Kriptográfia

Liptai Kálmán

Kriptográfia

Liptai Kálmán

Publication date 2011

Tartalom

Köszönetnyilvánítás

A kriptográfia egy végtelenül izgalmas és lenyűgöző fejezete az emberi gondolkodásnak. Kialakulását elősegítették a történelmi események és az emberi gondolkodás jeles képviselői. A háborúk és konfliktusok gyorsították a fejlődését, ami szomorú tény. Ugyanakkor kiváló tudósok bekapcsolódása a titkosítás világába megtermékenyítően hatott a területre, újabb és újabb diszciplínák segítették, illetve segítik most is, a biztonságos információ áramlást és tárolást a 21. században. Ez a nagyon összetett és sok forrásból táplálkozó tudományterület ma már tananyag a világ felsőfokú oktatásában, így Magyarországon is. Ahhoz, hogy a következő oldalakon olvasható, tanulható tananyag elkészüljön sok-sok segítséget kaptam hallgatóimtól és kollégáimtól. Valószínűleg az ő érdeklődésük és problémaérzékenységük nélkül nem is vállalkoztam volna erre a munkára.

Ezúton szeretnék köszönetet mondani Dr. Olajos Péternek és Dr. Tómács Tibornak a sok türelmes segítségért, amit mindenkor megkaptam Tőlük. Tanítványaim természetes kíváncsisága és az elkészült munkáik sokat lendítettek az elkészült munka színvonalán. Ezúton köszönöm Kiss Norbertnek és Mészáros Gábornak az AES demonstrációs programot, Győrfi Györgynek és Csintalan Ádámnak a Playfair programot, Csonka Istvánnak és Trombitás Viktornak a DES szemléltetését. Köszönettel tartozom Radácsy Tivadarnak, Vass Tamásnak, Mátéfi Beátának, Kovács Juditnak és sok szakdolgozómnak, hogy megmutatták nekem, hogy kimeríthetetlen érdekesség közelében vagyunk, amikor kriptográfiával foglalkozunk.

Külön köszönöm Dr. Egri-Nagy Attilának és Vrecenár Csabának, hogy a munka angol nyelven is elkészülhetett.

Végül köszönöm családomnak, hogy elviselte azt a lázas munkát, amit a jegyzet elkészítése igényelt.

1. fejezet - Történeti áttekintés

1. Bevezetés

A kriptográfia története legalább olyan bonyolult és szövevényes, mint az emberiség történelme. Valószínűleg nehéz ezt az állítást tételesen bizonyítani, de ha a teljesség igénye nélkül górcső alá vesszük az elmúlt évszázadokat, akkor szinte minden történelmi esemény egyben a kriptográfia pillanata is. Ahhoz, hogy pontosan értsük, hogy milyen utakat kellett bejárni a mai alkalmazásokig, tegyünk lépéseket az alapvető fogalmak megértéséhez.

A kriptográfia szó ógörög eredetű kifejezés, amely a „kryptos” azaz „rejtett”, illetve „gráph” azaz „írok” szavakból jött létre. Magyarul legegyszerűbben titkosírásnak fordíthatjuk, de mivel az írástól eléggé távol áll már a mai használat, szívesen használjuk a kriptográfia kifejezést.

Az alapproblémát egyszerűen úgy tudjuk megfogalmazni, hogyan tudunk üzenetet küldeni oly módon, hogy a fogadó fél könnyen fejtse a titkos levelet, ugyanakkor mindenki más részére a fejtés majdnem lehetetlen legyen vagy legalább is nagyon sok időbe teljen. A későbbiekben majd részletesen kitérünk arra, hogy mit is értünk nagyon sok időn, egyenlőre azonban megelégszünk a hétköznapi értelmezéssel.

A titkosítandó szöveg jelentése vagy jelentés nélkülisége számunkra lényegtelen, hisz legtöbbször már egy kódolt szöveget titkosítunk, ami feltehetőleg olvashatatlan betűk illetve számok halmaza csupán. Kódolás alatt a továbbiakban azt értjük, hogy a szövegben szereplő betűket (jeleket) számokkal helyettesítjük. Példa erre az a szokásosnak mondható kódolás, hogy az ABC betűit a sorszámukkal helyettesítjük.

A régi korokban a titkosított szöveg legtöbbször betűkből állt, jelenleg ezek a szövegek egyszerű bitsorozatok alakját veszik fel.

A következő fejezetekben jól elkülöníthető két rész, ami történetileg és szemléletét tekintve is igen különböző. Az egyik részt klasszikus kriptográfiának szokásos nevezni, amely története a 20. század közepéig tart. Ebben az irányban a találékonyság nagyon sokszor nélkülözi a matematikai módszereket, ötletek egymás utánja adja az alkalmazott módszert, amelyeket nagy titokban tartanak. Ezek a sokszor nagyon szellemes ötletek, egy-egy történelmi korhoz, történelmi eseményekhez kötődnek. Nagy többségük számítógép segítségével, a későbbiekben részletezett statisztikai módszerek segítségével megoldhatók.

A másik részt nyilvános kulcsú kriptográfiának nevezzük, utalva arra a tényre, hogy ezek a módszerek úgy működnek, hogy a titkosítási módszert és titkosítási kulcsokat nyilvánosságra hozzuk. Természetesen egy „titkos csapóajtót”, a fejtési kulcsokat megtartjuk, hogy a titkosság célját elérjük. Ezek a módszerek matematikai igazságokon nyugszanak és megfejtésükhöz elképesztő mennyiségű gépidő szükségeltetik.

A klasszikus és nyilvánoskulcsú kriptográfián kívül érdemes egy másik felosztást is megemlítenünk. Azoknál a módszereknél, ahol a küldőnek és a fogadónak is ismerni kell a titkosításhoz használt kulcsot, illetve lényegileg ugyanazzal a módszerrel titkosítunk és fejtünk, szimmetrikus kulcsú titkosításról beszélünk. Ilyen módszer az összes klasszikus módszer, de mai korunkban is találunk ilyeneket, például a későbbiekben megismert DES vagy AES is így működik.

Sokáig elképzelhetetlen volt, hogy legyen olyan módszer, amely jól működik a két fél közös titka nélkül, illetve úgy, hogy hiába ismerjük a titkosító kulcsot megfejteni nem tudjuk az üzenetet. Aztán a 20. században sikerült megoldani a talányt, az ilyen módszereket aszimetrikus kulcsú titkosításnak nevezzük. Ilyen például a később részletezett RSA módszer.

2. Alapvető fogalmak

A szövegtől és a titkosítás fajtájától függetlenül felírhatunk egy logikai sorrendet, amelyet többnyire követünk eljárásainknál. A természetes nyelvben megírt T-vel jelölt szöveget kódolnunk kell, majd titkosítani, ezek után a $C_T = E_k(T)$-vel jelölt titkosított szöveghez jutunk. Az így kapott szöveget, ha elég jó módszert sikerült választanunk nyugodtan továbbíthatjuk. A címzett a D_k-val jelölt fejtési kulcs ismeretében előállíthatja a $D_k(C_T)$ megfejtett szöveget, melyet dekódolva az eredeti szöveghez jutunk. Precízebben fogalmazva fogadjuk el a következő két definíciót kiindulási pontnak.

1.1. Definíció. Egy kódolási séma vagy kriptorendszer egy $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ötös a következő tulajdonságokkal:

1.

$\mathcal{P}$, $\mathcal{C}$ és $\mathcal{K}$ véges halmazok, $\mathcal{P}$ a nyílt szöveg tér, $\mathcal{C}$ a rejtett szöveg tér és $\mathcal{K}$ a kulcstér. $\mathcal{P}$ elemeit nyílt szövegnek, $\mathcal{C}$ elemeit rejtett szövegnek, $\mathcal{K}$ elemeit kulcsoknak nevezzük. Egy üzenet a nyílt szöveg szimbólumaiból álló szó.

2.

$\mathcal{E} = E_k | k \in \mathcal{K}$ azoknak az $E_k : \mathcal{P} \to \mathcal{C}$ függvényeknek a családja, amelyeket a rejtjelezéshez használunk.

$\mathcal{D} = D_k | k \in \mathcal{K}$ azoknak a $D_k : \mathcal{C} \to \mathcal{P}$ függvényeknek a családja, amelyeket a visszafejtéshez használunk.

3.

Mindegyik $e \in \mathcal{K}$ kulcshoz van egy $d \in \mathcal{K}$ kulcs, melyekre minden $p \in \mathcal{P}$ nyílt szöveg esetén

$$D_d(E_e(p)) = p.$$

Érdemes megjegyezni, hogy a jelölésrendszer erőteljesen kötődik az angol nyelvű szakirodalomhoz, amely igazán szerteágazónak mondható. (A T a „text” „szöveg” rövidítése, E_k az „encrypt” „titkosít”, D_k „decrypt” „fejt” szóból származik, ahol a k index az alkalmazott kulcsra utal.) A bőséges szakirodalomból a könyv végén található egy összefoglaló.

Sir Francis Bacon (1561-1626), aki politikával és filozófiával foglalkozott, elmélkedett arról is, hogy milyen is egy jó kriptorendszer. Véleménye szerint legyenek az E_k és D_k módszerek egyszerűek, a D_k fejtési kulcs nélkül ne lehessen fejteni, végül legyen a titkosított szöveg ártatlan kinézetű. Nyilvánvalóan a számítógépek korában minden bitsorozat ártatlan kinézetű, tehát ez a követelmény nem teljesíthető, de a többit továbbra is útmutatóul fogadjuk el.

Sir Francis Bacon

Valószínűleg mindenki számára nyilvánvaló, hogy senki nem teheti meg, hogy csak a titkosítással foglalkozzon, a feltörés próbája nélkül. Kitalált módszereink használhatóságát úgy tesztelhetjük, hogy az illegális betolakodó helyébe képzeljük magunkat és megpróbáljuk feltörni a rendszert. Sokszor izgalmasabb a rendszer feltörésén mesterkedni, mint a titkosítási módszert megalkotni. Ugyanakkor nagyon sok új ismerettel kecsegtetnek ezek a próbálkozások, a megismerés új dimenzióira nyitnak kaput.

A továbbiakban feltételezzük, hogy ismerjük a titkosítási módszert és fő feladatunk, hogy ráleljünk a megfejtésre.

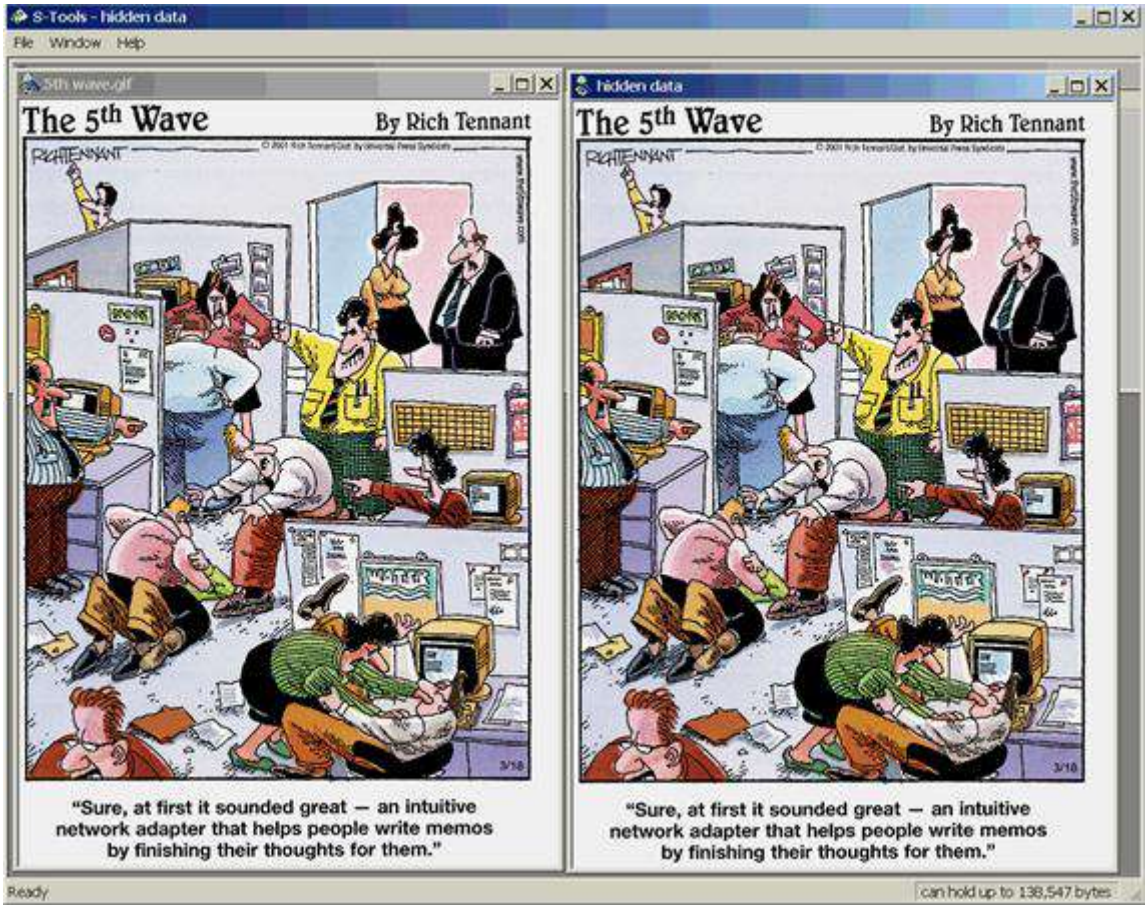
A fő kérdés, hogy mikor van egyáltalán lehetőségünk a fejtésre. Több esetet érdemes megkülönböztetni.

a) Tegyük fel, hogy ismert valamely titkosított szöveg, ami lehetőség szerint elég hosszú. Ekkor, ha rendelkezünk bizonyos statisztikai információval az adott nyelvről, akkor a klasszikus rendszerekben megpróbálkozhatunk a fejtéssel.

b) Ha ismerünk néhány $(T, E_k(T))$ párt, akkor szintén van esélyünk a fejtésre.

c) Ha elég ügyes a betolakodó és legális felhasználónak tünteti fel magát, akkor esély van olyan $(T, E_k(T))$ párok megszerzésére, amit ő választ. Így szintén jó az esély a fejtésre.

Itt említjük meg, hogy mivel főként matematikai nézőpontból vesszük szemügyre a kriptográfiát, eltekintünk néhány történetileg fontos titkosítási módszer tárgyalásától. Ilyen például a Kód könyvvel való titkosítás, amit a titkosítási rendszerek arisztokratájának is neveznek, ahol is mindkét félnek külön szótára van. Ide sorolható az üzenet elrejtése láthatatlan tintával vagy egy frissen borotvált fejen, amit a haj később benő. Az utóbbi módszereknek a neve steganográfia. A kriptográfia és a steganográfia közötti fő különbség, hogy míg az előbbinek az a célja, hogy megakadályozza illetéktelenek számára a titok elolvasását, az utóbbié az, hogy az illetéktelenek ne is tudjanak a titok létezéséről. A digitalizált képek remek lehetőségek adnak a steganográfia 21. századi alkalmazására. Ha a képpontok színét meghatározó információban egy bitet megváltoztatunk, a szemlélő számára a változás (nem túl sok pont használata esetén) nem érzékelhető, ugyanakkor a beavatott számára a megváltoztatott bitekből az információ kinyerhető. Hasonlóan lehet a digitálisan rögzített hangokat is felhasználni a steganográfiában.



steganography.zip

2. fejezet - Monoalfabetikus rendszerek

Ebben a fejezetben klasszikus titkosítási rendszereket vizsgálunk (kiváló áttekintés olvasható a témáról Simon Singh [17] munkájában). A régi idők titkosításait írjuk le, illetve fejtjük, megjegyezve, hogy ezeket a módszereket – a modern idők nyilvános kulcsú rendszereivel szemben – rejtették az avatatlan szemek elől.

Simon Singh

Az első titkosírást, amelyről tudunk, a spártaiak szkütaléját már a Kr. e. VII. században használták.

Aineiasz Taktikosz görög szerző Kr. e. 360 körül írt hadászati munkájában pedig több módszert felsorol. A klasszikus módszereket leggyakrabban háborús körülmények között használták, jellemző módon Aineiasz

Taktikosz munkája is a várvédelemmel foglalkozik. Azt azonban nem szeretnénk állítani, hogy ez az egyetlen oka a titkosításnak. A diplomácia, az államigazgatás, a tudomány és a magánélet mind-mind indokolhatták a kriptográfia használatát az elmúlt időkben.

Egy igazán különleges és magyar vonatkozású érdekesség Gárdonyi Géza naplója. Gárdonyi saját magának egy egyedi titkosírást fejlesztett ki, amely különös alakú jelekből állott. Használatukat annyira begyakorolta, hogy alkalmazásukkal a rendes folyóírással megegyező sebességgel tudott írni. Hogy gondolatait még jobban elrejtse, naplójának fedelére a „Tibetan grammar” felirat került. De a furcsa írás nem tibeti, de nem is kínai, koreai, vagy indiai: ezeket az írásjeleket a világon sehol sem használják. Ezek Gárdonyi saját találmányai, alakjuk azonban valóban valamiféle egzotikus írás képét idézi fel.

