Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Katedra informačních technologií

Student: **Filip Škarda**

Vedoucí bakalářské práce: **Ing. Luboš Pavlíček**

Oponent bakalářské práce: **RNDr. Radomír Palovský, CSc.**

TÉMA BAKALÁŘSKÉ PRÁCE:

JEDNORÁZOVÁ HESLA

ROK: 2011

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a že jsem uvedl všechny použité prameny a literaturu, ze kterých jsem čerpal.

V Praze dne 30. dubna 2011 .....

podpis

Poděkování

Na tomto místě bych chtěl poděkovat vedoucímu mé bakalářské práce Ing. Luboši Pavlíčkovi za cenné rady a připomínky.

Abstrakt

Bakalářská práce pojednává o jednorázových heslech a různých možnostech jejich implementace. Jednorázová hesla jsou hesla, které jsou platná pouze pro jedno přihlášení nebo transakci. Na rozdíl od klasických hesel jsou odolná proti opakovaným útokům. To znamená, že když potenciální vetřelec odposlechne heslo, které bylo použito k přihlášení nebo transakci, není schopen je zneužít, protože již není platné. V teoretické části práce jsou uvedeny důvody k použití jednorázových hesel a jsou vysvětleny různé způsoby jejich tvorby a distribuce k uživateli, zároveň jsou popsány některé konkrétní systémy. V praktické části je ukázáno využití open-source systémů jednorázových hesel na dvou různých příkladech. V prvním z nich je popsáno použití systému OTPasswd pro autentizaci k serveru přes SSH, druhý příklad využívá PHP třídu MultiOTP pro autentizaci k webové aplikaci.

Klíčová slova

Jednorázová hesla, OTP, dvoufaktorová autentizace, HOTP, TOTP

Abstract

This Bachelor thesis describes the one-time passwords and their various implementation possibilities. The one-time passwords are passwords, which are valid only for one login or transaction. In contrast to regular passwords, one-time passwords are immune against reply attacks, which means that if a possible attacker eavedroppes the used password, he cannot exploit it, because it is not valid anymore. The theoretical section of the thesis explains the reasons for using one-time passwords and demonstrates some particular OTP systems. The practical section shows two different examples of using open-source OTP systems. The first one describes the use of OTPasswd for authentication to a server over SSH, the other one uses MultiOTP PHP class for authentication to a web application.

Keywords

One-time password, OTP, two-factor authentication, HOTP, TOTP

Obsah

1. Úvod

Tématem této bakalářské práce jsou systémy jednorázových hesel.

V teoretické části se budu nejprve snažit vysvětlit, co jednorázová hesla jsou, jaké důvody vedou k jejich používání a před čím nás naopak neochrání. Poté popíšu veškeré způsoby tvorby jednorázových hesel a metody jejich distribuce k uživateli. Současně budu uvádět příklady konkrétních systémů a výhody a nevýhody jednotlivých řešení. V závěrečné části se pokusím některé metody srovnat a uvést možná bezpečnostní rizika.

. V praktické části se pokusím na dvou rozdílných příkladech ukázat různé přístupy k implementaci open-source systému jednorázových hesel. V prvním příkladu popíšu instalaci systému OTPasswd a následně předvedu dvoukanálovou autentizaci k serveru přes SSH s využitím jednorázových hesel zasílaných přes SMS zprávy. V druhém příkladu využiju PHP třídu MultiOTP pro autentizaci k ukázkové webové aplikaci s využitím různých softwarových mobilních kalkulátorů jednorázových hesel.

Toto téma jsem si zvolil z důvodu jeho aktuálnosti. Myšlenka jednorázových hesel není sice nijak nová či převratná, ale v dnešní době se stále více společností poohlíží po vícefaktorové autentizaci k webovým aplikacím (v nedávné době ji zavedl například Facebook či Google) a jednorázová hesla jsou oblíbeným řešením pro zvýšení bezpečnosti. Stejně tak téměř všechny banky používají nějakou formu jednorázových hesel pro přístup k internetovému bankovnictví.

1.1.Autentizace

Důležitou funkcionalitou informačních systémů je autentizace uživatele. Bez jistoty identity uživatele vstupujícího do systému ztrácí řada činností smysl. Při autentizaci prokazuje uživatel svou identitu tím, že poskytne určitá data, která má k dispozici pouze on.

Autentizaci uživatele je možné provést třemi metodami:

- Uživatel něco ví (pamatuje si heslo či PIN[1])
- Uživatel něco má (čipovou kartu, mobilní telefon, digitální certifikát apod.)
- Uživatel někým je (test biologických charakteristik člověka)

Při použití více než dvou zcela nezávislých metod autentizace (kombinací z výše uvedených skupin) mluvíme o dvoufaktorové (vícefaktorové) autentizaci. Prolomení jednoho z faktorů útočníkovi nestačí, protože k autentizaci je zapotřebí ještě druhý faktor. *[Kněžek, 2001]*

1.2.Klasická hesla

Nejjednodušším případem autentizace je využití jednoduchého hesla – alfanumerického řetězce. Důvodem je jednoduchost a nízká cena implementace - systém pouze porovná řetězec poskytnutý uživatelem s řetězcem uloženým v databázi na straně serveru. Snadná možnost uhodnutí hesla však nutí provozovatele systémů k nastavení přísných kriterií pro kvalitu řetězce tvořícího heslo a pro periodu změny hesla. Pro uživatele to znamená, že je pro něj obtížné si různá hesla pamatovat. I při používání kvalitních hesel je jednoduché klasické heslo uhádnout či odposlechnout keyloggerem (software či hardware, který sleduje a eviduje stisknuté klávesy).

Prevence před získáním hesel nabouráním se do databáze systému je jednoduchá – místo samotného hesla je v databázi uložen pouze jeho hash[2]. V okamžiku, kdy uživatel zadá heslo, aby se autentizoval, systém zavolá na zadané heslo hashovací funkci a výsledek se porovná s údajem uloženým v systému.

[1] Personal Identification Number
[2] Výsledek funkce, která řetězec znaků o libovolné délce převede na řetězec s pevnou délkou.

1.3.Jednorázová hesla

Jednorázová hesla (OTP – One Time Password) jsou hesla, které jsou platná pouze pro jedno přihlášení nebo transakci. Na rozdíl od klasických (statických) hesel jsou odolná proti opakovaným útokům. To znamená, že když potenciální vetřelec odposlechne heslo, které bylo použito k přihlášení nebo transakci, není schopen je zneužít, protože již není platné. Důležité je zvolit kvalitní metodu generování jednorázových hesel, aby útočník nemohl následující jednorázové heslo vypočítat. Detailní popis použitého algoritmu pro tvorbu jednorázových hesel nebývá u proprietárních (komerčních) systémů zveřejňován – výrobci jej často považují za své vlastnictví. Přesto existují standardy, které budou v práci dále popsány. Na druhou stranu je těžké si jednorázová hesla zapamatovat, proto potřebujeme různé technologie distribuce jednorázových hesel k uživateli. Spadají tedy do kategorie „uživatel něco má“.

Jednorázová hesla se při autentizaci používají ze tří důvodů:

- Ochrana před odposlechnutím hesla při zadávání
- Ochrana před odposlechnutím hesla při přenosu
- Jako jeden z faktorů ve vícefaktorové autentizaci

Typickým použitím jednorázových hesel je přihlašování k webovým aplikacím na nedůvěryhodných stanicích (veřejné počítače v internetových kavárnách, hotelích, ve škole), kde může být nainstalován keylogger, který odposlechne heslo při zadávání uživatelem. Keylogger bývá jednoduchá softwarová aplikace, hardwarové zařízení vložené mezi PC a klávesnici, existují však i varianty fungující na úrovni BIOSu.

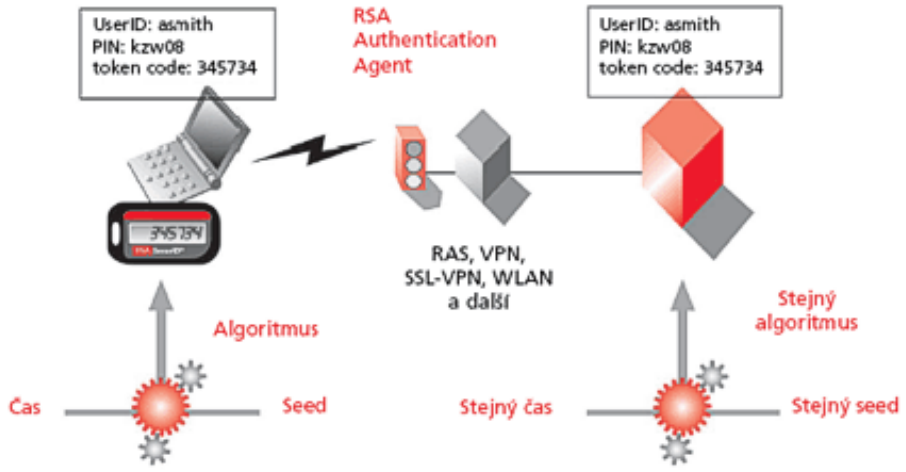
I v případě, že nedůvěryhodné počítače nepoužíváme, můžeme jednorázová hesla použít jako ochranu před odposlechnutím hesla při přenosu, pokud z nějakého důvodu nechceme nebo nemůžeme použít šifrovaný protokol TLS. (podpora hostingu, výkonnostní důvody) *[Vrána, 2006]*

Třetím důvodem je použití jednorázových hesel pro zvýšení bezpečnosti jako jeden z faktorů ve vícefaktorové autentizaci. S oblibou se používají v internetovém bankovnictví. *[Kmeť, 2006]*

V případě autentizace k virtuální privátní síti či bezdrátové síti je zapotřebí takzvaná vzájemná autentizace, která nás ochrání před podvrženým přístupovým bodem. K tomu můžeme použít protokol EAP-POTP[1] *[Nystroem, 2007].*

Systémy jednorázových hesel se obvykle skládají ze tří komponent:

- Autentizační server, který vyřizuje požadavky na ověření a zároveň udržuje databázi uživatelů
- autentizační agent, který je instalován na zabezpečovaný systém, a představuje zprostředkovatele autentizace - přijme uživatelem poskytnutá autentizační data a požadavek na ověření předá autentizačnímu serveru
- uživatelův zdroj jednorázových hesel - HW zařízení, SW aplikace, vytištěný seznam jednorázových hesel, SMS



Obr. 1.1. Využití autentizačního agenta **Zdroj: xanadu.cz**

Na obrázku 1.1 je znázorněna autentizace pomocí časově synchronizovaného hardwarového kalkulátoru RSA SecurID. Uživatel zadá zobrazené jednorázové heslo autentizačnímu agentovi, který je ověří na autentizačním serveru.

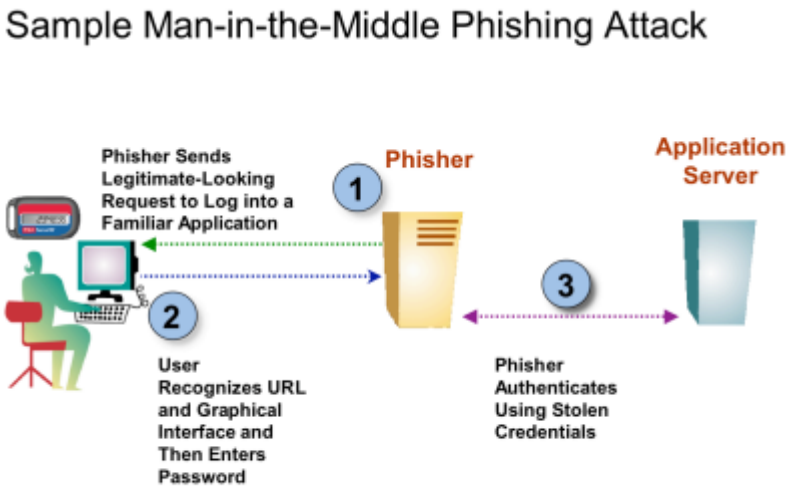
U některých systémů nemusí být jednotlivé komponenty jednoznačně rozlišeny, zdroj jednorázových hesel může být součástí autentizačního agenta nebo naopak autentizační agent může být sloučen s autentizačním serverem.

[1] http://tools.ietf.org/html/rfc4793

1.4.Co jednorázová hesla neřeší?

Společnost Trusteer uvádí použití jakéhokoliv systému jednorázových hesel jako účinnou ochranu před běžnými keyloggery. Stoprocentní jistotu však mít nemůžeme. Teoreticky může být na PC nainstalován speciální keylogger, který jednorázové heslo odchytí, okamžitě zašle útočníkovi a přeruší spojení uživatele se serverem. Jediná možná 100% ochrana je šifrování úhozů na úrovni ovladače klávesnice a dešifrování těsně před předáním webové aplikaci. *[Trusteer, 2007]*

Největším bezpečnostním rizikem je však útok „člověka uprostřed“ (MITM, man-in-the-middle attack). Podle společnosti Yubico neexistuje žádná technologie zamezující útoku člověka uprostřed, která by byla cenově příznivá a uživatelsky přívětivá. *[Yubico, 2009]* Útočník při něm vytvoří na svém serveru věrnou kopii přihlašovací obrazovky a nějakým způsobem přinutí uživatele, aby se na něj pokusil přihlásit (např. odkazem v e-mailu, modifikací DNS záznamu). Poté pomocí skriptu přesměrovává komunikaci uživatele a serveru. Útočník se na straně uživatele vydává za server a na straně serveru se vydává za uživatele. Obě strany tak mají zdání, že komunikují přímo mezi sebou.



Obr. 1.2. Útok člověka uprostřed komunikace **Zdroj:rsa.com**

Dalším rizikem zůstávají různé metody sociálního inženýrství, kdy útočník nějakým způsobem donutí uživatele, aby mu aktuální jednorázové heslo sdělil.

2. Další metody zabezpečení

2.1.Biometrie

Tato metoda autentizace je založena na rozpoznání fyzických charakteristik uživatele. Biometrických vlastností člověka, které je možné měřit, je celá řada. Patří mezi ně například otisk prstu, otisk dlaně, oční sítnice, oční duhovka, obličej, hlas.

