Kriptográfia Jegyzet

Dr. Pethő Attila

2001. január 30.

Előadó: Dr. Pethő Attila <mailto:pethoe@math.klte.hu>
Készítette: Balogh János, Kiss József
LaTeX konverzió: Mudrony László <mailto:mudry@infosrv.tech.klte.hu>
Revision : 1.5

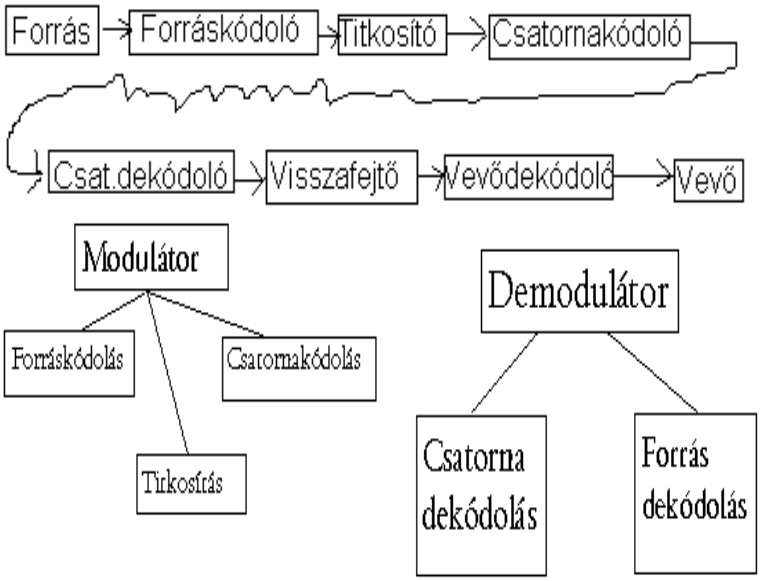
Irodalom:

- D.R. Stinson, Cryptography, Theory an Practice CRC, 1995
- A.Salomaa, Public-key Cryptography, 1990
- N.Kobbitz, a course in number theory and cryptography, 1994
- A.J.Meneres, P.C von Oorshot & S.A.Vanstone, Handbook of applyed cryptography, CRC, 1996
- B.Schneier, Applied Cryptography: Protocolls, Algoritms, and Source Code in C,1996
- `http://www.rsa.com`
- Ködmön József, Kriptográfia, Computer Books, 1999
- Megyeri Zoltán, Titkosírások, ?

Tartalomjegyzék

1 Bevezető

Más megközelítésben:



Forráskódolás feladata: A forrásból érkező jelek redundancia csökkentése egyértelműen dekódolható kóddal (Szabványosított). Ennek a párja a vevődekódoló. (Információelmélet, adatsűrítés)

Titkosító: A szabályos üzenet transzformációja olyan alakra, amelyik az illetéktelenek számára érthetetlen és visszafejthetetlen. (Kriptográfia, titkosítás)

Visszafejtő: Az illetékes vevő módszere a titkosított üzenet dekódolására.

Csatorna kódoló: Az üzenet transzformációja olyan alakra, amely az átvitel során óhatatlanul fellépő hibák automatikus kijavítását teszi lehetővé (Kódelmélet).

Csatorna dekódoló: A csat. kódolás inverz művelete.

A forrás és csatornakódolás elméletét Cl. Shanon ~1950 foglalta először tudományosan elemezhető rendszerbe. A titkosítás, bár gyökerei több ezer évesek, 1974-től (Diffie és Helmann) számítanak a tudományos vizsgálat tárgyának.

A kriptográfia tárgyban elfelejtkezünk a forrás- és a csatornakódolás problémáiról, és a három kódolási terület interakciójáról, csak a titkosítás problémáival foglalkozunk.

A bizalmas üzenettovábbítás nem modern igény. Az előadás során is elemezni fogjuk Julius Caesar római császár titkosítási technikáját, amelyik több mint 2000 éves. Ismerünk azonban még régebbi eljárásokat is üzenetek titkosítására. A számítógép hálózatok elterjedéséig azonban ezek a feladatok elsősorban a hadtudományok és a nemzetközi diplomácia szigorúan titkos eszköztárához tartoztak. A feladatuk az volt, hogy bizalmas üzeneteket oly módon alakítsanak át, hogy az adó által kódolt üzenetet az illetékes vevő dekódolni, értelmezni tudja, de az illetéktelen lehallgató ne tudja megfejteni. Az elmúlt évtizedben alapvetően megváltozott a helyzet. A bizalmas üzenettovábbítási igény bevonult (a számítógépes) mindennapjainkba.

Alkalmazási területek:

- Bizalmas információk cseréje a gazdasági és politikai élet minden területén.
- Banki tranzakciók.
- Gazdasági egységek belső információ áramlása.
- Szoftverek védelme.
- Adatbázisok védelme.
- Lehallgathatatlan telefonvonalak.
- Elektronikus publikáció.
- Elektronikus kereskedelem.
- Elektronikus pénz

Egy modern kriptorendszer fő feladatai:

- Üzenetek titkosítása (secrecy)
- Felhasználó azonosítás (identification)
- Üzenet eredetiségének biztosítása (authentication)

- Üzenet változatlanságának biztosítása (integrity)
- Digitális aláírás (key exchange)
- Kulcs szétosztás (key distribution)

A kriptográfia megjelenése kriptográfia és alkalmazása egy sor társadalmi és jogi problémát vetett fel és vet folyamatosan fel. Ki és milyen feltételekkel juthat hozzá elektronikus eszközökkel tárolt, küldött információkhoz? Milyen bonyolultan szabad üzenetet titkosítani? Ezek és hasonló kérdéseknek a megválaszolása a politika és a jog területe, amelyre nem kívánunk tévedni. Nem kívánok a kérdéskör egyre izgalmasabb technikai vonásaival sem foglalkozni. Például azzal, hogy hogyan lehet egy több száz vagy ezer számítógépből álló rendszert biztonsági szempontból felügyelni. Milyen intézkedéseket kell tenni, ha egy ilyen rendszerbe betörési kísérlet vagy betörés történt? Mit és hogyan kell módosítani a működési, működtetési feltételek közül? A fenti problémák megválaszolásához számos törvény, nemzetközi, országos és regionális ajánlás született és fog születni a jövőben. Ezek a környezettől függően állandóan változó feltételeket jelentenek, de ma már vannak a kriptográfiának egyértelműen megfogalmazható és tartós paradigmái is.

A kódolás vonatkozásában két dologra fogunk koncentrálni:

- titkosító algoritmusok
- protokollok

Elemezni kívánjuk a támadási lehetőségeket is ugyanezen szempontokból. Miért kell a támadási, feltörési módszerekkel is foglalkozni? Egy bizalmas információkat kezelő rendszert természetesen véletlen szerencsével is fel lehet törni. Ezt nevezhetnénk a ôvak tyúk is talál szemetö módszernek. Egy ilyen feltörésnek azonban jól megkonstruált kriptorendszer esetén nagyon kicsi az esélye. Tekintettel arra, hogy a modern kriptorendszerekben alkalmazott módszerek minden érdeklődő számára hozzáférhetőek (nyilvánosak) jó tudni, hogy melyek ezek ismert gyenge pontjai. A rendszeradminisztrátor, vagy inkább a rendszer maga védekezni tud az ismert betörési módszerek ellen. Ha egy védelmi rendszerről tudjuk, hogy a modern technikával szemben tehetetlen, akkor újat kell bevezetni. Ki építene ma már végvárakat hazának megvédésére? Az is hasznos lehet, ha tudjuk, hogy melyek a sebezhető pontjaink. A ômásodik védelmi vonalatö ezek biztosítására kell átcsoportosítani. Ez azt jelenti, hogy az algoritmusokat és protokollokat módosítani szükséges, vagy teljesen le kell cserélni.

1.1 A kriptorendszer fogalma

Az első előadáson meghatároztok az üzenettovábbítás általános modelljét, és rámutattunk, hogy a kriptográfiában e modell együttes részfeladatával foglalkozunk. Leegyszerűsítve tehát a következő szituáció érdekel minket:

Az adó az angol irodalomban tradicionálisan Alice (a) bizalmas üzenetet küld a nyilvános csatornán a vevőnek, akit Bob (B) személyesít meg. A bizalmas üzenetet természetesen úgy akarja kódolni, hogy

- A kódolás könnyen, gyorsan történjen.
- A dekódolás szintén gyorsan megvalósítható legyen. Ehhez A és B előzetes egyeztetése szükséges.
- A lehallgató nehezen tudja megfejteni az üzenetet, feltéve, hogy A és B megállapodását nem ismeri.

A klasszikus modellben Alice és Bob megállapodnak valamilyen általuk bonyolultnak talált eljárásban, amellyel üzeneteiket titkosítják, és ezt mindketten titokban tartják. Ha elég ügyesek és tisztességesek, akkor a lehallgató csak az elfogott, kódolt üzenetek elemzésével nyerhet információt az aktuális adatcsere tartalmáról. A modern kriptorendszerekben Bob már Alice megbízhatóságát sem tételezi fel, csak ő ismeri a dekódoláshoz szükséges többlet információt. Ekkor tehát Alice és Bob megállapodnak valamilyen módszerben, ezt és a titkosító kódot bárki ismerheti, de az üzenetet csak Bob tudja visszafejteni. Ez a nyilvános kulcsú titkosítás alapötlete.

Adó	Alice (A)	
Vevő	Bob (B)	
Lehallgató	Eve (E)	Passzív, csak figyeli az adót
	Mallory (M)	Aktív, be is avatkozik

1.2 A Kriptorendszer matematikai modellje

Az adó lehetséges üzenetei egy Σ_A abc feletti véges halmaz elemei. Az üzenetek egy $P \subseteq \Sigma_B^*$ nyelv elemei. A P halmazt a P (plaintext) jellel szimbolizáljuk.

Az adó és vevő megállapodnak abban, hogy a $w \in P$ szavakat egy Σ_B^*-beli szóra konvertálják, ahol B szintén egy véges abc. Megállapodnak tehát egy $E : P \to \Sigma_B^*$ leképzés családban. Ezt hívják titkosító vagy kódoló (encryption) függvénycsaládnak.

Az E függvénycsalád elemei kétváltozós függvények, egyrészt a kódolandó $w \in P$-től, általában azonban w előre rögzített hosszúságú, egymást követő résszavaitól

függnek. Az E elemei függnek továbbá egy k_E kódoló kulcstól, amelyik egy $K \subseteq \Sigma_C^*$ kulcstér elemei.

A kódolásban tehát kell egy $E : P \to \Sigma_B^*$ leképezés és egy $k_E \subseteq K \subseteq \Sigma_C^*$ kulcs, ahol E a k_E-től függ. A kódolás után az $w \in P$ szó egy $\pi \in C \subseteq \Sigma_B^*$ szóba megy át. A C halmazt a titkosított üzenetek (ciphertext) halmazának nevezzük.

Az π szó persze a vevő és a lehallgató számára érthetetlen. Ezért π-t vissza kell alakítani olyan formába, amelyik a vevő számára érthető. Erre szolgál a D dekódoló függvénycsalád és az ezek elemeihez tartozó dekódoló kulcsok, amelyekre teljesül, hogy bármely π^{-1} ahol π^{-1}-re $D_{k_D}(E_{k_E}(w) = w)$.

