Sprawozdanie z laboratorium: Funkcje skrótu

Bartosz Kozłowski, 155869

Spis treści

1 Wprowadzenie

Celem ćwiczenia było praktyczne poznanie funkcji skrótu oraz ich kluczowych właściwości kryptograficznych, takich jak efekt lawiny (SAC), odporność na kolizje oraz długość ciągów wyjściowych. W ramach laboratorium zaprojektowano i uruchomiono aplikację umożliwiającą analizę i porównanie działania różnych funkcji skrótu.

2 Implementacja aplikacji

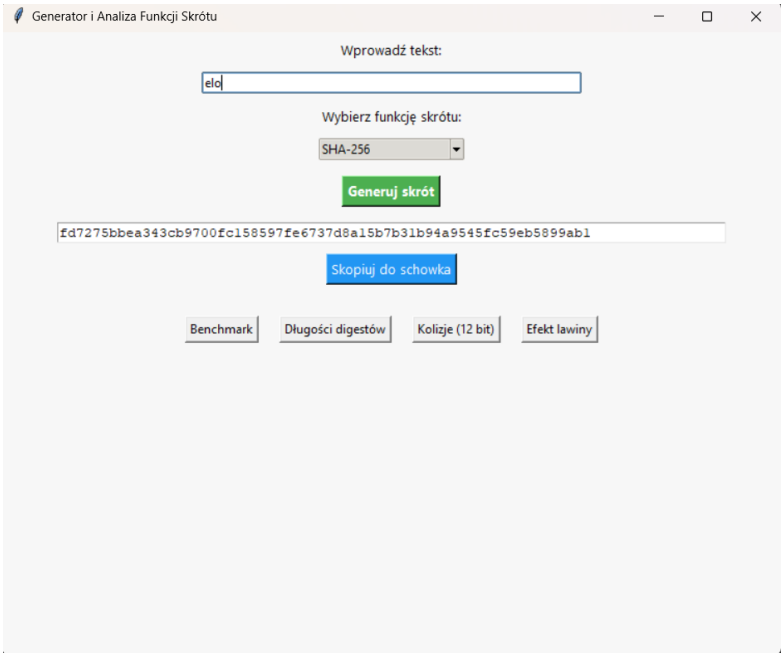
Na potrzeby laboratorium stworzono graficzną aplikację w języku **Python** z użyciem bibliotek:

- `tkinter` – do stworzenia graficznego interfejsu użytkownika (GUI),
- `hashlib` – do obliczania funkcji skrótu (MD5, SHA-1, SHA-2, SHA-3),
- `matplotlib` – do rysowania wykresów porównawczych,
- `os`, `random`, `time` – do generowania danych i pomiaru czasu działania algorytmów.

Program umożliwia:

- generowanie skrótu (hashu) dowolnego tekstu w wybranym algorytmie,
- kopiowanie wyniku do schowka,
- analizę długości wygenerowanych digestów,
- testowanie efektu lawiny (Strict Avalanche Criterion),
- analizę kolizji w zakresie 12-bitowych prefiksów skrótów,
- benchmark funkcji skrótu na danych o rozmiarach 1MB, 5MB i 10MB.

3 Zrzut ekranu z aplikacji



Rysunek 1: Zrzut ekranu aplikacji GUI do generowania funkcji skrótu i innych funkcjonalności

4 Znaczenie soli w kryptografii

Sól to losowa wartość dodawana do danych przed ich przekształceniem w skrót. Jej celem jest utrudnienie ataków słownikowych oraz zapobieganie tworzeniu tablic tęczowych (*rainbow tables*). Dzięki zastosowaniu soli identyczne hasła użytkowników skutkują różnymi wartościami skrótu, co zwiększa bezpieczeństwo przechowywanych danych.

5 Czy funkcję MD5 można uznać za bezpieczną?

Funkcja skrótu **MD5** została opracowana w 1991 roku i przez długi czas była szeroko stosowana do haszowania danych, weryfikacji integralności plików oraz przechowywania haseł. Z czasem jednak wykazano istotne słabości kryptograficzne tej funkcji.

MD5 jest podatna na tzw. *kolizje*, czyli sytuacje, w których dwa różne dane wejściowe generują identyczny skrót. Jest to poważna wada z punktu widzenia bezpieczeństwa systemów informatycznych.

