Lokalizirana spletna baza zgoščenih vrednosti

Nejc Škoberne

Viris d. o. o., Likozarjeva ulica 12, SI-1000 Ljubljana
nejc@viris.si

Povzetek Spletne baze preslikav zgoščenih vrednosti v izvorno besedilo niso nič novega. Slaba stran teh baz je, da ne vsebujejo zgoščenih vrednosti besedil, ki bi bila prilagojena različnim jezikom. Cilj te naloge je bil vzpostavitev lokalizirane baze zgoščenih vrednosti besedil, ki so tako ali drugače sestavljena iz besed slovenskega besedišča ter nečrkovnih znakov kot so številke ali ločila. V ta namen smo vzpostavili namenski strežnik, napisali spletno aplikacijo za iskanje in aplikacijo za generiranje zgoščenih vrednosti. Takšna baza je uporabna predvsem za strokovnjake na področju informacijske varnosti, ki izvajajo varnostne preglede in penetracijske teste pa tudi za končne uporabnike, ki bi želeli preveriti varnost svojih gesel.

Keywords: uporabniško geslo, varnost gesel, kriptografska zgoščevalna funkcija, zgoščena vrednost, spletna baza, lokalizacija, iskanje, penetracijski testi

1 Uvod

Avtentikacijska shema z uporabo para [uporabniško ime, uporabniško geslo] je danes kljub dostopnosti drugih, modernejših načinov (pametne kartice, biometrika, enkratna gesla) še vedno zelo uporabljana in zato vredna pozornosti. Tipičen primer uporabe sheme je prijava uporabnika v računalniški sistem. Uporabniško ime služi kot identifikator in mora biti navadno edinstveno za vsak sistem. Nanj so vezani različni viri in dovoljenja, zato tudi ni tajno. Nasprotno pa je uporabniško geslo tajno in bi ga načeloma moral vedeti le njegov lastnik. Zaželeno je, da računalnik z geslom operira čim manj časa, in da se operacije nad njim izvajajo le v delovnem pomnilniku, saj so tako možnosti, da bi nekdo geslo med procesiranjem prestregel, manjše. Kljub temu pa mora računalnik nekako znati preveriti pravilnost gesla, ki ga ob prijavi vnese uporabnik. Ker bi se radi izognili hranjenju gesel na trajnem pomnilniku (trdem disku) računalnika, raje hranimo zgoščene vrednosti teh gesel. Na ta način ostanejo uporabniška gesla tajna, preverjanje gesla pa poteka takole: uporabnik vnese uporabniško ime in geslo, računalnik v delovnem pomnilniku sproti izračuna zgoščeno vrednost vnesenega gesla in jo primerja s shranjeno na trajnem pomnilniku. Če sta zgoščeni vrednosti enaki, je bilo vneseno geslo pravilno, v nasprotnem primeru pa ne.

Da takšna avtentikacijska shema dobro deluje, je potrebno poleg ustrezne varnostne politike (dolžina, kompleksnost gesel) in same implementacije sheme