Gárdonyi Géza

A titkos napló 1922-től, az író halálától egészen 1965-ig megfejtetlen maradt. Ekkor az egri Gárdonyi Géza Emlékmúzeum nyílt pályázatot hirdetett az írás megfejtésére. Gilicze Gábor egyetemi hallgató és Gyürk Ottó honvéd egymástól függetlenül megoldották a problémát. A Titkosnaplót pedig teljes egészében kiadták.

Gárdonyi Géza naplója

A klasszikus titkosítások feltörésében nagy segítséget nyújtanak számunkra a nyelvészek által vizsgált betű illetve betűkapcsolatok statisztikái és természetesen a számítógépek.

Nem ismeretes ki jött rá elsőként, hogy a betűk gyakoriságának ismerete felhasználható a titkosírások megfejtésében, a módszer első írásba foglalójának nevét azonban ismerjük, Jákúb ibn Iszhák al-Kindi, az „arabok filozófusa", tette ezt meg IX. században. Legnagyobb értekezése, amelyet csak 1987-ben fedeztek föl az isztambuli Szulejmánia Ottomán Archívumban, a Titkos üzenetek megfejtése címet viseli.

A statisztikai módszer használatát a következőképpen kell elképzelnünk. A titkosított szöveget statisztikai módon megvizsgáljuk, azaz feltérképezzük az egyes betűk, betűpárok, sőt némely esetben nagyobb betűcsoportok előfordulásának gyakoriságát. Az így kapott gyakoriságokat összehasonlítjuk a természetes nyelv általunk ismert gyakoriságaival, így keresve megfelelő egyezéseket. Egyszerű esetben egy betű megtalálása esetén a rendszer feltörhető, de természetesen bonyolultabb rendszereknél ez nem ilyen egyszerű feladat.

Az első komolyabb gyakoriságanalízist a modern korban angol nyelven végezték el. Összesen 100362 betűn alapszik H. Beker és F. Piper állította össze, s első ízben a Cipher Systems The Protection of Communication című művükben adták közre. Az ő adataikat tartalmazzák a következő táblázatok.

ıbc	a	b	c	d	e	f	g	h	i	j
·lőfordulás	8,2	1,5	2,8	4,3	12,7	2,2	2,0	6,1	7,0	0,2

ıbc	k	l	m	n	o	p	q	r	s	t	ı
·lőfordulás	0,8	4,0	2,4	6,7	7,5	1,9	0,1	6,0	6,3	9,1	2,8

ıbc	v	w	x	y	z
·lőfordulás	1,0	2,4	0,2	2,0	0,1

A magyar nyelv statisztikai tulajdonságai szintén ismertek. A leggyakrabban előforduló magánhangzók az „a" és „e", még a mássalhangzók esetén „t, l" és „n" betűk.

Természetesen a statisztikai feltérképezés nem csak betűkre, hanem betűpárokra, betű hármasokra, illetve szavakra is kiterjed.

A nyelvre nem csak szavai, mondatszerkezete jellemző, hanem betűkészlete is. Egyes nyelvek olyan karakterrel rendelkeznek, melyek más nyelvekből hiányoznak még akkor is, ha alapvetően azonos írásmódot használnak. Ilyen vizsgálatokból általában kiderül, hogy melyik nyelvvel is van dolgunk.

A legtöbb esetben feltehető, hogy ismerjük a nyelvet, sőt az adott nyelv gyakoriság szempontjából jól fel van térképezve. Ritka nyelvcsalád természetesen jóval nehezebb a feladat, de ilyenkor nyilvánvaló a legális fejtő is bajban lehet, hiszen kevés ember érti az adott nyelvet.

Az egyik legismertebb példája a nem feltérképezett nyelvek használatának a második világháborúban használt navahó nyelv volt. Az egyik legnépesebb, de írásbeliséggel nem rendelkező indián törzs nyelve különösen alkalmas volt arra a feladatra, hogy szóbeli üzeneteket küldjenek egymásnak a hadszíntéren.

A titkosított szövegben „helyettesítő kifejezéseket” használtak, az üzeneteket nem fordították le navahóra, hanem kitaláltak egy meglehetősen bonyolult rendszert, amelyben az angol katonai szavak, fogalmak mindegyikének megfeleltettek egy navahó szót. A megfelelő szó állt ugyan valamilyen logikai kapcsolatban az angol kifejezéssel a memorizálást megkönnyítendő (például kézigránát helyett krumpli), de nem annak fordítása volt. Így a kódba be nem avatott navahó beszélő számára az üzenetek értelmetlenek voltak. A navahó kódbeszélők résztvettek a koreai és vietnámi háborúkban is. (Csak a teljesség kedvéért jegyezzük meg, hogy a titkosság miatt a résztvevő katonák semmiféle hivatalos elismerésben nem részesültek 1982-ig. Ekkor Reagan elnök hivatalosan is köszönetet mondott a katonáknak, és augusztus 14-ét a „navahó kódbeszélők napjának” nyilvánította. Arizona állam fővárosában, Phonixben, 2008-ban szobrot avattak tiszteletükre.)

Számítógépes segítség nélkül a klasszikus rendszerek kódolása és fejtése is igen nehéz feladat, ezért egyszerű segédprogramokat készítettünk a szemléltetés érdekében.

Gyakorlati szempontból megállapodunk abban, hogy a következőkben, ha titkosítunk, kizárólag ékezet nélküli betűket használunk és magyar nyelv használata esetén, kivesszük a ritkán előforduló q betűt. A továbbiakban tehát feltételezzük, hogy 25 betűből álló abc-vel dolgozunk.

Elsőként az úgynevezett monoalfabetikus rendszerekkel foglalkozunk, ez számunkra azt jelenti, hogy az egyes betűk helyettesei a titkosítás során nem változnak. Ez nagyon megkönnyíti fejtésüket, így nyilvánvalóan ezeket már nem használják, leginkább történetiségük miatt érdemes őket megemlíteni.

1. Ceasar titkosítás

Az első általunk vizsgált rendszer a Ceasar titkosítási rendszer, amely az ABC egy egyszerű elcsúsztatásából áll.

A behelyettesítéses módszer katonai célokra történő felhasználását Julius Caesar: A gall háborúk című műve dokumentálja először.

Julius Caesar

Caesar olyan gyakran folyamodott a titkosíráshoz, hogy Valerius Probus egy egész értekezést írt az általa használt kódról, ez azonban sajnos nem maradtak ránk. Suetoniusnak köszönhetően azonban, aki a II. században megírta Cézárok élete című művét, részletes leírást kapunk a Julius Caesar által használt behelyettesítéses kódról. Caesar minden betű helyett az ábécében utána következő harmadikat írta le.

C	D	...	W	X
F	G	...	Z	A

Nyilvánvalóan az eltolás mértékének, azaz egyetlen betű helyettesítőjének felismerése esetén a módszer fejthetővé válik. Így akár néhány átgondolt próbálkozás után könnyen eredményre jutunk.

Ceasar.zip

2. Kulcsszavas Caesar titkosítás

Ugyanazon az elven alapul, mint az előző Caesar módszer, csak itt van egy kulcsszavunk és azzal toljuk el az ABC-t. A kulcsszó választásánál (most és a továbbiakban is) arra kell ügyelnünk, hogy olyan szót válasszunk, amely különböző betűkből áll.

Titkosítsuk a kriptográfia szót!

Kulcsszó: SOMA

Nyílt:	A	B	C	D	E	F	G	H	I	J	K	L	M	...
Rejtjel:	S	O	M	A	B	C	D	E	F	G	H	I	J	...

KRIPTOGRAFIA = HQFNTLDQSCFS

A Ceasar rendszer kissé bonyolultabb fajtája, amikor a szöveget betűcsoportokra osztjuk és egy egységen belül az eltolás mértéke betűnként különböző. Ekkor, ha sikerül rátalálnunk, hogy hány betűnként azonos az eltolás

mértéke, hasonló módszerekkel, mint a Ceasar rendszernél, itt is célhoz érünk. Ennél az egyik legegyszerűbb titkosítási eljárásnál éppen úgy, mint a többi klasszikus rendszernél, egyszerű statisztikai vizsgálatok hamar célba juttatnak.

3. Polybios titkosítás

A következő réges régi titkosírás a Polybios. Polübiosz a harmadik pun háború nagy római hadvezérének, Cornelius Scipionak volt a tanácsadója. A következő kártya segítségével titkosíthatunk, ahol is minden betűnek egy betűpár felel meg.

A	E	I	O
A	B	C	D
F	G	H	I
K	L	M	N
P	Q	R	S
U	V	X	Y

Ebben az esetben tetszőleges betű sor és oszlop indexének leolvasásával titkosíthatunk. Minden betűnek egy betűpár felel meg. Igy például a K betűnek a IA pár, az O betűnek a IU felel meg. Az indexeket természetesen tetszőlegesen választhatjuk a betűk vagy esetleg más jelek világából. Az ábécé betűit magánhangzó párokkal helyettesítjük, ezeket a párokat észrevétlenül elrejthetjük szavakban.

Íme egy titkosított szöveg:

ITT ALUDT, AKI ELADOTT EGY UBORKAGYALUT. ITTHON CSÜCSÜLÖK. U

A fejtéshez gyűjtsük páronként össze a szöveg magánhangzóit. Ekkor a következő párokat kapjuk: IA UA IE AO EU OA AU IO UU OU.

Felhasználva az előzőekben megadott táblázatot megfejthetjük a titkosított üzenetet. Az elrejtett üzenet, KÜLDJ PÉNZT.

A titkosított szöveget az előzőekhez hasonlóan statisztikai módszerekkel fejthetjük, ügyelve arra, hogy betűpárok személyesítenek meg betűket.

4. Hill módszere

1929-ben Lester S. Hill fejlesztette ki a róla elnevezett titkosítást, amely mátrixokat használ és tetszőleges hosszúságú tömböket képes titkosítani.

Lester S. Hill

Hill módszerének alkalmazásához először egy egyszerű kódolást végzünk, amelyben az ABC betűit sorszámukkal helyettesítjük, azaz:

	B	C	D	.
	2	3	4	.

Ezen helyettesítés után minden kapott értéket $\pmod{25}$ tekintünk. A titkosításhoz egy tetszőleges $n \times n$ típusú invertálható M mátrixot használunk, amelynek elemeit természetesen $\pmod{25}$ írjuk.

A titkosítandó szavakat szóközök nélkül leírjuk, majd n betűs szakaszokra tagoljuk. Ezen szakaszokat kódoljuk és T_i n dimenziós oszlopvektorokat készítünk belőlük. Az említett műveletek elvégzése után a titkosítás képlete MT_i mátrix szorzással adható meg. A művelet T_i' oszlopvektorokat eredményez, amelyeket dekódolva egy titkosított szöveghez jutunk.

Példaként lássuk a MINDIG szó titkosítását egy 2×2-es mátrix segítségével.

Legyenek

$$M = \begin{pmatrix} 3 & 5 \\ 2 & 4 \end{pmatrix}, \quad T_1 = \begin{pmatrix} M \\ I \end{pmatrix} = \begin{pmatrix} 13 \\ 9 \end{pmatrix},$$

$$T_2 = \begin{pmatrix} N \\ D \end{pmatrix} = \begin{pmatrix} 14 \\ 4 \end{pmatrix}, \quad T_3 = \begin{pmatrix} I \\ G \end{pmatrix} = \begin{pmatrix} 9 \\ 7 \end{pmatrix}.$$

Majd képezzük az adott szabály szerint a T_1', T_2', T_3' vektorokat.

$$MT_1 = \begin{pmatrix} 57 \\ 101 \end{pmatrix}, \quad MT_2 = \begin{pmatrix} 50 \\ 86 \end{pmatrix}, \quad MT_3 = \begin{pmatrix} 41 \\ 73 \end{pmatrix}.$$

Az így kapott mátrixok elemeit $\pmod{25}$ véve a

$$T_1{}' = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad T_2{}' = \begin{pmatrix} 0 \\ 11 \end{pmatrix}, \quad T_3{}' = \begin{pmatrix} 16 \\ 23 \end{pmatrix}.$$

mátrixokat kapjuk. Így a HBALRY titkosított szöveget nyertük.

A fejtés nyilvánvalóan könnyű az M mátrix ismeretében, hiszen ha figyelmesen választottunk, akkor a mátrix invertálható és az $M^{-1}T_i'$ mátrixszorzat (az eredményeket $\pmod{25}$ véve) az eredeti szöveg betűinek kódjait adja.

Aki illegálisan akarja feltörni a rendszert annak két pár képének az ismerete szükséges. Ennek meghatározásához a betűpárok statisztikai eloszlását kell vizsgálnunk. A leggyakrabban előforduló betűpárok beazonosítása után van esélyünk a fejtésre.

Tegyük fel, hogy ismerjük a T_1 illetve a T_2 mátrixok képét. Ekkor az általunk választott M mátrixot az

$$\begin{pmatrix} 57 & 50 \\ 101 & 86 \end{pmatrix} \begin{pmatrix} 13 & 14 \\ 9 & 4 \end{pmatrix}^{-1}$$

mátrixszorzás adja. Némi szerencse is kell, hogy ez elsőre sikerüljön, ugyanis nem nyilvánvaló, hogy az inverz mátrix létezik. Ekkor más párt kell keresnünk.

Megjegyzés: Könnyű számolással adódik, hogy az említett inverz mátrix a következő

$$\begin{pmatrix} 2 & -\frac{5}{2} \\ -1 & \frac{3}{2} \end{pmatrix}$$

Megjegyezzük továbbá, hogy amennyiben az adott szöveg nem osztható n hosszúságú blokkokra, akkor az értelmet nem zavaró betűkkel kipótoljuk azt, vagy ez egyszer akarattal helyesírási hibát vétünk.

A módszer igen jónak bizonyult megalkotásakor, mert a műveletek elvégzése igen munkaigényes, ugyanakkor a számítógépek megjelenésével, mind a titkosítás, mind a fejtés nyilvánvalóvá vált.

5. Affin kriptorendszer

Az affin kriptorendszer a következő, általunk ismertetett titkosítási rendszer. Tételezzük fel, hogy a és b olyan természetes számok, melyre $0 \leq a, b \leq 24$ és $(a,25) = 1$. Ekkor az előzőekben megismert, szokásosnak mondható, kódolás elvégzése után, minden α kódú számot az $a\alpha + b \pmod{25}$ kifejezés értékével helyettesítjük, majd dekódoljuk a kapott értéket és így egy titkosított szöveghez jutunk.

Megjegyezzük, hogy az $(a,25) = 1$ feltétel ahhoz szükséges, hogy a végeredményhez szükséges $a\alpha + b \pmod{25}$ hozzárendelés kölcsönösen egyértelmű legyen. Máskülönben előfordulhatna, hogy különböző betűknek azonos képe van. Ugyanis, ha α és α_1 elemeket titkosítjuk, akkor az előállított képük $a\alpha + b \pmod{25}$ illetve $a\alpha_1 + b \pmod{25}$. Ezek akkor határoznak meg azonos betűket, ha $a(\alpha - \alpha_1) \equiv 0 \pmod{25}$ kongruencia teljesül, az pedig a feltételeket figyelembe véve csak akkor történhet, ha α és α_1 ugyanaz a szám.

A rendszer fejtése statisztikai módszerrel történik. Két betű megfejtése után a rendszer összeomlik.

6. Feladatok

1.

A KRIPTO kulcsszó segítségével titkosítsa a következő szöveget Caesar módszer felhasználásával. „A kocka el van vetve.”

2.

Affin kriptográfiai rendszert használjunk a következő szöveg titkosításánál, ahol $a = 6$ és $b = 2$. „A bölcs kevésből ért.”

3.

Titkosítsuk az „én magyar nemes vagyok” idézetet Hill módszerének segítségével, ahol

$$M = \begin{pmatrix} 6 & 5 \\ 2 & 3 \end{pmatrix}.$$

4.

Tervezzünk Polybios titkosítást geometriai alakzatok felhasználásával.

5.

A mellékelt statisztika készítő program felhasználásával fejtsük meg a szidd2.txt fájlban lévő titkosított szöveget. A titkosítás Ceasar módszerrel készült és az eredeti szöveg Hermann Hesse: Sziddharta című könyvéből való. A statisztika elkészítéséhez használjuk a stat.exe programot. (Segítségül közöljük, hogy a magyar nyelvben leggyakrabban előforduló magánhangzók az E, A, O, míg mássalhangzók esetében a T, S, N.)

3. fejezet - Polialfabetikus rendszerek

A Hill módszer pontosabb vizsgálatakor kiderül, hogy azonos betűpárok képe nem mindig ugyanaz. Ha például 2×2-es mátrixokkal titkosítunk, más lesz a képe az AK betűcsoportnak a $VAKSI$ illetve az $AKAR$ szóban.

Az ilyen titkosításokat tágabb értelemben vett monoalfabetikus helyettesítésnek nevezzük. Ez vezet át bennünket a fejezet címben említett polialfabetikus helyettesítésekhez, ahol is a szöveg titkosítása során az azonos szövegrészek helyettesítése más és más.

1. Playfair módszer

Az első ilyen módszer az úgynevezett Playfair titkosítás. A Playfair módszer egy szimmetrikus titkosítás, amelyet 1854-ben Charles Wheatstone fejlesztett ki.