Tehát, ha az $w \in P$ szót az E függvénnyel és a k_E kulccsal kódoljuk, és az eredményre a neki megfelelő D függvényt és k_D kulcsot alkalmazzuk, akkor visszakapjuk az eredeti w üzenetet. A kódolásnak tehát visszafejthetőnek, matematikai kifejezéssel invertálhatónak kell lennie, legalább is a megengedett üzenetek halmazán.

Ahhoz, hogy valóban kriptográfiai rendszerről beszélhessünk, még további feltételeknek is teljesülnie kell. Ezek:

- $E_{k_E}(w)$ legyen könnyen kiszámítható, azaz E_{k_E} legyen egy polinomiális időben kiszámítható függvény. A polinomialitás alatt itt a gyakorlatban lineáris vagy kis kitevőjű polinomiális időben kiszámítható függvényt értünk.
- Hasonlóképpen D_{k_D} is könnyen kiszámítható legyen az előbbi értelemben.
- Azonban, ha k_D-t nem ismerjük, akkor a $D_{k_D}(w)$ kiszámítása nehéz legyen. A dekódoló kulcs ismerete nélkül tehát a kódolt üzenetet nehezen lehessen visszafejteni.

A fenti három követelménynek eleget tevő titkosítási rendszert máig sem ismerünk. Vannak azonban olyan, a gyakorlat próbáit jól kiálló rendszerek, amelyek jelenlegi ismereteink szerint jól megfelelnek a megfogalmazott követelményeknek.

Más megközelítésben:

Üzenet: Legyen A egy véges abc. Az üzenet egy véges szó A felett. A lehetséges üzenetek halmazát P-vel jelöljük. $P \subseteq A^*$ (plaintext). A^* az A fölötti véges szavak (üzenetek) halmaza.

Kódoló (titkosító, encryption): E egy leképzés a $P \times K$ halmazból a B^* halmazba, ahol $K \subseteq A^*$ és B és C véges abc-k. $E : P \times K \to B^*$

Kulcstér: K-t a rendszer kulcsterének nevezzük (keyspace).

Dekódolás (decryption): Ez is egy leképzés a $B^* \times K$-ból A^*-ba. $D : B^* \times K \to A^*$. Minden E titkosító függvényhez és k_E kulcshoz léteznie kell olyan D dekódoló függvénynek és k_D kulcsnak, hogy $\forall w \in P$-re $D(E(w, k_E), k_D) = w$.

Kódolt szöveg (ciphertext): $E(w, k_E) \in C \subseteq B^*$

Kriptorendszer: $< A, C, K >$ ahol A^* lehetséges szavak halmaza, $C \subseteq B^*$ kódolt üzenetek halmaza, $K \subseteq A^*$ kulcstér.

1.3 Szimmetrikus és aszimmetrikus kriptorendszerek

Egy kriptorendszert szimmetrikusnak nevezünk, ha $k_E = k_D$ és aszimmetrikusnak, ha $k_E \neq k_D$. Az első esetben az adó és vevő nem csak a kódoló/dekódoló algoritmusban, hanem a kódoló/dekódoló kulcsban is megegyezik. Tehát legalább két szereplő ismeri a dekódoló kulcsot. A másodikban azonban csak a vevő rendelkezik a dekódoláshoz szükséges ismeretekkel. Így a lehallgató az üzenet megfejtéséhez a vevőtől kell megszerezni a megfelelő kulcsot.

Egy kriptorendszer szimmetrikus, ha a küldő kulcs megegyezik a dekódoló kulccsal $k_E = k_D$ (Ált:privát kulcsú)

Aszimmetrikus, ha $k_E \neq k_D$. (nyilvános kulcsú)

2 Klasszikus kriptográfiai rendszerek

Az üzenetek titkosítása, most a XXI. században már nyugodtan mondhatjuk, több ezer éves múltra tekint vissza. Julius Caesarról tudjuk, hogy alkalmazta az alábbi eltolásos titkosítást, amelyet a magyar abc-re alkalmazva illusztrálunk:

2.1 Eltolásos titkosítás

Az abc minden betűjének feletessünk meg egy számot, azt amelyet a felsorolásban elfoglal. Pl. A=1, ... Z=35. Az eltolásos titkosítás lényege az, hogy válasszunk egy k eltolási paramétert és az üzenetben az x kódú számnak az x+k kódú számot feleltessük meg természetesen úgy, hogy ha x+k>35, akkor annak a maradékát vesszük 35-tel osztva. Tehát pl. Z kódja K, 10-es eltolással, hiszen $Z = 35$, $35 + 10 = 45 \, mod \, 35 = 11 = K$.

Az eltolásos titkosítás esetén

$A = \{A, B, \ldots Z\}, P = A^* = C, K = \{1, \ldots 35\}$,
$E : x + k \, mod \, 35$, $D : x - k \, mod \, 35$

Példa: k=10

```
A KRIPTOGRÁFIA EGY MODERN TUDOMÁNY
I RYÕWAÚÓYÍOÕI MÓG TÚLMYU AÁLÚTÍUG
```

2.2 Helyettesítéses titkosítás

Ez egy nagy család.

$P = C = A^*$, mint az előbb.
$K = A$ permutációinak a halmaza.

$E : \pi$ ahol $\pi \in K$
$D : \pi^{-1}$ ahol $\pi^{-1}K$ a π inverze.

Számítógéppel könnyen lehet véletlen permutációt generálni. 35 elemre ezeket egyszerű tárolni. Hátránya, hogy visszafejteni is egyszerűen lehet.

Példa:

```
A KRIPTOGRÁFIA EGY MODERN TUDOMÁNY
M UOPRGAŐOSVPM QŐE ÍALQOÚ GŐLAÍSÚE
```

2.3 Affin titkosítás

A számítógépek elterjedéséig a helyettesítéses titkosítás nehezen volt használható. Speciális ismeretek és eszközök kellettek ahhoz, hogy az üzeneteket kódolják és dekódolják. Egy véletlen helyettesítést nehéz fejben tartani, ezért azt le kell írni, és így a jegyzet illetéktelen kezekbe is kerülhet. Az eltolásos titkosítás kis kulcstere miatt igen könnyen megfejthető. A helyettesítéses titkosítás kulcstere megfelelő méretű, megegyezik $|A|!$-al, ahol $|A|$ az A elemszáma. Ugyanakkor a kódoló és dekódoló kulcs bonyolult.

Ezeken a problémákon igyekszik segíteni a Julius Caesar által kitalált affin titkosítás. Ez általánosabb az eltolásos módszernél és a helyettesítéses titkosítás speciális esete.

Legyen

$P = C = A^*$
$K = \{(a, n) : (a, |A|) = 1, 0 > a, n < |A|\}$
$E : E_{(a,n)}(x) = ax + n \mod |A|$
$D : D_{(a,n)}(y) = a^{-1}(y - n) \mod |A|$

Példa:

```
A KRIPTOGRÁFIA EGY MODERN TUDOMÁNY
Z MVL??ÜTVÍKLZ STI DÜJSVN ?WJÜDÍNI
```

2.4 Eltolásos módszer

$P = C = A$
$P = \{0, 1, \ldots 35\}^*, \{A, B, \ldots Z\}^* = C$
$K = \{0, \ldots 34\}$
$E(x, k) = x + k \, mod \, |A|, 0 < k < |A|$
$D(y, k) = y - k \, mod \, |A|$
$D(E(a, k), k) = D(a + k \, mod \, 35, k) = (a + k \, mod \, 35) - k \, mod \, 35 = a$

Rossz, mert kicsi a kulcstere.

2.5 Helyettesítéses módszer

$P = C = A$
$P = C = \{A, B, \dots C\}$
K=$\{$P permutációinak halmaza$\}$=$\{$ a $P \to P$ bijektív leképezések$\}$
$E(a, f) = f(a)$
$D(b, f) = f^{-1}(B)$

Eltolásos módszer általánosítása.
$|K| = |A|!$ Nagy kulcstérrel rendelkezik.
Bijektív leképezést nehéz nyilvántartani.

2.6 Affin (Caesar) módszer

$P = C = \{A, \dots Z\}$
$K = \{(n, k) : 0 < n, k < |P|, (n, |P|) = 1\}$ (azaz n és $|P|$ relatív prímek)
$E(a, k_E) = an + k \, mod \, |P|$
$D(a, k_D) = (a - k)n^{-1} \, mod \, |P|$, ahol $n^{-1} * n \equiv 1 (mod \, |P|)$

2.7 Egy kis aritmetikai kitérő

Az affin titkosítás kulcsterébe azon (a, n) párok tartoznak, amelyekre $(a, |A|) = 1$, azaz és $|A|$ legnagyobb közös osztója 1. Kérdés mekkora ez a kulcstér, és hogy a dekódoló függvény valóban definiált és korrekt eredményt szolgáltat. Még arra is választ szeretnénk kapni, hogy milyen könnyen lehet az ôaö paramétert meghatározni. Egyszerűsítsük a jelölésünket azzal, hogy bevezetjük az $m = |A|$ jelet. Ekkor persze m^3 1 egész szám. ($m = 1$ nem túl izgalmas a szempontunkból.) Az (a, m) meghatározására a legjobb ismert eljárás az euklideszi algoritmus, amelyik több mint kétezer éves.

2.7.1 Euklideszi algoritmus (egyszerű)

Input: $a, m \in Z$
Output: (n, m)

- if n=0 { if a=0 output(“Hibás Adat”) }
  else output(a)
- do {
  $r \leftarrow a \, mod \, m$
  $a \leftarrow m$
  $m \leftarrow r$
  } while $m = 0$

- output(a)

Tétel: Az euklideszi algoritmus korrekt, kimenetele (a, m).

Bizonyítás: Ha $m = 0$, akkor az algoritmus nyilvánvalóan korrekt. Legyen $d = (a, m)$, ahol $m \neq 0$. A ciklus megkezdésekor $d|a$ és $d|m$. Végrehajtva a ciklust $r = a - m\left[\frac{m}{a}\right]$ azaz $d|a$ és $d|m$ miatt $d|r$. A $d|a$ és $d|m$ feltétel tehát mindig teljesül a ciklus elején. Amikor a ciklus befejeződik, akkor $m = 0$ és $d|a$, azaz (a, m) a kimenet.

Fordítva tegyük fel, hogy az algoritmus kimenete d, amelyik a ciklus végrehajtása során az utolsó nem nulla m érték volt. Továbbá $r = 0$ miatt $d|a$. Visszafelé figyelve könnyen látható, hogy d osztja a bemenetkor is a-t és m-et, azaz $d|(a, m)$. A $d|(a, m)$ és $(a, m)|d$ relációkból következik, hogy $d = (a, m)$.

A következő kérdés az, hogy ha $(a, m) = 1$, akkor lehet-e és ha igen hogyan olyan X számot találni, amelyre $ax \equiv 1(mod\, m)$.

Állítás: $ax \equiv 1(mod\, m)$ akkor és csakis akkor ha y<Z, hogy $ax + my = 1$.

Bizonyítás: $ax \equiv 1(mod\, m)$ $m|ax-1$ y<Z, hogy $ax-1 = my$ $ax+m(-y) = 1$. Tehát a $mod\, m$ inverzét meghatározni ekvivalens az $ax + my = 1$ egyenlet egy megoldásának a meghatározásával. Az euklideszi algoritmus kis megbarkácsolásával ezt az egyenletet is, sőt az $ax + my = (a, m)$ -t is meg tudjuk oldani.