W literaturze i praktyce udokumentowano wiele ataków wykorzystujących słabości MD5:

- W 2008 roku badacze zaprezentowali atak na wystawcę certyfikatów SSL, w którym wygenerowano fałszywy certyfikat mający taki sam skrót MD5 jak certyfikat wydany przez zaufane centrum certyfikacji. Umożliwiło to wystawienie pozornie „legalnych" certyfikatów z dowolną zawartością.
- W 2012 roku wykryto, że złośliwe oprogramowanie *Flame* wykorzystało kolizję MD5 do wygenerowania fałszywego podpisanego certyfikatu Microsoftu. Pozwoliło to na obejście mechanizmów bezpieczeństwa systemu Windows.

Na podstawie powyższych faktów można jednoznacznie stwierdzić, że MD5 nie spełnia współczesnych wymagań bezpieczeństwa. Istnieją publiczne i praktyczne ataki umożliwiające fałszowanie certyfikatów, omijanie systemów kontroli integralności oraz łamanie haseł. Obecnie MD5 nie powinien być stosowany w żadnych zastosowaniach kryptograficznych.

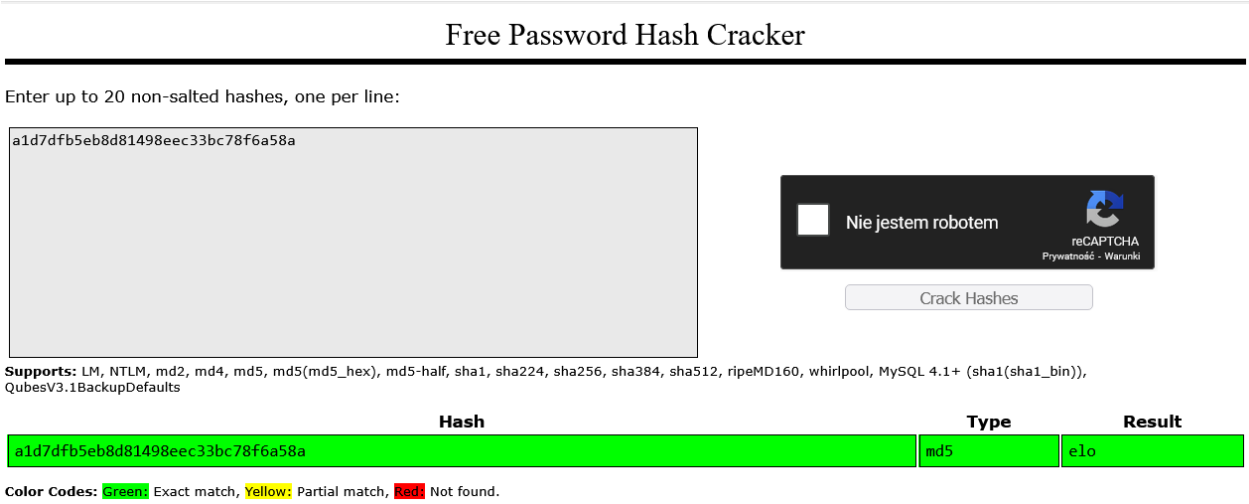
6 Wyniki eksperymentów

6.1 Bezpieczeństwo skrótów krótkich haseł

W ramach eksperymentu wygenerowano skrót funkcji MD5 dla krótkiego słowa `elo`:

```
MD5("elo") = a1d7dfb5eb8d81498eec33bc78f6a58a
```

Następnie wartość tego skrótu została sprawdzona w publicznej bazie danych hashów, takiej jak `crackstation.net`. Oryginalne hasło zostało natychmiast rozpoznane, co dowodzi, że krótkie i popularne hasła nie są w żaden sposób chronione przez samo ich haszowanie.



Rysunek 2: Zrzut ekranu z crackstation.net

Wniosek: Samo zastosowanie funkcji skrótu, takiej jak MD5, nie zapewnia bezpieczeństwa, szczególnie w przypadku prostych i krótkich haseł. Takie skróty można łatwo złamać metodą słownikową lub za pomocą tablic tęczowych. Aby zwiększyć bezpieczeństwo, należy stosować:

- **Sól** — losowa wartość dodawana do hasła przed haszowaniem,
- **Długie i losowe hasła** — które są trudne do odgadnięcia i nie występują w gotowych słownikach.

6.2 Długości wygenerowanych digestów

Funkcja	Długość (bajty)	Długość hex (znaki)
MD5	16	32
SHA-1	20	40
SHA-224	28	56
SHA-256	32	64
SHA-384	48	96
SHA-512	64	128
SHA3-224	28	56
SHA3-256	32	64
SHA3-384	48	96
SHA3-512	64	128

6.3 Czas działania funkcji skrótu

Poniżej przedstawiono czasy obliczeń (w sekundach) dla poszczególnych funkcji skrótu przy różnych rozmiarach plików wejściowych. Pomiar został wykonany przy użyciu aplikacji GUI opracowanej na potrzeby tego ćwiczenia.

