SZKOŁA GŁÓWNA HANDLOWA W WARSZAWIE

METODY ILOŚCIOWE W EKONOMII I SYSTEMY INFORMACYJNE

Piotr Kucharski

Nr albumu: 7431

Hasła w systemach komputerowych

Praca licencjacka

napisana pod kierunkiem naukowym

dr. Michała Golińskiego

w Katedrze Informatyki Gospodarczej

Warszawa 2007

Spis treści

1. Wstęp

Celem niniejszej pracy jest krytyczna analiza uwarunkowań dotyczących haseł w systemach komputerowych i prezentacja własnego programu do zmiany haseł, który ocenia siłę nowych haseł w oparciu o ich entropię. Analiza zostanie przeprowadzona na podstawie systemów sieci uczelnianej SGH.

W rozdziale 2 zarysowano ogólną tematykę bezpieczeństwa, której jednym z aspektów są hasła komputerowe, naszkicowano problemy związane z hasłami, a także przytoczono kilkanaście artykułów kodeksu karnego, które są powiązane z tematem haseł i ogólnie bezpieczeństwem informatycznym.

Podczas opracowywania programu do zmiany haseł wykorzystano teorię informacji przedstawioną w rozdziale 3, gdzie zostały także opisane domyślne ustawienia dotyczące haseł w systemach operacyjnych w SGH.

Rozdział 4 to prezentacja programu, który kompleksowo i znacznie lepiej niż istniejące do tej pory programy radzi sobie z problemem zmiany haseł we wszystkich systemach. W wyjaśnieniach szczególny nacisk położono na algorytm obliczania siły hasła, który został zacytowany in extenso.

W rozdziale 5 na podstawie szczegółowych logów z systemu działającego w SGH wyliczono koszt zmiany haseł, a także określono szacunkowe koszty przygotowania oprogramowania do zmiany haseł.

2. Bezpieczeństwo systemów informatycznych

Celem każdego zespołu politycznego jest zachowanie przyrodzonych
i nieulegających przedawnieniu praw człowieka.
Tymi prawami są wolność, własność, bezpieczeństwo i opieranie się uciskowi.
Deklaracja praw człowieka i obywatela, 1789

Bezpieczeństwo to jedno z tych kilku pojęć, które instynktownie rozumiemy, a które wcale nie jest tak łatwo wytłumaczyć. W Słowniku Języka Polskiego[1)] mamy definicję: *stan niezagrożenia, spokoju, pewności.*

Spróbujmy to odnieść do systemów informatycznych. Bezpieczeństwo w sensie informatycznym to stan pewności, że system teleinformatyczny będzie realizował tylko i wyłącznie cele zgodne z intencjami właściciela[2)].

Taka definicja jest dobrym przybliżeniem problemu, ale jest trudna do zastosowania w technice. Bezpieczeństwo odnoszące się do systemów informatycznych lepiej określać deskryptywnie przez atrybuty bezpieczeństwa, zwracając uwagę na jego interdyscyplinarny charakter: zbiór zagadnień z dziedziny informatyki związany z szacowaniem i kontrolą ryzyka wynikającego z korzystania z komputerów i sieci komputerowych, rozpatrywany z perspektywy poufności, integralności i dostępności danych. Najważniejsze atrybuty[3)] z wyjaśnieniem ich znaczenia zostały zebrane w tabeli 2.1.

[1)] *Słownik Języka Polskiego*, pod red. M. Szymczaka, Wydawnictwo Naukowe PWN, Warszawa, 1999, s. 139.

[2)] A. Białas, *Bezpieczeństwo informacji i usług w nowoczesnej instytucji i firmie*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2006, s. 27.

[3)] ibidem, s. 34.

nazwa	opis atrybutu
poufność	dane nie są udostępniane ani ujawniane nieautoryzowanym osobom, podmiotom lub procesom
integralność	dane nie zostały zmienione lub zniszczone w sposób nieautoryzowany, a system realizuje zamierzone funkcje w nienaruszony sposób, wolny od nieautoryzowanej manipulacji, celowej lub przypadkowej
dostępność	dane są dostępne i możliwe do wykorzystania na żądanie w założonym czasie przez kogoś lub coś, kto lub co ma do tego prawo
autentyczność	tożsamość podmiotu lub zasobu jest taka, jak deklarowana; dotyczy użytkowników, procesów, systemów lub instytucji; jest związana z badaniem, czy ktoś lub coś jest tym, za kogo lub za co się podaje
rozliczalność	działania podmiotu (np. użytkownika) mogą być jednoznacznie przypisane tylko temu podmiotowi
niezawodność	spójne, zamierzone zachowanie i skutki

Tabela 2.1. Atrybuty bezpieczeństwa

2.1. Polityka bezpieczeństwa

Gdy mowa o bezpieczeństwie systemu informatycznego, pierwsze skojarzenia to hasła i kopie zapasowe. Istotnie, są one jego elementami, ale jednymi z bardzo licznych, w dodatku nisko umieszczonymi w strukturze bezpieczeństwa. Skomplikowana struktura bezpieczeństwa wymaga mechanizmów i narzędzi do jej opisu, w innym przypadku można zbyt łatwo zapomnieć o jakimś jej fragmencie. Jednym z takich kompleksowych mechanizmów jest polityka bezpieczeństwa.

Ogólnie rzecz ujmując, jak pisze Stefanowicz[4)], polityka bezpieczeństwa to zestaw reguł określających wykorzystanie informacji, łącznie z jej przetwarzaniem, przechowywaniem, dystrybucją i prezentacją niezależnie od wymagań dotyczących bezpieczeństwa i celów bezpieczeństwa, a także plan lub sposób postępowania przyjęty w celu zapewnienia bezpieczeństwa systemu informatycznego.

[4)] B. Stefanowicz, *Informacyjne systemy zarządzania - przewodnik*, Oficyna Wydawnicza SGH, Warszawa, 2007, s. 79.

Polityka bezpieczeństwa firmy jest związana z profilem działalności danej firmy, stąd nie ma jednego obowiązującego standardu, aczkolwiek wiele szczegółowych wskazówek zostało zebranych przez różne organizacje, m.in. amerykański National Institute for Standards and Technology (NIST), brytyjski British Standards Institute (BSI), Polski Komitet Normalizacyjny (PKN) wydający Polskie Normy oraz międzynarodową International Organization for Standardization (ISO). Niektóre ze wskazówek zostały wydane jako przewodniki certyfikacji, które – po spełnieniu wszystkich warunków w nich zawartych – pozwalają na dobre oszacowanie własnego systemu bezpieczeństwa oraz zwracają uwagę na jego wrażliwe elementy.

ISO publikuje serię norm pod wspólną numeracją 27000 dotyczącą bezpieczeństwa informacji. W 1995 r. BSI wydał dokument BS 7799-1, w którym opisał podstawowe praktyki związane z zarządzaniem bezpieczeństwa informacji. Dokument ten służył jako podstawa dla organizacji ISO do wydania w 2000 r. normy ISO/IEC 17799 *Information Technology – Code of practice for information security management*, która po poprawkach z 2005 r. została w 2007 r. włączona do serii 27000 pod numerem 27002[5)]. W dokumencie opisano najlepsze praktyki dla wdrażających systemy zarządzania bezpieczeństwem informacji. Drugi ważny dokument ISO dotyczący bezpieczeństwa także wziął swój początek z normy BSI, a konkretnie BS 7799-2, wydanej w 1999 r. i zatytułowanej *Information Security Management Systems - Specification with guidance for use*, która skupiała się na sposobach implementacji systemu zarządzania bezpieczeństwem informacji. Wersja tego dokumentu z 2002 r. opisywała „Plan-Do-Check-Act", w której były zawarte zalecenia kontroli jakości poruszane także w serii ISO 9000. Pod koniec roku 2005 BS 7799-2 został włączony przez ISO do serii 27000 pod numerem 27001[6)]. Norma ta określa wymagania związane z systemem zarządzania bezpieczeństwem informacji pod kątem certyfikacji. Została przetłumaczona na język polski przez PKN i wydana pod nazwą PN-I-07799-2:2005[7)] (po numeracji widać źródło brytyjskie). Zagadnienia zawarte w kolejnych rozdziałach tej normy pokazano[8)] w tab. 2.2.

[5)] ISO/IEC 27002:2005, *Information technology – Security techniques – Code of practice for information security management*.

[6)] ISO/IEC 27001:2005, *Information technology – Security techniques – Information security management systems – Requirements*.

[7)] PN-I-07799: *Systemy zarządzania bezpieczeństwem informacji – Specyfikacje i wytyczne do stosowania*, PKN, 2005.

[8)] A. Białas, *Bezpieczeństwo...*, op. cit., s. 277.

rozdział	zagadnienia
3	Polityka bezpieczeństwa
4	Organizacja bezpieczeństwa
5	Klasyfikacja i kontrola aktywów
6	Bezpieczeństwo osobowe
7	Bezpieczeństwo fizyczne i środowiskowe
8	Zarządzanie systemami i sieciami
9	Kontrola dostępu do systemu
10	Rozwój i utrzymanie systemu
11	Zarządzanie ciągłością działania
12	Zgodność

Tabela 2.2. Zagadnienia zawarte w normie PN-I-07799

Bezpieczeństwo systemu informatycznego jest na tyle złożonym problemem, że OECD (Organisation for Economic Co-operation and Development) wydała broszurę, w której przedstawiła zbiór wytycznych[9)], które ułatwiają tworzenie *kultury bezpieczeństwa*:

— świadomość: uczestnicy powinni być świadomi potrzeby bezpieczeństwa systemów i sieci informatycznych oraz kroków, jakie mogą podjąć w celu poprawy bezpieczeństwa,
— odpowiedzialność: wszyscy uczestnicy są odpowiedzialni za bezpieczeństwo systemów i sieci,
— reakcja: uczestnicy powinni działać bez zwłoki i współpracować ze sobą w celu zapobiegania, wykrywania i reagowania na naruszenia bezpieczeństwa,
— etyka: uczestnicy powinni szanować uzasadnione dobra innych,
— demokracja: bezpieczeństwo systemów i sieci informatycznych powinno być zgodne z podstawowymi wartościami społeczeństwa demokratycznego,
— ocena ryzyka: uczestnicy powinni przeprowadzać oceny ryzyka,
— projektowanie i wdrażanie rozwiązań z zakresu bezpieczeństwa: uczestnicy powinni włączać rozwiązania z zakresu bezpieczeństwa do systemów i sieci informatycznych jako elementy kluczowe,
— zarządzanie bezpieczeństwem: uczestnicy powinni przyjąć całościowe podejście do zarządzania bezpieczeństwem,

[9)] *OECD Guidelines for the Security of Information Systems and Networks: Towards a Culture of Security*, OECD, Paris, 2002, s. 4.

— przegląd: użytkownicy powinni dokonywać przeglądów i ocen bezpieczeństwa systemów i sieci informatycznych oraz wprowadzać niezbędne zmiany do polityk, praktyk, środków i procedur dotyczących bezpieczeństwa.

Zbiór tych wytycznych został zaadaptowany jako Rekomendacje Rady OECD na 1037. sesji 25 lipca 2002 r. i włączony do zbioru dokumentów dotyczących szeroko pojętego bezpieczeństwa[10]. Postawy propagowane przez OECD są niezwykle istotne – w kontekście dużego udziału ludzi w bezpieczeństwie informacji nie możemy mówić o prawdziwym bezpieczeństwie bez *kultury bezpieczeństwa*.

Omówione wyżej zalecenia, normy i rekomendacje są szablonami do tworzenia podobnych dokumentów przystosowanych do potrzeb konkretnej firmy. Warunkiem sine qua non tworzenia polityki bezpieczeństwa jest podjęcie przez zarząd inicjatywy budowy systemu bezpieczeństwa instytucji i zadeklarowanie nadzoru oraz wsparcia dla całości przedsięwzięcia[11]. Dopiero wtedy można powoływać zespół specjalistów, który, projektując politykę bezpieczeństwa, obejmie nią wszystkich pracowników, przeanalizuje zapotrzebowanie firmy na bezpieczeństwo, wyszczególni problematyczne obszary, wdroży odpowiednie procedury, słowem – w możliwie najlepszym stopniu spróbuje zabezpieczyć aktywa informacyjne.

2.2. Hasła w systemie bezpieczeństwa

Jednym z najważniejszych obszarów polityki bezpieczeństwa jest poufność (tab. 2.1), czyli zapewnienie dostępu do danych zasobów tylko dla ściśle określonych osób. Zwykle stosuje się jeden lub kilka z poniższych sposobów weryfikacji:

— coś, co wiesz (np. hasło),
— coś, co masz (np. karta, token),
— coś, czym jesteś (np. odcisk palca),
— coś, co robisz (np. podpis długopisem).

Coś, co wiesz jest najbardziej rozpowszechnionym sposobem uwierzytelniania. Ma oczywiste zalety – nie potrzeba dodatkowych urządzeń, ale też i wady – hasło można zapomnieć, a jeśli się je zapisze, to ktoś inny może je znaleźć.

---

[10] *OECD Directorate for Science, Technology and Industry*. Adres: `http://www.oecd.org/sti/`

[11] A. Białas, *Bezpieczeństwo...*, op. cit., s. 199.

Coś, co masz jest całkiem popularne w systemach bankowych. Eliminuje problem zapomnienia hasła, ale tworzy nowy: trzeba coś ze sobą nosić, a także tego pilnować, żeby się nie okazało, że jest to coś, co ma już złodziej. Ostatnimi czasy zastępowane przez karty kodów jednorazowych.

Coś, czym jesteś, jak np. odcisk palca czy skan oka, łączy zalety dwóch poprzednich: nie trzeba nosić nic dodatkowo, nie można tego też zapomnieć. Niestety, ma też wady: urządzenia do odczytu danych biometrycznych są dość drogie, wciąż nie są dość dokładne, a także nie są zainstalowane wszędzie tam, gdzie byśmy chcieli skorzystać z dostępu do potrzebnych nam danych. Problemem jest także ich niezastępowalność. Jeśli komuś uda się podrobić odcisk palca, to pozostaje nam tylko zrezygnować z tej metody weryfikacji.

Coś, co robisz, jak podpis, jest ze względów technologicznych podobne do *tego, czym jesteś*, tj. także wymaga specjalnych czytników, ale ma dodatkową zaletę – często można z tego skorzystać także poza systemem, choćby na zwykłej kartce papieru.

Specjaliści polecają kombinację przynajmniej trzech z powyższych, a w przypadku mniej istotnych danych, kombinację dwóch. Nie wszędzie jednak da się to zrealizować i wybierany jest wariant najprostszy i najbardziej rozpowszechniony: hasła. Użycie hasła w celu uwierzytelnienia swojej tożsamości jest niemal tak stare, jak ludzka cywilizacja, zwłaszcza jej wojenna część. Procedury, które pozwalały wartownikowi przepuszczać tylko tych, którzy znają hasło, to starożytna forma oprogramowania uwierzytelniającego.

Hasła możemy podzielić[12)] na jednorazowe, wielokrotne oraz typu wezwanie-odpowiedź. Hasła jednorazowe, często realizowane w urządzeniach sprzętowych, są oparte na sekwencjach liczb pseudolosowych – kolejne hasła są obliczane na podstawie wartości początkowej. Bez znajomości wartości początkowej nie da się obliczyć następnego hasła, nawet jeżeli udało się podsłuchać lub przechwycić poprzednie hasło. Miało to szczególne znaczenie po rozpowszechnieniu sieci internetowej, a przed rozpowszechnieniem technik szyfrowania. Z haseł wielokrotnych, jak sama nazwa wskazuje, użytkownik korzysta do uwierzytelniania w systemie komputerowym wiele razy w ciągu określanego przez administratora okresu (np. 60 dni). Hasła takie są wrażliwe na podsłuchanie, co wymusza stosowanie technik szyfrowania transmisji, a także na złamanie i ponowne wykorzystanie przez atakującego (o czym więcej w rozdziale 3.1). Problemy związane z hasłami wielokrotnymi:

[12)] D. Pipkin, *Bezpieczeństwo informacji*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2000, s. 156-158.

— użytkownicy przestają korzystać z usług ze względu na zapomniane hasło lub wymóg posiadania zbyt trudnego hasła,
— użytkownicy wymagają przeszkolenia odnośnie polityki bezpieczeństwa, haseł, metod ich zmiany i zapobiegania zapomnieniu,
— użytkownicy nie przywiązują wagi do swoich haseł („żadnych danych tam nie mam"),
— użytkownicy są lekkomyślni i „pożyczają" swoje hasła innym,
— użytkownicy oddają hasło za czekoladę[13)],
— użytkownicy zapisują hasła na karteczkach (a karteczki gubią),
— użytkownicy zapominają hasła.