zagotoviti tudi ustreznost zgoščevalne funkcije, s pomočjo katere generiramo zgoščene vrednosti uporabniških gesel. Bistvena lastnost kriptografskih zgoščevalnih funkcij je, da ni mogoče izračunati njihovega inverza. Edini način za ugotovitev izvornega besedila je primerjava njegove zgoščene vrednosti s predhodno izračunanimi zgoščenimi vrednostmi vnaprej izbranih besedil. Poleg zgoščevalnih funkcij se za namen zgoščevanja uporabniških gesel lahko uporabi tudi algoritme za simetrično šifriranje, kjer je šifrirano besedilo nek v naprej določen niz, ključ pa uporabniško geslo. Knjižnica `crypt(3)`, na voljo na sistemih UNIX, je bila ena prvih, ki je nudila orodja za generiranje zgoščenih vrednosti uporabniških gesel. Sprva so namesto zgoščevalnih funkcij uporabljali prilagojeno različico šifrirnega algoritma DES, kasneje pa so shemo izboljšali tako, da lahko generira zgoščene vrednosti z uporabo zgoščevalnih funkcij MD5, SHA-1, pa tudi Blowfish (samo na nekaterih sistemih). Ker zgoščevalne funkcije navadno uporabljamo tudi za druge namene (digitalno podpisovanje, kontrolne vsote, zgoščevalne tabele), je pomembno, da so hitre. Ravno ta hitrost pa pri uporabi za shranjevanje uporabniških gesel pomeni slabost, saj večja hitrost pomeni možnost učinkovitejšega izvajanja napadov z grobo silo. Da bi se temu izognili, se za hranjenje gesel navadno ne uporabi samih zgoščevalnih funkcij, ampak algoritme, ki te funkcije uporabijo večkrat in na različne načine. Algoritem `md5crypt`, na primer, izračuna kar 1000 zgoščenih vrednosti z zgoščevalno metodo MD5, preden vrne končno zgoščeno vrednost. Sekunda zamika pri preverjanju gesla ob prijavi namreč za uporabnika ni moteča, je pa moteča za izvajalca napada z grobo silo.

Poleg napada z grobo silo poznamo več drugih vrst napadov na zgoščene vrednosti besedil (tako na tiste, ki so neposreden rezultat znanih zgoščevalnih funkcij (MD5, SHA-1 ...) kot na tiste, ki so rezultat algoritmov, ki osnovne zgoščevalne funkcije uporabljajo). V tem članku bom na kratko predstavil tudi napad z uporabo mavričnih tabel, napad z uporabo slovarjev in napad z uporabo baze zgoščenih vrednosti, ki smo jo razvili tudi v okviru te naloge. Ker je pričakovati, da bo pomemben delež uporabniških gesel v slovenskem prostoru vseboval tudi besede iz slovenskega besedišča, je takšno bazo smiselno zasnovati in razviti. Poleg tega so danes cene naprav za hranjenje podatkov (trdi diski) zelo nizke, kar omogoča hranjenje velikega števila preslikav na enem mestu.

Namen te seminarske naloge je zasnovati in razviti orodje, ki bo uporabno za izvajalce penetracijskih testov in za sistemske upravljavce, ki morajo skrbeti ustrezno kvaliteto gesel svojih uporabnikov. Orodje sestavlja baza v naprej pripravljenih preslikav med izvornimi besedili in zgoščenimi vrednostmi, ki so generirane na podlagi pravil, ki jih vzdrževalec baze sestavi z uporabo osnovnih operacij nad besedami.

2 Metode razbijanja gesel

Pod besedno zvezo “razbijanje gesel” imamo v mislih predvsem “pridobivanje izvornega besedila (gesla) s pomočjo njegove zgoščene oziroma šifrirane vrednosti”. Drugi način je neposredno poskušanje uporabe avtentikacijskih parametrov, denimo uporabniškega imena in gesla. Večina prijavnih sistemov pa dan-

danes vsebuje zaščito proti tovrstnim napadom v obliki omejitve števila neuspešnih prijav. V primeru, da je omejitev presežena, je prijava za nekaj časa (ali do posredovanja upravljavca) onemogočena. V okviru tega prispevka se bomo osredinili predvsem na prvi način, kjer imamo na voljo zgoščeno oziroma šifrirano vrednost izvornega besedila. V nadaljevanju si bomo ogledali nekaj najbolj poznanih metod.

2.1 Pametno ugibanje

V primeru, da želimo razbiti geslo katerega koli od večjega števila uporabnikov nekega sistema, ali da osebo ali osebe, katerih gesla želimo razbiti, osebno poznamo, je ta metoda vredna poskusa. V prvem primeru preizkušamo kombinacije uporabniških imen in najpogosteje uporabljenih gesel (primer seznama je v [7]). V drugem primeru pa sklepamo na podlagi informacij, ki jih imamo o uporabniku (osebna imena sorodnikov, datumi rojstev ...). Navadno se izkaže, da je ta metoda zelo učinkovita, saj smo ljudje načeloma zelo neizvirni pri izbiri svojih gesel, če v to nismo prisiljeni s strani sistema.