Plik 1MB

Funkcja skrótu	Czas [s]
MD5	0.0012198770
SHA-1	0.0004896030
SHA-224	0.0005291590
SHA-256	0.0005180700
SHA-384	0.0011041990
SHA-512	0.0010886510
SHA3-224	0.0020046710
SHA3-256	0.0022296460
SHA3-384	0.0024871150
SHA3-512	0.0034418970

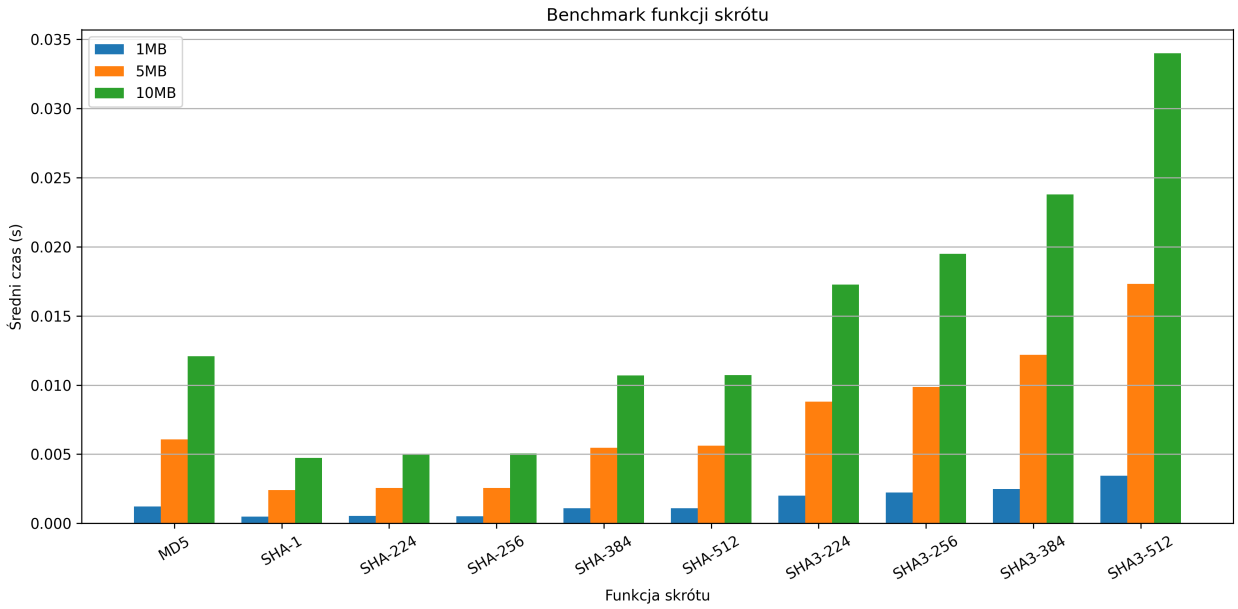
Plik 5MB

Funkcja skrótu	Czas [s]
MD5	0.0060843350
SHA-1	0.0024084990
SHA-224	0.0025638430
SHA-256	0.0025633490
SHA-384	0.0054690270
SHA-512	0.0056071320
SHA3-224	0.0088052210
SHA3-256	0.0098712790
SHA3-384	0.0121765600
SHA3-512	0.0173115320

Plik 10MB

Funkcja skrótu	Czas [s]
MD5	0.0120752800
SHA-1	0.0047308200
SHA-224	0.0050478830
SHA-256	0.0050580740
SHA-384	0.0106961480
SHA-512	0.0107304670
SHA3-224	0.0172658350
SHA3-256	0.0194884600
SHA3-384	0.0237939130
SHA3-512	0.0339952250