Część wspomnianych problemów można zignorować. Jeżeli użytkownicy przestają korzystać z usług tylko dlatego, że muszą pamiętać hasło, to widocznie usługi te nie są dla nich wystarczająco ważne. Oczywiście łatwiej jest prowadzić sztywną politykę haseł, jeżeli jesteśmy quasimonopolistą i użytkownicy muszą korzystać z naszych zasobów, choćby tylko dwa razy do roku. Nie da się jednak ukryć, że zwiększy to koszty obsługi użytkownika w okolicach tych właśnie terminów, kiedy użytkownicy będą musieli skorzystać z systemu. Niektórym użytkownikom może pomóc znajomość różnego rodzaju sztuczek mnemotechnicznych pomagających tworzyć bezpieczne, trudne, a jednocześnie banalnie łatwe do zapamiętania hasła. Umysł ludzki jest ewolucyjnie przystosowany do pamiętania sensownie łączących się w zdania związków wyrazowych, niemal każdy z nas bez problemu wyrecytuje jakiś wiersz czy zaśpiewa piosenkę. Gdyby chcieć nauczyć się ciągu tylu losowych liter, ile było wyrazów w takim utworze, to – choć na pewno da się to zrobić – byłoby to nieporównanie trudniejsze do przeprowadzenia. Na tym właśnie opiera się przykładowa sztuczka mnemotechniczna, zresztą zalecana przez ekspertów ds. bezpieczeństwa: należy wymyśleć jakieś zdanie, powiedzmy *Moja dobra przyjaciółka ma 30 lat, piękny wiek!*, wybrać z niego pierwsze (lub drugie, lub ostatnie, wszystko jedno) litery oraz interpunkcję i taki zbitek znaków *Mdpm30l,pw!* stosować jako hasło. Oczywiście w miejscach, gdzie jest to możliwe, lepiej po prostu zastosować całe takie zdanie.

[13)] Ponad 70% osób w zamian za czekoladę ujawniło swoje hasło ankieterom badania przeprowadzonego w trakcie konferencji „Infosecurity Europe" 27-29 kwietnia 2004 r. Wyniki nie były dużo lepsze podczas takiej samej konferencji trzy lata później. W sierpniu 2007 r. 61 ze 102 pracowników amerykańskiego urzędu skarbowego (IRS) zgodziło się na ujawnienie swojego loginu oraz zmianę swojego hasła na zasugerowane przez dzwoniącego bez żadnej identyfikacji dzwoniącego, a tylko ośmiu zgłosiło sprawę do oficera bezpieczeństwa.

Wymóg przeszkolenia jest wspomniany tutaj tylko dla formalności, nie stanowi problemu jako taki, choć jest to czynnik generujący koszty. Należy jednak przyjąć, że wszyscy użytkownicy powinni zapoznać się z systemem działającym w ich środowisku, a w trakcie takiego poznawania obsługa powinna przeszkolić użytkowników z obowiązującej polityki bezpieczeństwa, w tym haseł i dobrych praktyk korzystania z sieci.

Wielu użytkowników bagatelizuje problem bezpieczeństwa informacji w ogólności, a haseł w szczególności. Najczęściej używanym argumentem jest, że użytkownik nie ma na koncie nic takiego, co by chciał tak bardzo chronić mocnymi hasłami. To lekkomyślne podejście, gdyż rzadko kiedy użytkownik istnieje sam w systemie komputerowym, zwykle wszystkie osoby należące do organizacji mają konta w tym samym systemie. O ile pliki czy poczta tego konkretnego użytkownika mogą nie być przedmiotem zainteresowania, o tyle pliki innych użytkowników – jak najbardziej. Trzeba zdawać sobie sprawę z tego, że dla atakujących systemy komputerowe najtrudniejszym krokiem jest pierwszy: wejście do systemu z dowolnie niskim poziomem uprawnień. Uzyskanie uprawnień administratora jest o wiele łatwiejsze, jeżeli jest się zalogowanym do danego systemu komputerowego, o atakowaniu wewnętrznych systemów nie wspominając. Łańcuch jest na tyle silny, na ile silne jest jego najsłabsze ogniwo, nie należy więc umniejszać wartości swojego hasła.

O tym wszystkim obsługa sieci powinna poinformować na samym początku użytkowania systemu, takie szkolenia znakomicie by pomogły rozwiać niektóre szkodliwe mity krążące wokół problematyki bezpieczeństwa, a może nawet pozwoliłyby zapobiec niektórym zagrożeniom bezpieczeństwa.

Pożyczanie haseł innym użytkownikom może wydawać się niewinne, jeżeli np. służy tylko temu, żeby zalogować się na komputer, z którego potem ktoś będzie korzystał. Ale tylko pod warunkiem, że jest to uprawniony użytkownik systemu, o czym udostępniający użytkownik rzadko jest pewny (choć bardzo często przekonany). W przeciwnym wypadku mamy do czynienia ze zwykłą kradzieżą zasobów: komputera, miejsca, czasu, sieci. W czasie, w którym korzysta z systemu osoba nieuprawniona, ktoś inny nie powinien stać w kolejce do ograniczonego zasobu. Dodatkowo użyczone hasło to zmniejszenie bezpieczeństwa systemu jako całości, gdyż jeśli pożyczono hasło osobie o złych zamiarach, to będzie ona mogła w znacznie bardziej efektywny sposób przeprowadzić kolejne etapy ataku na system. Podobny problem uwidacznia się przy zapisywaniu haseł na karteczkach i przyklejaniu tych karteczek do monitora lub gubieniu ich. Hasła powinny być takie, żeby było je łatwo zapamiętać i nie trzeba było ich zapisywać. Należy trzymać się za-

sady, że sekretu najlepiej dotrzymuje jedna osoba. Hasło na kartce to hasło znane wielu osobom. Problem oddawania haseł za symboliczną czekoladę to kolejny problem braku przeszkolenia użytkowników.

Należy mieć przy tym świadomość, że pamięć ludzka jest ułomna. Jeżeli nakazujemy użytkownikom tylko pamiętać hasła, to – zwłaszcza jeżeli są to skomplikowane hasła, a użytkownicy nie znają żadnych sztuczek na tworzenie łatwych do zapamiętania, ale trudnych haseł – będziemy borykać się z problemem przywracania użytkownikom dostępu do usług. Zwykle dzieje się to przez nadanie nowego hasła przez upoważnioną obsługę, ale jest to dość kosztowna operacja, o czym napisano w rozdziale 5.1.

2.3. Uwarunkowania prawne problematyki bezpieczeństwa

Na bazie prawa polskiego, kiedy mowa ogólnie o systemie informatycznym i przestępstwach przeciwko niemu, mamy do dyspozycji cały rozdział Kodeksu Karnego zatytułowany „Przestępstwa przeciwko ochronie informacji”[14)], który miał ważną nowelizację[15)] rozszerzającą katalog przestępstw. W jego kolejnych artykułach znajdziemy odzwierciedlenie poszczególnych pól z dziedziny bezpieczeństwa z początku rozdziału, choć czasem ograniczone tylko do administracji państwowej.

Zastanówmy się, co się dzieje, gdy użytkownik nie stosuje się do zaleceń administratora systemu w zakresie tajemnicy haseł. Prawie nic (podkreślenia własne):

> Art. 269a. Kto, nie będąc do tego uprawnionym, przez transmisję, zniszczenie, usunięcie, uszkodzenie lub zmianę danych informatycznych, w istotnym stopniu zakłóca pracę systemu komputerowego lub sieci teleinformatycznej, podlega karze pozbawienia wolności od 3 miesięcy do lat 5.
>
> Art. 269b. § 1. *Kto* wytwarza, pozyskuje, zbywa lub *udostępnia innym osobom* urządzenia lub programy komputerowe przystosowane do popełnienia przestępstwa określonego w art. 165 § 1 pkt 4[16)], art. 267 § 2, art. 268a § 1 albo § 2 w związku

[14)] *Ustawa z dnia 6 czerwca 1997 r. – Kodeks karny*, Dz. U. z 1997 r. Nr 88, poz. 553, rozdział XXXIII.

[15)] *Ustawa z dnia 18 marca 2004 r. o zmianie ustawy – Kodeks karny, ustawy – Kodeks postępowania karnego oraz ustawy – Kodeks wykroczeń*, Dz. U. z 2004 r. Nr 69, poz. 626.

[16)] Art. 165 § 1. Kto sprowadza niebezpieczeństwo dla życia lub zdrowia wielu osób albo dla mienia w wielkich rozmiarach [...] pkt 4. zakłócając, uniemożliwiając lub w inny sposób wpływając na automatyczne przetwarzanie, gromadzenie lub przesyłanie informacji

> z § 1, art. 269 § 2[17)] albo art. 269a, a także *hasła komputerowe, kody dostępu lub inne dane umożliwiające dostęp do informacji przechowywanych w systemie komputerowym lub sieci teleinformatycznej, podlega karze pozbawienia wolności do lat 3.*

Wszystko oczywiście zależy od tego, komu użytkownik hasła ujawni, ale czy w życiu można być pewnym kogokolwiek? Lepiej haseł nie ujawniać. Administratorowi pozostaje wymuszanie polityki haseł i działania lub sankcje na poziomie organizacji. Jeżeli użytkownik podpisywał regulamin, w którym zobowiązywał się do nieujawniania hasła, sytuacja (dla administratora) może być trochę lepsza:

> Art. 266. § 1. Kto, wbrew przepisom ustawy lub przyjętemu na siebie zobowiązaniu, ujawnia lub wykorzystuje informację, z którą zapoznał się w związku z pełnioną funkcją, wykonywaną pracą, działalnością publiczną, społeczną, gospodarczą lub naukową, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 2.
> § 2. Funkcjonariusz publiczny, który ujawnia osobie nieuprawnionej informację stanowiącą tajemnicę służbową lub informację, którą uzyskał w związku z wykonywaniem czynności służbowych, a której ujawnienie może narazić na szkodę prawnie chroniony interes, podlega karze pozbawienia wolności do lat 3.

Jeżeli nie ma żadnego sposobu, żeby wymusić stosowanie dobrych haseł, użytkownicy nie będą ich stosowali (może trzeba w takim wypadku zmienić system?), a wszelkie konsekwencje, mimo że w pewien sposób zawinione przez użytkowników, spadną na administratora. Po fakcie można ścigać i karać włamywacza, tu katalog został ostatnio (w 1997 r.) znacznie rozszerzony.

[17)] Art. 269. § 1. Kto niszczy, uszkadza, usuwa lub zmienia dane informatyczne o szczególnym znaczeniu dla obronności kraju, bezpieczeństwa w komunikacji, funkcjonowania administracji rządowej, innego organu państwowego lub instytucji państwowej albo samorządu terytorialnego albo zakłóca lub uniemożliwia automatyczne przetwarzanie, gromadzenie lub przekazywanie takich danych, podlega karze pozbawienia wolności od 6 miesięcy do lat 8.
§ 2. Tej samej karze podlega, kto dopuszcza się czynu określonego w § 1, niszcząc albo wymieniając nośnik informacji lub niszcząc albo uszkadzając urządzenie służące do automatycznego przetwarzania, gromadzenia lub przekazywania danych informatycznych.

Art. 267. § 1. Kto bez uprawnienia uzyskuje informację dla niego nie przeznaczoną, otwierając zamknięte pismo, podłączając się do przewodu służącego do przekazywania informacji lub przełamując elektroniczne, magnetyczne albo inne szczególne jej zabezpieczenie, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 2.

§ 2. Tej samej karze podlega, kto w celu uzyskania informacji, do której nie jest uprawniony, zakłada lub posługuje się urządzeniem podsłuchowym, wizualnym albo innym urządzeniem specjalnym.

Art. 268. § 1. Kto, nie będąc do tego uprawnionym, niszczy, uszkadza, usuwa lub zmienia zapis istotnej informacji albo w inny sposób udaremnia lub znacznie utrudnia osobie uprawnionej zapoznanie się z nią, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 2.

§ 2. Jeżeli czyn określony w § 1 dotyczy zapisu na komputerowym nośniku informacji, sprawca podlega karze pozbawienia wolności do lat 3.

Art. 268a. § 1. Kto, nie będąc do tego uprawnionym, niszczy, uszkadza, usuwa, zmienia lub utrudnia dostęp do danych informatycznych albo w istotnym stopniu zakłóca lub uniemożliwia automatyczne przetwarzanie, gromadzenie lub przekazywanie takich danych, podlega karze pozbawienia wolności do lat 3.

§ 2. Kto, dopuszczając się czynu określonego w § 1, wyrządza znaczną szkodę majątkową, podlega karze pozbawienia wolności od 3 miesięcy do lat 5.

Art. 287. § 1. Kto, w celu osiągnięcia korzyści majątkowej lub wyrządzenia innej osobie szkody, bez upoważnienia, wpływa na automatyczne przetwarzanie, gromadzenie lub przekazywanie danych informatycznych lub zmienia, usuwa albo wprowadza nowy zapis danych informatycznych, podlega karze pozbawienia wolności od 3 miesięcy do lat 5.

Ściganie wyżej wymienionych przestępstw, z wyjątkiem tych przeciwko administracji państwowej, odbywa się na wniosek pokrzywdzonego.

Nie należy też zapominać, że większość przestępstw przeciwko informacji w firmach jest popełnianych przez obecnych lub byłych pracowników. Choć sankcje nie kończą się na kodeksie karnym – na polu prawa cywilnego na pewno znajdzie się wiele paragrafów, które może wykorzystać poszkodowany – firmy starają się ukrywać takie wydarzenia, żeby nie niepokoić inwestorów. Taka praktyka jest w długim okresie szkodliwa, gdyż nie eliminuje z rynku złych pracowników i pozwala im wykorzystywać kolejne podmioty.

W prawie polskim sprawa haseł jest wspomniana w kontekście zabezpieczania danych osobowych. W załączniku do rozporządzenia Ministra Spraw Wewnętrznych i Administracji[18)] w rozdziale „Środki bezpieczeństwa na poziomie podstawowym" znajduje się akapit:

> W przypadku gdy do uwierzytelniania użytkowników używa się hasła, jego zmiana następuje nie rzadziej niż co 30 dni. Hasło składa się co najmniej z 6 znaków.

a w rozdziale „Środki bezpieczeństwa na poziomie podwyższonym":

> W przypadku gdy do uwierzytelniania użytkowników używa się hasła, składa się ono co najmniej z 8 znaków, zawiera małe i wielkie litery oraz cyfry lub znaki specjalne.

Zbadanie siły takich haseł zostanie przeprowadzone w następnym rozdziale.

[18)] *Rozporządzenie Ministra Spraw Wewnętrznych i Administracji z dnia 29 kwietnia 2004 r. w sprawie dokumentacji przetwarzania danych osobowych oraz warunków technicznych i organizacyjnych, jakim powinny odpowiadać urządzenia i systemy informatyczne służące do przetwarzania danych osobowych*, Dz. U. z 2004 r. Nr 100, poz. 1024.

3. Hasła

Bezpieczeństwo systemu kryptograficznego
nie może być oparte na utrzymywaniu tajności algorytmu,
tylko na utrzymywaniu tajności klucza.
Zasada Kerckhoffa[19)]

3.1. Entropia haseł w teorii informacji

Przy wyborze haseł od samego początku jest prowadzony swoisty wyścig między administratorami systemów a włamywaczami. Administratorzy narzucają użytkownikom różne wymagania dotyczące haseł, które zminimalizują możliwości zgadnięcia hasła przez włamywacza, a jednocześnie nie uniemożliwią użytkownikom codziennego używania. Po stronie włamywaczy stoją postęp technologiczny umożliwiający szybsze sprawdzanie kolejnych haseł i, niestety, niefrasobliwość użytkowników.

Jakie powinny być hasła? Prawidłową odpowiedzią jest „to zależy". A zależy od tego, czego te hasła bronią. Przy mało istotnych danych, powiedzmy dostęp do darmowej, dostępnej tylko po zalogowaniu strony WWW, hasło nie musi być tak samo silne jak przy dostępie do konta bankowego. Problem oceny ważności danych najlepiej by było zostawić użytkownikom, ale, niestety, mają oni tendencje do niedoszacowywania wartości danych i znacznego przeszacowywania siły wymyślonego przez nich hasła. Ale skąd wiadomo, jak silne jest hasło? Jeżeli rozpatrujemy teoretyczną podatność hasła na przeszukiwanie całej przestrzeni haseł, to musimy odwołać się do pojęcia entropii.

Entropia to miara niepewności, nieporządku, chaosu. Im coś jest bardziej pewne, tym ma mniejszą entropię. Zdarzenie pewne ma entropię zerową. Płeć losowo wybranej osoby dostarcza 1 bitu informacji: albo jest to mężczyzna, albo kobieta (= nie-mężczyzna). Wynik rzutu kostką, 6 możliwych stanów, to niecałe 3 bity (całe 3 bity są dla 8 stanów) informacji. Entropia to najmniejsza *średnia* ilość informacji potrzebna do zakodowania faktu

[19)] A. Kerckhoff, *La Cryptographie militaire*, Libraire Militaire de L. Baudoin & Co., Paris, 1883.

zajścia zdarzenia ze zbioru zdarzeń o danych prawdopodobieństwach, co przy zdarzeniach równoprawdopodobnych jest tożsame z liczbą bitów dostarczanych przez każde zdarzenie. Pojęcie entropii jest często wykorzystywane w teorii informacji, dziale nauki utworzonym przez Claude'a E. Shannona w latach 40. ubiegłego wieku. Shannon w 1948 r. w pracy „A Mathematical Theory of Communication" dał podwaliny ilościowej teorii informacji[20].

