

LUFFA

Jednokierunkowa Funkcja Skrótu - Implementacja Referencyjna

Aleksander Wiącek

3 października 2023

1 Wstęp

Luffa jest funkcją skrótu, stworzoną na potrzeby konkursu SHA-3, w którym zakwalifikowała się do drugiego etapu. W trakcie trwania konkursu funkcja ta została zaktualizowana do wersji Luffav2, w której zmieniono postać skrzynki podstawieniowej, kolejność wejścia i wyjścia bitów z poszczególnych słów do S-skrzynki oraz w rundzie finalizacyjnej - runda z pustą wiadomością wykonywana jest zawsze niezależnie od długości wiadomości do skrócenia.

Zaimplementowana została funkcja w wersji drugiej, generująca 256 bitową długość skrótu. Dopełnienie wiadomości następuje poprzez dopisanie do ostatniego bloku 256b sekwencji bitów 100...00 tak, aby spełniona została kongruencja:

$$L + 1 + K \equiv 0 \pmod{256}, L - \text{długość wiadomości}, K - \text{ilość zer.}$$

Długość dopełnionej wiadomości powinna być zatem wielokrotnością liczby 256.

Funkcja rundy algorytmu, przyjmuje 256 bitową długość bloku wiadomości oraz trzy bloki wartości pośrednich również po 256 bitów każdy. W pierwszej rundzie bloki te są wartościami inicjalizującymi 'IV' postaci:

$V_{0,0} = 0x6d251e69$	$V_{0,1} = 0x44b051e0$	$V_{0,2} = 0x4eaa6fb4$	$V_{0,3} = 0xdbf78465$
$V_{0,4} = 0x6e292011$	$V_{0,5} = 0x90152df4$	$V_{0,6} = 0xee058139$	$V_{0,7} = 0xdef610bb$
$V_{1,0} = 0xc3b44b95$	$V_{1,1} = 0xd9d2f256$	$V_{1,2} = 0x70eee9a0$	$V_{1,3} = 0xde099fa3$
$V_{1,4} = 0x5d9b0557$	$V_{1,5} = 0x8fc944b3$	$V_{1,6} = 0xcf1ccf0e$	$V_{1,7} = 0x746cd581$
$V_{2,0} = 0xf7efc89d$	$V_{2,1} = 0x5dba5781$	$V_{2,2} = 0x04016ce5$	$V_{2,3} = 0xad659c05$
$V_{2,4} = 0x0306194f$	$V_{2,5} = 0x666d1836$	$V_{2,6} = 0x24aa230a$	$V_{2,7} = 0x8b264ae7$

Wartości pośrednie w funkcji rundy przechodzą przez dwie warstwy: Message Injection oraz przez warstwę nieliniowej permutacji. W wersji 256 bitowej algorytmu warstwa nieliniowej permutacji składa się z 3 bloków Qj , po jednym dla każdego bloku stanu pośredniego. Ostatnią operacją w Qj jest dodanie stałych do zerowego i czwartego słowa bloku stanu pośredniego. W implementacji stałe te są obliczane za pomocą funkcji przedstawionej w specyfikacji, jednak równie dobrze, wszystkie stałe mogłyby być przechowywane w pamięci.

2 Wyniki implementacji

Stworzono dwie wersje programu. Pierwszy plik wykonywalny luffa256.exe wypisuje tylko nazwę skracanego pliku, jego rozmiar, wartość heksadecymalną skrótu, jego długość oraz czas działania w sekundach.

Natomiast plik luffa256_show.exe to ten sam algorytm z tym wyjątkiem, że wypisuje wszystkie ważne wartości pośrednie.

Wyniki dla pliku binarnego zawierającego ciąg znaków 'abc':

```

Wiersz polecenia

D:\Microsoft_VS_Code\proj\LUFFA>luffa256.exe test.bin out.bin
Length of hash in bytes: 32

File name: test.bin
test.bin size: 3 B

HASH:
f29311b87e9e40de7699be23fbef5a47cb16ea4f5556d47ca40c12ad764a73bd

Hash length: 32 B
Time: 0.009000 [s]
D:\Microsoft_VS_Code\proj\LUFFA>

```

Rys. 1. Wynik działania luffa256.exe

```

Wiersz polecenia

D:\Microsoft_VS_Code\proj\LUFFA>luffa256_show.exe test.bin out.bin
Length of hash in bytes: 5

File name: test.bin
test.bin size: 3 B

Message block:
61626380 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Round 0

Message Injection:
H(0) block
Internal of 0 : 2dfbf234
Internal of 1 : 3c72435c
Internal of 2 : 8e729b83
Internal of 3 : c00ee149
Internal of 4 : e70e280f
Internal of 5 : a0a111fd
Internal of 6 : 97b4f048
Internal of 7 : db457d86
H(1) block
Internal of 8 : e208c448
Internal of 9 : c0720760

```

Rys. 2. Wynik działania luffa256_show.exe cz.1

```

C9271859 3C281083 CB48810D FCB0B80C 930
HASH:
Z0 = f29311b8
Z1 = 7e9e40de
Z2 = 7699be23
Z3 = fbeb5a47
Z4 = cb16ea4f
Z5 = 5556d47c
Z6 = a40c12ad
Z7 = 764a73bd

f29311b87e

Hash length: 5 B
Time: 0.470000 [s]
D:\Microsoft_VS_Code\proj\LUFFA>