2.2 Napad z "grobo silo"

Preizkušanju vseh možnih kombinacij nad nekim naborom znakov pravimo napad z "grobo silo" (angl. *brute force attack*), saj v tem primeru uporabljamo zgolj računsko moč računalnika, na katerem napad izvajamo. Po vrsti (abecedno) generiramo besede vedno večjih dolžin nad določenim naborom znakov (črke, številke, ločila, posebni znaki ali njihova kombinacija), za vsako besedo izračunamo zgoščeno ali šifrirano vrednost na enak način kot to počne sistem, ki gesla preverja. Te vrednosti nato primerjamo s seznamom zgoščenih ali šifriranih vrednosti gesel, ki smo jih vnaprej pridobili. Dobra stran te metode je izčrpnost (preizkušamo vse možnosti po vrsti), slaba pa njena časovna zahtevnost, saj ta z dolžino besede eksponentno narašča. Tako je recimo danes z uporabo "grobe sile" razmeroma težko v doglednem času preveriti vsa možna gesla dolžine 8, kjer je nabor znakov sestavljen iz majhnih in velikih črk angleške abecede ter številk. Kratek izračun pokaže, da moramo preveriti $(26+26+10)\cdot 8 \approx 2,18\cdot 10^{14}$ možnosti. Izkaže se, da jih moramo v povprečju preveriti samo polovico, torej nam ostane približno $1,06 \cdot 10^1 4$ možnosti. Če program, s katerim napad izvajamo, zmore preveriti milijon kombinacij na sekundo, bi tak napad trajal približno 3 leta in pol. S porazdelitvijo dela na več računalnikov je mogoče ta čas bistveno skrajšati, če pa namesto 8- uporabimo 9-znakovne besede, čas potreben za izračun vseh možnih kombinacij naraste za $26 + 26 + 10 = 62$-krat, torej na približno 217 let.

2.3 Mavrične tabele

Namesto da zgoščene ali šifrirane vrednosti generiranih besed računamo sproti, lahko najprej generiramo bazo teh vrednosti in nato po njej iščemo. Generiranje

baze vseh možnih vrednosti bi sicer vzelo eksponentno veliko časa (kot pri napadu z "grobo silo"), vendar bi jo lahko ponovno uporabili. Težava pa je tudi v tem, da bi izčrpna baza vseh možnih vrednosti zasedla zelo veliko količino pomnilnika. Mavrične tabele so kompromis — izračunane so le nekatere zgoščene ali šifrirane vrednosti, druge pa se računa sproti s preverjanjem. Primer obsežnega in učinkovitega projekta na tem področju je [1], kjer se za računanje mavričnih tabel uporablja procesorska moč prostovoljcev, ki k projektu pristopijo. Skupaj se gradijo velike mavrične tabele za različne zgoščevalne algoritme, ki so nato na voljo za prenos in uporabo. To je ena izmed bolj učinkovitih metod za razbijanje gesel, saj se izkaže, da je zaradi nizke cene obstojnega pomnilnika (trdih diskov) razmerje med potrebnim prostorom in časom za iskanje zadetkov zelo ugodno.