Mezi otiskem prstu a běžným heslem není z našeho pohledu příliš velký rozdíl. Jelikož se otisk nemění, bylo by jej možné na síti odchytit a následně zneužít. [Dostálek, 2001]

Biometrie má několik omezení, se kterými je zapotřebí počítat. Jsou místa a situace, kdy je její nasazení zbytečné drahé nebo přímo nevhodné. Především není stoprocentní – pracuje s do jisté míry abstraktními hodnotami. Vždy budou existovat získané hodnoty „téměř jisté“.

Dále je zapotřebí mít vždy připravená náhradní řešení (fyzické postižení či úraz a následná nemožnost poskytnout požadované „údaje“). Nakonec je tu pomalost sytému, který musí zpracovávat ohromné množství údajů. *[Přibyl, 2009]* Dalším problémem může být citelný zásah do soukromí lidí. *[Koláček, 2009].*

2.2.Asymetrická kryptografie

Asymetrická kryptografie je založena na jednocestných funkcích. Je využíván veřejný a soukromý klíč. Autentizace probíhá tak, že server vygeneruje náhodné číslo, které předá uživateli. Ten jej za pomoci svého soukromého klíče digitálně podepíše a předá serveru, který provede verifikaci na základě dešifrovacího soukromého klíče. Veřejné klíče jsou elektronicky podepsané certifikační autoritou, která ověřuje pravdivost údajů.

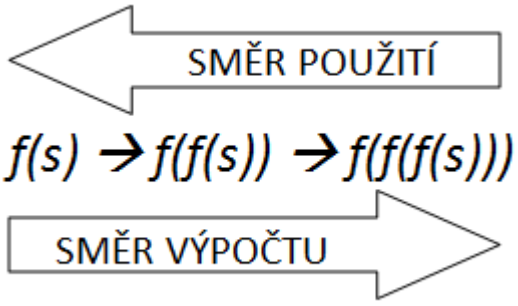
Soukromý klíč může být buď uložen na disku, což je sice praktické, ale lze jej snadno odcizit. Druhou možností je hardwarový klíč, což je technické zařízení (USB token, čipová karta), kde je klíč soukromý klíč uložený. Použití hardwarových klíčů v prostředí kontrolovaném třetí stranou (veřejné PC) se považuje za nebezpečné. *[Dostálek, 2001]*

3. Způsoby generování jednorázových hesel

3.1.Metoda založená na matematickém algoritmu

Nejznámějším algoritmem tvorby jednorázových hesel je tzv. Lamportovo schéma, které vytvořil v roce 1981 americký počítačový vědec Leslie Lamport. Každé nové heslo vytvořeno na základě předchozího použitého hesla a jednotlivá hesla musí být použita v předdefinovaném pořadí. *[Lamport, 1981]*

Tento algoritmus používá pro tvorbu hesel některou z hashovacích funkcí. Označme jej *f*. Dále si uživatel musí zvolit nějaký řetězec. Nazývá se seed (*s*) a je to uživatelovo tajemství. Hash řetězce vyjádříme jako *f(s)*. Pokud použijeme algoritmus dvakrát opakovaně, tj. *f(f(s))*, budeme psát *f2(s)* a obdobně *fn(s)* znamená, že jsme algoritmus *f* na řetězec *s* použili celkem *n*-krát. Při inicializaci se uživatel a správce aplikace dohodnou na čísle, např. 1000. Uživatel vyrobí *f1000(s)* a předá jej správci aplikace. Správce uloží do databáze k uživateli číslo 1000 a hodnotu *f1000(s).* Při autentizaci zašle server uživateli dotaz obsahující číslo (*n-1*), tedy nyní 999. Uživatel vygeneruje *f999(s)* a odešle serveru. Server provede porovnání *f(f999(s)) = f1000(s)*. Pokud jsou hodnoty stejné, je uživateli umožněn přístup a server uloží do databáze místo hodnoty *f1000(s)* hodnotu *f999(s)*, tedy při další autentizaci se provádí porovnání *f(f998(s)) = f999(s)*. Po použití všech 999 hesel musí uživatel změnit hodnotu řetězce seed a předat správci serveru novou hodnotu *f1000(s)*. *[Dostálek, 2001]*



Obr. 3.1. **Zdroj: Autor**

Pokud útočník heslo odposlechne, musí pro zjištění následujícího hesla vypočítat inverzní funkci, což je při použití hashovací funkce extrémně složité.

Na Lamportově schématu je založen systém jednorázových hesel S/KEY. Byl vytvořen počátkem 90. let 20. století společností Bellcore pro autentizaci v Unixu. Jako hashovací funkci používá MD4 a je popsán v RFC-1760[1]. *[Haller, 1995]*

[1] Request for Comments, seznam standardů

Mechanismus S/KEY byl dále rozšířen v OTP (A One Time Password System), popsáno v RFC-2289. *[Haller, 1998]* (původně RFC-1938). Přidává podporu dalších hashovacích algoritmů jako MD5 či SHA-1. Implementací systému S/KEY je projekt nazvaný OPIE („One time Passwords In Everything"), vytvořený v roce 1994 v U. S. Naval Research Laboratory. Kromě opravy mnohých chyb přidal podporu MD5 a poskytnul širší podporu operačních systémů a jednodušší systém instalace. *[McDonald, 1995]*

V roce 1995 popsal A.D. Rubin ze společnosti Bellcore systém tvorby jednorázových hesel, která na sobě nejsou závislá. Místo hodnoty posledního hesla je v databázi uložen čítač a tajný klíč, ze kterého se heslo spočítá. *[Rubin, 1995]*

Velmi rozšířeným algoritmem je HOTP („HMAC-based One Time Password"). Byl vyvinutý hnutím OATH (Initiative For Open Authentication) a publikován roku 2005 v RFC-4226 včetně implementace v Javě. HOTP je otevřený standard a mnoho výrobců nabízí hardwarové i softwarové tokeny, které tento algoritmus implementují. Jeho základem je synchronní tajný klíč, čítač a HMAC-SHA-1[1], což je mechanismus kombinující hashovací funkci SHA-1 s tajným klíčem – vstupem funkce nejsou pouze zpracovávaná data (čítač), ale i tajný klíč. *[Krawczyk et al., 1997]* Výstupem HMAC-SHA-1 je 160 bitový řetězec, který je následně oříznut a převeden na 6-8 místné číselné jednorázové heslo. *[Raihi et al., 2005]*

1.1.Metoda založená na časové synchronizaci

Pro tuto metodu je zapotřebí, aby každý uživatel obdržel autentizační kalkulátor, který obsahuje přesné hodiny. Ty jsou před distribucí k uživateli či při prvním použití synchronizovány s hodinami autentizačního serveru. Jednorázová hesla se tvoří jako hash aktuálního času a sdíleného tajného klíče. Ten musí být uložen v databázi na autentizačním serveru a také musí být vložen do kalkulátoru před distribucí k uživateli; může být i nastaven napevno při výrobě.

[1] Hash-based Message Authentication Code, http://rfc-ref.org/RFC-TEXTS/2104/

Jedno heslo tedy platí pouze po velmi krátkou dobu (obvykle půl minuty nebo minutu). Při autentizaci zašle uživatel serveru jednorázové heslo zobrazené na kalkulátoru a server je porovná s heslem, které vypočítal ze svého času. V případě shody je klientovi umožněn přístup.

Pokud se hodnoty neshodují, může to být způsobeno tím, že se hodiny rozházejí. V takovém případě server vypočítá některé hodnoty o několik minut vpřed i vzad a hledá, nedojde-li ke shodě. Pokud ano, umožní klientovi přístup a do konfiguračního souboru si poznamená časový posun. *[Dostálek, 2001]*

Autentizačním kalkulátorem může být hardwarové zařízení s displejem stejně tak jako softwarová (mobilní) aplikace. Heslo se každých 30s resp. 60s mění a automaticky se zobrazuje na displeji.

Standard popisující metodu tvoření hesel na základě časové synchronizace se nazývá TOTP (Time-based One-time Password Algorithm). Jedná se o rozšíření HOTP a pochází též od hnutí OATH. V současné době se nachází ve stádiu návrhu na RFC, poslední verze je z 24. února 2011. Princip je stejný jako u HOTP, pouze místo čítače je vstupem pro HMAC počet časových kroků mezi počátečním časem a současným unixovým časem[1]. Časový krok a počáteční čas jsou systémové parametry (výchozí hodnoty jsou časový krok 30s a výchozí čas 0). *[Raihi et al., 2011a]*

1.2.Metoda založená na výzvě a odpovědi

Při použití této metody vygeneruje server náhodný řetězec jako výzvu (challenge). Klient tento náhodný řetězec spojí se sdíleným tajným klíčem a z takto vytvořeného celku spočte hash, který vrátí jako jednorázové heslo (response) serveru. Jelikož server zná tajný klíč, je schopen stejné jednorázové heslo rovněž spočítat. Oba výsledky porovná, a pokud jsou shodné, umožní klientovi přístup do aplikace. Důležité je, aby se nezopakoval v krátkém časovém intervalu stejný dotaz. Je proto potřeba použít kvalitní systém na generování dotazů (dotaz může být např. zřetězením času a náhodného čísla apod.). *[Dostálek, 2001]*

[1] Počet sekund uplynulých od půlnoci UTC 1. ledna 1970

Zjednodušeně lze říci, že podobný mechanismus se použije i při autentizaci klienta pomocí asymetrické kryptografie. Server vygeneruje náhodné číslo a klient vrátí elektronický podpis tohoto čísla. *[Dostálek, 2001]*

Kalkulátory také mohou počítat hash z jednotlivých položek předávaných dat, příkladem může být zadávání bankovních příkazů. Při autorizaci platebního příkazu je zde nutné zadat na klávesnici kalkulátoru také důležitá pole příkazu. Kromě zadávání platebního příkazu do webového formuláře musí tedy uživatel zadat nějaká data ještě jednou do autentizačního kalkulátoru a zobrazené jednorázové heslo vložit do formuláře. *[Dostálek, 2001]*

Na stejném principu výzva-odpověď funguje Turingův test[1] CAPTCHA (Capture Characters), i když zde nepoužíváme přímo termín jednorázová hesla. Výzvou je zde obrázek se strojově těžko čitelným textem a odpovědí je řetězec, který z obrázku člověk snadno vyčte.

OATH standard využívající metodu výzva-odpověď se nazývá OCRA (OATH Challenge-Response Algoritmus), a je zatím stejně jako TOTP ve stádiu návrhu, poslední verze je z března 2011. *[Raihi et al., 2011b]*

1.3.Metoda založená na generátoru náhodných čísel

Poslední možností je tvoření hesel pomocí generátoru náhodných, resp. pseudonáhodných čísel. Pseudonáhodná čísla jsou čísla, která vytvářejí zdánlivě náhodnou posloupnost (pravé náhodnosti nelze na počítači přímo dosáhnout).

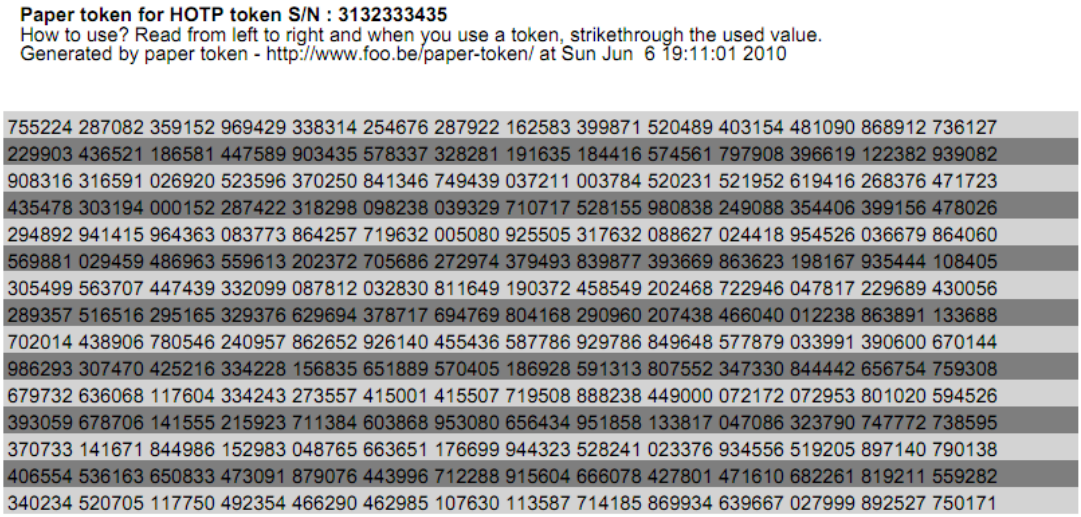
Abychom mohli náhodná čísla používat jako jednorázová hesla, musíme je předem vytvořit, všechna uložit na server a zároveň vytisknout na papír a předat uživateli. Tuto variantu používá například systém LastPass[2].

