Masarykova univerzita
Fakulta informatiky

Dynamické heslá

Bakalárska práca

Filip Guzma

Brno, jar 2016

Prehlásenie

Prehlasujem, že táto bakalárska práca je mojím pôvodným autorským dielom, ktoré som vypracoval samostatne. Všetky zdroje, pramene a literatúru, ktoré som pri vypracovaní používal alebo z nich čerpal, v práci riadne citujem s uvedením úplného odkazu na príslušný zdroj.

Filip Guzma

Vedúci práce: Ing. Mgr. et Mgr. Zdeněk Říha, Ph.D.

Poďakovanie

Rád by som poďakoval svojmu školiteľovi Ing. Mgr. et Mgr. Zdeňkovi Říhovi, Ph.D. za všetky rady, ochotu a pomoc pri tvorbe tejto práce a mojim najbližším za trpezlivosť a motiváciu pri tvorbe tejto práce.

Zhrnutie

Cieľom bakalárskej práce je návrh a implementácia systému autentizácie užívateľa pomocou dynamických hesiel.

Navrhli sme systém dynamickej autentizácie užívateľov využívajúcu časové premenné a vytvorili 33 dostupných funkcií, ktoré je možné použiť pri tvorbe dynamického hesla. V práci analyzujeme vytvorený systém, opisujeme a porovnávame existujúce alternatívne systémy autentizácie užívateľa s našou metódou.Vytvorený systém sme implementovali pomocou technológie JAAS.

Kľúčové slová

autentizácia užívateľa, dynamická autentizácia užívateľa, dynamické heslá, dátum a čas, časové heslá, časové funkcie

Obsah

1 **Úvod** 1
2 **Autentizácia** 3
2.1 *Statická autentizácia užívateľa* 3
2.2 *Biometrická autentizácia užívateľa* 4
2.3 *Dynamická autentizácia užívateľa* 4
2.4 *Viacfaktorová autentizácia užívateľa* 4
3 **Prelomenie hesla** 7
3.1 *Techniky prelomenia a odpozorvania statických hesiel* 7
3.1.1 Guessing 7
3.1.2 Shoulder surfing 8
3.1.3 Phishing 8
3.1.4 Pharming 8
3.1.5 Keylogging 9
3.2 *Ďalšie techniky* 10
3.2.1 Útok hrubou silou 10
3.2.2 Slovníkový útok 11
4 **Dynamické heslá** 13
4.1 *Systém autentizácie užívateľa pomocou hesla s dynamickým údajom* 13
4.2 *Reprezentácia vzoru hesla* 14
4.2.1 Gramatika generujúca heslo 15
4.3 *Ukladanie vzoru hesla* 16
4.4 *Vlastnosti vytvorenej metódy* 17
4.5 *Zoznam dostupných metód* 18
4.5.1 Časové a dátumové metódy 19
4.5.2 Reťazcové metódy 21
4.5.3 Boolean funkcia 22
4.5.4 Matematické operácie nad náhodnými dátami . 22
4.5.5 Ukážky použitia 23
5 **Porovnanie systému s alternatívnymi systémami pre zadávanie tajnej informácie** 25
5.1 *Virtual password using random linear functions for on-line services, ATM machines, and pervasive computing* 25
5.2 *How to fend off shoulder surfing* 26

5.3 *Spy-resistant keyboard: more secure password netry on public touch screen displays* . . . 26
5.4 *Reducing Shoulder-surfing by Using Gaze-based Password Entry* . . . 27
5.5 *PassShape* . . . 28
5.6 *One- time password* . . . 28
5.7 *A Hybrid Password Authentication Scheme Based on Shape and Text* . . . 30
5.8 *DPASS – Dynamic Password Authentication and Security System using Grid Analysis* . . . 30
5.9 *Evaluation of Eye-Gaze Interaction Methods for Security Enhanced PIN-Entry* . . . 31
5.10 *TimePasscode Pro* . . . 32
6 **Použité technológie** . . . 35
6.1 *JAAS* . . . 35
6.2 *PAM* . . . 35
6.3 *JEval* . . . 36
6.4 *NetBeans IDE 8.1* . . . 36
7 **Záver** . . . 39
Literatúra . . . 41

1 Úvod

V dnešnej digitálnej dobe sa s autentizáciou užívateľa stretáva takmer každý z nás. Medzi bežne používané metódy môžeme zaradiť využitie biometrickej autentizácie užívateľov, ale aj napríklad použitie tokenov či karty grid. Avšak základnou a najpoužívanejšou metódou je autentizácia užívateľa pomocou prihlasovacieho mena a statického hesla. Táto metóda však sama o sebe neposkytuje dostatočnú ochranu, hlavne kvôli príliš pohodlným respektíve neopatrným užívateľom, ktorí často volia príliš jednoduché heslá a nedávajú dostatočný pozor kde a ako toto heslo zadávajú, preto je jej bezpečnosť umelo navyšovaná rôznymi opatreniami ako napríklad minimálnou dĺžkou hesla alebo použitím kombinácie znakov malej a veľkej abecedy a čísel. Avšak všetky tieto opatrenia sú zbytočné za predpokladu, že útočník už dané heslo pozná. Nakoľko sa využíva statické heslo, ktoré je v čase nemenné, odpozorovanie alebo prelomenie tohto hesla v praxi znamená, že útočník má neobmedzený prístup k danému účtu častokrát bez toho aby o tom dotknutý užívateľ vôbec vedel. Tento problém sa snaží riešiť dynamická autentizácia užívateľa, vďaka ktorej sa zadávané heslo dynamicky mení. V prípade zistenia hesla získa útočník len jedno platné heslo viažuce sa k danej situácii. Či už sa heslo viaže k času a dátumu alebo ku vygenerovanej mriežke či inému dynamickému faktoru, útočník nemusí byť schopný získať úplný prístup ku chráneným informáciam. Avšak doteraz sa žiadna z vytvorených metód dynamickej autentizácie užívateľa výraznejšie nepresadila.

Cieľom tejto práce je analyzovať už vytvorené riešenia v oblasti dynamických hesiel a navrhnúť vlastný systém dynamických hesiel. Ďalším cieľom je porovnať novovytvorený systém s existujúcimi riešeniami dynamických hesiel a implementovať ho pomocou technológie JAAS.

Táto práca navrhuje nový spôsob autentizácie užívateľa, ktorý je dostatočne rezistentný voči prelomeniu hesla jednou z dnes často využívaných techník krádeží identity.Tento systém pracuje s časovými premennými, ktoré robia toto heslo dynamicky sa meniace v čase. Pri tvorbe tohto hesla je možné využiť číselné, reťazcové a boolovské funkcie spolu so statickými časťami, ktoré vhodnou kombináciou vytvárajú dostatočne odolné heslo. Tento systém autentizácie užívateľa

implementujem pomocou programovacieho jazyka Java a technológie JAAS. Vytvorený prototyp slúži ako ukážka možného fungovania tejto autentizácie užívateľa. Samozrejme, daný systém je možné využiť naprieč celým spektrom autentizácií užívateľa, či sa jedná o autentizáciu užívateľa na internete, zadávanie pin kódu pri využívaní platobnej karty alebo odomykania mobilného telefónu, či osobného počítača. Touto prácou by som chcel dosiahnuť zvýšenie bezpečnosti pri využívaní autentizácie užívateľa. Nakoľko si myslím, že prihlasovanie pomocou prihlasovacieho mena a statického hesla neposkytuje aj napriek mnohým protiopatreniam dostatočnú úroveň ochrany a získanie citlivých dát je dnes stále jednoduché. O to viac, ak sa jedná o nepozorného užívateľa.

Samozrejme, najkrajším výsledkom tejto práce by bolo jej globálne uplatnenie respektíve globálne uplatnenie základnej myšlienky tejto práce v praxi.

V druhej kapitole približujem autentizáciu užívateľa a jednotlivé druhy autentzácie užívateľa, s ktorými je dnes viac či menej možné sa stretnúť. V tretej kapitole opisujem techniky zamerané na prelomenie a odpozorvanie hesla. V štvrtej kapitole predstavujem mnou navrhovaný systém, jeho vlastnosti a schopnosti, zoznam metód, ktoré je možné využiť pri tvorbe dynamického hesla, ukážky použitia a opisujem technológie, ktoré som využil pri tvorbe tohto prototypu. V piatej kapitole porovnávam navrhovaný systém s ostatnými alternatívnymi systémami autentizácie užívateľa.

2 Autentizácia

Autentizácia, pôvodom z gréckeho slova authentikos (skutočný, pôvodný) je akt potvrdzovania pravosti atribútu. V našom ponímaní môžme povedať, že autentizácia je proces overovania prehlasovanej identity subjektu. K autentizácii užívateľa sa využívajú tri základné princípy:

1. čo užívateľ vie – heslo, pin kód,
2. čo užívateľ má – token, grid karta,
3. čo užíveteľ je – biometrické údaje (odtlačky prstov).

Na základe dôležitosti systému respektíve operácií, ktoré môže autentizovaný užívateľ vykonávať, sa menia aj použitia jednotlivých princípov autentizácií užívateľa.[1] [2]

2.1 Statická autentizácia užívateľa

Dnes je základným štandardom prihlasovanie pomocou loginu a statického hesla. Touto kombináciou je dnes možné dostať sa od užívateľského konta v počítači, cez prístup k emailu a jednotlivým účtom na sociálnych sieťach až k výpisu z bankových účtov. Pri používaní pinkódu pri výberoch z bankomatov alebo platení v obchodoch je tiež využívané statické heslo. Pre online manipuláciu s financiami sa však už používajú pokročilejšie stupne autentizácie užívateľa ako je napríklad kombinácia loginu a hesla s grid kartou alebo overacou sms. No aj z menej dôležitých zdrojov ako je napríklad email, účty v internetových obchodoch a na sociálnych sieťach a podobne vie útočník získať citlivé a častokrát aj kompromitujúce informácie. Jediné, čo potrebuje útočník urobiť, je iba raz odsledovať vaše heslo k danému účtu jednou z dnes už bežných techník, ktoré fungujú na statické heslá. A aby toho nebolo málo, užívatelia sú často neopatrní a používajú rovnaké respektíve veľmi podobné heslá na rôznych účtoch. Teda jedným odpozorovaním hesla môže útočník získať prístup nielen k jednému účtu, ale aj ku všetkým ostatným kde je nastavené totožné alebo veľmi podobné statické heslo. Avšak keď sa na to pozrieme z druhej strany:

bežný úživateľ má minimálne dva emailové účty, účty na sociálnych sieťach, účty v online obchodoch, účty k pracovným záležitostiam a podobne. Mať pre každý jeden účet odlišné heslo je síce vysoko odporúčané a tiež bezpečné, avšak prakticky takmer nerealizovateľné.