Adott m-re, amelyik relatív prim m-hez keressünk olyan x-et, amelyre $nx \equiv (mod\, m)$

- $m|mx - 1$
- y hogy $nx - 1 = m(-y)$
- $nx + my = 1$

2.7.2 Euklideszi algoritmus (bővített)

Input: $a, m \in Z$
Output: $x, y, d \in Z$, ahol $ax + my = d = (a, m)$

1. $\begin{pmatrix} u_1 & v_1 \\ u_2 & v_2 \\ u_3 & v_3 \end{pmatrix} \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ n & m \end{pmatrix}$
2. repeat
   $r \leftarrow u_3\, div\, v_3 = \left[\frac{u_3}{v_3}\right]$
   $\begin{pmatrix} u_1 & v_1 \\ u_2 & v_2 \\ u_3 & v_3 \end{pmatrix} \leftarrow \begin{pmatrix} u_1 & v_1 \\ u_2 & v_2 \\ u_3 & v_3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix}$
   until $v_3 = 0$

3. Output (u_1, u_2, u_3), Return

Tétel: Az euklideszi algoritmus korrekt, és ha $|n| < |m|$ akkor legfeljebb $O(log\,|n|(log\,|m|-log\,|d|+1))$ lépés után megáll.

Biz: d=(n,m) Ha m=0 akkor kész.

Legyen $n_0 = n, n_i = m$
$n_0 = q_0 n_1 + n_2 (r = n_2)$ {*}
$n_1 = q_1 n_2 + n_3$
$n_k = g_k n_{k+1} + n_k + 2$
Ha n_{k+2} akkor a Repeat ciklus befejeződik.

Az eljárás befejeződik, mert $|n_1| > |n_2| > \ldots > |n_{k+1}|$ természetes számok sorozata, csökkenő, megszakad. $d|n_0, d|n_1, d|n_2, d|n_{k+1}$ osztja az algoritmus kimenetét $d' = n_{k+1}$. Legyen az algoritmus kimenete $d' = n_{k+1}, d'|n_k, d'|n_{k-1}, d'|n_1, d'|(n,m) = d$.

Tehát $d'|d$ és $d'|d$ $d = d'$.

Végül azzal a kérdéssel foglalkozunk, hogy ha $m > 2$ egész, akkor hány eleme van az $M = \{(a,m) : 0|a < m, (a,m) = 1\}$ halmaznak. Az M elemszámát $\varphi(m)$-mel jelöljük és φ-t Euler féle φ függvénynek nevezzük. Mielőtt bizonyítás nélkül közölnénk egy tételt, megmutatjuk, hogy hogyan lehet adott m-re az M halmaz elemeit meghatározni. Módszerünket először egy példával illusztráljuk. Legyen $n = 30$. Írjuk fel a $0 \ldots 29$ számokat.

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29

Húzzuk ki a táblázatból 0-t, majd keressük meg az első prímszámot, ami 30-at osztja. Ez 2. Húzzuk ki a táblázatból 2 többszöröseit. Ezek nem lehetnek a 30-cal relatív prímek. A ki nem húzott számok közül keressük meg a legkisebb prímszámot, amelyik 30-at osztja. Ez 3. Folytassuk az eljárást, amíg a legkisebb még nem vizsgált prímszám kisebb, mint $\sqrt{30}$. Ez 5. A szitálás után megmaradt számok {1,7,11,13,17,19,23,29} alkotják az M halmazt $n = 30$-ra. Azaz $j(30) = 8$.

Hogyan határoztuk meg az M halmazt? Először is kihúztuk 2 többszöröseit, szám szerint 15-öt. Majd a 3 és az 5 többszöröseit, melyek száma 10, illetve 6. Tehát $\varphi(m)$=30-15-10-6. Ez így nem jó, mert azokat a számokat, amelyek 2-vel és 3-mal oszthatóak kétszer is levontuk. Tehát ezek számát hozzá kell adni az összeghez, azaz 30-15-10-6+5+3+2-1=8.

Ezt másképpen is írhatjuk:

$\varphi(30) = 30 - \frac{30}{2} - \frac{30}{3} + \frac{30}{6} - \frac{30}{5} + \frac{30}{15} - \frac{30}{30} = 30(1 - \frac{1}{2} - \frac{1}{3} + \frac{1}{6} - \frac{1}{5} + \frac{1}{15} - \frac{1}{30}) =$
$= 30\left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{3} - \frac{1}{5} + \frac{1}{15}\right) = 30\left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{3}\right)\left(1 - \frac{1}{5}\right)$

A fenti gondolatmenet tetszőleges m-re megismételhető, és akkor kapjuk a következő tételt.

Tétel: Legyen $p_1^{a_1} \dots p_t^{a_t}$, ahol $p_1 < p_2 < \dots < p_t$ prímszámok és $a_1 \dots a_t > 1$ (prímtényezös felbontás)

$$\varphi(m) = m\left(1 - \frac{1}{1-p_1}\right) \dots \left(1 - \frac{1}{1-p_t}\right)$$

A $\varphi(m)$ függvény tulajdonságaira később, az RSA algoritmus vizsgálatánál visszatérünk.

2.8 Az affin titkosítás és a helyettesítéses titkosítás megfejtése

Visszatérve tehát az affin titkosításra megállapodhatunk, hogy ha $(a, |A|) = 1$, akkor létezik olyan $a^{-1} \in Z$, hogy $aa^{-1} \equiv a(mod\, m)$ és a^{-1} a bővített euklideszi algoritmussal meghatározható. Ha most $y \equiv ax + n(mod\, |A|)$, akkor $a^{-1}(y-n) \equiv a^{-1}(ax+n-n) = a^{-1}ax = x(mod\, |A|)$ és mivel $0 < x < |A|$, így $a^{-1}(y-n) = x$. Tehát a dekódoló függvény korrekt. Azt is megállapíthatjuk, hogy $|K| = |A|\varphi(|A|)$. Belátható, hogy létezik olyan c konstans, hogy $\frac{\varphi(m)}{m} > \frac{c}{log\, log\, m}$, azaz $\varphi(m) > \frac{cm}{log\, log\, m}$, tehát $|K| > c\frac{|A|^2}{log\, log\, |A|}$. Ha például $|A| = 35$, akkor $\varphi(|A|) = 35\left(1 - \frac{1}{5}\right)\left(1 - \frac{1}{7}\right) = 24$. Tehát $|K| = 35 * 25 = 840$. Ez sem túl sok, de ha $|A| = 2^{64}$, akkor $|K| = 2^{16} * 2^{15} = 2^{31} = 2MB$, azaz pl. képek vagy hangdokumentumok titkosítására már hatékonyan alkalmazható.

A helyettesítéses titkosításnak a kis kulcstér mellett nagy hiányossága az, hogy a természetes nyelvekben a karakterek (betűk) előfordulásának a gyakorisága igen szigorú szabályoknak felel meg. A táblázatban feltűntettük az angol, német, finn, francia, olasz és spanyol nyelvekben a leggyakoribb betűk előfordulásának a gyakoriságát. Hasonló tendencia érvényes a magyar nyelvre is.

Eng	Ger	Fin	Fra	Ita	Spa	Hun
E 12,31	E 18,46	A 12,06	E 15,87	E 11,79	E 13,15	AÁ 13,75
T 9,59	N 11,42	I 10,59	A 9,42	A 11,74	A 12,69	EÉ 11,92
A 8,05	I 8,02	T 9,76	I 8,41	I 11,28	O 9,49	T 7.67
O 7,94	R 7,14	N 8,64	S 7,90	O 9,83	S 7,60	OÖ 6,91
N 7,19	S 7,04	E 8,11	T 7,26	N 6,88	N 9,95	L 6,86
I 7,18	A 5,38	S 7,83	N 7,15	L 6,51	R 6,25	N 6,31
S 6,59	T 5,22	L 5,86	R 6,46	R 6,37	I 6,25	S 6,05
R 6,03	U 5,01	O 5,54	U 6,24	T 5,62	L 5,94	K 4,42
H 5,14	D 4,49	K 5,20	L 5,34	S 4,98	D 5,58	I 4,08

A karaktereket három osztályba szokás sorolni: nagy, közepes és kis gyakoriságú. A helyettesítéses titkosítás nem módosítja a karakterek előfordulásának a gyakoriságát, hanem csak azt, hogy a betűket milyen jellel jelöltük. Ezek szerint,

ha egy kriptoanalitikus kap egy titkosított üzenetet (C), amely elég hosszú és tudja, hogy a természetes nyelvből helyettesítéses titkosítással keletkezett, akkor a következőket teszi:

- Megpróbálja megállapítani, hogy milyen nyelven készült az eredeti szöveg (Ez lehet, hogy csak az elemzés későbbi lépéseiben alakul ki.)
- Megállapítja C-ben a karakterek előfordulásának a gyakoriságát. Ez egyszerű leszámlálással és osztással történik. Ebből kiderül, hogy melyek a leggyakoribb karakterek C-ben.
- Kicserélve C-ben a leggyakoribb 2-3 karaktert a természetes nyelvben leggyakoribb karakterekkel megpróbál minél hosszabb összefüggő szövegrészletet azonosítani. Ezzel további karaktereket tud azonosítani a kisebb előfordulású jelek közül.
- Végül megfejti az egész üzenetet.

Manapság hatékony nyelvi elemzők állnak rendelkezésre a dekódolás támogatására.

3 DES (Date EEncription Standard)

64 bites üzenet blokkokat titkosít, formailag 64, de lényegében 56 bites kulcsok felhasználásával. Ez egy többfázisú (szorzat) titkosító rendszer.

Az algoritmus nagyvonalú leírása:

Input: legyen $u \in Z_2^{64}$ az üzenet egy 64 bites blokkja, $k \in Z_2^{64}$
Output: $v = DES(u, k) \in Z_2^{64}$

Paraméterek:
$IP : Z_2^{64} \to Z_2^{64}$ bijektiv leképezés
$IP^{-1} : Z_2^{64} \to Z_2^{64}$ bijektiv leképezés

1. $u_0 \in IP(u)$, $u_0 = L_0R_0$; ahol $R_0, L_0 \in Z_2^{32}$
2. for $i \leftarrow 1 \ldots 16$ do {
   $L_i \leftarrow R_{i-1}$
   $R_i \leftarrow L_{i-1} \otimes f(R_{i-1}, K_i)$
   };
3. $v \leftarrow IP^{-1}(R_{16}, L_{16})$

Az f függvényt és a K_i -t az alábbiakban definiáljuk. $\otimes = xor$

Az f(A,J) függvény leírása:

Input: $A \in Z_2^{64}$, $J \in Z_2^{48}$
Output: $f(A, J) \in Z_2^{64}$

Paraméterek:
$E : Z_2^{64} \to Z_2^{64}$ többértékű leképezés
$P : Z_2^{64} \to Z_2^{64}$ permutáció
$S_1 \dots S_8 : Z_2^{64} \to Z_2^{64}$ injektiv leképezés

1. Számítsuk ki E(A)-t. Ennek során minden bitet felhasználunk, 16 bitet kétszer.
2. $B \leftarrow E(A)?J$ és bontsuk fel $B \in Z_2^{48}$-at 8 db 6 bites sztring szorzatává. $B = b_1 b_2 \dots b_8$
3. A $B_i \in Z_2^6$-ra $1 < i < 6$ alkalmazzuk az S_i S-boxot.
   Ezek 4 sorból és 16 oszlopból álló táblázatok. Ha $B_i = b_1 b_2 \dots b_6$, akkor $b_1 b_6$-ot és $b_2 \dots b_5$ kettes számrendszerbeli számnak tekintve keressük meg S_i-ben a $b_1 b_6$ sorának és $b_2 \dots b_5$ oszlopának találkozásában lévő számot, c_i-t. Erre teljesül, hogy $0 < c_i < 15$, így $c_i \in Z_2^4$. Legyen $c_i \in S_i(B_i)$
4. $C \leftarrow c_1 \dots c_8$
   output $P(C) \in Z_2^{32}$

A kulcsok kiszámítása: az algoritmus a K 8, 16 ... 64. bitjét paritásellenőrző bitnek tekinti és ezeket ignorálja.