Wzór na entropię, wzięty wprost z teorii informacji:

$$H = -\sum_{i=1}^{n} p(i) \log_2 p(i) \tag{3.1}$$

gdzie $p(i)$ to prawdopodobieństwo zajścia zdarzenia i. Przy zdarzeniach równoprawdopodobnych, czyli w przypadku hasła przy tak samo losowym wyborze dowolnego znaku ze zbioru o mocy N, otrzymujemy entropię każdego znaku:

$$H = -\sum_{i=1}^{N} \frac{1}{N} \log_2 \frac{1}{N} = -N \frac{1}{N} \log_2 \frac{1}{N} = -\log_2 \frac{1}{N} = \log_2 N$$

a dla całego hasła

$$H = L \log_2 N \tag{3.2}$$

gdzie L to długość hasła, a N to moc zbioru znaków tworzących hasło, czyli liczba wszystkich możliwych znaków, z których można zbudować hasło. Ze względu na algorytmy łamiące hasła zbiory znaków klasyfikujemy w cztery kategorie: małe litery ($n = 26$), wielkie litery ($n = 26$), cyfry ($n = 10$) i znaki specjalne ($n = 33$). Czasem klasę znaków specjalnych dzieli się na trzy: spację, przecinek i kropkę ($n = 3$), znaki specjalne umieszczone na klawiaturze komputerowej na cyfrach ($n = 10$) i pozostałe ($n = 19$). Przy wystąpieniu w haśle znaku z danej klasy moc zbioru znaków hasła powiększa się o cały zestaw możliwych znaków danej klasy.

Dla przykładu sześcioznakowe hasło składające się tylko z małych liter od a do z (czyli 26 znaków) ma teoretyczną (przy założeniu doskonałej przypadkowości przy wyborze każdej z liter) entropię:

$$H = 6 \log_2 26 \approx 28$$

Czyli przy ograniczonym w ten sposób zestawie znaków dostajemy tylko 28 bitów chaosu, a nie, jak by można sądzić z samego zapisu znaków (każdy znak jest zapisany na 8 bitach),

[20] C. Shannon, *A mathematical theory of communication*, Bell System Technical Journal, 1948, t. 27, s. 379-423.

48 bitów. Utworzenie hasła 6-znakowego z liter małych i wielkich powiększa dostępną przestrzeń do 52 znaków i zwiększa entropię takiego hasła

$$H = 6 \log_2 52 \approx 34$$

do 34 bitów.

Teraz już można łatwo wyliczyć, jakie hasła zaleca MSWiA (por. s. 17): na poziomie podstawowym wymagane są hasła o entropii minimum 28 bitów (bardzo mało), a na poziomie podwyższonym min. 48 bitów (średnio, ale do przyjęcia).

Tu należy wyjaśnić, w jaki sposób system jest narażony przez słabe hasło. W praktyce rozważamy dwa przypadki ataku: zgadywanie hasła i łamanie hasła.

Ze zgadywaniem hasła mamy do czynienia w przypadku, gdy włamywacz próbuje się zalogować do jakiejś usługi w naszym imieniu. W tym celu wysyła do serwera „pytanie" z wymyślonym przez siebie hasłem i czeka na odpowiedź. Ze względu na czas trwania transmisji TCP, a także celowe opóźnienia wprowadzane przez system przy błędnych próbach logowania, taki sposób nie jest zbyt szybki, jak na świat komputerów. Powiedzmy, że przy bardzo optymistycznych założeniach można w ten sposób sprawdzić ok. 300 haseł na sekundę.

W przypadku 6-literowych haseł bez ograniczeń klasy znaków jest ich wszystkich $255^6 = 274941996890625$, ale ponieważ użytkownicy korzystają z mniejszego zbioru znaków drukowalnych (a także ze względu na oczywiste problemy z wpisywaniem pozostałych znaków przy różnych układach klawiatury na różnych komputerach), w praktyce rozważa się 95 możliwych znaków, co daje 374 razy mniejszą przestrzeń $95^6 = 735091890625$ możliwych haseł. Entropia takich haseł to $6 \log_2 95 \approx 39$ bitów.

To i tak nie jest mało. Przy dość niskiej wydajności sprawdzania haseł włamywacz nie próbuje *wszystkich* teoretycznych haseł, tylko zawęża pole poszukiwań do najbardziej prawdopodobnych. Dział nadużyć (ang. *fraud unit*) firmy Deloitte&Touche w 1997 r. opracował[21)] listę najczęściej używanych haseł. Na pierwszym miejscu było imię, nazwisko lub imię dziecka użytkownika, a na drugim słowo „sekret". Jeśli te najpopularniejsze nie pasują, pozostaje sprawdzanie haseł ze słownika, a potem dodawania do nich cyfr i kilka innych technik[22)]. Nawet upraszczając problem do wszystkich permutacji po małych

[21)] L. Light, *Hackers' delight*, Bussiness Week, wyd. 1997-02-10.

[22)] B. Schneier, *Kryptografia dla praktyków*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2002, s. 232-233.

literach i cyfrach, na początek zostaje $(26 + 10)^6 = 2176782336$ haseł do sprawdzenia. Entropia takich haseł wynosi $6 \log_2 36 = 31$ bitów. Jak łatwo policzyć, to wciąż zajmuje 84 dni nieprzerwanego zgadywania. Zwykle w tym czasie hasło zostanie zmienione na nowe. Rozszerzenie zestawu znaków o 26 wielkich liter powoduje zwiększenie przestrzeni haseł 26-krotnie[23] do $(26+10+26)^6 = 56800235584$, zwiększenie entropii do $6 \log_2 62 \approx 34$ bitów i wydłużenie czasu zgadywania do ponad 6 lat.

Na rysunku 3.1 przedstawiono teoretyczną entropię haseł w zależności od długości hasła i użytych w nim klas znaków: małych liter, wielkich liter, cyfr oraz znaków specjalnych (typu $, !, #, %), w sumie 95 znaków. Na rysunku widać, że aby hasło osiągnęło 60 bitów entropii, to musi składać się przynajmniej z 10 znaków, a gdyby używać tylko małych (lub wielkich) liter, to wystarczy 13 znaków. Można też zauważyć, że łatwiej uzyskamy silniejsze hasło przez jego wydłużenie, niż przez skorzystanie z większej liczby klas znaków. Jeżeli tylko hasło nie jest oparte na słowniku, zgadywanie haseł nie jest takim zagrożeniem.

Gorzej z łamaniem haseł. Hasła nie są (a przynajmniej nie powinny być) w ogóle w systemie przechowywane w postaci jawnej lub umożliwiającej odczytanie hasła[24]. W systemie jest przechowywany wynik albo funkcji szyfrującej z hasłem jako kluczem, albo funkcji skrótu z hasłem jako argumentem wejściowym.

Najstarsza funkcja szyfrująca w systemach Unix to `crypt()`, która korzysta z algorytmu DES (który z kolei jest osłabionym szyfrem Lucifer[25] zaprojektowanym przez Horsta Feistela). Funkcja ta, przy wykorzystaniu dodatkowych dwóch losowych znaków jako soli (ang. *salt*), przetwarza hasło na ciąg 11 znaków i zapisuje z solą na początku, np. `FbJAZZqjwCf8k` czy `RYtr5Q41Y871s`. Sól wprowadza dodatkowe mieszanie na 4096 sposobów, żeby utrudnić łamanie hasła przy pomocy specjalizowanych układów scalonych[26], a także, dzięki temu że nie widać, że mamy dwa takie same hasła (jak w przykładzie w poprzednim zdaniu), utrudnia skorzystanie z ataku z wykorzystaniem przygotowanych uprzednio tablic z zaszyfrowanymi hasłami (tzw. „rainbow tables").

Funkcja skrótu (mieszająca, hash) to funkcja, która jednoznacznie przypisuje dowolnie

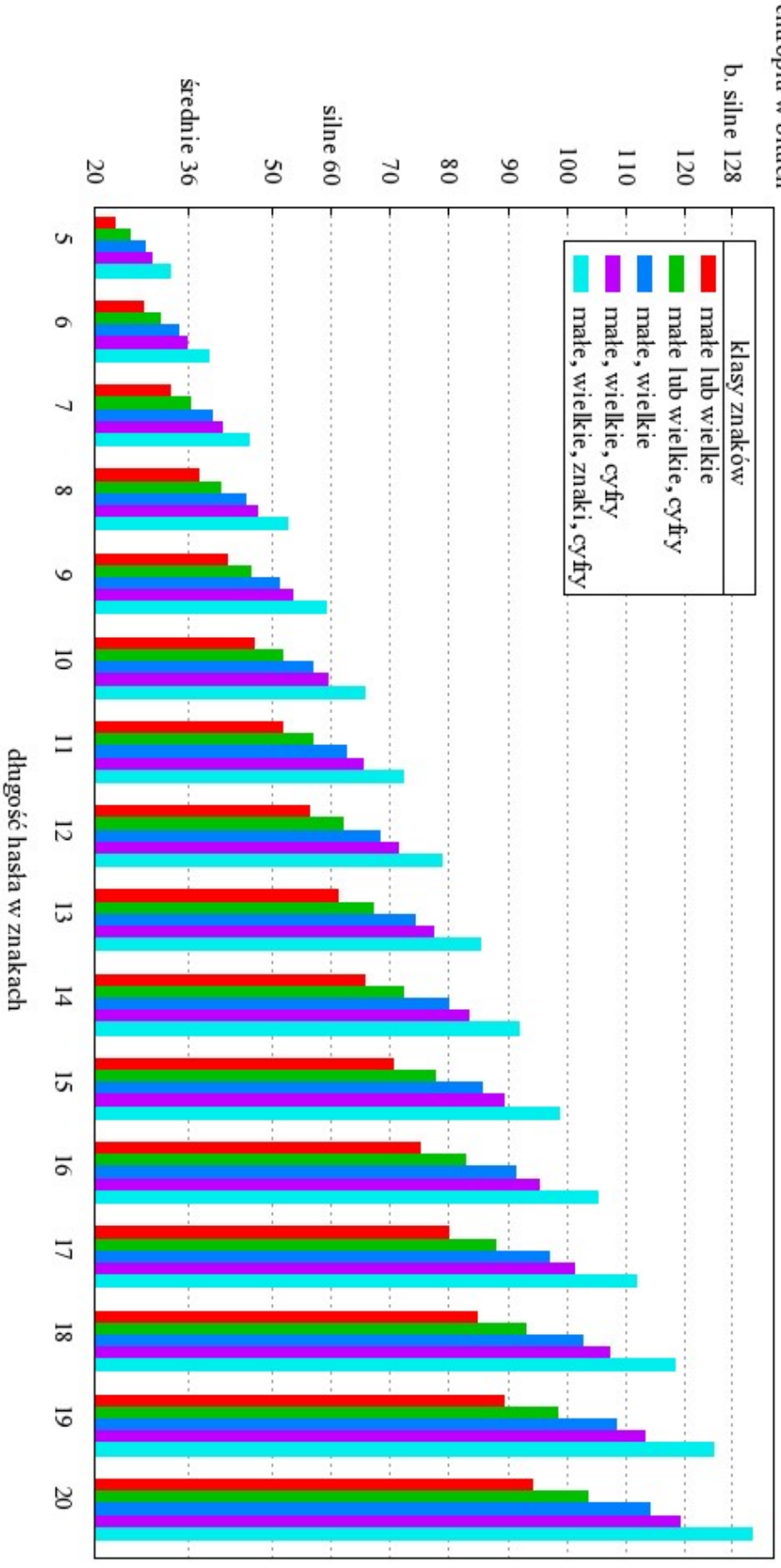
---

[23] Liczba 26 występuje tu zupełnie przypadkowo.

[24] Niestety, w wielu systemach to założenie nie jest spełnione, wykradzenie pliku z hasłami daje atakującemu natychmiastowy dostęp do naszych danych. Nie będziemy rozważać takich systemów.

[25] S. Singh, *Księga szyfrów*, Albatros, Warszawa, 2001, s. 266-269.

[26] B. Schneier, *Kryptografia...*, op. cit., s. 377.



Rysunek 3.1. Teoretyczna entropia hasła w zależności od użytych znaków i długości hasła.

Źródło: opracowanie własne.

długim ciągom znaków (wiadomościom) inną, krótszą, zwykle o stałej długości, wartość, zwaną skrótem wiadomości. W zastosowaniach kryptograficznych korzysta się z bezpiecznych funkcji skrótu (i tylko o takich będziemy mówić), które dodatkowo spełniają kilka warunków:

— niewielkie zmiany w danych wejściowych powodują znaczne zmiany w wyniku (w praktyce przyjmuje się, że zmiana jednego bitu wiadomości powinna zmienić przynajmniej połowę bitów wyniku),
— brak możliwości wnioskowania o danych wejściowych ze skrótu wiadomości,
— brak praktycznej możliwości wygenerowania wiadomości dającej zadany skrót,
— bardzo trudno znaleźć kolizje, tj. inny ciąg danych wejściowych, które zostaną przetworzone na taki sam wynik.

Funkcje skrótu (jak MD5 czy SHA) mają tę przewagę nad standardową funkcją `crypt()`, że nie ograniczają możliwej długości haseł do 8 znaków, a przecież im dłuższe hasło, tym trudniej je złamać.

Inne systemy (np. Windows NT) korzystają z innych algorytmów funkcji szyfrujących i skrótu, czasem gorszych (LM hash), ale ogólna zasada pozostaje ta sama.

Wróćmy do łamania haseł. Łamanie haseł jest stosowane wtedy, gdy atakujący w jakiś sposób wszedł w posiadanie zaszyfrowanego hasła, najczęściej w postaci pliku z hasłami. Przykładowy kawałek takiego pliku z systemu UNIX:

```
login1:ZJ9bvbH5yDZGk:0::60:7:365:15000:
login2:QXTXIdpfyAjAY:0::60:7:365:15000:
login3:bx8DSernZBpdo:0::60:7:365:15000:
```

W starszych systemach zaszyfrowane hasła były przechowywane w tym samym pliku, co nazwiska użytkowników, który był możliwy do odczytu dla wszystkich. W nowszych plik został rozdzielony na dwa, ten z hasłami do odczytu tylko przez administratora. Ale i tak w posiadanie zaszyfrowanych haseł można wejść na różne sposoby, niekoniecznie przez uzyskanie uprawnień administratora: czy to przez znalezienie takiego pliku w kopii zapasowej, czy błąd człowieka umożliwił odczyt takiego pliku, czy nawet jakiś program przerwał działanie z powodu przepełnienia bufora (ang. *buffer overflow*) i w ten sposób odkrył kawałek pamięci z hasłem. Ale nawet jeżeli ktoś uzyskał uprawnienia administratora, to ciągle może chcieć złamać hasła użytkowników. Zwykle bowiem bywa tak, że użytkownicy używają tych samych haseł na różnych systemach (choć nie powinni, rzecz

jasna), więc udana próba złamania hasła z jednego systemu może pozwolić na penetrację innych systemów.

W przypadku posiadania zaszyfrowanego hasła problem łamania hasła zdecydowanie się upraszcza – atakujący może zaprząc pełną moc komputera do sprawdzania kolejnych haseł, tzw. atak wyczerpujący[27] (ang. *brute-force*). W obecnych czasach możemy przyjąć, że na jednym komputerze z procesorem dwurdzeniowym o prędkości rzędu 3 GHz da się sprawdzać ok. 8 milionów haseł na sekundę, czemu nasze 6-znakowe hasło złożone z małych liter i cyfr będzie się opierać 4,5 minuty, a to, które wymagało 6 lat zgadywania, zostanie znalezione w 2 godziny. A pracę można rozłożyć na kilka, a nawet na kilkaset komputerów...

Sytuacja jednak jest jeszcze bardziej pesymistyczna. Atakujący nie sprawdza wszystkich teoretycznych haseł po kolei, tylko zaczyna od tych najbardziej prawdopodobnych. Czyli tak samo, jak w przypadku zgadywania, najpierw słownikowe, a potem ich różne przekształcenia[28].

Przy sprawdzaniu siły hasła należałoby jeszcze uwzględnić problem zwiększonego prawdopodobieństwa występowania znaków w parach, np. w języku polskim ponad 84 razy bardziej prawdopodobne jest, że po literze „o" wystąpi litera „d", niż że wystąpi drugie „o" (por. tabela na s. 39). Niektóre programy łamiące hasła uwzględniają te prawdopodobieństwa i najpierw szukają wśród haseł z takimi dwuznakami.

