informácie
`filetype:log username putty`	klientské protokoly PUTTY SSH

Tabuľka 7. Dotazy pátrajúce po užívateľských menách

Heslá sa vo väčšine prípadov na webe vyskytujú v zašifrovanej podobe, ako demonštruje obrázok 11, ktorý sa mi podarilo získať približne v apríli 2008, no ani to nie je dostatočný obranný mechanizmus. V mnohých prípadoch stačí tieto heslá vložiť do nejakého špeciálneho programu (napríklad John the Ripper alebo Cain and Abel), ktorý heslo dešifruje a zobrazí v podobe čistého textu. Takéto programy dokážu heslo zložené zo

4 znakov ASCII tabuľky prelomiť dokonca behom niekoľkých sekúnd [22]. Samozrejme, čas nevyhnutný k prelomeniu hesla závisí na jeho dĺžke a rozmanitosti použitých znakov.

Dotaz

```
filetype:pwd inurl:(service | authors | administrators)
inurl:_vti_pvt
```

kombinuje hľadanie pre niektoré z podporných súborov MS FrontPage.


-FrontPage- eracinelli:o5Q7L/O/aECw
-FrontPage- eracinelli:o5Q7L/O/aECw
[redacted]/_vti_pvt/service.pwd - 1k - Archiv - Podobné stránky

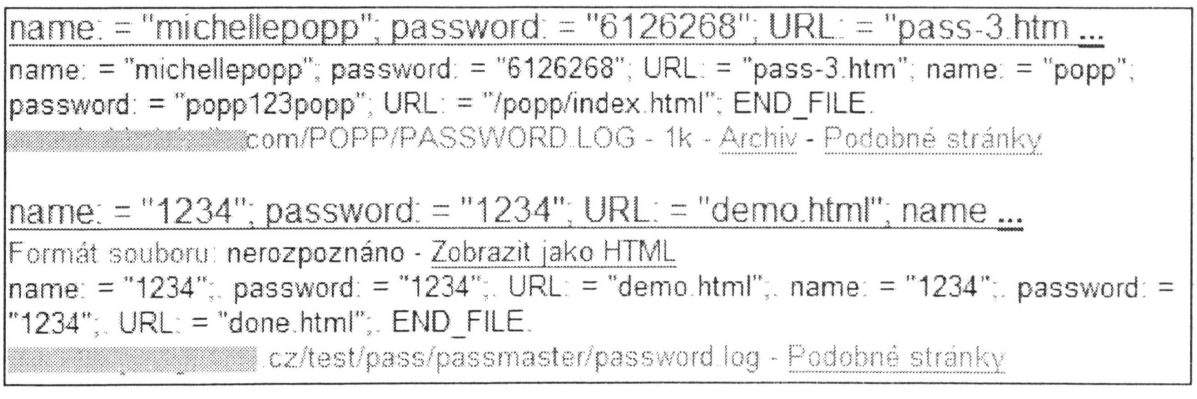
-FrontPage- contractor:9RPTxXVk.eTGM
-FrontPage- contractor:9RPTxXVk.eTGM
[redacted].org/_vti_pvt/service.pwd - 1k - Archiv - Podobné stránky

-FrontPage- lakeregioncreations:iNGdly8lDlk76:fake:invalid
Formát souboru: nerozpoznáno - Zobrazit jako HTML
-FrontPage- lakeregioncreations:iNGdly8lDlk76:fake:invalid
[redacted].com/tmp/_vti_pvt/service.pwd - Podobné stránky

-FrontPage- ekendall:bYld1Sr73NLKo:louisa:5zm94d7cdDFiQ
-FrontPage- ekendall:bYld1Sr73NLKo:louisa:5zm94d7cdDFiQ
[redacted].org/garderobe/_vti_pvt/service.pwd - 1k - Archiv - Podobné stránky

Obr. 11. Zašifrované heslá získané približne v apríli 2008

V niektorých prípadoch sa dajú dokonca nájsť súbory so všetkými potrebnými informáciami: užívateľské mená, nezašifrované heslá, a tiež hostitelia, ktorý pomocou týchto údajov overujú totožnosť užívateľov. Túto skutočnosť dokladá obrázok 12 s heslami, ktoré sa mi podarilo nájsť v apríli 2008, ilustračné dotazy na dáta spojené s heslami sú uvedené tabuľke 8 [4,8].


name: = "michellepopp"; password: = "6126268"; URL: = "pass-3.htm ...
name: = "michellepopp"; password: = "6126268"; URL: = "pass-3.htm"; name: = "popp";
password: = "popp123popp"; URL: = "/popp/index.html"; END_FILE.
[redacted]com/POPP/PASSWORD.LOG - 1k - Archiv - Podobné stránky

name: = "1234"; password: = "1234"; URL: = "demo.html"; name ...
Formát souboru: nerozpoznáno - Zobrazit jako HTML
name: = "1234";. password: = "1234";. URL: = "demo.html";. name: = "1234";. password: =
"1234";. URL: = "done.html";. END_FILE.
[redacted].cz/test/pass/passmaster/password.log - Podobné stránky

Obr. 12. Užívateľské mená, nezašifrované heslá, názvy hostiteľov (apríl 2008)

Konkrétne pravidlá pre vyhľadávanie mien a hesiel je však dosť naformulovať. Dobré výsledky sa dajú dosiahnuť s kombináciou slov *account, users, admin, administrators, passwd,* password v spojení s typmi súborov .xls, .txt, .doc, .mdf, .pdf. Taktiež je dobré venovať pozornosť adresárom obsahujúcim v mene slová *admin, backup* a podobne.

Dá sa očakávať, že hesiel nachádzajúcich sa v archíve Googlu nebude veľa, no ak natrafíme na nejaký výsledok, už len tento fakt prezrádza úroveň zabezpečenia serveru a ten sa pochopiteľne môže stať terčom ďalšieho útoku. Navyše väčšina hodnotných nálezov vyžaduje trpezlivosť, kreativitu, inteligenciu a tiež trochu šťastia. To platí rovnako pre hľadanie mien a hesiel, tak aj pre ostatné dôverné informácie.

Dotaz	Výsledok
„http://:*@www" site*	heslá na stránky *site*, zapísané v podobe http://username:password@www...
filetype:bak inurl:"htaccess | passwd | shadow | htusers"	záložné kópie súborov, v ktorých sa môžu nachádzať informácie o menách užívateľov a heslách
filetype:mdb inurl:"account | users | admin | administrators | passwd | passwords"	súbory typu mdb, ktoré môžu obsahovať informácie o heslách
inurl:admin inurl:backup intitle:index.of	adresáre obsahujúce v mene slová admin a backup
"Index of/" "Parent Directory" "ws_ftp.ini" filetype:ini ws_ftp PWD	konfiguračné súbory programu WS_FTP, ktoré môžu obsahovať heslá pre prístup k FTP serverom

Tabuľka 8. Heslá – ukážkové dotazy

3.6 Osobné dáta a dôverné informácie

Dôverné dokumenty sú bohužiaľ taktiež umiestňované na verejne prístupné miesta, alebo posielané bez patričného zabezpečenia. Na Internet je možné nájsť životopisy s adresou, telefónom, e-mailom v hojnom počte, napríklad dotazom

*intitle:"curriculum vitae" "phone *" "address *" "e-mail"*

Keďže veľa užívateľov Internetu vytvára rôzne elektronické adresáre, aj v nich sa dajú nájsť mená, čísla telefónov a e-mailové adresy. S použitím sociálneho inžinierstva je tieto informácie možné zneužiť, najmä ak sa týkajú ľudí v rámci nejakej uzatvorenej skupiny, napríklad firmy.

Dotaz	Typ dôvernej informácie
filetype:xls inurl:email.xls	súbory email.xls, ktoré môžu obsahovať elektronické dáta
"not for distribution" confidental	dokumenty so stupňom utajenia confidental
filetype:ctt "msn"	zoznam kontaktov MSN

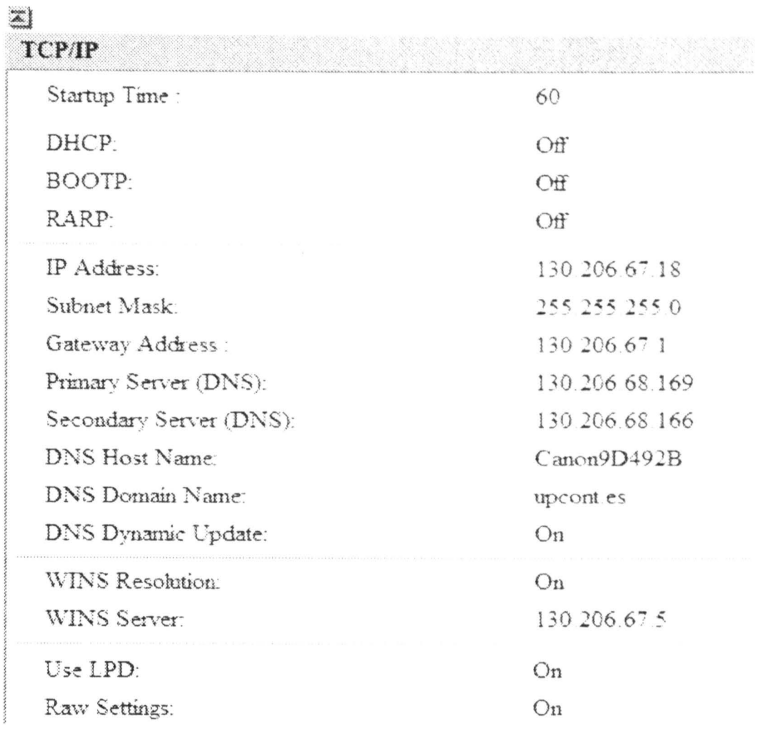
Tabuľka 9. Dotazy pátrajúce po osobných dátach

Na zabránenie úniku osobných informácií rovnako ako v prípade hesiel môžeme jedine zachovávať ostražitosť a mať prehľad o zverejnených dátach. Firmy a inštitúcie by mali vytvoriť a dodržiavať príslušné procedúry a postupy popisujúce vnútorný obeh informácií.

3.7 Sieťové zariadenia

Nie je nijak neobvyklé, že v prostredí počítačových sietí existujú zariadenia so svojou vlastnou webovou stránkou určenou pre konfiguráciu a ovládanie daného zariadenia. Niektorí správcovia však bezpečnosť takých zariadení, ako sieťové tlačiarne alebo web

kamery neberú vážne, a tým pádom vypátranie ich stránky predstavuje určité riziko, pretože často obsahujú informácie o cieľovej sieti, ako môžeme vidieť na obrázku 13.



TCP/IP	
Startup Time :	60
DHCP:	Off
BOOTP:	Off
RARP:	Off
IP Address:	130.206.67.18
Subnet Mask:	255.255.255.0
Gateway Address :	130.206.67.1
Primary Server (DNS):	130.206.68.169
Secondary Server (DNS):	130.206.68.166
DNS Host Name:	Canon9D492B
DNS Domain Name:	upcont.es
DNS Dynamic Update:	On
WINS Resolution:	On
WINS Server:	130.206.67.5
Use LPD:	On
Raw Settings:	On

Obr. 13. Informácie o sieti zo stránky tlačiarne Canon ImageReady nájdenej Googlom v marci 2008

Na webovú kameru pripojenú na sieť sa vo väčšine prípadov nazerá skôr ako na zdroj zábavy, než na niečo ohrozujúce bezpečnosť. Spoločnosť Netscape bola kedysi známa tým, že poskytovala zákazníkom pohľad na sídlo firmy a jej okolie [8]. No nie je ťažké si predstaviť zneužitie dát získaných z web kamery napríklad na priemyselnú špionáž, plánovanie prepadnutia a podobe.

Taktiež sieťové tlačiarne sú zdrojom ohromného množstva informácií, zobrazujú informácie o okolitej sieti a navyše veľa z takýchto zariadení sa nachádza v prednastavenej konfigurácii, ktorá umožňuje jednoduché zmocnenie sa takéhoto zariadenia. Tabuľka 10 zhŕňa niekoľko dotazov, výsledkom ktorých sú stránky sieťových zariadení.