[1] Test, který má za cíl odlišit člověka od stroje
[2] https://lastpass.com/otp.php

2. Způsoby distribuce jednorázových hesel

2.1.Seznam hesel

V nejjednodušším případě je vygenerován seznam hesel, který může být vytištěn na papír a předán uživateli. Na straně serveru může být buď stejný seznam, nebo údaje pro vypočítání hesel. Uživatel pak pro svou autentizaci zadává ze svého vytištěného seznamu jedno heslo po druhém a po zadání hesla si jej v seznamu škrtne. Jednorázová hesla mohou být i očíslována a systém při autentizaci napoví uživateli, kolikáté heslo má použít. Hesla ani nemusí být zadávána po sobě, některé systémy při autentizaci zobrazují náhodné číslo hesla, které uživatel musí zadat. Náhodný výběr hesel může být i s opakováním, pak se v podstatě nejedná o jednorázové heslo, požadavek na výběr dvou stejných hesel v krátké době je ale málo pravděpodobný. Pro zvýšení bezpečnosti se použití jednorázových hesel často kombinuje s klasickým heslem. Uživatel pak zadává heslo skládající se ze dvou částí: z klasického hesla, které si pamatuje, a z jednorázového hesla. *[Dostálek, 2001]*



Paper token for HOTP token S/N : 3132333435
How to use? Read from left to right and when you use a token, strikethrough the used value.
Generated by paper token - http://www.foo.be/paper-token/ at Sun Jun 6 19:11:01 2010

755224	287082	359152	969429	338314	254676	287922	162583	399871	520489	403154	481090	868912	736127
229903	436521	186581	447589	903435	578337	328281	191635	184416	574561	797908	396619	122382	939082
908316	316591	026920	523596	370250	841346	749439	037211	003784	520231	521952	619416	268376	471723
435478	303194	000152	287422	318298	098238	039329	710717	528155	980838	249088	354406	399156	478026
294892	941415	964363	083773	864257	719632	005080	925505	317632	088627	024418	954526	036679	864060
569881	029459	486963	559613	202372	705686	272974	379493	839877	393669	863623	198167	935444	108405
305499	563707	447439	332099	087812	032830	811649	190372	458549	202468	722946	047817	229689	430056
289357	516516	295165	329376	629694	378717	694769	804168	290960	207438	466040	012238	863891	133688
702014	438906	780546	240957	862652	926140	455436	587786	929786	849648	577879	033991	390600	670144
986293	307470	425216	334228	156835	651889	570405	186928	591313	807552	347330	844442	656754	759308
679732	636068	117604	334243	273557	415001	415507	719508	888238	449000	072172	072953	801020	594526
393059	678706	141555	215923	711384	603868	953080	656434	951858	133817	047086	323790	747772	738595
370733	141671	844986	152983	048765	663651	176699	944323	528241	023376	934556	519205	897140	790138
406554	536163	650833	473091	879076	443996	712288	915604	666078	427801	471610	682261	819211	559282
340234	520705	117750	492354	466290	462985	107630	113587	714185	869934	639667	027999	892527	750171

Obr. 3.1. Ukázka vytištěného seznamu jednorázových hesel **zdroj: http://foo.be/paper-token**

Seznam hesel si uživatel může uchovávat i v elektronické podobě, podstata je však stejná. Jednorázová hesla jsou vygenerována dopředu a po použití všech je třeba pořídit seznam nových hesel a bezpečně jej předat uživateli. U některých systémů si seznam nových hesel musí vždy vygenerovat sám uživatel. Nevýhodou seznamu je to, že uživatel jej vždy při autentizaci musí mít k dispozici. Jednoduše se tak může stát, že jej někde zapomene nebo ztratí. Naopak výhoda spočívá v zanedbatelných pořizovacích nákladech.

2.2.Hardwarové autentizační kalkulátory (tokeny)

Jedná se o malá fyzická zařízení, která jsou přizpůsobena k pohodlnému každodennímu nošení v kapse, peněžence či jako přívěsek na klíčích. (obr. 4.1.) Existují však i stolní varianty, jako například kalkulátor ActiveIdentity[1] na obrázku 4.2. Bezpečnostní tokeny mohou obsahovat digitální podpisy, biometrická data apod., my se budeme zabývat pouze těmi tokeny, které slouží ke generování jednorázových hesel. Jejich součástí je displej zobrazující heslo, případně ještě malá klávesnice pro zadání PIN.

Obr. 4.1. Zdroj: feitian.com

Obr. 4.2. Zdroj: activeidentity.com

Významnou výhodou autentizačních kalkulátorů je jejich naprostá nezávislost na bezpečnosti prostředí. U některých klientů však nemusí být populární nosit kalkulátor stále s sebou. *[Dostálek, 2001]* Kalkulátor také může uživatel někde ztratit, zapomenout, nebo mu může být odcizen. Další nevýhodou je nutnost udržovat kalkulátor stále nabitý (některé kalkulátory nelze nabíjet a musí u nich být vyměněna baterie, kalkulátor obvykle úroveň nabití či nutnost výměny baterie signalizuje).

Obr.4.3. Zdroj: verisign.com

Princip generování jednorázových hesel v autentizačním kalkulátoru může být založen na všech výše popsaných metodách (matematický algoritmus, časová synchronizace i výzva-odpověď). Dále je potřeba rozlišovat tokeny proprietární a tokeny implementující některý z RFC standardů.

[1] http://www.activeidentity.com

Velmi rozšířené jsou tokeny společnosti VeriSign[1] jejíž řešení využívá k bezpečnější autentizaci mimo jiné eBay a PayPal. VeriSign je členem OATH a tokeny tedy implementují HOTP a TOTP standardy. Uživatel si může vybrat ze tří typů tokenů. (obr. 4.3.)

Dalšími příklady hardwarových tokenů jsou např. Deepnet Security SafeID[2], Vasco DigiPass[3] či SecureMetric SecureOTP[4].

Tokeny mohou kombinovat kalkulátor jednorázových hesel s dalšími funkcemi. Jako příklad uveďme hybridní token SafeNet eTokenPass kombinující generátor hesel s úložištěm pro digitální podpis (obr. 4.4.) a Epass s čtečkou otisků prstů. (obr. 4.5)

Obr. 4.4. Zdroj: safenet-inc.com

Obr. 4.5. Zdroj: josafe.com

Zajímavou možností je token rozměrů běžné platební karty. Karta má obvyklé rozměry i tloušťku, ale obsahuje displej a klávesnici. Po zadání PIN se na displeji zobrazí jednorázové heslo. Příkladem je NagraID Security 306[5]. Výrobce nabízí verzi jak pro OATH HOTP tak i pro TOTP.

V roce 2010 oznámila společnost VISA uvedení platebních karet s kalkulátorem jednorázových hesel pod označením Visa CodeSecure. Mají sloužit k bezpečnějším online platbám. *[Finextra, 2010]*

Obr. 4.6. Zdroj: visa.com

[1] http://www.verisign.com
[2] http://www.deepnetsecurity.com/tokens/hardware/safeid/
[3] http://www.vasco.com
[4] http://www.securemetric.com/
[5] http://www.nidsecurity.com

Pro bezpečnější online používání klasických platebních karet existují jejich čtečky. Mohou buď jednoduše generovat jednorázové heslo např. pro přihlášení k internetovému bankovnictví, nebo autorizovat transakční informace jak bylo popsáno kapitole 3.3. Na obrázcích 4.7. a 4.8. jsou čtečky Vasco Digipass[1], které implementují standard Mastercard CAP (Chip Authentication Protokol).
[Drimer, Murdoch, Anderson, 2009]

Obr. 4.7. Zdroj: vasco.com

Obr. 4.8. Zdroj: vasco.com

Uživatel ani nemusí o využití sytému jednorázových hesel vědět. Příkladem je Swekey[2] (obr.4.7.) , což je USB zařízení obsahující kalkulátor jednorázových hesel založený na systému výzva-odpověď. Přítomnost zařízení a autentizace je řízena přes JavaScript. Uživatelské jméno se po připojení tokenu vyplní automaticky, uživatel pouze zadá klasické heslo a o autentizaci pomocí jednorázových hesel nemusí mít ani tušení. Swekey nabízí podporu pro phpBB, WordPress, Drupal, Joomla a mnoho dalších aplikací.

Podobným uživatelsky velice přívětivým systémem je YubiKey společnosti Yubico. Jedná se o USB token, který se chová jako USB klávesnice a po stisknutí tlačítka se sám postará o autentizaci založenou na HOTP.[3]

Obr. 4.7. Zdroj: swekey.com

Obr. 4.8. Zdroj: yubico.com

[1] http://www.vasco.com
[2] http://www.swekey.com

2.3.Softwarové autentizační kalkulátory

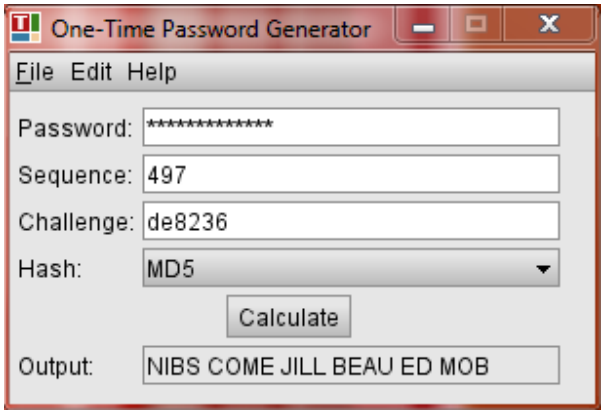
Nevýhodou autentizačních kalkulátorů je samotná existence kalkulátoru, tj. z hlediska provozovatele aplikace se kalkulátor musí pořídit, což není laciné. Z hlediska uživatele je zase nepříjemné, že se kalkulátor musí stále nosit s sebou. Naopak mobilní telefon má dnes téměř každý. Uživatelé jsou zvyklí jej s sebou nosit a starat se o něj a navíc si mobilní telefon pořizuje uživatel sám. *[Dostálek, 2001]*

Většina výrobců proto nabízí kromě hardwarového tokenu také softwarový token v podobě aplikace do mobilního telefonu. Stejně tak existuje mnoho open-source kalkulátorů implementující RFC standardy.

j2me-otp[1] je jednoduchý mobilní S/KEY kalkulátor napsaný v Javě. Je to vlastně J2ME verze online kalkulátoru jotp[2] Jeho alternativou je OTPGen.[3] Kromě javových kalkulátorů existují i specifické verze např. pro iPhone či Android.

Pro OATH HOTP je k dispozici oathtoken[4]. TOTP podporuje CAT (Cellular Authentication Token)[5] či KerPass[6]. Na časové synchronizaci je též založen projekt Mobile-OTP[7]. Mezi komerční softwarové tokeny patří Aradiom SolidPass[8]. Aradiom nabízí mnoho druhů mobilních tokenů založených na HOTP i TOTP ve verzích pro nejrůznější platformy.[9]

Softwarové kalkulátory nemusí být nutně mobilní aplikace, existují také kalkulátory na PC (obr. 4.9.). Příkladem je RSA SecurID Software Token for Microsoft Windows nebo S/KEY One-Time Password Generator.[10]



Obr. 4.9. Zdroj: softpedia.com

[1] http://tanso.net/j2me-otp/
[2] http://www.cs.umd.edu/users/harry/jotp/
[3] http://marcin.studio4plus.com/en/otpgen/
[4] http://code.google.com/p/oathtoken/
[5] http://www.megaas.com/CATProductsToken.asp
[6] http://www.kerpass.com
[7] http://motp.sourceforge.net/
[8] http://www.aradiom.com
[9] http://www.solidpass.com/
[10] http://otp-java.sourceforge.net/

2.4.Autentizace přes webové rozhraní

Existují různé webové služby, které pro distribuci hesla k uživateli nevyžadují žádný seznam hesel, kalkulátor ani nic podobného. Vše může probíhat pouze na straně serveru.

Metoda Confident ImageShield[1] je založena na schopnosti uživatele rozeznat předem vybrané kategorie zobrazených obrázků. Při registraci na serveru vybere uživatel několik kategorií věcí jako např. psi, auta, květiny. Při přihlašování se uživateli vždy zobrazí náhodně generovaná mřížka obrázků. Každý obrázek v mřížce má vygenerovaný alfanumerický znak. Uživatel najde obrázky spadající do předem vybraných kategorií a do vstupního formuláře napíše pro každý vybraný obrázek jeho vygenerovaný znak. Tím uživatel vytvoří jednorázové heslo a je mu umožněn přístup. (viz. obr. 4.10. – pokud si uživatel při registraci zvolí např. skupiny *lidé*, *květiny* a *hodiny*, je jednorázové heslo *CEI*) *[Needham, 2010]*

Obr. 4.10. **Zdroj: confidenttechnologies.com**

[1] http://www.confidenttechnologies.com

2.5.Jednorázová hesla přes SMS

Běžnou metodou pro doručování jednorázových hesel je SMS. Aplikace vygeneruje jednorázové heslo a v databázi uživatelů najde číslo jeho mobilního telefonu, na které pomocí SMS zprávy jednorázové heslo zašle.

Tento způsob autorizace kombinuje dva nezávislé komunikační kanály. Tato skutečnost podstatným způsobem omezuje možnost zneužití, protože případný útok by musel být veden společně na oba nezávislé kanály, což je vysoce náročné. Další podstatnou výhodou jsou nízké pořizovací náklady a relativně jednoduchá obsluha. Jistým omezením tohoto řešení je, že klient musí být vybaven mobilním telefonem a že tento pracuje pouze v místě, kde má dostupný signál. *[Dostálek, 2001]*

Pokud se SMS zasílají nešifrované, musíme mít na vědomí, že se mobilní operátor stává součástí důvěrného řetězce. K šifrování SMS slouží algoritmus A5/1, ve kterém však byly objeveny vážné bezpečnostní slabiny. *[Zandl, 1999]*

O šifrování může také starat speciální SIM Toolkitová aplikace. Takové aplikace se využívají u GSM Bankingu. K tomu je zapotřebí speciální SIM karta, kterou mobilní operátoři nabízejí. Službu musí podporovat jak mobilní operátor, tak banka. V chráněné oblasti SIM karty je uložen šifrovací klíč a zašifrované zprávy banky je schopna rozšifrovat pouze SIM Toolkit aplikace, mobilní operátor tuto možnost nemá. Používá se šifra 3DES. *[Řáda, 2004]*

Běžnou praxí je využití jednorázových hesel přes SMS v internetovém bankovnictví. Uživatel se přihlásí klasickým heslem a vždy před provedením nějaké transakce obdrží autorizační SMS s kódem, který musí zadat do formuláře.

