UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI

FACULTATEA DE INGINERIE MEDICALĂ

# INTRODUCERE AUTOMATĂ DE DATE

Prof. Coord.: Ing. Cristian-Alexandu TĂNASE

Studenți: Alina-Gabriela CARACUDĂ

Ioana-Roxana RÎCU

Grupa: 1445

# CUPRINS

1. INTRODUCERE ..... 3
2. MOD DE FUNCȚIONARE ..... 4
3. APLICAȚII ..... 6
3.1 Router centrat pe informații Libswift-PPSPP: Accelerator SHA1 [3] ..... 6
3.2 EAACK eficient energetic bazat pe SHA1 pentru MANET [4] ..... 6
4. MOD DE LUCRU ..... 8
5. BIBLIOGRAFIE ..... 11
ANEXĂ ..... 12

# 1. INTRODUCERE

Criptografia și securitatea unei rețele au concepte pentru a securiza comunicarea client-server prin rețeaua wireless. În prezent, securitatea Internetului a fost un interes în domeniul tehnologiei informației, în special în mediul cloud. Conform, pentru a securiza datele într-un mediu cloud, poate fi implementată o schemă de criptare complet homomorfă (FHE) cu eficiență îmbunătățită. Funcția hash criptografică este una dintre soluțiile pentru a proteja datele transferate. Este o componentă pentru aplicații de securitate pe internet cu diverse semnificații. Mai mult, acceptă orice lungime a mesajelor și o analizează în valoare de lungime fixă, care se numește abstractizare a informațiilor. Hash-ul nu poate fi inversat spre deosebire de criptare. Are o utilizare principală în susținerea integrității și confidențialității mesajelor și, din această cauză, algoritmii hash sunt utilizați în special în autentificarea autentificărilor și verificarea fișierelor. În plus, algoritmii de hashing au elemente esențiale în numeroase aplicații și practici de securitate. SHA-1 generează un mesaj hash bazat pe principii legate de cele folosite de Ronald L. Rivest care a proiectat algoritmii MD4 și MD5. Algoritmul a fost realizat ca parte a proiectului Capstone al Guvernului SUA și este utilizat pe scară largă în securizarea comunicațiilor client-server și a unei varietăți de aplicații. [1]

Secure Hash Algorithm (SHA 1), este un algoritm de hashing criptografic utilizat pentru a verifica autenticitatea și integritatea datelor. Diferențele acestui algoritm sunt de obicei aplicate de autoritățile de certificare SSL pentru a semna certificate și semnături digitale. Acest algoritm hash garantează că datele site-ului web nu sunt modificate sau modificate. Acesta generează un cod hash unic fix de 160 de biți de la orice dimensiune de fișier. Pe baza acestor valori hash, se poate verifica dacă fișierul a fost sau nu modificat prin compararea valorii hash generată din valoarea hash stocată în baza de date. Dar, SHA1 nu este perfect datorită valorii hash scurte. Pot fi ușor spart. Mai mult, cercetătorii au ajuns la primul atac de coliziune împotriva funcției hash SHA1, generând două fișiere PDF diferite. [1]

# 2. MOD DE FUNCȚIONARE

SHA 1 este un algoritm de securitate dezvoltat de Institutul Național de Standarde și Tehnologie (NIST) pe baza rezultatelor patch-urilor din algoritmul SHA. SHA1 este algoritmul hash principal bazat pe principiul algoritmului MD4. Deoarece produce o ieșire de 160 de biți, ceea ce face ca SHA1 să aibă nevoie de cinci registre de pe 32 de biți. Dar metoda de digerare a mesajelor și de completare a datelor funcționează ca algoritmul MD5. SHA are 4 runde principale care se repetă și mai multe ture au operatii de 20 de trepte. Acest algoritm ia ca mesaj de intrare cu o lungime maximă mai mică de 264 de biți și produce o ieșire de 160 de biți, așa cum se arată în Fig. 1.1. Intrările sunt procesate în blocuri de 512 biți. [1]