2.2 Biometrická autentizácia užívateľa

Biometrická autentizácia užívateľa je spôsob autentizácie na základe biologických vlastností užívateľov. Pri tomto spôsobe autentizácie užívateľa sa využíva princíp čo užívateľ je. Biometrické údaje, teda údaje, pomocou ktorých vykonávame biometrickú autentizáciu užívateľa, delíme na behaviorálne a fyziologické. Behaviorálne údaje sú charakteristiky spojené so správaním užívateľa. Medzi tieto charakteristiky radíme napríklad rytmus písania na klávesnici, analýzu chôdze, analýzu hlasu. Fyziologické údaje sú charakteristiky založené na tvaroch tela respektíve v tele. Medzi tieto charakteristiky radíme napríklad odtlačky prstov, žily dlane, rozpoznávanie tváre, DNA analýzu, geometriu ruky, skenovanie dúhovky a sietnice. Biometrická autentizácia sa využíva napríklad v dochádzkových systémoch, ale aj v dokladoch totožnosti. [3][4]

2.3 Dynamická autentizácia užívateľa

Táto autentizácia užívateľa využíva na prihlásenie do daného systému konštantné užívateľské meno a heslo, ktoré sa mení. Dynamická autentizácia užívateľa funguje na princípe challenge–response protokolu. Heslo je založené na základe tajnej informácie, ktorá umožňuje užívateľovi, ktorý túto informáciu pozná, úspešne sa autentizovať v systéme na základe výzvy odoslanej systémom užívateľovi. Dynamická autentizácia bude bližšie priblížená v ďalších častiach tejto práce.[5]

2.4 Viacfaktorová autentizácia užívateľa

Viacfaktorová autentizácia užívateľa garantuje prístup po úspešnom zadaní niekoľkých samostatných autentizácií užívateľov. Najčastejšie sa využíva kombinácia minimálne dvoch z vyššie uvedených princí-

pov autentizácie (dvojfaktorová autentizácia) užívateľa, teda čo užívateľ vie, čo má a čo je. [6]

3 Prelomenie hesla

Pod pojmom prelomenie hesla si môžeme predstaviť proces zistenia alebo uhádnutia hesla z dát, ktoré boli uložené v počítačovom systéme alebo prenesené cez počítačový systém. Avšak dôvod prelomovania hesla nemusí byť len za účelom získania citlivých údajov. Určite každý pozná situáciu, kedy zabudol svoje heslo a teda sa nemôže úspešne prihlásiť do daného systému. Najskôr vyskúša najpravdepodobnejšie možnosti hesiel, čo môžeme prirovnať k technike guessingu, respektíve útoku hrubou silou alebo slovníkovému útoku. Tieto vyššie uvedené techniky však môže zneužiť útočník na získanie používateľovho hesla.

3.1 Techniky prelomenia a odpozorvania statických hesiel

Dnes vďaka rôznym nástrojom a internetovým návodom sa môže naučiť prelomiť heslo ľubovoľný počítačovo zdatný užívateľ, ktorý má chuť a čas, respektíve silný záujem na odhalení hesla. Existuje mnoho techník, ktoré sú až triviálne jednoduché a zároveň pre neopatrného používateľa extrémne nebezpečné. Samozrejme, je takmer nemožné vymenovať všetky použiteľné techniky prelomenia hesla, pretože mnoho z nich je možné použiť len za veľmi konkrétnych podmienok. Preto nižšie uvádzam základné techniky, ktoré tvoria základné stavebné kamene mnohým odvodeným technikám.

3.1.1 Guessing

Jednoduchá techinka pracujúca s faktom, že užívateľa poznáme respektíve poznáme jeho osobné informácie, prípadne sme zachytili istú časť hesla. Medzi tieto informácie patria napríklad mená partnerov, detí, domácich miláčikov, dátum narodenia a podobne. Technika negarantuje zistenie hesla, avšak v mnohých prípadoch môže byť útočník úspešný, hlavne ak sa jedná o neskúseného užívateľa alebo užívateľa, ktorý si myslí že prelomovanie hesla a krádeže identity sa odohrávajú len v amerických akčných filmoch.

3.1.2 Shoulder surfing

Pod pojmom shoulder surfing si môžeme predstaviť asi najjednoduchšiu techniku zistenia hesla (teda ak nepočítame to, že nám užívateľ svoje heslo prezradí) a to odpozorovanie hesla pomocou obrazu, teda buď voľným okom alebo ukrytou kamerou či iným zariadením. Táto technika je veľmi častá hlavne pri používaní platobných kariet či už v bankomatoch, kde je využívaná hlavne skrytá kamera alebo pri platbách platobnou kartou, kde pri nepozornom užívateľovi nie je ani pre neskúseného útočníka problém odpozorovať pin kód.[7]

3.1.3 Phishing

Technika získavania osobných údajov ako sú heslá, čísla kreditných kariet, pinkódov a podobne na internete. Táto technika je založená na rozposielaní falošných emailov alebo instatných správ, v ktorých sa útočník predsavuje ako dôveryhodná autorita. Bežným príkladom je napríklad email z banky s textom nútiacim sa prihlásiť cez odkaz v emaili, ktorý vedie na falošnú stránku vyzerajúcu totožne ako originálna stránka banky. Po zadaní prihlasovacích údajov získava útočník vaše citlivé údaje bez toho aby ste to spozorovali. Toto je však len ukážkový príklad. Medzi ďalšie techniky založené na tomto princípe patrí napríklad vishing – voice phishing či evil twin. Pri vishingu sa jedná o správy najčastejšie od banky nabádajúce užívateľa aby zavolal na dané telefónne číslo, kde pre prihlásenie budú od neho vyžadovať pod zámienkou overenia identity číslo účtu a heslo. Evil twin je technika vytvárajúca verejnú sieť vystupujúcu ako legitímna verejná sieť, ktorú je možné nájsť napríklad na letiskách, v hoteloch alebo kaviarňach. Po prihlásení do siete sa podvodníci snažia zachytiť citlivé údaje, ktoré cez túto sieť užívateľ posiela.[8][9][10]

3.1.4 Pharming

Technika, ktorá podobne ako phishing využíva podvrhnuté stránky vyzerajúce veľmi podobne až totožne s pôvodnou stránkou. Avšak na rozdiel od phishingu funguje pharming na princípe zmeny IP adries v DNS a tým spôsobí presmerovanie užívateľa na podvrhnutú stránku. Útočník vyhľadá slabšie zabezpečené DNS servre, ktoré prekladajú URL adresu stránky do podoby IP. Pre lepšiu ilustráciu si DNS

môžeme predstaviť ako telefónny zoznam, v ktorom namiesto mena kontaktu je URL adresa a namiesto telefónneho čísla IP adresa. Útočník tento zoznam podvrhne, teda zamení IP adresu pri URL adrese banky za podvrhnutú stránku. Pri zadaní URL adresy banky užívateľom sa načíta podvrhnutá stránka vyzerajúca na nerozoznanie od pôvodnej stránky. [11]

3.1.5 Keylogging

Technika zaznamenávajúca stlačenia klávesnice typicky vykonávaná tak, že užívateľ, ktorý klávesnicu používa nevie o tom, že je monitorovaný. Existuje viacero metód keyloggingu, základné delenie predsavujú softwarové a hardwarové keyloggere. Softwarové keyloggere môžeme charakterizovať ako počítačové programy, ktoré však nemusia byť využívané len útočníkom na získanie citlivých informácií. Softwarové keyloggere sú používané aj v IT organizáciách na riešenie technických problémov s počítačmi a firemnými sieťami. Tieto keyloggere môžu zaznamenávané dáta ukladať do konkrétneho súboru v počítači alebo odosielať útočníkovi priamo do emailovej schránky. Pri hardwarových keyloggeroch nepotrebujeme mať v inkriminovanom počítači nainštalovaný žiadny program a je nezávislý od operačného systému nakoľko sa jedná o hardwareovú časť. Získané dáta sa uložia do vnútornej pamäte zariadenia, ktorá sa môže pohybovať od niekoľko kilobajtov až po niekoľko gigabajtov.Vo všeobecnosti sa dáta z tohto zariadenia získajú pomocou špeciálneho hesla, ktoré keylogger rozozná a do textového editoru vypíše zaznamenané dáta. Medzi základné hardwareové keyloggerre radíme PS/2 keylogger, ktorý sa zapojí medzi klávesnicu a počítač. Avšak tento typ dnes už nie je veľmi používaný, nakoľko väčšina dnešných klávesníc využíva USB port. Preto je dnes rozšírenejší hardwarový keylogger typu USB alebo sniffer, teda bezdrôtový keylogger, ktorý zbiera dátové pakety odosielané medzi bezdrôtovou klávesnicou a prijímačom a následne sa snaží prelomiť dešifrovací kľúč, ktorý zabezpečuje tieto pakety.[12]

3.2 Ďalšie techniky

Pri všetkých týchto technikách je možné získať statické heslo bez toho, aby užívateľ niečo spozoroval. Útočnikovi stačí využiť len jednu nepozornosť užívateľa na to, aby získal jeho citlivé údaje, ktoré môže v budúcnosti zneužívať. Samozrejme, každá z týchto techník je aplikovateľná aj na mnou navrhovaný dynamický systém, avšak pomocou týchto techník získa len jednorázové heslo platné len pre daný časovodátumový okamih. Vo vyššie uvedenom zozname úmyselne neuvádzam techniky útoku hrubou silou a slovníkového útoku, nakoľko za predpokladu dobre zvoleného statického hesla, teda hesla ktoré obsahuje netriviálnu kombináciu znakov malej a veľkej abecedy, čísel a špeciálnych znakov o dĺžke aspoň 8 znakov je vysoko neefektívne využiť niektorú z týchto techník. Avšak nakoľko aj vďaka týmto technikám sa v minulosti podarilo prelomiť mnoho hesiel a zároveň môžeme povedať, že tieto techniky sú rodičmi všetkých techník prelomovania hesla, patrí sa, aby boli predstavené.

