Kriptografija in teorija kodiranja – 3. domača naloga (do srede 28. nov., 2003)

---

1. Kompresijo podatkov pogosto uporabljamo za shranjevanje ali prenos podatkov. Kompresijo opravimo tako, da odstranimo določene informacije, brez katerih znamo rekonstruirati original. Predpostavimo, da uporabljamo kompresijo skupaj s šifriranjem. Ali je smiselno narediti naslednje operacije

   (i) kompresirati informacijo in jo potem zašifrirati,

   (ii) zašifrirati informacijo in jo potem kompresirati?

   Utemelji svoj odgovor (podaj vsaj dva razloga)!

2. Naj bo y output SPN algoritma za input x, kjer je $\ell = m = N_r = 4$, permutaciji π_S in π_P pa podani s tabelami:

z	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\pi_S(z)$	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

   ter

z	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\pi_P(z)$	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

   Naj bo ključ $K = (k_1, \ldots, k_{32}) \in \{0,1\}^{32}$ definiran z

$$K = \texttt{0011 1010 1001 0100 1101 0110 0011 1111},$$

   sedaj pa izberimo še razpored ključev tako, da je za $1 \leq r \leq 5$, krožni ključ K^r izbran kot 16 zaporednih bitov ključa K z začetkom pri k_{4r-3}:

$$\begin{aligned} K^1 &= 0011\ 1010\ 1001\ 0100 \\ K^2 &= 1010\ 1001\ 0100\ 1101 \\ K^3 &= 1001\ 0100\ 1101\ 0110 \\ K^4 &= 0100\ 1101\ 0110\ 0011 \\ K^5 &= 1101\ 0110\ 0011\ 1111 \end{aligned}$$

   Torej je

$$y = \mathrm{SPN}(x, \pi_S, \pi_P, (K^1, \ldots, K^{N_r+1})),$$

   kjer je $(K^1, \ldots, K^{N_r+1})$ razpored ključev. Poišči zamenjavo π_{S^*} in permutacijo π_{P^*} tako, da bo

$$x = \mathrm{SPN}(y, \pi_{S^*}, \pi_{P^*}, (K^{N_r+1}, \ldots, K^1)).$$

3. (a) (*Popravljanje napak na DES-ovemu tajnopisu*) Z DES-om zašifriramo s blokov čistopisa $m_1 m_2 \dots m_s$ v tajnopis $c_1 c_2 \dots c_s$. Pri prenosu se i-ti blok poškoduje. Koliko blokov tajnopisa se bo odšifriralo narobe, če uporabimo (i) ECB način oziroma (ii) CBC način?

   (b) (*DES-ova komplementarna lastnost*) Z $\overline{m}$ označimo komplement binarnega zaporedja m. Ni se težko prepričati, da je $\overline{c} = DES_{\overline{k}}(\overline{m})$ za $c = DES_k(m)$. Ali lahko uporabite to lastnost DES-a za izboljšanje časa požrešnega napada (i) pri poznanem čistopisu (ii) pri izbranem čistopisu?

   (c) Opišite napad s poznanim čistopisom na DES-ov CBC način, ki odkrije tajni ključ? Ocenite koliko DES širiranj/dešifriranj potrebuje Vaš napad.

4. Oglejmo si naslednji predlog za zaščito DES-a pred požrešnim napadom (tj. napadom, ki pregleda vse ključe). Tajni ključ je $k = (k_1, k_2)$, kjer je $k_1 \in \{0,1\}^{56}$ in $k_2 \in \{0,1\}^{64}$. Naj bo $m \in \{0,1\}^{64}$ čistopis. Šifriranje se opravi na naslednji način:

$$E_k(m) = DES_{k_1}(m) \oplus k_2.$$

   (a) Pokažite, da se s tem predlogom ne poveča čas, ki je potreben za požrešni napad (z drugimi besedami, poiskati morate napad, ki potrebuje reda velikosti 2^{56} DES šidriranj/dešifriranj). Privzamete lahko, da poznate majhno število parov čistopis/tajnopis $c_i = E_k(m_i)$.

   (b) Odgovorite na isto vprašanje, če opravite šifriranje na naslednji način:

$$E_k(m) = DES_{k_1}(m \oplus k_2).$$

5. Za dan simetričen šifrirni sistem E_k definirajmo naključni simetrični šifrirni sistem F_k:

$$F_k(m) = (E_k(r), r \oplus m),$$

   kjer je r zaporedje bitov enake velikosti kot zaporedje m. Output za $F_k(m)$ je torej enkripcija enkratnega-ščita r, skupaj z originalnim sporočilom m, ki mu prištejemo (XOR) naključno število r. Za vsako enkripcijo si izberemo novo/neodvisno naključno število.

   Oglejmo si dva napada, katerih cilj je odkriti tajni ključ k.

   (a) Pri napadu z izbranim čistopisom si lahko napadalec izbere zaporedja nizov $m_1, m_2, \dots$ in za vsak niz m_i najde ustrezen tajnopis.

   (b) Pri napadu z naključnim čistopisom napadalec dobi naključne pare čistopis/tajnopis. Opomba: napadalec nima kontrole nad naključnimi števili r, ki so uporabljena za generiranje parov čistopis/tajnopis.

   Dokaži, da je šifrirni sistem F_k varen pred napadom z izbranim čistopisom, če je E_k varen pred napadom z naključnim čistopisom.

6. (bonus) Če uporabimo SPN iz 2. naloge, s to razliko, da zamenjamo S-škatlo s funkcijo π_T, ki ni permutacija (bolj natančno to pomeni, da π_T ni surjektivna), potem poišči napad s samo poznanim tajnopisom (za katerega je bil uporabljen isti ključ), ki določi bite kluča iz zadnjega kroga.