Cztery i pół minuty na przeszukanie całej przestrzeni haseł 6-znakowych opartych na małych literach i cyfrach to bardzo mało. Nawet po dodaniu wielkich liter dwie godziny to jest wciąż mało, zwłaszcza w obliczu możliwości zrównoleglenia lub rozproszenia prac. Stosowanie znaków specjalnych niewiele polepsza sytuację, obliczenia na jednym komputerze wydłużają się do 25 godzin. To ciągle za mało, zwłaszcza jeśli weźmiemy pod uwagę statystyczne zmniejszenie o połowę czasu przeszukiwania[29] związane z paradoksem dnia urodzin[30]. Wniosek z tego płynie tylko jeden: hasła muszą być dłuższe.

Wydłużmy zatem hasła do 8 znaków, maksymalnej długości w przypadku systemów

---

[27] B. Schneier, *Kryptografia...*, op. cit., s. 207-222.

[28] ibidem, s. 232-233.

[29] ibidem, s. 225-226.

[30] *Ile osób musi być w grupie, żeby prawdopodobieństwo, że dwie z nich mają urodziny w tym samym dniu, było większe od 50%?* Odpowiedź jest zaskakująco niska: tylko 23 osoby. W kryptografii ma to znaczenie przy funkcjach mieszających, w których inherentnie występują kolizje, gdyż statystycznie wystarczy przeszukać tylko połowę przestrzeni kluczy, żeby znaleźć prawidłowy odcisk.

Unix korzystających ze standardowej funkcji `crypt()`[31]. Dla naszego początkowego zestawu małych liter i cyfr otrzymujemy $(26+10)^8 = 2821109907456$ możliwych haseł, entropia zwiększa się do $8 \log_2 36 \approx 41$ bitów, a czas przeszukania całej przestrzeni zwiększa się do czterech dni. Dla maksymalnego praktycznego zestawu znaków, czyli: wielkie litery, małe litery, cyfry, znaki specjalne, w sumie 95 znaków, mamy entropię hasła $8 \log_2 95 \approx 52$ bitów, a teoretyczną wielkość przestrzeni haseł $95^8 = 6634204312890625$, przeszukanie której zajmie włamywaczowi z jednym komputerem ponad 26 lat.

Ludzki mózg źle sobie radzi z losowymi ciągami liter. Łatwiej jest mu zapamiętać długi wiersz niż losowy ciąg znaków, nawet jeśli byłoby ich 100 razy mniej niż wyrazów w wierszu. Hasła nie są żadnym wyjątkiem. Może zatem zamiast pilnować, żeby hasła były bardzo skomplikowane, należałoby pozwolić na tworzenie długich haseł, np. całych zdań? Łatwo policzyć: hasło 12-znakowe to entropia 56 bitów i 378 lat szukania, a hasło 17-znakowe, które ciągle nie jest jeszcze bardzo długie, utworzone tylko z małych liter, daje entropię $17 \log_2 26 \approx 80$ bitów, zaś przeszukanie przestrzeni haseł zajmie tyle, ile istnieje Ziemia.

To już coś. Choć niedawno rosyjska firma Elcomsoft ogłosiła[32], że znalazła sposób na wykorzystanie kart graficznych Nvidia do równoległego wyszukiwania przestrzeni kluczy z prędkością około 200 mln haseł na sekundę, czyli 25 razy szybciej, niż zakładaliśmy wyżej. Czasy przeszukiwań zatem drastycznie się zmniejszają: z 4,5 min do 11 sekund, z dwóch godzin do niecałych dwóch minut, z czterech dni do czterech godzin. Ale w przypadku dłuższych haseł jest lepiej: czasy spadają z 26 lat do 13 miesięcy, z 378 lat do 15 lat, z 4,5 miliarda lat do 180 milionów lat. Nie należy zapominać, że powyższe szacunki dotyczą obliczeń przeprowadzanych na jednym komputerze[33], a przecież można je łatwo zrównoleglić na tysiące, a nawet setki tysięcy maszyn[34]. Trzeba także pamiętać

---

[31] Hasła dłuższe niż osiem znaków muszą być „przechowywane" przy pomocy innych mechanizmów, np. funkcji skrótu MD5.

[32] *Elcomsoft files patent for revolutionary technique to recover lost passwords quickly* [online]. Elcomsoft, 2007-10-22 [dostęp: 2007-11-28]. Adres: `http://www.elcomsoft.com/EDPR/gpu_en.pdf`

[33] Nick Breese, pracownik amerykańskiej firmy Security-assessments.com, przeprowadził eksperyment z wykorzystaniem konsoli do gier PlayStation3, wyposażonej w osiem procesorów. Okazało się, oczywiście, że łamie hasła jeszcze szybciej, z prędkością 1,4 miliarda na sekundę...

[34] W 2005 r. holenderska policja zatrzymała ludzi odpowiedzialnych za budowę i kontrolowanie sieci 100 tys. komputerów zombie.

o prawie Moore'a o podwajaniu się mocy obliczeniowej komputerów co dwa lata[35]. Żeby zabezpieczyć się przed takimi zagrożeniami, należy korzystać z haseł 12-znakowych przy wykorzystaniu wszystkich klas znaków, lub przynajmniej 16-znakowych, jeśli będziemy korzystać tylko z małych liter.

Ciągle zostaje też problem zmniejszenia entropii takich haseł przez zakłócenie pełnej losowości wyboru – niektóre litery będą występować częściej niż inne, podobnie w przypadku dwuznaków. Eksperci zalecają, żeby nie liczyć więcej niż 3 bity entropii na znak, a niektórzy wręcz się posuwają do 1,5 bita na znak (ponad 3 razy mniej, niż wynika z teorii).

Mając powyższą wiedzę, możemy w końcu odpowiedzieć na pytanie, po co w ogóle wyliczamy entropię hasła. Otóż funkcje szyfrujące oraz skrótu dają nam w wyniku k bitów zaszyfrowanego (lub skróconego) komunikatu. W przypadku `crypt()` są to 64 bity, dla MD5 – 128 bitów. Z pesymistycznego twierdzenia Shannona *każdy idealnie bezpieczny system musi zapewniać entropię klucza większą od entropii komunikatu*[36] wynika, że aby uzyskać idealną tajność, musimy użyć klucza równie długiego, jak przesyłana wiadomość[37]. Zatem im entropia hasła jest bliższa 64/128 bitów, tym hasło jest lepsze. Przy okazji im większa entropia, tym dłużej trzeba szukać hasła.

3.2. Domyślne ograniczenia haseł w wybranych systemach

Za dane przechowywane w systemie odpowiada właściciel tych danych, więc to on powinien ustalać, w jaki sposób mają być zabezpieczone. Zazwyczaj uprawnienia odnośnie ustalania polityki bezpieczeństwa delegowane są na administratora systemu. Dość często administrator zostawia ten problem w ustawieniach domyślnych.

Systemy informatyczne Szkoły Głównej Handlowej od samego początku były zdywersyfikowane. Na początku lat 90. jako podstawowy serwer plików i serwer dostępowy służył Netware 3.10, a maszyna z Solarisem 2.3 była serwerem poczty. Na przestrzeni lat Novell oddał pole na rzecz serwerów Windows, które obecnie są podstawowym systemem

[35] B. Schneier, *Kryptografia...*, op. cit., s. 209

[36] C. Shannon, *A mathematical...*, op. cit., s. 623-656.

[37] Na tym opiera się idealny, nie do złamania szyfr Vernama wymyślony w 1917 r., którego skuteczność Shannon udowodnił w 1949 r.

dostępowym na terenie uczelni, zaś maszyny z różnymi wersjami UNIX-a (FreeBSD oraz Solaris) umocniły się w zastosowaniach internetowych i pocztowych.

Od samego początku każdy z tych systemów miał swoje oddzielne bazy użytkowników, własne metody uwierzytelniania i zarządzania użytkownikami oraz – co z punktu widzenia klientów usług najważniejsze – odmienne procedury zmiany haseł. Ta ostatnia kwestia od samego początku przysparzała najwięcej kłopotów.

Dyskomfort klientów spowodowany tymi problemami wymusił stworzenie rozwiązania, które pozwala na łatwą zmianę hasła na wszystkich systemach – zarówno samodzielnie przez użytkownika, jak i użytkownikowi przez obsługę.

Mimo heterogeniczności środowiska dobra synchronizacja informacji o użytkownikach pomiędzy systemami umożliwiła założenie, że dany użytkownik posiada na każdym z systemów konto o takim samym identyfikatorze.

3.2.1. UNIX

Konta i hasła użytkowników trzymane były w plikach systemowych `/etc/passwd` i `/etc/shadow`. Przy istniejącej wielkości systemu (prawie 40 tys. klientów) czasem dochodziło do blokowania jednoczesnych zapisów, a w warunkach wysokiego obciążenia doprowadzało wręcz do uszkodzenia listy użytkowników. Dodatkowo system zmiany haseł nie był zbyt wygodny dla obsługi (działał w powłoce Uniksa, nie pozwalał na wyszukiwanie użytkowników).

Na początku 2005 r. użytkownicy UNIX-a zostali zmigrowani do LDAP-a, co wyeliminowało problemy z blokowaniem przy jednoczesnych zapisach i jednocześnie umożliwiło tworzenie aplikacji zarządzających danymi użytkowników z poziomu WWW[38)].

W systemie UNIX/Solaris domyślnie zasady dotyczące tworzenia przez użytkowników haseł są następujące:

— hasło musi mieć od 6 do 8 znaków,

— hasło musi zawierać przynajmniej dwie litery i jedną cyfrę lub znak specjalny,

— hasło nie może być oparte na loginie,

— hasło musi się różnić od poprzedniego przynajmniej trzema znakami.

Entropia hasła spełniającego minimalne wymagania to 31 bitów, czyli niewiele. Domyślnie nie są ustawiane terminy ważności hasła ani konta. Bez dodatkowych modułów nie

[38)] Czego przykładem jest obecna Książka Adresowa SGH `http://akson.sgh.waw.pl/ksiazka/`

jest możliwe ustawienie żadnych obostrzeń dotyczących haseł (z wyjątkiem minimalnej długości hasła).

3.2.2. Windows

Konta w systemie Windows NT były trzymane w plikach systemowych na kontrolerach domeny. Użytkownicy mogli zmieniać swoje hasła w standardowy dla Windows sposób („Zmień hasło" po Ctrl-Alt-Delete), ale tylko po zalogowaniu na komputerze włączonym do domeny, co nie zawsze było możliwe, np. przy korzystaniu z prywatnych komputerów przenośnych. Obsługa sieci mogła zmieniać hasła użytkownikom przy pomocy programu usermgr. Na początku 2006 r. został wdrożony Windows 2000 i system ActiveDirectory oparty na LDAP. Użytkownicy mogli zmieniać hasło jak dotychczas, natomiast obsługa sieci musiała korzystać z nowego programu: AdminTool.

Hasła użytkowników w domenie Windows (Active Directory) domyślnie nie podlegają żadnym ograniczeniom. Administrator musi samodzielnie włączyć poniższe ograniczenia:

Historia haseł pamięta wybraną (od 0 do 24) liczbę haseł użytkownika, zabraniając mu korzystania zbyt często z tego samego hasła.

Wiek hasła określa, jak długo można korzystać z danego hasła, innymi słowy, jak często trzeba je zmieniać.

Minimalny wiek hasła określa, po jakim czasie można zmienić hasło po raz kolejny. Ma to w teorii współgrać z historią haseł.

Minimalna długość hasła.

Wymagane złożone hasła włącza ograniczenia dotyczące wyglądu haseł, które muszą spełniać kilka warunków:

- min. 6 znaków,
- hasło musi zawierać przynajmniej trzy kategorie z następujących pięciu:
  1. wielkie litery (A-Z),
  2. małe litery (a-z),
  3. cyfry (0-9),
  4. pozostałe znaki ASCII (np. #, !, $, %),
  5. znaki Unicode,
- hasło może zawierać najwyżej dwa znaki z loginu,
- hasło nie może zawierać wyrazów, które składają się na nazwę użytkownika.

Przy braku działania ze strony administratora hasła mogą być dowolne, zatem także bar-

dzo słabe. Zalecane jest przynajmniej włączenie złożonych haseł – przy takim ustawieniu najsłabsze hasło spełniające te wymagania ma entropię 35 bitów – oraz wymuszanie zmian haseł co dwa miesiące.

3.2.3. Netware

W wersji Netware 3.xx dane o kontach i hasłach przechowywane były w strukturze bindery, użytkownicy mogli zmieniać swoje hasła DOS-owym programem setpass (oczywiście tylko wtedy, gdy korzystali z klienta VLM). Ze względu na zbyt duże uprawnienia wymagane do zmiany haseł innym użytkownikom przy pomocy programu setpass, obsługa sieci dostała jeszcze inne narzędzie: program syscon. Z powodu ograniczeń bindery na liczbę użytkowników, w 1999 r. nastąpiła migracja do NDS[39)], który także jest oparty na LDAP. Użytkownicy w dalszym ciągu korzystali z setpass, a obsługa sieci dostała kolejne narzędzie: chpass. Z czasem pojawiły się w eDirectory inne rodzaje haseł i żeby je zintegrować, wprowadzono hasło uniwersalne, do którego nie wystarczała dotychczasowa aplikacja chpass. Została wprowadzona zmiana haseł przez WWW, także dla obsługi sieci, a administrator mógł korzystać z programu NWadmn (NetWare Administrator), w wyglądzie podobnego do usermgr/AdminTool dla Windows.

W systemie Netware, podobnie jak w Windows, domyślnie nie ma żadnych ograniczeń co do haseł. Administrator ma możliwość włączenia różnego rodzaju ograniczeń, acz sytuacja jest trochę bardziej skomplikowana ze względu na występowanie kilku rodzajów haseł. Na domyślne hasło do eDirectory możemy założyć ograniczenia dotyczące długości hasła, jego ważności, historii haseł[40)] oraz liczbę tzw. „grace login". Po włączeniu haseł uniwersalnych zyskujemy możliwość tworzenia dość zaawansowanych dodatkowych reguł (polityk) dotyczących haseł: liczbę pamiętanych haseł w historii (od 1 do 255) lub czas ich pamiętania (do roku), powtarzające się znaki, minimalna i maksymalna liczba małych i wielkich liter oraz cyfr i znaków specjalnych w haśle (każda klasa oddzielnie), a także możliwość odzyskania hasła. Szczegóły można znaleźć w broszurze *Novell Password Management Administration Guide*[41)].

[39)] NetWare Directory Services, potem zwane Novell Directory Services, a teraz znane jako eDirectory.

[40)] Niestety, bez limitu. Użytkownik nie mógłby już nigdy więcej mieć drugi raz takiego samego hasła. Rzadko włączane.

[41)] Adres: `http://www.novell.com/documentation/password_management31/index.html`

4. System zmiany haseł

Informacja to przekazanie różnorodności.
W. Ross Ashby[42)]

Należy zwrócić uwagę na to, że wszystkie utrudnienia dotyczące konstrukcji haseł, które zostały zaprezentowane w rozdziale 3.2, to nic innego, jak próba narzucenia użytkownikowi wymagania, aby jego hasło miało większą entropię. I minimalna długość hasła, i obowiązek stosowania różnych klas liter to zgrubne sposoby na zwiększenie entropii hasła. Znacznie lepiej jest po prostu policzyć entropię hasła, być może z kilkoma oczywistymi ograniczeniami, niż stosować takie sztuczne wymagania. Po co żądać od użytkownika, który wpisał 16-znakowe hasło, żeby były w nim jeszcze duże litery i znaki specjalne? Długość hasła wystarczająco zwiększa entropię, a co za tym idzie, wystarczająco utrudnia jego złamanie.

4.1. Założenia programowe

Celem systemu jest dostarczenie możliwości zmiany hasła *w jednym miejscu* dla wszystkich systemów (UNIX, Windows, Netware), zarówno samodzielnie przez użytkownika, jak i użytkownikom przez obsługę. Zakres systemu obejmuje niewielki wycinek zarządzania kontem, czyli zmianę hasła.

Klientów systemu można podzielić na dwie kategorie: obsługę sieci (ok. 20 osób) i użytkowników sieci (ok. 13 tys. osób[43)], por. rys. 5.2). Użytkownicy sieci będą zmieniać swoje hasła, zaś obsługa sieci będzie zmieniać hasła użytkowników sieci. Istnieje jeszcze (choć bardzo rzadko występuje) administrator systemu (jedna osoba), który nadaje uprawnienia obsłudze systemu. System uprawnień zapewnia podstawową ziarnistość uprawnień[44)].

[42)] W. R. Ashby, *Wstęp do cybernetyki*, PWN, Warszawa, 1994, s. 181-191.

[43)] Kont w zasadzie jest ok. 42 tys., ale 2/3 z nich nie jest aktywnie używane.

[44)] D. Pipkin, *Bezpieczeństwo informacji*, op. cit., s. 167.