Dotaz	**Zariadenie**
`intitle:"Live View/ - AXIS"`	AXIS Video Live Camera
`inurl:"viewerframe?mode="`	Kamera Panasonic Network
`intitle:liveapplet inurl:LvAppl`	Sieťová kamera Canon
`intitle:"WJ-NT104 Main Page"`	Sieťová kamera Panasonic
`intitle:"remote ui:top page"`	Tlačiareň Canon ImageReady
`inurl:sts_index.cgi`	Kopírky RICOH
`intitle:"Sipura.SPA.Configuration" -.pdf`	VoIP zariadenia

Tabuľka 10. Príklady dotazov pátrajúcich po sieťových zariadeniach

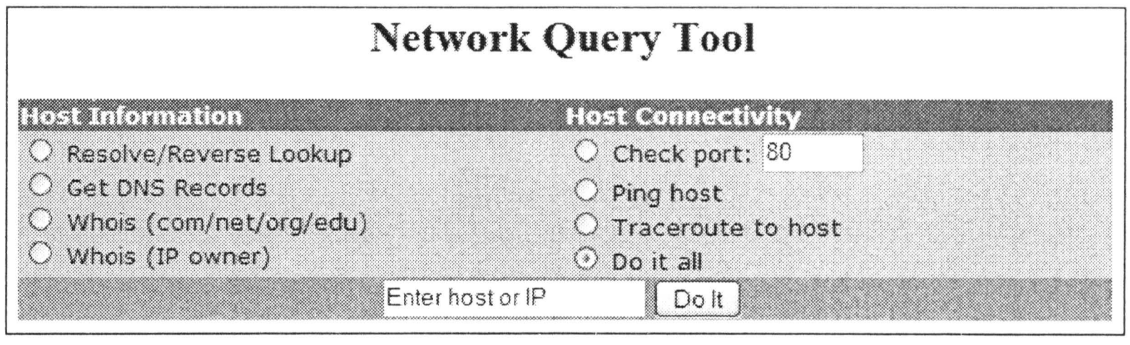
3.8 Webové utility

Google je veľmi účinný a flexibilný nástroj, ale prirodzene nevie všetko. Pri procese mapovania siete a získavaní poznatkov o určitom cieli či potenciálnej obeti je často efektívne využiť aj iné prostriedky, mimo Google. Napríklad úlohy ako *ping*, *whois*, utility *traceroute*, skenovanie portov a podobne – na všetky tieto funkcie existuje mnoho nástrojov. Jedným z nich je aj nástroj zvaný Network Query Tool (NQT), zobrazený na obrázku 14. NQT je webová aplikácia, čo znamená, že akýkoľvek užívateľ, ktorý má prístup na túto stránku, môže využívať jej funkcie proti akémukoľvek cieľu. Štandardná inštalácia umožňuje vyhľadávať IP názvy hostiteľov, vydávať dotazy DNS, vykonávať dotazy *whois*, overovať aktivitu na konkrétnych portoch či zisťovať trasy paketov. Je to veľmi šikovný nástroj aj z toho dôvodu, že jeho funkcie pochádzajú z webu hosťujúceho aplikáciu NQT, čo znamená, že webový server maskuje skutočnú adresu užívateľa, do určitej miery je teda zabezpečená anonymita.

Program NQT býva obvykle uložený v súbore s názvom *nqt.php* a vo svojej štandardnej konfigurácii zobrazuje titulok „Network Query Tool“. Takže jednoduchý dotaz

```
inurl:nqt.php intitle:"Network Query Tool"
```

nám vráti určité výsledky, v ktorých je možné sa dopátrať k fungujúcemu programu NQT.



Obr. 14. Program Network Query Tool

3.9 Exploity a ich využitie

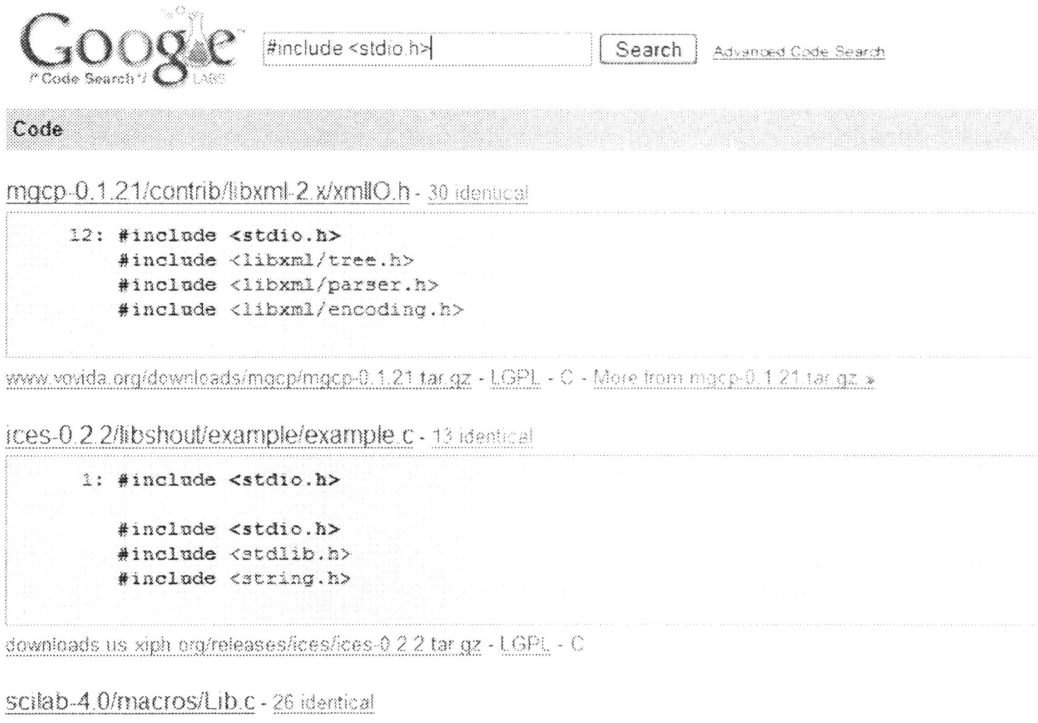
Pri násilných vniknutiach do systémov a cieľov, na ktoré sa hacker zameriava, využíva rôznorodé nástroje. Do tejto výbavy neodmysliteľne patria časti kódu, ktorým sa anglicky hovorí expoity. Exploit (z rovnakého francúzskeho slova s významom „úspech“), je kúsok softwaru, dát, alebo postupnosť príkazov, využívajúca výhody chyby, slabosti alebo zraniteľnosti s cieľom spôsobiť neúmyselné alebo nečakané správanie softwaru, hardwaru alebo čohokoľvek elektronického [10]. Vďaka tomu, že šírením podobného kódu sa zaoberá množstvo stránok, schopnosťami Googlu sa po nich pátra celkom ľahko.

Jedným so spôsobov, ako nájsť kód exploitov, je zamerať sa na príponu súboru zdrojového kódu a hľadať konkrétny obsah vo vnútri kódu. Veľké množstvo exploitov je napríklad napísaných v programovacom jazyku C, kde sa ako prípona so zdrojovým kódom obvykle používa .c (`filetype:c exploit`).

Iný spôsob, ktorým sa dá dopátrať k exploitom, je založený na znalosti bežne používaných reťazcov v samom zdrojovom kóde [9]. Napríklad, množstvo programov

jazyka C obsahuje štandardné knižnicové funkcie pre vstupno-výstupné operácie pridané vo vnútri zdrojového kódu pomocou príkazu *include*, ako napríklad *#include <stdio.h>*. Dotaz v tvare `"#include <stdio.h> exploit"` by vypátral zdrojový kód C obsahujúci slovo exploit, pričom sa však neobmedzuje iba na hľadanie dokumentov s príponou .c, ale ponúkne nám aj zachytený kód v HTML stránkach.

Účinným spôsobom na vypátranie exploitu je nová služba Google Code Search (*www.google.com/codesearch*), ktorá je primárne určená na hľadanie verejných zdrojových kódov. Je to v podstate akási prirodzená alternatíva k predošlým technikám, ktorá ponúka zaujímavé rozšírené možnosti. Predovšetkým je to povolenie dotazov s regulárnymi výrazmi a unikátne pokročilé operátori (*file*, *package*, *license* a pod.). Pre zaujímavosť uvádzam na obrázku 15 výsledky zobrazené po zadaní dotazu `#include <stdio.h>` do Code Search.



Obr. 15. Google Code Search

Ak sa pokúšame nájsť kód napísaný v jazyku C alebo C++, pomocou `lang:c` alebo `lang:c++` sa dopátrame k žiadanému výsledku. Aj keď by sa mohlo zdať, že je to veľmi podobné hľadaniu pomocou prípony súboru, nie je tomu tak. Služba Code Search totiž pracuje pokročilejšie, rozhodnutie, v akom programovacom jazyku bol kód napísaný padne až po dôkladnej analýze zdrojového kódu (bez ohľadu na príponu súboru).

Ako dokazujú výskumy mnohých bádateľov a bloggerov, Google Code Search môže byť taktiež využitý na lokalizáciu potenciálne zraniteľného softwaru. Pár zozbieraných príkladov uvádzam v tabuľke 11.

Dotaz	Popis zraniteľnosti
`lang:php (echo|print).*\$_(GET|POST|COOKIE|REQUEST)`	XSS (cross-side scripting) zraniteľnosť
`.*mysql_query\(.*\$_(GET|POST).*`	Možná SQL injektáž
`lang:php (system|popen|shell_exec|exec)\s*\(\$_(GET|POST|COOKIE| REQUEST).*\)`	Umožnenie vzdialeného vykonania kódu
`lang:php echo.*\$_SERVER\ ['PHP_SELF']`	XSS zraniteľnosť
`lang:php "WHERE username='$_"`	SQL injektáž

Tabuľka 11. Dotazy pre Google Code Search pátrajúce po zraniteľnom kóde

3.10 AJAX Search API - prekvapivé možnosti Google služieb

AJAX Search API je zaujímavá služba Googlu rozširujúca možnosti vyhľadávania. Jej zámerom bolo nahradiť staršiu vyhľadávaciu službu založenú na protokole SOAP, ktorej podpora bola pred určitým časom skončená. Primárny cieľ je umožniť externým stránkam hosťovať Googlom dodávané pomôcky buď v rámci, alebo aj mimo hosťujúcej stránky. Týka sa to vyhľadávania video klipov, máp, blogov, kníh a to všetko na jednom mieste získaním okamžitej spätnej väzby cez celú Google platformu. Pre použitie AJAX Search API musíme získať API (Application Programming Interface – rozhranie pre programovanie aplikácií) kľúč, jeho generácia je možná na domovskej stránke *http://code.google.com/apis/ajaxsearch*. Nebudem zachádzať do podrobností, ale uvediem aspoň základné myšlienky, ako je možné využiť túto službu, tak trochu neštandardným spôsobom. S trochou znalostí JavaScriptu je jednoduché vytvoriť vlastné vyhľadávacie enginy pre kladné, ale aj škodlivé účely. Prvý krok je s pomocou prostriedkov na monitorovanie sieťovej prevádzky (ako je napr. rozšírenie LiveHTTP Headers prehliadača Mozilla Firefox) zistiť, že so službou Googlu sa komunikuje prostredníctvom URL adresy podobnej tejto:

```
www.google.com/uds/GwebSearch?callback=our_callback&context=0&rsz=large&q=GHDB&key=internal&v=1.0
```

Povšimnime si najmä parameter *callback*, čo je funkcia JavaScriptu a parameter *key*, ktorého hodnota bude vygenerovaný API kľúč. Modifikáciou tejto URL a jej zakomponovaním do skriptu môžeme jednoducho komunikovať a získavať požiadavky zo služby Googlu GwebSearch. Táto technika spolu s technikou scrapingu (získavanie dát z web stránky) je nosnou myšlienkou on-line nástroja Google Hacking Database. Úspešnú implementáciu tohto projektu nájdeme na adrese *http://www.gnucitizen.org/ghdb/*. Táto aplikácia v podstate dynamicky extrahuje všetky informácie z GHDB Johnnyho Longa a prezentuje ich v úhľadnej, grafickej forme. Je možné prehliadať každú kategóriu z databázy a vybraním dotazu získame okamžitú a „živú“ spätnú väzbu – zobrazenie výsledkov, ktoré sú zabezpečené pomocou rozhrania Google AJAX Search API. Toto všetko ešte v kombinácii s ďalšou službou Googlu, Google Co-op (Vlastný vyhľadávač) otvára kreatívnym útočníkom nové možnosti pre vytváranie nebezpečných nástrojov.

