Sprawozdanie

LABORATORIUM 5. SKRÓTY KRYPTOGRAFICZNE.

Zadanie 5.1. Wykorzystanie algorytmów nieodwracalnej funkcji skrótu go generowania skrótu wiadomości

P.5.1. Dla każdego z tych plików oblicz wartość funkcji skrótu za pomocą trzech wybranych algorytmów wykorzystując funkcjonalność openssl. Omów i porównaj otrzymane skróty wiadomości.

```
student@Kubuntu:~/Documents/lab7/zad1$ dd if=/dev/urandom of=900B.txt bs=1 count=900
900+0 records in
900+0 records out
900 bytes copied, 0,00288004 s, 312 kB/s
```

```
student@Kubuntu:~/Documents/lab7/zad1$ dd if=/dev/urandom of=900kB.txt bs=1kB count=900
900+0 records in
900+0 records out
900000 bytes (900 kB, 879 KiB) copied, 0,0172431 s, 52,2 MB/s
```

```
student@Kubuntu:~/Documents/lab7/zad1$ dd if=/dev/urandom of=100MB.txt bs=1MB count=100
100+0 records in
100+0 records out
100000000 bytes (100 MB, 95 MiB) copied, 0,72314 s, 138 MB/s
```

```
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -md5 900B.txt
MD5(900B.txt)= d6165d47623641c4113a809e36b3a88f
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sh1 900B.txt
dgst: Unrecognized flag sh1
dgst: Use -help for summary.
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha1 900B.txt
SHA1(900B.txt)= 3a0c4b44840ada0804dc6bb9f547125a163aafb7
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha256 900B.txt
SHA256(900B.txt)= e31f1a39625c2212b38dae9afeb62c399e627cc4e8660a37eca537e0caa59527
```

```
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -md5 900kB.txt
MD5(900kB.txt)= c66ba0bd25ad97f98266beb8223b0645
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha1 900kB.txt
SHA1(900kB.txt)= cd21b272e18cac445fde81464a88aa9e9064bcea
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha256 900kB.txt
SHA256(900kB.txt)= e5813981d6425f893e1756a606d994e0345e053abd3c88621ac378f4c019dd83
```

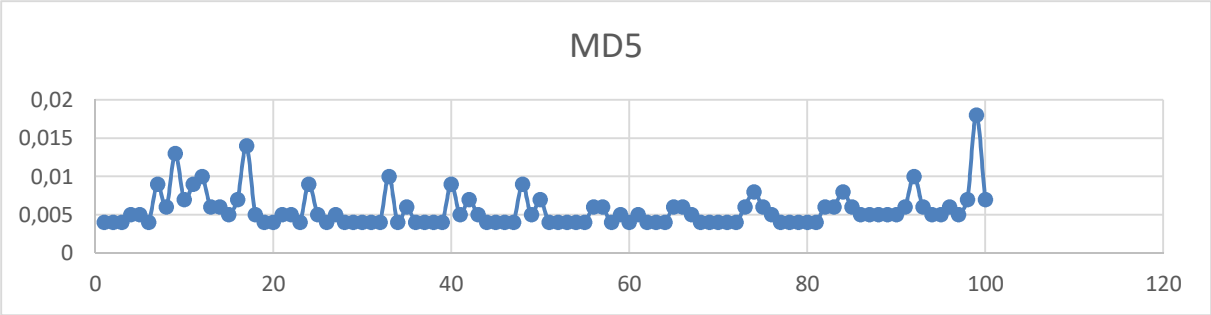
```
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -md5 100MB.txt
MD5(100MB.txt)= eaa184ddc81a192e115ca28ed55c5ad0
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha1 100MB.txt
SHA1(100MB.txt)= a8d1ad966e5853e1e169e7e39298fe09a1d3eb9a
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha256 100MB.txt
SHA256(100MB.txt)= 53957a864f67a3711d28241df710ea2aab9278190cb0535e0a876eb7777726e4
```

Algorytm sha256 jest najbezpieczniejszy, gdyż używa najdłuższej wartości skrótu.

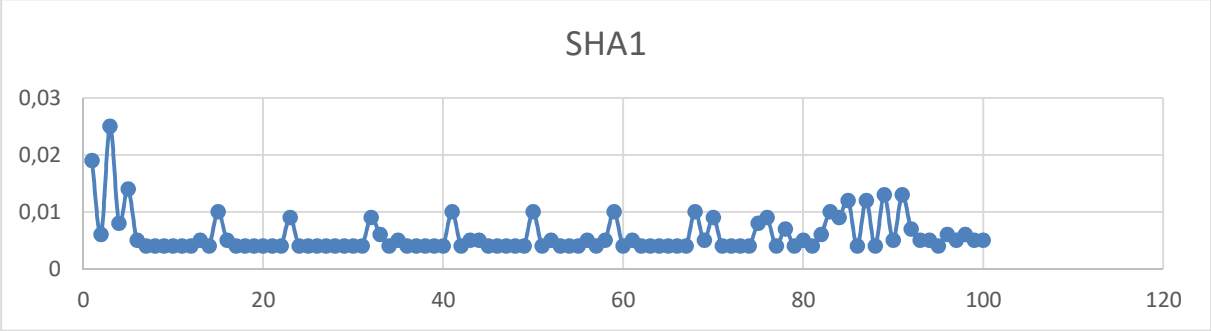
P.5.2. Określ średni czas przetworzenia 1MB danych dla każdej z kombinacji: rozmiar pliku – algorytm obliczenia skrótu. W celu określenia czasu wykonania polecenia skorzystaj z systemowego polecenia "time". Jako wynik zapisz sumy czasów user+sys. Wynik należy zapisać, jako średnia z minimum 100 pomiarów. Otrzymane wyniki umieścić w tabeli, a następnie sporządzić wykresy dla zastosowanych algorytmów. Przykładowe polecenie do wykonania 100 pomiarów.
Dla pliku 900B.txt:

Średni czas MD5: 0,00557s

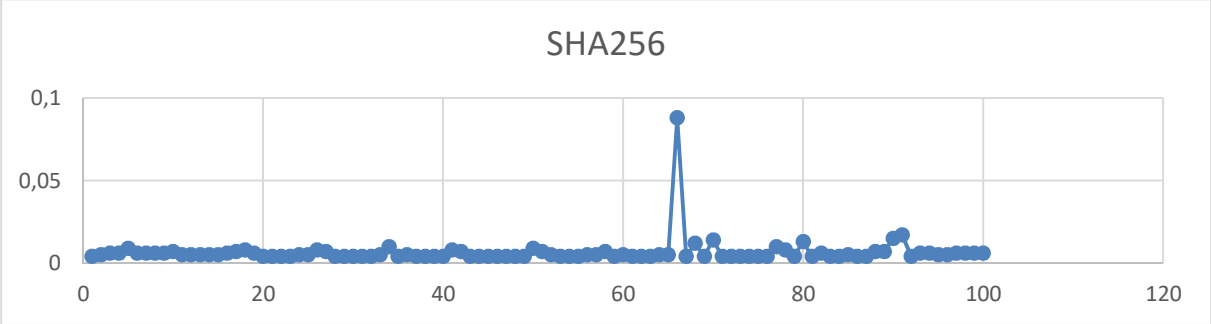
```
student@Kubuntu:~/Documents/lab7/zad1$ for n in {1..100}; do time openssl dgst -md5 900B.txt | grep real; done

real    0m0,012s
user    0m0,004s
sys     0m0,000s

real    0m0,004s
user    0m0,004s
sys     0m0,000s

real    0m0,013s
user    0m0,004s
sys     0m0,000s

real    0m0,004s
user    0m0,004s
sys     0m0,000s

real    0m0,014s
user    0m0,003s
sys     0m0,002s

real    0m0,015s
user    0m0,004s
sys     0m0,000s

real    0m0,007s
user    0m0,004s
sys     0m0,000s

real    0m0,014s
user    0m0,004s
sys     0m0,000s

real    0m0,013s
user    0m0,004s
sys     0m0,000s

real    0m0,005s
user    0m0,004s
sys     0m0,000s

real    0m0,013s
user    0m0,004s
sys     0m0,001s

real    0m0,012s
user    0m0,004s
sys     0m0,000s

real    0m0,011s
```



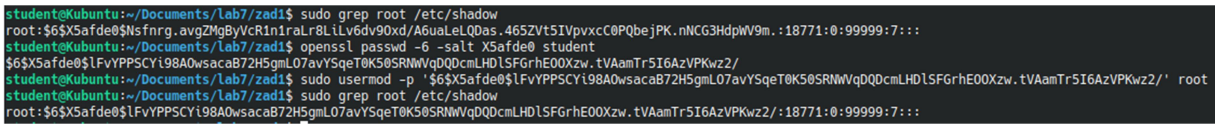
Średni czas SHA1: 0,00583s



Średni czas SHA256: 0,00646s



Zadanie 5.2. Funkcje skrótu a bezpieczeństwo przechowywanych haseł



P.5.3. Jakie zmiany zauważyłeś w pliku shadow? Zaloguj się na konta root lub student. Jakiego hasła użyłeś? Udokumentuj wykonanie ćwiczenia. Omów wykonane ćwiczenie i jego efekty. Czy dana metoda zmiany hasła jest bezpieczna? Jeśli nie zaproponuj zmiany i wyjaśnij dlaczego.

Do pliku shadow wprowadziliśmy nowe hasło dla roota. Metoda dobrze szyfruje hasło, więc jest bezpieczna jeżeli osoba trzecia nie widzi naszej konsoli.

Zadanie 5.3. Keyed Hash oraz HMAC

Tworzenie pliku:

```
student@Kubuntu:~/Documents/lab7/zad1$ dd if=/dev/urandom of=hmac.txt bs=1kB count=50
50+0 records in
50+0 records out
50000 bytes (50 kB, 49 KiB) copied, 0,000728431 s, 68,6 MB/s
```

Algorytm:

```
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -md5 -hmac "abcdefg" hmac.txt
HMAC-MD5(hmac.txt)= 788dc5d30168087f5503100eca99707e
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha1 -hmac "abcdefgh" hmac.txt
HMAC-SHA1(hmac.txt)= dbda55040eba89f55f000e7afbf6562e466b243d
student@Kubuntu:~/Documents/lab7/zad1$ openssl dgst -sha256 -hmac "abcdefghi" hmac.txt
HMAC-SHA256(hmac.txt)= 9a02534c4da169d66a0f6d20d2bcad4e7366ea6fd47b85d423a081863cf68861
```

P.5.4. Czy w algorytmie HMAC konieczne jest używanie kluczy o stałej długości?

Nie, klucze mogą być różnej długości.