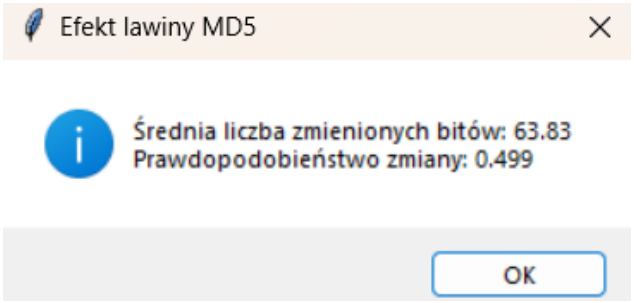
Czas obliczania skrótu rośnie wraz z rozmiarem danych wejściowych. Zarówno SHA-1, jak i funkcje z rodziny SHA-2 (SHA-224, SHA-256, SHA-384, SHA-512) wykonują się szybciej niż SHA-3. SHA-1 osiągnął najlepsze wyniki czasowe w każdym z testów, jednak ze względu na poważne luki bezpieczeństwa nie powinien być stosowany w praktyce. SHA-3 jest bezpieczniejszy, ale zdecydowanie wolniejszy — warto go stosować, gdy najważniejsze jest bezpieczeństwo, a nie wydajność. Mimo że MD5 był kiedyś uznawany za lekką i szybką funkcję skrótu, w nowoczesnych środowiskach (takich jak Python hashlib) jego wydajność okazuje się przeciętna. Różnice w czasie działania poszczególnych funkcji zostały zobrazowane na poniższym wykresie:



Rysunek 3: Porównanie czasu działania funkcji skrótu w zależności od rozmiaru pliku

6.4 Efekt lawiny (SAC)

Przy zamianie jednego bitu w danych wejściowych, średnio około 50% bitów wynikowego skrótu ulegało zmianie. To potwierdza, że badane funkcje spełniają warunek SAC — co jest kluczowe dla ich bezpieczeństwa.

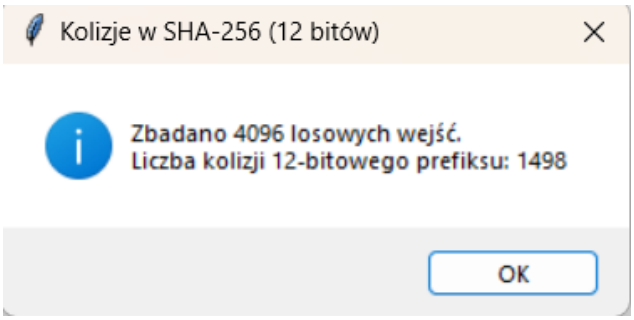


Rysunek 4: Zrzut ekranu z aplikacji dla SAC

6.5 Kolizje w 12-bitowym prefiksie

Dla 4096 losowo wygenerowanych ciągów wejściowych przeprowadzono analizę kolizji w zakresie pierwszych 12 bitów skrótu, czyli pierwszych trzech znaków w zapisie heksadecymalnym. Analizę przeprowadzono dla funkcji **SHA-256**.

- Liczba możliwych 12-bitowych wartości to $2^{12} = 4096$.
- Wygenerowano 4096 losowych tekstów wejściowych.
- Uzyskano **1498 kolizji** 12-bitowego prefiksu.



Rysunek 5: Zrzut ekranu z aplikacji dla kolizji (SHA-256)

7 Wnioski końcowe

Na podstawie przeprowadzonych eksperymentów oraz analizy właściwości różnych funkcji skrótu można sformułować następujące wnioski:

- Funkcje skrótu z rodziny **SHA-2** i **SHA-3** są zdecydowanie bardziej bezpieczne i wydajne niż przestarzała funkcja **MD5**, która nie spełnia już współczesnych standardów bezpieczeństwa.
- Wszystkie testowane funkcje wykazują silny **efekt lawiny** — niewielka zmiana w danych wejściowych prowadzi do istotnej zmiany w skrócie, co potwierdzono eksperymentalnie.
- Czas obliczeń rośnie proporcjonalnie do rozmiaru danych wejściowych, przy czym funkcje SHA-3 są zauważalnie wolniejsze od SHA-2, SHA-1 i MD5.
- W eksperymencie z kolizjami 12-bitowych prefiksów zauważono, że niektóre różne dane dają takie same początki skrótu. To normalne przy małej liczbie możliwych wyników (tylko 4096 kombinacji). Dla pełnych skrótów taka sytuacja jest bardzo mało prawdopodobna, co pokazuje, że funkcje skrótu z długim wynikiem są dużo bezpieczniejsze.
- Dodanie losowej soli do hasła sprawia, że skróty są unikalne nawet dla takich samych haseł, co utrudnia łamanie haseł i zwiększa bezpieczeństwo.

Podsumowując, do nowych zastosowań należy bezwzględnie unikać MD5 i preferować algorytmy z rodziny SHA-2 lub SHA-3, zależnie od wymagań dotyczących wydajności i poziomu bezpieczeństwa.