3.11 Automatizovaný zber a analýza informácií

Počítače pomáhajú zautomtizovať nudné a zdĺhavé úlohy. Aby však niečo bolo možné automatizovať, musí to ísť urobiť aj manuálne. A keď sa proces ručnej práce vyladí a zoptimalizuje, nastupuje na scénu algoritmus, ktorý tento proces preloží do počítačového programu.

Podobné je to aj s hľadaním a zberom informácií z Googlu. Keby sme tento proces chceli rozdeliť do logických častí, mohli by to byť:

- vytvorenie efektívneho vyhľadávacieho termínu
- získanie dát zo zdroja
- analýza a spracovanie získaných dát

Nás bude teraz zaujímať získavanie dát. Na najnižšej úrovni sa nadväzuje TCP spojenie so zdrojom dát (teda stránkou Googlu). Keďže Google je web aplikácia, pripojenie smeruje na port 80. Za normálnych okolností sa samozrejme používa webový prehliadač, ale keďže momentálne je naše zameranie na automatizáciu, potrebujeme komunikovať s Googlom na programovej úrovni.

Asi najflexibilnejšou metódou, no zároveň vyžadujúcou najväčšie úsilie, je získavanie (scraping) výsledkov pomocou špecializovaných nástrojov. Takými môžu byť napríklad NetCat, Wget, alebo Curl.

Pri bežnom hľadaní prostredníctvom Googlu sa vygeneruje URL adresa, s ktorou sme sa už oboznámili v jednej z predchádzajúcich kapitol. Je to vlastne GET požiadavka obsahujúca určité parametre. Vyššie spomenuté nástroje môžeme použiť k získaniu výsledkov žiadaného dotazu, server odpovie a vráti výsledky v HTML forme. Žiadosť o tento proces prostredníctvom Wget by vyzerala (v OS Windows príkaz zadaný z príkazového riadku, v UNIX-e z shellu):

```
wget "http://www.google.cz/search?hl=en&q=test" -O output.txt
```

V tomto momente sa však ešte k cieľu nedostaneme v dôsledku chyby 403, ako môžeme vidieť na obrázku 16, ktorý sa mi podarilo zachytiť, keď som tento postup skúšal. Dôvod je jednoduchý – Google nemá prílišné porozumenie k akejkoľvek automatizácii. Dokonca aj podmienky používania služieb Googlu zakazujú bez predchádzajúceho povolenia používanie nástrojov pripájajúcich sa k službám inak než prostredníctvom webového rozhrania (vrátane použitia rôznych skriptov).

```
wget "http://www.google.cz/search?hl=en&q=test" -O aaa
--01:17:57--  http://www.google.cz/search?hl=en&q=test
          => `aaa'
Resolving www.google.cz... done.
Connecting to www.google.cz[66.249.93.99]:80... connected.
HTTP request sent, awaiting response... 403 Forbidden
01:17:58 ERROR 403: Forbidden.
```

Obr. 16. Hlásenie o chybe pri posielaní HTTP žiadosti Googlu

Náš malý problém má však celkom jednoduché riešenie. Wget posiela Googlu HTTP hlavičku, v ktorej je obsiahnutá identifikácia prehliadača alebo programu (pole „User-Agent“ v hlavičke). Pomocou prepínača –U sa však Wget dokáže zamaskovať a

identifikovať ako čokoľvek iné, než je on sám, pretože Google ho v inom prípade rozpozná ako nebezpečný nástroj s možným použitím k automatizácii a nepovolí komunikáciu. Takže príkazom

```
wget -U moj_prehliadac „http://www.google.cz/search?hl=en&q=test" -O
output.txt
```

získame žiadané výsledky, uložené v súbore *output.txt*.

Veľmi rýchlo a účinne na mini hacky typu získanie počtu výsledkov na dotaz, sa dá použiť aj Lynx, textový prehliadač webových stránok. Uvážme príkaz:

```
$ lynx -dump "http://www.google.com/search?q=google" | grep Results | awk
-F "of about" '{print $2}' | awk '{print $1}'
    2,540,000,000
```

V tejto fáze máme požadované výsledky získané a uložené, nastupuje ich spracovanie podľa individuálnych potrieb subjektu. Najčastejšie sa na tento účel používajú šikovne napísané skripty, napríklad k extrahovaniu URL, telefónnych čísel, IP adries alebo e-mailov z výsledkov hľadania [9]. Ja sa pokúsim predviesť aspoň jeden zo spôsobov, akým sa dá ísť na získavanie e-mailových adries.

Predpokladajme, že chceme získať e-mailové adresy pre yahoo.com. Dotaz v tvare `„@yahoo.com" email` vyhľadávaný v sekcii Web nebude príliš účinný, preto to skúsime radšej v sekcii Skupiny. V prípadoch ako je tento, je najlepšie, keď sa pre pátraní po konkrétnych reťazcoch využijú regulárne výrazy. Na spracovanie stiahnutých dokumentov, ktoré chceme prehľadať, som skúsil použiť nasledujúci skript Perlu:

```
#!/usr/bin/perl
#
# Použitie: ./search.pl SUBOR_NA_PREHLADANIE ZOZNAM_SLOV
#
# Hľadanie slov v súbore
#
use strict;
open(SEARCHFILE,$ARGV[0]) || die("Can not open searchfile because $!");
open(WORDFILE,$ARGV[1]) || die("Can not open wordfile because $!");
my @WORDS=<WORDFILE>;
close(WORDFILE);
my $LineCount = 0;
while(<SEARCHFILE>) {
      foreach my $word (@WORDS) {
            chomp($word);
            ++$LineCount;
            if(m/$word/) {
                  print "$&\n";
                  last;
            }
      }
}
close(SEARCHFILE);
```

Skript príjma dva argumenty – súbor, ktorý sa má prehľadať a zoznam slov, ktoré sa majú vyhľadať. Program je pomerne jednoduchý, v podstate nerobí nič iné ako *grep*. Ale zaujímavý a užitočný začne byť, keď namiesto zoznamu slov použijeme regulárne výrazy. Na lokalizáciu rôznych foriem e-mailových adries by sme mohli využiť napríklad takýto regulárny výraz:

```
[a-zA-Z0-9._-]+@(([a-zA-Z0-9_-]{2,99}\.)+[a-zA-Z]{2,4})|((25[0-5]|2[0-
4]\d|1\d\d|[1-9]\d|[1-9])\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]\d|[1-9])\.(25[0-
5]|2[0-4]\d|1\d\d|[1-9]\d|[1-9])\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]\d|[1-9]))
```

Pre naše potreby si uložíme výsledky vyhľadávania do súboru *results.html* a regulárny výraz uvedený vyššie vložíme do súboru *wordfile.txt*. Na získanie výsledkov z príkazového riadku som použil spomínaný program Lynx:

```
lynx -dump
http://groups.google.com/groups?q=%22%40yahoo.com%22%20email&hl=en&num=10
0 > results.html
```

Parametrom *num=100* sťahujeme prvých 100 výsledkov. Parametrom *start* môžeme prípadne určiť, od ktorej pozície chceme zobrazovať výsledky. Pre druhú (101-200), resp. tretiu stovku výsledkov by sme teda pridali do URL Googlu ešte parameter *start=100*, resp. *start=200*. Po uložení výsledkov do súboru *results.html* naň pustíme skript *(./ssearch.pl results.html wordfile.txt)* a začnú sa hľadať výrazy e-mailových adries zodpovedajúce podmienkam regexpu vloženého do *wordfile.txt*. Na zúženie výsledkov som ešte použil presmerovanie výstupu do *„grep yahoo | head -10 | sort –u“*, aby sa vrátilo maximálne 10 jedinečných adries obsahujúcich slovo *yahoo*. Finálne výsledky sa mi podarilo zachytiť a sú publikované na obrázku 17.

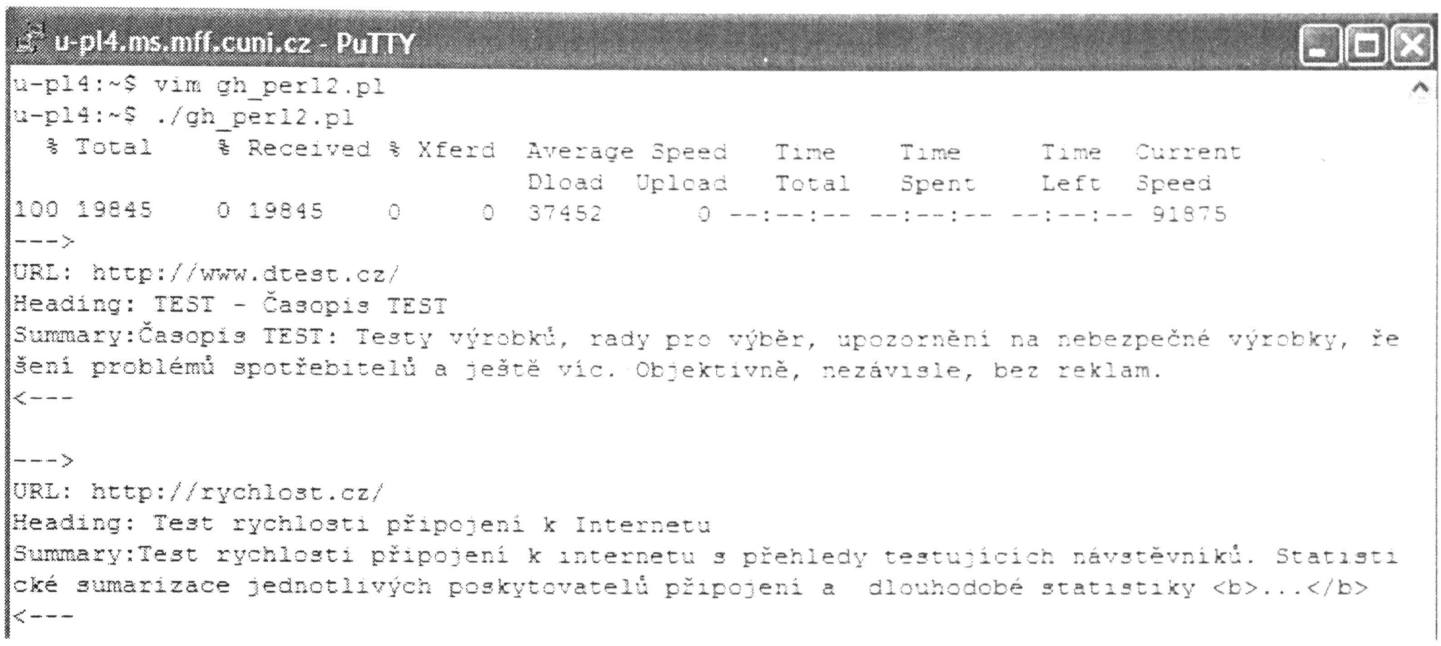
Obr. 17. Perl skript na lokáciu e-mailových adries