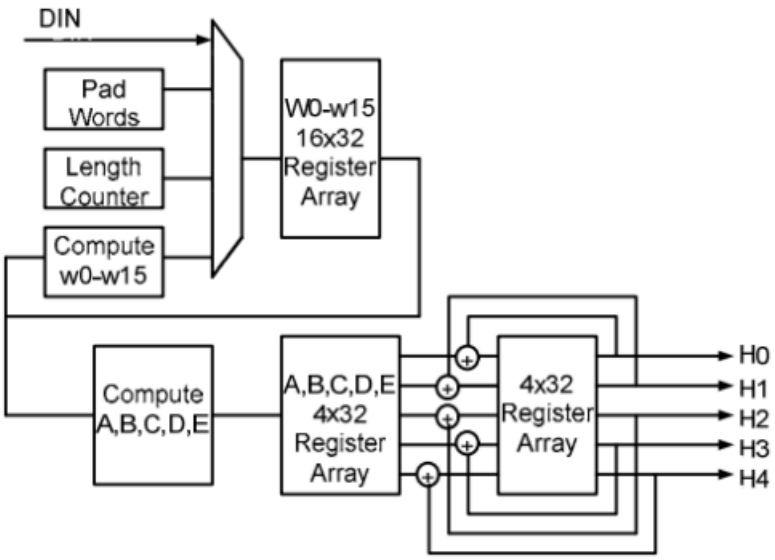


Fig. 1.1 Algoritm SHA1 [2]

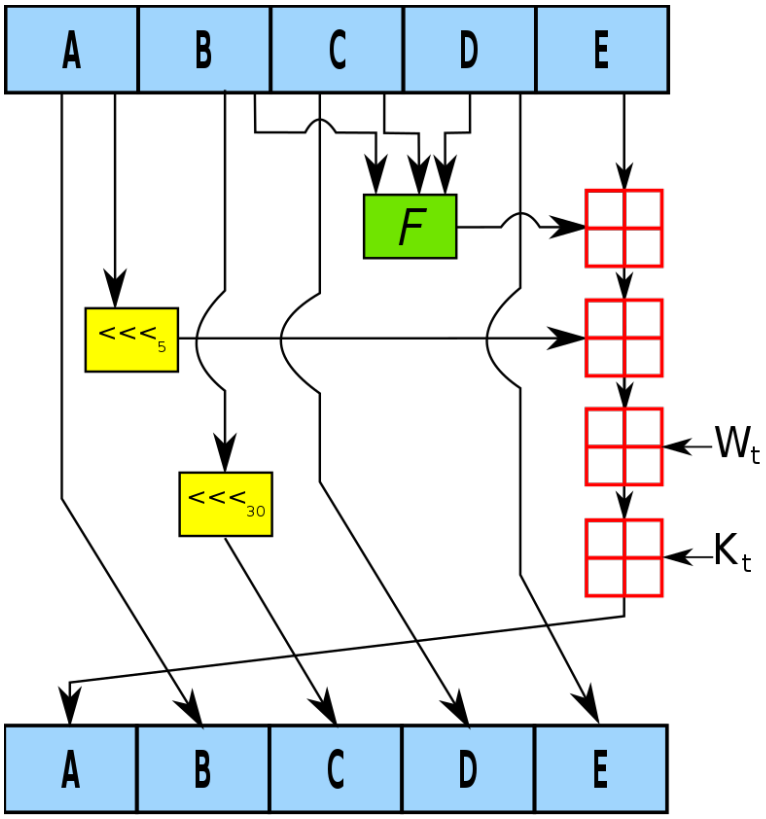


Fig. 1.2 O iterație în cadrul funcției de compresie SHA-1: A, B, C, D și E sunt cuvinte pe 32 de biți ale stării; F este o funcție neliniară care variază; $\lll_n$ indică o rotație a biților la stânga cu n locuri; n variază pentru fiecare operație; Wt este cuvântul mesaj extins din runda t; Kt este constanta rotundă a rundei t; $\boxplus$ denotă adiție modulo 232. [5]

# 3. APLICAȚII

## 3.1 Router centrat pe informații Libswift-PPSPP: Accelerator SHA1 [3]

Streamingul de conținut peer-to-peer este metoda de livrare de conținut de generație următoare, făcând ca modelul de comunicare client-server să fie depășit și nu este sustenabil în mediul actual al internetului. Un efort de standardizare a acestui nou protocol este reprezentat de Rețeaua centrată pe informații (ICN) LibswiftPPSPP (Peer to Peer Streaming Peer Protocol). Se descrie protocolul, se oferă un set de cerințe pentru routerul generic și se identifică un subset care se va implementa. Folosind platforma de dezvoltare NetFPGA, se construiește un accelerator de calcul hash (SHA1), care este unul dintre blocurile fundamentale pentru routerul Libswift ICN. Măsurătorile arată că prototipul depășește un CPU de uz general în calculul hash. Viteza măsurată este de 1,54, iar suprafața este mică, având în vedere sistemul complet. Se discută posibilele optimizări și impactul acestor optimizări este comparat cu implementarea. [3]