Input: $K \in Z_2^{64}$
Output: $k_1 \dots k_{16}$

Paraméterek:
$PC - 1 : Z_2^{64} \to Z_2^{64}$ permutáció
$PC - 2 : Z_2^{56} \to Z_2^{48}$ permutáció

1. $K = (C_0, D_0) \leftarrow PC - 1(K)$
2. for $i \leftarrow 1 \dots 16$ do {
   $C_i \leftarrow lshift_i(C_{i-1})$
   $D_{i+1} \leftarrow lshift_i(D_{i-1})$
   ($lshift_i$ balra eltolást jelent i-től függően 1 vagy 2 pozícióval. Ha i=1, 2, 9 és 16, akkor 1-el kell eltolni, különben kettővel történik az eltolás)
   $K_i \leftarrow PC - 2(C_i D_i)$
   output K_i
   }

Az f függvény kiszámításának a folyamatábrája

A dekódolás ugyanígy történik azzal a különbséggel, hogy a kulcsokat fordított sorrendben, azaz $K_{16}, K_{15} \dots K_1$ alkalmazzuk.

Ez az eljárás valóban korrekt, mert $R_i = L_{i+1}, i = 0 \dots 15$ és $R_{i+1} = L_i$, $f(R_i, K_{i+1}) = L_i$, $f(L_{i+1}, K_{i+1})$ miatt $L_i = R_{i+1}$, $f(L_{i+1}, K_{i+1})$, $i = 0 \dots 15$

3.1 A DES dekódolása

Tegyük fel, hogy az $u \in Z_2^{64}$ üzenetet a k_E kulccsal kódoljuk és eredményül a $v \in Z_2^{64}$ értéket kapjuk. Azt állítjuk, hogy ha v-re a DES algoritmussal, de a kulcsokat fordított sorrendben alkalmazva transzformáljuk, akkor az eredeti üzenetet kapjuk vissza.

Legyen tehát DES bemenete $v = IP^{-1}(R_{16}, L_{16})$ és a kulcsokat alkalmazzuk v-re fordított sorrendben, azaz először K_{16} majd $K_{15} \dots K_1$-et.

Az első lépésben $v_0 = (LE_0, RE_0) = IP(IP^{-1}(R_{16}, L_{16})) = (R_{16}, L_{16})$, azaz $LE_0 = R_{16}$, $RE_0 = L_{16}$.

Azt állítjuk, hogy a dekódolás során minden i-re $LE_i = R_{16} - i$ és $RE_i = L_{16-i}$. Ez az előző miatt nyilván igaz $i = 0$-ra.

Nézzük először az $i = 1$ re. Ekkor $LE_1 = RE_0 = L_{16} = R_{15}\, RE_1 = LE_0 \otimes f(RE_0, K_{16}) = R_{16} \otimes f(L_{16}, K_{16}) = (L_{15} \otimes f(R_{15}, K_{16})) \otimes f(R_{15}, K_{16}) = L_{15}$, hiszen a xor művelet asszociatív. Tehát az állítás igaz $i = 1$-re.

Most tegyük fel, hogy az állítás igaz $0 < i < 15$-re. Akkor $LE_{i+1} = RE_i = L_{16-i} = R_{16-i-1} = R_{16-(i+1)}$ és $RE_{i+1} = LE_i \otimes f(R_i, K_{16-i}) = R_{16-i} \otimes f(L_{16-i}, K_{16-i}) = (L_{16-i-1} \otimes f(R_{16-i-1}, K_{16-i})) \otimes f(L_{16-i}, K_{16-i}) = L_{16-(i+1)} \otimes (f(_{L16-i}, K_{16-i}) \otimes f(L_{16-i}, K_{16-i})) = L16_{-(i+1)}$

Végezetül tehát a 16. menet után $LE_{16} = R_0$ és $RE_{16} = L_0$. Tehát az IP^{-1} permutációt az $(RE_{16}, LE_{16}) = (L_0, R_0)$-ra alkalmazzuk, azaz a kimenet $IP^{-1}(L_0, R_0) = IP^{-1}(IP(u)) = u$ lesz.

A dekódolás tehát korrekt.

3.2 A DES konstrukciójánál alkalmazott elvek

1. Az S-boxok minden sorában a $0 \dots 15$ számok permutációi vannak.
2. Egyetlen S-box sem eredményezheti az input lineáris vagy affin függvényét.
3. Megváltoztatva a bemenet 1 bitjét bármely S-box kimenete legalább 2 bitben különbözzék.
4. Bármely S-boxra és $x \in \{0,1\}^6$-ra $S(x)$ és $S(x \otimes 001100)$ legalább 2 bitben különbözzék.

5. Bármely S-boxra, $x \in \{0,1\}^6$-ra és $e, f \in \{0,1\}$-re $S(x) \neq S(x \otimes 11EF00)$

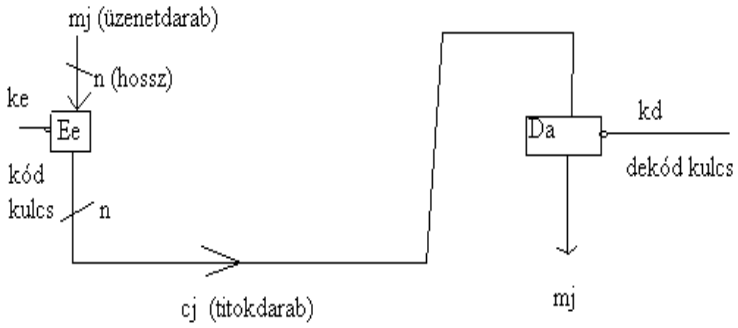
1977-ben Diffie és Hellmann leírtak egy olyan VLSI áramkört, amely 10^6 kulcsot vizsgálnak meg másodpercenként. Egy olyan gép, amelyik 10^6 ilyen chipet tartalmazna egy nap alatt végigvizsgálhatja a teljes kulcsteret és kb 20.000.000 US $-ba került volna.

1993-ban Michael Wiener olyan megoldást irt le, amellyel az összes DES kulcs átlagosan 1.5 nap alatt átvizsgálható és kb 100.000 US $-ba kerülne.

1992-ben a DEC bejelentett egy olyan DES chipet, amelyik 1 Gbit/sec. Sebességgel képes titkosítani az üzenetet.

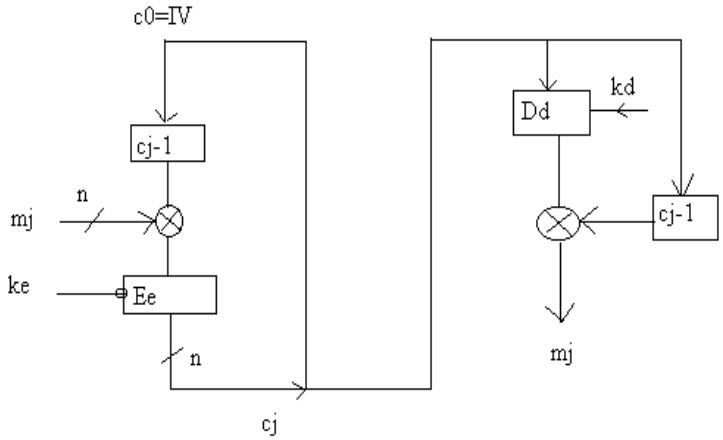
3.3 A DES legfontosabb alkalmazási módjai

ECB (electronic codebook mode)



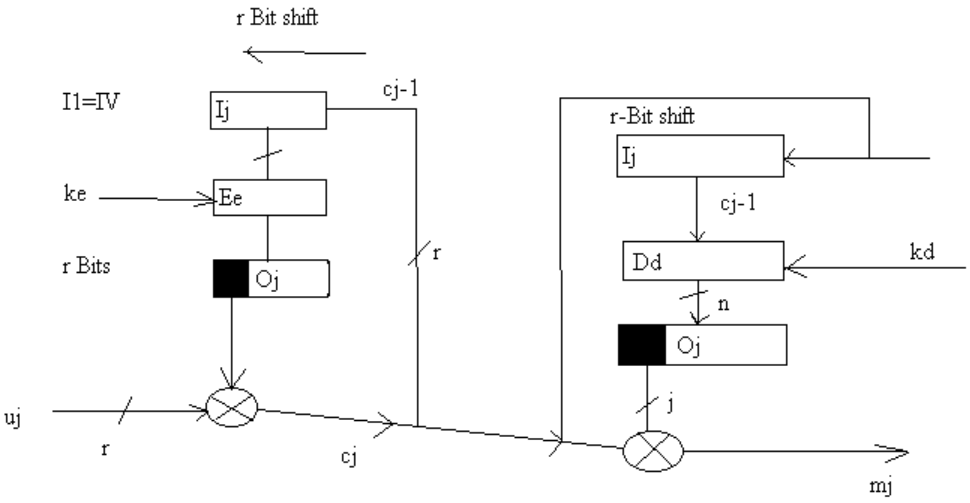
Előnye az optimális sebességű kódolás. Hibája, hogy ha szinkronizációs hiba lép fel, a dekódolt üzenet értelmezhetetlen. Meg kell ismételni.

CBC (cipher block changing mode)



Itt egy n hosszúságú bitsorozatot IV-t hurkolunk. Később a szinkronizációhoz a kódolt üzenet legutóbbi blokkját használjuk. A szinkronizációs hibát így ki lehet küszöbölni.

CFB (cipher feedback mode)



ECB-nél az üzenetet 64 bites szavakra bontjuk és kódoljuk a K kulcs felhasználásával.

CBC-nél az üzenetet 64 bites $x_1 x_2 \dots x_n$ szavakra bontjuk. Kiválasztunk egy $y_0 \in \{0,1\}^{64}$-et és $y_i = e_k(y_{i-1}, x_i)\, i = 1 \dots n$

Az utóbbi módszert jól lehet alkalmazni üzenetek hitelesítésénél. A hitelesítés azt jelenti, hogy a vevő bizonyos akar lenni, hogy az üzenetet nem változtatták meg és attól kapta, akitől várta.

Ekkor, ha Alice üzenetet akar küldeni Bobnak a következő képpen járhat el:

1. Kódolja $x_1 \dots x_n$ -et $y_0 = O^{64}$-el egy titkos DES kulcsot a CBC módszert használva, így megkapja az $y_1 \dots y_n \in \{0,1\}^{64}$ sorozatot.
2. Elküldi $x_1 \dots x_n$, y_n-et Bobnak.
3. Bob rekonstruálja $y_1 \dots y_n$-et $y_0, x_1 \dots x_n$ és K ismeretében és ellenőrzi, hogy a szövegben nem történt-e változás.