V predchádzajúcej sekcii som naznačil, ako pomocou nástrojov typu Wget a Curl možno získať surové HTML zo servera a následne vyhľadať e-mailové adresy. Zaujímalo ma aj, ako zautomatizovať a zozbierať individuálne výsledky z takéhoto neprehľadného HTML. Pokúsil som sa zostaviť skript, ktorý by mi niečo také umožnil. Najskôr som pomocou doplnku FireBug do prehliadača Firefox zisťoval štruktúru HTML kódu, ktorú by som potom mohol použiť na parsovanie výsledkov. Zistil som, že každý fragment obsahu, tzv. snippet, je uvedený kódom *<li class="g">*. To bolo základom pre nasledujúci skript:

```
1 #!/usr/bin/perl
2 use strict;
3 my $result=`curl -A mojbrowser "http://www.google.com/search?q=test&
hl=cs"`;
4
5 my $start;
6 my $end;
7 my $token="<li class=g>";
8
9 while (1){
10 $start=index($result,$token,$start);
11 $end=index($result,$token,$start+1);
```

```
12 if ($start == -1 || $end == -1 || $start == $end){
13 last;
14 }
15
16 my $snippet=substr($result,$start,$end-$start);
17 my ($pos,$url) = cutter("<a href=\"","\"",0,$snippet);
18 my ($pos,$heading) = cutter(">","</a>",$pos,$snippet);
19 my ($pos,$summary) = cutter("<div class=\"s\">","<br>",$pos,$snippet);
20
21 #remove <em> and </em>
22 $heading=cleanEm($heading);
23 $url=cleanEm($url);
24 $summary=cleanEm($summary);
25
26 print "--->\nURL: $url\nHeading: $heading\nSummary:$summary\n<---
\n\n";
27 $start=$end;
28 }
29
30 sub cutter{
31 my ($starttok,$endtok,$where,$str)=@_;
32 my $startcut=index($str,$starttok,$where)+length($starttok);
33 my $endcut=index($str,$endtok,$startcut+1);
34 my $returner=substr($str,$startcut,$endcut-$startcut);
35 my @res;
36 push @res,$endcut;
37 push @res,$returner;
38 return @res;
39 }
40
41 sub cleanEm{
42 my ($str)=@_;
43 $str=~s/<em>//g;
44 $str=~s/<\/em>//g;
45 return $str;
46 }
```

Na treťom riadku skriptu sa externe volá *Curl* na získanie výsledkov jednoduchého dotazu, ktoré sa následne uložia do premennej *$result*. Skript prechádza celú HTML a postupne extrahuje všetky snippety. V cykle sa prehľadáva výskyt tokena, ktorým je hľadaný reťazec *<li class="g">* uložený v premennej *$token* (riadok 4), až kým sa neobjaví žiaden výskyt (riadok 12). Na riadku 27 sa presúva pozícia, kde sa skončilo predchádzajúce vyhľadávanie na pozíciu, odkiaľ sa bude opätovne token vyhľadávať. Mojím cieľom bolo získať dáta vo forme extrahovanej URL, titulku odkazu a súhrnu zo stránky výsledkov. Na to je využitá funkcia *cutter* (riadky 30-39), ktorá príjma štyri parametre: reťazec so začiatočným tokenom, reťazec s koncovým tokenom, skalár, odkiaľ sa má vyhľadávať a HTML v rámci ktorej chceme vyhľadávať. Na riadkoch 16-19 je táto funkcia použitá (v rámci cyklu) na extrahovanie URL adresy, hlavičky a súhrnu z každého snippetu. Posledné pozorovanie je, že Google zvýrazňuje vyhľadávací termín vo výsledkoch, takže bolo nutné ešte odstrániť tagy a z výsledkov, na čo slúži funkcia *cleanEm*. Konečne celý skript v akcii je zachytený na obrázku 18.



Obr. 18. Výsledky Perl skriptu

Jeden z ďalších, podľa mňa užitočných spôsobov spracovania výsledkov Googlu, vyžadujúci však značnú procesnú silu by mohol byť zameranie sa detailnejšie na vrátené výsledky, nielen na jednotlivé fragmenty z výsledkov. Myšlienka je asi nasledujúca – získať URL stránok vrátených vo výsledkoch, stiahnuť celú stránku, prekonvertovať ju na čistý text a previesť algoritmy na dobývanie znalostí na text. Potom by sme sa mohli napríklad zamerať na frekvenciu výskytu jednotlivých slov na stránke. Určite by sme skončili so zoznamom obsahujúcim často sa vyskytujúce slová a spojky na vrchu, ktorý by sa však mohli pomocou bežne dostupných zoznamov s takýmito slovami pre určitý jazyk prefiltrovať. Výsledný text by podľa mňa podal celkovú myšlienku o čom daný dokument je, alebo teda ktoré slová sú bežné naprieč stránkami. Ďalší krok by bol zoskupiť slová do fráz a zamerať sa na tie, ktoré sa nevyskytujú vo vysokej frekvencii na jednej stránke, ale majú vysokú frekvenciu výskytu pri skúmaní vo viacerých rozličných dokumentoch. Tento postup by zase poskytol špecifickú informáciu o danom subjekte.

4 OBRANA PROTI HACKEROM GOOGLU

Google hacking ponúka široké možnosti. Pochopenie základných techník a taktických manévrov Google hackerov je dôležité pre adekvátnu obranu a ochranu pred osobami, ktorý sa snažia získať a zneužiť citlivé informácie. Google hacking nie je iba získavanie dôverných a citlivých informácií využitím možností, ktoré nám vyhľadávač Google ponúka, ale aj vniknutie do problematiky ochrany vlastných súborov a dát práve porozumením metód vedúcich k napadnutiu systému či prelomeniu bezpečnostného mechanizmu. Na tomto mieste sa posnažím podať konkrétne informácie ako zabrániť danému typu úniku informácií, ako napraviť už existujúci únik, jednoducho ako ochrániť náš web pred útokmi takéhoto druhu.

Trochu netechnickou, no nemenej dôležitou metódou z hľadiska bezpečnosti webu je účinná bezpečnostná politika. Je dôležité pochopiť, aká užitočná je prísna politika ohľadne publikovania rôznych dát na Internete, preto si myslím, že stojí za zmienku, aj keď sa jej podrobnejšie nebudem venovať.

Údaje, ktoré by mali podliehať ochrane sa často na webe objavujú nevedomky, alebo z nepozornosti a vystavujú ich na dosah samotní užívatelia. Myslieť by sme mali hlavne na to, že webový server je primárne určený pre ukladanie dát určených širokej verejnosti. Ak je naozaj dôležité úplné súkromie nejakých informácií, mali by sa radšej presunúť na iné miesto, napríklad intranet alebo špecializovaný server, ktorého jedinou úlohou je poskytovať informácie bezpečným spôsobom.

Taktiež nie je veľmi dobrý nápad prideľovať verejnému webovému serveru odlišné role na základe rôznych prístupových úrovní. Dôvod je jednoduchý. Ak by sa nachádzali citlivé informácie na webovom serveri hneď vedľa verejných informácií, potom by pri napadnutí tohto serveru mohlo ľahko dôjsť k úniku citlivých informácií [9].

4.1 Výpisy adresárov

S rizikami výpisu adresárov sme sa už zoznámili. Užívateľom webu dovoľujú vidieť väčšinu súborov z nejakého adresára a často zobrazujú okrem súborov aj iné položky, ktoré môžu byť zneužité. Z tohto dôvodu by mali byť výpisy adresárov vždy vypnuté, pokiaľ nie je naším cieľom umožniť prácu so súbormi v štýle FTP. Výpisy adresárov sa u niektorých webových serverov zobrazujú v prípade, ak chýba takzvaný indexovací súbor, najčastejšie pomenovaný *index.html*, *index.htm*, *default.asp* a podobne a je definovaný v konfigurácii daného serveru. Tento súbor by sa mal teda nachádzať v každom adresári, ktorý má užívateľovi zobraziť nejakú úvodnú stránku. Možnosťou vypnutia zobrazovania adresárov je aj vytvorenie súboru *.htaccess* v koreňovom adresári, v ktorom je potrebné pre tento účel nadefinovať pred slovo *Indexes* pomlčku, alebo znamienko mínus:

Options –Indexes

4.2 Blokovanie indexovacích robotov

Weboví roboti sú pomocníci internetových vyhľadávačov, ktorý automatizovane prechádzajú weby a indexujú nájdené informácie. Google na tento účel využíva robota s názvom Googlebot. Inštrukcie pre robotov sa nachádzajú v súbore *robots.txt*, ktorého

štruktúra je štandardizovaná a roboti by mali jeho obsah, resp. obmedzenia v ňom obsiahnuté rešpektovať. Umožňuje definovať, ktoré súbory a adresáre majú byť mimo dosah týchto webových robotov, teda ktoré časti webu majú zostať skryté pred vyhľadávacími službami. Avšak je potrebné si uvedomiť, že niektorí webový roboti môžu ignorovať súbor *robots.txt*, ale väčšina tých s „dobrou povesťou" tento súbor rešpektuje [14]. A ďalej, súbor *robots.txt* je verejne prístupný súbor a hocikto teda môže nahliadnuť do jeho obsahu. A skutočne, hackeri túto možnosť aj využívajú a je to jedným z ich trikov, aby získali predstavu o tom, ako sú na serveri rozdelené súbory a adresáre. Dotazom Googlu

```
inurl:robots.txt filetype:txt
```

sa môžeme presvedčiť, koľko existuje webov s prehľadaným súborom *robots.txt*.

Súbor *robots.txt* musí byť umiestnený v koreňovom adresári webového serveru a musí mať povolené také nastavenie prístupu, aby ho webový server mohol čítať. Je to textový súbor a každý jeho riadok okrem komentára musí začínať jedným z dvoch príkazov, *user-agent* alebo *disallow*. Pole *user-agent* obsahuje názov alebo typ robota, riadok disallow určuje, na čo sa robot nesmie dívať [14]. Pre robota Googlu by teda riadok *user-agent* vypadal takto:

```
User-agent: Googlebot
```

V poli *user-agent* sa dá použiť zástupný znak *, ktorým smerujeme pokyny všetkým robotom. Obsah súboru *robots.txt*, ktorý by zakazoval všetkým robotom indexovať všetky časti webu by vyzeral:

```
User-agent: *

Disallow: /
```

V prípade, že chceme povoliť nejakého konkrétneho robota, jednoducho mu nič nezakazujeme a príkaz *disallow* zostane prázdny, teda mu povoľujeme všetko (v zmysle indexovania adresárov). Príklad, kedy sa vylúčia všetci roboti okrem robota GoodBot:

```
User-Agent: *

Disallow: /

User-Agent: GoodBot

Disallow:
```

Štandardy súboru *robots.txt* zverejnené na stránke *www.robotstxt.org* deklarujú, že používanie regulárnych výrazov a zástupných znakov (až na výnimku „*" v poli *User-Agent*) nie je podporované ani v riadku *user-agent*, ani *disallow*. Robot Googlu však podporuje používanie prípon. Vzor pre *disallow* môže používať hviezdičku v zmysle zástupného znaku pre ľubovoľný počet znakov. Okrem toho sa používa aj znak $ indikujúci koniec názvu. Teda ak by sme chceli Googlebotovi zabrániť, aby prechádzal naše PDF dokumenty a navyše aj adresár */tmp*, obsah súboru *robots.txt* by vyzeral nasledovne [14]:

```
User-Agent: Googlebot

Disallow: /tmp
```

```
Disallow: /*.pdf$
```

V prípade, že súbor *robots.txt* nevytvoríme, alebo bude prázdny, bude to znamenať, že sme poskytli všetkým robotom právo prechádzať a indexovať celú našu adresárovú štruktúru.

Ako náhle máme súbor *robots.txt* vytvorený a správne umiestnený, môžeme overiť jeho platnosť validátorom, ktorý nájdeme pomocou Googlu, alebo napríklad na stránkach *www.sxw.org.uk/computing/robots/check.html* [9].