Zgodnie z metodyką programowania[45)], na początek należy określić wymagania funkcjonalne:

1. Klient (zarówno użytkownik, jak i obsługa sieci) musi się prawidłowo uwierzytelnić (czyli podać swój prawidłowy login i hasło).
2. Obsługa sieci musi mieć możliwość wyszukania użytkownika według różnych kryteriów (login, nazwisko).
3. Obsługa sieci nie może sama wybierać haseł dla użytkowników (doświadczenie uczy, że wybieraliby przewidywalnie proste, poza tym spowolniłoby to ich pracę).
4. Obsługa sieci musi mieć opisane uprawnienia (kto i komu może zmieniać hasło).
5. Klient musi wybrać systemy, na których chce zmienić hasło.
6. System musi zapisywać dziennik z pracy obsługi sieci (kto, komu i kiedy zmieniał hasło).
7. System musi zweryfikować hasło użytkownika pod kątem bezpieczeństwa.
8. System pokazuje umowną siłę hasła użytkownika i nie akceptuje zbyt słabych haseł.
9. Klient musi otrzymać komunikat informujący o poprawnej zmianie hasła lub komunikat błędu wyjaśniający (lub choć wskazujący) przyczyny błędu.
10. System musi być bezpieczny[46)], bez możliwości podsłuchu lub ujawnienia hasła.
11. System musi być zgodny ze standardem systemu otwartego[48)].

Wymagania niefunkcjonalne:

1. Wymagania dotyczące sprzętu: korzystamy z istniejącej infrastruktury UNIX[49)], system będzie miał niewielkie wymagania.
2. Wymagania dotyczące przepustowości łącz: ze względu na usługowy i bardzo prosty charakter systemu, nie powinno być problemów z korzystaniem z systemu na dowolnym łączu TCP/IP[50)].
3. Wymagania dotyczące technologii: język programowania PHP, bazy danych z użytkow-

---

[45)] J. Goliński, *Wybrane problemy organizacji projektowania i programowania komputerów*, Pracownia Poligraficzna Wydawnictw Uczelnianych SGPiS, Warszawa, 1978, s. 57-61.

[46)] Najlepiej przeprowadzić analizę podatności na ryzyko zgodnie metodyką prezentowaną przez Białasa[47)]

[47)] A. Białas, *Bezpieczeństwo informacji...*, op. cit., s. 261-263.

[48)] H. Egeman, Zagadnienia spójności systemów informatycznych zarządzania. Oficyna Wydawnicza SGH, 1993, s. 74.

[49)] ibidem, s. 74.

[50)] ibidem, s. 74.

nikami LDAP, baza danych na dziennik systemu PostgreSQL (zgodny ze standardem SQL[51)]), serwer aplikacji: Apache z modułem PHP.

4. Dane będą pobierane z serwera LDAP (ldap.sgh.waw.pl); dane będą zapisywane na serwerach: LDAP (ldap.sgh.waw.pl), Active Directory (aqua.sgh.waw.pl) oraz eDirectory (hikari.sgh.waw.pl). Dodatkowo będzie tworzony dziennik systemu w bazie danych PostgreSQL. Nie przewiduje się żadnych importów ani eksportów danych.
5. Ogólna charakterystyka systemu: cienki klient (WWW), aplikacja na serwerze.

4.2. Model danych

System w zasadzie nie korzysta z własnych danych, tylko manipuluje danymi zawartymi w innych systemach. W związku z tym zostaną opisane tylko fragmenty wykorzystywanych encji.

Jako podstawowe źródło danych został uznany serwer LDAP, w którym uwierzytelniają się użytkownicy UNIX-a. Przykład danych, z których korzystamy:

```
dn: uid=jagol,ou=People,dc=sgh,dc=waw,dc=pl
userPassword: {crypt}jakieśhasło
shadowMax: 60
shadowWarning: 7
shadowLastChange: 13814
personalTitle: prof. dr hab.
givenname: Jan
sn: GOLIŃSKI
employeetype: Pracownik dydaktyczny
departmentNumber: cn=Profesor zwyczajny@ou=Zakład Projektowania Systemów,
 ou=Katedra Informatyki Gospodarczej,ou=Kolegium Analiz
 Ekonomicznych,ou=SGH
departmentNumber: cn=Kierownik@ou=Katedra Informatyki Gospodarczej,
 ou=Kolegium Analiz Ekonomicznych,ou=SGH
```

Na pozostałych systemach tylko zmieniamy hasło, więc korzystamy z bardzo ograniczonego zestawu danych, których nawet nie czytamy, tylko od razu zapisujemy.

[51)] ibidem, s. 74.

```
dn: cn=jagol,ou=USERS,o=SGH
userPassword: jakieśhasło                          Netware
passwordExpirationTime: 19920102000000Z

dn: CN=jagol,CN=Users,DC=nt,DC=sgh,DC=waw,DC=pl
pwdLastSet: 128133316194375000                     Windows
unicodePwd: "jakieśhasło"
```

Dziennik wydarzeń trzymany jest w postaci pojedynczej, niepowiązanej tabeli w bazie SQL. Opis pól tabeli:

kto identyfikator obsługującego, który przeprowadził transakcję (text)

kiedy data transakcji w formacie UNIX timestamp (integer)

komu identyfikator użytkownika, któremu zmieniono hasło

proba lista (mapa bitowa: 1 – UNIX, 2 – Windows, 4 – Novell) systemów, na których próbowano zmienić hasło

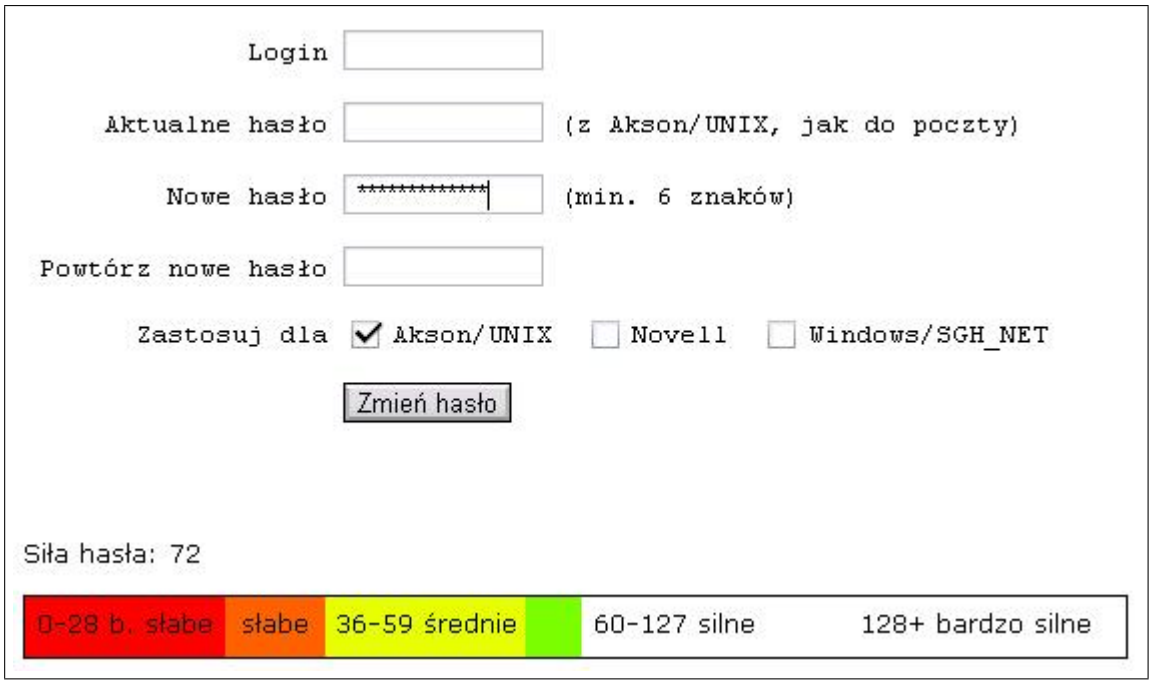
bledy lista (mapa bitowa, jw.) systemów, na których nie udało się zmienić hasła

Ze względu na prostotę obsługi dla użytkowników, system dla obsługi w ogóle nie jest widziany przez użytkowników – pomimo że wewnątrz są te same funkcje i interfejsy.

Użytkownicy dostają możliwie prosty interfejs graficzny (rys. 4.1) – tylko wymagane dane, kilka podpowiedzi, co gdzie wpisać. Dodatkowo, jeżeli użytkownik już był zalogowany w systemie, nie będą pokazywane pierwsze dwa pytania – w końcu jego login i hasło już znamy. W trakcie wpisywania hasła pod spodem pojawi się wizualnie oznaczona siła hasła, zaś samo hasło użytkownik będzie musiał przepisać dwukrotnie w celu wyeliminowania pomyłek przy wpisywaniu, co by skutkowało brakiem możliwości skorzystania z takiego hasła później (choć nie broni to przed wpisaniem hasła na klawiaturze o nietypowym układzie).

Można zwrócić uwagę, że choć akceptowane są hasła średnie i silniejsze, to dopiero hasła silne i bardzo silne są oznaczane kolorem zielonym, przekazującym wrażenie bezpieczeństwa w rozumieniu infologicznym[52)]. Dodatkowo użytkownicy korzystający z silnych i bardzo silnych haseł są wynagradzani przez wydłużenie okresu ważności hasła z 60 dni

[52)] W. Flakiewicz, *Pojęcie informacji w technologii multimedialnej*, Oficyna Wydawnicza SGH, Warszawa, 2005, s. 63.



Rysunek 4.1. Ekran dialogowy do zmiany hasła przez użytkownika

do odpowiednio 160 i 360 dni[53)]. Należy przy tym podkreślić, że hasła jako takie nie tracą ważności, tj. jeżeli użytkownik pamięta swoje hasło, to może przy jego pomocy zmienić sobie hasło na nowe w dowolnym momencie, nawet po 5 latach – ważność hasła nie jest problemem krytycznym wymagającym interwencji obsługi, co zmniejsza liczbę interwencji. Z przyjętych założeń programowych wynika także, że najważniejsze jest hasło do UNIX-a, gdyż znając je, można zmienić hasło na innych systemach.

4.3. Algorytm analizy siły hasła

Analizę siły hasła oparto na entropii z kilkoma negatywnymi modyfikatorami, które uniemożliwiają korzystanie ze zbyt trywialnych haseł.

Ze względu na graficzno-kolorystyczną wizualizację siły hasła wynik jest obliczany także dla haseł, które nie zostaną zaakceptowane. Do takich haseł należą hasła krótsze niż 6 znaków[54)], puste hasła i hasła oparte wprost na loginie (login wprost jest z hasła usuwany i dopiero jest sprawdzana jego siła); dodatkowo ze względu na korzystanie

[53)] To działa! Przez pierwszy miesiąc obowiązywania nowego systemu hasła silne utworzyło sobie 700 osób, a bardzo silne – 44 osoby z ogólnej liczby 5300 zmian.

[54)] Po zmianie systemu oceny jakości hasła na oparty na entropii w zasadzie nie ma takiej potrzeby – krótsze hasła i tak nie będą miały odpowiedniej entropii (32 bity wobec wymaganych 36).

przez użytkowników z różnych systemów poza naszą kontrolą, z ustawionymi różnymi kodowaniami znaków, nie możemy pozwolić na hasła zawierające znaki spoza zakresu ASCII[55)56)]. Nie pozwalamy też wprost na hasło, które jest użyte jako przykładowe na stronie do zmiany haseł – lepiej dmuchać na zimne. Te podstawe ograniczenia przedstawia następujący kod[57)] PHP:

```
if (strlen($newpass) < 6)
    $stop = "nowe hasło jest za krótkie (min. 6 znaków)";
if ($newpass == "" || $newpass2 == "")
    $stop = "puste nowe hasło";
$newpass = str_replace($login, '', $newpass);
if ($newpass != $newpass2 && $newpass == "")
   $stop = "hasło w całości oparte na loginie";
$newpass2 = str_replace($login, '', $newpass2);
$strlen = strlen($newpass);
if (ereg('[^ -~]', $newpass))
    $stop = "znaki spoza ASCII (np. polskie litery)";
if ($newpass == "Mdpm30l,pw!")
   $stop = "nie wolno korzystać z przykładowego hasła";
```

Następnie jest obliczana entropia hasła. Po wyszukaniu w haśle kolejnych klas znaków, odpowiednio: małe litery, wielkie litery, cyfry, odstęp z kropką i przecinkiem, znaki specjalne na cyfrach, pozostałe znaki specjalne i odpowiednim powiększeniu przestrzeni hasła, podstawiamy do wzoru (3.2).

[55)] Oryginalnie skrót od American Standard Code for Information Interchange, teraz najczęściej w znaczeniu systemu znaków, który używa 7 bitów na każdy znak, co pozwala na to, żeby każdej literze był przypisany oddzielny kod, w dodatku umożliwiając rozróżnienie wielkich liter od małych; niestety, nie obejmuje liter narodowych, jak polskie „ś" czy niemieckie „ß" – litery te trafiły do zestawu znaków dopiero wtedy, gdy zaczęto wykorzystywać ósmy bit, ale ponieważ to wykorzystanie nie było koordynowane, różne instytucje przypisywały tym samym kodom różne znaki.

[56)] T. Jennings, *An annotated history of some character codes or ASCII: American Standard Code for Information Infiltration*, 2004.

[57)] Wszystkie fragmenty programów w tej pracy są przepisane wprost z istniejących programów mojego autorstwa. Dla oszczędności miejsca pominąłem nawiasy klamrowe, bloki kodu są oznaczone wcięciami.

```
$space = 0;
if (ereg('[a-z]', $newpass)) $space += 26;
if (ereg('[A-Z]', $newpass)) $space += 26;
if (ereg('[0-9]', $newpass)) $space += 10;
if (ereg(' .,', $newpass)) $space += 3;
if (ereg('[!@#$%^&*()]',$newpass)) $space += 10;
if (ereg('[]=_+{}|~`/<>?;\':"\[\\-]', $newpass)) $space += 20;
$entropy1 = floor($strlen * log($space, 2));
```

W ten sposób otrzymaliśmy teoretyczną entropię hasła. Niestety, ludzie mają tendencję do tworzenia wymawialnych zbitek literowych, gdyż takie łatwiej zapamiętać. Korzystają z tego włamywacze, skorzysta z tego przedstawiany system przez wyliczenie entropii zmodyfikowanej o częstości występowania par znaków (entropia relatywna).

Wykorzystano tu tablicę częstości par znaków[58)] zmodyfikowaną w taki sposób, że częstości występowania liter diakrytycznych zostały dodane do częstości odpowiadających im liter łacińskich (wynikowa tabela częstości w załączniku 4.1). Przyjęto także założenie, że dla częstości występowania par liter w haśle wielkość liter nie ma znaczenia.

```
$entropy2 = 0;
$plower = strtolower($newpass);
$aidx = Get_Index($plower[0]);
for ($b = 1; $b < strlen($plower); $b++)
    $bidx = Get_Index($plower[$b]);
    $c = 1.0 - $Frequency_Table[$aidx * 27 + $bidx];
    $entropy2 += log($space) * $c * $c;
    $aidx = $bidx;
$entropy2 = round($entropy2);
```

Do dalszych obliczeń wykorzystano ważony wynik tych dwóch wyliczonych entropii, ich proporcje ustalono doświadczalnie przez wypróbowanie kilku haseł. Im większy wpływ na wynik ma entropia wyliczona z uwzględnieniem częstości par liter, tym hasło jest silniejsze i bliższe postulatowi ekspertów (por. na str. 26).

```
$score = round(0.8*$entropy1 + 0.2*$entropy2);
```

[58)] *Własności statystyczne języka polskiego – badania częstości występowania liter i par liter w tekstach polskich*

Tutaj można by zakończyć – ustalono już entropię hasła. Niestety, nie jest to takie proste. Z obserwacji wynika, że włamywacze najpierw próbują hasła, które są oparte na czymś, co wiedzą o użytkowniku. Podstawową taką informacją o użytkowniku jest jego login. Z tego powodu znacząco obniżono entropię hasła (choć to już nie będzie entropia per se, tylko umowny wskaźnik siły hasła), jeżeli hasło jest podobne do loginu użytkownika. Podobieństwo hasła do loginu obliczane jest przy pomocy dystansu Levenshteina, czy inaczej odległości edycyjnej. Odległość Levenshteina nazwano tak na cześć rosyjskiego uczonego Vladimira Levenshteina, który w 1965 r. pracował nad uogólnieniem odległości Hamminga[59] na ciągi znaków o różnej długości. Określił on trzy podstawowe operacje edycyjne: podmiana litery, usunięcie i wstawienie litery[61], zaś podobieństwo ciągów znaków określił przez liczbę operacji edycyjnych, którym trzeba poddać jeden ciąg, żeby przekształcić go w drugi. Dla przykładu „ona" jest odległe od „zła" o 2 operacje: „ona" → „oła" (zmiana „n" na „ł"), „oła" → „zła" (zmiana „o" na „z").