A hitelesítést össze lehet kapcsolni a titkosítással is. Ekkor a protokoll a következő:

1. Alice kódolja $x_1 \dots x_n$-et $y_0 = O^{64}$-el és k1 kulccsal a DES CBC módusát felhasználva. $x_{n+1} := y_n$.
2. Alice kódolja $x_1 \dots x_n x_{n+1}$-et k2 kulccsal az EBC módust használva és elküldi $y_1 \dots y_{n+1}$-et Bobnak.
3. Bob dekódolja $y_1 \dots y_{n+1}$-et k2 segítségével és megkapja az $\overline{x}_1 \dots \overline{x}_n, \overline{x}_{n+1}$ üzenetet.
4. Bob kiszámítja $\overline{y}_n$-t k1, $\overline{x}_1 \dots \overline{x}_n$ segítségével a CBC módszerrel. Ha $\overline{x}_{n+1} = \overline{y}_n$ akkor az üzenet hiteles.

4 Aszimmetrikus kriptorendszerek

Az eddigiekben olyan kriptorendszerekkel ismerkedtünk meg, amikor a vevő és kódoló ismeri egymás kulcsát, vagy a két kulcs között egyszerű kapcsolat áll fenn. Az aszimmetrikus kriptorendszerek alaptételét Diffie és Hellmann egy 1976-os dolgozatában vetette fel. Ennek lényege az, hogy Bob nyilvánosságra hoz egy kulcsot, mellyel kódolva neki üzenetet lehet küldeni. Alice ezt használva kódolja az üzenetét, amelyet Bob dekódolni tud. Ugyanakkor a lehallgató Eve csak az üzenetet és a kódoló kulcsot ismerve nem tudja dekódolni azt. Ez így elég homályos és paradox modellnek tűnik, hiszen Bob üzenete egy véges hosszúságú, mondjuk n hosszúságú bitsorozat. Éva például a következőképpen járhat el: kipróbálja az összes n hosszúságú bitsorozatra alkalmazva a kódoló eljárást és megnézi, hogy mikor kapja meg az Alice által elküldött bitsorozatot. Mivel összesen 2_n sok különböző n hosszú bitsorozat van, Eva előbb-utóbb megfejti a küldeményt. Ez nem egy absztrakt támadási lehetőség! Ha n nem nagy sikeresen alkalmazható. Ilyen szituációval találkozhatunk jelszóval védett rendszereknél, amikor a szótáros támadás hatásosan működhet.

Térjünk azonban vissza a Diffie és Hellmann által javasolt modellhez. Jelölje Z_n a modulo maradékosztályok halmazát $Z_n = \{0, 1, 2 \dots n-1\}$. Az $f : Z_n \rightarrow$

Z_n leképzést egyirányú függvénynek nevezzük, ha $\forall x \in Z_n$ -re f(x) log n -ben polinomiális időben kiszámítható, injektív függvény, de $f^{-1}(y)$ majdnem minden $y \in Z_n$-re csak n-nel arányos időben számolható ki. Kevésbé szigorúan, de érthetőbben fogalmazva, $f^{-1}(y)$ kiszámítására általában az összes rendelkezésünkre álló számítástechnikai kapacitás sem lenne képes, még akkor sem, ha azt néhány évtizedig erre a feladatra használjuk. A "triple" DES ismereteink szerint például megfelel ennek a követelménynek. De a DES dekódolásához szükségünk van a kódoló kulcs ismeretére.

Diffie és Hellmann azonban tovább mennek és bevezetik az egyirányú-csapóajtó függvény (one-way trapdoor) fogalmát. Az $f : Z_n \to Z_n$ függvényt egyirányú csapóajtó függvénynek nevezzük, ha $\exists d \in Z_n$ ismeretében $f^{-1}(y)$ már majdnem minden $y \in Z_n$-re $log\, n$-ben polinomiális időben.

Itt jegyzem meg, hogy nem tudjuk, hogy létezik-e egyirányú függvény és még kevésbé, hogy létezik-e egyirányú csapóajtó függvény. A következőkben néhány olyan eljárást ismertetünk, amelyek a gyakorlatban jól beváltak.

5 RSA

Az RSA kriptorendszert R. L. Rivest, A. Shamir és L. Adleman javasolta 1978-ban.

A kriptorendszer leírása:
Legyenek p és q prímszámok, $n = p * q$. Akkor $\varphi(n) = (p-1)(q-1)$.
$P, C \in Z_n$
$K = \{(n, p, q, e, d)\}$ ahol p, q prímek, $n = pq,\ 0 < e,\ d < \varphi(n),\ (e, \varphi(n)) = 1,$
$ed \equiv 1 (mod\, \varphi(n))$
$e_k(x) = x^e\ mod\, n$
$d_k(x) = x^d\ mod\, n$

Az n és e számok nyilvánosak, p, q és d titkosak.

Ahhoz, hogy belássuk, hogy d_k aze_k függvény inverze, szükségünk van a következő tételre:

Tétel: (Euler-Fermat) Ha $(x, n) = 1$, akkor $x^{p(n)} \equiv 1 (mod\, n)$.

Erre a tételre még hivatkozni fogunk a paraméterek meghatározása során.

Tétel: Az RSA kriptorendszer korrekt. Azaz, bármely olyan $0 < x < n$-re, amelyre $p < x$ és $q < x$ teljesül, hogy $d_k(e_k(x) = x$.

Bizonyítás:

$d_k(e_k(x) = (x^e\ mod\, n)^d\ mod\, n = x^{ed}\ mod\, n$, azaz $0 \leq d_k(e_k(x)) < n$.

Az $ed \equiv 1 (mod\, \varphi(n))$ feltételből következik, hogy $u \in Z$, hogy $ed + n\varphi(n) = 1$, azaz $ed = 1 + \varphi(n)(-n)$. Tehát $d_k(e_k(x)) = x^{ed} = x^{l+p(nx-n)} = x(x^{p(n)})^{-n} \equiv$

$x(mod\, n)$, azaz $n|d_k(e_k(x)) - x$. Mivel $0 < d_k(e_k(x)) < n$ és $0 < x < n$, így $d_k(e_k(x)) = x$.

$e_k(x)$ és $d_k(x)$ $O(log^3 n)$ időben kiszámítható. Először is azt kell tisztáznunk, hogy az RSA kódoló és dekódoló függvénye könnyen, azaz $O(log^3 n)$ időben kiszámítható. Ez a következő "intelligens hatványozó" algoritmussal bizonyítható:

Írjuk fel l -et 2-es számrendszerben: $l = e_l + e_{l-1} \cdot 2 + \ldots + e_0 \cdot 2^l = \sum_{i=0}^{l} e_{l-i} \cdot 2^i$, ahol $e_i \in 0, 1, e_0 = 1$.

Alkalmazzuk a következő algoritmust:

1. $y \leftarrow x, j = 1$
2. while $j < l$ do {
   $y \leftarrow y^2\, mod\, n$
   if $e_j = 1$ then $y \leftarrow y \cdot x\, mod\, n$
   $j \leftarrow j + 1$
   }
3. output y

Tétel: Az előző algoritmus korrekt és futási ideje $O(log^3 n)$.

Ha $l = 0$, akkor $e = 1$, a while ciklusban nem csinálunk semmit, így a kimenet $x = x^1$.

Tegyük fel, hogy az állítás igaz l-re és legyen egy $l + 1$ hosszúságú szám. Ekkor $e = e_{l+1} + e_l \cdot 2 + \ldots + e_0 \cdot 2^{l+1} = e_{l+1} + \sum_{i=0}^{l} e_{l-i} \cdot 2^i = e_{l+1} + 2f$, ahol f hossza l, hiszen $e_0 \neq 0$. Azaz $x^e = e^{e_(l+1)+2f} = e^{e_{l+1}} \cdot \left(e^f\right)^2$.

A futásidőre vonatkozó becslés abból következik, hogy a while ciklust log n-szer hajtjuk végre, és minden alkalommal legfeljebb 2 szorzást és 2 maradékos osztást hajtunk végre legfeljebb log n hosszúságú számokon. A hatványozás előbb ismertetett algoritmusa alapvető fontosságú minden mai kriptorendszernél.

5.1 Az RSA paramétereinek a meghatározása

Az RSA-ban négy paraméter: p, q, e és d fordul elő. Ha p és q-t ismerjük, akkor $n = p \cdot q$ és $\varphi(n) = (p-1)(q-1)$ könnyen meghatározható. Már rámutattunk, hogy $\frac{\varphi(m)}{m} > \frac{c}{log\, log\, m}$, ahol c egy pozitív konstans, így e véletlen választással gyorsan meghatározható. Az $(e, \varphi(m)) = 1$ feltétel és az $e \equiv 1(mod\, \varphi(n))$ kongruenciának eleget tevő d (hacsak $(e, \varphi(m)) = 1$) a bővített euklideszi algoritmussal ugyancsak könnyen számolható.

Az a kérdés maradt csak, hogy hogyan lehet a p és q prímszámokat meghatározni. A hagyományos módszer erre az, hogy választunk egy p számot. Ha ez összetett, akkor van neki egy $p_1 > p$ osztója. Legyen $p_2 = p|p1$, akkor $p_1 \leq p_2$ Tegyük

fel, hogy $p_1 \leq p_2$. Akkor $p_i = p_1 p_2 \geq p_1^2$, amelyből $p_1 \leq \sqrt{p}$ következik. Végigpróbálva a $2 \leq p_i \leq \sqrt{p}$ prímszámokat, ha egyik sem osztja p-t, akkor p prímszám.

Ez az eljárás nagyon jól működik, ha p kis szám, pl. könnyen programozható $p \leq 10^{10}$-re.

Mi történik azonban, ha p egy nagy szám pl. $p \geq 10^{40}$? Ehhez azt kell tudnunk, hogy $\sqrt{p}$-ig hány prímszám van. C. F. Gaufy sejtette a XVIII. sz. végén és C. Hudamand és de la Vallée Poussin bizonyította 100 évvel később, hogy ha $\pi(x)$-el jelöljük a prímszámok számát x-ig, akkor $\pi(x) \approx \frac{x}{log\, x}$ Ha tehát $x \approx 10^{40}$ akkor $\sqrt{x} \approx 10^{20}$ és így $\sqrt{x}$-ig kb. $\pi(10^{20}) \approx \frac{10^{20}}{log\, 10^{20}} = \frac{10^{20}}{20 \cdot log\, 10} \approx \frac{1}{2} \cdot 10^{19}$ prímszám van. Ahhoz tehát, hogy egy 40 jegyű számról megállapítsuk, hogy prímszám-e több mint $5 * 10^{18}$ osztást kell végezni.

Egy nagyon gyors számítógép ma 10^8 osztást tud elvégezni másodpercenként, így az előbbi feladat elvégzésére $5 \cdot 10^{18}$ másodperc időre, azaz 1585 évre lenne szüksége, ami természetesen megengedhetetlen.

A klasszikus "naív" eljárással az a baj, hogy több legyet akar ütni egy csapásra. Nem csak azt az információt szolgáltatja ugyanis, hogy p nem prím, hanem a p egy osztóját is. A fejezet elején idézett Euler-Fermat tételt megfelelően szemlélve azonban esetleg csak arra kapunk információt, hogy a vizsgált számunk biztosan nem prím!