4.3 META Tagy

V niektorých situáciách sa nám môže stať, že súbor robots.txt nemôžeme, alebo nechceme vytvárať. V tom prípade je obmedzenie robotov možné použitím nasledujúceho META tagu v hlavičke HTML dokumentu [4]:

```
<meta name="robots" content="noindex, nofollow">
```

Týmto špeciálnym META tagom hovoríme všetkým spolupracujúcim robotom, aby neidnexovali obsah stránky (hodnota `noindex` v položke *content*) a taktiež aby pri prehľadávaní stránky nesledovali odkazy vyskytujúce sa na stránke (`nofollow`).

Týmto prípadom však využitie špeciálnych META tagov ako metódy ochrany na našej stránke nemusí končiť. Môže sa nám vyskytnúť situácia, kedy chceme, aby vyhľadávacie roboty mali prístup a prehľadali stránku, avšak súčasne nechceme, aby Google vytváral archivovanú verziu stránky, teda aby sa vo výsledkoch vyhľadávania pri tejto stránky zobrazoval odkaz *Archív*. O bezpečnostných rizikách Google archívu už tiež bola reč. Zakázanie uloženia archivovanej verzie stránky teda prevedieme pomocou tagu:

```
<meta name="robots" content="noarchive">
```

Samozrejme aj tu platí, že túto našu reštrikciu budú dodržiavať iba spolupracujúci roboti. Ak však už došlo k archivácii kópie určitej stránky, ktorú sme zabudli zabezpečiť, dá sa z Google archívu následne odstrániť.

A ešte jeden tip z oblasti META tagov. Vložením tagu

```
<meta name="googlebot" content="nosnippet">
```

do hlavičky HTML dokumentu zabezpečíme, aby Google nezobrazoval fragmenty obsahu. Fragment obsahu je kúsok textu nachádzajúci sa na stránke výsledkov pod názvom nájdeného dokumentu a niekedy je našim cieľom zamedziť jeho zobrazovanie (napríklad v prípade autorizovaného prístupu k obsahu dokumentu). Zaujímavým vedľajším efektom meta-značky `NOSNIPPET` je to, že Google nebude daný dokument ani archivovať [4].

4.4 Štandardné nastavenia

Už vieme, že aj s minimálnym vynaložením úsilia je možné sa dopracovať k nejakým štandardným stránkam, frázam, titulkom stránok, programom a dokumentáciám. Tieto súbory väčšinou obsahujú určité charakteristické rysy, podľa ktorých nie je ťažké vystopovať a zistiť, aký software je na konkrétnom servery použitý, aké je štandardné nastavenie programu, aký typ zabezpečenia je zvolený. Ak je útočník vybavený napríklad

exploitom pre danú verziu softwaru, zvyšuje sa riziko úspechu jeho útoku, ako bolo už niekoľkokrát spomínané. Preto by sa mali odstraňovať spomínané položky z všetkého webového softwaru, ktorý sa na server inštaluje. Osvedčenou bezpečnostnou praktikou je odstránenie štandardných účtov a hesiel a tiež všetkých inštalačných skriptov alebo programov, ktoré boli dodané spoločne so softwarom.

4.5 Hľadanie bezpečnostných rizík – hacknutie vlastného webu

Pre získanie predstavy o potenciálnych bezpečnostných rizikách vlastného webu existuje jedna účinná metóda – pokúsiť sa naň podniknúť útok. Týmto spôsobom môžeme získať predstavu o slabých miestach nášho systému a podniknúť určité kroky k ich odstráneniu. V oblasti hackingu Googlom existujú nástroje pre automatizované skenovanie bezpečnostných rizík a slabín, ktoré nám sprostredkujú pohľad z perspektívy toho, ako samotný Google vidí náš web. Existujú však aj manuálne metódy, ktoré dobre poslúžia v prípade, že náš testovaný web nie je príliš rozľahlý. Pomocou operátoru *site* môžeme zistiť, čo všetko Google indexuje a dotaz `site:nasadomena.cz` by vypísal všetky Googlom archivované stránky zo serveru *nasadomena.cz*. Prostredníctvom týchto výsledkov si môžeme overiť, či skutočne všetky nájdené stránky sú určené pre verejnosť a neobsahujú citlivé informácie. Takéto ručné prehľadávanie je však namáhavé a časovo náročné, a preto je často vhodné použiť spomínaný proces automatizácie.

4.5.1 Automatizácia vyhľadávania a automatizačné nástroje

Existuje niekoľko spôsobov, ktorými sa dá hľadanie Googlom automatizovať, avšak Google oficiálne povoľuje automatizáciu dotazov iba prostredníctvom Google API (Application Programming Interface). Niektoré z ďalej spomínaných nástrojov spoliehajú na SOAP (Simple Object Access Protocol) API, ktoré však Google prestalo podporovať v dôsledku zavedenia AJAX API. Ak disponujeme starým SOAP API licenčným kľúčom, patríme medzi šťastlivcov, pretože tento kľúč stále funguje s nástrojmi, ktoré ho vyžadujú. Ak ho nemáme, stále je možnosť dohľadať ho pomocou Googlu, alebo zvážiť použitie programu Aura od spoločnosti SensePost (*www.sensepost.com/research/aura*) ako alternatívnu možnosť. Následne je potrebné upozorniť, že automatizačné nástroje, ktoré nevyžadujú zadanie licenčného kľúča môžu fungovať v rozpore s tým, ako Google definuje podmienky používania svojich služieb (Terms of Services). Viacej informácií o týchto podmienkach je dostupných na *www.google.com/accounts/tos*.

Google Hacking Database

Už sme sa oboznámili s princípom získavania citlivých údajov z databázy Googlu a tiež boli predstavené techniky tvorby dotazov, ktoré na tento účel slúžia. Dalo by sa povedať, že Google Hacking Database (GHDB) je ich najznámejším úložiskom a jej zakladateľ nie je nikto iný ako Johnny Long, uznávaný špecialista na informačnú bezpečnosť, ktorý o bezpečnosti sietí a hackingu Googlom už aj prednášal na niekoľkých konferenciách o bezpečnosti počítačov po celom svete. Na pravidelnom rozširovaní databázy sa však môže podieľať ktorýkoľvek dobrovoľník a nadšenec, v súčasnej dobe zahŕňa takmer 1500 dotazov, ktoré ďalej separuje do viacerých kategórií, ako napríklad [15]:

- *Informačné správy a zraniteľnosť*
- *Chybové hlásenia*
- *Súbory obsahujúce zaujímavé informácie*
- *Súbory obsahujúce heslá*
- *Stránky s prihlasovacím formulármi*

Tento depozitár hackerských techník realizovateľných pomocou Googlu, hosťovaný na *http://johnny.ihackstuff.com*, spomínam aj z dôvodu, že mnohé z automatizačných nástrojov ho vo veľkej miere využívajú a podporujú.

Charakteristika súčasných automatizačných nástrojov

Prvým z použiteľných nástrojov, o ktorom sa zmienim je **Gooscan**. Vytvoril ho Johnny Long, je založený na Linuxe, ponúka dávkové hľadanie Googlom a využíva výťažky z Google Hacking Database. Nebol vytvorený tak, aby spolupracoval s API Googlu, takže porušuje podmienky prevádzky služieb a je teda na rozhodnutí každého z nás, či chceme zámerne porušovať tieto podmienky pri zisťovaní prípadných únikov z nášho webu. Ak sa nakoniec rozhodneme použiť tento, alebo jemu podobný nástroj, ktorý podmienky prevádzky porušuje, Google môže zablokovať niektoré rozsahy IP adries a znemožniť používanie jeho vyhľadávacieho enginu [4].

Ďalšou aplikáciou je **Athena**, určená pre užívateľov operačného systému Windows. Rovnako ako Gooscan nie je založená na API Googlu a jej použitie je taktiež v rozpore s podmienkami používania služieb Googlu. Môžeme ju nájsť na adrese *http://snakeoillabs.com/*. Na vykonávanie základného vyhľadávania sa načítava XML súbor (môžeme stiahnuť alebo vytvoriť vlastný), štandardne sa dodáva aj súbor obsahujúci dotazy nájdené v GHDB.

Wikto od spoločnosti SensePost (*www.sensepost.com*) je webový skener s mnohými možnosťami. Nás prirodzene zaujíma predovšetkým aspekt Google dotazovania. Program funguje tak, že zadáme cieľ nášho skenovania a poprípade nejaké detailné informácie o serveri. Následne sa dostaneme do rozhrania, kde sa vyžaduje zadanie Google API kľúča. Táto sekcia je tak trochu chyták, lebo ako som už spomínal, Google prestal vydávať nové SOAP API licenčné kľúče. Ak ho však vlastníme, nič nám nebráni použiť ho, ak nie, možnosťou „obídenia" je použitie programu s názvom Aura od spoločnosti SensePost. Potom sa už ocitneme na hlavnej obrazovke, odkiaľ je možné realizovať samotné dotazy proti nášmu cieľu, teda testovanej stránke.

Zaujímavým nástrojom na skenovanie reťazcov z GHDB je aj **Goolag Scanner**. Na svedomí ho má hackerská skupina Cult of The Death Cow, nájdeme ho na web adrese *http://goolag.org*. Práve Goolag Scanner bol inšpiráciou pre spomínaný on-line nástroj Gnucitizien GHDB využívajúci AJAX API.

Na záver tejto stručnej charakteristiky som si nechal dva relatívne nové nástroje, a to **Google Rower** a **Google Site Indexer**. Obidva sa dajú stiahnuť zo stránky *http://www.tankedgenius.com*. Google Rower funguje ako rozšírenie do prehliadača Mozilla Firefox (ale tiež aj ako samostatný program) a na rozšírenie vyhľadávania používa techniku „brute force". Google Site Indexer používa špeciálne operátory *site* a *inurl* na vytvorenie akejsi súborovej a adresárovej mapy cieľovej web stránky. Posielaním dotazov

ako `site:tankedgenius.com` Google Site Indexer inkrementálne prejde všetky stránky, ktoré predtým zaindexoval Google [9].

4.6 Okamžité odstránenie z Google indexu

V prípade, že máme skontrolovaný náš web a zaznamenali sme potenciálne úniky informácií, nabáda sa otázka, aký je ďalší postup.

V prvom rade je potrebné tento obsah z nášho webu čo najrýchlejšie odstrániť. Je vhodné aj zistenie zdroja úniku a zaistenie, že sa úniky v budúcnosti nebudú opakovať. Google prevádzkuje výbornú stránku pomáhajúcu so zodpovedaním niektorých najčastejšie kladených dotazov z hľadiska webmastera, umiestnenú na webovej adrese *www.google.com/webmasters* [9].

Týmto však náš problém nemusí byť úplne vyriešený, stále treba pamätať na archívnu kópiu informácií. Možnosťou jej odstránenia je systém automatického odstránenia URL na adrese *http://www.google.com/webmasters/tools/removals*. Tento nástoj nás prevedie sériou otázok určených na overenie vlastníctva obsahu stránky a rozhodnutie, aký obsah sa pokúšame odstrániť.

4.7 Filtrovanie vyhľadávacích fráz

Prirodzene vývojári Googlu nespia na vavrínoch a aktívne sa snažia prispieť k zvýšeniu bezpečnosti Googlu a vyhľadávania. Implementácia filtrácie známych fráz, ktoré slúžia k vyhľadávaniu potenciálnych obetí napadnuteľných rôznymi internetovými vírusmi a červami je určite krokom dopredu. Táto metóda blokovania vyhľadávacích dotazov sa taktiež zameriava na rozšírené dotazy pátrajúce po citlivých informáciách, ako napríklad čísla kreditných kariet, ktoré by mohli vážnou mierou poškodiť mnohých užívateľov.

Tento detekčný systém je samozrejme chvályhodný počin, no treba upozorniť na to, že v niektorých prípadoch nie veľmi účinný. A to z jednoduchého dôvodu. Vyhľadávacie dotazy môžu byť veľmi ľahko modifikované, napríklad zmenením poradia kľúčových slov, veľkosti písmen a podobne, čo má za následok oklamanie a znefunkčnenie tohto systému, no na význam hľadania tieto modifikácie často veľký vplyv nemajú.

