1. Egy kommunikációs csatorna 10^{-6} bithiba-aránnyal működik. Hogyan alakul a bithiba-arány rejtjelezett esetben, ha a rejtjelezés

a.) 128 bites kódolás CBC módban? (2p)

b.) 64 bites kódolás OFB byte alapú folyamrejtjelezéssel? (2p)

A véletlen csatornahibák ellen hibajavító kódolást alkalmazunk és a hibajavító kódolást a rejtjelezést megelőzően végezzük. Helyesen járunk-e el a hibák javításával kapcsolatosan, ha

c.) CBC módban rejtjelezünk? (2p)

d.) OFB módban rejtjelezünk? (2p)

Megoldás:

a.) 65 E-6 b.) E-6

c.) Nem. A CBC mód hibaterjedés tulajdonsága szerint egy véletlen hiba esetén, hibázás utáni első blokk bitjeinek átlagosan fele hibás lesz, s még a rákövetkező blokk egy bitje. Ezt a nagymértékű meghibásodást csak igen költséges, komplex javító kóddal tudnánk eliminálni. A helyes megoldás a rejtjelezés utáni hibajavító kódolás alkalmazása.

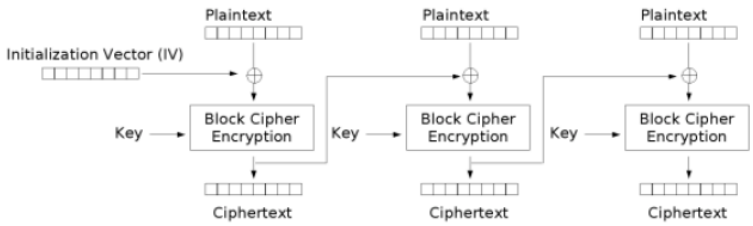
d.) Igen. Nincs hibaterjedés a kulcsfolyamatos típusú rejtjelezés mód miatt. Ez esetben alkalmazhatjuk a hibajavítást a rejtjelezést megelőzően.

Háttérismeret:

ECB (Electronic Codebook)

minden blokkot külön kódol ugyanazzal a kulccsal

CBC (Cipher Block Chaining)

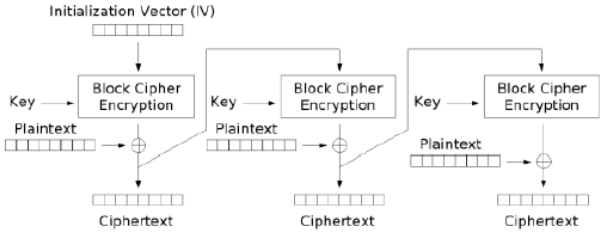


bithiba esetén az adott blokk bitjeinek fele és a következő blokkból egy bit lesz hibás

IV módosításával manipulálható az első blokk

érzékeny a szinkronvesztésre

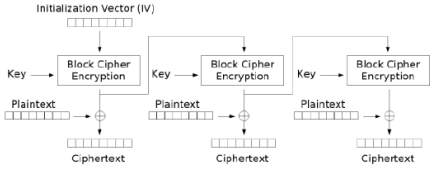
CFB (Cipher Feedback)



bithiba esetén az adott blokk bitjeinek fele és a következő blokkból egy bit lesz hibás

érzékeny a szinkronvesztésre

OFB (Output Feedback) / CTR (Counter Mode Encryption)



bithiba esetén csak az adott bitet rontja el

2. Tekintsük az alábbi H: $\{0,1\}^* \rightarrow \{0,1\}^n$ hash függvényt: $H(x)=h_1(\, x_1 \mid h_2(x_2)\,)$, ahol x_1 és x_2 az x bitsorozat első és második fele, valamint | a bitsorozatok összefűzését jelenti.
a.) Definiálja az ütközésellenálló hash függvényt (1p)
b.) Biztonságos-e a $H(x)$ konstrukció, ha tudjuk, hogy h_1: $\{0,1\}^* \rightarrow \{0,1\}^n$ ütközés-ellenálló, de h_2: $\{0,1\}^* \rightarrow \{0,1\}^n$ nem ütközés-ellenálló tulajdonságú. Indokoljon! (9p)

Megoldás:

a.)

Ütközésmentes (CRHF): nehéz feladat olyan X, X' (X≠X') üzenetpárt megadni, amelyeknek azonos a lenyomata, azaz amelyekre
$H(X)=H(X') \rightarrow D_A(H(X))= D_A(H(X'))$

b.) Nem. Legyen $m_{i,1}$ az m_i üzenet első fele, ugyanígy $m_{i,2}$ a második fele (i=1,2). Keressünk olyan $m_1 \neq m_2$ üzenetpárt, amire $m_{1,1}= m_{2,1}$, $m_{1,2} \neq m_{2,2}$, $h_2(m_{1,2})=h_2(m_{2,2})$. Ekkor $h_1(m_{1,1} \mid h_2(m_{1,2}))= h_1(m_{2,1} \mid h_2(m_{2,2}))$.

Tekintsük az alábbi $H(x,n) = h_1^{(n)}\left(h_2^{(n)}(x)\right)$ hash függvényt, ahol $h_1^{(n)}$ illetve $h_2^{(n)}$ a h_1 illetve h_2 hash függvények *n*-szeres alkalmazását jelenti. Milyen feltételek mellett, *n* mely értékeire lesz biztonságos a *H(x,n)* konstrukció, ha tudjuk, hogy

a) h_1 ütközés-ellenálló, de h_2 nem ütközés-ellenálló tulajdonságú? (8 p)

b) h_1 nem ütközés-ellenálló, de h_2 ütközés-ellenálló tulajdonságú? (8 p)

Válaszát, mindkét esetben Indokolja!

Megoldás:

a) Semmilyen n-re nem lesz biztonságos, mivel a belső hash leképezésre könnyű ütközést előállítani, amit a külső leképezés helyben hagy.

b) Ha h_2 nem őskép ellenálló, akkor H nem biztonságos. Indirekt. h_1 nem CRHF igy könnyen találunk h_1-őskép párt. Ha h_2 nem őskép ellenálló, akkor ezen párhoz könnyű találni h_2-őskép (=H-őskép) párt.

Háttérismeret:

Ütközésmentes (CRHF): nehéz feladat olyan X, X' (X≠X') üzenetpárt megadni, amelyeknek azonos a lenyomata, azaz amelyekre
$H(X)=H(X') \rightarrow D_A(H(X))= D_A(H(X'))$

1. *Mutassa meg*, hogy a CBC-MAC kulcsolt hashing nem teljesíti az egyirányúság követelményét egy a k kulcsot ismerő fél számára, azaz tetszőleges MAC ellenőrzőösszeghez képes csaló üzenetet előállítani. **(12 p)**

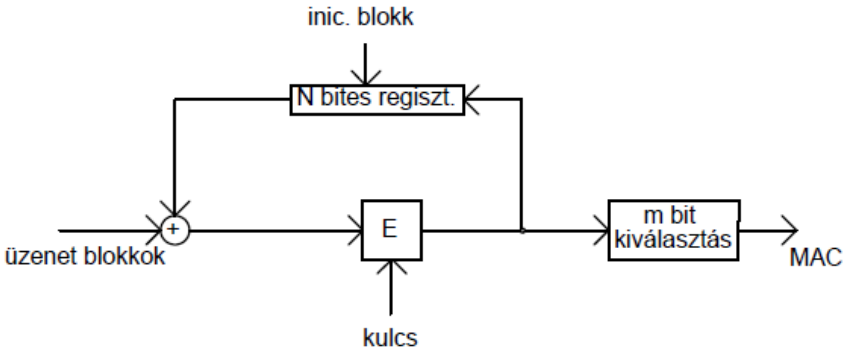
Megoldás:

Legyen μ tetszőlegesen választott MAC. Előállítható olyan üzenet, amelynek lenyomata μ. Tekintsünk egy $m = M_1|M_2|\ldots|M_r$ üzenetet, azaz M_i üzenetblokkok r hosszúságú sorozatát, amelyre $MAC_k(m)$ CBC-MAC blokkot kaptuk. Ha az $(r+1)$-edik üzenetblokkot $M_{r+1} = D_k(\mu) \oplus MAC_k(m)$ szerint választjuk, ellenőrizhető, hogy $m' = M_1|M_2|\ldots|M_{r+1}$ üzenetre az előírt μ MAC értéket kapjuk. Ugyanakkor vegyük azt is észre, hogy bár a konstruált sorozat első r blokkját tetszés szerinti csalárd tartalmúra beállíthatjuk, az $(r+1)$-edik blokk tartalma már véletlenszerűen alakul.