Legyen ugyanis p prím, akkor az E-F tétel azt mondja, hogy bármely $0 < x < p$ számra $x^{p-1} \equiv 1 (mod\, p)$.

Fermat teszt: Ha tehát valamely $0 < x < p$ számra x^{p-1} nem kongruens 1(mod p), akkor p összetett.

Nézzük ennek egy alkalmazását. Legyen $p = 65$ és $x = 2$. Akkor $p - 1 = 64 - 2^6$. Számítsuk ki 2^{2^6}-t, amelynek folyamatát az alábbi táblázat mutatja:

i	0	1	2	3	4	5	6
$2^2\, mod\, 65$	2	4	16	-4	16	-4	16

Mivel $2^{2^6} = 2^{64} \equiv 16\, nem \equiv 1 (mod\, 65)$, ezért 65 nem prímszám.

5.2 Prímszámok és a Miller-Robin teszt

A Fermat teszt még nem a történet vége, mert előfordulhat, hogy valamely összetett n számra és $0 < a < n$ számra a és n ugyan relatív prímek, de $a^{n-1} \equiv 1(mod\, n$. Tekintsük például a klasszikus $n = 341 = 11 \cdot 31, a = 2$ példát. Ekkor nyílván $(341, 2) = 1$. Másrészt $2^{340} = (2^{10})^{31} \equiv 1 (mod\, 11)$, hiszen 11 prímszám és $2^{10} \equiv 1 (mod\, 11)$ az Euler-Fermat tétel miatt. Hasonlóképpen $2^{30} = (2^5)^{68} = 32^{68} \equiv 1 (mod\, 31)$. Tehát $11|(2^{340} - 1)$ és $31|(2^340 - 1)$, aminek következtében

$341 = 11 \cdot 31 | (2^{340} - 1)$, azaz $2^{340} \equiv 1 (mod\, n)$. Tehát a Fermat teszt nem, csak az n osztói fedik fel, hogy n összetett. A legkisebb ilyen szám $n = 561 = 3 \cdot 11 \cdot 17$. Teljesül ugyanis, hogy $560 = 2 \cdot 280 = 10 \cdot 56 = 16 \cdot 35$. Így ha $(a, 561) = 1$, akkor persze $(a, 3) = (a, 11) = (a, 17)$ és így

$(a^2)^{280} \equiv 1 (mod\, 3)$
$(a^{10})^{56} \equiv 1 (mod\, 11)$
$(a^{16})^{35} \equiv 1 (mod\, 17)$

tehát $a^5 60 \equiv 1 (mod\, 561)$ bármely $0 < a < n, (a, n) =$ 1-re. R. Alford, A. Granville és C. Pomerance, 1994-ben megmutatták, hogy végtelen sok Carmichael szám létezik, azaz olyan n, amelyik összetett és amelyre teljesül, hogy bármely $0 < a < n, (a, n) =$ 1-re $a^{n-1} \equiv 1 (mod\, n)$

Jó hír azonban, hogy a Fermat teszt tovább fejleszthető. 1976-ban G. Miller javasolt egy erős prímtesztnek nevezett módszert, amelyről M. Robin 1980-ban megmutatta, hogy valószínűségi teszttté finomítható. A teszt a következő tételeken alapul:

Tétel: Legyen n páratlan prímszám, $n - 1 = 2^5 \cdot t$, ahol t páratlan. Legyen $0 < a < n$ olyan, $hogy (a, n) = 1$ akkor az

$$a^{\frac{n-1}{2^n}} - 1,\ a^{\frac{n-1}{2^n}} + 1,\ a^{\frac{n-1}{2^{n-1}}} + 1 \ldots a^{\frac{n-1}{4}} + 1,\ a^{\frac{n-1}{2}} + 1$$

számok valamelyike osztható n-nel.

Bizonyítás: Az Euler-Fermat tétel szerint $n | a^{n-1} - 1$. Az utóbbi számot felírhatjuk

$a^{n-1} - 1 = (a^{\frac{n-1}{2}} + 1)(a^{\frac{n-1}{2}} - 1) = (a^{\frac{n-1}{2}} + 1)(a^{\frac{n-1}{4}} + 1)(a^{\frac{n-1}{4}} - 1) = \ldots$
$= (a^{\frac{n-1}{2}} + 1)(a^{\frac{n-1}{4}} + 1) \ldots (a^{\frac{n-1}{2^n}} + 1)(a^{\frac{n-1}{2^n}} - 1)$

Alakban. Egy prímszám azonban, ha oszt egy szorzatot, akkor osztja valamely tényezőjét. Ebből közvetlenül következik a tétel állítása. A tételből közvetlenül következik az alábbi prímteszt:

Következmény: Legyen n egy páratlan szám és $n - 1 = 2^3 t$, ahol t páratlan. Legyen továbbá $0 < a < n$ olyan, hogy vagy $(a, n) > 1$ vagy pedig az 5.2 számok egyike sem osztható n-nel, akkor n összetett. Tekintsük példaként az $n = 561$ számot. Ekkor $n - 1 = 560 = 2^4 \cdot 35$. Válasszuk az $a = 2$ számot. Ekkor azt kell vizsgálnunk, hogy $2^{35} - 1$, $2^{35} + 1$, $2^{70} + 1$, $2^{140} + 1$, $2^{280} + 1$ számok valamelyike osztható-e 561-gyel. Ezeknek a számoknak a maradéka 561-el osztva 262, 264, 167, 68, 2 tehát nem élte túl a Miller-Robin tesztet. Ez nem véletlen, M. Robin ugyanis bebizonyította, hogy

Tétel: Legyen n egy páratlan összetett szám és $n - 1 = 2^3 t$, ahol t páratlan. Jelölje $MR(n)$ azon $0 < a < n$ számok halmazát, amelyekre nézve n túléli a Miller-Robin tesztet. Azaz legyen $MR = \{a : 0 < a < n,\ (a, n) = 1, a^{\frac{n-1}{2^n}}\ nem \equiv 1 (mod\, n)$ és a $a^{\frac{n-1}{2^n}}$, $a^{\frac{n-1}{2^n}}$, $\ldots$ $a^{\frac{n-1}{4}}$, $a^{\frac{n-1}{2}}\ nem \equiv -1 (mod\, n)\}$

Akkor: $\frac{MR}{n} \leq \frac{1}{4}$.

Ezt a tételt nem bizonyítjuk, hanem inkább annak alkalmazásával foglalkozunk. Tegyük fel, hogy $MR(n)$ elemi egyenletesen oszlanak el a $0 < 163 < n$ intervallumban. k-szor végrehajtva a Miller-Robin tesztet arra következtethetünk, hogy annak a valószínűsége, hogy egyszer sem választottunk ki olyan számot, amelyik n összetett tulajdonságát felfedte volna $\frac{1}{4^k}$. Mivel a $[0, n)$ intervallumban n szám van, ezért, ha $\frac{n}{4^k} < 1$, azaz ha $k > log\, n/log\, 4$, akkor n nagy valószínűséggel prímszám.

Ha tehát a Miller-Robin tesztet elég sokszor végrehajtjuk és a tesztelendő n szám minden esetben "valószínűleg nem összetettnek" bizonyul, akkor nagyon valószínű, hogy n prímszám. Például $25 \cdot 10^9$-ig egyetlen olyan szám van, az $n = 3215031751 = 151751 \cdot 28351$, amelyik az $a = 2, 3, 5$ és 7 mindegyikére túlélte a Miller-Robin tesztet.

Megjegyzem, hogy létezik a Miller-Robin tesztnél is hatékonyabb valószínűségi prímteszt. Ezt J. Grantham publikálta 1998-ban. H. Cohen és H. W. Lenstra Jr. (1984), Atkin-Morain (1993) determinisztikus prímteszteket találtak, amelyeknek implementációival 2500 decimális jegyű számokról is el lehet dönteni, hogy prímszám-e.

5.3 Támadási lehetőségek az RSA ellen

Az RSA ellen a ma ismert egyetlen támadási lehetőség az n modulus faktorizációja. Ismerünk olyan megfontolásokat is, hogy ha az e vagy a d számok túl kicsik, akkor az RSA részben vagy teljesen kompromittálható.

Itt most csak a faktorizáció kérdésével foglalkozunk. A következő módszereket ismerjük ezen feladat megoldására.

1. Osztás kis prímszámokkal. Ekkor azt vizsgáljuk meg, hogy n osztható-e valamely $r < B$ prímszámmal, ahol B egy tapasztalatok által meghatározott konstans. Így ki lehet szűrni a kis prímosztókat.
2. Brillhant-Morisson módszer lánctörteket használ.
3. Shank SQUFOF módszere is lánctörteket használ.
4. Fermat faktorizáció akkor használható, ha $n = pq$ és $p - q$ kicsi.
5. Kvadratikus szita, PC. Pomerance
6. Elliptikus görbéket használó faktorizáció. H. W. Lenstra
7. Számtest szita, Polland

Nagy számok 70-80 decimális jegyétől kezdve csak az 5., 6., és 7., módszer használható. Mindegyik várható futási sebessége szubexponenciális.

5.3.1 Fermat faktorizáció

Legyen $n = pq = \left(\frac{p+q}{2}\right) - \left(\frac{p-q}{2}\right)$. Ahhoz tehát, hogy n-et faktorizáljuk, elegendő az

$$n = x^2 - y^2 = (x-y)(x+y)$$

egyenletet megoldani úgy, hogy $x - y \neq 1$

Algoritmus: Fermat farktorizáció

Input: n páratlan összetett szám
Output: d az n legnagyobb olyan osztója, amelyre $d \leq \sqrt{n}$

1. $x \leftarrow 2\left[\sqrt{2}\right]^2 + 1$
   $y \leftarrow 1$
   $r \leftarrow \left[\sqrt{n}\right]^2 - n$
2. while $r > 0$ do {
   $r \leftarrow r - y$
   $y \leftarrow y + 2$
   }
3. if $r = 0$ then { output $\frac{x-y}{2}$, return }
4. $r \leftarrow r + y$
   $x \leftarrow x + 2$
   goto 2

Az 5.3.1 egyenlet megoldása persze túl sokat követel. Az n faktorizációjához igazából elegendő olyan x és y egészeket találni, hogy $x nem \equiv 1 (mod\, n)$ és $x^2 \equiv y^2 (mod\, n)$. Ekkor ugyanis $n|(x^2 - y^2) \equiv (x-y)(x+y)$ és mivel $n > (x+y), (x-y)$, így $lnko(n, x-y)$ vagy $lnko(n, x+y)$ az n egy nem triviális osztója. Legyen például $n = 4630$. Akkor $x = 77$-et választva $77^2 = x^2 \equiv 36^2 (mod\, 4633)$. Mivel $77 nem \equiv 36(4633)$ így $(4633, 41) = 41$ és $(4633, 113) = 113$ és $4633 = 41 \cdot 113$. A 5.3.1-nak eleget tevő x, y számokat nehéz találni. Itt sajnos nem segít a véletlen választás, mert 5.3.1-nak összesen 4 nem triviális megoldása van, ha n két prímszám szorzata. Ezen segít a következő faktorbázist alkalmazó módszer.

Legyen $B = \{p_0, p_1, p_2 \ldots p_t\}$ ahol $p_0 = -1$, és p_i az i-edik prímszám. Tehát $p_1 = 2, p_2 = 3 \ldots t$-t tapasztalat és a hardverfeltételek határozzák meg. Legyenek $x_1, x_2 \ldots$ egész számok egy sorozata.