5 GOOGLE BOMBY

Na mnohých miestach v mojej práci som sa snažil poukázať na ohromný potenciál Googlu. Ten samozrejme nezostáva nepovšimnutý a aj preto nie je núdza o stále nové a nové spôsoby jeho využitia, resp. zneužitia.

Využitie potenciálu tohto vyhľadávača v niečí prospech (a v neprospech niekoho iného) sme v našich podmienkach mohli najviditeľnejšie badať približne v roku 2006 v podobe takzvaných Google bômb, ktoré zasiahli predných českých politikov a vládne inštitúcie. Čo to vlastne Google bomba je?

Google bomba označuje techniku umožňujúcu ovplyvniť internetový vyhľadávač, ktorý potom vracia nerelevantné výsledky [17]. Algoritmus vyhľadávačov je veľmi komplexná záležitosť, musí zohľadňovať radu faktorov a zbierať množstvo informácií o stránkach. Okrem obsahu webu sa obvykle zaznamenávajú aj údaje o odkazoch z iných stránok a obsah týchto odkazov potom hrá svoju úlohu pri zostavovaní výsledkov vyhľadávania. Ak sa teda objaví rada stránok, ktoré budú odkazovať na *www.opensource.cz* s textom odkazu napríklad „český server o open source", bude Google reagovať na hľadanie fráze `"český open source"` odkazom práve na tento server. Tento postup je samozrejme správny, vyhľadávač zohľadňuje aj informácie, ktoré o odkaze uvedú ďalšie zdroje, ale bohužiaľ pomerne dobre zneužiteľný.

Aj preto sme sa mohli stretnúť s tým, že po zadaní vyhľadávacích fráz ako `"namyšlenej buran"`, `"senilní ješita"` alebo `"prasopes"` Google vrátil odkazy na stránky vrcholných predstaviteľov českej vlády. Autor jednoducho vytvoril na internetových stránkach pod konkrétnym označením odkaz vedúci na politikov web a keď bolo podobných odkazov viacej na ďalších stránkach, vyhľadávač vo výsledkoch prisúdil cieľovej stránke na dané slovo väčšiu relevanciu, dôveryhodnosť a následne vyššie umiestenie. Medzi ďalšie známe Google bomby v českom prostredí patria [18]:

`"ministerstvo brutality"`

`"Velký bratr"`

`"zločinci a vrazi"`

Mierne odbočím a spomeniem, že Google pred istým časom začal ponúkať na svojej hlavnej vyhľadávacej stránke službu zobrazenia návrhu dotazu vo vyhľadávacom políčku. To znamená, že v prípade častého vyhľadávania typu

`"[meno politika] je [nelichotivý výraz]"`

sa Google učí tieto vyhľadávacie frázy a následne ich bude ponúkať ako návrh vyhľadávania aj iným užívateľom pri hľadaní `"[meno politika] je"`. A čím viacej bude podobných vyhľadávaní, tým je pravdepodobnejší výskyt tohto návrhu. Tento prístup je na jednej strane užitočný, no na druhej nebezpečný nielen možnou kompromitáciou konkrétnych osôb, ale aj tým, že Google si tým pádom uloží osobné údaje (telefónne čísla), ktoré boli niekedy vyhľadávané a mohol by ich začať ponúkať ako návrh hľadaného dotazu. Skúsil som podobné dotazy aj na zahraničných stránkach Googlu a jednotlivé návrhy sa líšia podľa jazykovej verzie hlavnej stránky Googlu, čo asi nie je až také prekvapivé. Skôr ma zarazilo, že sa Google učí aj nezmyselné výrazy typu náhodných kombinácií písmen, čo len potvrdzuje teóriu o možnom zásahu do osobných údajov.

Google bomby sa začali vyskytovať v prostredí Internetu už od roku 2001 a neubránili sa im ani známe osobnosti v zahraničí. Medzi dotknutých patrili napríklad aj bývalý prezident USA George W. Bush, kedy po zadaní dotazu `miserable failure` do Googlu sa medzi prvými výsledkami objavila stránka s životopisom prezidenta. Túto Google bombu odštartoval nevinný odkaz na blogu nadšenca, ktorý vyzýval ľudí slovami:

„Je to zábava, je to ľahké, stačí vložiť

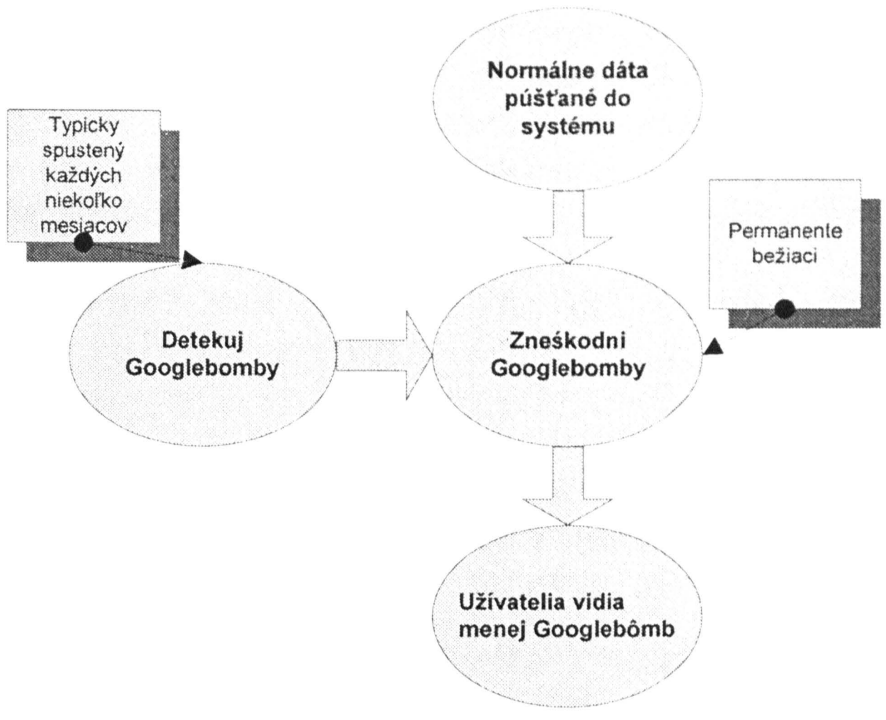
Miserable Failure

na svoju web stránku."

V súčasnosti je výskyt podobných bômb do určitej miery obmedzený, pretože Google sa ich od roku 2007 rozhodol riešiť vyvinutím algoritmu, ktorý má minimalizovať ich dopad a detekovať ich v rôznych jazykoch.

A práve tento algoritmus, respektíve to, ako funguje, je predmetom môjho záujmu. Aj preto, že Google nehovorí ako presne pracuje, iba to, že pracuje automaticky bez akéhokoľvek ľudského zásahu, ktorý by Google bombu eliminoval. Spomínaná Google bomba, ktorá vracala stránku s biografiou prezidenta Busha medzi prvými výsledkami bola odstránená skoro až po dvoch rokoch, ako tento nový algoritmus začal fungovať (v decembri 2008). Avšak krátko na to, v januári 2009, Danny Sullivan, šéfredaktor uznávaného webového portálu Search Engine Land zistil a na svojom blogu publikoval, že tentokrát na slovo `failure` Google vracia podobné výsledky a nie je vylúčené, že sa tak dialo aj dovtedy. A ako Danny vo svojom článku predpokladal, koncom mesiaca (po zdedení odkazov Bieleho Domu) sa na prvé miesta vo výsledkoch po zadaní dotazu `failure` objavil Barack Obama, nový prezident USA. Toto zistenie rozpútalo vášnivú diskusiu a po pár dňoch bola Googlebomba `failure` deaktivovaná. To nám predostiera zaujímavú otázku. Biely Dom nič nezmenil. Dáta odkazujúce na Bushovu stránku, ktoré Google používal pre hodnotenie – dáta zdedené Obamovou stránkou – sa nezmenili. Takže oprava Google bomby, ktorá nefungovala dlhú dobu zrazu začala fungovať po pár hodinách od publikácie článku. Ako je to teda s preklamovanou „automatizáciou" algoritmu na elimináciu Google bômb?

Možné vysvetlenie je nasledovné. Google prevádza dve veci, obidve automaticky - detekciu Google bomby a následné zmiernenie jej dopadu. Druhý algoritmus (zmiernenie dopadu Google bômb) permanentne beží v Google systémoch. Prvý algoritmus (detekcia Google bomby) musí sprocesovať celý web index, takže v najtypickejších prípadoch je zo strany Googlu tendencia nespúšťať algoritmus zakaždým, keď sa prelezú nové web dáta. Toto do určitej miery potvrdil aj šéf webspamového týmu Googlu Matt Cutts na svojom blogu s tým, že v roku 2008 bol Googlebomb detection algoritmus spustený asi 5-6 krát. Na celý algoritmus by sme sa mohli pozerať nasledovne (obrázok 19):



Obr. 19. Proces zneškodnenia Google bomby

Takže algoritmus na zneškodnenie je neustále bežiaci, zatiaľ čo algoritmus na zistenie Google bomby je spúšťaný iba občas. Po priebehu celého procesu detekcie a zneškodnenia Google bomby systém minimalizuje ich dopad a teda vo výsledkoch by mali byť zobrazované najmä články a komentáre o daných dotazoch. Teda na podnet diskusie, ktorá sa odohrala po zverejnení článku Dannyho Sullivana, bola celá procedúra spustená a Google bomby potlačené. Tento algoritmus, ako priznávajú aj sami tvorcovia, však nie je stopercentný, teda ak bol algoritmus spustený v decembri 2008 a Google bomba `failure` nebola odhalená, Google nič neodstraňoval manuálne.

Avšak stále nie je celkom jasné, ako presne nový systém funguje bez toho, aby mal efekt na PageRank, alebo na stránky, ktoré sa snažia odkazovať na seba v rámci SEO (Serach Engine Optimalization). Pohľad na slová, ktoré ľudia používajú, keď sa odkazujú na nejakú inú stránku je kľúčová komponenta ako Google určuje relevanciu. Napríklad, tisíce kvalitných odkazov na Amazon.com s použitím textu *books* v odkaze spravodlivo a správne pomáha portálu Amazon dostať sa na predné pozície vo výsledkoch pre hľadanie `books`. Záver z môjho skúmania, ako by to celé mohlo fungovať, by som zhrnul do zjednodušeného algoritmu:

IF odkaz na stránku obsahuje text odkazu `[ BOMBA ]` **AND**

- Je veľa iných odkazov s `[ BOMBA ]` v anchor texte (texte odkazu) **AND**
- `[ BOMBA ]` je najčastejšie sa vyskytujúci odkaz na stránku **AND**
- Je veľmi málo odkazov tvaru [iné slová ako `BOMBA` / súvisiace frázy ku `BOMBA`]

THEN ignoruj všetky odkazy s `[ BOMBA ]` v texte odkazu

Tento algoritmus by ochránil hodnotenie stránok, ktoré spomínajú daný text aj v tele dokumentu a naopak diskvalifikoval stránky, na ktoré sa odkazuje iba sprostredkovane cez text v odkaze.

Algoritmus, ktorý by riešil podobný problém, bol zverejnený aj v patente Anny L. Patterson, ktorý sa mi podarilo nájsť na stránkach US Patent and Trademark Office [27]. V určitej modifikácii a v kombinácii napríklad s minialgoritmom vyššie by mohol fungovať pre účely identifikácie a zneškodňovania Google bômb. Pokúsim sa naznačiť jeho ideu. Pre každý odkaz z dokumentu URL0 na nejaký dokument URLi indexovací systém vytvorí dvojicu ohodnotení – „outlink" skóre textu odkazu vychádzajúceho z URL0 a „inlink" skóre textu odkazu ukazujúceho na URLi - ktoré závisia na určitých faktoroch. Predpokladajme dva dokumenty URL0 a URL1 a text odkazu A (zo stránky URL0 na URL1). Indexovací systém vytvorí bitový vektor príbuzných fráz dokumentu URL0 pre frázu A – identifikuje príbuzné frázy ku fráze A. Ako, to je ďalšia zo záhad Googlu, ak by prirodzene používal algoritmus založený na vektore príbuzných fráz. Tento bitový vektor bude použitý ako „outlink" skóre pre odkaz z URL0 na URL1 obsahujúci text odkazu A (Anchor text).

