Lab 7 - Moroianu Theodor 334

Ex 1

A. Adevarat. Este imposibil sa luam o prajitura si sa o schimbam inapoi in faina / lapte / oua / unt etc.

B. Fals. in 2004 s-a aratat ca MD5 nu este sigur la coliziuni.

C. Adevarat. SHA256 are output de 256 de biti.

D. Adevarat.

E. Fals. Criptarea presupune ca mesajul criptat poate fi adus inapoi la starea initiala prin decriptare.

F. Adevarat. Daca calcularea unui hash este prea costisitoare computational, atunci nu este o functie hash buna.

G. Fals. Hash-ul dat este MD5("parola123").

Ex 2

Pentru a decompresa arhiva, putem executa:

```
$ unzip SSI\ L7\ -\ Reference-Implementation.zip
```

Comentam linia din `photondriver.c` care adauga sirul initial in fisierul de hashuri:

```
// fprintf(f, "%s ::::: ", line);
```

Pentru a compila codul, executam:

```
$ cc -D_PHOTON80_ -D_TABLE_ photon.c photondriver.c sha2.c timer.c -o
photon80 -O3
```

Pentru a ne genera date de test in fiiserul `input.txt`, rulam:

```
$ python -c "print('\n'.join([str(i) for i in range(10**7)]))" >
input.txt
```

Acum rulam `photon80`:

```
$ time ./photon80 -f
> ________________________________________________________
> Executed in   25.17 secs    fish           external
> usr time   24.61 secs  735.00 micros   24.61 secs
> sys time    0.38 secs  900.00 micros    0.38 secs
```

Acum verificam daca exista colisiuni:

```
$ sort < output.txt | uniq -D
```

`uniq` nu intoarce nimic, deci nu exista nicio coliziune (toate liniile din `output.txt` sunt distincte).

Asadar, NU am gasit nicio coliziune.

Acest lucru este de asteptat, avand in vedere ca avem hashuri pe `80` de biti. Asadar, din paradoxul zilelor de nastere, este nevoie de aproximativ 2^{40} care este aproximativ 10^{12}. Noi avem numai 10^7 hasuri.

Ex 3

Exemplu 1

Exista mai multe probleme:

- Cheia privata este hardcodata in cod. Asta inseamna ca:
  - codul nu poate fi facut public.
  - Orice modificare a cheiei necesita recompilarea codului.
  - Sysadminii nu pot verifica in mod reliable care este cheia folosita (in caz ca sunt mai multe).
- Noi dorim sa calculam **HAASH**-urile parolelor, NU sa le encriptam. Nu dorim ca atunci cand cineva afla `bettysuper`, sa poata decripta parolele inapoi in plaintext.

Exemplu 2

- Hashuieste si username-ul, care poate fi problematic in cazul unei coliziuni.
- Cu timing attack, putem afla id-ul unui user (in ce pozitie apare in vectorul `listaConturi`).

Exemplu 3

- Nu adauga salt la hash.

Exemplu 4

- Salt-ul ar trebui sa fie generat pentru fiecare utilizator in parte.
- Ideal, ar trebui toate stringurile sa fie ASCII.

Exemplu 5

- Foloseste encodare `ASCII`. Totusi, parolele pot fi si de tip `unicode` / `utf-8`.

- Nu adauga salt la parola.
- Foloseste `md5`, care e slab la coliziuni.