Tema – Securitatea functiei hash PHOTON-80/20/16

Informatii:

- Tema se realizeaza individual;
- Tema presupune verificarea securitatii unei functii hash definita pentru dispozitivele cu capacitate reduse (*constrained devices*) – *PHOTON Lightweight Hash Functions;* Mai exact:

  Puteti sa gasiti un alt input care genereaza o valoare hash data?

  Altfel spus, testam *second-preimage resistance* sau mai general *collision resistance*.
- Mai multe informatii despre familia de functii hash PHOTON:
  - [1] https://sites.google.com/site/photonhashfunction/design
  - [2] https://eprint.iacr.org/2011/609.pdf
- Din familia de functii hash PHOTON, tema se refera strict la prima varianta, cu output pe 80 de biti: *PHOTON-80/20/16;*

Pasi de urmat:

- Download *Reference-Implementation.zip* de pe Moodle; **Atentie!** Codul sursa este modificat fata de cel postat la [1] pentru a permite calculul functiilor hash pentru valori oarecare, citite din fisierul *input.txt;* Mai exact, se va calcula valoarea hash pentru fiecare linie din fisierul *input.txt,* si se va scrie in fisierul *output.txt*;
  Ca exemplu de fisiere de input si output, vedeti *input_test.txt* si *output_test.txt*, postate pe Moodle;
- **Atentie!** Folositi **implementarea bazata pe tabele** (*table-based implementation*), este mult mai rapida, si versiunea functiei hash pe **80 de biti**. Mai exact:

```
icc / gcc / clang / etc. -D_PHOTON80_ -D_TABLE_ photon.c
photondriver.c -o photon80 sha2.c timer.c -O3
```

- Ca valori estimative de executie, pentru 1,6 GHz Intel Core i5, 2 cores - `231 cycles per byte` **vs.** `–1798 cycles per byte.` Pentru verificarea vitezei de executie:

```
./photon80 -s
```

Atentie! Timpul de executie depinde in mod direct de valoarea cycles per byte, deci o valoare cat mai mica va permite sa calculati mai repede un numar mai mare de hash-uri;

- Generati-va un fisier *input.txt* care sa contina **pe fiecare linie cate o valoare distincta** (pentru aceasta se va calcula hash-ul). Un mod simplu de generare este sa plecati cu o valoare initiala (un cuvant, o propozitie scurta, etc.) si sa

adaugati la final un contor; orice alta modalitate de generare a fisierului este acceptata, cat timp valorile sunt **distincte si diferite de cele din *input_test.txt*;**
- Daca vreti sa calculati si sa stocati functiile hash pentru valorile din *input.txt* in *output.txt* folositi:

```
./photon80 -f
```

- Ca valori extimative, pentru un fisier *output.txt* (rulat la 231 cycles per bye, cu input de tip *photon<contor>*):
  - 1 000 000 linii – 40 MB - timp de executie: aprox. 10 sec
  - 10 000 000 linii – 410MB – timp de executie: aprox. 1 min
  - 100 000 000 linii – 4.2GB – timp de executie: 12 min

- Verificati pentru fiecare hash obtinut egalitatea cu fiecare hash din *output_test.txt*; **Atentie!** Devine dificil sa lucrati cu fisiere mari, deci puteti face verificarea direct la generarea valorii hash, fara stocarea acesteia in fisier. Analog, puteti face generarea string-ului de input direct, fara stocare si apoi citire din fisierul *input.txt;*

- **Activitatea voastra consta in generarea a 100 000 000 de valori de input, si verificarea egalitatii hash-ului cu valorile hash din *output_test.txt*.** In cazuri exceptionale, daca timpul de executie nu poate fi redus sau aveti probleme cu capacitatea de stocare, se accepta si un numar mai mic. Verificarea unui numar mai mare de input-uri este apreciata.

Date importante / termene limita:
- Transmiterea temelor: **20 aprilie, ora 23:55**

Mod de transmitere al temelor:
- Temele trebuie transmise prin Moodle.
- Studentii care nu transmit tema pana la termenul limita pierd punctajul aferent acesteia.
- Temele se transmit prin Moodle (prin completarea unui formular) conform cu specificatiile postate. Acestea trebuie sa contina:

  - **Am gasit o coliziune**: DA / NU; (daca da, atunci indicati inputurile si hash-ul)
  - **Viteza**: …. cycles per byte
  - **Numarul de testari: …**
  - **Mod de generare input:** (explicati care sunt si cum au fost generate valorile din *input.txt*) …
  - **Output (10-15 valori):** Primele 10 linii ale fisierului *output.txt* generat (chiar daca nu folositi fisiere, generate output-ul pentru primele 10 input-uri in aceeasi forma).

Exemplu de raspuns (pastrati aceeasi structura si denumirile campurilor):

  - **Am gasit o coliziune**: NU;

- **Viteza**: 231 cycles per byte;
- **Numarul de testari**: 100 000 000;
- **Mod de generare input:** am plecat de la “photon”, apoi am concatenat valoarea unui contor;
- **Output (10-15 valori):**

S = 4, D = 5, Rate = 20, Ratep = 16, DigestSize = 80
photon ::::: 6c2bf8cad416a071d11e
photon0 ::::: 9cdcc99b9e68cbb273d5
photon1 ::::: 22a5a77ccc941ea042cf
photon2 ::::: 8e0368802e8a1d1f4c82
photon3 ::::: f41198894e0d8b74a0e8
photon4 ::::: a62e7dea24ac2eb9fee6
photon5 ::::: 5d06b6668377b1f656a1
photon6 ::::: ee7276c10429a02deda9
photon7 ::::: eb64cbee5d53640eafc9
photon8 ::::: 03ac754af5f6e6072798
photon9 ::::: f9d99e0b5fcb2083a6b9

Criterii de notare:

- Transmiterea temei;
- Transmiterea tuturor informatiilor cerute;
- Transmiterea informatiei in formatul cerut.