Charles Wheatstone

Lord Playfair tudományban jártas politikusként támogatta a rendszer kifejlesztését, őt tisztelhetjük névadóként. Az említett redukálással élve az ABC 25 betűjét elhelyezzük egy 5×5-ös négyzetben. A szöveget úgy alakítjuk, hogy páros számú betű szerepeljen benne. Ezt páratlan számú betű esetén úgy érhetjük el, hogy valamilyen helyesírási hibát ejtünk vagy vagy valamely betűt megkettőzzük.

Ezek után a szöveget kettes blokkokba tagoljuk úgy, hogy egy blokkba két azonos betű ne szerepeljen (alkalmazhatjuk az előző trükkök valamelyikét). Ha az így kapott betűpár nem helyezkedik el azonos sorban vagy oszlopban, akkor a betűket egy képzeletbeli téglalap két szemközti csúcsának tekintve a másik két csúcspontban elhelyezkedő betűk adják a titkosított képet. Ha egy sorban vagy oszlopban helyezkednek el, akkor megegyezés szerint le vagy fel, illetve balra vagy jobbra toljuk a betűpárt és az így kapott betűk adják a titkosított képet.

Y	D	W
I	P	U
C	A	X
N	O	G
K	M	J

Az ábráinkról leolvashatók az említett titkosítási eljárások. Például az AE párnak a képe az FO betűpár, a HA pár titkosított megfelelője CX, az IN párnak CK.

Az előző módszert alkalmazva a titkosítás nem változik, ha ciklikus oszlop vagy sor cserét hajtunk végre. Itt is alkalmazhatjuk a kulcsszavas ötletet. Válasszuk kulcsnak a KUNHARCOS szóösszetételt, majd soroljuk fel a kimaradt összes betűt, ügyelve az ismétlődés elkerülésére.

A titkosítás fejtése bonyolultabb, mint az előzőek esetén. Betűpárok, hármasok, négyesek figyelése és statisztikai feldolgozása vezet célhoz. Az így kapott adatokat kell összehasonlítanunk az adott nyelv törvényszerűségeivel.

Kulcsszavas esetben a kulcsszó hosszának megfejtése elvezet a titkosítási módszer feltöréséhez, hiszen a kulcsszó után ABC sorrendben vannak a betűk.

A titkosítónak természetesen számtalan lehetősége van, hogy megnehezítse a fejtést. Minden levelet lehet különbözőképpen titkosítani vagy esetleg egy másik nyelvre lefordítani.

Néhány esetet saját magunk is kipróbálhatunk a Playfair.exe program segítségével.

2. Vigenére kriptorendszer

Bár a módszer a Vigenére sifre nevet viseli, több alkotó is közreműködött a megalkotásában. Eredete egy XV. századi firenzei polihisztorig, Leon Battista Albertiig vezethető vissza. Az 1404-ben született tudós a reneszánsz egyik kiemelkedő alakja volt, sok kiváló műve mellett legjelentősebb alkotása a Trevi-kút.

Alberti gondolkozott el először azon, hogy a monoalfabetikus titkosítást föl lehetne váltani egy több abc-t használó rendszerrel. Sajnos nem öntötte végleges formába felfedezését, így mások vitték diadalra az ötletet. Az első az 1462-ben született Johannes Trithemius német apát volt, őt az 1535-ös születésű Giambattista della Porta olasz tudós követte, majd egy 1523-ban született francia diplomata, Blaise de Vigenére zárta a sort.

Blaise de Vigenére

Vigenére huszonhat éves korában, egy kétéves római kiküldetés alkalmával ismerte meg Alberti, Trithemius és Porta műveit. Érdeklődése eleinte kizárólag gyakorlati szempontok miatt, diplomáciai feladataival

kapcsolatosan fordult a kriptográfia felé. Később, pályája elhagyása után kovácsolta elgondolásaikat egy új, egységes és erős kódrendszerré.

Blaise de Vigenére (1523-1596) Vigenére munkássága a Traicté des Chiffres (Értekezés a titkosírásról) című, 1586-ban megjelent dolgozatában csúcsosodott ki, és bár a módszer „le chiffre indéchiffrableként” (feltörhetetlen kódként) idézték, sokáig mégis feledésbe merült.

A következőkben részletezzük a módszert. A részletes leírásához szükségünk lesz a következő ábrára:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Titkosítsuk a „Nem mind arany ami fénylik” közmondást. Válasszunk az ismert feltételek szerint egy kulcsszót, jelen esetben legyen ez a MARS szó. Írjuk periódikusan a kulcsszót a titkosítandó szöveg fölé!

M	A	R	S	M	A	R	S	M	A	R	S	R	S	M
N	E	M	M	I	N	D	A	R	A	N	Y	A	M	

Ezek után az N-edik sor M-edik eleme lesz N helyettese, azaz Z. Az E-edik sor A-adik eleme lesz E helyettese, azaz E. Ugyanilyen lépésekkel jutunk el a titkosított szöveghez.

Hasonló négyzet készíthető, mint a fenti, annyi különbséggel, hogy a betűk sorrendje fordított. Ezt a megalkotója, Sir Francis Beaufort admirális, után Beaufort négyzetnek nevezzük. Az admirálisról egy szélsebesség mérték is kapott nevet.

A Vigenére rendszer egy tipikus példája annak a kriptográfiai módszernek, amikor egy kulcsszót ismétlünk periódikusan és ennek felhasználásával történik a titkosítás. Polialfabetikus rendszer volta miatt nyilvánvalóan nem használható az eddig jól bevált statisztikai módszer. Ha ismerjük azonban a kulcsszó hosszát akkor a rendszert egy monoalfabetikus rendszerre redukálhatjuk.

Tételezzük fel, hogy tudjuk a kulcsszó hosszát, jelen esetben ez négy. A titkosított szöveget helyezzük el négy oszlopban a következő módon:

b	c	
2	3	
6	7	
10	11	1
⋮	⋮	

A számok a betűk pozícióját jelölik a titkosított szövegben. Ugyanabban az oszlopban ugyanaz a betű azonos betűt reprezentál az eredeti szövegből. Ez azt jelenti, hogy ha lenne egy jó módszerünk a kulcsszó hosszának megsejtésére, akkor az előző elrendezés megvalósítása után alkalmazhatnánk a jól bevált statisztikai módszert.

Friedrich Kasiski német titkosító az 1860-as években kifejlesztett egy módszert, melynek segítségével megtalálhatjuk a kulcsszó hosszát. A róla elnevezett Kasiski módszert 1863-ban publikálta és lényege abban áll, hogy titkosított szövegben azonos betűcsoportok többszöri előfordulását vizsgáljuk. Megfigyeljük, hogy ezek az ismétlődések milyen távol vannak, azaz hány betű távolságban követik egymást.

Tegyük fel például, hogy a RUNS betűcsoport ismétlődésére talált rá egy számítógépes program. Egy ilyen betűcsoport előfordulása lehet véletlen, de minél hosszabb betűcsoport ismétlődését tudjuk megfigyelni, annál valószínűbb, hogy ugyanolyan szövegrészt titkosított a küldő. Ha az említett szövegrész előfordulása olyan, hogy:

. . . RUNS 28 betű RUNS 44 betű RUNS 68 betű RUNS . . .

Ekkor feltételezhetjük, hogy a kulcsszó hossza megegyezik ezen számok legnagyobb közös osztójával, ami jelen esetben 4.

Ha több betűcsoport ismétlődését figyeljük meg, akkor mód nyílik feltevéseink ellenőrzésére. Szerencsés esetben ezek egyértelműsítik a kulcsszó hosszát. Ellenkező esetben csak az oszlopos felosztás és a statisztikai módszerek végrehajtása után derül ki, hogy melyik változat az igazi.

Itt is az igaz, mint az előzőekben, a módszer meglehetősen időigényes, ha nem használunk számítógépet, esetünkben nyilvánvalóan gyorsan célhoz érünk.

Megjegyezzük, hogy Kasiskitól függetlenül Charles Babbage is kiötlötte ezt a módszert még 1846-ban.

Charles Babbage

3. Autoclave rendszer

A Vigenére módszer egy titkosított változata az Autoclave rendszer, melyet a híres matematikus Gerolamo Cardano (1501-1576) talált ki.

Gerolamo Cardano

Ebben a rendszerben a forrás szöveget használjuk titkosítási kulcsként egy bizonyos eltolás közbeiktatásával.

Legyen az eltolás mértéke 4 betű és titkosítsuk a jól ismert közmondást, „Aki mer az nyer”. Ekkor így néz ki a titkosítás:

Forrás szöveg: $AKIMERAZNYER$

Kulcs: $NYERAKIMERAZ$

A kulcsot úgy használjuk, mint a Vigenere rendszerben. A kimaradt részt kitölthetjük a forrásszöveg végével, mint ahogy előbb láttuk vagy kitalálhatunk egy éppen ideillő kulcsszót. Jelen esetben megfelelő választás a SOMA név. Így meghatározhatjuk a titkosított szöveget.

Forrás szöveg: $AKIMERAZNYER$

Kulcs: $SOMAAKIMERAZ$

Titkos szöveg: $SYUMEBIMRPEQ$

A legális fejtőnek nyilvánvalóan könnyű dolga van, hiszen a kulcsszó ismeretében megkapja az eredeti szöveg néhány első betűjét, amelyek a további titkosítási kulcsot jelentik.

Lehetséges egy másik variáció használata is. Ekkor is egy titkosítási kulcsszót választunk, de a másik módszertől eltérően nem a forrásszöveg, hanem a titkosított szöveg betűi adják az alkalmazott kulcsot.

Forrás szöveg: $AKIMERAZNYER$

Kulcs: $SOMASYUMQVRZ$

Titkos szöveg: $SYUMWPULDTVQ$

Az illegális fejtőnek a fő célja a kulcsszó hosszának a meghatározása. Az előzőekben részletesen kifejtett Kasiski módszer itt is lehetőséget ad a kulcsszó hosszának a meghatározására. Megfigyelhetjük azonban, hogy ebben az esetben a módszer nem olyan erős, mint az előző esetben, hiszen csak elegendően hosszú szövegben fordulhat elő nagy valószínűséggel, hogy ugyanolyan betűcsoport titkosít ugyanolyan betűcsoportot.

Az eredeti módszerben szükséges a kulcsszó kitalálása is. Gyakoriság táblázat segítségével választunk egy tetszőleges kezdő betűt (25 választás lehetséges). Ez a betű a titkosított szöveg első betűjével együtt meghatározza a forrásszöveg első betűjét. Mivel a forrásszöveg betűit használtuk a titkosításhoz, sikerül meghatároznunk a titkosítási kulcs egy újabb betűjét. Eredeti példánkban, ahol a kulcsszó négy betűből állt, megtalálhatjuk a titkosítási kulcs ötödik betűjét. Az eljárást folytatva meghatározhatjuk $9, 13, 17, \ldots$ pozícióban lévő forrásszöveg betűit. Ha ezen betűk gyakorisága ellentmond a statisztikai eredményeknek, akkor új betűvel próbálkozunk. Hasonlóan határozzuk meg a többi, kulcsszóban szereplő betűt.

Az első fejezetben áttekintettünk néhány régi titkosítási rendszert. Megfigyelhettük, hogy legfőbb segítségünk a betűk statisztikai eloszlásának ismerete. Ebből következik, hogy a titkosítás fejtőjének egyik fő feladata, hogy rendelkezzen pontos információval, hogy milyen nyelv szavait titkosították.

Nyilvánvalóan mindenféle lehetőséget kitalálnak a küldők, hogy megnehezítsék az illegális fejtők dolgát. Az egyik legnépszerűbb trükk, hogy egy jól ismert nyelven meglévő információt egy ritka, statisztikailag nem feltérképezett nyelvre fordítják le és úgy titkosítják. Itt érvényes főleg a kriptográfia fő mottója, mely szerint: „Soha ne becsüljük le a titkosítót”.

Ezen megjegyzésekkel azonban már egy a titkosításon túli területre tévedünk, amit politikának, hírszerzésnek, ármánykodásnak nevezünk, így itt a mi kíváncsiságunk félbeszakad.

4. Feladatok

1.

A fentiekben ismertetett Playfair módszer segítségével titkosítsa a „valoszinusegszamitas” szót.

2.

Vigenére módszer segítségével titkosítsa Petőfi Sándor „A magyar nemes” című versének egy sorát. „Tán a tudománynak éljek?”. Kulcsszónak válasszuk a „vers” szót.

3.

Az Autoclave módszer felhasználásával titkosítsuk, a fejezetben említett kitalálójának, Gerolamo Cardanonak a nevét, kulcsszónak használjuk a „matek” szót.

4.

Végezzük el az előző titkosítást úgy, hogy a kulcsszó használata után a titkosított szöveg legyen a titkosító kulcs.

5.

Fejtsük meg a következő Playfair módszerrel titkosított szöveget, ahol a kulcsszó a „kezdo” szó volt. Szöveg: „pcckxilrklndvnjlmylrcbszzrglgobvbvldfu”

4. fejezet - Matematikai alapok

1. Oszthatóság

A következőkben a továbbiak megértéséhez elengedhetetlenül szükséges matematikai alapokat tárgyaljuk. Jelen fejezetben nem térünk ki az elliptikus görbék elméletére, amely a későbbiekben kerül tárgyalásra.

4.1. Definíció. Azt mondjuk, hogy a b természetes szám osztható az a természetes számmal, ha van olyan x természetes szám, melyre $b = ax$.

A fentiekre az $a \mid b$ jelölést fogjuk használni, ha a nem osztható b-vel, akkor az $a \nmid b$ jelölést használjuk.

A következőkben néhány fontos oszthatósági tulajdonságot sorolunk fel.

4.2. Tétel. Minden a, b, c természetes szám esetén

1. $a \mid b$-ből következik $a \mid bc$ minden egész c esetén,
2. $a \mid b$ és $b \mid c$-ből következik $a \mid c$,
3. $a \mid b$ és $a \mid c$-ből következik, hogy

   $a \mid (bx + cy)$

   minden egész x, y esetén,
4. ha $m \in \mathbb{N}$, akkor $a \mid b$ és $ma \mid mb$ ekvivalensek.

4.3. Tétel (A maradékos osztás tétele). Teszőleges $a > 0$ és b egész számokhoz létezik olyan, egyértelműen meghatározott q és r egész szám, amelyekre

$$b = aq + r, \quad 0 \leq r < a.$$

4.4. Tétel. Ha a és b egész számok közül legalább az egyik nem 0, akkor közös osztóik legnagyobbikát a és b legnagyobb közös osztójának nevezzük és (a, b)-vel jelöljük.

4.5. Tétel. Ha a b és c számok legnagyobb közös osztója g, akkor létezik olyan x_0 és y_0 egész úgy, hogy

$$g = (b, c) = bx_0 + cy_0.$$

4.6. Tétel. Az a és b számok g legnagyobb közös osztója jellemezhető a következő két módon:

1. a $ax + by$ alak legkisebb pozitív értéke, ahol x és y végigfut az egész számokon;
2. a és b közös osztója, amely a és b minden közös osztójával osztható.

4.7. Tétel. Minden pozitív m számra

$$(ma, mb) = m(a, b).$$

4.8. Tétel. Ha $d \mid a$ és $d \mid b$ és $d > 0$, akkor

$$\left(\frac{a}{d}, \frac{b}{d}\right) = \frac{1}{d}(a, b).$$

Ha $(a, b) = g$, akkor

$$\left(\frac{a}{g}, \frac{b}{g}\right) = 1.$$

4.9. Definíció. Azt mondjuk, hogy a és b relatív prímek, ha $(a, b) = 1$.

4.10. Tétel. Minden x esetén

$$(a, b) = (a, b + ax).$$

Az egyszerű tulajdonságok bemutatása után a legnagyobb közös osztó meghatározására szolgáló tételt ismertetünk. Nevét az ókori görög matematikusról Euklidészről kapta.

Euklidész

Euklidész híres tankönyvéről az Elemekről, sokan állítják, hogy a Biblia után a legtöbbször megjelentetett mű. A róla elnevezett algoritmusról a történészek úgy vélik, hogy elmúlt korok munkáiból származik, nem saját eredmény.

4.11. Tétel (Euklideszi algoritmus). Adott b és $c > 0$ egészekre ismételten alkalmazzuk a maradékos osztás tételét, s ezzel az egyenletek következő sorozatát kapjuk:

$$\begin{aligned}
b &= cq_1 + r_1, \quad 0 < r_1 < c, \\
c &= r_1 q_2 + r_2, \quad 0 < r_2 < r_1, \\
r_1 &= r_2 q_3 + r_3, \quad 0 < r_3 < r_2, \\
&\vdots \\
r_{j-2} &= r_{j-1} q_j + r_j, \quad 0 < r_j < r_{j-1}, \\
r_{j-1} &= r_j q_{j+1}.
\end{aligned}$$

A b és c számok (b, c) legnagyobb közös osztója r_j, az osztási eljárás utolsó nemnulla maradéka.

2. Prímek

A prímek, mint az atomok az anyag világában, nagyon fontos szerepet játszanak a számelméletben és a kriptográfiában is.

4.12. Definíció. A $p > 1$ egész számot prímszámnak nevezzük, ha p-nek nincs olyan d osztója, melyre $1 < d < p$. Ha az a egész nem prím, akkor összetett számnak nevezzük.

4.13. Tétel (A számelmélet alaptétele, Gauss 1801). Bármely $n > 1$ egész szám felbontása prímek szorzatára egyértelmű, eltekintve az egységfaktortól és a prímek sorrendjétől.