3.2.1 Útok hrubou silou

Brute force attack alebo útok hrubou silou patrí medzi najjednoduchšie útoky. Zakladá sa na technike overovania všetkých možností hesiel. V minulosti bol tento útok pomerne úspešný, nakoľko si mnoho užívateľov volilo až príliš jednoduché heslo. Avšak dnes už mnohé inštitúcie používajú pri tvorení hesla prísnejšiu politiku skladajúcu sa z minimálneho počtu znakov a povinnosti malých a veľkých písmen v kombinácií s číslami. Týmito podmienkami a teda zamedzením užívateľovi zadať si jednoduché heslo sa znemožňuje útok pomocou tejto techniky. Pre ilustáciu, počet kombinácií 8 znakového hesla, ktoré môže obsahovať znaky malej a veľkej abecedy, čísla a špeciálne znaky (bez možnosti použitia národných znakov, ako sú “á ”, ”ž ”, ”ľ ”, ”í ” a podobne) je $95^8 = 6634204312890625$. Teda pri rýchlosti overovania 10 279 000 kľúčov za sekundu, čo je viac ako veľmi optimistické overenie všetkých možností hesla, by overenie všetkých možností hesla trvalo 645413397.5 sekúnd, čo je v prepočte 20 rokov (počítame, že rok má vždy 365 dní), 170 dní, 1 hodinu, 29 minút a 57,5 sekundy. [13][14]

3.2.2 Slovníkový útok

Slovníkový útok je priamo založený na útoku brute force, teda skúšaniu dostupných možností. Líši sa však v spôsobe skúšania dostupných možností, kde neskúša celú množinu potencionálnych hesiel, ale len určitú časť hesiel, ktoré sú uvedené v slovníku. Teda ako jeden z povinných vstupov tu vystupuje slovník (zoznam), v ktorom sú uvedené najpravdepodobnejšie heslá a ktorého hodnoty sa budú testovať. Zdrojom kľúčov tohto slovníka býva najčastejšie slovník užívateľovho materkého jazyka, kde sa predpokladá, že užívateľ si zvolil ako heslo jedno slovo alebo slovné spojenie, ktoré môže byť doplnené o číslo nakonci alebo kapitálu na začiatku. Príprava slovníka síce môže byť časovo náročnejšia, no hlavne v prípade, že hľadáme heslá väčšej množiny užívateľov nám stačí len raz spočítať tento slovník, teda kľúč a jeho hash, a potom môžme jednoducho a efektívne vyhľadávať jednotlivé hashe v databáze hesiel užívateľov. Ako ochrana voči tomuto typu útoku sa používa solenie. Solenie sa zakladá na pridaní jedinečnej informácií pre každého užívateľa ku heslu. Teda v databáze bude miesto reťazca "hash(password)" uložený reťazec salt+delimiter+hash(salt+password) kde salt je solenie- náhodne vygenervaný reťazec pre každého užívateľa a delimeter je oddeľovací znak, pomocou ktorého vieme, kde začína hash. Vďaka soleniu sa tento typ slovníkového útoku stáva nepoužiteľným. Avšak aj dnes existuje množstvo systémov, ktoré ukladajú heslá v otvorenej forme a teda tento typ útoku môže byť úspešne použitý.[15][16]

4 Dynamické heslá

Na rozdiel od statických hesiel, ktoré sú až do zmeny užívateľom nemenné, sa dynamické heslá menia v závislosti na čase a systéme. Existuje množstvo rôznych systémov, ktorých výsledkom je dynamická autentizácia užívateľa, avšak každý je mierne odlišný od ostatných. Bohužiaľ dynamická autentizácia užívateľa nie je v súčastnosti dostatočne rozšírená. Medzi možné dôvody môžeme zaradiť systémovú, respektíve užívateľskú náročnosť, ako aj nedostatočný tlak na zmenu zaužívaného systému. Nedostatočný tlak nie je možné vyriešiť návrhom systému, avšak ostatné vyššie spomenuté problémy rieši tento navrhovaný systém, ktorý pri miernej optimalizácií je možné použiť naprieč takmer celým spektrom autentizácie užívateľa. Či sa jedná o bežné prihlasovanie k online účtu, cez odomknutie mobilného telefónu až po pinkódy, jeho používanie výrazne zvýši bezpečnosť a minimálne zvýši systémovú a užívateľskú náročnosť.

4.1 Systém autentizácie užívateľa pomocou hesla s dynamickým údajom

Základnou myšlienkou tohto systému je fakt, že naše statické heslo môže byť kľudne sebeviac zložité, stačí aby ho útočník raz zistil a heslo je zbytočné. V dnešnej dobe, keď bežný užívateľ nedbá príliš na ochranu svojho počítača pred vírusmi a zároveň je hrozba infikovania práve nejakým nežiadúcim softwarom čoraz vyššia je pravdepodobnosť infikovania veľmi vysoká. Preto by sa dalo povedať, že v porovnaní ku dnes existujúcej hrozbe straty virtuálnej identity je využívanie statických hesiel hra s otvoreným ohňom. Samozrejme je jasné, že ani tento systém v extrémnych prípadoch nemusí pomôcť a útočník môže získať heslo a tým prístup k citlivým informáciam, avšak na rozdiel od použitia klasického statického hesla nemeniaceho sa v čase, je pre útočníka ďaleko obtiažnejšie toto heslo zistiť. Tento systém dynamických hesiel na základe časových údajov funguje na jednoduchom princípe. Vždy, keď sa niekde autentizujeme nachádzame sa v jedinečnom časovom okamihu, ktorý sa už nebude opakovať. A na základe tejto vlastnosti môžeme mať v každom časovom okamihu jedinečné heslo.

Stačí, aby sme do hesla pridali určitú premennú, ktorej hodnota bude priamo závislá na nejakej časovej premennej. Kombináciou týchto časových premenných so základnými matematickými operáciami a statickými znakmi získavame takmer nekonečnú množinu rôznych vzorov (patternov). Pre podporenie užívateľskej prívetivosti si užívateľ tento vzor volí sám. Tento vzor sa stáva jeho kľúčom, ktorý bude v budúcnosti pri prihlasovaní používať. Využíva sa teda autentizácia užívateľa pomocou princípu "čo užívateľ vie". Vzor je následne uložený a pri autententizácii sa vyhodnocuje a porovnáva so vstupom, ktorý zadal v závislosti od aktuálneho času a dátumu.

4.2 Reprezentácia vzoru hesla

Vzor, ktorý si užívateľ vytvorí je reprezentovaný v implementácií pomocou triedy Password. Táto trieda obsahuje dva atribúty: číselnú hodnotu freeChar a List objektov Elements. Atribút freeChar označuje, na ktorom mieste môže byť použitý ľubovoľný znak. Teda pri hesle „admin“ a hodnote tohto atribútu 0 budú platné aj heslá „Kdmin“, „5dmin“, „+dmin“ a podobne. Elements je jednoduchá trieda, ktorá má štyri privátne atribúty: String type , String value, List ifTrue a List IfFalse. Týmto poskytuje táto trieda dostatočnú robustnosť pre všetky potencionálne typy. Samozrejme, nie všetky argumenty sú vždy využité. Argument List ifFalse je využívaný len pre typ bool. Arguemnt ifTrue využíva okrem typu bool aj pokročilejšie metódy typu sm. Hodnota typu špecifikuje, o aký typ časti sa jedná a na jej základe sa určujú operácie, ktoré sa s hodnotu budú vykonávať. Môže nadobúdať nasledujúce hodnoty:

- pt – (plain text) predsavuje statický text, ktorý sa nebude nejako modifikovať. Využitím iba tohto prvku získame klasické statické heslo.
- int – (integer) predstavuje reťazec, ktorý po zadaní bude matematicky vyhodnotený pomocou knižnice JEval a jej triedy Evaluator a nadefinovaných metód.
- sm- (string method) predstavuje reťazec, ktorý bude vyhodnotení ako jedna z reťazcových funkcií. V tomto prípade je možné

za numerický parameter dosadiť iné dynamické funkcie spolu v kombinácií s matematickými operáciami. Ako reťazcový parameter sa zadáva pole triedy Elements. Teda je možné do tohto parametru zadať dynamickú hodnotu. Príklad : sSkipChar(iGetMinutes(),pt(123)int(iGetDayMonth()))

- bool-(boolean) predstavuje trojicu: String výraz, List pravda, List lož. Výraz je podmienka, ktorá bude vyhodnotená a na základe ktorej bude vrátená buď pravda alebo lož. Výraz je vyhodnotení pomocou knižnice JEval a triedy Evaluator a teda môže obsahovať len matematické výrazy s kombináciou s nadefinovanými metódami. Pravda a lož sú typu List triedy Elements.

V parametri value sa ukladá konkrétna hodnota danej časti vo forme dátového typu String. Argumenty ifTrue a ifFalse obsahujú List typu Elements, vďaka čomu je možné pri type bool zadať tzv. podvzory, teda vzory, ktoré sú platné na základe vstupnej podmienky.

Ukážky využitia vzorov a voľného znaku sú uvedené v časti Ukážky použitia.