Háttérismeret:

CBC-MAC (Message Authentication Code)

$[x_1, \ldots, x_n, MAC(x_1,\ldots,x_n)]$



$y_i = E_k(x_i \oplus y_{i-1})$, y_0 = inic., $i=1,\ldots,n$

$MAC = g(y_n)$

2. *Igazoljunk hash függvény tulajdonságokat*:
a.) az egyszerű modulo-32 ellenőrzőösszeg (32 bites szavak 32 bites összege) nem egyirányú,
b.) az $h(x)=x^2-1 \bmod p$ (p prím) leképezés nem OWHF,
c.) az $h(x)=x^2 \bmod pq$ (p és q "nagy", titkos prímek) leképezés OWHF, de nem CRHF.
d.) az $h(x,k)= E_k(x)$ (E a DES rejtjelezés) leképezés (x,k) párban nem OWHF
(12p=4*3p)

Megoldás:

b.) modulo p szerinti gyökvonás ismert, "könnyű" feladat
c.) $(+x)^2= (-x)^2 \pmod p$

Háttérismeret:

Egyirányú (OWHF): egy y hash érték (lenyomat) ismeretében nehéz feladat olyan X' üzenetet (ősképet) kiszámítani, amelyre h=H(X')

Ütközésmentes (CRHF): nehéz feladat olyan X, X' ($X \neq X'$) üzenetpárt megadni, amelyeknek azonos a lenyomata, azaz amelyekre $H(X)=H(X') \rightarrow D_A(H(X))= D_A(H(X'))$

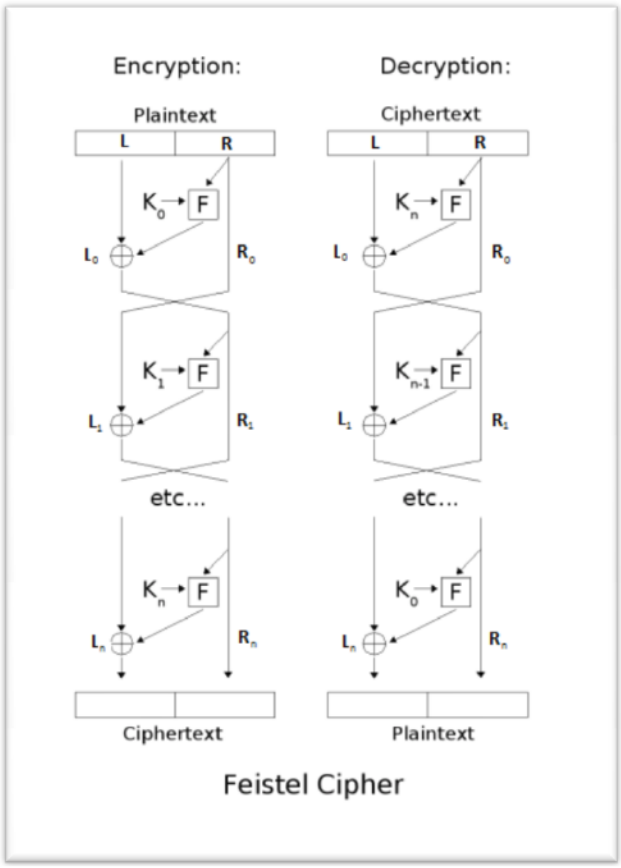
19. DES Feistel technika

A nyílt szövegblokkot két egyforma részre vágjuk: L, R.
Sorozatban végrehajtjuk rajta az F transzformációt.

Az (i+1). iteráció során:
az aktuális kulcsot a kulcsütemező szolgáltatja: K_i
$L_{i+1} = R_i$
$R_{i+1} = L_i \oplus F(R_i, K_i)$

Ez invertálható, tetszőleges F-re, mert:
$L_i = R_{i+1} \oplus F(L_{i+1}, K_i)$
$R_i = L_{i+1}$

A dekódolás egyszerű: a rejtett szöveget be kell adni a rejtjelező bemenetére, csak az iterációk kulcsainak sorrendjét kell megfordítani.



2. CRHF tulajdonságúak-e az alábbi kriptográfiai hash függvény jelöltek:

a.) $f(x)=x^e \bmod pq$ RSA kódoló leképezés?

b.) $f(x, y, z)=(x \cdot y) OR\ z \cdot (x+y))$, $|x|=|y|=|z|$, $f: \{0,1\}^{3n} \rightarrow \{0,1\}^n$ *(x op y)* bitenként értelmezett AND és XOR) iterációs (kompressziós) függvényre épülő hash függvény, amely DM megerősítést is alkalmaz?

Megoldás:

a.) Nem CRHF. x és x' "lenyomata" azonos, ha x'=x mod pq

b.) Nem CRHF. Mivel f(x,y,z)=f(y,x,z), ezért nem CRHF a kompressziós függvény.

{a.) feladat esetén – mivel hash függvényről van szó – az x argumentum tetszőleges nemnegatív egész. Ezzel rokon feladat ($f(x)=x^2 \bmod pq$) került előadáson elmondásra.
b.) feladat és megoldása az előadás slide-on is szerepel}

Háttérismeret:

Ütközésmentes (CRHF): nehéz feladat olyan X, X' ($X \neq X'$) üzenetpárt megadni, amelyeknek azonos a lenyomata, azaz amelyekre $H(X)=H(X') \rightarrow D_A(H(X))= D_A(H(X'))$

2. Tekintsük az alábbi integritásvédelmi kódolást, ahol rejtjelezés és MAC kombinációját használjuk: $E_k(m \mid MAC_{k'}(m))$
Jelölések: $E_k(.)$ CBC módú rejtjelezés, az m üzenet N blokkból áll, $MAC_{k'}(.)$ egy CBC-MAC és egyben az N+1-dik rejtjelezendő blokk, továbbá a rejtjelezésre (IV; k), míg a MAC számításra (IV'; k') (inicializáló vektor, kulcs) pár kerül alkalmazásra.

a.) Elfogadható-e, ha $IV = IV'$; $k = k'$ egyszerűsítő választással élünk? Indokoljon! (10p)

b.) Milyen tanulságot tud levonni ebből? (2p)

Megoldás:

a.) Nem. Tekintsünk pl. egy blokk hosszú m üzenetet. A CBC mód lépéseit végiggondolva látható, hogy a kódolás eredménye:

$E_k(m+IV) \parallel E_k(0)$

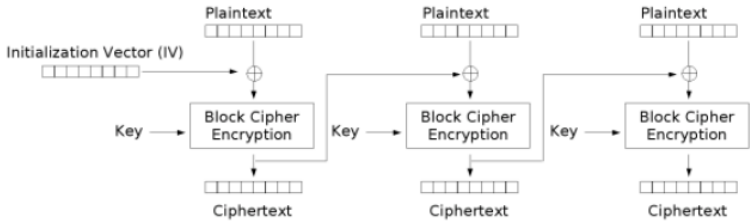
az üzenettől függetlenül, azaz semmiféle integritásvédelmet nem kapunk (itt E_k az ECB üzemű kódolást jelöli)

b.) A tanulság az, hogy ennél a megoldásnál, eltérő kulcsot kell alkalmazni a kétféle funkcióra Az IV megválasztásának szabadságát meghagyjuk az alkalmazásnak.