A tételt Carl Friedrich Gaussnak (1777-1855) köszönhetjük, akit gyakran a „matematika fejedelmének” is szoktak nevezni. Gauss a matematika több ágában is maradandót alkotott.

Carl Friedrich Gauss

Már kicsiny gyermekkorában nyilvánvaló volt kimagasló tehetsége, több anekdota keringett az ifjú Gaussról. A 24 éves korában megírt Disquisitiones Arithmeticae című munkája a számelmélet egyik legalapvetőbb műve, amelyből a fenti tétel is származik.

Megjegyzések a faktorizációról

A következőkben igazoljuk, hogy egy tetszőleges n összetett szám legkisebb faktora kisebb, mint $\sqrt{n}$. Legyen

$$n = fa \quad \text{és} \quad f \leq a.$$

Ebben az esetben

$$\begin{aligned} a = \frac{n}{f} \Rightarrow f &\leq \frac{n}{f} \Rightarrow f^2 < n \\ f &< \sqrt{n}. \end{aligned}$$

Az előző eredmény érdekes gondolatkísérletre ad lehetőséget, ami a prímek rejtélyes tulajdonságaira és a kriptográfiában való alkalmazhatóságukra utal. 100 jegyű szám esetén

$$n > 10^{100} \Rightarrow \sqrt{n} > 10^{50}$$

Az egyszerűség kedvéért tegyük fel, hogy 10^{10} lépést végez a számítógép másodpercenként. Ez elég jó közelítése a valóságos helyzetnek. Ekkor 10^{40} másodperc, kb. 10^{31} év szükséges, hogy aprólékos kereséssel megtaláljuk a legkisebb prímfaktort. Ahhoz, hogy elég jó összehasonlításunk legyen az időtényező szemléléséhez, tudnunk kell, hogy az univerzum életkora kb. $2 \cdot 10^{11}$ év.

Mivel a prímek száma, előfordulásuk és eloszlásuk fontos kérdés, ha kriptográfiai alkalmazhatóságukat vizsgáljuk, a számelméleti eredményekhez fordulhatunk bizalommal.

4.14. Tétel (Euklidész). A prímszámok száma végtelen.

4.15. Tétel. A prímek sorozatában tetszőleges nagy hézeg van, másszóval tetszőleges k egész számhoz létezik k egymás után következő összetett szám.

Georg Friedrich Bernhard Riemann (1826-1866) nagyon fiatalon elhunyt kiváló matematikus volt.

Georg Friedrich Bernhard Riemann

Lenyűgöző alkotást hagyott az utókorra analízis, differenciálgeometria és az analitikus számelmélet terén. Az általa megfogalmazott sejtés (Riemann sejtés) a hét Millenniumi Probléma egyike, amelyek megoldására 2000-ben magas pénzjutalommal járó díjat alapított az amerikai Clay Matematikai Intézet. A következő definíciót ő alkotta a prímszámok viselkedését vizsgáló munkájában.

4.16. Definíció. Jelölje $\pi(x)$ minden valós x-re az x-nél nem nagyobb prímszámok számát.

Pafnutyij Lvovics Csebisev (1821-1894) orosz matematikusnak sikerült igazolnia, hogy minden természetes szám és kétszerese között van prím. Számelméleti munkásságából származik a következő tétel.

Pafnutyij Lvovics Csebisev

4.17. Tétel (Csebisev). Létezik olyan a és b pozitív állandó, hogy

$$a\frac{x}{\log x} < \pi(x) < b\frac{x}{\log x}.$$

A 19. század egyik leghíresebb problémája a prímszámtétel volt, amelyet egymástól függetlenül Jacques Hadamard és de la Vallée Poussin igazolt 1896-ban.

Jacques Hadamard

de la Vallée Poussin

4.18. Tétel (Prímszámtétel, 1896).

$$\lim_{x\to\infty} \frac{\pi(x)\log x}{x} = 1.$$

A következőkben néhány érdekes prímtulajdonságra térünk ki, illetve bemutatunk néhány klasszikus problémát.

4.19. Tétel. Minden P prímszám előállítható négy négyzetszám összegeként.

4.20. Tétel. Adott egy $f(x)$ egész együtthatós polinom, végtelen sok pozitív m létezik, amelyre $f(m)$ összetett.

Mint ahogy később látni fogjuk, a prímek megtalálása, főleg nagy prímek esetén, nem egyszerű dolog. Mindig nagy álma volt a matematikusoknak, hogy olyan kifejezést találjanak, amely bizonyos paraméterek esetén prímeket állít elő. Ezek közül a próbálkozások közül két, történetileg jelentőset említünk meg a következőkben.

4.21. Definíció. Az $M(n) = 2^n - 1$ alakú számokat, ahol n nem negatív egész Mersenne-számoknak nevezzük.

Marin Mersenne (1588-1648) francia szerzetes, matematikus és fizikus volt.

Marin Mersenne

Az érdekesség kedvéért érdemes megemlíteni, hogy ugyanabba a jezsuita iskolába járt, ahová később René Descartes. A róla elnevezett Mersenne számok közül azokat a prímeket nevezzük Mersenne-prímeknek, ahol a kitevőben szereplő n prím.

A Mersenne számok felszínre kerüléséhez érdemes egy kis kitérőt tennünk a tökéletes számok birodalmába. Tökéletes számnak nevezzük azt a természetes számot, amely egyenlő a tőle kisebb oszóinak az összegével. Például a 6 tökéletes szám, hiszen $6 = 1 + 2 + 3$.

Euklidész észrevette, hogy az első négy tökéletes szám $2^{n-1}(2^n - 1)$ alakú, ahol $2^n - 1$ prím. Ezekben az esetekben $n = 2,3,5,7$. A sejtést, miszerint Euklidész képlete az összes tökéletes számot leírja, több mint másfél ezer évvel utána, Leonhard Euler bizonyította be.

Leonhard Euler

Mersenne Cogitata Physica-Mathematica (1644) munkájában azt a hibás állítást mondta ki, hogy az $n \leq 257$ esetében mindig prímeket kapunk, még a többi esetben összetett számokat.

A későbbiek során Leonhard Euler (1707-1783) svájci matematikus és fizikus igazolta, hogy az $n = 31$ eset tényleg prímet szolgáltat. Az így kapott prím volt több, mint száz évig a legnagyobb ismert prím. Később kiderült, hogy az előző lista helyesen $n = 2, 3, 5, 7, 13, 17, 19, 31, 61, 89, 107, 127$.

Eddig összesen 47 Mersenne-prímet találtak. A legutóbbit 2009. áprilisában, ahol is $n = 42643801$. Érdekesség, hogy ez a szám 12837064 számjegyből áll. További Mersenne-prímek keresése világméretű összefogással folyik, nagy számú számítógép felhasználásával. (További részletek tekinthetők meg a http://www.mersenne.org/ honlapon.)

További érdekességgel szolgálnak a Fermat-számok.

4.22. Definíció. Az $F(n) = 2^{2^n} + 1$ alakú prímeket, ahol n nem negatív egész Fermat-prímeknek nevezzük.

Pierre de Fermat (1601-1665) francia jogász volt, aki szívesen és eredményesen foglalkozott szabad idejében matematikával is.

Pierre de Fermat

Az említett probléma érdekes ugyan, mégsem ez tette nevezetessé Fermat, hanem a következő sorok. „Lehetetlen egy köbszámot felírni két köbszám összegeként, vagy egy negyedik hatványt felírni két negyedik hatvány összegeként, általában lehetetlen bármely magasabb hatványt felírni két ugyanolyan hatvány összegeként igazán csodálatos bizonyítást találtam erre a tételre. A margó azonban túlságosan keskeny, semhogy ideírhatnám.” A Fermat által megfogalmazott állítás margónyi bizonyítását azóta sem találják a matematikusok. Andrew Wiles, princetoni professzor, 1995-ben igazolta a sejtés igazságát több, mint 100 oldalon.

Fermat nem fektetett nagy hangsúlyt a bizonyításokra, így az a sejtése, hogy a $2^{2^n} + 1$ alakú számok mindig prímek, is csak sejtés maradt. Euler 1732-ben igazolta, hogy 641 osztja $F(5)$-öt.

Jelen témánkkal kapcsolatban is rengeteg megoldatlan probléma van a számelméletben. Nem tudjuk, hogy létezik-e végtelen sok Mersenne prím, Fermat prím vagy létezik-e páratlan tökéletes szám.

3. Kongruenciák

A kongruenciák elméletét, a mai formában, Carl Friedrich Gauss dolgozta ki Disquisitiones Arithmeticae című művében.

4.23. Definíció. Legyenek a és b egész számok. Ha az m nemnulla egész osztja az $a-b$ különbséget, akkor azt mondjuk, hogy az a szám kongruens b-vel modulo m. A továbbiakban $a \equiv b \pmod{m}$ módon jelöljük.

4.24. Tétel. Legyenek a, b, c, d, x és y egész számok. Ha $a \equiv b \pmod{m}$ és $b \equiv c \pmod{m}$, akkor $a \equiv c \pmod{m}$. Ha $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m}$, akkor $ax + cy \equiv bx + dy \pmod{m}$. Ha $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m}$, akkor $ac \equiv bd \pmod{m}$.

4.25. Tétel. Legyen f egész együtthatós polinom. Ha $a \equiv b \pmod{m}$, akkor $f(a) \equiv f(b) \pmod{m}$.

4.26. Tétel. $ax \equiv ay \pmod{m}$ akkor és csak akkor, ha $x \equiv y \pmod{\frac{m}{(a,m)}}$.

4.27. Tétel. Ha $ax \equiv ay \pmod{m}$ és $(a,m) = 1$ akkor $x \equiv y \pmod{m}$.

4.28. Definíció. Ha $x \equiv y \pmod{m}$, akkor y-t az x szám m szerinti maradékának nevezzük. Az $x_1, \ldots, x_m$ számok halmazát teljes maradékrendszernek nevezzük modulo m, ha tetszőleges y egész számhoz létezik egy és csak egy x_j, amelyre $y \equiv x_j \pmod{m}$.

4.29. Definíció. Az r_i egész számok halmazát redukált maradékrendszernek nevezzük modulo m, ha $(r_i, m) = 1$; $r_i \not\equiv r_j \pmod{m}$, valahányszor $i \neq j$, és tetszőleges, m-hez relatív prím x egész számhoz található olyan, halmazbeli r_i, hogy $x \equiv r_i \pmod{m}$.

Jelölés. Minden redukált maradékrendszer$\pmod{m}$ ugyanannyi elemet tartalmaz. Ezt a közös elemszámot $\phi(m)$-el jelöljük és Euler-féle ϕ függvénynek nevezzük.

4.30. Tétel. A $\phi(m)$ szám az m-nél nem nagyobb, m-hez relatív prím pozitív egészek száma.

4.31. Tétel (Euler). Ha $(a,m) = 1$, akkor

$$a^{\phi(m)} \equiv 1 \pmod{m}.$$

4.32. Tétel (Fermat). Legyen p prímszám és tegyük fel, hogy $(a,p) = 1$, ekkor

$$a^{p-1} \equiv 1 \pmod{p}.$$

4.33. Tétel. Legyen $g = (a,m)$. Ha $g \nmid b$, akkor az $ax \equiv b \pmod{m}$ kongruenciának nincs megoldása; ha viszont $g \mid b$, akkor a kongruenciának g megoldása van és a megoldások: az

$$x \equiv \frac{b}{g} x_0 + t\frac{m}{g} \pmod{m},\ t = 0,1,\ldots g-1$$

értékek, ahol x_0 az

$$\frac{a}{g} x \equiv 1 \pmod{\frac{m}{g}}$$

kongruencia tetszőleges megoldása.

4.34. Példa. Oldjuk meg a $15x \equiv 25 \pmod{35}$ lineáris kongruenciát.

A megoldáshoz a 4.33 tétel eredményét használjuk. Mivel $(15,35) = 5$ és $5 \mid 25$ a kongruencia megoldható. Könnyen látható, hogy

$$3x \equiv 1 \pmod 7 \Rightarrow x_0 = 5.$$

Megoldás: $x \equiv 25 + 7t \pmod{35}$ és $t = 0, 1, 2, 3, 4.$

A következő, több kongruenciából álló szimultán kongruenciarendszerekről szóló állítást, már több mint 2000 évvel ezelőtt ismerte egy kínai matematikus, Szun Cu, innen kapta a tétel mai nevét.

4.35. Tétel (Kínai maradéktétel). Ha az $m_1, m_2, \ldots, m_r$ pozitív egészek páronként relatív prímek, és a továbbiakban $a_1, a_2, \ldots, a_r$ tetszőleges egész számok, akkor az

$$x \equiv a_i \pmod{m_i}\, i = 1,2,\ldots r$$

kongruenciáknak van közös megoldása. Bármely két megoldás kongruens modulo $m_1 m_2 \cdots m_r$.

Módszer. Legyen $m = m_1 m_2 \cdots m_r$ és

$$\frac{m}{m_j} b_j \equiv 1 \pmod{m_j}.$$

Ekkor

$$x_0 = \sum_{j=1}^{r} \frac{m}{m_j} b_j a_j$$

4.36. Példa. Válasszunk egy 60-nál kisebb számot, osszuk el 3,4,5 számokkal és közöljük a maradékot.

A gondolt szám „kitalálására” a következő módszert javasolja a kínai könyv, a $40a + 45b + 36c$ szám 60-nal való osztási maradéka, feltéve ha a maradékok rendre a, b, c.

Például, ha a választott szám 29, akkor $40 \cdot 2 + 45 \cdot 1 + 36 \cdot 4 = 269$, amely 60-nal való osztás után tényleg adja a végeredményt.

A 4.35 tétel alapján a megoldás a következő

$$20b_1 \equiv 1 \pmod 3$$
$$15b_2 \equiv 1 \pmod 4$$
$$12b_3 \equiv 1 \pmod 5$$

Ekkor $b_1 = 2, b_2 = 3, b_3 = 3$ és

$$x_0 = 20 \cdot 2 \cdot 2 + 15 \cdot 3 \cdot 1 + 12 \cdot 3 \cdot 4 \pmod{60} = 29.$$

4. Véges testek

A véges testek elméletének kidolgozása Evariste Galois (1811-1832) munkásságával kezdődött. Az utóbbi években erőteljes alkalmazása révén (például az algebrai kódok elmélete, kriptográfia) különösen nagy jelentőségre tett szert.

A következőkben egy nagyon egyszerű bevezetését adjuk a véges testek elméletének.

4.37. Definíció. Csoportnak nevezünk egy olyan nem üres G halmazt, amelyen definiálva van egy $\star$ kétváltozós művelet, és teljesülnek a következő feltételek:

1.

A $\star$ művelet asszociatív,

2.

A G halmaz rendelkezik úgynevezett neutrális elemmel, azaz van olyan e eleme, hogy a G halmaz bármely a elemére $e * a = a * e = a$ teljesül,

3.

A G halmaz bármely a eleméhez hozzárendelhető egy olyan b G-beli elem, hogy $a * b = b * a = e$. A b elemet az a elem inverzének nevezzük, és a^{-1}-gyel jelöljük.

4.38. Definíció. Ha a csoportművelet kommutatív, azaz G minden a, b elempárjára $a * b = b * a$ teljesül, akkor a csoportot kommutatív csoportnak vagy Abel-csoportnak nevezzük.

4.39. Definíció. Egy G multiplikatív csoportot ciklikusnak nevezünk, ha van olyan $a \in G$ eleme, hogy minden $b \in G$ esetén van olyan j egész, melyre $b = a^j$. Az ilyen a elemeket a ciklikus csoport generátorának nevezzük.

4.40. Definíció. Legyenek $+$ és $.$ kétváltozós műveletek az R halmazon, amelyeket összeadásnak, illetve szorzásnak nevezünk. Az R halmaz gyűrű, ha a következő feltételek teljesülnek:

1.

$(R; +)$ Abel-csoport,

2.

a szorzás az összeadásra nézve disztributív, azaz $(a + b) \cdot c = a \cdot c + b \cdot c$ és $c \cdot (a + b) = c \cdot a + c \cdot b$ tetszőleges $a, b, c \in R$ esetén,

3.

R tetszőleges a, b és c elemeire teljesül, hogy $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, azaz a szorzás asszociatív művelet.

4.41. Definíció. Az R gyűrűt egységelemes gyűrűnek nevezzük, ha tartalmaz olyan 1-gyel jelölt elemet, hogy $a \cdot 1 = 1 \cdot a = a$ tetszőleges $a \in R$ esetén.

4.42. Definíció. Az R gyűrűt kommutatív gyűrűnek nevezzük, ha tetszőleges a és b elemeire teljesül, hogy $a \cdot b = b \cdot a$, azaz a szorzás kommutatív művelet.

4.43. Definíció. Az R gyűrű nullától különböző a elemét bal nullosztónak nevezzük, ha létezik olyan nullától különböző b elem, hogy $ab = 0$. Hasonlóan definiáljuk a jobb nullosztó fogalmát. Ha az R gyűrű nem tartalmaz sem bal sem jobb nullosztót, akkor nullosztómentesnek nevezzük.

4.44. Definíció. A kommutatív, egységelemes és nullosztómentes gyűrűt integritástartománynak nevezzük.

4.45. Definíció. Egy egységelemes gyűrűt ferdetestnek nevezünk, ha bármely nullától különböző elemének van multiplikatív inverze. Egy kommutatív ferdetestet testnek nevezünk.