4.2.1 Gramatika generujúca heslo

Pre lepšiu ilustráciu uvádzam gramatiku, ktorá popisuje generovanie hesiel.
Gramatika G je štvorica$(N, \sum, P, \perp)$, kde:

- $N = \{S, S1, PT, PT1, INT, INT1, INT2, BOOL, SM\}$
- $\sum$ =Množina všetkých tlačiteľných ASCII znakov
- $\perp = S$
- P : { $S \rightarrow PTS1|INTS1|BOOLS1|SMS1$,
  $S1 \rightarrow PTS1|INTS1|BOOLS1|SMS1|\epsilon$,
  $PT \rightarrow$(ľubovolný znak z $\sum$)$PT1$,
  $PT1 \rightarrow PT|\epsilon$,
  $INT \rightarrow INT1|0|-INT1$,
  $INT1 \rightarrow$(ľubovolná číslica z intervalu<1,9>)$INT2$,
  $INT2 \rightarrow INT1|0|0INT1|\epsilon$,
  $BOOL \rightarrow S$,

$SM \rightarrow S|PT2$,
$PT2 \rightarrow$ľubovolný znak z $\sum$.}

Vyššie uvedená gramatika je len názorná a teda nespĺňa podmienky pre klasické gramatiky.

4.3 Ukladanie vzoru hesla

Pri autentizácii užívateľa pomocou statického hesla sa, ako je vyššie spomenuté, ukladá hash H(heslo+solenie) a solenie. Tento systém ukladania hesla spolu so solením využíva fakt, že je hashovacia funkcia jednosmerná, teda je jednoduché ju spočítať jedným smerom, v tomto prípade je jednoduché zahashovať heslo, avšak je veľmi výpočetne náročné dopočítať z hashu hesla jeho pôvodnú hodnotu. Ako reálny príklad jednosmernej funcie si môžeme predstaviť rozklad súčinu dvoch netriviálnych prvočísel. Je pre nás náročné zistiť, ktoré dve čísla spolu dávajú súčin 2627, avšak je pre nás jednoduché spočítať súčin čísel 71 a 37. [17] Tiež je dnes stále používané ukladanie hesla v otvorenej forme. Tento spôsob však už dnes podporujú len menej dôležité systémy a služby preto nemá zmysel sa ním viac zaoberať.

Systém ukladajúci heslá vo forme hashu a solenia nemôže byť použitý na vzor ako celok, avšak je možné ho použiť na statické časti hesla, teda časti, ktorých typ je „pt“.V prípade jednej statickej časti v hesle je možné použiť tento systém, teda uložiť statickú časť ako H(heslo+solenie) a solenie. Ďalšou možnosťou je rozdelenie jedinej „pt“ časti na dve a následne s nimi pracovať ako s dvomi „pt“ časťami. V prípade, že sa v hesle nachádza viac ako jedna časť označená ako „pt“, je možné tieto časti uložiť rôznymi spôsobmi, ako napríklad H(H(001+pt1)+H(002+pt2)+....), kde H () znázorňuje hashovaciu funkciu. V prípade sčitovania hashov je potrebné, aby boli pred jednotlivé „pt“ časti predreťazované identifikačné reťazce. Vďaka týmto reťazcom sa nemôže stať, že v prípade vzoru [pt,hajdy]...[pt,brita]...[pt,tesa] by bolo akceptované heslo, v ktorom by bolo prehodené poradie „pt“ častí, teda názorne: „brita...hajdy...tesa“. Z dôvodu, že všetky ostatné časti vzoru sú dynamické, nie je možné ich ukladať v zahashovanej podobe, a teda je nutné ich uložiť otvorene čo predstavuje určitú bezpečnostnú slabinu, hlavne čo sa týka ukradnutia databázy hesiel. Preto je veľmi prospešné, až nutné, vložiť do vzoru minimálne jednu sta-

tickú „pt“ časť. Avšak Nakoľko sa jedná len o funkčný prototyp nového spôsobu autentizácie užívateľa, ukladanie vzoru hesla priamo implementačná časť nerieši. Rovnako aj vyššie uvedené informácie je nutné brať len ako ukážku možností ukladania tohto vzoru nie ako záväznú implementáciu.

4.4 Vlastnosti vytvorenej metódy

Každý spôsob autentizácie užívateľov má svoje výhody a nevýhody. Či sa jedná o autentizáciu užívateľa pomocou statického hesla, cez biometriky až po tokeny, každý z nich má väčšie či menšie výhody a rovnako tak aj nevýhody. Podľa dnes rozšírených spôsobov autentizácie užívateľov je zrejmé, že medzi hlavné kritériá pre úspešný a rozšírený spôsob autentizácie užívateľa patria zo strany užívateľov pohodlnosť a zo strany poskytovateľov, teda voči autorite, voči ktorej sa autentizujeme, je to nenáročnosť na hardware. Pod pohodlnosťou si môžeme predstaviť jednoduchosť autentizácie užívateľa, respektíve pamäťovú nenáročnosť, teda aby autentizácia užívateľa dlho netrvala a aby si nemusel pamätať zložité heslo. Tieto problémy dnes riešia napríklad biometrické autentizácie, ktoré sú pre používateľa minimálne náročné a zároveň riešia aj problém pamäti, nakoľko nie je potrebné, aby si užívateľ niečo zapamätal. Podobne fungujú napríklad aj čipové karty, ktoré stačí priložiť a systém autentizuje užívateľa automaticky. Pod náročnosťou na hardware si môžeme predstaviť fakt, že ešte nie je zavedená globálna autentizácia užívateľa pomocou biometrických údajov. Samozrejme, do tejto kategórie spadá aj finančná náročnosť, ktorá implementovaním a spravovaním tohto systému pripadá na poskytovateľa.

Táto metóda nemá žiadne špeciálne požiadavky, ktoré by ju robili či už pre užívateľa alebo poskytovateľa nezaujímavú. Z pohľadu bežného užívateľa môže vzniknúť mierny problém pri prvotnom zapamätaní si vzoru, ktorý zadal, respektíve pri spočítavaní dynamickej časti hesla. Je možné, že najpoužívanejšie budú heslá obsahujúce jednu statickú časť s jednou dynamickou časťou, kde bude zadaná len jednoduchá funkcia typu iGetMonth()+3. Avšak dostatočne opatrný užívateľ, ktorému nevadí dlhšie a zložitejšie heslo, si môže vyskladať natoľko pokročilé dynamické heslo, že aj v prípade, že by útočník získal väčšie množstvo

kombinácií dvojíc dátum a čas a platné heslo pre konkrétny časový úsek, nebol by schopný spätne zrekonštruovať pôvodný vzor.

Pre poskytovateľov ako takých nastáva zmena oproti používaniu statických hesiel pri implementácií autentizácie užívateľa a pri zmene štruktúry databáz s heslami. Nie je potrebný žiadny dodatočný hardware či už vo forme väčšieho množstva databáz alebo ako dodatočný hardware ako napríklad snímač odtlačkov prstov.

Tento spôsob autentizácie užívateľov je z pohľadu poskytovateľov možné používať fakticky okamžite (po doriešení implementačných detailov). Pre užívateľov, ktorí nedbajú dostatočne na možnosť prelomenia hesla sa nič nemení. Stále si budú môcť zvoliť triviálne heslo, na ktoré nepotrebujú byť dobrí v matematike. Avšak pre uvedomelých užívateľov, ktorí si uvedomujú nástrahy krádeže identity, poskytuje tento spôsob možnosť zabezpečiť si svoje údaje a dáta omnoho dôkladnejšie ako pomocou bežného statického hesla.

Samozrejme, tento systém môže byť používaný celým spektrom autentizacií užívateľa. Jednoduchou modifikáciou sme schopní implementovať tento systém na pin kódy pri využívaní platobnej respektíve sim karty. Rovnako je možné upraviť systém tak, aby bolo jeho používanie možné na mobilných telefónoch. Je fakt, že všade, kde sa dnes využíva autentizácia užívateľa hesla pomocou statických údajov je možné použiť aj tento dynamický spôsob autentizácie užívateľov.

4.5 Zoznam dostupných metód

Nižšie uvedené metódy ponúkajú široké spektrum možností tvorby dynamických hesiel. Samozrejme, ani tento systém pri jednoducho zvolenom vzore hesla nie je schopný dostatočne zabezpečiť bezpečie používateľom. Preto je vhodné, aby sa z daných funkcií vytvorila vhodná kombinácia. Tiež je odporúčané nepoužívať časové a dátumové prvky bez ich dodatočnej modifikácie, nakoľko práve tieto typy hesiel budú ako prvé testované pri prípadnom útoku na váš účet. Pre názorné ukážky budeme pracovať s konštantým dátumom a časom : 13.06.2015 18:24, sobota.

4.5.1 Časové a dátumové metódy

Táto sada metód poskytuje základné operácie s dátumom a časom. Metódy pokrývajú širokú škálu dynamických hodnôt s rôznou intenzitou zmeny a spolu s kombináciou so základnými matematickými funkciami ponúkajú neobmedzený priestor pre kreativitu pri vytváraní vlastného vzoru. Dá sa povedať, že dostatočne zodpovedný užívateľ je schopný pomerne jednoducho nastaviť svoj vzor tak, že na jeho uhádnutie bude útočník potrebovať veľké množstvo platných dvojíc heslo, dátum a čas. Na použití týchto základných metód stojí celý navrhovaný systém, preto je vysoko odporúčané, aby boli tieto metódy vo veľkej miere zastúpené vo vzore hesla.