„Inlink" skóre sa určí nasledovne. Pre každý odkaz vedúci na URL1 obsahujúci frázu A indexovací systém preskenuje URL1 a rozhodne, či sa fráza A objavuje v tele dokumentu A. V prípade, že áno, je predpokladaná príbuznosť medzi danými stránkami ku konceptu, ktorý predstavuje fráza A. Teda „inlink" skóre pre odkaz vedúci z URL0 na URL1 obsahujúci frázu A predstavuje bitový vektor príbuzných fráz frázy A vo vzťahu k URL1.

V prípade, že text odkazu (fráza) sa nevyskytuje v tele dokumentu URL1, na vytvorenie „inlink" skóre sa použije mierne odlišný postup. Indexovací systém opäť vytvorí bitový vektor príbuzných fráz dokumentu URL1 pre frázu A indikujúci, ktoré z príbuzných fráz frázy A sa objavujú v dokumente URL1. Ako „inlink" ohodnotenie pre odkaz z URL0 na URL1 je potom použitý tento bitový vektor.

Pre lepšie pochopenie a názorný príklad uveďme nasledujúce frázy prítomné v URL0 a URL1, ako ukazuje tabuľka 12:

	Fráza v texte odkazu	**Bitový vektor príbuzných fráz**				
Dokument	Austrálsky pastier	Austrálčan	austrálsky ovčiak	dlhosrstá kolia	trikolóra	tréning agility
URL0	1	1	0	0	0	0
URL1	1	0	1	1	1	0

Tabuľka 12.

Riadok URL0 je „outlink" ohodnotenie pre odkaz s textom odkazu A a riadok URL1 je „inlink" ohodnotenie odkazu. V tomto prípade URL0 obsahuje frázu v texte odkazu „Austrálsky pastier", ktorá ukazuje na URL1. Zo štyroch príbuzných fráz iba jedna – „Austrálčan" – sa vyskytuje v ULR0. Intuitívne teda URL0 asi ťažko pojednáva o austrálskych pastieroch. V porovnaní s tým URL1 nielenže má v tele dokumentu „Austrálsky pastier", ale obsahuje aj ďalšie príbuzné frázy ako „Austrálsky ovčiak" či „trikolóra". Toto bol príklad prvého prípadu a to, že fráza v texte odkazu sa nachádza aj v tele dokumentu.

Druhý prípad, ktorý som popisoval vyššie bol, že odkazujúca fráza sa nevyskytuje v URL1. Indexovací systém teda preskenuje URL1 a určí, ktoré z príbuzných fráz „Austrálčan", „Austrálsky ovčiak" a pod. sú obsiahnuté v URL1 a vytvorí bitový vektor príbuzných fráz približne nasledovne (tabuľka 13):

	Fráza v texte odkazu	**Bitový vektor príbuzných fráz**				
Dokument	Austrálsky pastier	Austrálčan	austrálsky ovčiak	dlhosrstá kolia	trikolóra	tréning agility
URL0	1	1	0	0	0	0
URL1	0	0	1	1	1	0

Tabuľka 13.

Teda vidíme, že URL1 neobsahuje frázu odkazu „Austrálsky pastier", ale obsahuje príbuzné frázy ako „austrálsky vlčiak", „trikolóra" apod.

Ako ďalšie rozšírenie podobného postupu by som možno navrhoval zaviesť váhové ohodnotenie jednotlivých príbuzných fráz nejakou váhovou funkciu, prípadne zapojiť sekundárne príbuzné frázy. Každopádne importovaním bitového vektora súvisiacich fráz z cieľového dokumentu URL1 ku fráze A, bitový vektor súvisiacich fráz pre dokument URL0 eliminuje spoliehanie sa vyhľadávacieho systému iba na vzťah frázy A v URL0 odkazujúcej na URL1 ako indikátor významnosti. Teda jednoducho povedané zamedzí sa, aby po zadaní textu odkazu ako vyhľadávacieho dotazu boli vrátené stránky, ktoré nemajú nič spoločné s touto frázou.

V súvislosti s Google bombami, ale aj celkovo so zvyšovaním PageRanku musia prevádzkovatelia blogov a fór čeliť s tzv. „link spammingom", kedy sú tieto stránky zaplavované zoznamami odkazov. Pre spammerov to má tú výhodu, že nimi zneužité stránky niesú nijak podozrivé. Jedinou proti zbraňou je tu Googlom prezentovaný atribút „nofollow" pre tag Anchor. Takto označené odkazy Google ignoruje a nezaindexuje ich. Spammerov to ale nezastaví – naďalej plnia blogové komentáre v nádeji, že predsa len narazia na blogy, ktoré „nofollow" nepoužívajú. Prevádzkovatelia webových stránok môžu odkazový spam dostať pod kontrolu jedine vlastnoručne napísanými antispamovými rutinami alebo prostredníctvom antispamových služieb. Iní spammeri nasadzujú hneď vlastné blogy, ktorými sa pokúšajú vnútiť Googlu svoje ponuky. Principiálne nejde o nič iného než o klasické „odkazové farmárenie" – vytváranie webových stránok, ktoré pozostávajú z nezmyselného textu a hypertextových odkazov. Aj s tým si však Google už dlho vie poradiť.

Z nových techník vyhľadávačských spammerov spomeniem ešte aspoň „cloaking", ktorý spočíva v zastieraní pravej tváre webovej stránky. Tá potom vyhľadávaču predkladá iný obsah než návštevníkovi. Umožňuje to skutočnosť, že robot Googlu zodpovedný za prehľadávanie webových stránok sa vždy dá rozpoznať (napr. pomocou IP adresy). Ako náhle taká stránka zistí, že ju otvoril „Googlebot", nakŕmi vyhľadávač špeciálnym obsahom, ak však stránku otvorí bežný návštevník, nájde na nej úplne iný, pre publikum vytvorený obsah.

6 PRÍPADOVÉ ŠTÚDIE

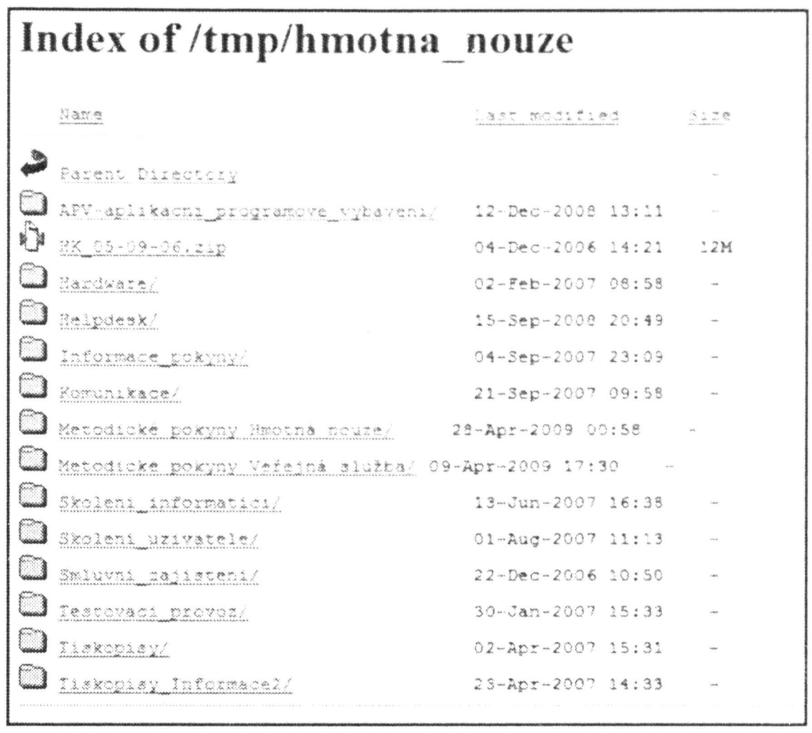
6.1 České prostredie

6.1.1 Web ministerstva podľahol Google hackingu

Prípad začal jednoduchým hľadaním informácií o `policy-map class`. Toto hľadanie vrátilo odkaz na zaujímavý dokument vo formáte PDF *"Připojení pracovišť HN k WAN MPSV – kraj Liberecký"* (obrázok 20), v ktorom okrem iného stálo, že "technické a obchodné informácie v tomto dokumente musia byť držané v tajnosti, materiál ani informácie v ňom obsiahnuté nesmú byť poskytnuté alebo vyzradené inej strane". Samozrejme zaujímavý bol celý obsah zložky *http://www.mpsv.cz/tmp/hmotna_nouze/* (dostupný na obrázku 21).

Obr. 20. Výsledky s odkazom na citlivý dokument (obázok prevzatý z Internetového zdroja)

Prípad sa odohral v januári 2007 a išlo vlastne o nový systém, pracujúci vo WAN MPSV ČR (Ministerstvo práce a sociálních věcí ČR), ktorý dodala istá spoločnosť. Server Lupa.cz, ktorý o prípade informoval a zaoberal sa ním, zavolal v tej dobe najprv vedúcemu projektu, ktorý redaktorov po otázke o získaných dokumentoch odkázal na zodpovednú osobu z MPSV s tým, že on o umiestení dokumentov na webe nič nevie. Osoba z MPSV o prístupnosti dokumentov vedela a označila ho za postup dohodnutý s dodávateľskou spoločnosťou s tým, že bol použitý ako jediná vhodná možnosť ich distribúcie na viac než päťsto obecných úradov obcí s rozšírenou pôsobnosťou.



Obr. 21. Obsah adesára /tmp/hmotná_nouze

Z tohto dokumentu by mohol útočník zistiť veľa užitočných informácií, ktoré by mu v okamihu útoku uľahčili život a zjednodušili prácu. Okrem približnej podoby siete a jej hierarchie boli odhalené predovšetkým IP adresy jednotlivých prvkov, proti ktorým je možné viesť útok. Sprístupnené boli taktiež konfigurácie VPN. Je funkčnosť založená na SSL? Útok Man-in-the-middle a máme získané heslá. Nehovoriac o tom, že v prístupných dokumentoch boli uvedené URL adresy testovacej aplikácie prístupnej cez web, jednotné heslo a v rovnakom adresári aj zoznam loginov. A ako bolo potvrdené od účastníkov diskusie na serveri Lupa.cz, loginy aj heslo skutočne fungovali. A ak by sa nám tieto údaje nezdali dostatočne citlivé, prístup k TACACS (Terminal Access Controller Access-Control System) ním bezpochyby je.

6.2 Svet

6.2.1 Čísla kreditných kariet odhalené pomocou Google archívu

Nasledujúci prípad je zaujímavý okrem iného aj tým, že sa odohral pomerne nedávno – v marci 2009. Austrálsky IT pracovník náhodou odhalil dáta o 22 000 záznamoch kreditných kariet na zaniknutom serveri pomocou Google archívu (o jeho možnosti zneužitia som sa zmieňoval v kapitole 3.1). Okrem zjavne slabej bezpečnosti serveru k tomu dopomohol Google archív (cahce) tým, že stránky mŕtveho web serveru obsahujúce citlivé adresáre zostali archivované a prístupné komukoľvek. O prípade ako prvý informoval austrálsky server iTnews, ktorý mal prístup k daným dátam a podľa vyjadrenia serveru prístupné boli čísla kreditných kariet vrátane expiračných dátumov, mien a adries, čo úplne postačuje pre zneužitia napríklad prostredníctvom nákupu po internete. Viacej ako 19 000 z týchto účtov mohli byť aktívne, čísla kreditných kariet boli pre účty spoločností Visa, Mastercard, American Express, ale aj iné. Google od zverejnenia prípadu blokuje inkriminovaný archív s výpisom adresárov. URL pre archivované adresáre obsahovali rôzne internetové obchody so športovými potrebami alebo oblečením, čo vedie k záveru, že ak to nebol sklad pre ukradnuté účty, jednalo sa o veľmi slabo zabezpečený server spracovávajúci platby.