Egy tetszoleges K test feletti egyváltozós polinomok halmaza, amit $K[x]$-el jelölünk, integritástartomány. A maradékosztályok Z_n halmaza is gyűrű. Könnyen igazolható, hogy minden F test nullosztómentes.

4.46. Tétel. Minden F véges integritási tartomány test.

4.47. Tétel. $\mathbb{Z}_n$ akkor és csak akkor test, ha n prím.

Például Z_2, Z_3 és Z_5 véges testek, de Z_9 nem az, mivel a 3-maradékosztálynak nincs multiplikatív inverze Z_9-ben. A -elemű véges testeket -el jelöljük, ahol a a „Galois field” angol kifejezés rövidítése.

4.48. Tétel. Minden p prímszám és minden n természetes szám esetén létezik $q = p^n$-elemű test.

4.49. Definíció. Az F test karakterisztikája az a legkisebb m természetes szám, melyre

$$ma = \sum_{i=1}^{m} a = a + a + \cdots + a = 0,$$

minden $a \in F$ elemre (az összegzés a testbeli összeadás). Ha nem létezik ilyen m szám, akkor azt mondjuk, hogy a test karakterisztikája 0.

Megjegyezzük, hogy $(\mathbb{Z}_3, +, \cdot)$ gyűrű karakterisztikája 3, a $(\mathbb{Z}_4, +, \cdot)$ karakterisztikája 4, a $(\mathbb{Z}_n, +, \cdot)$ karakterisztikája n. A $(\mathbb{Z}, +, \cdot)$ és a $(\mathbb{Q}, +, \cdot)$ gyűrűk karakterisztikája pedig 0.

Könnyen belátható, hogy ha F egy p karakterisztikájú test, akkor F-ben van egy p-elemű $\mathbb{GF}(p)$ résztest, amelynek elemei

$$1, 1 + 1, \ldots, \sum_{i=1}^{m} a = 0.$$

Ezek az elemek mind különbözőek, a szorzásra és az összeadásra nézve zárt halmazt alkotnak, valamint a nullától különböző elemeknek létezik additív és multiplikatív inverzük. Ez a p-elemű résztest izomorf Z_p-vel, így mondhatjuk, hogy minden véges, p karakterisztikájú test Z_p. A $\mathbb{GF}(p)$-t a p karakterisztikájú F véges test prímtestének nevezzük.

A továbbiakban F^* jelöli az F test nullától különböző elemeinek halmazát.

4.50. Definíció. Az F^* egy α elemét primitívnek nevezzük, ha az F test minden nem nulla eleme egyértelműen felírható α valamely pozitív kitevős hatványaként.

4.51. Tétel. Ha az F test egy q-elemű véges test, akkor a test minden α elemére teljesül $\alpha^q = \alpha$, tehát az F minden eleme gyöke az $f(x) = x^q - x$ polinomnak.

A véges testek egyszerű struktúráját a következő tétel mutatja.

4.52. Tétel. A $\mathbb{GF}(q)$ nullától különböző elemei a szorzásra nézve ciklikus csoportot alkotnak.

4.53. Tétel. Minden $F = \mathbb{GF}(q)$ testben létezik primitív elem.

4.54. Tétel. Minden véges test tekinthető egy Z_p feletti vektortérnek, s ha ez a vektortér $n \in \mathbb{N}$ dimenziós, akkor a test elemeinek száma p^n, ahol p prím.

4.55. Definíció. Az F véges test elemeinek számát a test rendjének nevezzük.

5. Feladatok

1.

Oldjuk meg az $25x \equiv 15 \pmod{29}$ és a $5x \equiv 2 \pmod{26}$ kongruenciákat.

2.

Ha egy kosár tojást 2, 3, 4, 5 vagy 6-osával ürítünk ki, rendre 1, 2, 3, 4, 5 tojás marad benne. Ha azonban 7-esével vesszük ki a tojásokat, akkor egy sem marad benne. Adja meg a kosárban lévő tojások minimális számát. (Brahmagupta i.sz. VII.sz.)

3.

Milyen maradékot ad 4444^{4444}, ha elosztjuk 9-cel

4.

Kongruenciák segítségével oldja meg a következő diophantoszi egyenleteket $4x + 51y = 9$, illetve $5x - 53y = 17$.

5.

Számítsuk ki 12543 és 29447 legnagyobb közös osztóját.

6.

Van egy 12 és egy 51 literes hordónk. Tele lehet-e tölteni ezek segítségével egy 5211 literes tartályt úgy, hogy a hordókat akárhányszor teletöltve, azok teljes tartalmát beönthetjük a tartályba, és onnan a víz nem csordul ki?

7.

Mutassuk meg, hogy tetszőleges a, b egészekre $(a^2, b^2) = (a, b)^2$.

8.

Az euklideszi algoritmussal számítsuk ki $a = 255$ és $b = 111$ legnagyobb közös osztóját, valamint a $g = ax + by$ lineáris kombinációs előállításhoz az x és y együtthatókat.

9.

Teljes maradékrendszer-e $7, 22, 37, 52, \ldots, 11632, 11647 \pmod{777}$?

10.

Redukált maradékrendszer-e $5, 15, 25, 35, 45, 55, \ldots, 155 \pmod{32}$?

11.

A kínai maradéktétel segítségével oldjuk meg a következő lineáris kongruenciarendszert

$$5x \equiv 1 \pmod{7}$$
$$4x \equiv 1 \pmod{9}$$
$$8x \equiv 1 \pmod{13}$$

5. fejezet - DES

Egészen a 2000-es évekig a kriptográfiában leginkább használatos algoritmus a DES (Data Encryption Standard) volt.

Az IBM az 1960-as évek végén indított el egy kutatási projektet egy szimmetrikus, titkos kulcsos titkosítási rendszer fejlesztésére. Horst Feistal vezetésével 1971-re kifejlesztették az akkor LUCIFER-nek nevezett algoritmust, amely 128 bites blokkokra osztotta a nyílt szöveget és 128 bites kulcsot alkalmazott a titkosításhoz.

Horst Feistal

A LUCIFER-t eladták a londoni Lloyd's biztosítónak, amely egy szintén az IBM által fejlesztett készpénz-elosztó rendszerben alkalmazta. Carl Meyer és Walter Tuchman egyetlen chipen akarta implementálni a LUCIFER algoritmust végrehajtó célhardvert, amelyhez némi változtatást is végrehajtott az algoritmusban.

A 70-es évek közepe táján hirdetett pályázatot az NSA (National Security Agency) egy olyan titkosítási eljárásra, amely szabványosítható. Erre a pályázatra nyújtotta be az IBM Carl Meyer és Walter Tuchman az általuk kitalált eljárást, amely messze a legjobb volt az összes benyújtott pályázat között, amit aztán 1977-ben DES néven szabványosítottak is.

A módszer kiválóan illeszkedett a rohamosan fejlődő elektronikus adatfeldolgozás lehetőségeihez. Magas szintű biztonságot nyújtott, amelyet egyszerű felépítéssel valósított meg. A hardver megoldások jóval hatékonyabbak, mint a szoftveresek, hiszen a DES rengeteg bitszintű műveletet végez. Az algoritmus rendelkezik az úgynevezett lavinahatással, ami azt jelenti, hogy ha a bemeneti blokk kis változására a kimeneti blokk erőteljesen változik meg.

1. Feistel titkosítás

A DES algoritmust szokás Feistel titkosításnak is hívni. Az algoritmus egy 64 bites blokkos algoritmus, vagyis a nyílt szöveg egy 64 bites blokkjához egy ugyanekkora rejtjelezett blokkot rendel hozzá. A hozzárendelés csak a használatos kulcstól függ.

Minden lépés az előző lépés eredményét használja fel, mégpedig ugyanolyan módon, bár kulcstól függően. Egy ilyen lépést körnek (round) nevezünk, és ezen köröknek a száma a használatos algoritmus jellemzője.

Legyen t a blokk hosszúság. Legyen f_K a kódoló függvény a K kulcshoz, amelyet körfüggvénynek nevezünk és amelytől nem várjuk el, hogy invertálható legyen. Rögzítünk egy $r \geq 3$ számot (Feistel titkosítások esetén ez páros szám) a sorozathoz, a $\mathcal{K}$ kulcs teret és egy módszert, hogy egy tetszőleges $k \in \mathcal{K}$ kulcshoz generálhassunk egy $K_1, \ldots, K_r$ kulcssorozatot.

A kódoló E_k függvény a következőképpen működik. Legyen p a nyílt szövegtér $2t$ hosszúságú része. Kettévágjuk két t hosszúságú részre, azaz $p = (L_0, R_0)$, ahol L_0 a bal, R_0 a jobb oldali rész. Ekkor a sorozat

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus f_{K_i}(R_{i-1})), \quad 1 \leq i \leq r$$

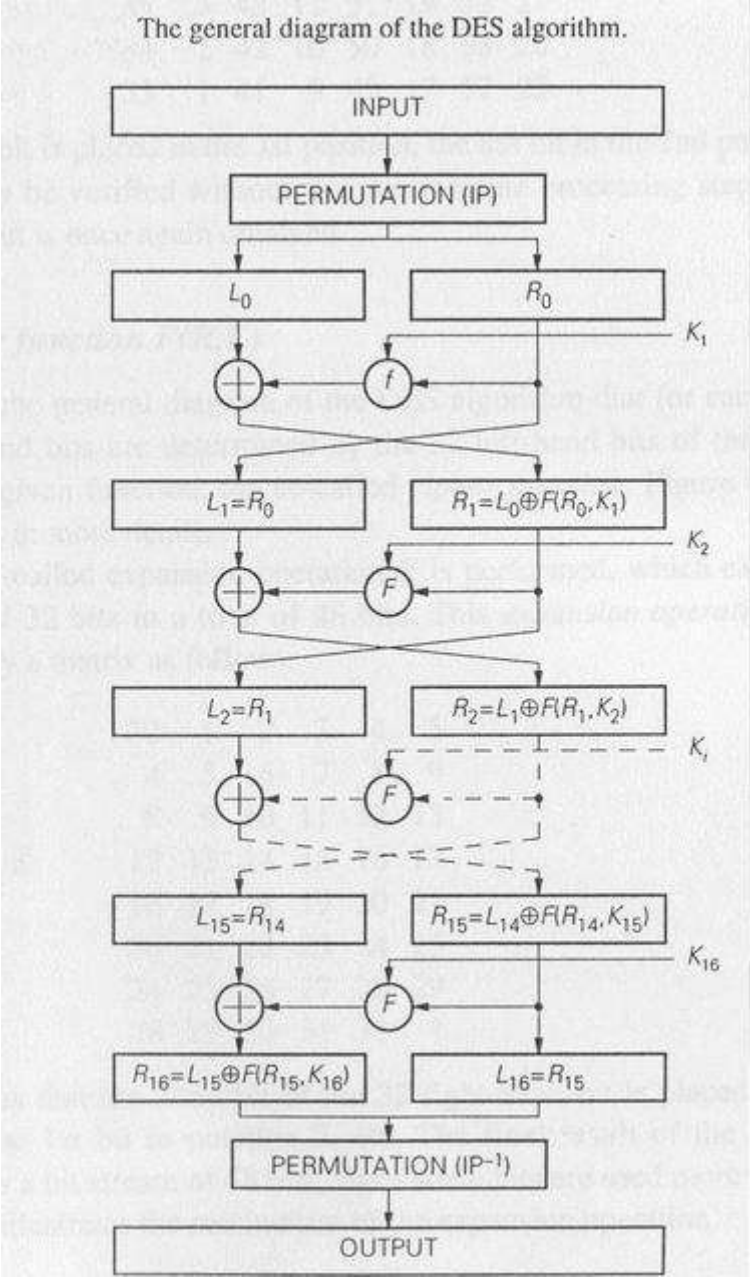
módon jön létre, és

$$E_k(L_0, R_0) = (R_r, L_r).$$

A lépésekben alkalmazott $\oplus$ művelet a szokásos XOR műveletet jelenti. A biztonság növelhető a körök számának növelésével. A dekódolás a következőképpen megy:

$$(R_{i-1}, L_{i-1}) = (L_i, R_i \oplus f_{K_i}(L_i)), \quad 1 \leq i \leq r.$$

Ezt használva r-szer a $K_r, \ldots, K_1$ kulcssorozattal visszakapjuk a (L_0, R_0) eredeti szöveget az (R_r, L_r)-ből.



2. DES algoritmus

A DES kulcsmérete 64 bit, azonban minden nyolcadikat kihagyjuk a felhasználásból. Az elhagyott biteket ellenőrzési célokra használják. Így a valódi kulcsméret 56 bit lesz csak. A DES kulcsok száma $2^{56} \approx 7.2 \cdot 1016$.

Például egy érvényes DES kulcs hexadecimális alakban a következő lehet:

$133457799BBCDFF1$

vagy bináris kifejtésben a következő táblázat mutatja

0	1	0	0
1	1	0	1
0	1	0	1
1	1	1	0
0	1	1	0
1	1	1	1
0	1	1	1
1	1	0	0

Az első lépésben a bemenet bitjeit jól összekeverjük, még utolsó lépésben ennek pont az inverzét alkalmazzuk. A DES titkosításban ezt inicializációs permutációnak (IP) nevezzük.

Ez egy bitpermutáció 64 bites vektorokhoz, azaz független a választott kulcstól. Az IP és inverze látható a következő ábrán. A táblázatot a következőképpen kell értelmezni: ha $p \in \{0,1\}^{64}$, $p = p_1p_2p_3 \dots p_{64}$, akkor $\mathrm{IP}(p) = p_{58}p_{50}p_{42} \dots p_7$.

IP								IP^{-1}							
58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

Ezután a 16 körös Festel kódolást alkalmazzuk a permutált nyílt szövegre. Végül a titkosított szöveget az IP inverzével kapjuk, azaz

$c = \mathrm{IP}^{-1}(R_{16}, L_{16})$.

3. A belső blokk kódolás

Az abc a $\{0,1\}$, a blokk hosszúság 32, és a kulcstér a $\{0,1\}^{48}$. Ekkor egy $f_K : \{0,1\}^{32} \to \{0,1\}^{32}$ kódoló függvényt használunk egy $K \in \{0,1\}^{48}$ kulccsal. Az $R \in \{0,1\}^{32}$ rész kibővítésre kerül egy $\mathrm{E} : \{0,1\}^{32} \to \{0,1\}^{48}$ függvénnyel. Ezen függvény megadása látható a következő ábrán:

E					R		
1	2	3	4	5	16	7	10
5	6	7	8	9	29	12	28
9	10	11	12	13	1	15	23
13	14	15	16	17	5	18	31
17	18	19	20	21	2	8	24
21	22	23	24	25	32	27	3
25	26	27	28	29	19	13	30
29	30	31	32	1	22	11	4

Ha $R = R_1 R_2 R_3 \dots R_{32}$, akkor $\mathrm{E}(R) = R_{32} R_1 R_2 \dots R_{32} R_1$. A következő lépés a $\mathrm{E}(R) \oplus K$ kiszámítása és az eredmény felosztása 8 blokkra, melyeket jelöljünk B_i-vel ($1 \leq i \leq 8$). Ezek 6 hosszúságúak, azaz

$\mathrm{E}(R) \oplus K = B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8$.

A következőkben az S_i

$S_i : \{0,1\}^6 \to \{0,1\}^4, \quad 1 \leq i \leq 8$

függvényeket használjuk (ezeket S-dobozoknak is nevezik). Ezen függvények használatával kapjuk a

$C = C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8$

sztringet, ahol $C_i = S_i(B_i)$. Ezek 32 hosszúságúak. Erre még alkalmazzuk a P permutációt. Így megkaptuk az $f_K(R)$-et.

4. S-dobozok

Ezek alkotják a DES algoritmus „szívét”, ugyanis (nagyon) nem lineárisak. Minden egyes S-doboz reprezentálható egy táblázattal, amely 4 sorból és 16 oszlopból áll. Minden egyes $B_i = b_1 b_2 b_3 b_4 b_5 b_6$ sztring esetén az $S_i(B_i)$ érték a következőképpen számolható ki:

Az az egész, amelynek bináris alakja b_1b_6 lesz a sorindex. A $b_2b_3b_4b_5$ bináris számnak megfelelő egész számot pedig oszlopindexként használjuk. A táblázatban megkeressük az megfelelő értéket, megadjuk a bináris alakját, és ha szükséges hozzárakunk további 0-kat, hogy a hossza 4 legyen. Így megkaptuk az -t.

Például határozzuk meg az $S_1(001011)$-t. Ekkor a sorindexet a 01, az oszlopindexet a 0101 adja meg. Ezek éppen az 1 és 5 egész számokat jelentik. Az első S-dobozban a megfelelő cellaérték a 2, melynek a bináris alakja az 10, azaz a 4 hosszúság miatt az $S_1(001011) = 0010$.

5. Kulcsok

Végül a kulcsok generálása maradt hátra. Legyen $k \in \{0,1\}^{64}$ egy DES kulcs. Ebből generáljuk a K_i, $1 \leq i \leq 16$ kulcssorozatot, melyek 48 hosszúak.