Facebook zavedl jednorázová hesla přes SMS jako jednu z možností bezpečné antentizace koncem roku 2010. Služba však zatím funguje pouze v USA. Uživatel k svému účtu vloží telefonní číslo a při autentizaci pak pouze zašle SMS ve tvaru „OTP“ na číslo 32665. Obratem obdrží jednorázové heslo, které je platné 20 minut. *[Brill, 2010]*

Stejně tak Google zavedl v únoru 2011 pro všechny uživatele Google aplikací možnost vícefaktorové autentizace. Na výběr je forma SMS či mobilního kalkulátoru. *[Shah, 2011]*

3. Další systémy, které stojí za zmínku

3.1.RSA SecurID

RSA SecurID je proprietární autentizační systém vyvinutý společností RSA Security. Ta je jedničkou na trhu s bezpečnostními systémy. SecurID používá více než 40 milionů uživatelů v 30 tisících organizací po celém světě. *[Long, 2009].* K výběru je mnoho typů časově synchronizovaných tokenů, jak hardwarových, tak softwarových. Podle RSA je životnost jejich tokenu mezi 3 až 5 lety. K výpočtu hesel se od roku 2003 využívá 128-bitový řetězec a šifra AES. Předtím se používal proprietární hashovací algoritmus, jehož bezpečnost byla zpochybněna. *[Biryukov, Lano, Preneel, 2003]*. AES je dosud považován za bezpečný.

SecurID systém se skládá ze tří částí:

- Token na straně uživatele
- Aplikace, ke které se uživatel chce autentizovat (autentizační agent)
- Autentizační manažer

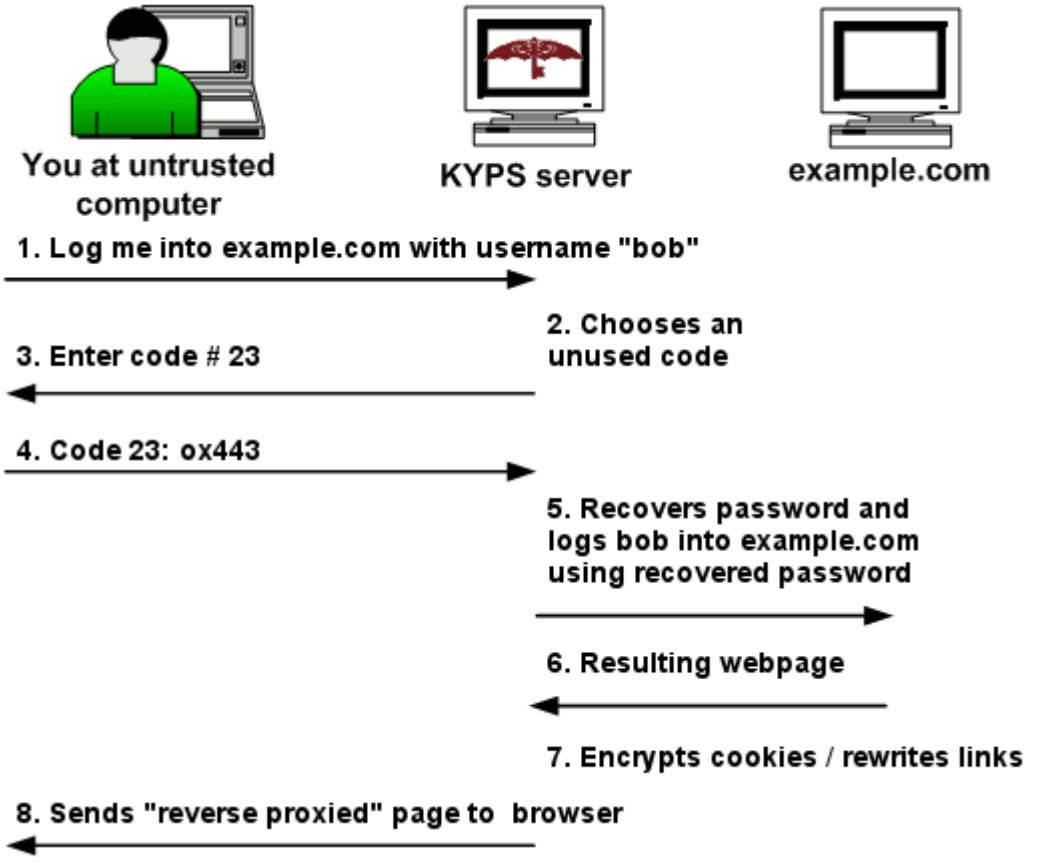
Při autentizaci zašle uživatel své jméno, PIN a vygenerované heslo autentizačním agentovi (webserver, VPN). Autentizační agent předá požadavek autentizačním manažerovi, který klienta autentizuje.

Obr. 5.1. Různé typy RSA kalkulátorů. Zdroj: rsa.com

Dne 17. března 2011 zaslala společnost RSA svým klientům otevřený dopis[1], podle kterého se stala obětí „extrémně sofistikovaného“ útoku, který měl za následek únik jistých informací. Ty by mohly být použity k „snížení efektivity současné implementace dvoufaktorové autentizace“ *[Stevens, 2011]*

3.2.KYPS

KYPS (Keep Your Password Secret)[2] je bezplatná webová služba, která umožňuje uživatelům přihlašovat se k různým webovým stránkám prostřednictvím jednorázových hesel. Uživatel má dvě možnosti; buď sdělí své klasické heslo k přihlášení na požadovanou stránku přímo serveru KYPS, který pak vygeneruje seznam jednorázových hesel, nebo sdělí své heslo Java Apletu, který jednorázová hesla vygeneruje na uživatelově počítači. Přihlášení pak za uživatele provádí server KYPS, jak je znázorněno na obrázku 5.2.



Obr. 5.2. Způsob autentizace prostřednictvím služby KYPS. **Zdroj: kyps.net**

[1] http://www.rsa.com/node.aspx?id=3872
[2] http://www.kyps.net

4. Porovnání bezpečnosti různých systémů jednorázových hesel

Prvním předpokladem bezpečnosti je vhodně zvolená hashovací funkce, která se používá pro generování jednorázových hesel. Systém S/KEY standardně používá funkce MD4 či MD5. V těchto funkcích byly v roce 2004 objeveny kolize[1], proto mohou být považovány za méně bezpečné. *[Klíma, 2004]* Bezpečnost SHA-1 byla též zpochybněna, ale při použití v HMAC spolu se sdíleným tajemstvím se ho kolize hashovacích funkcí netýkají. *[OATH, 2011]*

Problém může nastat, pokud útočník získá přístup na server, na kterém je uložena databáze sdílených tajemství jednotlivých uživatelů. Může se zdát jako výhoda, že S/KEY na serveru ukládá pouze poslední použité jednorázové heslo. Přesto je zapotřebí zaručit to, že se jeho hodnota nezmění. Pokud můžeme zaručit, že útočník nezíská práva root, můžeme zde uložit také sdílené tajemství. *[Rubin, 1995]*

Při použití vytištěného seznamu hesel hrozí útok „přes rameno“. Protože na seznamu jsou vytištěna následující hesla, útočník může seznam hesel například nepozorovaně vyfotografovat. Stejné nebezpečí hrozí u hardwarových kalkulátorů, které nejsou chráněny PINem a současná jednorázová hesla se na nich neustále zobrazují. *[Lindell, 2007]*

V případě jednorázových hesel přes SMS zase hrozí nainstalování mallware[2] do telefonu. V únoru 2011 se tomu tak stalo v Polsku. Zákazníci ING Bank byli na podvržené stránce vyzváni, aby zadali své telefonní číslo a model telefonu. Poté jim přišla SMS s odkazem na mallware, který po nainstalování přeposílal jednorázová hesla útočníkovi. *[Goodin,2011]* Dalším rizikem může být odposlechnutí nešifrované SMS zprávy nebo možnost změny telefonního čísla pro jejich zasílání ve webové aplikaci.

Ochranou před hádáním hesla hrubou silou je nastavení limitu počtu neúspěšných pokusů o přihlášení, po jehož překročení je uživatel zablokován. Dále je potřeba zvolit vhodnou délku jednorázového hesla a znaky, ze kterých se skládá.

[1] Stejný hash pro různé vstupy
[2] Souhrné označení veškerý škodlivý software (viry, trojské koně)

HOTP a TOTP kalkulátory jsou z hlediska bezpečnosti velmi podobné. Pro odcizení HOTP jednorázového hesla je nutný fyzický přístup ke kalkulátoru (stisknutí tlačítka) a není nutné je použít ihned. Naopak většina TOTP kalkulátorů stále zobrazují platná hesla, takže je útočník může nepozorovaně přečíst, ale za to platí pouze po omezenou dobu. U kalkulátorů chráněných PINem nehrozí výše uvedná rizika vůbec. *[Lindell, 2007]*

Vlastnosti	YUBIKEY	LCD TOKEN	Čtečka SMART CARD	SMS	Mobilní aplikace	Vytištěný seznam
Garantovaná jedinečnost hesla	ANO	ANO	ANO	ANO	NE	NE
Ochrana proti virům	ANO	ANO	ANO	ANO	?	ANO
Challenge-response klientský software	ANO	NE	ANO	NE	NE	NE
Automatické zadání OTP	ANO	NE	ANO	NE	NE	NE
Nezávislost na platformě	ANO	ANO	NE	ANO	NE	ANO
Vejde se do peněženky/na klíče	ANO	?	NE	?	?	ANO
Automatická identifikace	ANO	NE	ANO	NE	NE	NE
Funguje bez USB portu	NE	ANO	NE	ANO	ANO	ANO
Automatická synchronizace	ANO	NE	ANO	NE	NE	NE
Pro uživatele s vadou zraku	ANO	NE	ANO	NE	NE	NE
Nastavitelná délka statického hesla	ANO	NE	ANO	NE	NE	NE
Možnost dvou identit	ANO	NE	ANO	NE	NE	NE
Open-source server software	ANO	NE	?	NE	NE	NE
U klienta není potřeba nic instalovat	ANO	ANO	NE	ANO	NE	ANO
Nepotřebuje baterii	ANO	NE	ANO	NE	NE	ANO
Jde poslat běžnou poštou	ANO	?	NE	?	?	ANO
Žádné poplatky mobilnímu operátorovi	ANO	ANO	ANO	NE	ANO	ANO
Odolný vůči pádu či zničení	ANO	NE	NE	NE	NE	NE
Voděodolný	ANO	NE	NE	NE	NE	?

Tab. 1.1. Zdroj: yubico.com, zjednodušeno

5. První příklad – autentizace k serveru

5.1.Systém OTPasswd

Prvním praktickým příkladem je využití open-source systému jednorázových hesel OTPasswd při přihlašování k linuxovému serveru (Ubuntu 10.10 Server) prostřednictvím SSH.

Systém OTPasswd byl zvolen z toho důvodu, že je aktuální a stále ve vývoji a hlavně z důvodu podpory dvoukanálové autentizace - jednorázových hesel zasílaných přes SMS. OTPasswd implementuje a rozšiřuje systém jednorázových hesel PPPv3[1] (Perfect Paper Passwords) a může být nasazen ve všech systémech podporujících autentizační moduly PAM[2]. Alternativou by mohl být například PAM modul implementující HOTP autentizaci, projekt Barada[3].

Základem systému je šifra AES, která šifruje 128-bitové bloky 256-bitovým klíčem. Při tom se vygeneruje 128-bitů dat, která jsou použita k výpočtu hesla. Během generování klíče má uživatel 256bitový sekvenční klíč a 128bitový čítač. Kdykoliv je heslo validováno nebo generováno, čítač je zašifrován klíčem a tím je vytvořeno heslo. Na rozdíl od PPPv3 přidává OTPasswd podporu „solení“ čítače – u PPPv3 je čítač 128bitová hodnota začínající na nule. Útočník může heslo a čítač odposlechnout a tak mu ke spočítání následujícího hesla chybí pouze klíč. Prolomení AES je i bez „solení“ téměř nemožné, OTPasswd však do hodnoty čítače ukládá 96bitů náhodných dat a útočník hodnotu čítače zjistit nemůže. *[Fortuna, 2010]*

5.2.SSH

SSH je zabezpečený komunikační protokol, který se běžně používá při vzdáleném přístupu k shellu. SSH nabízí tři základní metody autentizace uživatele:

- Pomocí veřejného klíče
- Jednoduchým heslem
- Interaktivně přes klávesnici

[1] https://www.grc.com/ppp.htm
[2] Sada knihoven, jejichž cílem je oddělit aplikace od konkrétních autentizačních mechanismů.
[3] http://barada.sourceforge.net/

Pro autentizaci pomocí jednorázových hesel se používá třetí možnost. Je to univerzální metoda popsaná v RFC 4256[1], při které zasílá server uživateli výzvy a ten na ně pomocí klávesnice odpovídá. O vše ostatní se postará PAM modul.

5.3.Instalace

Prvním krokem instalace je samozřejmě stažení zdrojových kódů. Ty si musíme zkompilovat, k tomu potřebujeme balíčky *cmake, build-essential, gettext* a vývojářský balíček pro PAM *libpam0g-dev*.

```
root@ubuntu:~#: apt-get install cmake build-essential gettext libpam0g-dev
root@ubuntu:~#: wget http://download.savannah.gnu.org/releases/otpasswd/otpasswd-0.7_rc1.tar.bz2
root@ubuntu:~#: tar -jxvf otpasswd-0.7_rc1.tar.bz2
root@ubuntu:~#: cd otpasswd-0.7_rc1
root@ubuntu:~# cmake . && make
root@ubuntu:~# make install
```

V adresáři */etc/otpasswd* musíme vytvořit konfigurační soubor *otpasswd.conf* (zkopírujeme ukázkový soubor */etc/otpasswd/otpasswd.conf.dist*, možnosti nastavení konfiguračního souboru budou popsány dále). Dále je třeba nakonfigurovat SSH s PAM modulem OTPasswd. V souboru */etc/ssh/sshd_config* nastavíme hodnotu *ChallengeResponseAuthentication yes* a v souboru */etc/pam.d/sshd* zakomentujeme všechny řádky začínající *auth* a místo nich vložíme odkaz na soubor *otpasswd-login*. V případě Ubuntu 10.10 zakomentujeme řádek *auth required pam_env.so* a řádek *@include common-auth*. Místo nich vložíme řádek *auth include otpasswd –login.*

5.4.Konfigurace otpasswd.conf

Jediným povinným parametrem pro spuštění systému je nastavení operačního módu. OTPasswd totiž může běžet ve dvou módech, které se liší úložištěm uživatelských dat. V módu „USER DB“ jsou uživatelská data uchovávána v domovských adresářích uživatelů (soubor .otpasswd), to znamená, že uživatel je může libovolně měnit, použití jednorázových hesel je dobrovolné. Druhou možností je mód „GLOBAL DB“, který využívá systémové úložiště, do kterého uživatelé nemají přístup; administrátor tedy může nastavit pevná pravidla pro všechny uži-

[1] http://www.ietf.org/rfc/rfc4256.txt

vatele. V současné době se používá soubor */etc/otpasswd/otshaddow*, je plánována podpora MySQL. Pro náš příklad můžeme ponechat nastavení *DB=USER*.