Názov metódy	Opis	Hodnota
iGetMinutes()	vráti aktuálny počet minút	24
iGetMinutesInv()	vráti doplnok k aktuálnemu počtu minút	36
iGetHours12()	vráti aktuálny počet hodín v 12-hodinovom formáte	6
iGetHours12Inv()	vráti doplnok k aktuálnemu počtu hodín v 12-hodinovom formáte	6
iGetHours24()	vráti aktuálny počet hodín v 24-hodinovom formáte	18
iGetHours24Inv()	vráti doplnok k aktuálnemu počtu hodín v 24-hodinovom formáte	6
iGetPhaseOfDay()	Vráti 0 pre čas od 00:00 do 11:59, 1 od 12:00 do 23:59	1
iGetPhaseOfDayInv()	vráti inverznú hodnotu ku funkcií getPhaseOfDay()	0
iGetDayWeekSB()	vráti hodnotu daného dňa v týždni so začiatkom týždňa v nedeľu (SundayBased)	7
iGetDayWeekSBInv()	vráti doplnok ku funkcií iGetDayWeekSB()	0
iGetDayWeekMB()	vráti hodnotu daného dňa v týždni so začiatkom týždňa v pondelok (MondayBased)	6

iGetDayWeekMBInv()	vráti doplnok ku funkcií iGetDayWeekMB()	1
iGetDayMonth()	vráti hodnotu aktuálneho dňa v mesiaci	13
iGetDayMonthInv()	vráti počet dní do konca mesiaca	17
iGetDayYear()	vráti hodnotu daného dňa v aktuálnom roku	164
iGetDayYearInv()	vráti počet dní do konca roka	201
iGetWeekOfYear()	vráti hodnotu daného týždňa v roku	11
iGetMonth()	vráti hodnotu aktuálneho mesiaca v roku	6
iGetMonthInv()	vráti počet zostávajúcich mesiacov do konca roka	6
iGetYear()	vráti hodnotu aktuálneho roku	2015
iGetHours24Mins()	vráti zreťazenie počtu hodín v 24-hodinovom formáte a počtu minút	1824
iGetHours12Mins()	vráti zreťazenie počtu hodín v 12-hodinovom formáte a počtu minút	624
iGetMinsHours24()	vráti zreťazenie počtu minút a počtu hodín v 24-hodinovom formáte	2418
iGetMinsHours12()	vráti zreťazenie počtu minút a počtu hodín v 12-hodinovom formáte	2406

Poznámka: Posledné štyri metódy môžu naznačovať, že napríklad pri čase 07:04 vráti metóda iGetHours24Mins() štvorčíslie “0704”, avšak tento názor nie je správny. Dané metódy vrátia hodnotu bez úvodných núl. Teda pri čase 00:01 vráti rovnaká metóda hodnotu “1”. V čase 00:00 by táto metóda vrátila hodnotu “0”.

Názov metódy	Opis	Hodnota
iGetHours24MinsInv()	vráti doplnok ku zreťazeniu počtu hodín v 24-hodinovom formáte a počtu minút	8176
iGetHours12MinsInv()	vráti doplnok ku zreťazeniu počtu hodín v 12-hodinovom formáte a počtu minút	9376

iGetMinsHours24Inv()	vráti doplnok ku zreťazeniu počtu minút a počtu hodín v 24-hodinovom formáte	7582
iGetMinsHours12Inv()	vráti doplnok ku zreťazeniu počtu minút a počtu hodín v 12-hodinovom formáte	7582

Poznámka: Pre úplnú presnosť a jasnosť týchto metód je potrebné uviesť, že táto doplnková hodnota sa počíta podľa vzorca „value=10000-iGet...()“, a teda v čase 00:00 dané metódy vrátia hodnotu 10000.

4.5.2 Reťazcové metódy

Pre väčšiu možnosť originality vzorov hesiel sú pridané aj ďalšie metódy pracujúce jednak s celými číslami, ale aj so znakmi a reťazcami. Pomocou týchto metód sme schopní spraviť aj z reťazcovej časti hesla dynamický prvok. Samozrejme, nakoľko sa jedná o fakticky dva neporovnateľné typy (reťazec a číslo), je aj množstvo metód výrazne limitované. Tiež môžeme s istotou povedať, že časová náročnosť vyhodnotenia danej metódy je pre užívateľa omnoho vyššia ako pri číselných metódach. Avšak pravidelným používaním sa aj tieto metódy stanú viac automaticky vyhodnocované používateľom a aspoň čiastočne sa zníži ich časová náročnosť. Je vysoko odporúčané použiť tieto metódy vo svojom vzore.

Názov	Opis	Ukážka
sRotate(int *rot*,List *s1*)	zrotuje *s1* o *rot* znakov. *rot*<0 rotuje doľava, *rot*>0 rotuje doprava	sRotate(2,filip)=lipfi
sSkipChar(int *pos*, List *s1*)	vynechá z *s1* znak na *pos*% *s1*.length() pozícií.	sSkipChar(0,auto)=uto
sAddASCIIChar(int *val*)	pridá znak z tlačiteľných ASCII znakov podľa vzorca *val* % 95 + 32	sAddASCIIChar(15)=/

sAddAlphaNum(int *val*)	Pridá znak podľa vzorca *val*%62. Pre hodnotu <0 vráti 0, pre 0-9 vráti číslo, pre 10-35 vráti písmená malej abecedy pre 36-61 vráti písmená veľkej abecedy	sAddAlphaNum(37)=B

4.5.3 Boolean funkcia

Hoci je boolean funkcia len jedna, poskytuje natoľko širokú funkcionalitu, že ďalšia boolean funkcia by sa javila ako zbytočná.

Názov	Opis	Ukážka
bCheckBoolean(String *con*, List *true*, List *false*)	vráti *true* alebo *false* na základe vyhodnotenia podmienky *con*	Ukážka sa nachádza v časti ukážky použitia

4.5.4 Matematické operácie nad náhodnými dátami

V prípade, že by bolo možné vo vzore použiť len vyššie uvedené matematické metódy bez možnosti ďalších matematických operácií, nebolo by pre útočníka náročné odhaliť tento vzor na základe jedného odpozorovania hesla. Tým pádom by fakticky celý tento systém stratil svoj význam. Preto je možné vytovriť si vlastnú kombináciu dynamických premenných s matematickými funkciami a výrazmi. V týchto výrazoch je možné využiť kombináciu viacerých dynamických prvkov so statickými matematickými prvkami (konštantné čísla) za použitia rôznych matematických funkcií. Pre efektívnu ochranu je odporúčané využiť túto vlastnosť techniky a použiť zložitejšie matematické výrazy zložené z nasledujúcich matematických operácií:

Operácia	Znak	Príklad
Sčitavanie	+	iGetMonth()+13
Odčitavanie	-	iGetDayMonth() – iGetMonth()
Násobenie	*	4*iGetYear()

Zvyšok po delení	%	iGetHours12()%2
Zátvorkovanie	( )	(iGetPhaseDay()+iGetMinutes())*2

4.5.5 Ukážky použitia

Nižšie uvedené príklady ukazujú možné použitia jednotlivých metód vo vzoroch. Samozrejme, v týchto ukážkach nie sú použité všetky dostupné metódy a funkcie. Ukážky slúžia ku znázorneniu komplexnosti vytvoreného systému a možnosti využitia jednotlivých metód. Všetky nižšie uvedené príklady sú vyhodnotené v konštantom čase a dátume: 13.06.2015 18:24, sobota. Vďaka použitiu voľného znaku nám vzniká širšia množina správnych hesiel pre jednotlivé vzory, avšak pre jednoduchosť bude uvedená len jedna alternatíva. Pre prehľadnosť je v tabuľke voľný znak vyznačený tučným znakom „**F**“. Pomocou kurzívy sú označené typy jednotlivých častí.

Voľný znak	Vzor	Výsledok
-1	*pt*(welcome) *sm*(sSkipChar(iGetDayMonth(), *pt*(home)))	welcomehme
0	*int*(3*iGetHours24Mins()) *sm*(sAddAlphaNum(iGetMonth()))	**F**4726
20	*sm*(sRotate(-iGetHours12(), *pt*(iGetYear()) *int*(iGetYear())))	ar()20**F**5iGetYe
-5	*pt*(ziadny volny znak)	ziadny volny znak
iGetMinutes() +3	*int*(((3+iGetMonth())* (55/iGetDayMonthInv()+ iGetMinutesInv()) % (iGetYear()-2000)+1) *pt*(math rules)	15m**F**th rules
3	*bool*(iGetMinutes()%2==0, *pt*(pravda),*pt*(loz))	pra**F**da

iGetMinutes Inv()-30	*bool*(iGetDayMonthInv()<10, *bool*(iGetHours12MinsInv()>3000, *bool*(iGetPhaseOfDay()=0,*pt*(pravda v pravde v pravde),*pt*(loz v pravde v pravde)), *pt*(loz v pravde)), *bool*(iGetWeek()==15,*pt*(pravda v lozi) *int*(13*iGetMinutes()),*pt*(loz v lozi) *sm*(sAddASCIIChar(iGetYear()))	loz v lozi4
iGetPhaseOf DayInv()	*pt*(vsetko pod) *int*(iGetMinutesInv()-6* iGetMonthInv()+1) *sm*(sRotate(iGetDayWeekSB()+1, *pt*(strechou))) *bool*(iGetYear()==2015* iGetMinutes()/iGetMinutes(), *pt*(!!!),*pt*(nedostupne))	**F**setko pod1strechou!!!
3	*int*(iGetMinutes()*3)*pt*(nope) *int*(iGetMonth()%2) *sm*(sRotate(iGetDayMonth(), *pt*(rotacia)))	72n**F**pe0otaciar

5 Porovnanie systému s alternatívnymi systémami pre zadávanie tajnej informácie

Samozrejme, problém ľahkej možnosti zistenia tajnej informácie sa snaží riešiť mnoho rôznych metód, techník a systémov. Tieto systémy sa zameriavajú na sťaženie možnosti odpozorovania či zistenia tajnej informácie, avšak každý využíva viac alebo menej odlišný systém bezpečnosti, a teda je vhodný na iné použitie. V tejto kapitole opisujem a porovnávam niekoľko z týchto systémov so systémom autentizácie užívateľa pomocou hesla s časovým údajom.

5.1 Virtual password using random linear functions for on-line services, ATM machines, and pervasive computing

Tento systém sa snaží riešiť problém statických hesiel, ktoré sú ľahko prelomiteľné pomocou bežných útokov ako sholdersurfing, keylogging respektíve trójsky kôň alebo phishing. Tento problém rieši pomocou rozloženia hesla na statickú a dynamickú časť. Tento systém používa pevnú dynamickú lineárnu funkciu na generovanie dynamickej časti hesla. Táto funkcia je navrhnutá, aby odolala útokom shoulderingu, troyana a phishingu[18].
Zhody:

- Myšlienka dynamicky sa meniaceho hesla, ktoré sa pri každom prihlásení mení.
- Heslo zložené zo statickej a dynamickej časti.
- Ochrana pred útokmi shouldersurfingu, keyloggingu, trójskym koňom, phishingom a podobne.