Háttérismeret:

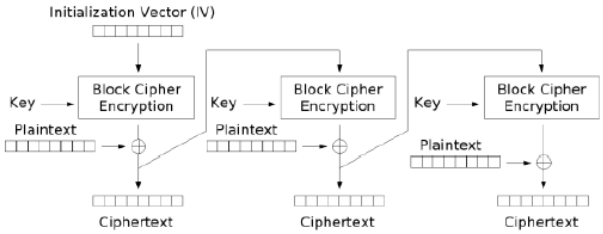
ECB (Electronic Codebook)
minden blokkot külön kódol ugyanazzal a kulccsal

CBC (Cipher Block Chaining)



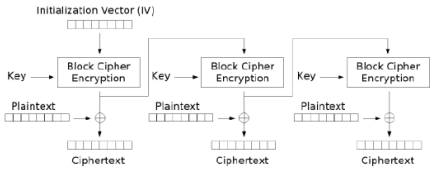
bithiba esetén az adott blokk bitjeinek fele és a következő blokkból egy bit lesz hibás
IV módosításával manipulálható az első blokk
érzékeny a szinkronvesztésre

CFB (Cipher Feedback)



bithiba esetén az adott blokk bitjeinek fele és a következő blokkból egy bit lesz hibás
érzékeny a szinkronvesztésre

OFB (Output Feedback) / CTR (Counter Mode Encryption)



bithiba esetén csak az adott bitet rontja el

3. Ha mind a titkosság, mind pedig az adatintegritás biztosítása szükséges, egy lehetséges módszer a rejtjelezés és a lenyomatkészítés együttes alkalmazása, az

$E_k(m \mid H(m))$

kódolás, azaz az üzenetet lenyomattal meghosszabbítjuk, majd rejtjelezünk.

a.) Képezzük a lenyomatot olyan módon, hogy az üzenetblokkokat XOR összeadjuk, továbbá a rejtjelezés legyen CBC módú. Helyes-e a védelemnek ez a módja?

b.) Segít-e az előző problémán az, ha a lenyomatképzést a jól ismert lineáris CRC-vel (Cyclic Redundancy Code) végezzük?

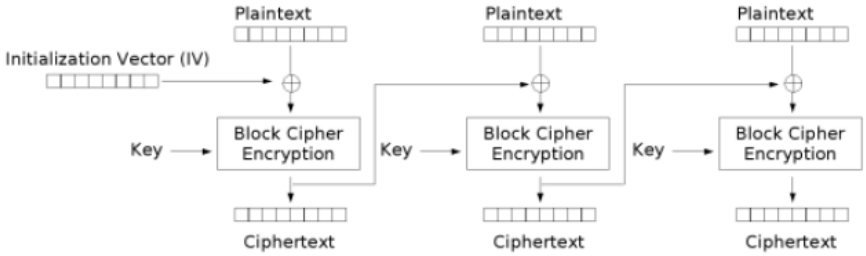
Megoldás:

Elegendő, ha ellenőrizhető és elegendő bitméretű redundanciát csatolunk az üzenethez. A CBC rejtjelezett üzenet ugyanis - nagy valószínűséggel - nem módosítható úgy, hogy ne okozzon detektálható változást a dekódolt üzenetben. Tehát
a. Biztonságos.
b. Biztonságos.

{Ha egy integritásvédő kriptográfiai ellenőrzőösszeg lett volna a feladat $[m,E_k(H(m))]$ alakú konstrukcióval, akkor egyik megoldás sem lenne biztonságos. Ezen utóbbi, mint megoldott feladat szerepelt előadáson, míg a fenti 3. feladat technikája az integritásvédő protokollok között a rejtjelezéssel történő védelem alpontban került elmondásra.}

Háttérismeret:

CBC (Cipher Block Chaining)



bithiba esetén az adott blokk bitjeinek fele és a következő blokkból egy bit lesz hibás
IV módosításával manipulálható az első blokk
érzékeny a szinkronvesztésre

4. Egy n karakterből álló $M = (m_1, m_2, \ldots, m_n)$ üzenetet OFB módban kódolunk és az eredményül kapott $C = (c_1, c_2, \ldots, c_n)$ rejtjeles üzenetet tároljuk. Később kiderül, hogy az i. karaktert nem kell tárolnunk, ezért C-t dekódoljuk, az i. karaktert töröljük, és az így kapott $M' = (m_1, \ldots, m_{i-1}, m_{i+1}, \ldots, m_n)$ üzenetet újra kódoljuk. Legyen a második kódolás eredménye $C' = (c'_1, c'_2, \ldots, c'_{n-1})$. A második kódolás során ugyanazt a kezdeti értéket (IV) és kulcsot használjuk, mint az első kódolásnál. (Vegyük észre, hogy ekkor minden $0 < t < i$ esetén $c'_t = c_t$, ezért lényegében az első i-1 karaktert nem is kell újra kódolnunk.) Tegyük fel, hogy egy támadó hozzáfér C-hez és C'-höz, és megtudja a törölt m_i karakter értékét is. Fenyegeti-e veszély az M' üzenet titkosságát? Válaszát indokolja! (10 p)

Megoldás:

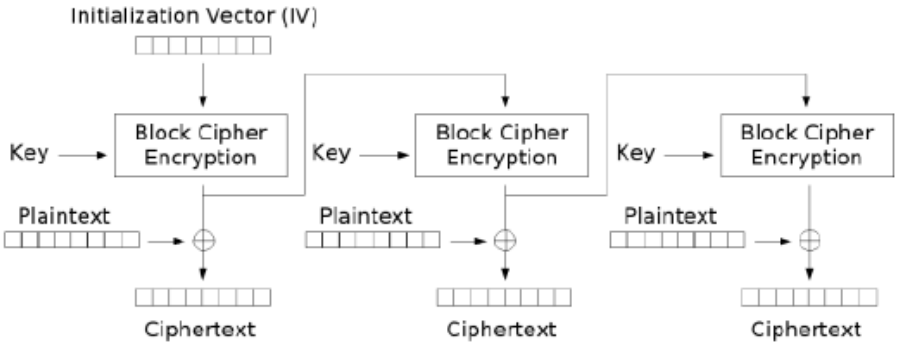
Legyen az OFB által generált kulcskarakterek sorozata $k_1, k_2, \ldots$. Ekkor $c_t = m_t + k_t$ minden $t = 1, 2, \ldots, n$ esetén, valamint $c'_i = m_{i+1} + k_i$, $c'_{i+1} = m_{i+2} + k_{i+1}$, stb.
A támadó a következőket tudja kiszámolni:

$k_i = c_i + m_i$
$m_{i+1} = c'_i + k_i$
$k_{i+1} = c_{i+1} + m_{i+1}$
$m_{i+2} = c'_{i+1} + k_{i+1}$
…

A támadó tehát dekódolni tudja az $m_{i+1}, m_{i+2}, \ldots, m_n$ karaktereket, azaz komoly veszély fenyegeti az M' titkosságát ha $i < n$.

Háttérismeret:

OFB (Output Feedback) / CTR (Counter Mode Encryption)



3. Egy kliens és egy szerver közötti biztonsági protokoll a következő részekből áll:

– kölcsönös partnerhitelesítés a kapcsolat elején (C – kliens, S – szerver):

C → S: N_C

S → C: N_S, $MAC_K(N_C)$

C → S: $MAC_K(N_S)$

ahol N_X egy 128 bites véletlen szám (nonce), K egy korábban létrehozott hosszú élettartamú közös AES kulcs C és S között, $MAC_K()$ pedig a CBC-MAC függvény az AES blokkrejtjelezővel, K kulccsal és IV = 0 beállítással;

– kapcsolat során küldött üzenetek rejtjelezése az AES blokkrejtjelezővel CTR módban a fent említett K kulccsal;

– rejtjelezett üzenetek integritásvédelme CBC-MAC módszerrel az AES blokkrejtjelezővel, K kulccsal, és IV = 0 beállítással.

Mutasson támadást, melyben a támadó tetszőleges, a fenti módszerrel rejtjelezett üzenetet dekódolni tud! (12 p)

Megoldás:

Bármely fél rejtjelező-orákulumként használható a partnerhitelesítés protokollt felhasználva:

A → S: X

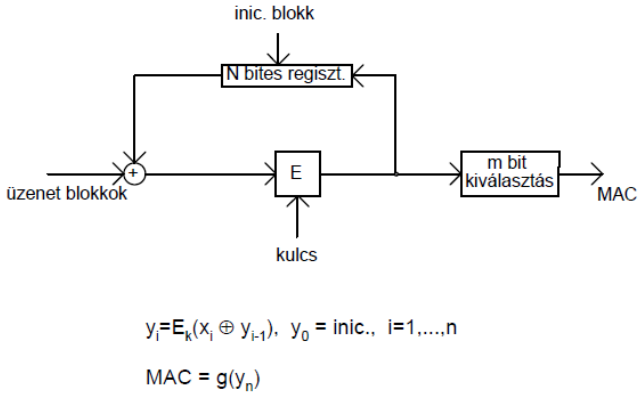
S → A: Y, $MAC_K(X) = AES_K(X+0) = AES_K(X)$

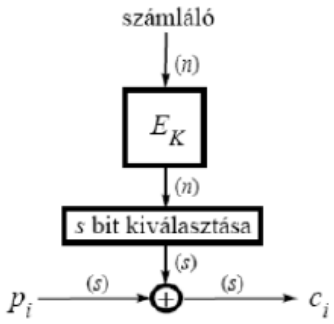
A rejtjelezett üzenet i. blokkja így áll elő: M_i XOR $AES_K(C_i)$ ahol M_i az i. nyílt blokk, C_i az i. számláló érték. Tehát tetszőleges rejtjeles blokk dekódolható úgy hogy megszerezzük $AES_K(C_i)$ – t az orákulumtól.