Pokud budeme chtít používat zasílání hesel přes SMS, musíme v parametru *PAM_OOB* nastavit způsob autentizace. Máme čtyři možnosti:

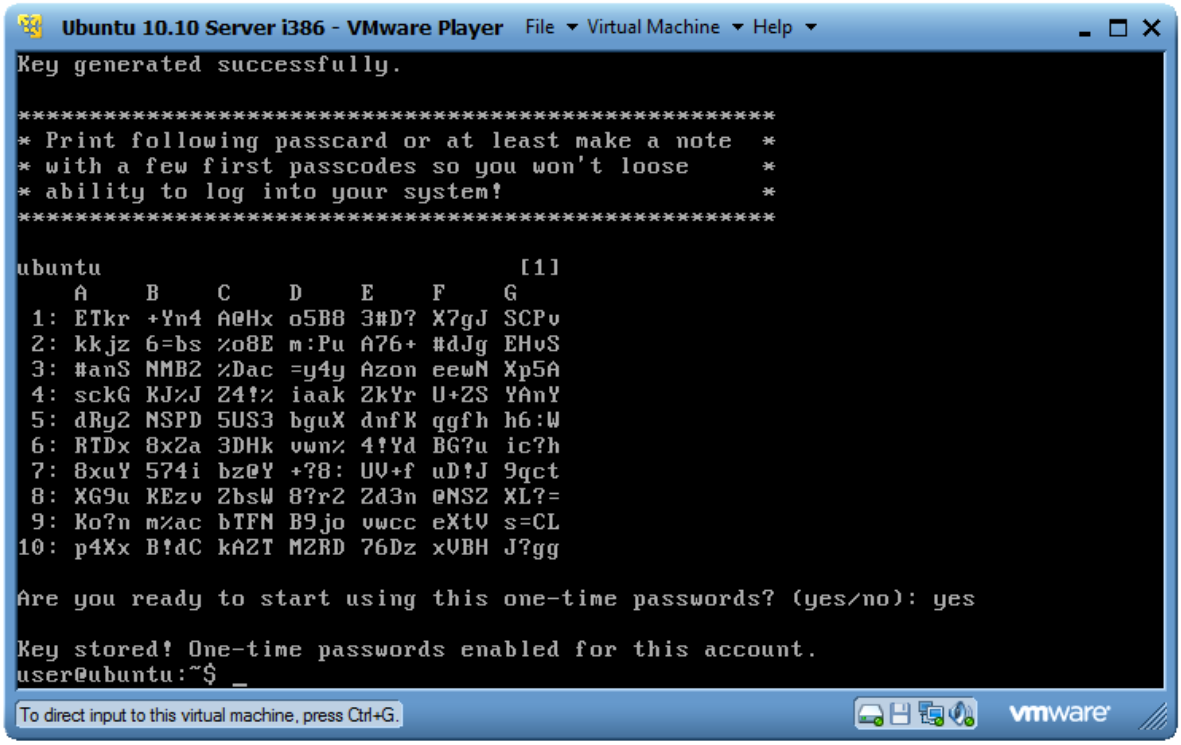
- *PAM_OOB=0* Zasílání SMS je vypnuto, používají se vytištěné karty s hesly
- *PAM_OOB=1* Zaslání zprávy na žádost uživatele napsáním tečky '.' místo hesla
- *PAM_OOB=2* Zaslání zprávy na žádost uživatele + běžné heslo
- *PAM_OOB=4* Zaslání SMS ihned na počátku autentizace

Dále musíme nastavit cestu ke skriptu, který zprávu odesílá a uživatele, pod kterým se skript spouští. Další postup je popsán v kapitole 7.6.

Při použití módu „GLOBAL DB" můžeme v konfiguračním souboru nastavit pravidla tvorby hesel (délka, možné znaky) platná pro všechny uživatele, v módu „USER DB" si pravidla nastavují uživatelé přímo programem *otpasswd* parametrem *--config*. Dále můžeme zakázat uživatelům generovat či tisknout hesla, nastavit možnosti opakování a podobně.

5.5.Generování hesel

Uživatel zprovozní přihlašování pomocí jednorázových hesel příkazem *otpasswd --key.* (obr. 7.1.) Při tomto kroku je vytvořen uživatelův klíč a také je vygenerována první karta s hesly, kterou by si uživatel měl vytisknout. Po vyčerpání všech jednorázových hesel z karty si musí vygenerovat a vytisknout kartu novou (Po použití posledního hesla z karty je zobrazeno upozornění). Generovat nové karty je možné buď v ASCII přes parametr *--text* nebo jako LaTeXový soubor přes parametr *--latex*, ve kterém je 6 karet na stránce A4.



Obr. 7.1. Výsledek příkazu *otpasswd –key* **Zdroj: Autor**

Nejjednodušší metoda vytištění hesel je převést vygenerovaný LaTeXový soubor do PDF a ten vytisknout. K tomu poslouží program *pdflatex* z balíčku *texlive-latex-base*.

```
root@ubuntu:~#  otpasswd --latex next > tmp.latex
root@ubuntu:~#  pdflatex tmp.latex
root@ubuntu:~#  lp tmp.pdf

# Důležité je po vytištění odstranit všechny
# dočasné soubory

root@ubuntu:~#  rm tmp.latex tmp.pdf
```

[Fortuna, 2010]

ubuntu [2]

	A	B	C	D	E	F	G
1:	sN3T	6Er2	C%H7	a4Df	HM!6	ME8i	C5vj
2:	j%sx	@P#7	43=L	uVxa	5exj	nM@K	5GsJ
3:	P8@u	!C5T	56tv	q@Kc	o6dF	UMPT	k?Ea
4:	zqJf	#G?J	RAZg	7uNV	WaTh	NMJa	JbZg
5:	%yn%	Gfui	%c=w	T2xk	Si%K	x6K4	f%zT
6:	yTsa	:kxD	Y6gK	Ey8J	B4Y+	jDNk	rxmg
7:	E=AU	BSuu	Nmta	YpRJ	mp!U	eAp6	R=j%
8:	r#2=	ThtS	DR6G	DfNg	5TGK	R3=e	Kosk
9:	:mdG	jBCn	=Px:	PECU	24W:	e:YS	3eJ%
10:	gAkB	YMai	Fno#	5:@R	K#Y5	Fo8%	A9Y2

Obr. 7.2. Výřez jedné karty z latexového souboru. Zdroj:Autor

5.6.Nastavení zasílání jednorázových hesel přes SMS zprávy

Po nastavení způsobu autentizace a cesty ke skriptu v konfiguračního souboru musíme vytvořit samotný skript, který se postará o zaslání SMS zprávy. Žádný ukázkový skript není k dispozici a musíme ho tedy napsat sami v závislosti na požadovaném řešení.

Nejjednodušší cestou, jak poslat z internetu SMS zprávu na mobilní telefon, je zaslání emailu na speciální adresu. Počet znaků může být značně omezen, to nám však v případě zasílání jednorázových hesel nevadí.

- *Operátor O_2* *tel.cislo@sms.cz.o2.com*
- *Operátor Vodafone* *uziv.jmeno@vodafonemail.cz (po aktivaci služby)*
- *Operátor T-Mobile* *uziv.jmeno@t-email.cz (po aktivaci služby)*

Velkou nevýhodou této metody je to, že nemáme zaručené spolehlivé doručení SMS zprávy. SMS brána může být přetížená a pokud SMS přijde pozdě nebo vůbec, nemůžeme se do systému přihlásit. Mnohem lepší by bylo využít nějakou komerční SMS bránu nebo přímo mobilní telefon připojený k serveru. Pro naše účely však toto řešení postačuje, SMS do sítě O_2 běžně chodí do několika vteřin.

Při autentizaci se skript volá se dvěma parametry – první parametr je jednorázové heslo a druhý parametr je kontakt na uživatele. Při využití výše uvedené metody stačí předat parametry programu mail, který zprávu odešle přes SMTP server (Ubuntu používá standardně POSTFIX)

```
root@ubuntu:~# vi /etc/otpasswd/otpasswd_oob.sh

#!/bin/bash

echo „Vase jednorazove heslo je: $2“ | /usr/bin/mail –s “OTP” $1
```

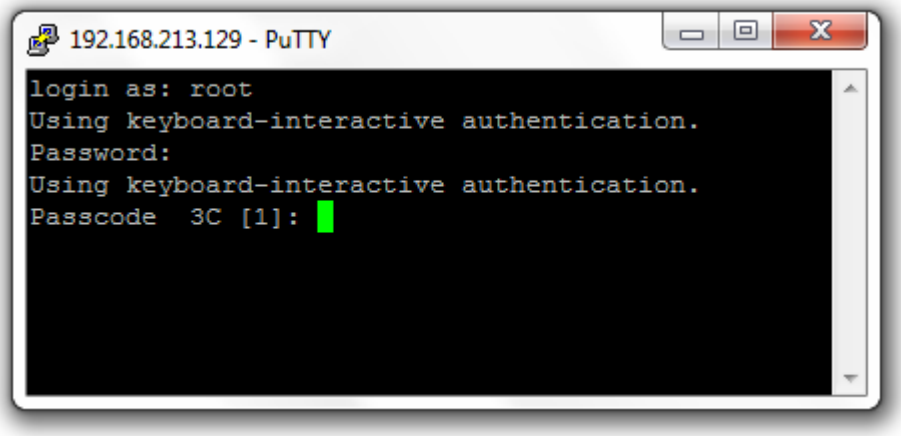
Nastavení kontaktu si každý uživatel provede přímo programem otpasswd, případně editací souboru .otpasswd v domovském adresáři.

```
root@ubuntu:~# otpasswd --config contact=728123456@sms.cz.o2.com
```

Pokud uživatel uvede klasickou e-mailovou adresu, budou mu jednorázová hesla chodit e-mailem.

5.7.Ukázka autentizace

První krok - zadání uživatelského jména a klasického hesla

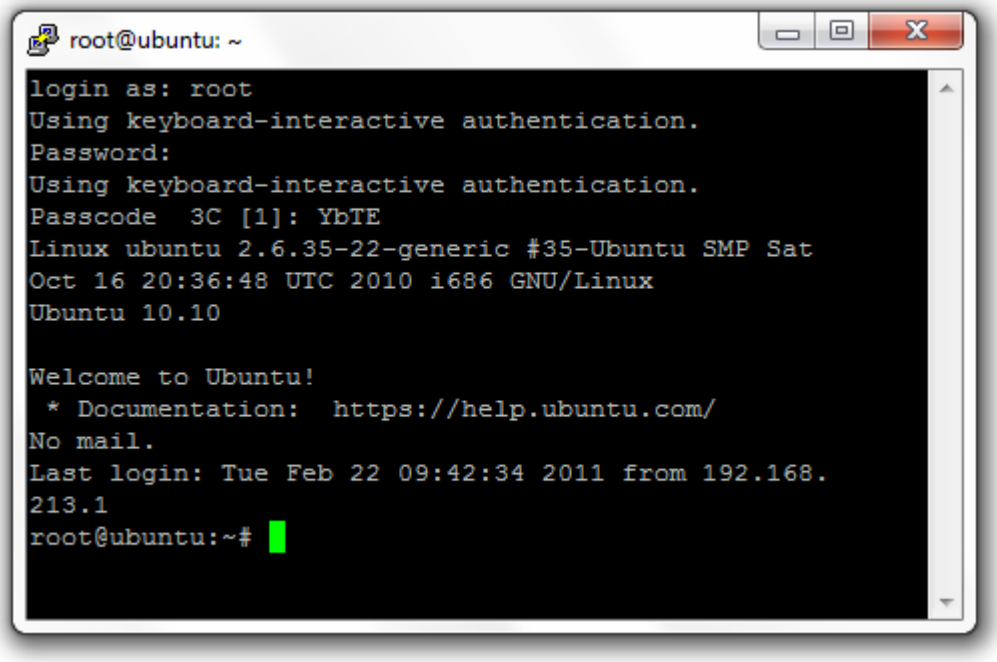


Obr. 7.3. Zdroj: Autor

Druhý krok - uživatel obdrží SMS

Obr. 7.4. Zdroj: Autor

Třetí krok - zadání jednorázového hesla z SMS a úspěšná autentizace



Obr. 7.5. **Zdroj: Autor**

6. Druhý příklad – autentizace k webové aplikaci

6.1.MultiOTP

Druhým příkladem je využití jednorázových hesel při přihlašování k webové aplikaci. K tomu využijeme MultiOTP[1], což je open-source PHP třída vyvíjená švýcarskou společností SysCO. Podporuje algoritmy OATH HOTP, OATH TOTP a MOTP[2], je tedy možné použít veškeré hardwarové i softwarové autentizační kalkulátory, které podporují některý z výše uvedených algoritmů. MultiOTP je možné buď integrovat s RADIUS serverem, nebo přímo použít PHP třídu bez externího autentizačního serveru – to si ukážeme na příkladu.

