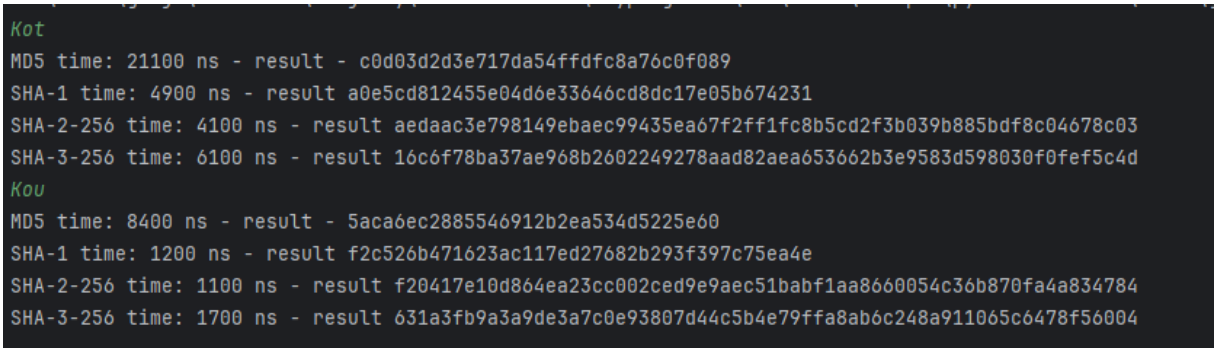
Laboratorium 3 – Funkcje Skrótu

1. Screenshot z aplikacji



2. Omówienie sposobu implementacji

Wykorzystana została wbudowana biblioteka hashlib, która zapewnia łatwy dostęp do popularnych funkcji skrótu.

Pomiar czasu operacji realizowany przez time.time_ns(), co pozwala na dokładne porównanie wydajności poszczególnych funkcji.

W eksperymentach SAC i badaniu kolizji wykorzystane zostało dodatkowe przetwarzanie – konwersję skrótów do postaci binarnej oraz losowe generowanie wejść.

3. Określenie roli soli w tworzeniu skrótów

Sol to losowy ciąg dodawany do hasła przed jego skróceniem.

Zapobiega to atakom przy użyciu precomputed hash tables (rainbow tables), ponieważ nawet dla takich samych haseł skrót będzie inny, jeśli sol jest inna.

4. Odpowiedź dotycząca bezpieczeństwa MD5

MD5 nie jest uważany za bezpieczny, ponieważ znane są praktyczne ataki kolizyjne.

Można znaleźć różne dane wejściowe, które generują ten sam skrót MD5, co czyni go nieodpowiednim do zastosowań wymagających wysokiego poziomu bezpieczeństwa.

Przykłady kolizji:

Kolizja: 845 dla aKNXWuL0 oraz oxWGso5m
Kolizja: 19c dla iVSTxTaE oraz eQVJ4mUW
Kolizja: 268 dla ELs0P1kP oraz k4v8iQS5

5. Wnioski

Porównanie szybkości: MD5 i SHA-1 są zazwyczaj najszybsze, ale kosztem bezpieczeństwa, natomiast SHA-256 i SHA3-256 są wolniejsze, ale bezpieczniejsze i generują dłuższe skróty.

Badanie kolizji na pierwszych 12 bitach: Przy analizie dużej liczby wejść można zauważyć, że kolizje na pierwszych 12 bitach występują stosunkowo wcześnie (przy około 4096 różnych wejściach) – co jest zgodne z teorią.

Właściwość SAC: Eksperyment pokazuje, że nawet niewielka zmiana wejścia (np. zmiana jednego znaku) skutkuje znaczną zmianą w skrócie, co potwierdza kryterium SAC.