Határozzuk meg $x_i^2\, mod\, n$-t, ahol most mod a legkisebb abszolút maradékot jelenti. Azaz, ha n páratlan, akkor $\frac{n-1}{2} \leq x\, mod\, \leq \frac{n-1}{2}$. ??

Faktorizáljuk $x_i^2\, mod\, n$-t a B faktorbázis felett. Azaz legyen
$x_i^2\, mod\, n = p_0^{a_n} p_1^{a_n} \ldots p_t^{a_n}, 0 \leq \alpha_i$. Ha ez nem lehetséges, akkor x_i-t dobjuk el és vegyük helyette a következő sorozatelemeket.

Képezzük az $\widehat{\alpha} = \alpha_{i0}\, mod\, 2, \alpha_{i1}\, mod\, 2 \ldots \alpha_{it}\, mod\, 2$ vektorokat.

Ha találunk olyan $\widehat{\alpha}_{i1} \dots \widehat{\alpha}_{i5}$ vektorokat, amelyek a kételemű F_2 test felett lineárisan függőek, azaz $\lambda_{i1}\widehat{\alpha}_{i1} + \dots \lambda_{i5}\widehat{\alpha}_{i5} = 0$ $\lambda_{i1} \dots \lambda_{i5} = 1$ mellett, akkor ez azt jelenti, hogy $\sum_{j=1}^{5} \lambda_{ij}\widehat{\alpha}_{jn}$ páros $n = 0 \dots t$-re, azaz $\sum_{j=1}^{5} \lambda_{ij}\widehat{\alpha}_{jn} = 2\beta_n$ $n = 0 \dots t$. Összeszorozva a megfelelő relációkat kapjuk, hogy

$$\prod_{j=1}^{5} x_{ij}^{\lambda_{ij}} \equiv \prod_{u=0}^{t} \prod_{j=1}^{5} p_u^{a_{ij}a_{ij}u} = \prod_{u=0}^{t} p_u^{2\beta_u} = \left(\prod_{u=0}^{t} p_u^{\beta_u} \right)^2$$

$x = \prod_{j=1}^{5} x_{ij}^{\lambda_{ij}}$ és $\prod_{u=0}^{t} p_u^{\beta_u}$ választással így $x \equiv y (mod\, m)$.

Ha még $x \, nem \equiv y$, akkor sikerült n egy nem triviális osztóját megtalálni.

A kvadratikus és a számelméleti viták a fenti alapötletet fejlesztik tovább oly módon, hogy a x_i alappontok szisztematikus megválasztását garantálják. A faktorizációs módszerek csúcsteljesítményét jelenti az, hogy 1999-ben egy kutatócsoportnak sikerült az RSA 155-öt faktorizálni. Ez egy 155 decimális jegyű összetett szám, amelyik két 77 és 78 jegyű prímszám szorzata. A faktorizációs módszerek hatékonyságának növekedésének következtében ma már ott tartunk, hogy az RSA-t a szakértők csak akkor tartják biztonságosnak, ha az n modulus 1024 bites és két darab közel 512 bites prímszám szorzata. Tekintettel arra, hogy ilyen nagy számokkal már nehezen és lassan lehet számolni, ezért az RSA-ba vetett bizalom csökkent. Megjegyezzük azt is, hogy az RSA sebessége kb. 1000-szer kisebb, mint a DES. Ez is befolyásolja az RSA alkalmazási lehetőségét.

6 Diszkrét logaritmus probléma

Legyen G egy véges ciklikus csoport és $G =< g >$ valamint $ord\, G = m$. Legyen $0 \leq x < m$ és $y = g^x$. A DLP probléma azt jelenti, hogy g, y, és m ismeretében határozzuk meg x-et. A legegyszerűbb estben $G = Z_p^*$, ahol p prímszám.

6.1 Az El Gamal kriptorendszer

Z_p^*-ban:

$P := Z_p^*$ üzenetek halmaza, $g \in Z_p^*$ primitív gyök Z_p^*-ban.
$C := Z_p^* \times Z_p^*$ kódolt üzenetek halmaza

$K := \{(p, q, a, \beta) := \beta = g^a (mod\, p)\}$ A p, g, β, Z_p^* nyilvános, a titkos

Válasszunk egy titkos véletlen $k \in Z_{p-1}$ kulcsot és legyen $e_K(x, k) = (y_1, y_2)$, ahol $y_1 \equiv \alpha^k (mod\, p)$, $y_2 \equiv x\beta^k (mod\, p)$

A dekódoló leképzés: $d_K(y_1, y_2) = y_2 (y_1^a)^{-1} (mod\, p)$.

6.2 Hogyan válasszuk meg a paramétereket

p-ről már beszéltünk. Z_p^* rendje $p-1$.

Állítás: Legyen g prim. gyök z_p^*-ban. $h = g^k$ szintén prim. gyök $\Leftrightarrow (k, p-1) = 1$.

Bizonyitás: Ha $< h >= Z_p^* \Rightarrow \exists x$, hogy $h^x = g$, azaz $g^x k = g \Rightarrow xk \equiv 1 (mod\, p-1) \Rightarrow (k, p-1) = 1$.

Tétel: Z_p^*-ban $\varphi(p-1)$ primitív gyök van.

$$p - 1 \leq \varphi(p-1) log(p-1) \Rightarrow \frac{\varphi(p-1)}{p-1} \geq \frac{1}{log(p-1)}$$

Tehát $log(p-1)$ lépés után találunk prim. gyököt.

Legyenek p, g, y adottak, p prím. Azt az $0 \leq x < p$ számot, amelyre $g^x \equiv y (mod\, p)$ az y g alapú indexének nevezzük $mod\, p$. Jelölés: $ind_g y$.

6.3 D. Shank "baby step - giant step" algoritmusa

1. Próbálgatás: Kiszámítjuk g^x-et $x = 0, 1 \ldots p-1$-re. Vegyük észre, hogy mivel g prim. gyök, így $g^{\frac{p-1}{2}} \equiv -1 (mod\, p)$ és így $g^{\frac{p-1}{2}+x} \equiv -g^x (mod\, p)$ teljesül $\forall x : 0 \leq x < p-1$-re. Elegendő tehát $\frac{p-1}{2}$ hatványt kiszámítani. De az is sok.

2. Legyen adva egy m szám. Számítsuk ki és tároljuk $yg^{-j}\ mod\, p$-t $0 \leq j < m$-re. Most számítsuk ki g^{km}-et $k = 0, 1 \ldots \left[\frac{p}{m}\right]$-ra és hasonlítsuk össze az előbbi értékkel.

Ha $x \equiv g^x\ mod\, p$ és $x = qm + j$ alakban $0 \leq j < m$, akkor $y \equiv g^x = g^{mq} g^j$, azaz $g^{mq} \equiv yg^{-j} (mod\, p)$. Amint g^{mq}-t kiszámítottuk megtaláljuk yg^{-j}-t és így $x = qm + j$-t.

Hány szorzást kell végezni? $m + \left[\frac{p}{q}\right] \approx m + \frac{p}{m}$ számtani és mértani közepek miatt $m + \frac{p}{m} \geq 2\sqrt{m\frac{p}{m}} \geq 2\sqrt{p}$ ami egyenlőség pontosan akkor ha $m = \frac{p}{m}$, azaz $m = \sqrt{p}$.

6.4 Pohling-Hellman algoritmus

Tegyük fel, hogy $p - 0 = \prod_{i=1}^{y} p_i^{a\,i}$. Ha $\forall i$-re ki tudjuk számítani azt az x_i-t amelyre $a \equiv x_i (mod\, p_i^{a\,i})$, akkor a kínai maradéktétel miatt α is ismert. Tegyük fel hogy $q^c | p-1$, de $q^{c+1} p-1$. $0 \leq a \leq q^c$ és $x \equiv a (mod\, q^c$. Legyen $x = \sum_{x=0}^{c-1} a_i q^i$. Másrészt $\exists 0 < s, a = x + q^c s$.

Állítjuk, hogy:

$$\beta^{\frac{p-1}{q}} \equiv g^{frac(p-1)a_0 q}(mod\, p)$$

Valóban

$$\beta^{\frac{p-1}{q}} = g^{\frac{(p-1)a_0}{q}} = g^{\frac{(x+q^c s)(p-1)}{q}} \equiv g^{\frac{(p-1)a_0}{q}}(mod\, p) \Leftrightarrow$$

$$\frac{(x+q^c s)(p-1)a_0}{q} - \frac{(p-1)a_0}{q} = \frac{(p-1)}{q}(x+q^c s - a_0) =$$

$$\frac{p-1}{q}(a_1 q + \ldots + a_{c-1}q^{c-1} + sq^c) \equiv 0(mod\, p-1)$$

Először kiszámítjuk $\beta^{\frac{p-1}{q}}$-t. Ha $\beta^{\frac{p-1}{q}} = 1$, akkor $a_0 = 0$. Ha nem, akkor kiszámítjuk $\gamma = g^{\frac{p-1}{q}}$-t, $\gamma_2, \gamma_3 \ldots$ ameddig $\beta^{\frac{p-1}{q}}$-t nem kapjuk és így a_0-t. Ha $c = 1$, akkor készen vagyunk. Ellenkező esetben $a_1, a_2 \ldots$-t is meg kell határozni. Legyen $\beta_1 = \beta g^{-a_0}$. Ekkor $\beta_1^{\frac{p-1}{q^c}} \equiv g^{\frac{(p-1)a_1}{q}}(mod\, p)$ és elegendő megkeresni azt az a_1-et, amelyre $\gamma^{a_1} \equiv \beta_1^{\frac{p-1}{q^c}}(mod\, p)$ stb. Ha tehát $p-1$-nek sok kis prímosztója van, akkor az index könnyen kiszámítható!

Specializáljuk most G-t, mégpedig legyen $G = Z_p^*$, ami egy $p-1$-ed rendű ciklikus csoport.

Skans $\Rightarrow p \leq 10^{40}$

Pohling-Hellman $\Rightarrow p-1$-nek legyen legalább egy nagy prímosztója.

Paraméterválasztás:

Indexkalkulus: Z_p^*-ban rendelkezésünkre áll egy olyan speciális módszer is, amellyel $ind_p(x)$ szubexponális időben kiszámítható. Legyen $B = \{p_1, p_2 \ldots p_t\}$ az első t prímszám. Válasszunk egy $x_2, x_2 \ldots$ sorozatot és próbáljuk meg $a^{x_j} mod\, p$-t faktorizálni B felett, azaz

$$g^{x_j} = p_1^{a_{j1}} \ldots p_t^{a_{jt}}(mod\, p) \Leftrightarrow x_j \equiv \alpha_{j_1} ind_p(p_1) + \ldots + ind_p(p_t)\alpha_{j_t}(mod\, p-1)$$

Így kapunk egy egyenletrendszert $mod\, p-1$ és ha elég sok relációnk van akkor azt reméljük, hogy az ismeretlen $ind_p(p_1) + \ldots + ind_p(p_t)\alpha_{j_t}$ mennyiségeket meg tudjuk határozni. Amint ez sikeresen befejeződött válasszunk véletlenszerűen egy s számot és számítsuk ki a $\gamma \equiv \beta g^c(mod\, p)$ számot. Ha γ faktorizálható B felett, azaz ha $\gamma = p_1^{c_1} \ldots p_t^{c_t}$, akkor $c_1 ind_p(p_1) + \ldots + c_t ind_p(p_t) \equiv s + ind_p(\beta)\, mod\, (p-1)$ és mivel $ind_p(\beta)$ kivételével minden mennyiség ismert $ind_p(\beta)$-t ki tudjuk számítani.