Definiáljuk a ν_i értékeket a következőképpen:

$$\nu_i = \begin{cases} 1, & \text{minden } i \in \{1,2,9,16\}\text{-ra,} \\ 2, & \text{egyébként.} \end{cases}$$

A következő algoritmussal ill. függvényekkel kapjuk a kulcsokat:

$\mathrm{PC1}\colon \{0,1\}^{64} \to \{0,1\}^{28} \times \{0,1\}^{28}$,
$\mathrm{PC2}\colon \{0,1\}^{28} \times \{0,1\}^{28} \to \{0,1\}^{48}$,

ahol a fenti függvényeket az alábbiak szerint adjuk meg.

PC1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Az algoritmus:

1. Legyen $(C_0, D_0) = \mathrm{PC1}(k)$.
2. Minden $1 \leq i \leq 16$-ra tegyük a kövekezőket:
   a. Legyen C_i az a sztring, melyet a C_{i-1}-ből ciklikus, ν_i pozícióval történő balra eltolással kapunk.
   b.

Legyen D_i az a sztring, melyet a D_{i-1}-ből ciklikus, ν_i pozícióval történő balra eltolással kapunk.

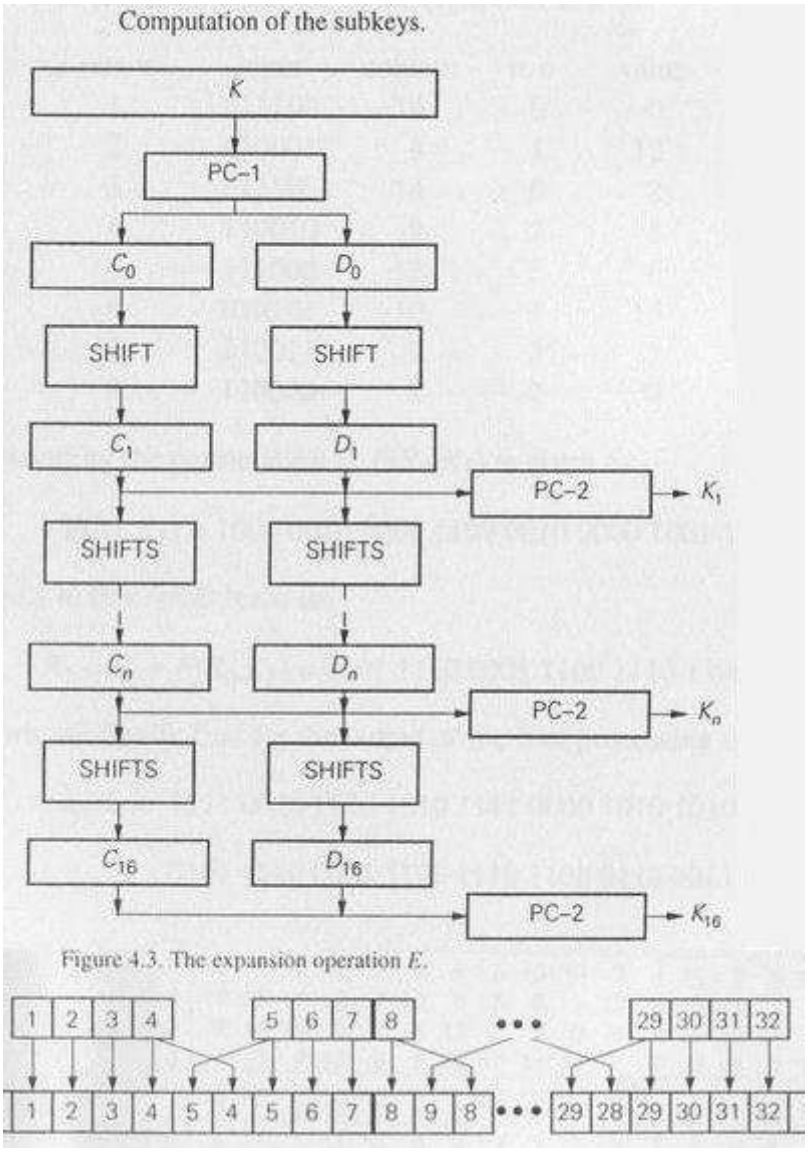
c.

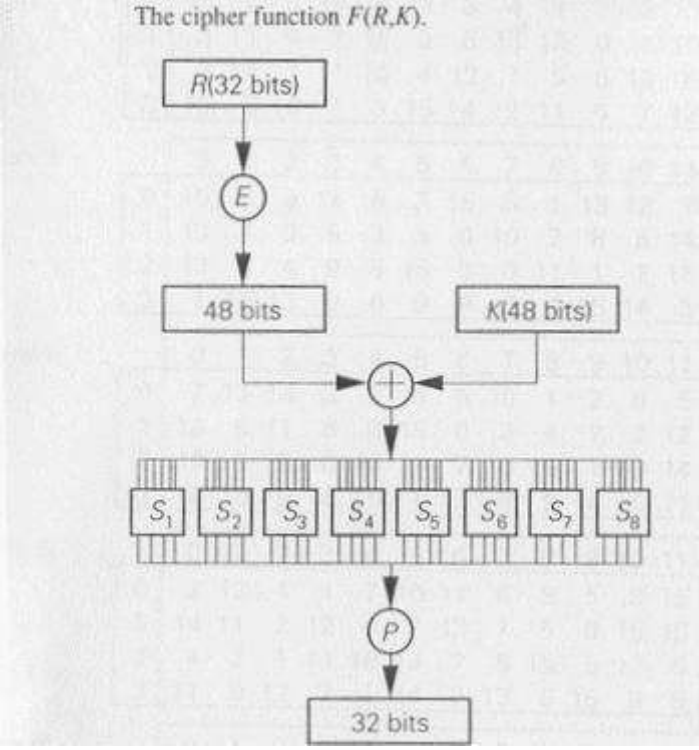
Határozzuk meg a $K_i = \mathrm{PC2}(C_i, D_i)$-t.

A PC1 függvény C és D, két 28 hosszúságú sztringet ad vissza a 64 hosszúságú kulcsból. Ez a táblázatból jól látható. Például, ha $k = k_1k_2k_3 \ldots k_{64}$, akkor $C = k_{57}k_{49} \ldots k_{36}$. A PC2 függvény egy (C, D) párból ad vissza egy 48 hosszúságú sztringet. Például $\mathrm{PC2}(b_1b_2 \ldots b_{56}) = b_{14}b_{17} \ldots b_{32}$.

A DES alapján történő kódolással kapott titkos szöveget a fordított sorrendben alkalmazott kulcssorozat kódolással kapjuk vissza.

A lépéseket a következő ábrákon követhetjük legjobban nyomon.





6. Egy DES példa

Készítsük el a $p = 0123456789ABCDEF$ nyílt szöveg kódolását a DES segítségével. Ennek bináris alakja az

0	0	0	0
1	0	0	0
0	0	0	1
1	0	0	1
0	0	1	0
1	0	1	0
0	0	1	1
1	0	1	1

Alkalmazzuk a korábban már látott IP permutációt, ekkor kapjuk

0	0	1	1
0	0	0	0
0	0	1	1
1	1	1	1
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0

azaz

$L_0 = 11001100000000001100110011111111,$
$R_0 = 11110000101010101111000010101010.$

Használjuk a korábbi példában megismert DES kulcsot, azaz az esetünkben $133457799BBCDFF1$-et, melynek a bináris alakja

0	1	0	0
1	1	0	1
0	1	0	1
1	1	1	0
0	1	1	0
1	1	1	1
0	1	1	1
1	1	0	0

Számoljuk ki az első kulcsot:

$C_0 = 11110000110011001010101111,$
$D_0 = 01010101011001100111100011111,$
$C_1 = 11100001100110010101011111,$
$D_1 = 10101010110011001111100011110.$

Ebből kapjuk a

$K_1 = 000110110000001011101111111111000111000001110010$

kulcsot. Ezt felhasználva kapjuk a

$\mathrm{E}(R_0) \oplus K_1 = 011000010001011110111010100001100110010100100111,$

és

$f_{K_1}(R_0) = 00000011010010111010100110111011,$

végül

$R_1 = 11001111010010110110010101000100.$

A többi forduló hasonlóan számolható ki.

A DES.exe program segítségével lépésről lépésre, a legapróbb részleteket is megértve, végig követhetjük a DES módszer működését.

7. A DES biztonsága

A feltalálása óta a DES biztonságát intenzíven vizsgálták. Speciális technikákkal támadták a DES-t, de máig sem sikerült olyan algoritmust találni, amely a kulcs ismerete nélkül feltöri a rendszert. Ugyanakkor, mivel a kulcstér nem túl nagy, a mai számítási kapacitások mellett az úgy nevezett „brute force” támadásokkal szemben tehetetlen a rendszer. Ezekben az esetekben „egyszerűen” végig nézzük a összes lehetséges kulcsot a megfejtés érdekében.

Nehezíthetjük a feltörést a DES módszer többszöri, egymás utáni alkalmazásával, az így kapott módszerek a TripleDES, illetve 3DES neveket viselik. Az elsőnél három különböző kulcsot alkalmaznak egymás után, még a másodiknál a három alkalmazott kulcsból kettő megegyezik.

Ebben a vonatkozásban fontos tudni, hogy a DES nem csoport. Ez azt jelenti, hogy ha rendelkezünk egy k_1 és egy k_2 kulccsal, akkor nincs olyan harmadik k_3 kulcs, hogy $\mathrm{DES}_{k_1} \circ \mathrm{DES}_{k_2} = \mathrm{DES}_{k_3}$.

Nyilvánvalóan fordított esetben a többszörös titkosítás nem növelné a biztonságot.

Végezetül meg kell említenünk, hogy nem csak a számítási kapacitások gyors növekedése dolgozik a DES ellen, hanem a növekvő tudásunk az elosztott rendszerekről is. Azokban az esetekben, amikor a feladat felbontható egymástól független részfeladatokra, a számítógépek összehangolt működése nagyon gyorsan eredményre vezethet.

6. fejezet - Az AES kriptográfiai rendszer

A DES algoritmus 1976-ban való bejelentése óta nagyot változott az informatika világa. Egyre növekedett a hálózati adatforgalom, javult a számítógépek gyorsasága és a szakemberek számára egyre nyilvánvalóbbá vált, hogy DES már nem nyújtja azt a biztonságot, amit az előző évtizedekben.

Most a 21. század elején úgy véljük, hogy egy kriptográfiai rendszer élettartama körülbelül 20 év és elvárjuk a titkosítási módszertől, hogy a beszüntetése után még jó ideig (10-50 év) őrizze titkunkat.

1996-ban az Amerikai Egyesült Államokban a National Institute for Standards and Technology (NIST) megkezdte egy új kriptográfiai algoritmus előkészítését. Az új algorimustól való elvárásokat 1997-ben publikálták és az AES (Advanced Encryption Standard) nevet kapta.

A következő elvárásokat fogalmazták meg az új rendszerrel szemben:

1. szimmetrikus kulcsú, blokkos algoritmust kell megvalósítani,
2. 128-as blokkokat kell használnia,
3. 128-192-256 bites kulcsmérettel kell dolgoznia,
4. gyorsabb legyen, mint a 3DES és nyújtson jobb védelmet,
5. a számítógép erőforrásait hatékonyan használja,
6. legyen eléggé flexibilis, alkalmazkodjon jól a különböző platformok lehetőségeihez.

A megmérettetésben nagy cégek (IBM, RSA Laboratories, Nippon Telegraph and Telephone Corporation), illetve kutatócsoportok is részt vettek. A végleges győztest három megtartott konferencia, szigorú vizsgálatok és törési kísérletek után hirdették ki 2000. október 3-án. A pályázat nyertese a Rijndael szimmetrikus kulcsú algoritmus lett, amely a tervezők után, Vincent Rijmen és Joan Daemen, kapta a nevét.

Vincent Rijmen

Joan Daemen

Az algoritmus kiválóan teljesítette a fentebb megfogalmazott elvárások mindegyikét. Érdemes megemlíteni, hogy az eljárás nem áll szabadalom alatt. Az algoritmus nagyon stabil, ellenáll a mai támadási módszereknek, az egyetlen lehetséges módszer az összes lehetőség végig próbálása (brute-force támadás).

1. Alapok

A Rijndael helyettesítő és lineáris transzformációkat ötvöző módszer. Hatalmas előnye, hogy a körkulcsok készítése gyors és a megvalósítás párhuzamosítható, ami nyilvánvaló előny a sebességet nézve. Az ismétlődő körfüggvények négy egymástól független transzformációból állnak, ezeket a továbbiakban rétegeknek nevezzük és az alábbiakban részletezzük.

1.

   A lineáris keverőréteg feladata a dobozok nagyfokú keveredésének megvalósítása. A MixColumns nevű réteg (oszlopszinten párhuzamosítható), még a ShiftRows nevű sorszinten párhuzamosítható.

2.

   A nem lineáris réteg egyetlen S-dobozt használ és a SubBytes réteg bájtszinten párhuzamosítható.

3.

   A kulcsfüggő réteg (key addition layer) teszi kulcsfüggővé a végső eredményét. A módszer egyszerű XOR műveletet használ és minden körben más-más Roundkey körkulcsot. Az AddRoundKey réteg bájtszinten párhuzamosítható. Megjegyezzük, hogy a többi réteg kulcsfüggetlen.

A körfüggvényeket az általunk továbbiakban tárgyalt AES–128 esetében 10-szer kell ismételni, az elsőből 9 kört kell elvégezni, még a másodikból egyet. A ki- és bemeneti adatokat egy State nevű struktúrában tároljuk.

1. Round (State, Roundkey)
   a. SubBytes (State)
   b. ShiftRows(State)
   c. MixColumns(State)
   d. AddRoundKey(State, RoundKey)
2. FinalRound (State, RoundKey)
   a. SubBytes(State)
   b. ShiftRows(State)
   c. AddRoundKey(State, RoundKey)

Az utolsó kör kicsit eltér az előzőektől, kimarad egy réteg. A zárójelekben láthatjuk, hogy mikor használjuk a kulcsot és mikor nem.

Az algoritmus megértéséhez szükségünk lesz néhány fogalomra, a szó 32 bitet jelent, a blokkméret (N_B) a blokkok méretét jelenti szavakban kifejezve, amely esetünkben $N_B = 4$. A kulcsméret (N_K) a kulcsméretet jelenti szavakban megadva, azaz esetünkben $N_K = 4$.

A körök száma függ a blokk- és kulcsmérettől is, esetünkben, ahogy fentebb is említettük ez 10 kört jelent $(N_R = 10)$.

2. A körfüggvény rétegei

2.1. State struktúra

A state-struktúrát kényelmesen ábrázolhatjuk egy 4x4-és négyzet segítségével, ahol is minden négyzet egy-egy bájtot jelent.

,0	$s_{0,1}$	$s_{0,2}$	$s($
,0	$s_{1,1}$	$s_{1,2}$	s_1
,0	$s_{2,1}$	$s_{2,2}$	s_2
,0	$s_{3,1}$	$s_{3,2}$	s_3

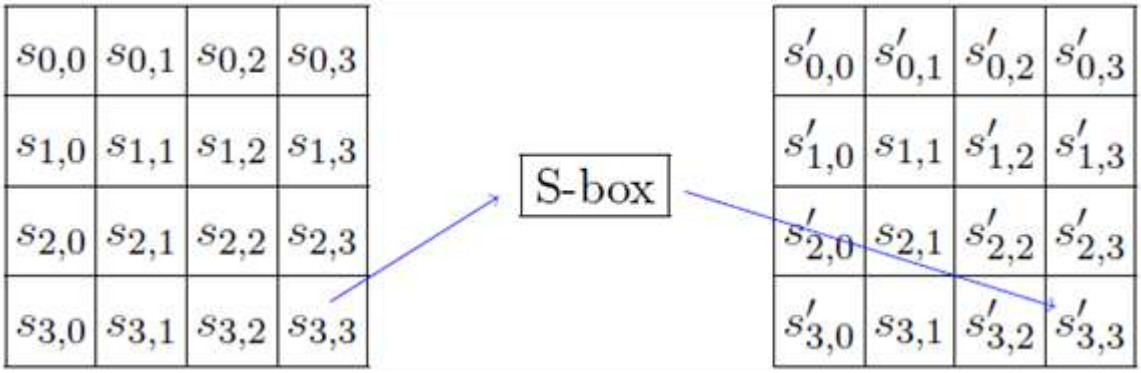
A state struktúra feltöltésekor, a kulcs és a titkosítandó anyag feltöltésekor fentről lefelé és balról jobbra kell haladni. Az state-struktúra oszlopvektorait tekinthetjük szavaknak.

2.2. SubBytes transzformáció

A SubBytes transzformáció egy nemlineáris, invertálható S-dobozt alkalmaz, minden bájt helyettesítése ugyanazzal az S-dobozzal történik. A következő összefüggés mutatja a műveletek elvégzésének szabályát, ahol b_i az adott bájt i-edik bitjét jelenti és c_i az i-edik bitjét a $C = 01100011$ kettes számrendszerbeli számnak, ahol $0 \leq i \leq 7$. A

$$b'_i = b_i \oplus b_{(i+4) \pmod 8} \oplus b_{(i+5) \pmod 8} \oplus b_{(i+6) \pmod 8} \oplus b_{(i+7) \pmod 8} \oplus C_i$$

egyenlőségben bitszintű műveletek vannak definiálva, ahol a bitek számozása a szokásos módon jobbról-balra történik. Minden esetben a vesszővel jelölt betű a megváltozott értéket jelzi. Az értékeket előre ki lehet számolni, a használatos S-doboz hexadecimális formában megtalálható a FIPS közleményében [4]. A kövekezőképpen tudjuk elképzelni a folyamatot:



Nyilvánvaló, hogy a fejtéshez is szükség van információra. Ezt az S-doboz inverze fogja szolgáltatni (a szakirodalomban InvSubBytes módon jelölik), amely egyszerűen meghatározható.