Przyjęto, że odległość mniejsza niż cztery oznacza hasło zbyt podobne do loginu, zatem jest penalizowane proporcjonalnie do podobieństwa. Analogicznie w przypadku podobieństwa nowego hasła do starego, choć tutaj nowe hasło jest określane jako bezwartościowe, jeśli różni się od starego w mniej niż trzech miejscach.

```
$leven1 = levenshtein($login, $newpass);
if ($leven1 < 4)
   $score -= (4-$leven1) * 5;
if (levenshtein($oldpass, $newpass) < 3)
   $score -= 50;
```

Niektóre osoby mogłyby się pokusić o utworzenie długiego hasła przez powtarzanie krótkich ciągów znaków. Niestety, włamywacze też o tym myślą, stąd nie należy pozwalać na takie hasła. W kolejnym bloku kodu nowe hasło jest dzielone kolejno na ciągi 2-, 3- i 4-literowe i penalizuje się powtarzanie tych krótkich ciągów w obrębie całego hasła.

---

[59] Nazwanej na cześć Richarda Hamminga, który jest bardziej znany ze swojej pracy[60] o kodach wykrywania błędów (ECC).

[60] R. Hamming, *Error Detecting and Error Correcting Codes*, Bell System Technical Journal, 1950, t. 29, s. 147-160.

[61] V. Levenshtein, *Binary codes capable of correcting deletions, insertions and reversals*, Doklady Akademia Nauk SSSR, 1965, t. 163(4), s. 845-848.

```
for ($i = 2; $i <= 4; $i++)
    $temp = str_split($newpass, $i);
    $score -= $i*(ceil($strlen / $i) - count(array_unique($temp)));
```

Ze względu na popularność klawiatur w układzie QWERTY nie należy pozwalać na tworzenie haseł opartych na prostych układach klawiaturowych, tj. haseł tworzonych przez wpisywanie kolejnych znaków z klawiatury. Przy okazji zabroniono tu też haseł alfabetycznych opartych na ciągach znaków w kolejności alfabetycznej. W obu przypadkach sprawdzane są ciągi w obu kierunkach, na klawiaturze z lewej do prawej i z prawej do lewej, a dla alfabetycznych w kolejności alfabetycznej i od końca.

```
$qwerty = "~1234567890-=qwertyuiop[]\asdfghjkl;'zxcvbnm,./"
        . "~!@#$%^&*()_+QWERTYUIOP{}|ASDFGHJKL:\"ZXCVBNM<>?"
        . "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";
for ($i = 0; $i < 3; $i++)
    foreach (str_split(substr($newpass,$i), 3) as $temp)
        if (strlen($temp)<3) continue;
        if (stripos($qwerty, $temp)!==FALSE)
           $score -= 3;
        if (stripos(strrev($qwerty), $temp)!==FALSE)
           $score -= 2;
```

Ostateczny wynik `$score` to ważona entropia hasła pomniejszona o wymienione sztuczne modyfikatory. Hasła bardzo słabe i słabe to takie, które zostały ocenione poniżej 36 (w zamyśle bitów entropii), hasła średnie to 36-59, silne – 60-127, a bardzo silne powyżej 127. Jest to powiązane z bitami entropii zaszyfrowanego hasła, o czym wspomniano przy pesymistycznym twierdzeniu Shannona w rozdziale 3.1.

Pokazywanie siły hasła po każdym znaku jest rozwiązane przy pomocy krótkiego kodu AJAX[62)], czyli Javascriptu, który po każdym naciśnięciu klawisza w polu nowego hasła formularza wysyła zawartość pola do skryptu na serwerze, który przetwarza otrzymywane dane korzystając z ww. funkcji i zwraca wynik liczbowy. Wynik ten jest umieszczany u klienta przy użyciu Javascriptu, zgodnie z filozofią działania AJAX – bez przeładowania całej strony. Przy okazji skrypt proporcjonalnie koloruje schematyczną tabelę z wynikami, jak widać na rys. 4.1 na s. 34.

[62)] Asynchronous JavaScript and XML.

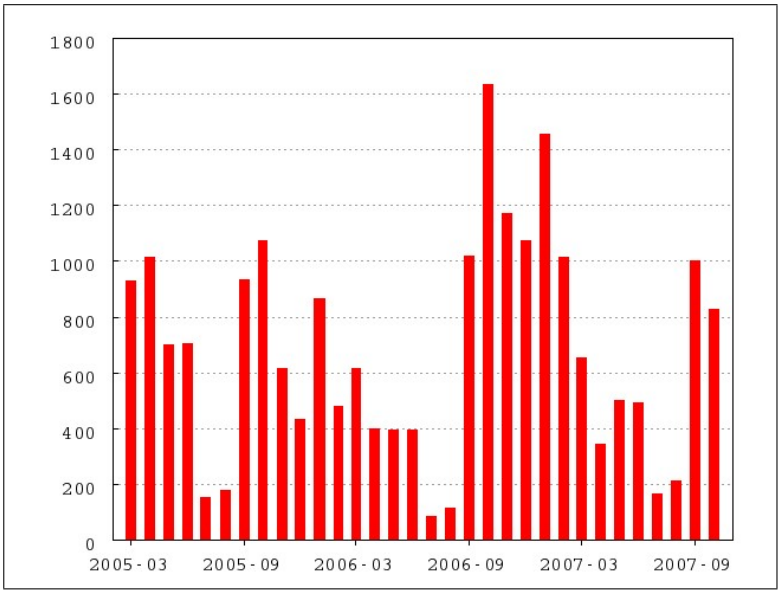
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	_
a	0,049	1,715	1,529	3,026	0,205	0,373	1,297	0,850	7,677	4,983	4,351	7,600	2,787	11,086	0,037	3,079	0,000	6,274	1,660	6,135	0,129	0,019	5,930	0,007	0,022	8,949	3,454
b	0,858	0,114	0,024	0,110	1,171	0,002	0,002	0,013	0,130	0,057	0,112	0,536	0,126	0,013	3,386	0,000	0,001	0,099	0,058	0,005	0,467	0,000	0,025	0,001	0,643	0,563	5,484
c	7,244	0,094	0,034	0,285	4,246	0,002	0,001	0,326	3,409	0,262	0,151	0,246	0,112	0,741	2,534	0,147	0,000	0,468	3,180	0,126	1,294	0,000	0,523	0,000	3,997	1,478	6,151
d	3,997	0,014	0,042	0,153	4,249	0,000	0,909	0,012	0,658	0,239	0,003	0,089	0,040	0,476	5,599	0,001	0,000	0,742	0,007	0,015	1,242	0,000	0,344	0,000	0,497	1,015	7,707
e	0,011	1,157	1,787	1,541	0,057	0,165	0,520	0,400	30,000	5,152	0,457	6,993	0,934	4,227	0,037	1,142	0,000	2,819	0,731	4,773	0,023	0,026	1,926	0,003	0,031	11,939	0,480
f	0,205	0,010	0,001	0,003	0,090	0,010	0,001	0,003	0,056	0,002	0,002	0,137	0,018	0,071	0,221	0,007	0,000	0,022	0,052	0,008	0,066	0,000	0,002	0,000	0,037	0,003	0,734
g	1,055	0,001	0,008	0,039	4,290	0,000	0,034	0,006	0,374	0,041	0,005	0,071	0,042	0,146	1,721	0,001	0,000	0,171	0,007	0,003	0,588	0,000	0,012	0,001	0,398	0,389	3,563
h	0,049	0,000	8,227	0,002	0,019	0,009	0,009	0,000	0,005	0,001	0,001	0,011	0,004	0,019	0,058	0,021	0,000	0,002	0,032	0,156	0,004	0,002	0,003	0,000	0,011	0,003	0,601
i	0,184	2,510	6,079	0,208	0,122	0,438	0,874	0,269	0,328	0,383	4,745	4,840	5,809	16,364	0,371	2,313	0,000	0,593	6,301	0,113	0,023	0,065	8,182	0,011	0,023	5,094	5,929
j	2,820	0,031	0,769	0,046	4,261	0,000	0,000	0,005	0,218	0,005	0,004	0,010	0,015	0,006	1,818	0,000	0,000	0,002	0,072	0,006	0,880	0,000	0,013	0,000	0,595	0,237	6,069
k	3,959	0,133	0,201	0,394	3,244	0,002	0,001	0,012	1,056	0,083	0,074	1,390	0,168	0,653	2,008	0,060	0,000	0,350	2,296	1,217	0,613	0,000	0,285	0,000	0,953	0,956	6,493
l	12,831	0,711	0,032	1,556	3,002	0,122	1,876	0,212	3,603	0,043	1,642	0,166	0,135	0,024	3,266	0,848	0,000	0,277	2,081	0,292	0,720	0,000	0,632	0,000	3,493	1,174	2,584
m	3,706	0,008	0,043	0,207	5,690	0,001	0,047	0,113	1,405	0,221	0,012	0,059	0,033	0,010	2,131	0,001	0,000	0,214	0,998	0,042	0,748	0,000	0,019	0,000	3,204	0,693	7,401
n	6,269	0,270	0,149	2,002	4,135	0,026	0,495	0,265	2,244	0,367	0,483	1,311	1,226	0,722	4,207	0,246	0,000	0,669	1,091	0,787	0,740	0,000	1,322	0,000	1,061	3,506	12,145
o	0,071	1,439	1,549	5,076	0,226	0,259	5,279	1,335	1,385	0,291	4,970	5,155	3,072	3,274	0,062	9,181	0,000	6,904	1,381	5,409	0,027	0,014	2,779	0,000	0,124	2,140	6,790
p	1,374	0,002	0,030	0,288	1,059	0,000	0,001	0,025	0,222	0,059	0,021	0,051	0,279	0,023	1,405	0,074	0,000	0,133	1,918	0,064	0,550	0,001	0,228	0,001	0,667	0,369	16,035
q	0,000	0,000	0,001	0,000	0,001	0,000	0,000	0,000	0,001	0,000	0,000	0,000	0,000	0,001	0,001	0,000	0,000	0,000	0,000	0,000	0,001	0,000	0,000	0,000	0,000	0,000	0,033
r	4,322	1,408	0,080	1,209	3,734	0,141	0,907	0,126	0,209	0,257	1,847	0,019	0,153	0,016	4,499	6,237	0,000	0,080	0,275	3,098	0,997	0,001	0,657	0,000	0,294	0,486	4,074
s	3,953	0,153	0,024	0,263	4,714	0,003	0,008	0,032	2,303	0,658	0,793	0,280	0,135	0,663	5,455	0,270	0,000	0,471	0,183	0,038	1,729	0,000	1,865	0,000	2,784	0,229	14,383
t	2,890	0,008	0,108	0,085	1,814	0,013	0,003	0,040	0,758	0,023	1,881	0,193	0,061	0,817	2,278	0,157	0,000	0,507	7,687	0,062	0,628	0,000	0,218	0,002	0,976	0,454	8,410
u	0,352	0,512	0,354	0,732	0,162	0,069	0,340	0,278	0,682	0,766	1,540	2,237	1,443	0,423	0,045	0,595	0,037	1,372	0,646	1,242	0,008	0,001	0,237	0,000	0,020	1,306	2,571
v	0,013	0,000	0,000	0,002	0,010	0,000	0,000	0,000	0,010	0,001	0,001	0,003	0,000	0,001	0,010	0,000	0,000	0,002	0,000	0,001	0,000	0,000	0,000	0,006	0,000	0,000	0,077
w	2,961	0,018	0,038	0,541	1,498	0,000	0,101	0,377	1,099	0,105	0,207	0,132	0,040	0,048	8,837	0,001	0,000	0,636	1,161	1,220	0,387	0,000	0,010	0,000	1,319	1,171	13,396
x	0,002	0,000	0,000	0,000	0,003	0,000	0,000	0,000	0,007	0,000	0,000	0,000	0,000	0,003	0,009	0,000	0,000	0,007	0,000	0,000	0,001	0,000	0,000	0,004	0,000	0,000	0,020
y	0,031	3,293	1,386	2,053	0,021	0,038	0,000	0,253	0,002	0,000	0,002	2,582	1,923	3,419	0,006	0,342	0,000	2,126	0,588	3,075	0,002	0,001	3,882	0,000	0,001	8,723	0,046
z	3,418	0,011	8,861	6,218	2,907	0,000	0,024	0,016	0,820	0,120	0,120	0,077	0,030	0,032	3,332	0,009	0,000	9,423	6,869	0,026	1,132	0,000	0,409	0,000	0,561	0,040	12,939
_	20,595	0,385	5,694	2,010	26,400	0,086	0,239	4,278	13,513	3,767	3,177	7,134	8,423	2,460	14,861	0,145	0,001	0,763	2,115	2,161	4,970	0,008	5,802	0,019	12,087	6,475	13,731

Tabela 4.1. Częstość w promilach występowania par liter (po konwersji do ASCII) w tekstach w języku polskim.
Źródło: opracowanie własne na podstawie *Własności statystyczne języka polskiego – badania częstości występowania liter i par liter w tekstach polskich.*

5. Koszty

5.1. Koszty zmiany hasła

Od uruchomienia systemu zintegrowanej zmiany haseł (rozdział 4) na początku marca 2005 r. do końca października 2007 r. zostało przeprowadzonych 21650 administracyjnych zmian hasła użytkownikom, czyli ok. 22 hasła dziennie. Po uwzględnieniu szczegółowych danych dziennych zostało 812 dni (odpadły święta i (rzadkie) dni, w których nikt nie potrzebował zmieniać sobie hasła), co daje 27 zmian haseł dziennie, także w soboty i niedziele, kiedy z usługi korzystają studenci zaoczni i niektórzy pracownicy.



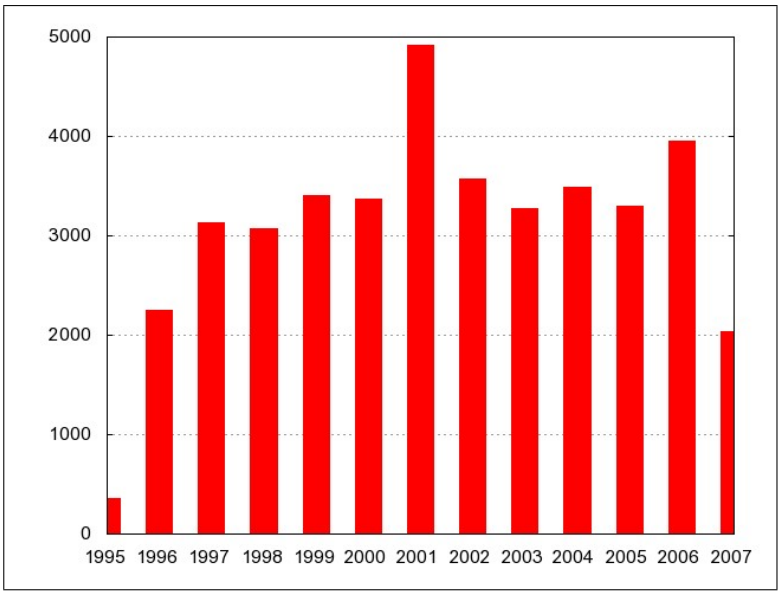
Rysunek 5.1. Statystyki zmian haseł przez obsługę miesięcznie.
Źródło: opracowanie własne.

Jedna zmiana hasła w zależności od sprawności operatora i użytkownika szacunkowo zajmuje między 5 a 15 minut, gdyż trzeba doliczyć czas na ręczne sprawdzenie tożsamości użytkownika (legitymacja lub indeks), wyszukanie użytkownika w interfejsie, wybranie odpowiedniej osoby, zmiana hasła, wydrukowanie lub przepisanie na kartkę nowego ha-

sła, krótka instrukcja dla użytkownika, gdzie może hasło tymczasowe zmienić na własne, czasem też instrukcja dokładniejsza, jak to hasło zmienić. W sumie otrzymano pracę między 2,5 a 7 godzin dziennie (także soboty i niedziele), przyjęto bardzo optymistycznie i szacunkowo, że tylko cztery godziny dziennie, co miesięcznie daje 120 godzin, czyli 0,7 etatu. Oczywiście nie może to być jedna osoba z racji pracy w soboty i niedziele, a nie powinny to być tylko dwie z racji urlopów i chorób. Na szczęście to w niewielkim stopniu wpływa na koszty, dla tych pozostałych osób znajdzie się na pewno inna praca przy obsłudze użytkowników. Przy medianie płacy w dziale obsługi klienta w 2006 r. wynoszącej 1970 zł miesięcznie[63)], a po uwzględnieniu składek na ZUS 2376 zł, to 0,7 etatu wynosi 1663 zł, co przekłada się dla pracodawcy na koszt zmiany jednego hasła w wysokości 2 zł 46 gr. Do tego należy doliczyć koszt komputera (ze względów bezpieczeństwa lepiej mieć dedykowany tylko do tego celu, nawet jeśli jest wykorzystywany w 70%), co można oszacować na ok. 2500 zł raz na trzy lata, co powiększy koszt zmiany jednego hasła o ok. 10 gr. Nie zostały tu uwzględnione koszty stałe (prąd, światło, ogrzewanie), bo te koszty i tak by zostały poniesione. Można się zastanawiać nad doliczeniem kosztu napisania aplikacji do zmiany haseł (o czym więcej w rozdziale 5.2), gdyż choć akurat w SGH zostało to zrobione bezpłatnie, to taka aplikacja znacząco zmniejsza czas obsługi użytkowników (zmiana hasła w jednym miejscu, nie w trzech), przez co wymiernie wpływa na obniżenie kosztu zmiany hasła. Szacując koszt takiej aplikacji na 10560 zł (20% kosztu całkowitego C_K z rozdziału 5.2.3, po uwzględnieniu oszacowania Boehma opisanego na str. 51), zaś czas życia aplikacji na 7 lat, dokładamy do kosztu zmiany hasła jeszcze 19 gr. W sumie 2 zł 75 gr.