7 Elliptikus görbék véges testek felett

Legyen $p > 3$ és $a, b \in Z$, úgy, hogy $4a^3 + 27b^2 \, nem \equiv 0 (mod\, p)$. Akkor az $E = \{(x, y) \in Z_2^2 : y^2 = x^3 + ax + b\} \cup \{\vartheta\}$ halmazt Z_p feletti elliptikus görbének nevezzük.

Pl: $p = 11, a = 1, b = 6$

x	$x^3 + x + 6 (mod\, 11)$	$y^2 = x^3 + x + 6 (mod\, 11)$ megoldható	y
0	6	-	-
1	8	-	-
2	5	+	4,7
3	3	+	5,6
4	8	-	-
5	4	+	2,9
6	8	-	-
7	4	+	2,9
8	9	+	3,8
9	7	-	-
10	4	+	2,9

Tehát $E = \{$(2,4), (2,7), (3,5), (3,6), (5,2), (5,9), (7,2), (7,9), (8,3), (8,8), (10,2), (10,9)$\} \cup \{\vartheta\}$, $|E| = 13$.

Az $E(Z_p)$ elliptikus görbén összeadást lehet definiálni a köv. képpen:

Legyenek $p = (x_1, y_1)$, $Q = (x_2, y_2) \in E(Z_p)$. Ha $x_1 = x_2$ és $y_1 = -y_2$ akkor $P + Q = \vartheta$.

Különben $P + Q = (x_3, y_3)$, ahol $x_3 = \lambda^2 - x_1 - x_2$, $y_3 = \lambda(x_1 - x_3) - y_1$,

$$\lambda = \left\{ \begin{array}{c} \frac{y_2 - y_1}{x_2 - x_3}, ha\, P \neq Q \\ \frac{3x_1^2 + a}{2y_1}, ha\, P = Q \end{array} \right\}$$

Végezetül P+ϑ=ϑ+P=P

Tétel: $E(Z_p)$ az előbb definiált + műveletre véges Abel csoportot alkot. $E(Z_p)$ vagy ciklikus vagy két ciklikus csoport direkt szorzata, $E(Z_p) \cong Z_{md} \oplus Z_m$ és $m|(p-1)$.

Pl: $P_2^+ + P_2^- = (2, 4) + (3, 6)$
$aa^{-1} \equiv 1 \, (mod\, m)$
$x_3 = 4 - 2 - 3 = -1 = 10 \, (mod\, 11)$
$y_3 = 2(2 + 1) - 4 = 2 \, (mod\, 11)$

Azaz $P_2^+ + P_2^- = p_1 0^+$)

Tétel: (Hasse) $||E(Z_p)| - p + 1| \leq 2\sqrt{p}$

Legyenek $a, \gamma, z, n \in Z$ és vizsgáljuk az $x^\gamma \equiv z \,(mod\, m)$ kongruenciát. Az RSA titkosítás kulcsa, ha n összetett és γ és z ismert, ekkor x-et csakis úgy tudjuk kiszámítani, hogy megismerjük n prímfelbontását.

A kérdést vizsgáljuk abból az aspektusból is, hogy x, z és n ismert, és keressük γ-t. Ezt nevezzük manapság diszkrét logaritmus problémának, azaz DLP-nek.

8 Elliptikus görbéken alapuló kriptorendszerek

R. Schoof algoritmusával $E(Z_p)$ elemszámát polinomiális $O((log\, p)^8)$ időben meg lehet határozni.

Tétel: $E(Z_p) \cong Z_{n_1} \otimes Z_{n_1}$, ahol $n_2 | n_1$ és $n_2 | p-1$. $E(Z_p)$ tehát általában majdnem egy ciklikus csoport.

8.1 Az El Gamal titkosítás algoritmusa

$g \in G$ úgy, hogy $< g >= G$.

$y^2 = x^3 + x + 6$

?? $x \in Z_{|G|}$ véletlen elem.

$e_K(x,k) = (g^k, xh^k) = (y_1 y_2)$

$E(Z_p) \cong Z_{n_1} \otimes Z_{n_1}$

Hogyan adaptáljuk ezt az ötletet $E(K)$-ra? G, a választása és így h kiszámítása világos. Azonban ??-t nehéz biztosítani, így kiegészítő ötletre van szükség.

8.2 Meneres-Vanstone titkosítás

Legyen E egy Z_p felett definiált elliptikus görbe ??.

Legyen $||E(Z_p)| - p + 1| \leq 2\sqrt{p}$, $C = Z_p^* \times Z_p^*$ és $K = \{(\alpha, a, \beta) :< a >= E, \beta = a\alpha\}$, ?? nyilvános, a titkos!

Kódolás:

(Alice) választ egy $k \in < \alpha >$ számot és $x = (x_1, x_2) \in Z_p^*$-ra meghatározza $e_K(x,k) = (y_0, y_1, y_2)$ hármast, ahol $y_0 = k\alpha$, $(c_1, c_2) = k\beta$, $y_1 = c_1 x_1 \,(mod\, p)$, $y_2 = c_2 x_2 \,(mod\, p)$

Bob megkapja a $y(y_0, y_2, y_2) \in Z_p^4$ üzenetet, ahol $y_0 = k\alpha$. Kiszámítja $\alpha y_0 = ka\alpha = k\beta = (c_1, c_2)$-t és így $d_k(y) = (y_1 c_1^{-1}\, mod\, p, y_2 c_2^{-1\, mod\, p}$-t ami $= (x_1, x_2)$.

Kulcscsere (Key exchange):

Alice kiválaszt egy véletlen számot és elküldi g^x-et Bob-nak. Hasonlóképpen B kiválaszt egy y véletlen számot és g^y-t elküldi A-nak.

A g^x-t felemeli az y hatványra és megkapja g^{xy}-t. B g^y-t felemeli az x hatványra és megkapja g^{xy}-t. g^{xy} a közös kulcs, amit pl DES-ben használunk.

9 Néhány érdekes kriptográfiai protokoll

A kódoló algoritmusok megismerése után szeretnénk néhány érdekes és fontos protokollt bemutatni.

Eddig beszéltünk már a

- Szimmetrikus kriptorendszert alkalmazó üzenetküldésről,
- Aszimmetrikus kriptorendszert alkalmazó üzenetküldésről,
- A kulcscseréről.

9.1 Azonosítás (identification)

Alice be akarja bizonyítani Bobnak, aki rendszerint egy számítógép, hogy ő Alice.

1. Alice üzen Bobnak, hogy azonosítani akarja magát.
2. Bob kéri Alice azonosítóját (user name)
3. Alice megadja azt
4. Bob kéri Alice jelszavát (password)
5. Alice megadja azt
6. Bob egy f egyirányú függvényt használva kiszámítja $f(A)$ értékét és megvizsgálja, hogy $f(A)$ szerepel-e az A user kódolt jelszavai között. Ha igen, elfogadja Alice személyazonosítóját, különben elutasítja őt.

A protokollnak két gyenge pontja van:

- Ha Alice távoli terminálról jelentkezik be, akkor a felhasználói neve és jelszava is nyilvános csatornán keresztül jut el Bobhoz. Így akár a passzív hallgató Eve is bejelentkezhet máskor Alice személyében. Ezt a protokollt használja a népszerű TELNET és FTP. Manapság a jelszavat Alice terminálja kódolja és már az kerül a hálózatra.
- Ha a jelszó nem elég biztonságos, akkor szótár használatával könnyen feltörhető. A dragon és a MI hallgatói jelszavak közül tavaly 25%-ot különösebb nehézség nélkül meg lehetett fejteni.

9.2 Digitális aláírás (Digital Signature Schemes)

Egy irat aláírása arra szolgál, hogy bárki, aki az iratot elolvassa vagy felhasználja tudja, hogy az kitől származott. Másrészt, ha az iratot valaki megváltoztatta, akkor az eredeti aláíró bizonyítani tudja, hogy az irat nem tőle származik. A digitális aláírásra a legegyszerűbb protokoll az RSA-ból vezethető le.

Alice alá akar írni egy I iratot és elküldeni Bobnak.

Tegyük fel, hogy Alice nyilvános kulcsai $e, m (0 < I < m)$ privát kulcsa pedig d.

1. Alice kiszámítja $J = I^d \, mod \, m$-et és elküldi (I, J)-t Bobnak.
2. Bob kiszámítja $J^e \, mod \, m$-et (e-t a nyilvános telefonkönyvből ismeri). Ha $J^e \, mod \, m = I$, akkor elfogadja Alice aláírását, különben nem fogadja el.

Ez a protokoll nem hatékony, ha I hosszú irat. Ezért inkább ilyenkor úgynevezett hash függvényeket használunk. Ezek egyirányú, de nem feltétlenül egyértelműen dekódolható fv-ek. $h : \{0,1\}^* \to \{0, 1 \ldots d-1\}, d = 2^{160}$.

A h hash fv-t használva az előbbi protokoll a következőképpen alakul:

1. Alice kiszámítja $h(I)$-t.
2. Alice kiszámítja $J = h(I)^d \, mod \, m$-et és (I, J)-t elküldi Bobnak.
3. Bob kiszámítja $I' = J^e \, mod \, m$-et.
4. Bob kiszámítja $h(I)$-t.
5. Ha $h(I) = I'$, akkor elfogadja az aláírást, különben elutasítja.

9.3 Digital Signature Algorithm (DSA)

1991 USA szabvány.

1. Kulcsgenerálás
   (a) Alice kiválaszt egy $2^{159} < q < 2^{160}$ prímszámot.
   (b) Alice kiválaszt egy p prímszámot, amelyik a köv. tulajdonságokkal rendelkezik: $2^{511} < p < 2^{512+64t}, t \in \{0, 1 \ldots 8\}, q | p-1$
   (c) Alice kiválaszt egy x primitívgyököt $mod \, p$ és kiszámítja $g = x^{(p-1)/q} \, mod \, p$.
   (d) Alice választ egy $a \in \{1, 2 \ldots q-1\}$ számot véletlenül és $k ? p$

   Alice nyilvános kulcsa (p, q, g, A), a privát kulcsa pedig a.

2. Aláírás
   Alice az u üzenetet akarja aláírni a $h : \{0,1\}^* \to \{0 \dots q-1\}$ hash fv. felhasználásával.

   (a) Alice kiválaszt egy véletlen $k \in \{0 \dots q-1\}$ számot és számolja $r = (g^k \bmod p) \bmod q$ és $s = k^{-1}(h(u) + ar) \bmod q$-t, ahol $kk^{-1} \equiv 1 (mod\, q)$.

   (b) Elküldi (r,s)-t Bobnak.

3. Ellenőrzés
   Bob megkapja (r,s)-t. Beszerzi (p,q,g,A)-t és h-t.

   (a) Bob ellenőrzi, hogy $1 \leq r \leq q-1$ és $0 \leq s \leq q$. Ha valamelyik nem igaz, az aláírás érvénytelen.

   (b) Ellenőrzi, hogy

$$r = \left(\left(g^{s^{-1} h(u) \bmod q} A^{r s^{-1} \bmod q} \right) \bmod p \right) \bmod q$$