```

Rys. 3. Wynik działania luffa256_show.exe cz.2

Wartości wektora testowego:

C-2 *Luffa-256*

The message digest of the message “abc” is

$$\begin{aligned}
 Z_{0,0} &= 0xf29311b8, & Z_{0,1} &= 0x7e9e40de, \\
 Z_{0,2} &= 0x7699be23, & Z_{0,3} &= 0xfbeb5a47, \\
 Z_{0,4} &= 0xcb16ea4f, & Z_{0,5} &= 0x5556d47c, \\
 Z_{0,6} &= 0xa40c12ad, & Z_{0,7} &= 0x764a73bd.
 \end{aligned}$$

Rys. 4. Wektor testowy dla luffa256

Test dla wiadomości: 'abcdbcdecdefdefgefghfghighijhijkijkljklm':

```

Wiersz polecenia

D:\Microsoft_VS_Code\proj\LUFFA>luffa256.exe test2.bin out2.bin
Length of hash in bytes: 32

File name: test2.bin
test2.bin size: 40 B

HASH:
369114dce84af18e8e0178b8b6f87af0e4b5c1db16400bc75a0d035c5efa9ec3

Hash length: 32 B
Time: 0.009000 [s]
D:\Microsoft_VS_Code\proj\LUFFA>

```

Rys. 5. Wynik działania luffa256.exe dla testu 2

```

Wiersz polecenia

D:\Microsoft_VS_Code\proj\LUFFA>luffa256_show.exe test2.bin out2.bin
Length of hash in bytes: 8

File name: test2.bin
test2.bin size: 40 B

Message block:
61626364 62636465 63646566 64656667 65666768 66676869 6768696a 68696a6b

Round 0

Message Injection:
H(0) block
Internal of 0 : 2dfbf2d0
Internal of 1 : 5e112739
Internal of 2 : ed16fee5
Internal of 3 : a46b872e
Internal of 4 : 82684f67

```

Rys. 6. Wynik działania luffa256.exe_show dla testu 2 - blok 1

```

Message block:
696a6b6c 6a6b6c6d 80000000 00000000 00000000 00000000 00000000 00000000

Round 1

Message Injection:
H(0) block
Internal of 0 : ce5a1f4f
Internal of 1 : 97b83cb5
Internal of 2 : 94d23948
Internal of 3 : 764e100a
Internal of 4 : ea57b015
Internal of 5 : e52ca248
Internal of 6 : 7497cd58
Internal of 7 : 5a1d9f87
H(1) block
Internal of 8 : ec2c8e24
Internal of 9 : 23b206ec

```

Rys. 7. Wynik działania luffa256_show.exe dla testu 2 - blok 2

```

FINALIZATION

Message block:
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Message Injection:
H(0) block
Internal of 0 : a608764
Internal of 1 : 9cc6835
Internal of 2 : 8b9a3135
Internal of 3 : 67078b2b
Internal of 4 : d08be43e
Internal of 5 : 13366270
Internal of 6 : 7a585380
Internal of 7 : ee4a906c
H(1) block
Internal of 8 : 8db88ce8
Internal of 9 : 8292936
Internal of 10 : 83abd08b
Internal of 11 : c28c8870
Internal of 12 : 5d855f74
Internal of 13 : d5a98969

```

Rys. 8. Wynik działania luffa256_show.exe dla testu 2 - finalizacja

```

HASH:
Z0 = 369114dc
Z1 = e84af18e
Z2 = 8e0178b8
Z3 = b6f87af0
Z4 = e4b5c1db
Z5 = 16400bc7
Z6 = 5a0d035c
Z7 = 5efa9ec3

369114dce84af18e

Hash length: 8 B
Time: 0.752000 [s]
D:\Microsoft_VS_Code\proj\LUFFA>

```

Rys. 9. Wynik działania luffa256_show.exe dla testu 2 - rezultat

Wynik podany przez autorów funkcji dla skracanej wiadomości:

```

intermediate_value_256.txt — Notatnik
Plik  Edycja  Format  Widok  Pomoc

-----

Z[0] = c95fd853 ^ 9efd8775 ^ 61334bfa = 369114dc
Z[1] = 02709cf7 ^ 68080f1c ^ 82326265 = e84af18e
Z[2] = a86bf4df ^ c7f4ee64 ^ e19e6203 = 8e0178b8
Z[3] = e28dc67e ^ 98c976f3 ^ ccbcca7d = b6f87af0
Z[4] = ea2851a9 ^ 1d6ccb8b ^ 13f15bf9 = e4b5c1db
Z[5] = e1cc0f45 ^ 6fcff7ee ^ 9843f36c = 16400bc7
Z[6] = d6db9171 ^ d08acf82 ^ 5c5c5daf = 5a0d035c
Z[7] = 8253ed11 ^ 6612312a ^ babb42f8 = 5efa9ec3

Message Digest: 369114dc e84af18e 8e0178b8 b6f87af0
                  e4b5c1db 16400bc7 5a0d035c 5efa9ec3

```

Rys. 10. Rezultat dla 'abcdbcdecdefdefgefghfghighijhijkijkljklm'