Háttérismeret:

CBC-MAC (Message Authentication Code)

$[x_1, ..., x_n, MAC(x_1,...,x_n)]$





CTR mód, blokkséma

Tegyük fel, hogy A és B kölcsönösen azonosítani kívánják egymást, amelyet arra kívánnak alapozni, hogy ismerik egymás jelszavát (P_A,P_B).

a.) Biztonságos-e az alábbi protokoll? (15 pont)

1. A→B: ID_A , k^p_A
2. B→A: ID_B , k^p_B
3. A→B: $z1=E_B(P_A)$
4. B→A: $z2=E_A(P_B)$
5. A: $D_A(z2)=P_B$?
   B: $D_B(z1)=P_A$?

b.) Biztonságos-e a protokoll, ha A és B ismerik egymás nyilvános kulcsát, s nem kerül sor az 1-2. lépésekre? (10 pont)

(ID: azonosító, k^p:nyilvános kulcs, E,D: nyilvános kulcsú kódoló, dekódoló transzformáció)

Megoldás: (szerintem)

a) Csak passzív támadás ellen véd. Nem biztosítja a kulcskonfirmációt, kulcsfrissességet és a partnerhitelesítést.
b) Kulcsfrissességet még mindig nem biztosít.

Háttérismeret:

Implicit kulcshitelesítés
a protokoll sikeres lefutása után A biztonsággal feltételezheti, hogy rajta kívül csak a feltételezett másik fél, mondjuk B, és esetleg egy megbízható harmadik fél (pl.
a kulcsszerver) férhet hozzá a protokoll során létrehozott kapcsolatkulcshoz
Kulcskonfirmáció
a protokoll sikeres futása után A meggyőződhet arról, hogy a másik résztvevő (B), valóban birtokában van a protokoll futása során létrehozott kapcsolatkulcsnak
Explicit kulcshitelesítés
a protokoll egyszerre biztosítja az implicit kulcshitelesítést és a kulcskonfirmációt
Kulcsfrissesség
a protokoll sikeres futása után A meg van győződve arról, hogy a létrehozott kapcsolatkulcs új, és nem egy korábban már használt (esetleg megfejtett) kulcs
Partnerhitelesítés
a protokoll sikeres futtatása után A meg van arról győződve, hogy a feltételezett másik résztvevő valóban részt vett a protokoll végrehajtásában.

2. a.) CBC-MAC definiciója (folyamatábra, formula, a jelölések magyarázatával) **(3p)**

b.) Definiáljuk egy üzenethitelesítő (MAC) konstrukciót az alábbi módon. Adott egy CBC-MAC eljárás, amely n bithosszú blokkokra osztja a bemenetét. Mielőtt egy m üzenet CBC-MAC értékét kiszámolnánk a hagyományos módon, kiegészítjük az üzenetet vagy egy 1^n blokkal (n hosszú csupa 1), ha m páratlan számú blokkra osztható fel, vagy egy 0^n blokkal (n hosszú csupa 0), ha m páros számú blokkra osztható fel. Az ilyen módon kiegészített adat CBC-MAC értéke lesz az m üzenet hitelesítő kódja.

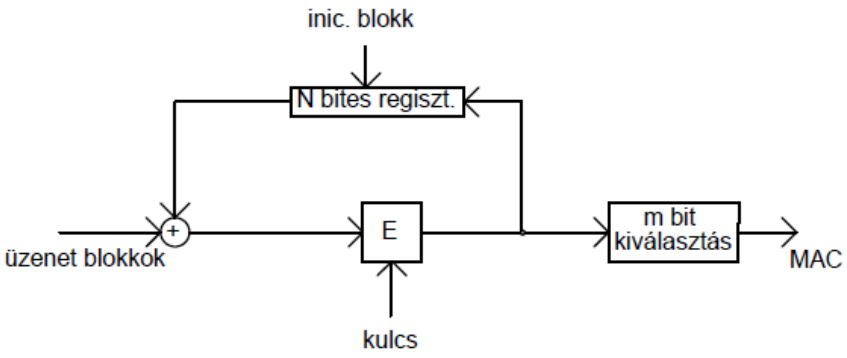
Biztonságos-e a konstrukció? Miért? **(7p)**

Megoldás:

2. a.

CBC-MAC (Message Authentication Code)

$[x_1, ..., x_n, MAC(x_1,...,x_n)]$



$y_i = E_k(x_i \oplus y_{i-1})$, y_0 = inic., $i=1,...,n$

$MAC = g(y_n)$

2. b. Nem biztonságos.
MAC orákulum kérések: m_1 üzenet MAC-je $MAC_k(m_1)$, m_2 üzenet MAC-je $MAC_k(m_2)$, ahol m_1 és m_2 egy blokk méretű.
Támadás:Legyen $m_3=m_1 | 1^n | MAC_k(m_1) + m_2$, ahol + mod 2 bitekénti összeadás.
$MAC_k(m_3)= MAC_k(m_2)$.

3. Az AWP (Another Weak Protocol) célja két távoli fél között átküldött üzenetek titkosítása, integritásvédelme, és a visszajátszás elleni védelem. Feltesszük, hogy a felek között már van egy megosztott 128 bites K szimmetrikus kapcsolatkulcs. Egy M üzenet integritásvédő ellenőrző-összegét úgy számoljuk ki, hogy M-et 128 bites M_i (i = 1, 2, …) blokkokra osztjuk, ha az utolsó blokk, M_{last} rövidebb 128 bitnél, akkor 0 bitekkel 128 bitre egészítjük ki, majd az így kapott blokkok (bitenkénti) XOR összegét számolva kapjuk az ellenőrző-összeget: ICV = $M_1 + M_2 + ... + M_{last}$ (ahol + jelöli az XOR-t). Ezután az ICV-t az eredeti üzenet végére csatoljuk, és az (M | ICV) bitsorozatot rejtjelezzük AES rejtjelezővel CTR módban. A rejtjelezésnél használt számláló kezdeti értéke C, és a számlálót minden blokk rejtjelezése után eggyel növeljük. A kezdő C értéket a csomag fejlécében küldjük át a vevőnek, így az átküldött csomag formátuma a következő: C | AES-CTR$_{K,C}$(M | ICV), ahol AES-CTR$_{K,C}$() jelöli az AES-sel történő CTR módú rejtjelezést K kulccsal és C kezdeti számláló értékkel. A kapcsolat kezdetén (az első üzenet küldésekor) C-t 0-ról indítjuk, majd minden üzenet küldésekor eggyel növeljük. Így C üzenetsorszámként is funkcionál, s ez biztosítja a visszajátszás elleni védelmet. Új kapcsolat esetén új K kapcsolatkulcsot használunk, s C-t ismét 0-ról indítjuk.

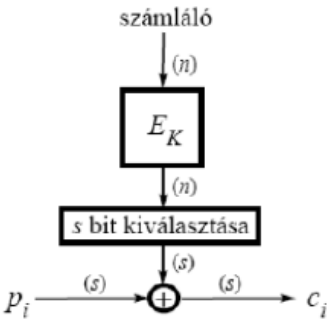
Konstruáljon támadást az AWP protokoll integritásvédelmi mechanizmusa ellen **(4p)** *és titkosítási eljárása ellen* **(6p)**!