2.3. ShiftRows transzformáció

A ShiftRows a legegyszerűbb rétegfüggvény, mindössze annyit csinál, hogy a sorokat különböző mértékben eltolja jobbra. Esetünkben az első sort helyben hagyja, a másodikat eggyel, a harmadikat kettővel, még a negyediket hárommal tolja el. Ha visszagondolunk a klasszikus részekben mondottakra, akkor ezek ugyanazok a lépések, amelyeket a Playfair módszernél alkalmazunk, ha a választott betűpár egy sorban van. Nyilvánvalóan az InvShiftRows inverz művelet ugyanezek a lépéseket tartalmazza a másik irányban.

2.4. MixColumns transzformáció

Amennyire egyszerű volt a ShiftRows transzformáció, olyannyira bonyolult MixColumns transzformáció, aminek azért illik örülnünk, hisz egy kiváló szimmetrikus módszernél illik merész ötleteket felvonultatni.

Ahhoz, hogy megértsük ezen réteg lelkét, némi matematikai háttérrel fogunk megismerkedni. Az AES működése valójában bájt szintű műveleteken alapszik, amit az előző rétegeknél már láttunk. Legyenek egy B bájt bitjei rendre $B_7B_6B_5B_4B_3B_2B_1B_0$ és rendeljük ehhez hozzá a

$$b(x) = B_7x^7 + B_6x^6 + B_5x^5 + B_4x^4 + B_3x^3 + B_2x^2 + B_1x + B_0x^0$$

polinomot. Az ilyen polinomok együtthatói nullák és egyesek, így bármely 8 tagú bitsorozathoz egyértelműen rendelhetünk egy polinomot és viszont. Példaképpen az $x^7 + x + 1$ polinomhoz egyértelműen hozzárendelhetjük a $\{10000011\}$ bitsorozatot, illetve a $\{83\}$ hexadecimális számot.

Ilyen esetekben a polinomok közötti összeadást úgy végezzük, hogy az azonos hatványok együtthatóit $(\mathrm{mod}\ 2)$ összeadjuk. Például, ha az előző polinomhoz hozzáadjuk az $x^6 + x^4 + x^2 + x + 1$ polinomot, akkor az $x^7 + x^6 + x^4 + x^2$ polinomot kapjuk. A bináris jelölést használva az

$$\{10000011\} \oplus \{01010111\} = \{11010100\}$$

egyenlőséget kapjuk. Ugyanezt az eredményt akkor is, ha az összeadást bájt szinten végezzük hexadecimális számokkal $\{83\} \oplus \{57\} = \{d4\}$. Másképpen fogalmazva az AES algoritmus a $\mathbb{GF}(2^8)$ véges testet használja a MixColumns réteg definiálásakor.

Tekintsük most a szorzás műveletét, ehhez szükségünk lesz az AES algorimusnál használt $m(x) = x^8 + x^4 + x^3 + x + 1$ irreducibilis polinomra (lásd [4]), az általunk definiált hexadecimális írásmódban ez így írható $m(x) = \{01\}\{1B\}$. A továbbiakban a szorzás $(\bullet)$ művelete a két polinom szokásos szorzatának $m(x)$-el történő maradékát jelenti számunkra.

Az előbb részletezett módszert követve, hexadecimális írásmódot alkalmazva, azt kapjuk például, hogy $\{57\} \bullet \{83\} = \{c1\}$. Ennek az igazát úgy tudjuk bizonyítani, hogy elvégezzük először a szokásos szorzást. Természetesen ügyelünk arra, hogy az összeadásokat a fentebb említett módon végezzük, így

$$(x^6 + x^4 + x^2 + x + 1)(x^7 + x + 1) = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1.$$

A következő lépésben végezzük el az AES algoritmusban használatos irreducibilis polinommal való maradékos osztást, amelynél a megjósolt végeredményt kapjuk

$$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \pmod{x^8 + x^4 + x^3 + x + 1} = x^7 + x^6 + 1.$$

Tudjuk, hogy a modulus képzés kiváló arra, hogy a vektorokban lévő szabályosságot teljesen összezavarja. Itt sincs ez másképp, nincs olyan más egyszerű bináris művelet, amely az így kapott eredményt előállítaná, tehát remek az ötlet. Megjegyezzük, hogy az osztás után kapott végeredmény legfeljebb 7-ed fokú, így az együtthatók kiválóan ábrázolhatók egy bájton.

A $(\bullet)$ művelet asszociatív és a $\{01\}$ elem a struktúrában az egység. Tetszőleges 8-ad fokúnál kisebb fokszámú bináris polinom inverze meghatározható a kibővített Euklidészi algoritmus használatával.

Már csak egy lépés szükséges a tisztán látásunkhoz. Figyeljük meg, hogy mi változik, ha a $b(x)$ polinomot megszorozzuk az $x^1 = \{02\}$ polinommal. Először x-el szorzunk, így a

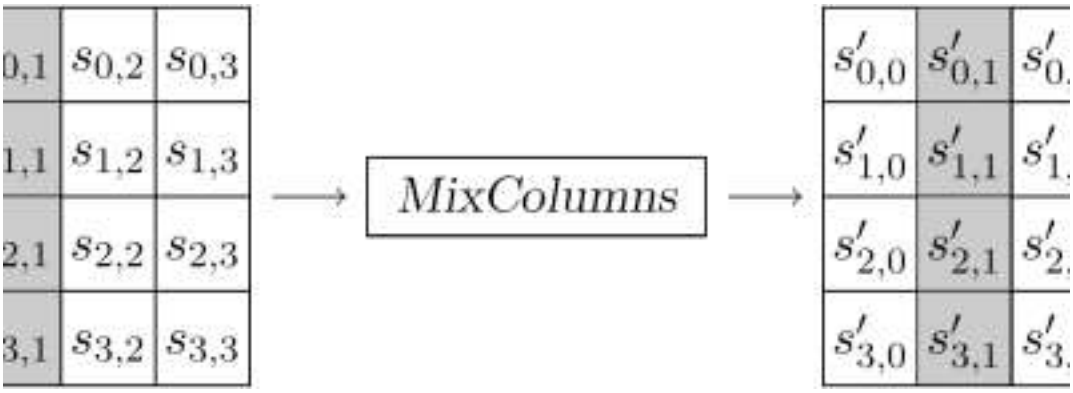
$$B_7x^8 + B_6x^7 + B_5x^6 + B_4x^5 + B_3x^4 + B_2x^3 + B_1x^2 + B_0x^1$$

polinomot kapjuk. A $(\bullet)$ művelet előírja $\mathrm{mod}\ m(x)$ modulusképpzést az így kapott polinommal. Ha $B_7 = 0$, akkor semmi teendőnk nincs, hisz a modulus képzés semmit nem változtat. Ha $B_7 = 1$ akkor az $m(x)$ polinomot vonjuk ki a kapott polinomból, avagy egyszerűbben XOR-oljuk össze $m(x)$-el. Azt láthatjuk, hogy a $\{02\}$ polinommal a szorzás egyszerű, az ábrázolt $b(x)$ polinom együtthatóit egy hellyel balra toljuk, és ha a

bájtból kilépő érték 1-es, akkor a számot XOR-oljuk $\{1b\}$-vel. Ezt a műveletet a Rijndael módszer dokumentációjában xtime() műveletnek nevezik. A magasabb hatványokkal való műveleteket eddigi tudásunk birtokában már kényelmesen el tudjuk végezni.

A MixColumns transzformáció a state-struktúra bájtjait alakítja át, minden esetben úgy, hogy a bájtokat egy előre meghatározott polinommal szorozza meg a fentebb ismertetett módon. Minden egyes új bájt függ az eredeti bájt oszlopban lévő összes bájttól. Nyilvánvaló, hogy akármilyen kis változás egy bájtban a kép teljes megváltozását vonja maga után. A következő összefüggések definiálják az új oszlopokat:

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$
$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$
$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$



A fentiekhez hasonló összefüggés definiálja az InvMixColumns utasítást, amelyet legális fejtés esetén kell használnunk. Ez a művelet, ahogy a nevéből is kiderül, a MixColumns művelet inverze. Az itt definiált $(\bullet)$ művelet az előzőekben ismertetett művelettel egyezik meg.

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$
$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$
$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

2.5. AddRoundKey transzformáció

Ez a réteg teszi kulcsfüggővé a titkosítási módszerünket. A művelet maga sokkal egyszerűbb, mint az előzőekben ismertetett MixColumns. A művelet egy egyszerű összeadás (XOR) az eddigiekben kialakult struktúra és a körkulcs bájtjai között.

Az általunk megadott titkos kulcsból egy hosszú, úgynevezett kiterjesztett kulcsot készít az algoritmus. A kiterjesztett kulcs elejére a titkos kulcs másolatát teszik, majd minden további szó a korábbi szavakból származtatható. Egy körkulcs N_B szót tartalmaz, azaz esetünkben 4-et, és $N_R + 1$ darab körkulcsra van szükségünk a megadott titkos kulcsot is beleszámítva. Az AES-128 esetében a körkulcs hossza $N_B(N_R + 1)$, azaz 44 szó. Minden egyes esetben a kiterjesztett kulcsot a state-struktúrával egyező méretű darabokra kell vágni. A hozzárendelésnél egy kicsit vigyázni kell, mert az első körkulcs, azaz az első N_B darab szó a nulladik körhöz, a második körkulcs, azaz a második N_B darab szó az első körhöz tartozik. Ezt értelemszerűen folytatva kapjuk a további megfeleléseket. A körkulcsokban a szavak rendre az első, második, harmadik, illetve negyedik oszlophoz tartoznak, azaz ilyen sorrendben kell elvégeznünk a XOR műveleteket. A lépéseket a következő egyenlőség írja le

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus w_{round*N_B+c},\ 0 \leq c < N_B,$$

ahol a round kifejezés az aktuális kör számát jelenti, a w vektorról pedig a következőkben írunk.

A körkulcs generálás teljes megértéshez szükségünk van két újabb függvényre. A SubWord függvény bemenete és kimenete egyaránt egy 4 bájtos szó, a bemeneti négy bájt mindegyikére a SubBytes S-dobozát alkalmazzuk. A RotWord függvény kimenete szintén egy négy bájtos szó, amely az (a,b,c,d) betűsorrendet (b,c,d,a) betűsorrenddé alakítja át.

Minden körhöz tartozik egy konstans (Rcon[i]), amelyet az előzőekben megismert írásmódot használva az $[x^{i-1},\{00\},\{00\},\{00\}]$ kifejezés határoz meg, ahol a hatványozás a fejezetben megismert módon történik. Megtartva a hexadecimális jelölést x-et $\{02\}$-vel jelöljük. Az i index kezdőértéke 1.

A kiterjesztett kulcs első N_K szava a titkos kulcsot tartalmazza, minden további szót, jelölje ezt w_i, az eggyel korábbi w_{i-1}, és az N_K-val korábbi w_{i-N_K} szavak között végzett XOR művelet adja.

Azon szavak esetén, amelyek pozíciója N_K többszöröse, a w_{i-1} és az Rcon[i] közötti XOR művelet adja azt az eredményt, amelyre ezután a SubWord és a SubBytes műveleteket alkalmazzuk.

A fentebb bevezetett w_i 4 bájtos szó, ahol $0 \leq i < N_B(N_R+1)$.

Most már minden részlettel tisztában vagyunk. A Rijndael titkosító algoritmusnál beállítunk egy titkosító kulcsot, majd elkészítjük a kiterjesztett kulcsot. Ezek után $N_R - 1$ körben a fejezet elején ismertetett Round függvényt alkalmazzuk, majd pedig FinalRound függvénnyel készítjük el a titkosított képet.

aes.msi

A fejtéskor a titkosító algorimus lépéseinek inverzeit használjuk. Azért, hogy világosan lássuk a sorrend helyességét, újra leírjuk a titkosításnál alkalmazott sorrendet is.

1. AddRoundKey (State, 0. körkulcs)
   (a) SubBytes (State)
   (b) ShiftRows(State)
   (c) MixColumns(State)
   (d) AddRoundKey(State, 1. körkulcs)

⋮

9. AddRoundKey (State, 8. körkulcs)
   (a) SubBytes (State)
   (b) ShiftRows(State)
   (c) MixColumns(State)
   (d) AddRoundKey(State, 9. körkulcs)
10. FinalRound (State, RoundKey)
   (a) SubBytes(State)
   (b) ShiftRows(State)
   (c) AddRoundKey(State, 10. körkulcs)

És itt van az inverz sorrend.

1. AddRoundKey (10. körkulcs)
   (a) InvShiftRows (State)

(b) InvSubBytes(State)

(c) AddRoundKey(State, 9. körkulcs)

(d) InvMixColumns(State)

⋮

10. FinalRound (State)

(a) InvShiftRows(State)

(b) InvSubBytes(State)

(c) AddRoundKey(State, 0. körkulcs)

A leírásból az is nyilvánvaló, hogy az AddRoundKey réteg önmaga inverze.

Az AES úgy tűnik elnyeri azt a helyet a titkosítás világában, amit neki szántak. Jól működik különböző platformokon és az általa nyújtott titkosság is eléri a kívánt színvonalat. Úgy tűnik jelenleg, hogy nem létezik jobb módszer a feltörésére, mint a nyers erő („brute-force”), azaz a lehetőségek szisztematikus átnézése.

3. Titkos kommunikáció

A jelen fejezetben megismert AES algoritmus egy olyan szimmetrikus algoritmus, amely tartalmaz egy kulcsfüggő részt, amely a biztonsága egyik záloga is. Ugyanakkor a kulcsokat meg kell osztani a résztvevő felekkel, amely nem mindig olyan egyszerű dolog.

Amennyiben fizikailag biztonságos csatornával van lehetőségünk dolgozni a dolog elég egyszerűnek mondható, hiszen csak a csatorna rongálásával tud a harmadik fél hozzáférni a titkos kulcsunkhoz. Ez a módszer kis távolságok esetén kiválóan megoldható, még nagy távolságok esetén nem érdemes ilyet választanunk, mert biztonsága nem garantálható és még drága is.

A biztosított csatornának nincs fizikai védelme, tehát a támadó csatlakozhat a csatornához és veszélyeztetheti az adatközlés biztonságát. A kommunikációhoz, ilyen esetekben különböző protokollokat használnak a felek, amely biztosítja adataink sérthetetlenségét.

Könnyen meg tudjuk mutatni, hogy n számú kommunikáló partner esetén $\frac{n(n-1)}{2}$ kulcsra van szükségünk, ha minden lehetséges párnak adni akarunk egy közös kulcsot. Ez nagy számú partner esetén elég sok kulcs generálását teszi szükségessé és szimmetrikus módszer használatakor mindenkinek mindenkivel meg kell állapodnia. A nyilvánvalóan bonyolult egyezkedéseket az aszimetrikus kulcsok felhasználása teszi szükségtelenné.

Érdemes megemlíteni, hogy előzetes kulcscsere nélkül is meg tudunk állapodni egy közös kulcsban, amelyre egy példa a háromutas kulcsforgalom. A folyamatot képzeljük el úgy, hogy két fél, Alice és Bob akar kommunikálni (az elnevezésben megtartjuk az angol nyelvű szakirodalom szokásos neveket). Az üzenetet egy ládában kívánják elküldeni egymásnak és mindketten rendelkeznek egy lakattal és természetesen egy hozzátartozó kulccsal is. Először Alice beteszi az üzenetet a ládába és bezárja a saját lakatjával, majd elküldi Bobnak. Bob is ráteszi a saját lakatját a dobozra, majd visszaküldi Alicenek. Alice leveszi a saját lakatját, majd visszaküldi a dobozt Bobnak. Végül Bob leveszi a saját lakatját is hozzáfér az üzenethez.

Az üzenet minden esetben biztonságosan lett elküldve, hiszen volt rajta lakat, ugyanakkor a két fél saját kulcsát sem kellett elküldeni. Egyetlen feltétel kellett, hogy teljesüljön a módszer alkalmazásánál, a titkosítási és a fejtési módszereknek felcserélhetőknek kellett lennie, azaz

$$E_k(D_k(T) = D_k(E_k(T).$$

4. Feladatok

1.

Határozzuk meg, hogy a {2A} és {75} hexadecimális számokhoz milyen polinomok tartoznak a fejezetben ismertetett reprezentációban.

2.

Határozzuk az xtime({57}) művelet értékét!

3.

Legyenek $a(x) = x^3 + x^2 + 1$ és $b(x) = x^6 + x^4 + x^3 + x$ adott polinomok. Határozzuk meg $a(x) + b(x)$ és $a(x) \bullet b(x) \pmod{m(x)}$ értékeket!

4.

Legyenek $s_{0,1} = 00111000$, $s_{1,1} = 10011000$ $s_{2,1} = 10011101$ és $s_{3,1} = 00000111$ adott feltöltései egy oszlopnak. Határozzuk meg a MixColumns művelet eredményét.