## 3.2 EAACK eficient energetic bazat pe SHA1 pentru MANET [4]

Îmbunătățirea tehnologiei și reducerea costurilor hardware au fost încurajatoare să folosim tehnologia wireless în locul celei cu fir. MANET (Mobile Adhoc Network) este cea mai cunoscută și utilizată tehnologie wireless. În MANET, toate nodurile sunt mobile în natură. Comunicarea directă are loc atunci când două noduri intră în raza de comunicare unul de celălalt. Uneori, comunicarea are loc prin noduri intermediare dacă emițătorul și receptorul nu se află în raza de comunicare. Chiar dacă MANET are caracteristici foarte bune, are și probleme. Puține noduri intermediare nu transmit date pentru a-și economisi energia, astfel de noduri sunt denumite noduri egoiste. Prezența nodurilor egoiste scade raportul de livrare a pachetelor (PDR) și degradează performanța MANET. Deoarece MANET este tehnologie wireless, este vulnerabil la atacurile de rețea. Pentru a salva MANET de atacurile de rețea sunt folosite IDS (Intrusion Detection Systems). S-a folosit sistemul EAACK. EAACK este un sistem IDS bazat pe confirmare. EAACK suferă de problema falsificării recunoașterii. Sistemul utilizat propus utilizează SHA1 pentru a proteja recunoașterea de falsificare. În cele din urmă, s-a comparat EAACK bazat pe SHA1 cu EAACK

bazat pe RSA în ceea ce privește raportul de livrare a pachetelor (PDR) și consumul mediu de energie (AEC). [4]

# 4. MOD DE LUCRU

În domeniul criptografiei, funcțiile hash reprezintă un instrument esențial pentru securitatea informatică. Printre acestea, SHA-1 (Secure Hash Algorithm 1) este o funcție hash larg utilizată, care transformă o intrare de orice dimensiune într-un rezumat de 160 de biți, echivalent cu 20 de octeți. Acest rezumat, cunoscut și sub numele de rezumat al mesajului, este de obicei reprezentat ca o secvență de 40 de cifre hexazecimale.

Scopul nostru este de a introduce toate literele alfabetului (de la 'a' la 'z') în funcția SHA-1 folosind Python și de a salva rezultatele obținute într-un fișier Excel. Astfel, vom putea observa cum fiecare literă este transformată într-un hash SHA-1 unic.

Codul implementat folosește bibliotecile ‘hashlib’ și ’pandas’ pentru a calcula hash-urile SHA-1 pentru literele alfabetului și pentru a le salva într-un fișier Excel după următorii pași:

1. Se importă bibliotecile necesare: ’hashlib’ și ’pandas’.
2. Se definește ’alphabet’ ca un șir de caractere care conține literele alfabetului de la 'a' la 'z'.
3. Se creează o listă goală ’results’ care va conține rezultatele calculului hash-ului SHA-1.

```
import hashlib
import pandas as pd

alphabet = "abcdefghijklmnopqrstuvwxyz"

results = []
```

Fig. 4.1

4. Se parcurge fiecare literă din alphabet folosind un loop for.
5. Pentru fiecare literă, se calculează hash-ul SHA-1 utilizând ’hashlib.sha1()’ și ’hexdigest()’. Apoi, ’letter.encode()’ convertește litera într-un șir de bytes, deoarece funcția’ hashlib.sha1()’ primește ca argument un șir de bytes.

6. Rezultatul hash-ului SHA-1 este adăugat în lista ’results ’ sub forma unei liste ’ [letter, sha1_hash] ’.
7. După ce s-a terminat bucla, lista ’results’ este transformată într-un obiect ’DataFrame’ utilizând ’pd.DataFrame()’. Se specifică numele coloanelor ca ’ ["Letter", "SHA1 Hash"]’.
8. Obiectul ’DataFrame’ este salvat într-un fișier Excel cu numele "sha1_results1.xlsx" folosind metoda ’to_excel()’. Argumentul ’index=False’ este utilizat pentru a evita salvarea indexului ’DataFrame’ în fișierul Excel.

```
for letter in alphabet:
    sha1_hash = hashlib.sha1(letter.encode()).hexdigest()

    results.append([letter, sha1_hash])

df = pd.DataFrame(results, columns=["Letter", "SHA1 Hash"])

df.to_excel("sha1_resultate.xlsx", index=False)
```

Fig. 4.2

9. Acest cod va genera un fișier Excel cu două coloane: "Letter" și "SHA1 Hash", conținând rezultatele calculului hash-ului SHA-1 pentru fiecare literă a alfabetului.