Megoldás:

3. Páros számú blokk azonos módosítását nem detektálja az integritásvédelmi mechanizmus ($M_1+X+M_2+X+M_3+…+M_{last} = M_1+M_2+M_3+…+M_{last}$).
Továbbá a blokkokon könnyű célzott módosítást végezni a kulcsfolyam-rejtjelezés miatt (M+Q+M' = (M+M')+Q, ahol Q az AES-CTR által generált kulcsfolyam).

Két egymást követő üzenet kódolásához használt két kulcsfolyam nagy mértékben átfed (azonos). Pl. a C. üzenethez használt kulcsfolyam: $AES_K(C)$, $AES_K(C+1)$, $AES_K(C+2)$, ... és a C+1. üzenethez használt kulcsfolyam: $AES_K(C+1)$, $AES_K(C+2)$, ... Ezért az első kódolt üzenet i+1. és a második kódolt üzenet i. blokkját XOR-olva, a nyílt üzenetek megfelelő blokkjainak XOR összegét kapjuk: $Y_{i+1} + Y'_i = (M_{i+1}+AES_K(C+i)) + (M'_i+AES_K(C+i)) = M_{i+1}+ M'_i$. Ha ismerjük az egyik üzenetet, akkor ebből ki tudjuk számítani a másikat.

Háttérismeret:



CTR mód, blokkséma

2. Tekintsük egy blokk méretű m üzenet RSA aláírását, azaz $Sig(m) = m^d \pmod n$ transzformációt, ahol d a titkos kitevő es n a modulus. Megengedjük, hogy egy Z támadó kéréseket küldhessen az aláíró orákulumnak, majd generálnia kell egy aláírást egy új üzenetre.

Konstruáljon támadást. (Segítség: $(ab)^i \equiv a^i b^i$) (10 pont)

Megoldás:

2. Z kettő kérést küld az aláíró orákulumnak: m_1=2, m_2=2m, ahol m egy új üzenet.

$Sig(2m) = Sig(2) \cdot Sig(m) \pmod n$

egyenlőség alapján

$Sig(m) = Sig(2)^{-1} \cdot Sig(2m) \pmod n$

(Sig(2) invertálható (mod n), mivel 2 invertálható (mod n), ugyanis l.n.k.o(2,n)=1)

Diffie-Hellman kulcsmegegyezés protokoll

a.) Definiálja a protokollt! (7 pont)

b.) Nyújtja-e a protokoll az alábbi szolgáltatásokat (egy-egy mondatban definiálja is az alábbi fogalmakat)? (6 pont)

- kulcskonfirmáció (igen/nem):
- explicit kulcshitelesítés (igen/nem):
- kulcsfrissesség (igen/nem):
- partnerhitelesítés (igen/nem):

c.) Támadható-e a protokoll? Ha igen, hogyan, ha nem, miért nem? (6 pont)

Megoldás:

Diffie-Hellman protokoll

Kulcsmegegyezés protokoll (mindkét kommunikáló fél részt vesz a kapcsolatkulcs generálásában). Feltesszük, hogy Alice és Bob birtokában vannak egy nagy p prímszám, és az általa generált véges Z_p multiplikatív csoport egy g generátor eleme. A protokoll a diszkrét logaritmus problémát használja ki ($g^x(mod\ p) = M$ esetén, ha g és M adott, akkor x kiszámítása nehéz feladat). A kulcshitelességet nem biztosítja, ezért csak passzív támadás ellen véd.

1. Alice -> Bob: $g^x(mod\ p)$	Alice generál egy $1 < x < p-1$ véletlenszámot, majd kiszámítja $g^x(mod\ p)$ értéket és elküldi Bobnak.
2. Bob -> Alice: $g^y(mod\ p)$	Bob generál egy $1 < y < p-1$ véletlenszámot, majd kiszámítja $g^y(mod\ p)$ értéket és elküldi Alicenek.
3.a) Alice: $k = (g^y)^x(mod\ p) = g^{x \cdot y}(mod\ p)$ 3.b) Bob: $k = (g^x)^y(mod\ p) = g^{x \cdot y}(mod\ p)$	Mindkét fél a kapott számot a saját véletlenszámának hatványára emeli, a kapott eredmény adja a közös kulcsot (ami a hatványozás kommutativitása miatt valóban mindkét oldalon ugyanaz).

Implicit kulcshitelesítés
a protokoll sikeres lefutása után A biztonsággal feltételezheti, hogy rajta kívül csak a feltételezett másik fél, mondjuk B, és esetleg egy megbízható harmadik fél (pl.
a kulcsszerver) férhet hozzá a protokoll során létrehozott kapcsolatkulcshoz
Kulcskonfirmáció
a protokoll sikeres futása után A meggyőződhet arról, hogy a másik résztvevő (B), valóban birtokában van a protokoll futása során létrehozott kapcsolatkulcsnak
Explicit kulcshitelesítés
a protokoll egyszerre biztosítja az implicit kulcshitelesítést és a kulcskonfirmációt
Kulcsfrissesség
a protokoll sikeres futása után A meg van győződve arról, hogy a létrehozott kapcsolatkulcs új, és nem egy korábban már használt (esetleg megfejtett) kulcs
Partnerhitelesítés
a protokoll sikeres futtatása után A meg van arról győződve, hogy a feltételezett másik résztvevő valóban részt vett a protokoll végrehajtásában.

c) Közbeékelődéses támadással támadható, mivel nem biztosított a kulcskonfirmáció.

3. RSA rejtjelezés:
a.) Fermat-prímteszt: Melyek a Carmichael számok? Mit tudunk ezen prímteszt szűrési képességéről?
b.) Mi történik, ha RSA dekódolás kapcsán, ha Fermat-álprímet használunk az RSA rejtjelezőnk generálásakor? Mi a helyzet, Carmichael szám esetén? (Formalizált levezetéssel.) **(15p)**

Megoldás:

a)

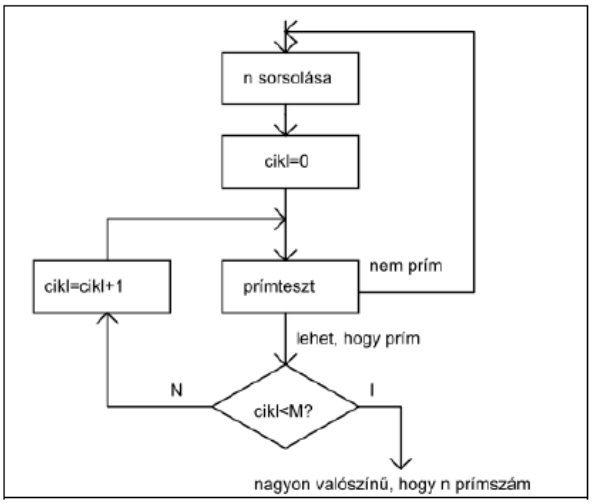
33. Álprím

Egy n összetett szám (Fermat-) álprím egy b bázisra nézve, ha

$$b^{n-1} = 1(mod\ n), ahol\ 1 < b < n\ és\ (b,n) = 1$$

Carmichael-szám: olyan összetett szám, amely tetszőleges b bázisra álprím (pl: 561).

34. Fermat primteszt



- n sorsolása: első és utolsó bitet 1-re állítjuk, a többit egyenként pénzfeldobás alapon választjuk meg
- N alkalommal végrehajtjuk rajta a prímtesztet: kiválasztjuk n egy tetszőleges b bázisát, ha $b^{n-1} \neq 1(mod\ n)$, akkor biztos, hogy n összetett szám
- annak az esélye, hogy egy (nem Carmichael) szám átmegy egy prímteszten, legfeljebb $^1/_2$, így N alkalommal különböző bázisokra végrehajtva a tesztet, 2^{-N}az esélye annak, hogy N összetett szám

b)

Legyen p Carmichael-szám. Mivel

$$x^{1+k\phi(N)} = x\left(x^{p-1}\right)^{k(q-1)} = x \cdot 1 = x\ (\mathrm{mod}\ p),$$

ezért a dekódolás tetszőleges x nyílt szöveg blokkra helyesen megtörténik. Ha viszont p összetett szám nem Carmichael-szám, akkor a kis Fermat-tétel alapú tesztnél látottak szerint $x^{p-1} = 1\ (\mathrm{mod}\ p)$ legalább a nyílt szöveg blokkok felére nem áll fenn, azaz nem működik a dekódolás.