Součástí třídy je i rozhraní pro příkazový řádek a binární soubor pro Windows obsahující MultiOTP spolu s PHP interpretem, pomocí něhož můžeme MultiOTP ovládat:

· Vytvoříme uživatele test, který používá HOTP, zadáme seed, pin, počet znaků, pozici události

multiotp -create test HOTP
3132333435363738393031323334353637383930 1234 6 0

· Zadáme první jednorázové heslo

multiotp -debug test 755224
0 OK: Token accepted

· Stejné heslo už není platné

multiotp -debug test 755224
99 ERROR: Authentication failed

[Liechti,2010]

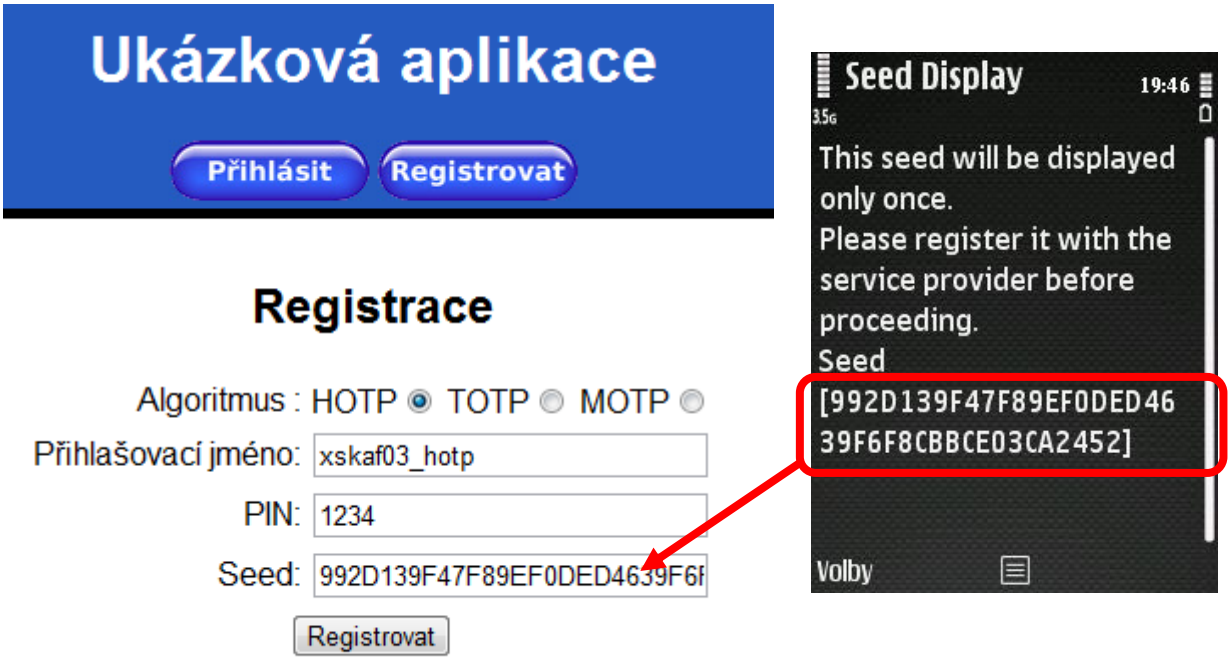
[1]MultiOTP http://developer.sysco.ch/multiotp/
[2] MobileOTP, http://motp.sourceforge.net/. Algoritmus založený na časové synchronizaci

Jako autentizační kalkulátory byly vyzkoušeny dvě mobilní javové aplikace: Mobile-OTP Java MIDIet[2] a DS3 OATH TOKEN[1] podporující HOTP. Existuje spousta kalkulátorů pro iPhone a Android (Google Authenticator, OATH Token, Android Token, iOTP Token)[2], ale javových či symbianových aplikací bohužel moc není. Proto byl pro vyzkoušení TOTP použit Android Token[3] v emulátoru.

Využití PHP třídy MultiOTP si ukážeme na jednoduchých webových stránkách na adrese http://xskaf03.php5.cz. Nejprve si popíšeme jejich funkcionalitu a pak se vrátíme k implementaci.

6.2.Registrace

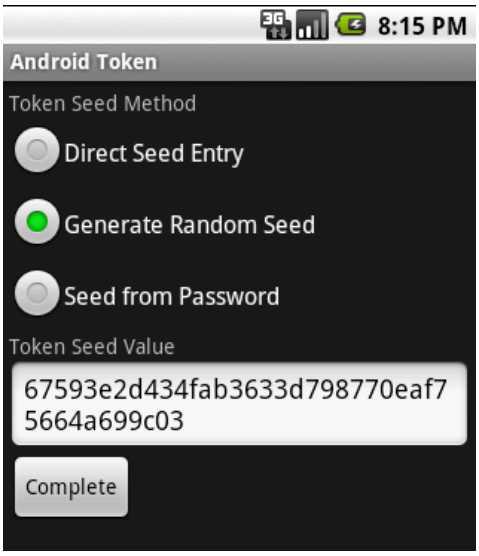
Uživatel zadá přihlašovací jméno, čtyřmístný PIN a v závislosti na použitém autentizačním kalkulátoru vybere algoritmus a zadá seed, který je obvykle zobrazen při prvním spuštění softwarového kalkulátoru nebo dodán spolu s hardwarovým kalkulátorem. Z hlediska bezpečnosti je ale toto řešení nevhodné. Registrace uživatelů by se měla provádět přímo na serveru nebo přes šifrovaný protokol, protože pokud někdo odposlechne zadané údaje, nemá další použití jednorázových hesel smysl. Toto řešení bylo použito pro účely jednoduché ukázky.



Obr. 8.1. **Zdroj: Autor**

[1] http://www.ds3global.com
[2] http://www.rcdevs.com/tokens/
[3] http://code.google.com/p/androidtoken/

Obdobně je seed zobrazen při prvním spuštění Android Tokenu i Mobile-OTP.

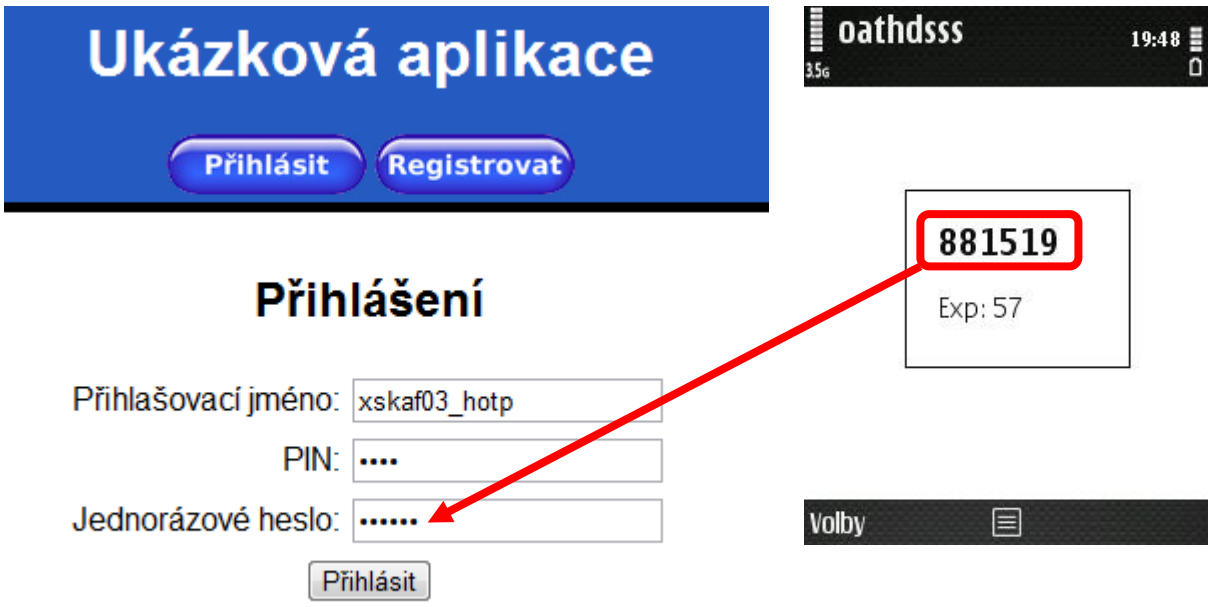


Obr. 8.2. Android Token Zdroj: Autor

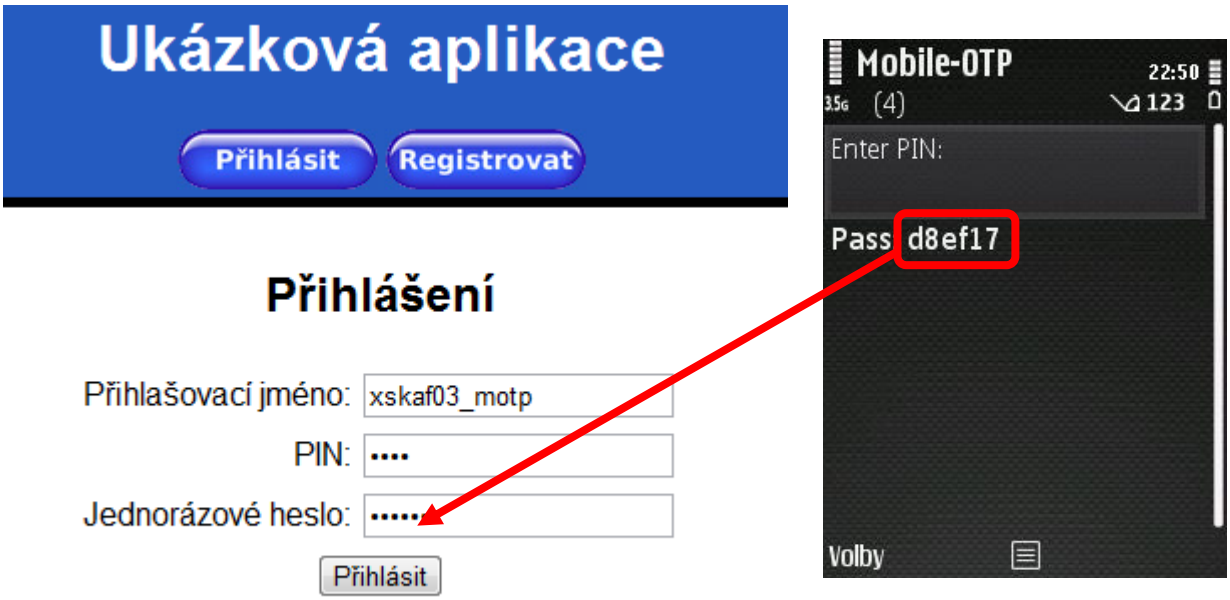
Obr. 8.3. Mobile-OTP Zdroj: Autor

6.3. Přihlášení

Uživatel zadá přihlašovací jméno, PIN a jednorázové heslo vyčtené z kalkulátoru. Po úspěšné autentizaci je spuštěna session a zobrazeno jméno přihlášeného uživatele.

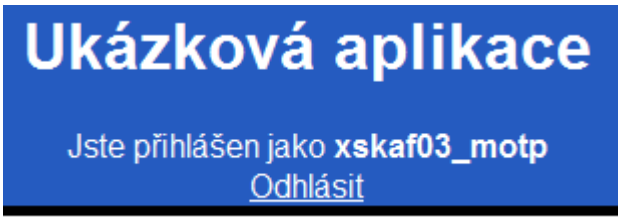


Obr. 8.4. Autentizace uživatele xskaf03_hotp Zdroj: Autor



Obr. 8.4. Obdobná autentizace uživatele xskaf03_motp **Zdroj: Autor**

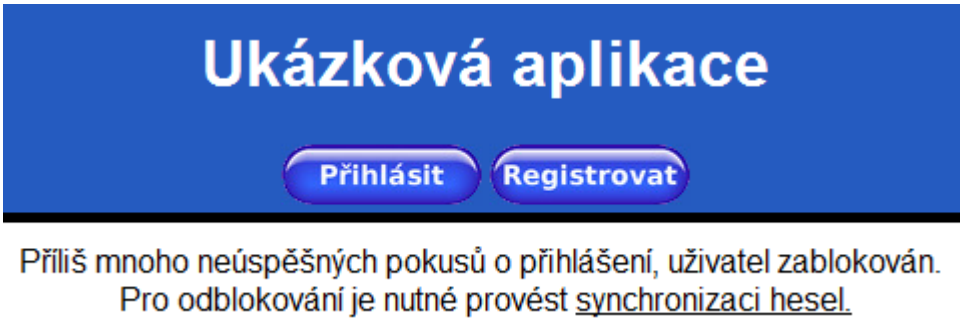
Po úspěšné autentizaci je spuštěna session a zobrazeno jméno přihlášeného uživatele.



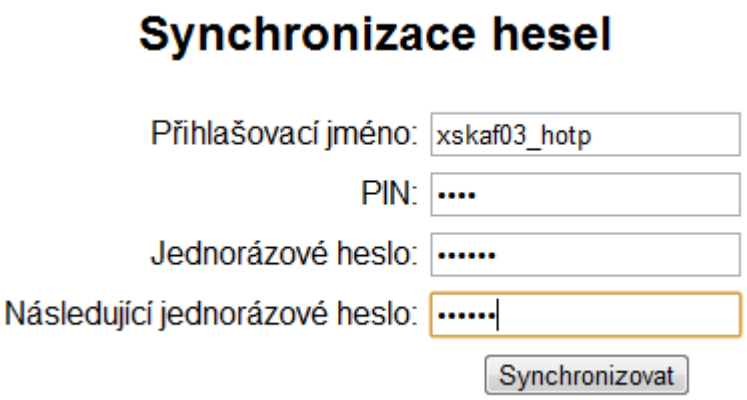
Obr. 8.5. **Zdroj: Autor**

6.4.Synchronizace hesel

Po několika neúspěšných pokusech přihlášení (standardně 6) je uživatel zablokován a pro odblokování musí zadat dvě následující jednorázová hesla.



