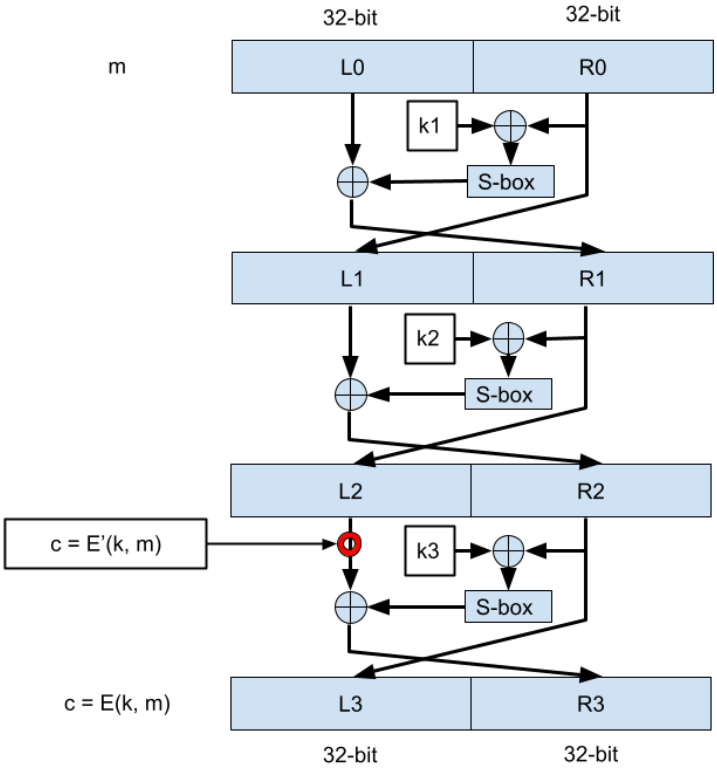
Temă IC 2022: Differential Cryptanalysis

În această temă vom implementa atacul Differential Cryptanalysis[1] publicat de Eli Biham și Adi Shamir (doi mari criptografi) în 1992.

Pentru temă, am dezvoltat un block cipher simplificat (IC Encryption Cipher – ICEC) cu bloc de 64 biți și cheia pe 96 biți, implementat cu 3 runde, fiecare rundă fiind o simplificare a rundelor Feistle din DES, și folosind S-box-urile din AES.
Schema de operație este cea din figura de mai jos, unde fiecare cheie de rundă (k1, k2, k3) este pe 32 biți (4 octeți) și împreună formează cheia principală (k = k1 | k2 | k3):



În fiecare rundă, la intrare avem 64 biți (8 octeți) împărțiți în două jumătăți, L_(i-1) și R_(i-1), unde i este indexul rundei (i=1, 2, 3). Putem calcula ușor starea la sfârșitul rundei, sub forma celor două jumătăți L_i și R_i, cu formulele:

L_i = R_(i-1)
R_i = L_(i-1) XOR S-box(k_i XOR R_(i-1))
cu mențiunea că operația de S-box (folosim S-box din AES, care primește 8 biți și returnează 8 biți) se aplică pe fiecare dintre cei 4 octeți din [k_i XOR R_(i-1)] în parte.

[1] https://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/1991/CS/CS0708.pdf

Pentru atacul Differential Cryptanalysis avem nevoie de un punct de atac, pentru care știm că o diferență între două mesaje (Dx = m1 XOR m2) duce la o anumită diferență Dy în acel punct, cu o anumită probabilitate p. Ar fi bine ca în acest punct să putem ajunge și dinspre ciphertext, ca o funcție de k3 (cheia din ultima rundă). Pentru asta am ales punctul marcat cu roșu în figură, numind rezultatul din acel punct cu E'(k, m). Putem observa că din c=E(k,m) putem obține E'(k, m) astfel:
E'(k, m) = R3 XOR S-box(k3 XOR L3),
adică o funcție de c = L3 | R3 și k3 (aici "|" este operația de concatenare).

Acum mai avem nevoie de o relație între Dx și Dy în acest punct. Din figură putem observa că pentru oricare două mesaje m1 și m2, dacă între ele avem o diferență Dx doar în prima parte (L0), atunci această diferență va trebui să existe și pentru E'(k, m). Așadar, dacă avem m1 = L0_1 | R0 și m2 = L0_2 | R0, cu Dx = L0_1 XOR L0_2, atunci ar trebui să avem și Dy = E'(k, m1) XOR E'(k, m2) = Dx cu probabilitate p=1.

O ultimă observație importantă este că ICEC nu are o funcție de difuzie, astfel încât un octet din Dx afectează exact același octet în Dy.

Cu aceste informații, putem lansa o criptanaliză diferențială pentru a afla fiecare octet din cheia k3 din ultima rundă în felul următor (repetați pașii aceștia pentru fiecare octet din k3):

1. Obțineți multe perechi de mesaje m1, m2 cu același diferențial Dx (puteți folosi același diferențial pentru a ataca toți octeții din k3)
2. Obțineți criptarea E(k,m1) și E(k,m2) pentru fiecare pereche (m1, m2)
3. Pentru fiecare valoare posibilă k3[i] a octetului i din k3:
    a. Obțineți punctele intermediare E'(k3, m1)[i] și E'(k3, m2)[i] pentru toate perechile (m1,m2), unde E'(k3, m)[i] reprezintă octetul i din E'(k3, m)
    b. Calculați diferențialul D(k3)[i] = E'(k3, m1)[i] și E'(k3, m2)[i] pentru toate perechile (m1, m2)
    c. Verificați pentru câte dintre perechile (m1, m2) diferențialul D(k3)[i] corespunde cu diferențialul Dx[i] al mesajelor (vezi punctul 1)
4. Selectăm valoarea k3[i] ca fiind cea cu care am obținut numărul cel mai mare de diferențiale D(k3)[i] egale cu Dx[i] (știm că pentru cheia corectă probabilitatea ca Dx[i]=D(k3)[i]

Pentru tema voastră aveți deja un schelet de cod (diff_crypto_attack.py), cu care veți putea implementa atacul. Cu funcțiile icc_enc_server și icc_dec_server din cipher_server.py veți putea obține (de la un server) criptarea c=E(k,m) pentru mesaje alese de voi (chosen-plaintext attack). În cipher.cy aveți implementarea algoritmului ICEC.

Vă sunt date un ciphertext țintă (pe care va trebui să-l decriptați) și primii 8 octeți (k1|k2) din cheia cu care a fost criptat. Voi trebuie să aplicați atacul diferential cryptanalysis (vedeți mai sus) pentru a obține k3 și astfel să puteți apoi să apelați funcția de decriptare (icc_dec) cu cheia completă (k=k1|k2|k3) pentru a afla care este mesajul criptat.