Odlišnosti:

- Nie je možné zvoliť si vlastnú dynamickú funkciu.
- Možnosť len jednej statickej a jednej dynamickej časti.
- Nemožnosť širokého uplatnenia. Zameranie len na ATM.

5.2 How to fend off shoulder surfing

Systém zameriavajúci sa na ochranu proti odpozorovaniu pinkódu. Systém funguje nasledovne: pre každú číslicu zo štvormiestneho pinkódu sa vygeneruje jedinečná mriežka čísel, kde každé číslo je zafarbené buď na bielo alebo na čierno. Každá z týchto mriežok sa zobrazí na 0,5 sekundy a následne je užívateľ vyzvaný, aby zadal správnu kombináciu farieb polí, ktoré zodpovedajú správnemu pinkódu.[19]
Zhody:

- Pri jednotlivých pokusoch sa zadávaná kombinácia mení, čo navodzuje pocit dynamickosti hesla.
- Poskytuje ochranu proti shoulder surfingu.

Odlišnosti:

- Zamerané len na ochranu pinkódov.
- Skutočný pinkód ostáva statický, mení sa len jeho zadávanie na dynamické.

5.3 Spy-resistant keyboard: more secure password netry on public touch screen displays

Jedná sa o software pre klávesnice na verejných miestach. Je vytvorená nová klávesnica, kde každá klávesa obsahuje tri znaky a indikátor. Znak, ktorý je najvyššie je písmeno malej abecedy, v strede sa nachádza písmeno veľkej abecedy a na spodku je buď číslo alebo špeciálny znak. Indikátor určuje, ktorý znak je aktuálne zvolený.

Rozloženie znakov na klávesoch je náhodné a nakoľko je čísel spolu so špeciálnymi znakmi dokopy 42 a písmen iba 26, niektoré písmená sa opakujú. Po nájdení správnej klávesy, kde sa hľadaný znak nachádza, je potrebné podať indikátor na správny znak. Teda je potrebné si zapamätať polohu klávesy a zároveň aj polohu indikátoru. Polohu znaku nie je potrebné memorovať, nakoľko podľa vyššie zmienených pravidiel je vždy presne určiteľné, na ktorej pozícii sa znak nachádza. [20]

Zhody:

- Pri jednotlivých pokusoch sa zadávaná kombinácia mení, čo navodzuje pocit dynamickosti hesla.
- Poskytuje ochranu proti shoulder surfingu.

Odlišnosti:

- Zamerané len na prihlasovanie na verejných miestach.
- Skutočné heslo ostáva statické, mení sa len jeho zadávanie na dynamické.

5.4 Reducing Shoulder-surfing by Using Gaze-based Password Entry

Tento systém sa snaží vytvoriť účinnú ochranu proti shoulder surfingu. Navrhuje zadávanie hesla pomocou pohľadu. Na obrazovke sa zobrazí klávesnica (môže sa jednať o klasickú, ale aj o modifikovanú klávesnicu), ktorá má v strede každej klávesy červenú bodku, ktorá slúži na ľahšie zameranie sa na danú klávesu. Pre zistenie, kde sa užívateľ pozerá, metóda využíva Gaze tracking, ktorý využíva techniky na sledovanie užívateľovej očnej šošovky na vypočítanie pozície, kde sa užívateľ pozerá. Tieto techniky boli pôvodne vyvinuté pre postihnutých užívateľov, kde mali nahradiť klasický vstup klávesnice a myši.[21]
Zhody:

- Poskytuje ochranu proti shoulder surfingu a keyloggingu.

Odlišnosti:

- Využíva statické heslá.
- Neposkytuje ochranu proti útokom phishingu a podobne.
- Systém je nepoužiteľný bez dotatočného hardwaru.

5.5 PassShape

Systém, ktorého cieľom je uľahčiť ľuďom zapamätanie si aj zložitejšieho pinkódu. Vychádza z vlastného výskumu, v ktorom uviedlo 55 % účastníkov, že aspoň raz už zabudli pinkód a zároveň z neho vyplynulo, že 40% zúčastnených si svoj pinkód pamätá podľa geometrického tvaru, ktorý vytvoria pospájaním jednotlivých čísel. Autori navrhujú, aby sa namiesto číselného pinkódu využíval pinkód vytvorený zo smerov pohybu, teda hore, dole, doprava, doľa, šikmo hore doprava, šikmo hore doľava, šikmo dole doprava, šikmo dole doľava. Autori predpokladajú, že táto metóda umožní ľuďom zapamätať si ťažšie heslo, a teda sa vyhnú bezpečnostným rizikám ako sú napríklad triviálne kombinácie pinkódu typu 1234 alebo osobným pinkódom, teda napríklad dátum narodenia.[22] Zhody:

- Vďaka možnosti zapamätania si zložitejšieho pinkódu prispieva táto technika ku zvýšenej bezpečnosti.

Odlišnosti:

- Využíva statické heslá.
- Neposkytuje ochranu proti žiadnemu typu útoku.

5.6 One- time password

Ako z názvu vyplýva, jedná sa o systém, v ktorom je heslo platné len pre jedno prihlásenie. Využíva statické heslá, teda nemeniace sa v čase, avšak vďaka faktu, že dané heslo sa použije vždy len raz, vytvára pocit dynamickosti a zároveň chráni systém, v ktorom je one-time password systém implementovaný pred širokým spektrom útokov. Mnoho implementácií One-time password využíva dvojfaktorovú autentizáciu užívateľa, teda niečo čo užívateľ má, ako napríklad token vo forme one-time password kalkulačky alebo mobilný telefón a niečo, čo užívateľ vie, ako napríklad pinkód. Najväčšia výhoda one-time password v porovnaní so statickými heslami je, že sú odolné voči útoku opakovaním. Teda ak potencionálny útočník zachytí one-time password, ktoré už bolo použité na prihlásenie do systému, heslo už nebude ďalej platné,

a teda útočník sa nedostane k citlivým údajom. Ďalšou veľkou výhodou je, že užívateľ, ktorý používa rovnaké alebo podobné heslá na rozličných systémoch nezníži zraniteľnosť týchto systémov, ak útočník získa heslo na jeden z týchto systémov. Množstvo one-time password systémov sa snaží zabezpečiť, aby relácia nebola ľahko zachytiteľná alebo sa nedala vydávať bez znalosti nepredvídateľných dát vytvorených v priebehu predchádzajúcej relácie, čo znižuje riziko útoku. O one-time password sa diskutovalo ako o možnom nástupcovi klasických statických hesiel. Avšak na druhej strane, one-time passwords sú ťažko zapamätateľné pre užívateľov. Preto je potrebné používať dodatočnú technológiu, ako napríklad vyššie spomenuté one-time password kalkulačky(generátory one-time password). Tieto generátory typicky využívajú náhodný alebo pseudonáhodný algoritmus, vďaka ktorému je ťažké uhádnuť na základe jedného one-time password nasledujúci one-time password. Tieto algoritmy sú najčastejšie založené na :

- časovej synchronizácii medzi autentizujúcim serverom a osobným tokenom. Tento token si môžeme predstaviť ako malú kalkulačku alebo kľúčenku s LCD displejom, ktorý zobrazuje pravidelne meniace sa číslo, či dokonca mobilný telefón, na ktorý príde sms s daným one-time password. Tieto one-time password majú avšak istú časovú platnosť.
- použití matematického algoritmu, ktorý vygeneruje nové heslo na základe predchádzajúceho hesla.
- použití matematického algoritmu, kde je nové heslo vygenerované na základe istej výzvy (tzv. challenge-response)[23].

Zhody:

- Heslá sú pri každom prihlásení odlišné.
- Poskytuje ochranu proti útokom shoulder surfingu, phissingu, keyloggeru a podobne.

Odlišnosti:

- Je potrebné ďalšie zariadenie na získanie hesla a teda je menej užívateľsky prívetivé.

5.7 A Hybrid Password Authentication Scheme Based on Shape and Text

Tento systém využíva mriežku, v ktorej každé políčko má svoju hodnotu. Na začiatku sa zvolí vzor, teda kombinácia týchto políčok, kde záleží na poradí jednotlivých polí. Táto kombinácia sa stáva užívateľovým prístupovým heslom. Pri overovaní užívateľa sa do políčok vygenerujú reťazce a užívateľ zadáva ich kombináciu ako svoje heslo. Pri každom zadávaní sa reťazce generujú nanovo, teda užívateľ pri prihlasovaní v rôznych časoch zadáva rôzny reťazec. Tieto reťazce v políčkach sa generujú tak, aby bolo pre potencionálneho útočníka čo najťažšie odpozorovať užívateľov vzor, a teda jednotlivé políčka môžu obsahovať reťazce ako napríklad: „k“, „f“, „kf“, „fk“, „kk“, „ff“. Tiež je potrebné, aby výslednú kombináciu bolo možné vytvoriť pomocou viacerých vzorov, aby útočník nebol schopný ľahko zistiť, o aký vzor sa jedná.[24]
Zhody:

- Heslá sú pri každom prihlásení odlišné.
- Poskytuje ochranu proti útokom shoulder surfingu, phissingu, keyloggeru a podobne.
- Využíva sa vzor, ktorý sa uplatňuje pri každom prihlásení.

Odlišnosti:

- Nevyužívajú sa časovo-dátumové premenné.

5.8 DPASS – Dynamic Password Authentication and Security System using Grid Analysis

Táto technika je čiastočne podobná s predchádzajúcim systémom. Tiež využíva mriežku, ktorej znaky sa náhodne generujú a užívateľ si pamätá len vzor polí, ktoré je potrebné zvoliť. Tieto mriežky sú v závislosti od kapacity servera buď stále náhodne generované, za predpodkladu že server má dostatočnú výpočetnú kapacitu, alebo sa vygeneruje určitý počet mriežok a uloží ich v databáze za predpokladu, že server nemá dostatočnú výpočtovú kapacitu na generovanie

mriežky na počkanie. Pri snahe prihlásiť sa je náhodne vybratá jedna mriežka, odošle sa užívateľovi a označí sa ako použitá. Proces skontroluje povedzme každú hodinu, či je podiel nepoužitých tabuliek väčší ako 20%. V prípade, že tento pomer klesne pod 20%, vygeneruje sa nová sada mriežok. V tejto metóde je možné použiť aj tzv. komplexný znak. Tento znak určuje pozíciu, na ktorej môže užívateľ zadať ľubovoľný znak. Táto myšlienka je implementovaná aj v nami navrhovanej technike. [25]
Zhody:

- Heslá sú pri každom prihlásení odlišné.
- Poskytuje ochranu proti útokom shoulder surfingu, phissingu, keyloggeru a podobne.
- Využíva sa vzor, ktorý sa uplatňuje pri každom prihlásení.