Osoba, ktorá objavila karty, tvrdila, že objav nastal ako výsledok Google Alertu. Google Alert je služba, ktorá sprostredkováva notifikáciu na e-mail v prípade, ak sa v prvej desiatke výsledkov objaví nová stránka, ktorá odpovedá vyhľadávaciemu dotazu zadanému pri zriaďovaní služby. Ďalej podľa vyjadrenia oznámenie (Alert) začínalo s hromadou čísel, preto užívateľ išiel na web stránky a podľa jeho slov našiel virtuálny výpis adresárov s hromadou súborov. Spoločnosti VISA a Mastercard boli o prípade upovedomené a vyšetrovania sa ujala aj polícia.

Ako vidno, Google občas môže byť obeťou vlastnej efektívnosti tým, že má zaindexovaný všetok obsah od kriminálneho zásobovacieho serveru až po obyčajný spravodajský server. Tento prípad však zďaleka nie je prvý, kedy crawleri Googlu zaindexovali citlivé dáta. V máji 2008 firma Finjan zaoberajúca sa sieťovou bezpečnosťou reportovala podobný prípad, kde bankové prístupové údaje a ďalšie dáta boli uskladňované na crimeware serveri prístupnom pomocou Google vyhľadávacích dotazov (*http://www.finjan.com/MCRCblog.aspx?EntryId=1957*).

6.2.2 Masívny útok na SQL databázy

Tento prípad je špecifický tým, že jeho podstatou bola dnes často podceňovaná a bežná SQL Injection. Slabina poskytuje útočníkovi možnosť pracovať s databázou webového serveru, ktorá často obsahuje všetky dáta, ktoré sú na stránkach použité, vrátane prihlasovacích údajov. Medzi obeťami útoku, ktorý úspešne infikoval viac ako pol milióna webov, sa objavili aj weby s obrovskou návštevnosťou a štátne weby z USA a UK, výlučne postavené na systéme Microsoft IIS, MS SQL a ASP.

Ako odrazový bod útok bol využitý práve Google. Pre uľahčenie práce a vyhnutie sa vytváraniu vlastného programu na testovanie slabín, využili útočníci práve jeden z dotazov, odhaľujúci a zneužívajúci chybu SQL databázy (spomínané v kapitole 2.4) a ktorá je rovnaká pre niekoľko stoviek tisíc webov. Celý útok bol rozdelený na 2 kroky, ktorými sa útočníci chceli zbaviť prílišnej pozornosti. Prvý krok zahŕňal otestovanie zraniteľnosti jednoduchými príkazmi, ktoré by aj pri odhalení nevzbudili žiadnu veľkú pozornosť, maximálne len „opravu“ slabiny na webe. Po zistení prítomnej slabiny na aktuálne testovanom web boli vykonané ďalšie príkazy. Zakaždým boli ukončené jedným, ktorý stránku infikoval nebezpečným kódom.

Ukážka príkladu:

```
id=z;DECLARE%20@S%20NVARCHAR(4000);SET%20@S=CAST(0x440045004300..7200
%20AS%NVARCHAR(4000));EXEC(@S);-
```

čo znamená

```
DECLARE @S NVARCHAR(4000);

SET @S=CAST(0x440045004300..7200 AS NVARCHAR(4000));

EXEC(@S);-
```

V prvom prípade je určená premenná S ako NVARCHAR, čo v MSSQL znamená, že dáta ukladané ako NVARCHAR budú vo formáte UNICODE, ktorý reprezentuje viacjazyčné dáta, čiže jedno z hlavných riešení pre uchovávanie informácií v rôznych jazykoch. V príklade sa do tejto premennej uloží hexadecimálne slovo a jediným dôvodom je sťaženie webadministrátorom prácu – analýza týchto znakov. V príklade funkcia CAST

zmení kódovanie z hexadecimálneho na unicode, čo urobí z ťažko čitateľného reťazca ľahko čitateľný. Následne sa celý príkaz vykoná – vloženie škodlivého kódu do databázy – SQL Injection.

Do databáze je teda vložený kód napísaný v JavaScripte uložený na inom serveri. Začne sa zobrazovať na napadnutej stránke, ktorá návštevníka presmeruje na web *http://www.211796*.net*, kde sa ho pokúsi infikovať malwarom, cez slabiny v prehliadači Internet Explorer. Tento incident je zaujímavý okrem použitia Googlu aj preto, že namiesto získania citlivých dát za pomoci SQL Injection vkladá do databáze škodlivý kód – takzvaný Cross-site scripting, čím sa útočníci snažia vytvoriť časť obrovského botnetu. Útok bol veľmi úspešný, pričom zasiahol obrovské množstvo veľkých portálov, vrátane desiatok štátnych webov [11, 29].

7 ZÁVER

Google hacking svoj najväčší rozmach zaznamenáva najmä od roku 2005 vďaka praotcovi Johnnymu Longovi, ktorý ako prvý upozornil na jeho potenciál a skryté nebezpečenstvo. Google hacking patrí medzi frekventovane rozoberané tematiky v oblasti bezpečnosti, dalo by sa povedať stáva sa fenoménom poslednej doby aj vďaka relatívnej nenáročnosti. Podľa môjho názoru však samotné vyhľadávanie pri Google hackingu predstavuje len akési pozadie, spôsob mapovania kyberpriestoru, kúsky do skladačky, ktoré môžu byť zúročené pri útokoch ostatných typov.

Navyše sila, ktorou Google oplýva a dopad, ktorý majú výsledky ním ponúkané priam motivuje či už individualistov, alebo aj rôzne skupiny k útokom typu Google bômb, k diskusiám o vyhľadávacom engine, PageRanku, o možnostiach ako dostať určitú stránku (či už vlastnú, alebo cudziu) na prvé pozície a podobne.

Dúfam, že sa mi v mojej práci podarilo poukázať na hrozbu, ktorú Google hacking predstavuje a aspoň trochu poodhaliť aj algoritmickú stránku fungovania Googlu a jeho procesov, o ktorých sa až tak veľa nevie, pretože Google ako komerčná firma už podobné informácie drží v tajnosti. Primárnym faktorom umožňujúcim hacking pomocou Googlu je človek. Najmä jeho nedbalosť, alebo nevedomosť spôsobujú, že sa na Internete vyskytujú dôverné informácie, ktoré by mali zostať skryté. Koncepcia získavania osobných informácií a ich následného využitia v oblasti sociálneho inžinierstva teda predstavuje podľa môjho názoru do budúcnosti hrozbu, ktorú je potrebné brať na zreteľ.

Predpokladom pre dosiahnutie čo najväčšieho počtu relevantných výsledkov je najmä použitie vyhľadávača s kvalitnými vyhľadávacími algoritmami a mohutnosť databázy indexovaných dokumentov. V kombinácii s možnosťou automatizácie vyhľadávania a analýzy výsledkov teda Google predstavuje ideálny prostriedok v oblasti vyhľadávania citlivých informácii a aj preto sa ustálil termín „Google hacking“.

Rozmach webu a jeho obsahu so sebou prirodzene prináša aj zväčšovanie databáze Googlu a rozsahu jeho služieb, s čím je spojený aj potenciál Google hackingu do budúcnosti, keďže s tým súvisí aj nárast množstva citlivých informácií získavaných jeho prostredníctvom. Tento trend sa pravdepodobne tak skoro nezmení a teda zraniteľných cieľov a spôsobov získavania citlivých informácii bude neustále pribúdať. Záleží len na šikovnosti a kreativite hackerov, ako si poradia so zadávaním dotazov a ako získané informácie zúročia.

ZOZNAM POUŽITEJ LITERATÚRY

[1] ISKRA, J., Google : vyhledávání, Gmail, Google Talk a další služby, Computer Press, 2006, ISBN 80-251-1043-5.

[2] VISE, D. A., MALSEED, M., Google Story, Paradigma, 2007, ISBN 978-80-7349-034-8.

[3] Internetové stránky spoločnosti ComScore.

Zdroj: http://www.comscore.com/press/release.asp?press=2018

[4] LONG, J., Google hacking, Zoner Press, 2005, ISBN 80-86815-31-5.

[5] Internetové stránky spoločnosti Google – podpora.

Zdroj: http://www.google.com/support/bin/static.py?page=faq.html&hl=cs

[6] DORNFEST, R., BAUSCH, P., CALISHAN, T., Google Hacks: Tips & Tools for Finding and Using the World's Information, O'Reilly Media, 2006, ISBN-13 978-0596527068

[7] Internetové stránky Google Frequenty Asked Questions – File Types.

Zdroj: http://www.google.cz/help/faq_filetypes.html

[8] PIOTROWSKI, M., Nebezpečný Google – vyhledávání důvěrných informací, Časopis Hakin9, 2005.

Zdroj: www.goci.xf.cz/security/ hakin9_4_2005_google_cz.pdf

[9] LONG, J., Google hacking for penetration testers, volume 2, Syngress, 2008, ISBN-13 972-1-59749-176-1.

[10] Internetové stránky Wikipedie.

Zdroj: http://en.wikipedia.org/wiki/Exploit_%28computer_security%29

[11] Internetové stránky Živé.sk.

Zdroj: http://www.zive.sk/Titulna-strana/Internetom-sa-prehnal-masivny-utok-na-SQL-databazy/sc-21-sr-1-a-277145/default.aspx

[12] Internetové stránky Google Cheat Sheet.

Zdroj: http://www.adelaider.com/google-cheat-sheet/

[13] Internetové stránky i-Hacked.com

Zdroj: http://www.i-hacked.com/content/view/23/42/

[14] Internetové stránky The Web Robots.

Zdroj: http://www.robotstxt.org

[15] Internetové stránky Google Hacking Database.

Zdroj: http://johnny.ihackstuff.com/ghdb.php

[16] Internetové stránky Synopsi Blog

Zdroj: http://blog.synopsi.com/2008-02-23/goolagorg-predstavil-google-dork-scanner

[17] Internetové stránky Root.cz

Zdroj: http://www.root.cz/clanky/google-bomby-uz-nevybuchuji/

[18] Internetové stránky Jan Ambrož

Zdroj: http://www.ambroz.org/299502_clanek.php

[19] Internetové stránky iDnes.cz

Zdroj: http://zpravy.idnes.cz/domaci.asp?r=domaci&c=A060315_153840_domaci_ton

[20] Internetové stránky Jargon File

Zdroj: http://www.catb.org/jargon/html/H/hacker.html

[21] Internetové stránky Novinky.cz

Zdroj: http://www.novinky.cz/clanek/120035-jak-vydelava-cesky-hacker.html

[22] Internetové stránky Reboot.cz

Zdroj: http://reboot.cz/howto/hacking/hacker-zlomi-heslo-za-10-sec/articles.html?id=620

[23] LANGVILLE N. A., MEYER D. C., Google's PageRank and Beyond: The Science of Search Engine Rankings, Princeton University Press, 2006, ISBN-13: 978-0-691-12202-1

[24] Internetové stránky Google Guide

Zdroj: http://www.googleguide.com/google_works.html

[25] SMIČKA R., Optimalizace pro vyhledávače - SEO

[26] BRIN, S., PAGE, L., The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Networks and ISDN Systems, 1998

[27] PATTERSON L. A., Phrase-based generation of documen description, United States Patent Application Publication, 2006, Pub. No.: US 2006/0020571 A1

[28] KUBÍČEK, M., Velký průvodce SEO, Computer Press, 2008, ISBN 978-80-251-2195-5

[29] Internetové stránky

Zdroj: http://hackademix.net/2008/04/26/mass-attack-faq/

Knihovna Mat.-fyz. fakulty
Informatické oddělení
Malostranské náměstí 25
118 00 Praha 1