4. Formalizálja, majd vezesse le a CBC mód biztonságával kapcsolatos azon hiányosságot, miszerint ha egy üzenet rejtjelezésekor két rejtjeles blokk megegyezik, akkor következtetni tudunk kapcsolódó nyílt blokkok mod 2 összegére. **(8p)**

Megoldás:

Jelölje ismét $P_1, P_2, \ldots, P_N$ a nyílt blokkokat, és $C_1, C_2, \ldots, C_N$ a rejtjeles blokkokat. A kódolást tehát a következőképpen írhatjuk le formálisan:

$$C_1 = E_K(P_1 \oplus IV) \tag{5.3}$$

$$C_i = E_K(P_i \oplus C_{i-1}) \quad i = 2,3,\ldots,N, \tag{5.4}$$

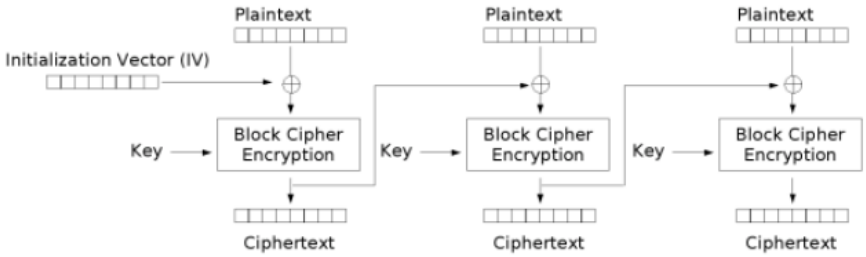
A CBC mód egy ismert gyengesége a következő: Ha két rejtjeles blokk, C_i és C'_j megegyezik, akkor (5.4) használatával kapjuk, hogy

$$C_{i-1} \oplus C'_{j-1} = P_i \oplus P'_j. \tag{5.8}$$

Mivel a támadó C_{i-1}-et és C'_{j-1}-et ismeri, ezért (5.8) alapján két nyílt blokk XOR összegére vonatkozóan jut információhoz. Innen a nyílt üzenetek redundanciáját felhasználva, a támadó sikerrel kísérelheti meg P_i és P'_j megfejtését. A gyakorlatban ennek a támadásnak az szab korlátot, hogy a rejtjeles blokkok egyenletes eloszlásúak, ezért annak a valószínűsége, hogy kettő megegyezik, a születésnapi paradoxon alapján becsülhető. A támadás akkor hatékony, ha a megfigyelt rejtjeles blokkok száma $2^{\frac{n}{2}}$ nagyságrendjébe esik. A tipikus $n = 64$ esetben ez több gigabájt adat megfigyelését igényli.

Háttérismeret:

CBC (Cipher Block Chaining)



bithiba esetén az adott blokk bitjeinek fele és a következő blokkból egy bit lesz hibás
IV módosításával manipulálható az első blokk
érzékeny a szinkronvesztésre

Tegyük fel, hogy DES rejtjelezést használtunk. A 64 bites üzenetblokkon belül, a blokk végén 4 bites hibadetekciós ellenőrzőösszeget alkalmazunk. Kimerítő kulcskereséses támadást végzünk. Elegendő lenne-e 12 rejtjeles blokk megfigyelése a gyakorlatilag egyértelmű kulcsazonosításhoz? (20 pont)

Megoldás:

1. Nem.
Annak a valószínűsége, hogy egy téves kulccsal helyes paritásúra dekódolunk egy rejtett szöveg blokkot, 2^{-4}. Annak a valószínűsége, hogy 12 rejtjeles blokk mindegyikét helyes paritásúra dekódoljuk téves kulcs mellett 2^{-48}, tehát a nem kiszűrt téves kulcsok száma várható értéke a kulcstér teljes végigkeresése után $2^{56}2^{-48}=2^{8}$ $=256$ lenne.

Megj.: Ha a várható érték 1 körül lenne, akkor a válasz IGEN lenne!

Megj. 2: A DES kulcsmérete 56 bit.

Háttérismeret:

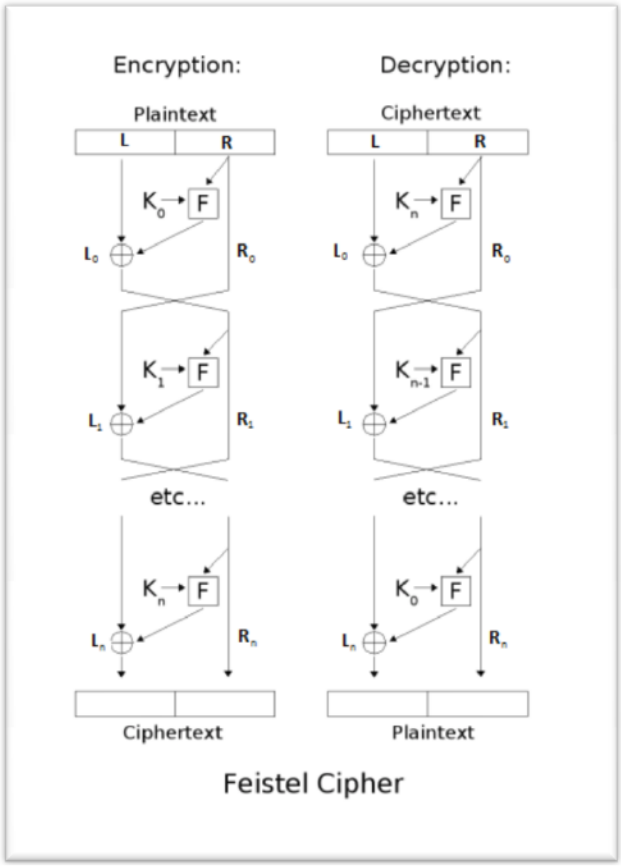
19. DES Feistel technika

A nyílt szövegblokkot két egyforma részre vágjuk: L, R.
Sorozatban végrehajtjuk rajta az F transzformációt.

Az (i+1). iteráció során:
az aktuális kulcsot a kulcsütemező szolgáltatja: K_i
$L_{i+1} = R_i$
$R_{i+1} = L_i \oplus F(R_i, K_i)$

Ez invertálható, tetszőleges F-re, mert:
$L_i = R_{i+1} \oplus F(L_{i+1}, K_i)$
$R_i = L_{i+1}$

A dekódolás egyszerű: a rejtett szöveget be kell adni a rejtjelező bemenetére, csak az iterációk kulcsainak sorrendjét kell megfordítani.



1. Tegyük fel, hogy DES rejtjelezést használtunk (DES kulcsméret 56 bit) 64 bites véletlen üzenet blokkok rejtjelezésére, ahol az üzenet blokkok utolsó bitje paritásbit. Rejtett szövegeket figyelhet meg a támadó, majd kimerítő kulcskeresést végez. 50 rejtjeles blokk megfigyelése elegendő-e a feladat sikeres végrehajtásához, azaz várhatóan sikerül-e kiválasztani a keresett kulcsot? (A DES kódolást és dekódolást véletlen függvényként modellezheti.) **(7p)**

Megoldás:

1. Nem.
P(egy téves kulccsal helyes paritásúra dekódolunk egy rejtett szöveg blokkot) = 2^{-1}
P(50 rejtjeles blokk mindegyikét helyes paritásúra dekódoljuk egy téves kulcs mellett) = 2^{-50}
A szűrésen átment kulcsok átlagos száma a kulcstér teljes végigkeresése után $2^{56} \cdot 2^{-50} = 64 >> 1$

Megj.: Ha a várható érték 1 körül lenne, akkor a válasz IGEN lenne!