Odlišnosti:

- Nevyužívajú sa časovo-dátumové premenné.

5.9 Evaluation of Eye-Gaze Interaction Methods for Security Enhanced PIN-Entry

Technika, rovnako ako jedna z vyššie uvedených, využíva pohyb očnej zreničky, avšak v tomto prípade sa zameriava na písanie číslic pohľadom, teda jedná sa o techniku využivajúcu sa hlavne pri pinkódoch. Užívateľ stlačí tlačidlo signalizujúce, že je pripravený pohľadom znázorniť danú číslicu a následne urobí pohyb očami. Nakoľko táto technika nepotrebuje lokalizovať presný bod tak ako systém vyššie, ale stačí jej rozpoznať pohyb očami, jej nároky na hardwarové zariadenie sú podstatne nižšie v porovnaní s metódou určovania jednotlivých kláves na displeji. Podľa autorov je dostačujúca obyčajná web kamera a infračervená kamera, ktorá zaistí väčší kontrast v snímani, nakoľko vytvorí efekt červených očí.[26]
Zhody:

- Poskytuje ochranu proti útokom shoulder surfingu.

Odlišnosti:

- Zameriava sa na pinkódy.
- Neposkytuje ochranu proti útokom keyloggingu, phishingu a podobne.

5.10 TimePasscode Pro

TimePasscode Pro je rozšírenie pre jailbreaknuté iPhony, iPody, iPady a AppleTV (dalej iZariadenia), ktoré môžete nájsť v Cydii. Jailbreaking je proces, pri ktorom sa odstraňujú obmedzenia softwareu urobené IOSom. IOS je oficiálny operačný systém používaný na iZariadeniach. Jailbreaknuté zariadenia umožňujú inštaláciu dodatočných aplikácií, rozšírení a tém, ktoré nie sú dostupné pre iZariadenia bez jailbreaku. Cydia je aplikácia ktorá umožňuje vyhľadať a nainštalovať rôzne softwarové balíčky do jailbreaknutého iZariadenia.

TimePasscode Pro poskytuje základné dynamické metódy pre zadávanie štvormiestneho hesla na odomknutie iZariadenia. V dostupnej funkcionalite (Pro verzia) je zahrnuté (budeme pracovať s časom 13:26) :

- reverzné zadávanie hesla – heslo sa zadá v opačnom poradí. Heslo: 6231,
- voľba hodinového formátu- možnosť zvoliť si medzi 12-hodinovým a 24-hodinovým formátom. Heslo: 1326 alebo 0126,
- dodatočná hodnota na pripočítaní - hodnota v minútach, ktorá bude pripočítaná k času. Hodnota:13, heslo 1339,
- prepínač na zmenu dodatočnej hodnoty na odpočítanie - možnosť odpočítať dodatočnú hodnotu. Hodnota 13, heslo 1313.

Tieto funkcie je možné ľubovoľne kombinovať. [27]

Podobný nástroj je dostupný aj pre operačný systém Android na Google Play[28], kde je širšia škála týchto nástrojov. Zhody:

- Heslá sú pri každom prihlásení odlišné.
- Poskytuje ochranu proti útokom shoulder surfingu.

- Využíva sa vzor, ktorý sa uplatňuje pri každom prihlásení.

Odlišnosti:

- Dostupné len na mobilnej platforme.
- Nemožnosť pracovať s inými časovými premennými, ako je napríklad deň v mesiaci.

6 Použité technológie

Praktická časť tejto práce bola vytvorená v programovacom jazyku JAVA verzia 1.8. Pri práci na praktickej časti tejto práce som pracoval s technológiami JAAS, JEval. Používal som vývojové prostredie NetBeans IDE 8.1.

6.1 JAAS

Java Authentication and Authorization Service, skrátene JAAS je javovská implementácia štandartného PAM modulu. JASS bol predstavený ako rozširujúca knižnica pre Java Standard Edition 1.3 a bola integrovaná vo verzií 1.4. Hlavným cieľom JAAS je oddeliť koncern užívateľskej autentizácie, takže táto autentizácia môže byť spravovaná nezávisle. Zatiaľ, čo minulý autentizačný mechanizmus obsahoval informáciu o tom, odkiaľ kód pochádza a kto ho podpísal, JAAS pridal označovač o tom, kto kód spustil. Rozšírením verifikačných vektorov JAAS rozšíril bezpečnostnú architektúru pre Java aplikáciu, ktorá vyžaduje autentizačný a autorizačný modul.[29]

6.2 PAM

Hlavnou myšlienkou PAMu bolo zaistiť, aby programy boli nezávislé na konkrétnych autentizačných postupoch. Program zaujíma, či má danú službu poskytnúť konkrétnemu užívateľovi, avšak nezaujíma ho, akým spôsobom sa užívateľ autentizuje. Základom systému PAM sú pripájateľné autentizačné moduly. Tieto moduly si môžeme predstaviť ako zdieľané objektové súbory, ku ktorým pristupuje aplikácia v dvoch vrstvách. Prvou vrstvou je systémová knižnica pripojená ku programu, čím dostáva program k dispozícií autentizačné služby modulu. Druhou vrstvou je systémová konfigurácia. Tu administrátor určuje, čo všetko musí užívateľ splniť, aby mu bola služba poskytnutá.[30]

6.3 JEval

JEval je pokročilá knižnica pre pridanie vysokovýkonných matematických, booleanovských a funkcionálnych výrazov parsovaním a vyhodnotením. [31]

Evaluator je trieda rozširujúca java.lang.Object, ktorá je vstupnou bránou do JEval API. Táto trieda podporuje nasledujúce výrazy:

- Matematické výrazy obsahujúce čísla. S číslami pracuje ako s typom double, takže návratová hodnota bude obsahovať aspoň jedno desatinné miesto. Avšak v našom prípade pri využívaní tejto triedy parsujeme desatinnú hodnotu na celočíselnú.
- Stringové výrazy môžu byť spájané, porovnávané a podobne.
- Booleanovské výrazy, ktoré sa vyhodnotia ako pravda (vráti 1.0) alebo lož (vráti 0.0).
- Funkcionálne: Užívateľ si môže vytvoriť mnoho matematických a stringových funkcií, ktoré JEval zásobuje touto triedou.

Operátori, ktoré podporuje trieda Evaluator sú [32] :

(	Otvorené zátvorky	)	Zatvorené zátvorky
+	Sčitovanie čísel a stringov/unárne plus	-	Odčitovanie/ unárne mínus
*	násobenie	/	Celočíselné delenie
%	Zvyšok po delení	=	Rovná sa pre čísla a stringy
<=	Menšie alebo sa rovná pre čísla a stringy	>	Väčšie ako pre čísla a stringy
!=	Nerovná sa pre čísla a stringy	<	Menšie ako pre čísla a stringy
>=	Väčšie alebo sa rovná pre čísla a stringy	&&	Booleanovský prienik
||	Booleanovské zjednotenie	!	Booleanovská negácia

6.4 NetBeans IDE 8.1

NetBeans IDE je oficiálne open-source vývojové prostredie pre programovací jazyk Java 8. Je vydávaný pod licenciou CDDL. Toto vývojové

prostredie je vytvorené v programovacom jazyku Java, a teda je prenášateľné medzi jednotlivými operačnými systémami. Je primárne vyvíjaný pre vývoj v jazyku Java, avšak tiež podporuje aj niektoré ďalšie jazyky ako sú PHP, C, C++ a HTML5. Medzi rokmi 2000 až 2010 bol tento projekt financovaný firmou Sun Microsystems. Od akvizície v roku 2010, kedy Oracle akvizovala Sun a spolu s ním aj NetBeans zastrešuje vývoj tohto vývojového prostredia práve firma Oracle.[33]

7 Záver

Cieľom práce bolo analyzovanie dostupných riešení zaoberajúcich sa dynamickými heslami a návrh vlastného riešenia dynamickej autentizácie užívateľa. Navrhnutý systém mal byť implementovaný v programovacom jazyku JAVA pomocou technológie JAAS. Ďalším cieľom bola analýza navrhnutého systému. Všetky ciele, ktoré boli vytýčené, sa podarilo úspešne splniť.

V rámci práce boli priblížené potencionálne útoky na prelomenie hesla. Navrhnutý systém spĺňa zaradenie do zoznamu dynamických autentizácií užívateľov nakoľko využíva časové premenné. Jednotlivé metódy, ktoré je možné využiť pri tvorbe vzoru boli navrhnuté tak, aby pokryli čo najširšie spektrum využiteľnosti časových premenných a zároveň, aby poskytli užívateľovi dostatočné pohodlie. Preto neboli implementované metódy, ako napríklad iGetSeconds(), ktorá by vrátila aktuálny počet sekúnd. Pri použití tejto metódy by bolo veľmi obtiažne zladiť užívateľove zadávanie hesla so serverovým časom. Na vytvorenom prototype, ktorý ponúka základnú implementáciu tohto systému, je možné ľahko a efektívne odtestovať implementovaný systém. Bolo vytvorených 33 funkcií, ktoré produkujú dynamické časti hesla. Z nich 28 priamo vracia časovú premennú s intervalom zmeny od jednej minúty po jeden rok. 5 funkcií využíva dynamické funkcie na dynamickú zmenu parametra, ktorý je do danej funkcie posielaný. Vytvorený systém je detailne opísaný a použiteľné metódy sú prehľadne uvedené.