2.4 Slovarji besed

Verjetnost, da bodo ljudje izbrali smiselne ali vsaj delno smiselne besede za svoja gesla, je večja od verjetnosti, da bodo uporabili naključen niz znakov. To kažejo izkušnje iz varnostnih pregledov informacijskih sistemov. Zato je predvsem v sistemih, kjer ni rigidnih varnostnih politik glede oblike in dolžine uporabniških gesel, smiselno poskusiti z v naprej pripravljenimi slovarji besed. Lahko uporabimo tematske slovarje, slovarje najpogosteje uporabljenih gesel ali besedišča posameznih jezikov. Tako bomo morali izračunati le zgoščene vrednosti določene podmnožice besed, ki je bistveno manjša od množice vseh možnih kombinacij določene dolžine nad nekim naborom znakov. Največji izziv te metode pa je uporabiti dobre slovarje – takšne, ki bodo prilagojeni problemu – uporaba seznama angleških besed bo verjetno bistveno manj uspešna v slovenskem prostoru kot v angleškem oz. ameriškem.

2.5 Kriptoanaliza kriptografskih zgoščevalnih funkcij

Kriptoanaliza je znanstvena disciplina, ki raziskuje postopke dešifriranja skritega besedila brez predhodnega poznavanja ključa. V primeru kriptografskih zgoščevalnih funkcij je cilj kriptoanalize zmanjšati časovno zahtevnost razkritja izvornega besedila s pomočjo zgoščene vrednosti tako, da bo ta manjša od časovne zahtevnosti preiskovanja z "grobo silo". Druga vloga kriptoanalize je iskanje kolizij zgoščevalnih funkcij – različnih izvornih vrednosti, ki imajo enako zgoščeno vrednost. Navedimo primer: marca 2005 so raziskovalci Arjen Lenstra, Xiaoyun Wang in Benne de Weger prikazali konstrukcijo dveh certifikatov v obliki X.509 z različnima javnima ključema in isto zgoščeno vrednostjo MD5, kar je bila dokončna potrditev, da kriptografski zgoščevalni algoritem MD5 ni več primeren za uporabo tam, kjer je potrebna visoka stopnja varnosti.

2.6 Baze vnaprej izračunanih zgoščenih vrednosti

Kot zadnjo opišimo metodo, ki jo uporablja tudi sistem, razvit v okviru našega projekta. Osredinili se bomo predvsem na spletne baze, saj so najbolj pogoste in

preproste za uporabo. Primeri takšnih baz so dostopni na [2], [4], [6] in [3]. Razen v primeru [3] gre povsod za preprost koncept: skrbniki baze generirajo množico besed (način generiranja se lahko od baze do baze razlikuje), izračunajo njihove zgoščene oz. šifrirane vrednosti in obojne vpišejo v bazo, po kateri je mogoče iskati. Nekatere baze so zelo obsežne, recimo [4], ki vsebuje več kot 13 milijard enoličnih preslikav iz zgoščenih vrednosti MD5 v izvorna besedila. Razlikujejo se tudi v naboru zgoščevalnih funkcij oz. šifrirnih algoritmov, ki jih podpirajo. Baza [4] tako vsebuje samo preslikave za zgoščevalno funkcijo MD5, medtem ko baza [2] vsebuje zgoščene vrednosti za algoritme MD5, SHA-1, SHA-224, SHA-256, SHA-384 in SHA-512.

3 Motivacija

Pri izvajanju varnostnih pregledov in penetracijskih testov informacijskih sistemov večkrat naletimo na datoteke z avtentikacijskimi podatki, kjer so gesla navadno predstavljena v obliki zgoščenih oz. šifriranih vrednosti. Eden ključnih ciljev pri izvajanju penetracijskega testa je prav pridobivanje gesel uporabnikov. Drug primer uporabe orodij oz. metod za iskanje izvornih besedil s pomočjo zgoščenih oz. šifriranih vrednosti je preverjanje varnosti gesel, ki ga lahko rutinsko izvaja sistemski administrator (oz. sistem) ali pa uporabniki sami. Primer podobne spletne storitve za uporabnike je na voljo na [5]. Pri iskanju si lahko pomagamo z metodami, opisanimi v prejšnjem poglavju. Kljub temu pa smo želeli naše iskanje nadalje optimizirati, zato smo razvili prilagojeno spletno bazo zgoščenih vrednosti, ki ima v primerjavi z drugimi že obstoječimi sorodnimi bazami določene prednosti.