Należy zwrócić uwagę, że przy czynnej populacji użytkowników ok. 13 tys. osób (6 tys. studentów dziennych, 6 tys. studentów zaocznych i 1 tys. pracowników) i rotacji ok. 3 tys. rocznie (piąty rok studentów odchodzi, pierwszy rok przychodzi, rotacje pracowników pomijalne) przez 32 miesiące pracy systemu na jednego użytkownika przypadło 1,1 zmiany hasła. Zatem statystycznie każdy użytkownik raz na trzy lata zapominał hasła i korzystał z pomocy obsługi do jego odzyskania. W rzeczywistości gdyż przez ten czas z systemu skorzystało tylko nieco ponad 10 tys. różnych użytkowników (a nie, z uwzględnieniem rotacji, ponad 19 tys.), a rekordzistą był użytkownik, który skorzystał aż 21 razy. Zwa-

[63)] Sedlak & Sedlak, *Internetowe Badanie Wynagrodzeń 2006*, tab. 4.



Rysunek 5.2. Statystyki zakładanych kont studenckich rocznie.
Źródło: opracowanie własne.

żywszy, że wszyscy studenci mają obowiązek[64)] korzystania z tego hasła przynajmniej dwa razy w roku, to sytuację można ocenić jako dobrą.

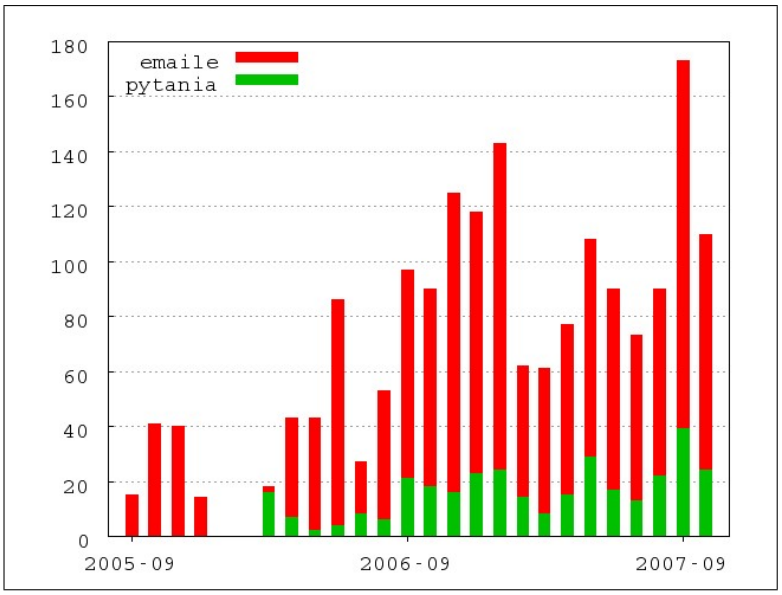
Jest kilka sposobów na zmniejszenie kosztów zmiany haseł. Po pierwsze, koniecznie należy dać użytkownikom możliwość samodzielnego przypomnienia sobie hasła. W SGH zostało to zaimplementowane we wrześniu 2005 r. w dwóch wariantach, każdy z nich możliwy do ustawienia tylko wtedy, gdy użytkownik zna swoje obecne hasło:

poczta elektroniczna użytkownik ma możliwość podania adresu email, na który w trakcie procesu odzyskiwania hasła zostanie wysłany klucz, dzięki któremu użytkownik będzie mógł zmienić sobie hasło bez znajomości poprzedniego;

pytania kontrolne użytkownik ma możliwość podania trzech pytań (w tym jedno wymuszone o PESEL) i odpowiedzi na nie; w trakcie odzyskiwania hasła zostaną mu one wszystkie zadane i po poprawnej odpowiedzi hasło zostanie od razu zmienione i podane użytkownikowi na ekranie.

Z tego sposobu zabezpieczenia się przed zapomnieniem hasła skorzystało przez dwa lata 4540 osób. Statystyki wykorzystania samodzielnego odzyskiwania hasła pokazano na rys. 5.3. W sumie 1800 zmian haseł przy pomocy emaila i 326 przy pomocy pytań bezpieczeństwa, średnio 82 miesięcznie, czyli mniej więcej 12% średniej liczby zmian haseł w tym czasie przez obsługę. Poprawienie tej statystyki może nastąpić tylko i wyłącznie

[64)] Do składania deklaracji przedmiotów na kolejny semestr w Wirtualnym Dziekanacie.



Rysunek 5.3. Statystyki samodzielnego odzyskania haseł w ujęciu miesięcznym. Źródło: opracowanie własne.

przez edukację użytkowników, żeby o takiej możliwości wiedzieli i z niej korzystali. Należy podkreślać, że z tego sposobu zmiany hasła można skorzystać w dowolnym miejscu z dostępem do Internetu, nie jak w przypadku zmiany hasła przez obsługę, gdzie trzeba się stawić osobiście i z legitymacją.

Kolejną możliwość obniżenia kosztu zmiany haseł daje skrócenie czasu obsługi użytkownika. Można to osiągnąć na kilka sposobów. Po pierwsze, można skrócić czas uwierzytelniania i wyszukiwania użytkownika w bazie. Ostatnio wdrażane są legitymacje elektroniczne, można by z nich skorzystać, gdyby miały zakodowany login[65)]. Użytkownik wkładałby legitymację do czytnika, aplikacja odczytywałaby login, przekazywała do programu do zmiany haseł, a ten od razu przechodził do kroku drugiego, czyli wyboru systemów, na które trzeba zmieniać hasło. Powinno to oszczędzić dwie minuty z założonych średnio 8-9, czyli do 25% – istotna oszczędność, ale wymaga wdrożenia legitymacji elektronicznych, które nie są tanie. Wdrażanie ich tylko z powodu oszczędności w systemie zmiany haseł byłoby, delikatnie rzecz ujmując, bardzo niegospodarne. Ale jeżeli są i tak wdrażane

[65)] Legitymacje mają zakodowany PESEL, ale, niestety, nie mamy w obecnej bazie LDAP, z której uwierzytelniani są użytkownicy, wszystkich numerów PESEL. Można się spierać, co jest tańsze, dopisywanie loginów do legitymacji, czy dopisywanie numerów PESEL do bazy LDAP. Argumentem przeważającym za wyborem loginu jako podstawowego wyróżnika jest brak numeru PESEL dla studentów zagranicznych. Argumentem przeważającym za wyborem innej informacji niż login jako podstawowego wyróżnika jest możliwość zmiany loginu.

z innego powodu[66)], to pozostaje tylko oszacowanie kosztu przystosowania aplikacji do odczytywania informacji z legitymacji oraz kosztu czytnika.

Po drugie, można zrezygnować z kroku wyboru systemów i zmieniać po prostu na wszystkich. Jednak to zaoszczędzi niewiele czasu, około 10 sekund na użytkownika, niecałe 2%. Po trzecie, zamiast przepisywać hasło na karteczkę i tłumaczyć, jak i gdzie je zmieniać, można drukować te informacje. Tu oszczędność może być bardzo duża, rzędu 4-5 minut, czyli 50% kosztów, za cenę wydrukowania kartki.Średniej klasy drukarka laserowa (wystarczy monochromatyczna) to koszt ok. 1000 zł (przy założeniu trzyletniej amortyzacji), ryza papieru (500 szt.) to 16 zł, przez trzy lata zmienimy ok. 25 tys. haseł, czyli 50 ryz papieru w sumie za 800 zł, do tego jeszcze koszt toneru, około 200 zł za opakowanie toneru wystarczającego na wydrukowanie 2000 stron A4 przy 5% zadruku strony, czyli 12 dodatkowych sztuk za 2400 zł. Razem $1000 + 800 + 2400 = 4200$ zł, czyli 17 gr za stronę. Można kupić droższą drukarkę, która ma mniejszy koszt wydruku na stronę, ale przy tym wolumenie wydruku nie wydaje się to lepszym rozwiązaniem. Ten sposób pozwala zaoszczędzić 1 zł 23 gr kosztem 17 gr, skrócenie czasu obsługi zwiększy zadowolenie klienta, a posiadanie wydrukowanej instrukcji zmiany hasła, gdzie można też poinformować użytkowników o alternatywnych metodach odzyskiwania hasła, może przyczynić się do ogólnego spadku liczby zmian haseł.

Kolejny sposób na zmniejszenie kosztów zmiany haseł to wymuszenie stosowania dłu-

---

66) Minister Nauki i Szkolnictwa Wyższego wprowadził[67)]możliwość wydawania legitymacji elektronicznych, a później określił tę formę jako podstawową[68)]. Minister Infrastruktury określił[69)], jakie wzory dokumentów są ważne, jeśli chodzi o przejazdy ulgowe komunikacją zbiorową i kolejową, zaś Minister Transportu przedłużył[70)]ważność papierowych legitymacji studenckich jako dokumentu uprawniającego do przejazdów ulgowych tylko do końca roku 2007. Dla wszystkich szkół wyższych jest to wystarczający powód dla wdrożenia legitymacji elektronicznych w tym terminie.

67) *Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 18 lipca 2005 r. w sprawie dokumentacji przebiegu studiów*, Dz. U. z 2005 r. Nr 149, poz. 1233.

68) *Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 2 listopada 2006 r. w sprawie dokumentacji przebiegu studiów*, Dz. U. z 2006 r. Nr 224, poz. 1634.

69) *Rozporządzenie Ministra Infrastruktury z dnia 25 października 2002 r. w sprawie rodzajów dokumentów poświadczających uprawnienia do korzystania z ulgowych przejazdów środkami publicznego transportu zbiorowego*, Dz. U. z 2002 r. Nr 179, poz. 1495.

70) *Rozporządzenie Ministra Transportu z dnia 21 sierpnia 2006 r. zmieniające rozporządzenie w sprawie rodzajów dokumentów poświadczających uprawnienia do korzystania z ulgowych przejazdów środkami publicznego transportu zbiorowego*, Dz. U. z 2006 r. Nr 153, poz. 1098.

gich, bardzo bezpiecznych haseł, najlepiej w postaci jakiegoś zdania, i znaczne zmniejszenie częstotliwości ich zmiany. Im rzadziej zmieniane hasło (i im częściej używane), tym łatwiej je w końcu zapamiętać.

Sposobem na zlikwidowanie wszelkich kosztów zmiany haseł jest zrezygnowanie z haseł w ogóle. Nie oznacza to, rzecz jasna, zrezygnowania z zabezpieczenia dostępu do konta. Można skorzystać z jednego z kilku alternatywnych sposobów uwierzytelniania użytkownika. Wspomniano wcześniej o systemach biometrycznych, ale są to stosunkowo drogie urządzenia – najtańszy czytnik linii papilarnych to ok. 100 zł, ale taki czytnik trzeba by umieścić przy każdym komputerze, a gdyby zrezygnować z haseł w ogóle, to trzeba by dać go praktycznie każdemu użytkownikowi, co może mieć sens tylko w niektórych zastosowaniach (bankowych, wojskowych). Skądinąd niektóre banki już stosowały generatory kodów jednorazowych, tzw. tokeny, do zabezpieczania transakcji w Internecie, jednak ze względu na wysokie koszty samych urządzeń zaczęły je zastępować kartami kodów jednorazowych lub kodami jednorazowymi przesyłanymi do telefonu komórkowego przez SMS. Takie rozwiązania wymagają jednak szczegółowych analiz, gdyż jedne koszty (zmiany haseł) zastępuje się drugimi (urządzenia, drukowanie kart, koszty SMS-ów).

5.2. Analiza kosztów oprogramowania

Od samego początku budowania systemów informatycznych kierownicy chcieli otrzymywać oszacowania, ile projekt informatyczny będzie kosztował. Przyjmowano różne metody szacowania takich kosztów, m.in. oparte o projektowaną liczbę linii kodu. Po uwzględnieniu doświadczeń na kilkunastu lub kilkudziesięciu projektach pracownicy dużych firm programistycznych proponowali własne modele matematyczne. W kolejnych rozdziałach przy pomocy kilku modeli szacowania liczby osobogodzin wymaganych do budowy projektu zostanie przedstawione oszacowanie kosztów budowy systemu zmiany haseł z rozdziału 4.

Projekt jest niewielki i niezbyt skomplikowany, tworzony w modelu buduj-i-poprawiaj; szacowana wielkość kodu nie powinna przekroczyć 2,5 tys. linii.

5.2.1. Modele algorytmiczne oparte o wielkość kodu

Oszacowanie nakładów programistycznych w zależności od szacowanej wielkości kodu wynikowego liczonego w tysiącach linii (KLOC) intuicyjnie wydaje się łatwe. Niestety,

tak nie jest, a badacze przedstawiają różne modele, zazwyczaj pasujące tylko do tych projektów, które sami badali.

Algorytmiczne oszacowanie nakładów według najpopularniejszych modeli zależnych od wielkości kodu:

$$\begin{aligned} \text{Walston \& Felix}^{71)} \quad & E = 5,2 \cdot \text{KLOC}^{0,91} \approx 11,97 \,\text{osobomiesięcy} \\ \text{Bailey-Basili}^{72)} \quad & E = 5,5 + 0,73 \cdot \text{KLOC}^{1,16} \approx 7,61 \,\text{osobomiesięcy} \\ \text{Boehm}^{73)} \quad & E = 3,2 \cdot \text{KLOC}^{1,05} \approx 8,37 \,\text{osobomiesięcy} \end{aligned}$$

5.2.2. Modele algorytmiczne oparte o punkty funkcyjne

Inni badacze zauważyli, że oszacowanie na podstawie liczby wyprodukowanych linii kodu nie jest łatwe na etapie projektowania. Z tego powodu zaproponowali modele matematyczne oparte na liczbie punktów funkcyjnych (FP), czyli liczby funkcjonalności, które dostaje użytkownik. Miary te mogą być modyfikowane przez różne czynniki techniczne, które pozwalają na zróżnicowanie trudności, a zatem i kosztu różnych funkcji.

Szacunki punktów funkcyjnych oprogramowania do zmiany haseł zostały przedstawione w tab. 5.1.

	złożoność komponentu		
komponent	proste	przeciętne	złożone
wejścia	1	2	0
wyjścia	4	0	0
zapytania	4	1	0
pliki główne	3	0	0
interfejsy	4	0	0

Tabela 5.1. Punkty funkcyjne

[71)] C. E. Walston i C. P. Felix, *A method of programming measurement and estimation*, IBM Systems Journal, 1977, t. 16, nr 1, s. 54-73.

[72)] J. W. Bailey i V. R. Basili, *A Meta-Model for Software Development Resources Expenditures*, IEEE Press, 1981, s. 107-116.

[73)] B. Boehm, *Software Engineering Economics*, Prentice Hall PTR, Upper Saddle River, NJ, 1981.

czynniki techniczne	stopień wpływu (0-5)
szybkość przesyłania danych	3
przetwarzanie rozproszone	0
częstotliwość wykonywania transakcji	3
kryteria wydajnościowe	3
stopień obciążenia sprzętu	2
wprowadzanie danych online	5
umiejętności użytkownika	3
natychmiastowa aktualizacja	5
złożoność wykonywanych obliczeń	1
przydatność do wielokrotnego użycia	3
łatwość instalacji	1
łatwość obsługi	5
przenośność	3
konserwowalność systemu	3
$\sum$	40

Tabela 5.2. Czynniki techniczne (DI, degree of influence)

Suma ważona[74)] niezmodyfikowanych punktów funkcyjnych:

$$\begin{aligned}\mathrm{UFP} = {} & 1 \cdot 3 + 2 \cdot 4 + 0 \cdot 6 + 4 \cdot 4 + 0 \cdot 5 + 0 \cdot 7 \\ & + 4 \cdot 3 + 1 \cdot 4 + 0 \cdot 6 + 3 \cdot 7 + 0 \cdot 10 + 0 \cdot 15 \\ & + 4 \cdot 5 + 0 \cdot 7 + 0 \cdot 10 = 84\end{aligned}$$

W tabeli 5.2 został podsumowany wpływ 14 czynników technicznych, co pozwala nam obliczyć zmodyfikowaną wielkość punktów funkcyjnych:

$$\mathrm{FP} = (0,65 + 0,01 \cdot 40) \cdot \mathrm{UFP} = 88,2$$

Czynniki techniczne są ocenione subiektywnie, można w kilku miejscach dyskutować, czy na pewno zostały dobrze dobrane, ale przy najlepiej dopasowanym modelu różnice są rzędu 10%, gdy inne modele odbiegają od niego o setki procent.