Háttérismeret:

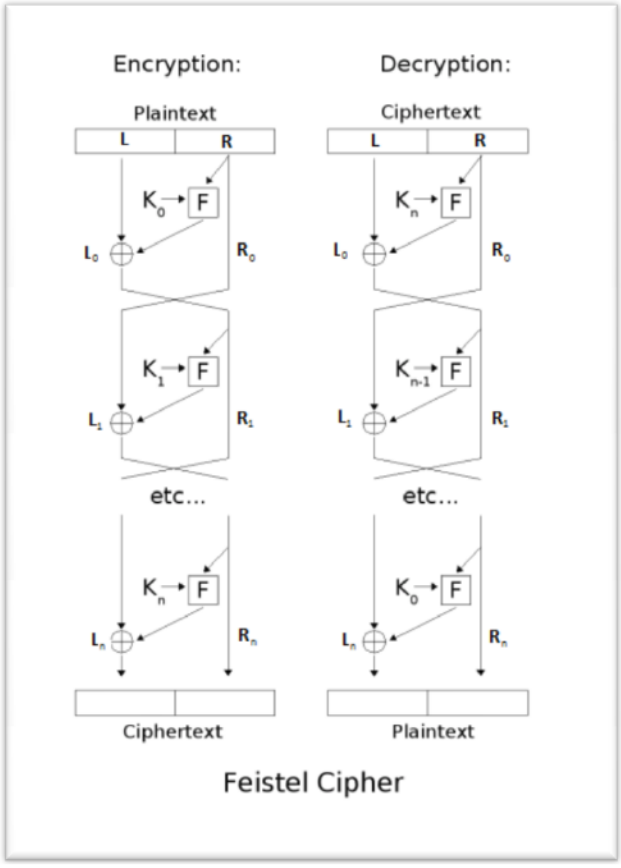
19. DES Feistel technika

A nyílt szövegblokkot két egyforma részre vágjuk: L, R.
Sorozatban végrehajtjuk rajta az F transzformációt.

Az (i+1). iteráció során:
az aktuális kulcsot a kulcsütemező szolgáltatja: K_i
$L_{i+1} = R_i$
$R_{i+1} = L_i \oplus F(R_i, K_i)$

Ez invertálható, tetszőleges F-re, mert:
$L_i = R_{i+1} \oplus F(L_{i+1}, K_i)$
$R_i = L_{i+1}$

A dekódolás egyszerű: a rejtett szöveget be kell adni a rejtjelező bemenetére, csak az iterációk kulcsainak sorrendjét kell megfordítani.



2. Bináris üzenetfolyam bitjeit kulcsfolyamatos rejtjelezéssel rejtjelezzük: Van egy kulcsfolyam és egy üzenetfolyam; az egyes üzenetbitekhez a kulcsfolyam egyes bitjeit adjuk modulo 2: y=x+k mod 2. A üzenetfolyamon illetve a kulcsfolyamon belül a bitek függetlenek, és a két folyam független. Az üzenetbitek eloszlása P(x=0)=1/2, P(x=1)=1/2, a kulcsbitek eloszlása P(k=0)=3/4, P(k=1)=1/4.
Péter a következőt állítja: mivel az üzenetbitek pénzfeldobás-sorozatot alkotnak, továbbá a mod 2 összeadásos rejtjelezés kommutatív, a rejtjelezés tökéletes (shannon-i értelemben).
a.) Definiálja a tökéletes rejtjelezést. (2p)
a.) Pénzfeldobás-sorozatot alkot-e a rejtjelezett szöveg? (4 p)
b.) Tökéletes-e a rejtjelezés? (számítással indokoljon) (5 p)

Megoldás:

12. Tökéletes rejtjelezés

Egy titkosítás tökéletes, ha a nyílt (X) és a titkosított szöveg (Y) statisztikailag függetlenek, vagyis kölcsönös információtartalmuk zérus: I(X, Y) = 0. A támadó a megfigyelt rejtett szövegek alapján, erőforrásaitól függetlenül nem képes a nyílt szöveg megfejtésére. A One-Time Pad tökéletes titkosító.

a.) Igen. A rejtjeles szöveg eloszlása ugyanaz marad mint OTP esetén: kommutatív művelet, s az egyik operandus pénzfeldobás-sorozat.
b.) Nem. A nyilt szöveg bemenet és a rejtett szöveg kimenet nem független:
P(y=0|x=0)=P(k=0)=3/4, P(y=0|x=1)=P(k=1)=1/4.

1. Az üzenetek halmaza M = {a, b}, ahol P(a)=1/3, P(b)=2/3. A kulcsok halmaza K= {k1, k2, k3}, ahol P(k1)=1/3, P(k2)=1/2 , P(k3)=1/6. A rejtett szövegek halmaza C= {1, 2, 3}. A rejtjelező kódolás:

	a	b
k1	1	3
k2	2	1
k3	3	2

a.) Tökéletes-e a rejtjelező?

b.) A kulcsok feletti eloszlás módosításával változtatható-e az előző kérdésre adható válasz?

Megoldás:

a.) Nem tökéletes. Pl. P(1/a)=1/3, de P(1/b)=1/2, így nem független a kódoló bemenő és kimenő v.v. .

b.) Igen, lehetséges. Egyenletes a kulcseloszlás esetén tökéletes.

Háttérismeret:

12. Tökéletes rejtjelezés

Egy titkosítás tökéletes, ha a nyílt (X) és a titkosított szöveg (Y) statisztikailag függetlenek, vagyis kölcsönös információtartalmuk zérus: I(X, Y) = 0. A támadó a megfigyelt rejtett szövegek alapján, erőforrásaitól függetlenül nem képes a nyílt szöveg megfejtésére. A One-Time Pad tökéletes titkosító.

1. Tekintsen egy ElGamal rejtjelezőt mod 11 szorzócsoport felett g=2 primitív elemmel. Legyen a titkos kulcs x = 7. Tegyük fel, hogy a véletlen generátorán a következő output 6.

a.) Definiálja az ElGamal rejtjelezést (kódolás, dekódolás)! **(3p)**
b) Adja meg a nyilvános kulcsot! **(2p)**
c.) Kódolja az m=5 nyílt szöveget! **(2p)**
d.) Dekódolja a (9,8) rejtett szöveget! **(3p)**

Megoldás:

a.)

36. El-Gamal rejtjelező

Legyen G egy multiplikatív csoport, g pedig egy generátor eleme.
Titkos kulcsnak válasszuk a K_s=x véletlen elemet a $\{1,2,\dots,|G|\}$ halmazból.
Publikus kulcsnak legyen K_p={X, G, g}, ahol $X=g^x$.

Az $m \in M$ üzenet rejtjelezése:

- kiválasztunk egy véletlen y elemet az M halmazból.
- legyen $Y=g^y$ és $b = X^y \cdot m$
- ekkor a rejtjelezett üzenet: $E_{K_p}(m) = (Y, b)$

Az (Y, b) üzenet dekódolása:

- $D_{K_s}(Y, b) = \frac{b}{Y^x} = \frac{X^y \cdot m}{Y^x} = \frac{g^{xy} \cdot m}{g^{xy}} = m$

b.) $b=2^7=7 \pmod{11}$

c.) rejtett szöveg$=(2^6, 2^{6\cdot 7} \cdot 5)=(9, 2^2 \cdot 5)=(9,9) \pmod{11}$

d.) nyílt szöveg$=8\cdot 9^{-7}=8\cdot(9^{-1})^7=8\cdot(5)^7=8\cdot 5\cdot 5^2\cdot 5^2\cdot 5^2=8\cdot 5\cdot 3\cdot 3\cdot 3=2$

Nevezzük rossz kulcsnak az olyan kulcsot, amelyre a kódoló transzformáció egyben dekódoló transzformáció is!

a.) Keresse meg a $E_k(x)=x+k;\ x,k\in Z_{26}$ shift rejtjelező rossz kulcsait! (10 pont)

b.) Mutassa meg, hogy $E_{a,b}(x)=ax+b,\ a,b,x\in Z_n$ affin rejtjelező esetén $k=(a,b)$ rossz kulcs akkor és csak akkor, ha $a^{-1}=a \pmod n$ és $b(a+1)=0 \pmod n$! (10 pont)

Megoldás:

a) $k=0$, $k=13$: $D_k(E_k(x))=(x+k)+k=x \rightarrow 2k=0 \pmod{26}$

b.) $a(ax+b)+b=x \pmod n$, azaz $a^2x+ab+b=x \pmod n$ a.cs.a., ha $a^2=1 \pmod n$, $ab+b=0 \pmod n$

Tekintsük az alábbi rejtjelezést: a, b, c, d, k∈ {0,1,2,...,28}
nyílt szöveg: x=[a, b],
rejtett szöveg: y=[c, d] , ahol c=(k·a - a) mod 29, d=(a + b·k^2) mod 29
kulcs: k

a.) Van-e megszorítás a lehetséges kulcsokra? (1p)
b.) Adja meg a dekódoló transzformációt! (2p)
c.) Dekódolja az y=[7,22] rejtett blokkot k=3 kulcs esetén. (2p)

Megoldás:

1a.) $a = c \cdot (k-1)^{-1} \bmod 29,\ b = [d - c \cdot (k-1)^{-1}] \cdot k^{-2} \bmod 29,$
$k \neq 0,1$
1b.) $k^{-1} = 10,\ (k-1)^{-1} = 15$, [a,b]=[18,23]
1c.) [a=1, b tetszőleges] nyílt szöveg választása esetén k=c+1.