3.1 Lokalizacija

Trenutno obstoječe baze načeloma niso vezane na noben fizični prostor in tako večinoma vsebujejo preslikave v izvorna besedila v angleškem jeziku ali v naključna izvorna besedila določenega števila znakov. Pomembna lastnost naše baze je prilagojenost na prostor, kjer se baza uporablja. V našem primeru smo se osredinili na slovenski prostor, vendar se jo lahko brez posebnih težav prilagodi tudi drugim prostorom. Pričakujemo namreč lahko, da bo nek delež gesel v slovenskem prostoru vseboval ali bil sestavljen izključno iz besed besedišča maternega jezika uporabnikov, v našem primeru slovenskega jezika. To pomeni, da besede vključujejo tudi znake, specifične za ta jezik, torej šumnike š, č, ž, Š, Č, Ž in znake đ, ć, Đ in Ć, ki jih je tudi mogoče natipkati s slovensko tipkovnico. Takih besed druge baze navadno ne vsebujejo.

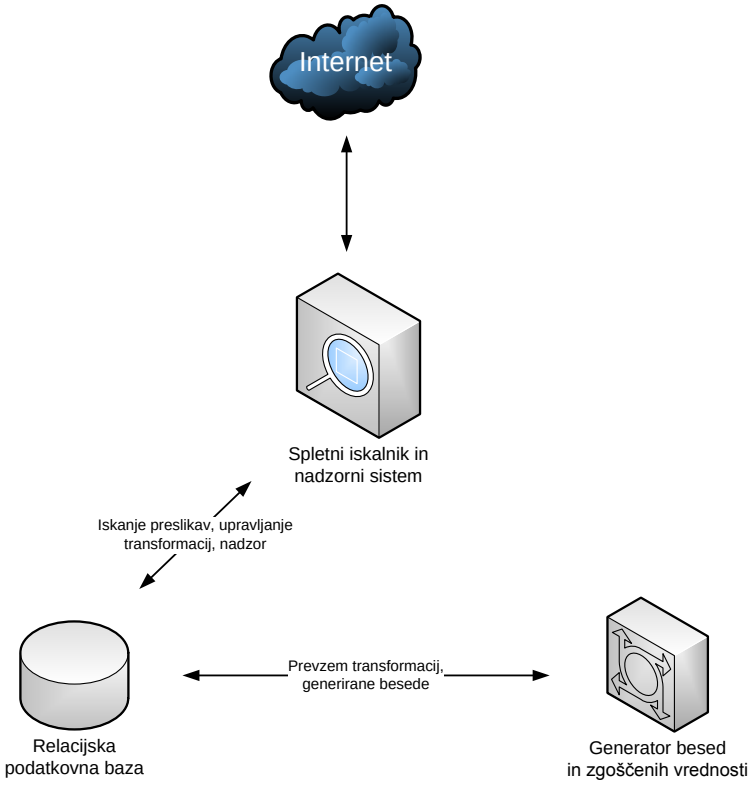
3.2 Transformacije

Avtentikacijski sistemi od uporabnikov pogosto zahtevajo nastavitev gesel, ki ne vsebujejo zgolj črk ampak tudi druge znake, recimo številke in ločila. Zato je smiselno v naprej izračunati tudi preslikave besed, ki vsebujejo dodatne znake. V

ta namen smo definirali nabor transformacij, ki omogočajo tvorbo besed oblike beseda123, Beseda345, 23adeseB ipd.

