Podstawowe zagadnienia kryptograficzne
Funkcja skrótu

Anna Dziuba, 146393

1. Aplikacja

```
1. Funkcje skrótu dla tekstu wprowadzanego przez użytkownika
2. Wejście losowe o różnych długościach
Wybierz opcję (1 lub 2): 1
Wprowadź tekst: Testowy tekst
MD5:
Wartość skrótu: 6431a03fb990d6e84a931c3be1ae9f72
Długość: 32 znaków (128 bitów)
Czas wykonania: 0.0287 ms

SHA-1:
Wartość skrótu: 0a20194d6deccf191e0743437d67805ce6da28b7
Długość: 40 znaków (160 bitów)
Czas wykonania: 0.0106 ms

SHA-256:
Wartość skrótu: 4b435ce2b0c4bdbe227c993b462ea999b7bd4f89c6320f52e1f3ed10b9503569
Długość: 64 znaków (256 bitów)
Czas wykonania: 0.0101 ms

SHA-512:
Wartość skrótu: 51cfbf81359b3d8253f042056a0fd6b6ddcf31d2412218ddda2835fcedd461cd4bf50103c9790acce8c301a3eeb1a6fb05cff8f380bffb315ec817297ca869d1
Długość: 128 znaków (512 bitów)
Czas wykonania: 0.0115 ms

SHA3-256:
Wartość skrótu: 546b990c01b495e403c1772295bb337a26bae942be3d8141dd235498903f82d0
Długość: 64 znaków (256 bitów)
Czas wykonania: 0.0086 ms

SHA3-512:
Wartość skrótu: 722f6b24d2e0f3668afeea867f48a745b64c569ed7a994b24eb92c852534c288a1149d59ebe0bd03f63d484b204236e90aa0c499db65ddbe1c5d090e797dcde5
Długość: 128 znaków (512 bitów)
Czas wykonania: 0.0035 ms
```

1.1. Sposób implementacji

Program został zaimplementowany w języku Python z wykorzystaniem wbudowanej biblioteki hashlib, która zapewnia dostęp do funkcji skrótu – MD5, SHA-1, SHA-256, SHA-512, SHA3-256 i SHA3-512. Program działa w dwóch trybach: użytkownik może wpisać własny tekst lub przetestować funkcje na losowych danych o różnych długościach.

Dla każdej funkcji skrótu mierzony jest czas wykonania (średnia z wielu iteracji), a także prezentowana jest długość wygenerowanego skrótu w bitach.

1.2. Rola soli w tworzeniu skrótów

Sól to losowo wygenerowany ciąg znaków, który dodaje się do danych wejściowych przed ich skrótem (np. hasła), aby zwiększyć bezpieczeństwo.

Bez soli dwa identyczne hasła miałyby ten sam skrót, co ułatwia ataki z użyciem rainbow tables (ataki słownikowe). Sole sprawiają, że nawet proste czy krótkie hasła generują unikalne skróty. Dodatkowo, jeśli dwóch użytkowników ma takie samo hasło, ale różne sole to ich skróty będą różne, co zapewnia unikalność skrótów dla tych samych haseł.

2. Porównaj szybkość działania poszczególnych funkcji oraz długość ciągów wyjściowych.

Użyj zbioru danych wejściowych o zróżnicowanej długości.

Poszczególne funkcje skrótu przetestowano dla 4 różnych długości danych wejściowych – 10, 100, 1000 i 10000 losowych znaków. Długości ciągów wyjściowych dla poszczególnych funkcji skrótu nie zmieniała się w zależności od długości danych wejściowych i wynosiła:

- MD5: 32 znaki (128 bitów)
- SHA-1: 40 znaki (160 bitów)
- SHA-256: 64 znaki (256 bitów)
- SHA-512: 128 znaków (512 bitów)
- SHA3-256: 64 znaki (256 bitów)
- SHA3-512: 128 znaków (512 bitów)

Pomiary szybkości działania funkcji skrótu (wyniki w milisekundach) – średnia z 5 pomiarów:

Długość ciągu wejściowego	MD5	SHA-1	SHA-256	SHA-512	SHA3-256	SHA3-512
500	0,0022	0,0014	0,0024	0,0024	0,0027	0,0037
1000	0,0022	0,0019	0,0030	0,0026	0,0043	0,0057
5000	0,0075	0,0056	0,0109	0,0080	0,0143	0,0269
10000	0,0137	0,0101	0,0207	0,0151	0,0275	0,0462

Dla wszystkich funkcji skrótu czas działania funkcji skrótu rosną wraz z zwiększeniem długości ciągu wejściowego.
SHA-1 dla wszystkich długości ciągu wejściowego okazała się najszybszą funkcją skrótu. Podobnie szybka, jednak nieco wolniejsza okazała się funkcja MD5. Obie te funkcje są szybkie, jednak mniej bezpieczne.
Wolniejsze niż MD5 są SHA-512 oraz SHA-256, przy czym SHA-512 jest nieco szybsza. Oba warianty SHA-3 są wolniejsze niż poprzednie funkcje skrótu, gdyż ich algorytm jest najbardziej złożony. Funkcja SHA3-512 jest najwolniejsza.

Hierarchia szybkości:
1. SHA-1
2. MD5
3. SHA-512
4. SHA-256
5. SHA3-256
6. SHA3-512

3. Wygeneruj skrót dla słowa wejściowego, nie dłuższego niż 4 znaki. Skopiuj wartość uzyskaną dla funkcji MD5 i sprawdź, czy wartość wejściowa jest powszechnie znana. Co można powiedzieć o bezpieczeństwie skrótów z krótkich haseł składowanych w bazach danych?

Słowo wejściowe: test
Uzyskana wartość skrótu: 098f6bcd4621d373cade4e832627b4f6
Użyto strony: https://crackstation.net/

Wartość wejściowa jest powszechnie znana – wprowadzenie wartości funkcji skrótu na stronie zwróciło słowo „test”

Skróty krótkich haseł są bardzo niebezpieczne, ponieważ istnieją gotowe bazy skrótów popularnych krótkich haseł. Tego typu bazy to tzw. rainbow tables. W przypadku 3-4 znakowych słów, istnieje bardzo duże prawdopodobieństwo, że jego skrót MD5 znajduje się w publicznych zbiorach, co oznacza, że można go łatwo odwrócić – znaleźć oryginalne hasło na podstawie skrótu.

4. Na podstawie powszechnie dostępnych źródeł odpowiedz na pytanie – czy funkcję MD5 można uznać za bezpieczną? Czy dotychczas zostały znalezione dla niej jakiekolwiek kolizje?

Funkcji MD5 nie można uznać za bezpieczną. Opublikowany został algorytm ataku, dzięki któremu do podrobienia podpisu wystarczyła godzina działania klastrowego komputera IBM P960. Poza tym, odkryto lukę umożliwiającą podrobienie dowolnego certyfikatu SSL w taki sposób, że zostanie on zaakceptowany przez wszystkie popularne przeglądarki internetowe.

W 2005 roku zaprezentowano metodę umożliwiającą znalezienie kolizji dla algorytmu MD5 i przeprowadzenie ataku polegającego na wysłaniu dwóch różnych wiadomości chronionych tym samym podpisem cyfrowych. Opublikowany został też algorytm, który potrafił znaleźć kolizję w ciągu minuty, używając metody nazwanej tunneling.

Źródło: https://bboczkowski.blogspot.com/2017/04/szyfrowanie-2-porownanie-funkcji-skrotu.html

5. Dla wybranej przez siebie funkcji skrótu, zbadaj kolizje na pierwszych 12 bitach skrótu.

Wygenerowałam 500 losowych ciągów znaków, dla których obliczałam funkcję skrótu SHA3-256, a następnie analizowałam pierwsze 12 bitów (3 znaki hex) tego skrótu. Sprawdzałam, które ciągi wejściowe otrzymały takie same pierwsze 12 bitów ciągu wyjściowego, aby wykryć kolizje. Program wypisuje wszystkie grupy kolizyjne oraz liczy ich łączną liczbę.

Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'bde' dla ciagow wejsciowych:
mNquTsx5By
OPrPoY8F5L
Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'e81' dla ciagow wejsciowych:
TuuPurYLlq
05Vi147c0n
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '2a9' dla ciagow wejsciowych:
LwRkbyp3rW
YRSxGS8v1D
fpk7zfCnCT
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '189' dla ciagow wejsciowych:
3dflEe4wQM
RJUUBatoI8
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '2dc' dla ciagow wejsciowych:
8SLV5gwxXH

Xe1NN22ov0
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '167' dla ciagow wejsciowych:
ZazybJEWSP
MwpFNTavnC
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '030' dla ciagow wejsciowych:
NVJqo9zOe2
QCH0a6tosu
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '3fb' dla ciagow wejsciowych:
k4gYktqMjW
N1lsY8WOT5
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '385' dla ciagow wejsciowych:
HKK2RXByQF
ykdPYXZIqS
Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'da4' dla ciagow wejsciowych:
5Igdzvd8PO
j1Dd8ihbT7
Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'b45' dla ciagow wejsciowych:
YozrFTIN4O
yE4FdCaSyK
Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'f70' dla ciagow wejsciowych:
nR1o1oKXj1
SNOWAZ3vse
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '5bc' dla ciagow wejsciowych:
Mlkr2o7D2W
yXiXJOxlEO
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '27f' dla ciagow wejsciowych:
trc053iWXS
G8PQrTqM2t
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '3ae' dla ciagow wejsciowych:
zX6Ax3nlWF
gXKWlDOACH
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '742' dla ciagow wejsciowych:
yOzPA5x5vV
jhVxgziT5f
Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'acf' dla ciagow wejsciowych:
MRSYqnN7PS
1VGuOYQ0y0
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '1bc' dla ciagow wejsciowych:
oRccFA42Oo
zdzvXWB92m
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '9ad' dla ciagow wejsciowych:
50j6ZpNBH4
Vg7W1ZUanm
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '5b7' dla ciagow wejsciowych:
wnjFWZaMUl
iOatIMeiIf
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '7e1' dla ciagow wejsciowych:

V0hiHVcNWw
2KeGSn4pYQ
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '48d' dla ciagow wejsciowych:
mNCwY4kDvI
Jejlq9K78u
Kolizja dla pierwszych 12 bitów ciągu wyjściowego 'a08' dla ciagow wejsciowych:
InY8swb2RC
VVTFWfRB8Q
Kolizja dla pierwszych 12 bitów ciągu wyjściowego '80d' dla ciagow wejsciowych:
S1xTgxrcES
TTHGNkUknS
Znaleziono 24 kolizji.

6. Losowość wyjścia funkcji skrótu (kryterium SAC – Strict Avalanche Criteria) – przy zmianie pojedynczego bitu na wejściu, wszystkie bity wyjściowe powinny zmienić się z prawdopodobieństwem 0,5 każdy. Dla wybranej funkcji skrótu zbadaj tę własność.

Program losuje łańcuch znaków, oblicza jego skrót SHA-256, a następnie modyfikuje jeden bit w pierwszym znaku tego ciągu. Dla zmienionego ciągu również obliczany jest skrót. Na końcu program porównuje wynikowe ciągi bit po bicie, aby sprawdzić, czy zmiana pojedynczego bitu na wejściu spowodowała zmianę około połowy bitów w wyjściu, zgodnie z zasadą SAC (Strict Avalanche Criteria). Nadano toleracje 10%. Wybrany algorytm spełnia tę własność.

```
Wartość funkcji skrótu: 6de1e76ec758a66350b5cffcceb45a52c975f1d2bbe8cc98563fb64b35eefc4a
Wartość nowej funkcji skrótu: eb96ff1ce956ec2570bf31395fe6ca929cd91302436e3c858ed8d1818ce1d779
Zmieniła się około połowa bitów: 119/256 (46.484375%)
```