Obr. 8.6. **Zdroj: Autor**

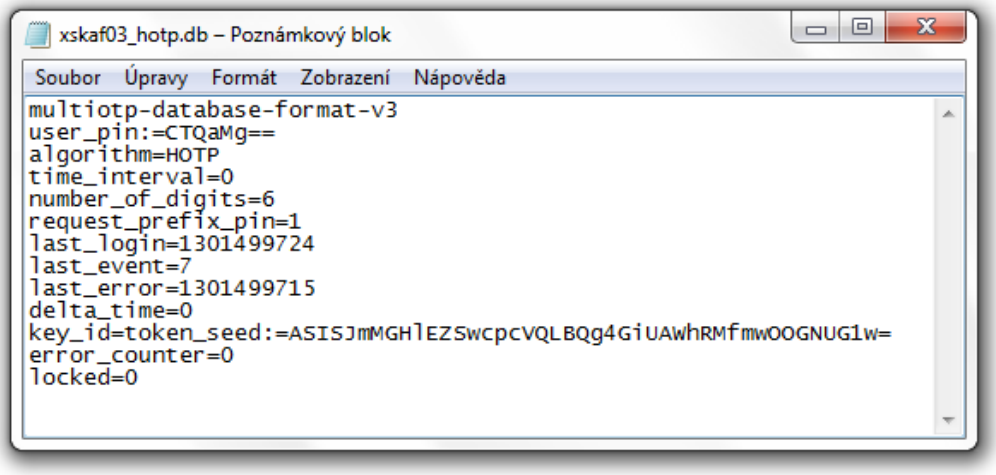


Obr. 8.7. **Zdroj: Autor**

6.5.Implementace

Nyní se můžeme podívat, jak to vlastně funguje. O vše se stará soubor *multiotp.class.php* z balíčku *multiotp.zip.*[1]. Žádný další soubor není potřeba, protože na webserveru nepotřebujeme rozhraní pro příkazový řádek.

Nejprve musíme vědět, že veškerá uživatelská data MultiOTP se ukládají v souborech *[jmeno_uzivatele].db*, standardně v adresáři */users/*. Je zde uložen typ algoritmu, hash PINu a seedu, počet událostí, časové intervaly, počet neúspěšných přihlášení kvůli zablokování apod. (obr. 8.8.)



Obr. 8.8. Soubor s nastavením pro uživatele xskaf03_hotp **Zdroj: Autor**

[1] http://developer.sysco.ch/multiotp/

Když toto víme, můžeme se podívat přímo na kousky kódu:

6.5.1. Registrace

```
require_once('multiotp.class.php');                    //vložíme PHP třídu MultiOTP

$multiotp = new Multiotp();                            //vytvoříme instanci MultiOTP

$multiotp -> SetUsersFolder('../data/users/');         //nastavíme cestu k adresáři
                                                       s uživatelskými daty mimo složku
                                                       public_html, aby nebyly zvenku přístupné

if ($multiotp->ReadUserData($_POST['jmeno'])) echo "Uživatel již existuje";
else {                                                 //uživatel nesmí existovat

$multiotp->SetUser($_POST['jmeno']);                   //zvolíme jméno

 $multiotp->SetUserAlgorithm($_POST['alg']);           //zvolíme algoritmus

$multiotp->SetUserTokenSeed($_POST['seed']);           //zvolíme seed

$multiotp->SetUserPin($_POST['pin']);                  //zvolíme pin

$multiotp->SetUserPrefixPin('1');                      //heslo se bude testovat společně s PINem

$multiotp->SetUserTokenNumberOfDigits('6');            //nastavíme délku jednorázového hesla
                                                         na 6 znaků (výchozí hodnota)
```

Dále je potřeba nastavit některé hodnoty v závislosti na použitém algoritmu (výchozí hodnoty podle rozhraní příkazové řádky)

```
switch ($_POST['alg']) {
case "MOTP":  $multiotp->SetUserTokenTimeInterval('10'); break;   //MOTP interval 10s

case "TOTP":   $multiotp->SetUserTokenTimeInterval('30'); break;  //TOTP interval 30s

case "HOTP":  $multiotp->SetUserTokenLastEvent(-1); break;        //HOTP počáteční událost
 }

 if ($multiotp->WriteUserData()) echo "Uživatel byl úspěšně vytvořen";  //zapíšeme všechna
                                                                          data
else  echo "Nastala chyba při vytváření uživatele";
```

6.5.2. Přihlášení

```
require_once('multiotp.class.php');                    //vložíme PHP třídu MultiOTP

$multiotp = new Multiotp();                            //vytvoříme instanci MultiOTP

$multiotp -> SetUsersFolder('../data/users/');    //nastavíme cestu k adresáři s uživ. daty

if (!$multiotp->ReadUserData($_POST['jmeno'])) echo "Uživatel neexistuje";
else {                                                 //uživatel musí existovat

$multiotp -> SetUser($_POST['jmeno']);                 //zvolíme uživatele

$token = $_POST['pin'].$_POST['heslo']          //spojíme PIN a jednorázové heslo ke kontrole

$hodnota = $multiotp->CheckToken($token);      //zde konečně otestujeme jednorázové
                                                       heslo  a obdržíme návratovou hodnotu
switch ($hodnota) {
case '99': echo "Nesprávné heslo nebo PIN"; break;   //hodnota 99 značí neúspěšnou
                                                       autentizaci

case '24': echo "Uživatel zablokován“; break;          //hodnota 24 značí, že je uživatel
                                                       zablokován

case '0': {                                            //hodnota 0 značí úspěšnou autentizaci
        session_start();
        $_SESSION["jmeno"] = $_POST["jmeno"];
        ...
        ...             // kód, který se vykoná po úspěšné autentizaci uživatele
        ...
break;}
}}
```

6.5.3. Synchronizace hesel

Při synchronizaci hesel zadáme metodě CheckToken dva parametry – jednorázové heslo a následující jednorázové heslo.

$multiotp->CheckToken($token1,$token2));

Pokud byla synchronizace úspěšně provedena, je návratová hodnota 14, při chybě obdržíme 27.

7. Závěr

V této bakalářské práci jsem se pokusil vysvětlit důvody vedoucí k použití jednorázových hesel, popsat různé možnosti jejich tvorby a distribuce k uživateli a uvést příklady jednotlivých řešení. V praktické části jsem nastínil použití open-source systému OTPasswd pro autentizaci k serveru přes SSH s využitím dvoukanálové autentizace a PHP třídy MultiOTP pro autentizaci k webové aplikaci.

Jednorázová hesla jsou vhodným řešením, pokud chceme zamezit nebezpečí odposlechnutí hesla při zadávání uživatelem. Hodí se tedy v případě častého přihlašování z nedůvěryhodných veřejných počítačů, kde může být nainstalován keylogger, který by klasické heslo odchytil. Použití „chytřejšího“ keyloggeru, který odcizí i jednorázové heslo je dle mého názoru už příliš specifický útok.

Stejně tak mají jednorázová hesla význam ve vícefaktorové autorizaci v kombinaci s dalšími metodami zabezpečení. Dvoukanálová autentizace pomocí SMS zpráv je oblíbená a podle mě i účinná, ale hrozí zde nebezpečí, že si uživatel do telefonu nainstaluje mallware, který SMS zprávy přepošle útočníkovi.

Myslím si, že bezpečnostní riziko může nastat, pokud uživatel používá jedno zařízení (smartphone) k přihlašování i přijímání SMS zpráv s jednorázovým heslem. Pokud přidáme možnost odposlechnutí nešifrované SMS zprávy, vidím hardwarový či softwarový kalkulátor jako účinnější řešení.

Musíme však mít na paměti, že jednorázová hesla neřeší útok „člověka uprostřed“ a také nás neochrání před různými metodami sociálního inženýrství, kdy útočník nějak uživatele donutí, aby mu jednorázové heslo nebo informace k němu vedoucí sdělil.

8. Seznam použitých zdrojů

BACHFELD, Daniel. Einmal und nie wieder : Sichere Logins mit Einmalpasswörtern. *Heise Security* [online]. 2007-04-02, [cit. 2011-02-12]. Dostupný z WWW: <http://www.heise.de/security/artikel/Einmalpasswoerter-fuer-den-Heimgebrauch-270884.html>.

BIRYUKOV, Alex; LANO, Joseph; PRENEEL, Bart. *Cryptanalysis of the Alleged SecurID Hash Function* [online]. 2003 [cit. 2011-04-01]. Dostupné z WWW: <http://www.springerlink.com/content/whk1f5tnalm440g4/fulltext.pdf>.

BRILL, Jake. *Facebook Blog* [online]. 2010-10-12 [cit. 2011-02-12]. More ways to stay secure. Dostupné z WWW: <http://blog.facebook.com/blog.php?post=436800707130>.

DOSTÁLEK, Libor. *Velký průvodce protokoly TCP/IP, Bezpečnost*. První vydání. Praha : Computer Press, 2001. 566 s. 4.2. Jednorázová hesla. s. 124-132. Dostupné z WWW: <http://www.cpress.cz/knihy/tcp-ip-bezp/Tutorial/Tutorial.htm>. ISBN 80-7226-513-X.

DRIMER, Saar; MURDOCH, Steven J.; ANDERSON, Ross. *Optimised to Fail : Card Readers for Online Banking*[online]. a. Computer Laboratory, University of Cambridge, UK : A, 2009 [cit. 2011-04-04]. Dostupné z WWW: <http://www.cl.cam.ac.uk/~sjm217/papers/fc09optimised.pdf>.

FORTUNA, Thomasz. *The One-Time Password Authenticaiton System* [online]. 2010 [cit. 2011-04-01]. OTPasswd. Dostupné z WWW: <http://otpasswd.thera.be>.

GOODIN, Dan. ZeuS trojan attacks bank's 2-factor authentication. The Register [online]. 22.2.2011, 0, [cit. 2011-04-29]. Dostupný z WWW: <http://www.theregister.co.uk/2011/02/22/zeus_2_factor_authentication_attack/>.

GRIFFIN, Dan. Safer Authentication with a One-Time Password Solution. *MSDN Magazine* [online]. Květen 2008, [cit. 2011-02-12]. Dostupný z WWW: <http://msdn.microsoft.com/en-us/magazine/cc507635.aspx>.

HALLER, Neil. The S/KEY One-Time Password System. *Request for Comments* 1760 [online]. Únor 1995, [cit. 2011-02-11]. Dostupný z WWW: <http://tools.ietf.org/html/rfc1760>.

HALLER, Neil. A One-Time Password System. *Request for Comments* 2289 [online]. Únor 1998, [cit. 2011-02-13]. Dostupný z WWW: <http://tools.ietf.org/html/rfc2289>.

HONTANÓN, J. Ramón. *Linux, praktická bezpečnost*. Praha : Grada, 2003.440s.s.400. ISBN 80-247-0652-0.

KLÍMA, Vlastimil. Hašovací funkce MD5 a další prolomeny!. *Root.cz* [online]. 25. 8. 2004, [cit. 2011-02-13]. Dostupný z WWW: <http://www.root.cz/clanky/hasovaci-funkce-md5-a-dalsi-prolomeny/>.

KMEŤ, Alexander. Trendy ve světě internetového bankovnictví. Bankovnictví iHNed [online]. 20. 10. 2006, [cit. 2011-04-26]. Dostupný z WWW: <http://bankovnictvi.ihned.cz/c3-19571060-900000_d1-trendy-ve-svete-internetoveho-bankovnictvi>.

KNĚŽEK, Ivo. Řešení silné autentizace uživatele. *Systemonline* [online]. 2001, 7-8/2001, [cit. 2011-04-04]. Dostupný z WWW: <http://www.systemonline.cz/clanky/reseni-silne-autentizace-uzivatele.htm>.

KUHN, Markus. OTPW – A one-time password login package. *University of Cambridge* [online]. 2003 [cit. 2011-02-15]. Dostupné z WWW: <http://www.cl.cam.ac.uk/~mgk25/otpw.html>.

KRAWCZYK, H., et al. HMAC: Keyed-Hashing for Message Authentication. RFC 2104 [online]. 1997), [cit. 2011-04-26]. Dostupný z WWW: <http://tools.ietf.org/html/rfc2104>.

LAMPORT, Leslie. Password Authentication with Insecure Communication. *Communications of the ACM 24.11* [online]. Listopad 1981, [cit. 2011-02-11]. Dostupný z WWW: <http://research.microsoft.com/en-us/um/people/lamport/pubs/password.pdf>.

LIECHTI, André. *MultiOTP : Free Strong Authentication Presentation and Deployment* [online]. 3.0.4., 15. září 2010 [cit. 2011-03-30]. Dostupné z WWW: <http://www.multiotp.net/multiOTP_Free_Strong_Authentication_Presentation.pdf>.

LINDELL, Andrew Y. Time versus Event Based One-Time Passwords. SafeNet [online]. 2007, [cit. 2011-04-04]. Dostupný z WWW: <http://www3.safenet-inc.com/blog/pdf/Time_vs_Event_Based_OTP.pdf>.

MCDONALD, L. D.; ATKINSON, J. R. One Time Passwords In Everything (OPIE): Experiences with Building and Using Stronger Authentication. *USENIX* [online]. 1995, [cit. 2011-02-12]. Dostupný z WWW: <http://www.usenix.org/publications/library/proceedings/security95/full_papers/mcdonald.pdf>.

MENEZES, J. A.; OORSCHOT, C. P; VANSTONE, A. S.. *Handbook of Applied Cryptography*. 2001. 816 s. 10.2.5. One-time passwords. s. 395-397. Dostupné z WWW: <http://www.cacr.math.uwaterloo.ca/hac/>. ISBN 0-8493-8523-7.

NEEDHAM, Sarrah. Onfident Technologies Delivers Image-Based, Multifactor Authentication to Strengthen Passwords on Public-Facing Websites. *Marketwire* [online]. 28. října 2010, [cit. 2011-02-14]. Dostupný z WWW: <http://www.marketwire.com/press-release/Confident-Technologies-Delivers-Image-Based-Multifactor-Authentication-Strengthen-Passwords-1342854.htm>.

NYSTROEM, Magnus. The EAP Protected One-Time Password Protocol (EAP-POTP). *Request for Comments* [online]. 2007, 4793, [cit. 2011-04-26]. Dostupný z WWW: <http://tools.ietf.org/html//rfc4793>.

ORTIZ, Sixto Jr. One-Time Password Technology. *PROCESSOR* [online]. 13. dubna 2007, 29, [cit. 2011-02-12]. Dostupný z WWW: <http://www.processor.com/editorial/article.asp?Article=articles/p2915/30p15/30p15.asp&GUID=042B642DCFAD41C88FD517A5D84F2E70>.