4 Opis lokalizirane spletne baze hashdb.si

Lokalizirana spletna baza zgoščenih vrednosti hashdb.si [10] je sestavljena iz več komponent, kar prikazuje Slika 1: spletni strežnik in relacijska podatkovna baza nudita osnovno infrastrukturo za dve aplikaciji: generator besed in zgoščenih vrednosti, ki polni bazo ter spletni iskalnik in nadzorni sistem, ki omogoča iskanje preslikav iz zgoščenih vrednosti v izvorna besedila ter upravljanje transformacij in pregled nad stanjem baze. Trenutno vse tri komponente tečejo na istem fizičnem strežniku, vendar je mogoča tudi razdelitev vlog na več fizičnih strežnikov, kot je nakazano v zadnjem poglavju.



Slika 1. komponente lokalizirane spletne baze HashDB.si

4.1 Generator besed in zgoščenih vrednosti

Bazo smo generirali s pomočjo slovarja besed slovenskega besedišča, ki je del programske opreme GNU Aspell. To je odprtokodni črkovalnik, ki vsebuje dva slovarja besed – pravilne in nepravilne oblike besed slovenskega besedišča. Oba slovarja smo združili v en slovar, ki služi kot osnovni slovar za generiranje besed, ki so vsebovane v naši bazi. Število besed v tem osnovnem slovarju je 1.174.305. Nad besedami v osnovnem slovarju lahko izvajamo vrsto transformacij, ki jih zapišemo s pomočjo pravil. Pravilo moramo vnesti v bazo, generator pa nato glede na vneseno pravilo prične generirati preslikave in jih zapisovati v bazo. Trenutno podprte transformacije so naslednje:

- I: inverzna transformacija (angl. *inverse*), zamenja vrstni red znakov v besedi, primer: *beseda* $\rightarrow$ *adeseb*,
- An: dodajanje pripone (angl. *append*), doda števila od 00...0 do $10n - 1$ na konec besede,
- Pn: dodajanje predpone (angl. *prepend*), doda števila od 00...0 do $10n - 1$ na začetek besede,
- Rn, m: dodajanje medpone (angl. *insert*), doda števila od 00...0 do $10n - 1$ na mesto m v besedi,
- C: kapitalizacija prve črke besede,
- L: kapitalizacija zadnje črke besede,
- Z: kapitalizacija vseh črk v besedi.

Primer zapisa pravila, ki najprej zamenja vrstni red črk v besedi, nato kapitalizira zadnjo črko te besede ter na koncu besede doda števila od 000 do 999 je naslednji: $ILA3$. Generator bi s pomočjo tega pravila iz besede *beseda* generiral 1000 besed: *adeseB000*, *adeseB001* ... *adeseB999*. Vsem generiranim besedam, ki jih generator tvori s pomočjo pravil sproti izračuna še pripadajoče zgoščene vrednosti za podprte zgoščevalne funkcije in jih vpiše v obliki parov v ustrezne tabele v bazo. Trenutno sta podprti zgoščevalni funkciji MD5 in SHA-1, nadgradnja celotnega sistema z drugimi pa je razmeroma preprosta.

4.2 Relacijska podatkovna baza

Preslikave in pravila sistem hrani v relacijski podatkovni bazi MySQL. Za vsako vrsto zgoščevalne funkcije obstaja ločena tabela, ki vsebuje pare [*izvorno besedilo*, *zgoščena vrednost*] in za vsak par tudi identifikator pravila, z uporabo katerega je bil generiran. Pravila so shranjena v ločeni tabeli, kjer je za vsako shranjen identifikator, pravilo sámo zapisano z zgoraj opisano sintakso in stanje pravila, ki pove, kolikšen delež preslikav, ki bodo (so) generirane s tem pravilom, se že nahaja v bazi.

4.3 Spletni iskalnik in sistem za upravljanje

S pomočjo spletnega iskalnika lahko uporabniki iščejo med preslikavami v bazi. Uporabnik v iskalno okno vpiše zgoščeno vrednost, za katero ne ve izvornega

besedila, iskalnik zahtevo posreduje podatkovni bazi, ki vrne ustrezen zadetek ali javi zgrešitev. Na podlagi dolžine vnesene zgoščene vrednosti iskalni algoritem ugotovi, za katero vrsto zgoščevalne funkcije gre in na podlagi tega ustvari poizvedbo, ki se izvede na ustrezni tabeli v bazi.