Tabel 1 Rezultate

| Letter | SHA1 Hash |
|---|---|
| a | 86f7e437faa5a7fce15d1ddcb9eaeaea377667b8 |
| b | e9d71f5ee7c92d6dc9e92ffdad17b8bd49418f98 |
| c | 84a516841ba77a5b4648de2cd0dfcb30ea46dbb4 |
| d | 3c363836cf4e16666669a25da280a1865c2d2874 |
| e | 58e6b3a414a1e090dfc6029add0f3555ccba127f |
| f | 4a0a19218e082a343a1b17e5333409af9d98f0f5 |
| g | 54fd1711209fb1c0781092374132c66e79e2241b |

| | |
|---|---|
| h | 27d5482eebd075de44389774fce28c69f45c8a75 |
| i | 042dc4512fa3d391c5170cf3aa61e6a638f84342 |
| j | 5c2dd944dde9e08881bef0894fe7b22a5c9c4b06 |
| k | 13fbd79c3d390e5d6585a21e11ff5ec1970cff0c |
| l | 07c342be6e560e7f43842e2e21b774e61d85f047 |
| m | 6b0d31c0d563223024da45691584643ac78c96e8 |
| n | d1854cae891ec7b29161ccaf79a24b00c274bdaa |
| o | 7a81af3e591ac713f81ea1efe93dcf36157d8376 |
| p | 516b9783fca517eecbd1d064da2d165310b19759 |
| q | 22ea1c649c82946aa6e479e1ffd321e4a318b1b0 |
| r | 4dc7c9ec434ed06502767136789763ec11d2c4b7 |
| s | a0f1490a20d0211c997b44bc357e1972deab8ae3 |
| t | 8efd86fb78a56a5145ed7739dcb00c78581c5375 |
| u | 51e69892ab49df85c6230ccc57f8e1d1606caccc |
| v | 7a38d8cbd20d9932ba948efaa364bb62651d5ad4 |
| w | aff024fe4ab0fece4091de044c58c9ae4233383a |
| x | 11f6ad8ec52a2984abaafd7c3b516503785c2072 |
| y | 95cb0bfd2977c761298d9624e4b4d4c72a39974a |
| z | 395df8f7c51f007019cb30201c49e884b46b92fa |

# 5. BIBLIOGRAFIE

[1] Analysis of Cybersecurity Standard and framework components - proquest. (n.d.). https://www.proquest.com/openview/fd5b07154b6b748faab729a62a6b1264/1?pq-origsite=gscholar&cbl=52057

[2] Ignatius Moses Setiadi, D. R., Faishal Najib, A., Rachmawanto, E. H., Atika Sari, C., Sarker, M. K., & Rijati, N. (2019). A Comparative Study MD5 and SHA1 Algorithms to Encrypt REST API Authentication on Mobile-based Application. 2019 International Conference on Information and Communications Technology (ICOIACT). doi:10.1109/icoiact46704.2019.893

[3] MSC thesis - TU delft. (n.d.-b). http://ce-publications.et.tudelft.nl/publications/1424_libswiftppspp_information_centric_router_sha1_accelerator.pdf

[4] Energy efficient EAACK based on SHA1 for manet - researchgate. (n.d.-b). https://www.researchgate.net/profile/Rahul-Joshi-9/publication/319097041_Energy_efficient_EAACK_based_on_SHA1_for_MANET/links/5c8136e7299bf1268d444b70/Energy-efficient-EAACK-based-on-SHA1-for-MANET.pdf

[5] Wikimedia Foundation. (2023, May 5). SHA-1. Wikipedia. https://en.wikipedia.org/wiki/SHA-1#:~:text=In%20cryptography%2C%20SHA-1%20(,rendered%20as%2040%20hexadecimal%20digits.

# ANEXĂ

```
import hashlib

import pandas as pd

alphabet = "abcdefghijklmnopqrstuvwxyz"

results = []

for letter in alphabet:

    sha1_hash = hashlib.sha1(letter.encode()).hexdigest()

    results.append([letter, sha1_hash])

df = pd.DataFrame(results, columns=["Letter", "SHA1 Hash"])

df.to_excel("sha1_resultate.xlsx", index=False)
```