PERRIN, Chad. How one-time passwords fit in with multifactor authentication. *TechRepublic* [online]. 2011-01-04, [cit. 2011-02-11]. Dostupný z WWW: <http://www.techrepublic.com/blog/security/how-one-time-passwords-fit-in-with-multifactor-authentication/4872>.

PŘIBYL, Tomáš. Biometrika jako na houpačce . ICTSecurity [online]. 19.1.2009, [cit. 2011-04-04]. Dostupný z WWW: <http://www.ictsecurity.cz/09/08/2-autentizace/biometrika-jako-na-houpacce.html>.

ŘÁDA, Matěj. Je mobilní telefon ideálním nástrojem pro ovládání účtu? .Bankovnictví Ihned [online]. 20.4.2004, [cit. 2011-04-04]. Dostupný z WWW: <http://bankovnictvi.ihned.cz/c4-10066530-14246910-900000_d-je-mobilni-telefon-idealnim-nastrojem-pro-ovladani-uctu>.

RAIHI, M., et al. HOTP: An HMAC-Based One-Time Password Algorithm. RFC 4226 [online]. 2005, 0, [cit. 2011-04-26]. Dostupný z WWW: <http://tools.ietf.org/html/rfc4226>.

RAIHI, M., et al. TOTP: Time-based One-time Password Algorithm. RFC draft [online]. 2011, 08, [cit. 2011-04-27]. Dostupný z WWW: <http://tools.ietf.org/html/draft-mraihi-totp-timebased-08>.

RAIHI, M., et al. OCRA: OATH Challenge-Response Algorithms. RFC draft [online]. 2011, 14, [cit. 2011-04-27]. Dostupný z WWW: <http://tools.ietf.org/html/draft-mraihi-mutual-oath-hotp-variants-14>.

RUBIN, Aviel D. Independent One-Time Passwords. USENIX UNIX Security Symposium [online]. 1995, [cit. 2011-04-27]. Dostupný z WWW: <http://usenix.org/publications/library/proceedings/security95/full_papers/rubin.pdf>.

SHA-1 & HMAC OTP Frequently Asked Questions. OATH [online]. [cit. 2011-04-29]. Dostupný z WWW: <http://www.openauthentication.org/pdfs/Attacks%20on%20SHA-1%20FAQ.pdf>.

SHAH, Nishit. Advanced sign-in security for your Google account. Google Blog [online]. 10.2.2011, [cit. 2011-04-28]. Dostupný z WWW: <http://googleblog.blogspot.com/2011/02/advanced-sign-in-security-for-your.html>.

STEVENS, Tim. RSA hacked, data exposed that could 'reduce the effectiveness' of SecurID tokens. Engadget [online]. 18.3.2011, [cit. 2011-04-04]. Dostupný z WWW: <http://www.engadget.com/2011/03/18/rsa-hacked-data-exposed-that-could-reduce-the-effectiveness-o/>.

Trusteer. Anti-Keylogger Myths. Trusteer [online]. 2007, [cit. 2011-04-29]. Dostupný z WWW: <http://www.trusteer.com/sites/default/files/Anti_Keylogger_Myths.pdf>.

Visa CodeSure gets commercial green light. *Finextra* [online]. 2. června 2010, 0, [cit. 2011-02-13]. Dostupný z WWW: <http://www.finextra.com/news/fullstory.aspx?newsitemid=21447>.

VRÁNA, Jakub. Bezpečné přihlašování uživatelů. *ROOT* [online]. 12.4.2006, [cit. 2011-04-26]. Dostupný z WWW: <http://www.root.cz/clanky/bezpecne-prihlasovani-uzivatelu/>.

Yubico. YubiKey Security Evaluation . YubiKey Security Evaluation [online]. 10.2009, [cit. 2011-04-29]. Dostupný z WWW: <http://static.yubico.com/var/uploads/pdfs/Security_Evaluation_2009-09-09.pdf>.

ZANDL, Patrick. Šifrovací algoritmus A5.1 prý praskl - konec bezpečnosti GSM?.Mobil.cz [online]. 9.9.1999, [cit. 2011-04-04]. Dostupný z WWW: <http://mobil.idnes.cz/mob_tech.asp?r=mob_tech&c=A991208_0002145_mob_tech>.

9. Terminologický slovníček

Termín	Význam	Zdroj
AES	Advanced Encryption Standard, symetrická bloková šifra	http://www.ietf.org/rfc/rfc3268.txt
Autentizace	Ověření pravosti	http://prirucka.ujc.cas.cz
Bezpečnostní token	Zařízení či software sloužící k autentizaci	Autor
HASH	Výsledek funkce, která řetězec znaků o libovolné délce převede na řetězec znaků s pevnou délkou.	http://www.abclinuxu.cz/slovnik
HOTP	HMAC-based one time password	http://tools.ietf.org/html/rfc4226
Keylogger	Software, který tajně snímá stisky jednotlivých kláves	Autor
MD5	Message-Digest algorithm, hashovací funkce	http://tools.ietf.org/html/rfc1321
OATH	Initiative For Open Authentication	http://www.openauthentication.org/
OTP	One-time password, jednorázové heslo	Autor
Proprietární	Software, který není svobodný (nejsou k němu volně k dispozici zdrojové kódy)	http://www.gnu.org/philosophy
RFC	Request for Comments, seznam standardů	http://www.abclinuxu.cz/slovnik/
Seed	Náhodné číslo, které se používá při inicializaci generátoru pseudonáhodných čísel	http://www.abclinuxu.cz/slovnik/
SHA-1	Secure Hash Algorithm, hashovací funkce	http://www.faqs.org/rfcs/rfc3174.html
TLS	Transport Layer Security, protokol sloužící k šifrování komunikace na internetu	http://www.ietf.org/rfc/rfc2246.txt

10. Přílohy

10.1. Soubor registrace.php

```
  <!DOCTYPE html
    PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
  <html>
     <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Ukázková aplikace - Registrace</title>
        <link rel="stylesheet" type="text/css" href="styl.css">
     </head>
     <body>
        <div class="horni">
        <h1>Ukázková aplikace</h1>
        <? include "stav.php"; ?>
        </div>
        <div class="telo">
  <?

    function jeJmeno ($jmeno) {
         return ereg("^[a-zA-Z0-9]+[_a-zA-Z0-9-]+$", $jmeno);
    }

    function jePIN ($pin) {
         return ereg("^[0-9][0-9][0-9][0-9]$", $pin);
    }

  if (isset($_POST['odesli'])) {
    if (!$_POST['alg']) {echo 'Nebyl vybrán algoritmus';}
    elseif (!$_POST['jmeno']) {echo 'Nebylo zadáno přihlašovací jméno';}
    elseif (!jeJmeno($_POST['jmeno'])) {echo 'Přihlašovací jméno obsahuje ne-
povolené znaky';}
    elseif (!$_POST['pin']) {echo 'Nebyl zadán PIN';}
    elseif (!jePIN($_POST['pin'])) {echo 'Pin musí být čtyřčíselný';}
    elseif (!$_POST['seed']) {echo 'Nebyl zadáno seed';}

    else  {

      require_once('multiotp.class.php');
      $multiotp = new Multiotp();
      $multiotp -> SetUsersFolder('../data/users/');

      if ($multiotp->ReadUserData($_POST['jmeno'],'TRUE')) echo "Uživatel již
existuje";
      else {

      $multiotp->SetUser($_POST['jmeno']);
      $multiotp->SetUserAlgorithm($_POST['alg']);
      $multiotp->SetUserTokenSeed($_POST['seed']);
      $multiotp->SetUserPin($_POST['pin']);
      $multiotp->SetUserPrefixPin('1');
      $multiotp->SetUserTokenNumberOfDigits('6');

        switch ($_POST['alg']) {

          case "MOTP": $multiotp->SetUserTokenTimeInterval('10');break;

          case "TOTP": $multiotp->SetUserTokenTimeInterval('30'); break;

          case "HOTP": $multiotp->SetUserTokenLastEvent(-1); break;
       }
      if ($multiotp->WriteUserData()) echo "Uživatel byl úspěšně vytvořen";
```

```
        else  echo "Nastala chyba při vytváření uživatele";
        }
      }
    }
    ?>

    <h2>Registrace</h2>
      <form method="post" action="registrace.php">
      <table class="form">
        <tr>
          <td>Algoritmus :</td>   <td>
          HOTP<input type="radio" name="alg" value="HOTP">
          TOTP<input type="radio" name="alg" value="TOTP">
          MOTP<input type="radio" name="alg" value="MOTP"></td>
        </tr>
        <tr>
          <td>Přihlašovací jméno:</td>
          <td><input name="jmeno" size="27"></td>
        </tr>
            <tr>
          <td>PIN:</td>
          <td><input name="pin" size="27"</td>
        </tr>
        <tr>
          <td>Seed:</td>
          <td><input name="seed" size="27"></td>
        </tr>
        <tr>
          <td align="center" colspan="2"><input type="Submit" name="odesli" va-
lue="Registrovat"></td>
        </tr>
      </table>
      </form>
    </div>
    <div class="pata">2011 Filip Škarda, xskaf03@vse.cz</div>
    </body>
    </html>
```

10.2. Soubor prihlaseni.php

```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Ukázková aplikace - přihlášení</title>
      <link rel="stylesheet" type="text/css" href="styl.css">
   </head>
   <body>
      <div class="horni">
      <h1>Ukázková aplikace</h1>
      <? include "stav.php"; ?>
      </div>
      <div class="telo">
<?

if (isset($_POST['odesli'])) {
if (!$_POST['jmeno']) {echo 'Nebylo zadáno přihlašovací jméno';}
elseif (!$_POST['pin']) {echo 'Nebyl zadán PIN';}
elseif (!$_POST['heslo']) {echo 'Nebylo zadáno jednorázové heslo';}
else  {

  require_once('multiotp.class.php');
```

```
  $multiotp = new Multiotp();
  $multiotp -> SetUsersFolder('../data/users/');

  if (!$multiotp->ReadUserData($_POST['jmeno'])) echo "Uživatel neexistuje";
  else {

    $multiotp -> SetUser($_POST['jmeno']);
    $token = $_POST['pin'].$_POST['heslo'];

    switch ($multiotp->CheckToken($token)) {

    case '99': echo "Nesprávné heslo"; break;

    case '24': echo "Příliš mnoho neúspěšných pokusů o přihlášení, uživatel
zablokován. <br>
    Pro odblokování je nutné provést <a href='synchronizace.php'>synchronizaci
hesel.</a>";
    break;

    case '0': {
      session_start();
      $_SESSION["jmeno"] = $_POST["jmeno"];
      header( "refresh:0;url=index.php" );
    break; }

    default:
      echo "Nastala chyba číslo"; echo $multiotp->CheckToken($token);break;
    }
    }
  }
}
?>

<h2>Přihlášení</h2>
<form method="post" action="prihlaseni.php">
<table class="form">
 <tr>
      <td>Přihlašovací jméno:</td>
      <td><input name="jmeno"></td></tr><tr>
      <td>PIN:</td>
      <td><input name="pin" type ="password"></td>  </tr><tr>
      <td>Jednorázové heslo:</td>
      <td><input name="heslo" type ="password"></td>
      </tr><tr><td align="center" colspan="2">
      <input type="Submit" name="odesli" value="Přihlásit">
 </td>
</tr>
</table>
</form>
</div>
<div class="pata">2011 Filip Škarda, xskaf03@vse.cz</div>
</body>
</html>
```

10.3. Soubor synchronizace.php

```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Ukázková aplikace - Synchronizace hesel</title>
      <link rel="stylesheet" type="text/css" href="styl.css">
   </head>
```

```
   <body>
      <div class="horni">
      <h1>Ukázková aplikace</h1>
      <? include "stav.php"; ?>
      </div>
      <div class="telo">
<?
if (isset($_POST['odesli'])) {
  if (!$_POST['jmeno']) {echo 'Nebylo zadáno přihlašovací jméno';}
  elseif (!$_POST['pin']) {echo 'Nebyl zadán PIN';}
  elseif (!$_POST['heslo1']) {echo 'Nebylo zadáno jednorázové heslo';}
  elseif (!$_POST['heslo1']) {echo 'Nebylo zadáno druhé jednorázové heslo';}

  else {
    require_once('multiotp.class.php');
    $multiotp = new Multiotp();
    $multiotp -> SetUsersFolder('../data/users/');
    $multiotp -> SetUser($_POST['jmeno']);

    if (!$multiotp->ReadUserData($_POST['jmeno'])) echo "Uživatel neexistuje";
    else {

      $token1 = $_POST['pin'].$_POST['heslo1'];
      $token2 = $_POST['pin'].$_POST['heslo2'];

      switch ($multiotp->CheckToken($token1,$token2)) {

        case '14': echo "Synchronizace úspěšně provedena";break;
        case '27': echo "Chyba při synchronizaci";break;
        case '21': echo "Uživatel neexistuje";break;
        default : echo "Nastala chyba číslo"; echo $multiotp-
>CheckToken($token1,$token2);
      }
    }
  }
}
?>

<h2>Synchronizace hesel</h2>
  <form method="post" action="synchronizace.php">
  <table class="form">
    <tr>
      <td>Přihlašovací jméno:</td>
      <td><input name="jmeno" value=""></td>
    </tr><tr>
      <td>PIN:</td>
      <td><input name="pin" type = "password"></td>
    </tr>
    <tr>
      <td>Jednorázové heslo:</td>
      <td><input name="heslo1" type = "password"></td>
    </tr>
    <tr>
      <td>Následující jednorázové heslo:</td>
      <td><input name="heslo2" type = "password"></td>
    </tr>
    <tr>
      <td></td>
      <td align="center" colspan="2"><input type="Submit" name="odesli" va-
lue="Synchronizovat"></td>
    </tr>
  </table>
  </form>
</div>
<div class="pata">2011 Filip Škarda, xskaf03@vse.cz</div>
</body>
</html>
```