Na základe porovnania vytvoreného systému s odlišnými technológiami, ktoré sa zameriavajú na dynamickú autentizáciu užívateľov, môžeme usúdiť, že tento systém využíva odlišnú časť dynamickej autentizácie užívateľov, v ktorej sa zatiaľ nepublikovalo veľké množstvo prác.

Autentizácia užívateľov bola, je a vždy bude veľmi dôležitý článok pri používaní výpočtovej techniky, kde je nutné overenie identity užívateľa. Hoci technológia stále napreduje a dnes už poznáme niektoré typy autentizácií užívateľa, ktoré sú dostatočne pokročilé, stále sú vo veľkom používané techniky, ktoré podľa môjho názoru neposkytujú dostatočnú ochranu pred potencionálnymi útokmi. A práve tento systém by to mohol zmeniť. Vytvorený systém dynamických hesiel

má ešte samozrejme svoje nedostatky, medzi ktoré môžeme určite zaradiť napríklad spôsob ukladania vzoru do databázy, avšak ani Rím nebol postavený za jeden deň. Verím, že s prispením odbornej verejnosti môže táto forma autentizácie užívateľa nadobudnúť obrovskú bezpečnostnú silu a stať sa rovnako používanou ako sú v súčasnosti statické heslá. Prvým krokom bude pravdepodobne optimalizácia metódy a minimalizovanie nedostatkov, alebo aspoň úprava, aby nedostatky tejto metódy boli nižšie ako súčasne používaných metód. Ďalším krokom potom bude zviditeľnenie a tlak na zmenu systému, aby sa vytvorila vôľa zo strany sveta o implementáciu tejto metódy do praxe, a tým zvýšenie bezpečnosti užívateľov. Pretože spoločne môžeme zmeniť svet.

Literatúra

1. *Authentication*. 2015. Dostupné tiež z: http://searchsecurity.techtarget.com/definition/authentication.
2. SMITH, Richard E. *Authentication: from passwords to public keys*. Boston: Addison-Wesley, c2002. ISBN 02-016-1599-1.
3. *Biometrics*. 2015. Dostupné tiež z: http://searchsecurity.techtarget.com/definition/biometrics.
4. *Introduction to Biometrics Systems*. 2005. Dostupné tiež z: http://www.csee.wvu.edu/~natalias/biom426/biom426_fall05.html.
5. TILBORG, Henk C. A. van; JAJODIA, Sushil (ed.). Encyclopedia of Cryptography and Security. In: Boston, MA: Springer US, 2011, kap. Challenge-Response Protocol, s. 198–199. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_1269.
6. TILBORG, Henk C. A. van; JAJODIA, Sushil (ed.). Encyclopedia of Cryptography and Security. In: Boston, MA: Springer US, 2011, kap. Challenge-Response Protocol, s. 638–638. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_1269.
7. *Shoulder surfing*. 2005. Dostupné tiež z: http://searchsecurity.techtarget.com/definition/shoulder-surfing.
8. HE, Jingrui; ZHU, Yada. Encyclopedia of Social Network Analysis and Mining. In: ed. ALHAJJ, Reda; ROKNE, Jon. New York, NY: Springer New York, 2014, kap. Social Engineering/Phishing, s. 1777–1783. ISBN 978-1-4614-6170-8. Dostupné z DOI: 10.1007/978-1-4614-6170-8_290.
9. RAMZAN, Zulfikar. Handbook of Information and Communication Security. In: ed. STAVROULAKIS, Peter; STAMP, Mark. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, kap. Phishing Attacks and Countermeasures, s. 433–448. ISBN 978-3-642-04117-4. Dostupné z DOI: 10.1007/978-3-642-04117-4_23.
10. *Meeting the Ëvil Twin"*. 2005. Dostupné tiež z: http://www.wi-fiplanet.com/columns/article.php/3482021.
11. *Pharming*. 2007. Dostupné tiež z: http://searchsecurity.techtarget.com/definition/pharming.
12. ESTES, Aaron. Encyclopedia of Cryptography and Security. In: ed. TILBORG, Henk C. A. van; JAJODIA, Sushil. Boston, MA: Sprin-

ger US, 2011, kap. Keylogging, s. 691–691. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_782.

13. *Brute force cracking*. 2006. Dostupné tiež z: http://searchsecurity.techtarget.com/definition/brute-force-cracking.
14. *Brute Force Calculator*. Dostupné tiež z: http://calc.opensecurityresearch.com/.
15. ADAMS, Carlisle. Encyclopedia of Cryptography and Security. In: ed. TILBORG, Henk C. A. van; JAJODIA, Sushil. Boston, MA: Springer US, 2011, kap. Dictionary Attack, s. 332–332. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_74.
16. ADAMS, Carlisle. Encyclopedia of Cryptography and Security. In: ed. TILBORG, Henk C. A. van; JAJODIA, Sushil. Boston, MA: Springer US, 2011, kap. Salt, s. 1075–1075. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_94.
17. PAAR, Christof; PELZL, Jan. Understanding Cryptography: A Textbook for Students and Practitioners. In: Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, kap. Hash Functions, s. 293–317. ISBN 978-3-642-04101-3. Dostupné z DOI: 10.1007/978-3-642-04101-3_11.
18. LEI, Ming; XIAO, Yang; VRBSKY, Susan V.; LI, Chung-Chih. Virtual password using random linear functions for on-line services, ATM machines, and pervasive computing. *Computer Communications*. 2008, roč. 31, č. 18, s. 4367–4375. ISSN 0140-3664. Dostupné tiež z: http://www.sciencedirect.com/science/article/pii/S0140366408002752.
19. ROTH, Volker; RICHTER, Kai. How to fend off shoulder surfing. *Journal of Banking & Finance*. 2006, roč. 30, č. 6, s. 1727–1751. ISSN 0378-4266. Dostupné tiež z: http://www.sciencedirect.com/science/article/pii/S0378426605001822.
20. TAN, Desney S.; KEYANI, Pedram; CZERWINSKI, Mary. Spy-resistant keyboard: more secure password entry on public touch screen displays. In: *Proceedings of the 17th Australia conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*. Narrabundah, Australia, Australia: Computer-Human Interaction Special Interest Group (CHISIG) of Australia, 2005, s. 1–10. OZCHI 05. ISBN 1-59593-222-4. Dostupné tiež z: http://dl.acm.org/citation.cfm?id=1108368.1108393.

21. KUMAR, Manu; GARFINKEL, Tal; BONEH, Dan; WINOGRAD, Terry. Reducing shoulder-surfing by using gaze-based password entry. In: *Proceedings of the 3rd symposium on Usable privacy and security*. New York, NY, USA: ACM, 2007, s. 13–19. SOUPS 07. ISBN 978-1-59593-801-5. Dostupné tiež z: http://doi.acm.org/10.1145/1280680.1280683.
22. DE LUCA, Alexander; WEISS, Roman; HUSSMANN, Heinrich. PassShape: stroke based shape passwords. In: *Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining User Interfaces*. New York, NY, USA: ACM, 2007, s. 239–240. OZCHI 07. ISBN 978-1-59593-872-5. Dostupné tiež z: http://doi.acm.org/10.1145/1324892.1324943.
23. RAYES, Mohamed Omar. Encyclopedia of Cryptography and Security. In: ed. TILBORG, Henk C. A. van; JAJODIA, Sushil. Boston, MA: Springer US, 2011, kap. One-Time Password, s. 885–887. ISBN 978-1-4419-5906-5. Dostupné z DOI: 10.1007/978-1-4419-5906-5_785.
24. ZHENG, Ziran; LIU, Xiyu; YIN, Lizi; LIU, Zhaocheng. A Hybrid Password Authentication Scheme Based on Shape and Text. *Journal of Computers*. 2010, roč. 5, č. 5, s. 765–772. ISSN 1796-203X. Dostupné tiež z: http://ojs.academypublisher.com/index.php/jcp/article/view/0505765772/1644. Text-based password authentication scheme tends to be more vulnerable to attacks such as shoulder surfing or a hidden camera. To overcome the vulnerabilities of traditional methods, visual or graphical password schemes have been developed as possible alternative solutions to text-based password schemes. Since it also has some drawbacks to simply adopt graphical password authentication, schemes using graphic and text have been developed. In this paper, a hybrid password authentication scheme based on shapes and texts is proposed. Shapes of strokes are used in the grid as the original passwords and users can log in with textual passwords via traditional input device. The method provides strong resistance to shoulder surfing or a hidden cameraa??Moreover, the scheme has high scalability and flexibility to enhance the authentication process security. The analysis of the security level of this approach is also discussed.
25. S. ARUMUGA PERUMAL., edited by. *ICECT 2011 2011 3rd International Conference on Electronics Computer Technology: 8-10, April 2011,*

Kanyakumari, India. Piscataway, NJ: IEEE Press, 2011. ISBN 978-142-4486-793.

26. DE LUCA, Alexander; WEISS, Roman; DREWES, Heiko. Evaluation of eye-gaze interaction methods for security enhanced PIN-entry. In: *Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining User Interfaces*. New York, NY, USA: ACM, 2007, s. 199–202. OZCHI 07. ISBN 978-1-59593-872-5. Dostupné tiež z: http://doi.acm.org/10.1145/1324892.1324932.
27. *How to set your iOS passcode to your device's current time*. 2014. Dostupné tiež z: http://www.idownloadblog.com/2014/02/03/timepasscode/.
28. *Google Play*. 2016. Dostupné tiež z: https://play.google.com/.
29. *Oracle Java SE Documentation*. 2016. Dostupné tiež z: http://docs.oracle.com/javase/7/docs/technotes/guides/security/jaas/JAASRefGuide.html.
30. *Linux-PAM*. Dostupné tiež z: https://www.kernel.org/pub/linux/libs/pam/whatispam.html.
31. *JEval*. 2008. Dostupné tiež z: http://jeval.sourceforge.net/.
32. *JEval Evaluator*. 2007. Dostupné tiež z: http://jeval.sourceforge.net/docs/api/net/sourceforge/jeval/Evaluator.html.
33. *NetBeans IDE*. 2016. Dostupné tiež z: https://netbeans.org/features/index.html.