1. Egy fonetikus jelszó generátor működése a következő: kiválaszt kettő 3 betűs jelszó töredéket, hogy egy 6 betűs jelszót generáljon. A töredékek a következő módon épülnek fel: Magánhangzó – Mássalhangzó – Magánhangzó. Az ABC legyen a 26 betűs angol ABC (Magánhangzó: a,e,i,o,u,y).

a.) Hány különböző jelszó képzelhető el? (2p)
b.) Milyen valószínűséggel lehet egy jelszót megtippelni, ha háromszor tippelhetünk? (2p)
c.) Hány jelszót kell kérni a generátortól, hogy nagy (~40%) valószínűséggel legyen két egyforma (ütköző) jelszó? (4p)

Megoldás:

a.) $26 = 6 + 20$ $N = (6 * 20 * 6)^2 = 720\text{^}2 \approx 2^{19}$

b.) ~3/720^2= 5.8 E-6

c.) Születésnapi paradoxon: ~√720^2= 720

Háttérismeret:

43. Születésnapi paradoxonok

Egy 365 fős csoportból elég $\sqrt{365} \approx 19$ különböző embert kiválasztani ahhoz, hogy kb. $^1/_2$ valószínűséggel legyen közöttük két olyan, akik ugyanazon a napon születtek.

Tehát ha van egy M méretű halmazunk, akkor abból egy $\sqrt{M}$ nagyságrendű részhalmazt véletlenszerűen kiválasztva nagy a valószínűsége annak, hogy két azonos tulajdonságú elem lesz benne.

Egy módosított változata: egy M alaphalmazból véletlenszerűen választva egy r méretű U és V részhalmazt, mekkora az esélye annak, hogy a két részhalmaz metszete nem üres? A válasz: r = $\sqrt{M}$ esetén $P \approx 0.95$

44. Születésnapi paradoxon és hash fv. támadása

A születésnapi paradoxon segítségével egy hash függvény esetén becsülhető egy ütközés megtalálásának valószínűsége. Ha a H(x) hash-függvény kimenete n bites, akkor $2^{\frac{n}{2}}$ darab véletlenül választott üzenet között nagyjából $^1/_2$ valószínűséggel lesz két egyező.

Védekezés ha n-t kellően nagynak választjuk, akkor egy $2^{\frac{n}{2}}$ méretű halmaz ütközésellenőrzése kivitelezhetetlenné válik.

1. Tekintsünk egy mini-RSA rejtjelezőt p1=101 és p2=103 prímekkel. Tegyük fel, hogy egy támadó megfigyelt egy x dokumentumot és a hozzátartozó RSA aláírást, ahol az x dokumentum egy RSA blokk hosszúságú. A támadó úgy szeretné feltörni az aláíró algoritmust (kitalálni az aláíró kulcsot), hogy arra számít, hogy az x dokumentum (természetes szám ábrázolásban) nem relatív prím a modulussal. Nagyobb-e a sikere esélye, mint egy 4 decimális számjegyből álló PIN kód véletlen eltalálásának? (7 pont)

Megoldás:

1. Igen

$$P = \frac{m - \Phi(m)}{m} = 1 - \frac{(p_1 - 1)(p_2 - 1)}{p_1 p_2} = 1 - \frac{(p_1 p_2 - p_1 - p_2 + 1)}{p_1 p_2} = \frac{1}{p_1} + \frac{1}{p_2} - \frac{1}{p_1 p_2} \gg 10^{-4}$$

Háttérismeret:

31. RSA kulcs setup

1. Kiválasztunk két nagy prímszámot: p és q.
2. $m = p \cdot q, \quad \varphi(m) = (p-1) \cdot (q-1)$
3. Válasszunk ki egy tetszőleges $1 < e < \varphi(m)$ számot, amelyre $\left(e, \varphi(m)\right) = 1$
4. Számítsuk ki e inverzét modulo $\varphi(m)$-re: $d = e^{-1} (mod\ \varphi(m))$
5. A kulcs publikus része: K_p = (m, e)
6. A kulcs titkos része: K_s = (d, p, q)

Az x nyílt szöveg kódolása: $y = x^e (mod\ m), \ \ 1 \leq x < m$
Az y rejtett szöveg dekódolása: $x = y^d (mod\ m), \ \ 1 \leq y < m$

1. Tekintsünk egy mini RSA rejtjelezést p_1=23, p_2=71 prímekkel.
   a) Hány olyan x üzenet van ($1 \leq x < m=p_1p_2$), amelynek nincs közös faktorja a titkos prímekkel? (6 p)
   b) Mekkora a valószínűsége, hogy egy véletlenszerű x üzenet m faktorizációjára ad lehetőséget? (8 p)
   c) Adja meg a d kódoló kitevőt, ha e = 3 a publikus kitevő! (8 p)

Megoldás:

a) $\Phi(m)=22*70=1540$

b) 0.057

$$P = \frac{m-\Phi(m)}{m} = 1-\frac{(p_1-1)(p_2-1)}{p_1p_2} = 1-\frac{(p_1p_2-p_1-p_2+1)}{p_1p_2} = \frac{1}{p_1}+\frac{1}{p_2}-\frac{1}{p_1p_2}$$

c) $1540=513\cdot3+1 \rightarrow (-513)\cdot3=(-1)\cdot1540+1 \rightarrow d=-513=1027 \pmod{1540}$

Háttérismeret:

31. RSA kulcs setup

1. Kiválasztunk két nagy prímszámot: p és q.
2. $m = p \cdot q, \quad \varphi(m) = (p-1)\cdot(q-1)$
3. Válasszunk ki egy tetszőleges $1 < e < \varphi(m)$ számot, amelyre $\left(e, \varphi(m)\right) = 1$
4. Számítsuk ki e inverzét modulo $\varphi(m)$-re: $d = e^{-1} (mod\ \varphi(m))$
5. A kulcs publikus része: K_p = (m, e)
6. A kulcs titkos része: K_s = (d, p, q)

Az x nyílt szöveg kódolása: $y = x^e (mod\ m), \quad 1 \leq x < m$
Az y rejtett szöveg dekódolása: $x = y^d (mod\ m), \quad 1 \leq y < m$

5. *Egy játék RSA algoritmus esetén: p1=5, p2=11 prímeket választottuk.*
   a.) Adja meg a legkisebb lehetséges kódoló kulcsot, s az ehhez tartozó dekódoló kulcsot! (15p)
   b.) Mi az x=2 üzenet aláírása? (5p)

Megoldás:

m=55, $\Phi(m)=4\cdot10=40=2\cdot2\cdot2\cdot5 \rightarrow e=3$,
d számítása: $40=13\cdot3+1 \rightarrow 1\cdot40+(-13)\cdot3=1 \rightarrow d=-13=27 \bmod 40$
$D(x)=2^{27}=18 \bmod 55$ $(2^{27}=(2^6)^4\cdot2^3=(9)^4\cdot2^3=(26)^2\cdot8=16\cdot8=128=18 \bmod 55$

Háttérismeret:

31. RSA kulcs setup

1. Kiválasztunk két nagy prímszámot: p és q.
2. $m = p \cdot q, \quad \varphi(m) = (p-1)\cdot(q-1)$
3. Válasszunk ki egy tetszőleges $1 < e < \varphi(m)$ számot, amelyre $\left(e, \varphi(m)\right) = 1$
4. Számítsuk ki e inverzét modulo $\varphi(m)$-re: $d = e^{-1}(mod\ \varphi(m))$
5. A kulcs publikus része: K_p = (m, e)
6. A kulcs titkos része: K_s = (d, p, q)

Az x nyílt szöveg kódolása: $y = x^e (mod\ m), \ \ 1 \leq x < m$
Az y rejtett szöveg dekódolása: $x = y^d (mod\ m), \ \ 1 \leq y < m$