Spletni iskalnik nudi tudi pregled nad pravili in njihovimi stanji, na podlagi katerih je bila baza ustvarjena. S pomočjo posebnega obrazca lahko uporabniki predlagajo svoja pravila, ki pa jih mora pred vnosom v bazo potrditi upravitelj sistema, saj bi v nasprotnem primeru lahko prišlo do neustreznih pravil (množica besed, ki naj bi jih pravilo generiralo, bi lahko bila prevelika). Smiselno bi bilo tudi implementirati samodejno zaščito proti neustreznim pravilom, ki bi zavrgla vsa pravila, ki generirajo število besed, večje od dovoljenega, kot neustrezna.

Upravitelj sistema ima preko posebne strani možnost tudi neposredno upravljati s pravili (jih brisati in jih neposredno dodajati). Poleg tega je mogoče izbrisati tudi preslikave, ki so bile (lahko tudi samo delno) generirane s pomočjo določenih pravil.

5 Zaključek

Naš sistem trenutno še ni javno dostopen, zato ni mogoče govoriti o povratnih informacijah s strani uporabnikov, ki bi lahko podali oceno o uporabnosti in delovanju sistema. Ko bo to mogoče, bomo ponovno ovrednotili nekatere smernice, ki smo se jih držali pri razvoju, in ki so bile določene tudi s pomočjo intuicije.

Možnosti za nadgradnjo lokalizirane spletne baze je več. Trenutno je mogoče iskati samo eno samo vrednost na enkrat. Morda bi bilo smiselno sistem razširiti tako, da bi omogočal iskanje več vrednosti hkrati. Druga možnost je sprotno generiranje in prikazovanje statistike iskanj, ki bi bila uporabna predvsem kot zanimivost. Nadalje bi bilo morda smiselno podpreti tudi dodajanje lastnih preslikav v ločeno bazo, ki bi bila prav tako vključena v iskalni algoritem. Na ta način bi uporabniki sami prispevali znanje v bazo.

Generiranje velikega števila preslikav bi lahko v razmeroma kratkem času povzročilo zapolnitev trenutnih pomnilniških kapacitet. Da bi se temu izognili bi bilo potrebno v prvem koraku ločiti posamezne komponente sistema – za podatkovno bazo bi bilo smiselno uporabiti ločen ali več ločenih strežnikov, kar bi omogočalo tudi nadgradljivost v prihodnje.

6 Zahvala

Operacijo delno financira Evropska unija, in sicer iz Evropskega socialnega sklada. Operacija se izvaja v okviru Operativnega programa razvoja človeških virov za obdobje 2007 — 2013, 1. razvojne prioritete: Spodbujanje podjetništva in prilagodljivosti, prednostne usmeritve 1.1.: Strokovnjaki in raziskovalci za konkurenčnost podjetij.

Literatura

1. Free rainbow tables: Distributed rainbow table project, http://www.freerainbowtables.com
2. Hashdatabase, http://www.hash-database.net
3. md5crack, http://www.md5crack.com
4. Md5decrypter.co.uk, http://www.md5decrypter.co.uk
5. Microsoft online safety, https://www.microsoft.com/protect/fraud/passwords/checker.aspx
6. Passcracking.ru, http://passcracking.com
7. The top 500 worst passwords of all time (Nov 2008), http://www.whatsmypass.com/the-top-500-worst-passwords-of-all-time
8. Burnett, M.: Perfect Passwords: Selection, Protection, Authentication. Syngress (2005)
9. Menezes, A.J., Oorschot, P.C.V., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
10. Skoberne, N.: Lokalizirana spletna baza HashDB.si (Jan 2010), http://www.hashdb.si