Algorytmiczne oszacowanie nakładów według modeli opartych na punktach funkcyjnych:

[74)] Wagi zgodnie z zaleceniami International Function Point User Group (IFPUG).

Albrecht & Gafney[75)] $E = -13,39 + 0,0545 \cdot \mathrm{FP} \approx -8,59$ osobomiesięcy

Kemerer[76)] $E = 60,62 \cdot 7,728 \cdot 10^{-8} \cdot \mathrm{FP}^3 \approx 3,19$ osobomiesięcy

Matson, Barret & Mellichamp[77)] $E = 585,7 + 15,12 \cdot \mathrm{FP} \approx 1916,26$ osobomiesięcy

Z powyższych oszacowań najbliższy rzeczywistości okazał się model Kemerera.

5.2.3. Podstawowy model COCOMO

Na podstawie swoich doświadczeń z szacowaniem kosztów na podstawie wielkości kodu w tysiącach linii Boehm[78)] zaprojektował w 1981 r. model COCOMO (ang. *COnstructive COst MOdel*). Uogólnił w nim przez wielkości wykorzystywanych współczynników kilka rodzajów projektów programistycznych.

Oszacowanie według podstawowego modelu COCOMO dla małego zespołu i znanego środowiska, innymi słowy dla organicznej klasy przedsięwzięcia:

$$E = 2,4 \cdot \mathrm{KLOC}^{1,05} = 6,28128 \approx 6,28 \text{ osobomiesięcy}$$

$$D = 2,5 \cdot E^{0,38} = 5,0253 \approx 5,03 \text{ miesięcy}$$

$$P = E/D = 1,2485 \approx 1,25 \text{ osób}$$

gdzie E jest wyrażone w osobomiesiącach, D to czas rozwijania oprogramowania w miesiącach, a P – wymagana do projektu liczba osób. Przy medianie płacy w branży IT na poziomie 3500 zł miesięcznie[79)] przybliżony koszt systemu wynosi:

$$C = 6,28 \cdot 3500 \approx 22000 \text{ zł}$$

Kwota ta może wydawać się duża, ale należy pamiętać, że obejmuje koszt projektowania, kodowania, wdrożenia, testowania i tworzenia dokumentacji (zarówno użytkownika, jak i programowej).

---

75) A. Albrecht i J. Gaffney, *Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation*, IEEE Transactions on Software Engineering, b.m., 1983, t. SE-9, nr 6, s. 639-648

76) C. Kemerer, *An Empirical Validation of Software Cost Estimation Models*, Communications of the ACM, 1987, t. 30, nr 5.

77) J. Matson, B. Barret, J. Mellichamp, *Software Development Cost Estimation Using Function Points*, IEEE Transactions on Software Engineering, 1994, t. 20, nr 4, s. 275-287.

78) B. Boehm, *Software Engineering Economics*, Prentice Hall PTR, Upper Saddle River, NJ, 1981.

79) Sedlak & Sedlak, *Internetowe...*, op. cit., tab. 4

Przy dodatkowym narzucie kosztów korporacyjnych (pomieszczenia, wyposażenie, księgowość itd.) ostatecznie szacowany koszt systemu wyniósłby:

$$C_K = 2,40 \cdot 22000 = 52800\,\text{zł}$$

5.2.4. Pośredni model COCOMO

Rozszerzeniem podstawowego modelu COCOMO jest pośredni model COCOMO, gdzie szacuje się czas programisty potrzebny do wyprodukowania oprogramowania. Najważniejszą rolę grają tutaj współczynniki nośników kosztów, które trzeba oszacować dla danego typu projektu. W tym modelu dla organicznej klasy przedsięwzięcia mamy wzór:

$$E = 3,2 \cdot \text{KLOC}^{1,05} \cdot \prod_{i=1}^{15} a_i$$

gdzie a_i to współczynniki nośników kosztów z tabeli 5.3.

Czas rozwijania i osobochłonność liczy się tak samo jak w modelu podstawowym; ostatecznie mamy:

$$E = 3,2 \cdot 2,5^{1,05} \cdot 2,25 = 18,84 \approx 19\,\text{osobomiesięcy}$$

$$D = 2,5 \cdot E^{0,38} \approx 7,65\,\text{miesięcy}$$

$$P = E/D \approx 2,5\,\text{osoby}$$

Wygląda na to, że zbyt pesymistycznie oceniono jakość pracy i doświadczenie analityków i programistów; jeśli przyjąć iloczyn współczynników równy jeden, otrzymujemy $E = 8,37$, $D = 5,6$ i $P = 1,5$.

Podsumowanie

Wszystkie fazy budowania systemu łącznie z programowaniem przeprowadzono samodzielnie, a powyższy opis jest próbą podsumowania i oszacowania kosztu takiego projektu już post factum. Ponieważ podczas tworzenia tego systemu powstawały potrzeby realizacji innych projektów pomocniczych (np. aplikacja serwująca zdjęcia w zależności od ustawień użytkownika) oraz ze względu na wieloetapowość budowania-i-poprawiania (np. moduł zmieniający hasła na Windows został dodany bardzo późno), ogólny czas napisania i wdrożenia tego systemu wynosił około pół roku, z czego projektowanie i kodowanie trwało ok. 5 tygodni.

Czas realizacji pokrywa się z oszacowaniem w modelu COCOMO. O ile całkowity szacunek zgadza się tylko przypadkiem, o tyle w części dotyczącej kodowania, jeśli uwzględnić specyficzny charakter projektu (nie miał on dodatkowych kosztów korporacyjnych,

Nośniki kosztów		Ocena	Mnożnik
Atrybuty produktu			
RELY	wymagana niezawodność	przeciętna	1,00
DATA	wielkość bazy danych	przeciętna	1,00
CPLX	złożoność produktu	przeciętna	1,00
Atrybuty komputera			
TIME	ograniczenia czasu wykonania	przeciętne	1,00
STOR	ograniczenia pamięci operacyjnej	przeciętne	1,00
VIRT	niestabilność pamięci wirtualnej	mała	0,87
TURN	czas obrotu zadania	przeciętny	1,00
Atrybuty personelu			
ACAP	możliwości analityków	małe	1,19
AEPX	doświadczenia w danym typie aplikacji	przeciętne	1,00
PCAP	możliwości programistów	przeciętne	1,00
VEXP	doświadczenie w stosowaniu pamięci wirtualnej	bardzo małe	1,21
LEXP	doświadczenie w danym języku programowania	wysokie	0,95
Atrybuty przedsięwzięcia			
MODP	nowoczesne praktyki programistyczne	brak	1,24
TOOL	zastosowanie narzędzi programowych	bardzo małe	1,24
SCHED	ograniczenia w czasie wytwarzania	bardzo małe	1,23
		$\prod$	2,25

Tabela 5.3. Nośniki kosztów modelu COCOMO

nie ma dokumentacji, zaś projektowanie i wdrożenie były przeprowadzane jednocześnie z kodowaniem), zgadza się wręcz idealnie. Według Boehma[80)] dla małego projektu (rzędu 2KLOC) samo kodowanie i testy modułów to 42% wyliczanych wartości. Nie podaje on niestety, ile z tego przypada na testy modułów, ale przy założeniu, że około połowa, to część przeznaczoną na kodowanie razem z poprawianiem błędów można przyjąć ok. 20%, co ostatecznie daje wyżej wspomniany $6,28 \cdot 20\% = 1,25$ osobomiesiąca.

Organizacja (SGH) nie poniosła w związku z tworzeniem tego systemu żadnych kosztów. Z systemu można korzystać pod adresem: `https://akson.sgh.waw.pl/passwd/` (od momentu powstania został znacznie rozbudowany i przekształcił się w aplikację do zarządzania kontem).

[80)] B. Boehm, *Software Engineering Economics*, Prentice Hall PTR, Upper Saddle River, NJ, 1981, s. 65, tab. 5-2.

6. Wnioski

Poczucie bezpieczeństwa, nawet najbardziej usprawiedliwione, jest złym doradcą.
Joseph Conrad, *Zwierciadło morza*, 1906

Systemy informatyczne będą jeszcze przez długi czas posługiwać się hasłami do uwierzytelniania i zabezpieczania dostępu. Jednocześnie rozwój technologiczny spowoduje, że coraz więcej mocy obliczeniowej będzie dostępne dla zwykłej osoby, a tym bardziej włamywaczy. Mając to na uwadze, należy postulować stosowanie jak najbezpieczniejszych haseł, możliwie najdłużej opierających się atakom. Jednym ze sposobów pozwalających na spełnienie takiego postulatu jest sprawdzanie siły hasła w trakcie jego zmiany i odrzucanie haseł, które są zbyt słabe. Najlepszym sposobem sprawdzania siły haseł wydaje się być wyliczenie zmodyfikowanej entropii hasła, co zostało zaprezentowane w postaci programu komputerowego. Dodatkową jego zaletą jest zmniejszenie kosztów obsługi przez równoczesne zmiany haseł na wszystkich dostępnych systemach informatycznych.

Jestem przekonany, że taki system zmiany haseł będzie mógł służyć w SGH przez długie lata. Jednak w przypadku przełomowych przyspieszeń na polu technologii i mocy obliczeniowych nie wystarczy już zwiększanie progów, od których uznajemy, że hasło jest silne – trzeba będzie zbadać systemy alternatywne. Równocześnie należy zastanowiać się nad metodami zwiększenia udziału użytkowników w *kulturze bezpieczeństwa*, gdyż żadne metody komputerowe nie dadzą efektów, jeżeli nie będą zaakceptowane, rozumiane i przestrzegane przez ludzi.

Spis rysunków

Spis tabel

Bibliografia

[1] Allan J. Albrecht i John E. Gaffney, *Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation*, IEEE Transactions on Software Engineering, 1983, t. SE-9, nr 6.

[2] W. Ross Ashby, *Wstęp do cybernetyki*, PWN, Warszawa, 1994.

[3] John W. Bailey i Vic R. Basili, *A Meta-Model for Software Development Resources Expenditures*, IEEE Press, 1981.

[4] Barry W. Boehm, *Software Engineering Economics*, Prentice Hall PTR, Upper Saddle River, NJ, 1981.

[5] Andrzej Białas, *Bezpieczeństwo informacji i usług w nowoczesnej instytucji i firmie*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2006.

[6] Henryka Egeman, *Zagadnienia spójności systemów informatycznych zarządzania*, Oficyna Wydawnicza SGH, Warszawa, 1993.

[7] Wiesław Flakiewicz, *Pojęcie informacji w technologii multimedialnej*, Oficyna Wydawnicza SGH, Warszawa, 2005.

[8] Jan Goliński, *Wybrane problemy organizacji projektowania i programowania komputerów*, Pracownia Poligraficzna Wydawnictw Uczelnianych SGPiS, Warszawa, 1978.

[9] Richard W. Hamming, *Error Detecting and Error Correcting Codes*, Bell System Technical Journal, 1950.

[10] Chris F. Kemerer, *An Empirical Validation of Software Cost Estimation Models*, Communications of the ACM, 1987, t. 30, nr 5.

[11] Auguste Kerckhoff, *La Cryptographie Militaire, Journel des Sciences militaires*, Libraire Militaire de L. Baudoin & Co., Paris, 1883.

[12] Vladimir I. Levenshtein, *Binary codes capable of correcting deletions, insertions and reversals*, Doklady Akademia Nauk SSSR, 1965.

[13] Jack E. Matson, Bruce E. Barret, Joseph M. Mellichamp, *Software Development Cost Estimation Using Function Points*, IEEE Transactions on Software Engineering, 1994, t. 20, nr 4.

[14] Donald L. Pipkin, *Bezpieczeństwo informacji*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2000.

[15] Bruce Schneier, *Kryptografia dla praktyków*, Wydawnictwa Naukowo-Techniczne, Warszawa, 2002.

[16] Claude E. Shannon, *A mathematical theory of communication*, Bell System Technical Journal, Murray Hill, NJ, 1948, t. 27.

[17] Claude E. Shannon, *Communication theory of secrecy systems*, Bell System Technical Journal, Murray Hill, NJ, 1949, t. 28.

[18] Simon Singh, *Księga szyfrów*, Albatros, Warszawa, 2001.

[19] *Słownik Języka Polskiego*, pod red. Mieczysława Szymczaka, Wydawnictwo Naukowe PWN, Warszawa, 1999.

[20] Bogdan Stefanowicz, *Informacyjne systemy zarządzania - przewodnik*, Oficyna Wydawnicza SGH, Warszawa, 2007.

[21] Claude E. Walston i Charles P. Felix, *A method of programming measurement and estimation*, IBM Systems Journal, 1977, t. 16, nr 1.

Ustawy i rozporządzenia

[22] *Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 18 lipca 2005 r. w sprawie dokumentacji przebiegu studiów*, Dz. U. z 2005 r. Nr 149, poz. 1233.

[23] *Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 2 listopada 2006 r. w sprawie dokumentacji przebiegu studiów*, Dz. U. z 2006 r. Nr 224, poz. 1634.

[24] *Rozporządzenie Ministra Infrastruktury z dnia 25 października 2002 r. w sprawie rodzajów dokumentów poświadczających uprawnienia do korzystania z ulgowych przejazdów środkami publicznego transportu zbiorowego*, Dz. U. z 2002 r. Nr 179, poz. 1495.

[25] *Rozporządzenie Ministra Spraw Wewnętrznych i Administracji z dnia 29 kwietnia 2004 r. w sprawie dokumentacji przetwarzania danych osobowych oraz warunków technicznych i organizacyjnych, jakim powinny odpowiadać urządzenia i systemy informatyczne służące do przetwarzania danych osobowych*, Dz. U. z 2004 r. Nr 100, poz. 1024.

[26] *Rozporządzenie Ministra Transportu z dnia 21 sierpnia 2006 r. zmieniające rozporządzenie w sprawie rodzajów dokumentów poświadczających uprawnienia do korzystania z ulgowych przejazdów środkami publicznego transportu zbiorowego*, Dz. U. z 2006 r. Nr 153, poz. 1098.

[27] *Ustawa z dnia 6 czerwca 1997 r. - Kodeks karny*, Dz. U. z 1997 r. Nr 88, poz. 553.

[28] *Ustawa z dnia 18 marca 2004 r. o zmianie ustawy – Kodeks Karny, ustawy – Kodeks Postępowania Karnego oraz ustawy – Kodeks Wykroczeń*, Dz. U. z 2004 r. Nr 69, poz. 626.

Standardy i zalecenia

[29] *ISO/IEC 27001:2005, Information technology – Security techniques – Information security management systems – Requirements*

[30] *ISO/IEC 27002:2005, Information technology – Security techniques – Code of practice for information security management*

[31] *OECD Guidelines for the Security of Information Systems and Networks: Towards a Culture of Security*, OECD, Paris, 2002.

[32] PN-I-07799: *Systemy zarządzania bezpieczeństwem informacji – Specyfikacje i wytyczne do stosowania*, PKN, Warszawa, 2005.

Artykuły internetowe

[33] *Elcomsoft files patent for revolutionary technique to recover lost passwords quickly*, Elcomsoft, 2007-10-22 [dostęp: 2007-11-28]. Adres: `http://www.elcomsoft.com/EDPR/gpu_en.pdf`

[34] *Internetowe Badanie Wynagrodzeń 2006*, Wynagrodzenia.pl Sedlak & Sedlak, 2006-02-07 [dostęp: 2007-11-28]. Adres: `http://www.wynagrodzenia.pl/artykul.php/wpis.1011`

[35] Tom Jennings, *An annotated history of some character codes or ASCII: American Standard Code for Information Infiltration*, 2004-10-29 [dostęp: 2007-11-28]. Adres: `http://www.wps.com/projects/codes/`

[36] Larry Light, *Hackers' delight*, Bussiness Week, 1997-02-10 [dostęp: 2007-11-28]. Adres: `http://www.businessweek.com/1997/06/b351314.htm`

[37] *Novell Password Management Administration Guide*, Novell Documentation, 2007-03-09 [dostęp: 2007-11-28]. Adres: `http://www.novell.com/documentation/password_management31/index.html`

[38] *OECD Directorate for Science, Technology and Industry*, OECD, [dostęp: 2007-11-28]. Adres: `http://www.oecd.org/sti/`

[39] *Własności statystyczne języka polskiego – badania częstości występowania liter i par liter w tekstach polskich*, Kryptomania, 1999-06-11 [dostęp: 2007-11-28] Adres: `http://www-users.mat.uni.torun.pl/~krypto/zasoby/wlasn_stat_liter.htm`