5.

Az AES módszerben használt (Rcon[i]), az előzőekben megismert írásmódot használva, $[x^{i-1}, \{00\}, \{00\}, \{00\}]$ módon írható, ahol a hatványozás a fejezetben megismert módon történik. Megtartva a hexadecimális jelölést x-et $\{02\}$-vel jelöljük. Határozzuk meg az értékeket az $i = 1$, $i = 2$ és $i = 3$ esetekben.

7. fejezet - Knapsack

Láttuk az előző rész tárgyalásakor, hogy a klasszikus rendszerek az ügyes, olykor szerencsés, fejtők előtt fejet hajtanak. Ha ismerjük a módszert és ismerjük a titkosítási kulcsot, akkor egyszerűen eljutunk a forrásszöveghez.

Tovább kellett folytatni tehát a kutatást olyan módszerek után, melyek ezektől sokkal nagyobb fejtörés elé állítják az illegális betolakodót.

Az 1970-es években Ralph Merkle, Whitfield Diffie és Martin Hellman egy új típusú kódot javasoltak, az úgynevezett nyilvános jelkulcsú titkosírást.

Ralph Merkle

Whitfield Diffie

Martin Hellman

Olyan titkosítást szerettek volna megvalósítani, melynél a titkosítási módszert publikussá is lehetne tenni, ugyanakkor nagyon nehéz visszafejteni. Persze nem lenne túl épületes a módszer, ha a legális felhasználónak is rengeteg munkájába kerülne a fejtés. Ezt úgy lehetne megoldani, hogy egy parányi információt eltitkolunk az avatatlan szemlélőtől, amely aztán bennünket a gyors megfejtéshez vezet.

A matematikában egy kicsit is jártas olvasó bizonyára rögtön fennakad a „nagyon nehéz” kifejezésen, ugyanis ez nem egy jól definiált fogalom. Ha precízebben akarunk gondolkodni a fogalomról, akkor el kell mélyednünk egy kicsit a bonyolultságelmélet problémáiban. Ebben a témában sok kiváló kiváló művet olvashatunk (lásd például [6], [10]).

A továbbiak megértéséhez egy rövid kitérőt teszünk, hogy alapvető benyomásaink legyenek arról, hogy mit is értünk a „könnyű” és „nehéz” kifejezéseken. Tetszőleges algoritmus időkomplexicitása a bemenő adatok hosszúságának egy függvénye. Akkor mondjuk, hogy az adott algoritmus $f(n)$ időkomplexicitású, ha tetszőleges n hosszúságú bemenet esetén az algoritmus legfeljebb $f(n)$ lépésben véget ér.

J. Edmonds és A. Cobham úgy látták, hogy az algoritmusoknak bonyolultság tekintetében két fő osztály van. Az egyik a „jó” algoritmus, amelynél az előző függvény polinomiális, a másik a „rossz” algoritmus, amelynél exponenciális.

Az egyik esetben tehát $f(n) c_1 n^s$ típusú, még a másik esetben $c_2 a^n$ alakú, ahol c_1, c_2, s, a az algoritmushoz rendelt konstansok.

(Itt jegyezzük meg, hogy egy olyan algoritmus, ahol $f(n) = 10^{300} n^{150}$, „jó” algoritmus a definíciónk szerint, de gyakorlatilag használhatatlan. Más esetben pedig, ha $f(n) = exp(10^{-500} n)$, akkor ez definíció szerint egy „rossz” algoritmus, bár nagyon sok n esetén jól működik.)

Jelöljük P-vel azon problémák osztályát, amelyek polinomiális idő alatt megoldhatók. Jelölje NP a nemdeterminisztikus polinomidejű problémákat, amelyekre az igaz, hogy egy adott n és B esetén az $f(n) < B$ egyenlőtlenség teljesülését polinomiális idő alatt ellenőrizni tudjuk.

Máig megoldatlan probléma, hogy a $P = NP$ egyenlőség igaz-e. Az nyilvánvaló, hogy P-beli problémák egyben NP-beliek is. A másik irányra az a sejtés, hogy nem igaz. Hallatlanul nehéz viszont bizonyítani, ugyanis az összes lehetséges algoritmust figyelembe kellene venni és megmutatni, hogy egyikük sem hatékony.

További bonyodalmat okoz, hogy azok a problémák, amelyek NP-beliek és úgy gondoljuk, hogy nincsenek benne P-ben, általában azonos nehézségűek.

Egy NP problémát NP-teljesnek mondunk, ha abból, hogy az megoldható polinomidő alatt, minden NP-beli probléma polinomidejű megoldhatósága következik. A szóbajöhető problémákról kiderült, hogy -teljes problémák, tehát nagyon távol vagyunk az említett probléma megoldásától. (Megjegyezzük, hogy a prímtényezőkre bontás nem -teljes.)

Megemlítenék néhány ilyen jellegű problémát.

1.

(Rendezési probléma) Egészek egy adott halmazát rendezzük növekvő sorrendbe.

2.

(Ládapakolási probléma) Adott különböző méretű tárgyak egy halmaza, helyezzük el őket a lehető legkevesebb rögzített méretű ládában.

3.

(Utazó ügynök probléma) Adottak bizonyos városok, amelyek mindegyikét pontosan egyszer kell felkeresni. Mi a legrövidebb út?

4.

(Hozzárendelési probléma) Adottak a kurzusok, tanárok és diákok adatai, készítsünk olyan órarendet, amelyben nincs ütközés.

Ezen megjegyzések után elképzeljük egy kriptográfiai rendszer tervezését.

Válasszunk egy „nehéz” problémát, azaz egy olyat amely nem P-beli. Vegyük ennek a P-beli problémának egy „könnyű” P_k alapproblémáját, ami jelentse azt, hogy megoldható lineáris idő alatt.

Ezek után valamilyen matematikai módszer segítségével toljuk el ezt a P_k problémát egy P'_k problémába, amely inkább P-re emlékeztet, mint P_k-ra.

Ezek után P'_k-t publikáljuk és titkos csapóajtóként elrejtjük a P_k problémába való visszajutás módszereit.

Így a legális felhasználónak egy „könnyű” problémát kell megoldania a fejtéshez, még az illegálisnak egy „nehezet”.

Jellemző, hogy a nyilvános rendszerek mélyen fekvő problémáiról nagyon keveset tudunk. Nagy valószínűséggel ezek a problémák NP-teljes vagy magasabb komplexicitásúak. Ilyen probléma például egy egész szám faktorizációja, prímség megállapítása, vagy bizonyos nagyságú prímek keresése.

1. Hátizsák probléma

Az első olyan probléma, amely alkalmas arra, hogy megvalósítsuk a nyilvánosság igényét, a Hátizsák probléma.

Maga a probléma a titkosítási lehetőségek nélkül is régen foglalkoztatja a matematikusokat. A kérdés az, hogy ha van egy hátizsákunk és egy csomó apróságunk, van-e módszer annak eldöntésére, hogy melyeket válasszunk ki azért, hogy a hátizsák tele legyen.

Első olvasáskor is érezhető, hogy ha hatalmas hátizsák van birtokunkban és rengeteg kisebb nagyobb aprósággal akarjuk telezsúfolni, akkor a próbálkozós módszer elég sok ideig fog tartani. Precízebben fogalmazva legyen adva egy $A := (a_1, a_2, \ldots, a_n)$ vektor, melynek elemei pozitív egészek, és egy pozitív k szám. Határozzuk meg azon a_i értékeket, melyeknek az összege k. A próbálkozások biztos módszerét követve 2^n lehetőséget végig próbálva biztosan eredményre jutunk. Ez 10 elemnél nem is tart túl sokáig, hiszen csak 2048 próbálkozásra van szükségünk. Ellenben, ha 300 elemet kell végig próbálni, akkor már egy „nehéz” problémával kell szembe néznünk.

Ha nem tévesztjük el célunkat, azaz továbbra is egy titkosítási módszer előállítása a cél, akkor a kérdés az, hogy van-e a Hátizsák problémának olyan változata, amikor a bepakolás egyszerű. El tudunk képzelni ilyen kellemes helyzetet.

Legyen ugyanis minden csomag olyan nagy, hogy a tőle kisebb összes csomag együttesen beleférjen, de ne töltse ki. Ekkor a pakolás egyszerű. Vegyük elő a zsákot és rakjuk bele a lehető legnagyobb csomagot. Nyilvánvaló az előzőek miatt, hogy ha ezt kihagyjuk, akkor az összes többi kicsi ekkora helyet már nem tölthet ki. A maradék helyre megint tegyük bele a lehető legnagyobb csomagot, amely lépés szükségszerűségét az előzőek indokolják. Ha folytatjuk ezeket a lépéseket, rövid időn belül célhoz érünk, azaz tele lesz a hátizsák vagy meggyőződünk róla, hogy nem lehetséges a pontos bepakolás.

1982-ben Ralph Merkle felállított egy nyilvános kulcsú titkosítást, amelyet a Hátizsák problémára épített. A kitaláló 1000 dollárban fogadott, hogy a kód feltörhetetlen.

A pontosabb tárgyaláshoz szükségünk lesz a szupernövekvő vektor definíciójára. A továbbiakban feltételezzük, hogy a betűk pozitív egész számokat jelölnek.

Az $A := (a_1, a_2, \ldots, a_n)$ vektor szupernövekvő, ha minden szám nagyobb az előzőek összegénél, azaz

$$a_j > \sum_{i=1}^{j-1} a_i \qquad j = 2,3,\ldots,n.$$

Ezek után lássuk az ötletet. Az egyszerűség kedvéért tegyük fel, hogy az AB betűpárt akarjuk titkosítani és a szupernövekvő vektorunk 10 elemű. Legyen adott a $C := (c_1, c_2, \ldots, c_{10})$ szupernövekvő vektor.

Feleltessük meg az A betűt a 00001, a B betűt a 00010 bitsorozatnak. Más betűk esetén hasonló kódolást végzünk, azaz a betű ABC-beli sorszámát kettes számrendszerben 5 biten ábrázoljuk. Mondhatjuk, hogy AB betűpárnak az $X_{AB} = (0,0,0,0,1,0,0,0,1,0)$ vektor felel meg.

Képezzük ezután a C és X_{AB} vektorok CX_{AB} skaláris szorzatát. Nyilvánvaló, hogy az így kapott összeghez a szupernövekvő vektornak csak azok a komponensei adnak járulékot, amelyek megfelelői a kódolt szövegben 1-gyel egyenlőek.

Így tehát egy egész számot kaptunk, amelyből a titkosított szöveget csak az tudja visszafejteni, aki meg tudja mondani, hogy ez a szám az A vektor mely komponenseinek összegeként áll elő. Ekkor fel tudunk írni egy X egyesekből és nullákból álló vektort, amelyben egyes szerepel egy adott helyen, ha a megfelelő helyen lévő A-beli elem benne van az összegben és nulla, ha nem.

Ezután már csak dekódolnunk kell a nullákból és egyesekből álló számsorozatot és a megfejtés kész.

Szupernövekvő vektor esetén ez nem jelent problémát. Az előzőekben említett hátizsákba való bepakolás szisztémáját követve általános esetben a következőképpen járhatunk el. Legyen adva egy $A := (a_1, a_2, \ldots, a_n)$ szupernövekvő vektor és tételezzük fel, hogy a kódolt szöveg (ami csupa egyesből és nullából áll) pontosan egy n dimenziós X sorvektornak felel meg. Az AX skaláris szorzat értékét jelöljük k-val. Ekkor a fejtő a következőképpen járhat el. Megvizsgálja, hogy a $k \geq a_n$ egyenlőtlenség teljesül-e. Ha igen, akkor az X vektor utolsó helyén egyes áll, ha nem, akkor nulla. Ezután k_1-et úgy definiáljuk, hogy

$$k_1 := \begin{cases} k & , \text{ha } k < a_n, \\ k - a_n & , \text{ha } k \geq a_n. \end{cases}$$

Hasonlóan folytatva az eljárást a_1-ig megkapjuk, hogy mely helyeken vannak egyesek és mely helyeken nullák. Ez az alak lesz a keresett betűknek a kódolt alakja. A dekódolás ezután egy egyszerű táblázat segítségével könnyen megvalósítható.

Jól látható, ha az A vektort közöljük a megfejtés nagyon egyszerű. A továbbiakban az egyszerűség kedvéért az (A, α) probléma megoldásáról beszélünk, ha egy α számot bontunk fel egy A vektorban szereplő komponensek összegére.

A probléma az, hogy a fejtés az illegális fejtőnek is nagyon könnyű, ami pedig ellentmond kitűzött céljainknak.

Meg kell tehát vizsgálni, hogy hogyan tudjuk „elrontani” az A vektort úgy, hogy ne látszódjon szupernövekvőnek, de mi vissza tudjuk belőle állítani az eredeti vektort. Így megvalósulna a nyilvános kulcsú kódolás gondolata, hisz egy tetszőleges vektor komponensei összegeként előállítani egy számot nagyon nehéz feladat, feltéve ha a komponensek elegendően nagyok és sokan vannak.

Ezek a nem pontosan körülírt fogalmak azt jelentik, hogy olyan vektort kell választanunk, amelyekből az összeg tagjainak a kiválasztása a fejtés szempontjából irreálisan hosszú ideig tart. Ami azt jelenti, hogy az illegális fejtőnek nagyon nehéz dolga van, a szupernövekvő vektor ismerője pedig könnyen megoldhatja az előző módon a fejtést.

Válasszunk egy olyan m természetes számot, melyre

$$m > \sum_{i=1}^{n} a_i.$$

Ez az m nyilvánvalóan sokkal nagyobb bármelyik a_i-nél, hisz az A vektor szupernövekvő volt. Legyen t olyan természetes szám, melyre $(t, m) = 1$. Az m számot modulusnak, a t-t szorzónak nevezzük.

A t választásából következik, hogy létezik egy t^{-1}-el jelölt természetes szám, melyre

$$tt^{-1} \equiv 1 \pmod{m}.$$

Ezen választások után képezzük a $ta_i,\ (i = 1,2,\ldots,n)$ szorzatokat és redukáljuk $(\mathrm{mod}\ m)$, így rendre $b_1, b_2, \ldots, b_n$ értékeket kapunk. Az így kapott $B = (b_1, b_2, \ldots, b_n)$ vektort publikáljuk titkosító kulcsként. A B vektor nem szupernövekvő, így hiába publikus a kulcs, a fejtés gondot okoz. Az A vektoron végzett műveletek sorozatát erős moduláris szorzásnak nevezzük m-re és t-re vonatkozóan. A t, t^{-1} és m értékek képezik a titkos csapóajtót, amiknek segítségével a legális felhasználó visszafejtheti a B vektorból az A vektort. Ugyanakkor meghatározhatja a titkosításból származó β összeg α „ősképét”, amit aztán A ismeretében és az említett algoritmus segítségével megfejthetünk. A csapóajtó használatát és a módszer jóságát a következő tétel bizonyítja.

7.1. Tétel. Tegyük fel, hogy $A = (a_1, a_2, \ldots, a_n)$ egy szupernövekvő vektor és a B vektort A-ból $u \neq 0$ és t^{-1}-re vonatkoztatott erős moduláris szorzással származtatjuk. Tegyük fel továbbá, hogy $(\bmod m)$, tetszőleges pozitív egész és

$$\alpha \equiv u\beta \pmod{m}, \quad 0 < \alpha < m.$$

Ekkor érvényesek a következő állítások:

- Az (A, α) hátizsák probléma lineáris idő alatt megoldható. Ha létezik megoldás, akkor az egyértelmű.
- A (B, β) hátizsák problémának legfeljebb egy megoldása van.
- Ha a (B, β) problémának létezik megoldása, akkor az megegyezik az (A, α) hátizsák probléma egyetlen megoldásával.

Bizonyítás. Az első rész nem szorul bizonyításra, hisz láttuk a megoldási módszer algoritmusát.

A harmadik rész bizonyításához tételezzük fel, hogy létezik egy n bites D vektor, amely megoldása a (B, β) hátizsák problémának, azaz $BD = \beta$. Következésképpen a

$$\alpha \equiv u\beta = uBD \equiv u(tA)D \equiv AD \pmod{m}.$$

Mivel m nagyobb A komponenseinek összegénél, ezért $AD < m$. Az $\alpha < m$ egyenlőtlenség is teljesül α definíciója miatt. Így nyilvánvalóan érvényes az $AD = \alpha$ egyenlőség, azaz D megegyezik az (A, α) hátizsákprobléma egyetlen megoldásával. Ami így a második rész érvényességét is igazolja. □

Ralph Merkle az 1000 dolláros fogadását elvesztette. Adi Shamir a kód egyik változatát azonnal feltörte, bár ez nem volt elég a fogadás megnyeréséhez.

Adi Shamir

Ernest Brickellnek azonban 1985-ben sikerült egy gyors algoritmust találnia a hátizsákprobléma megoldására, azaz sikerült feltörnie az előzőekben megismert titkosítási módszert.

8. fejezet - Az RSA titkosítási rendszer

A 70-es években egyre inkább foglakoztatta az informatikusokat a kulcsmegosztás problémája. Elkezdődött a számítógép hálózatok kialakulása és a legtovább látók látták, hogy a kulcsok megosztása lesz az egyik legégetőbb kérdés a jövő informatikájában.

A délibábnak tűnő vállalkozásban persze csak néhány eltökélt tudós vett rész, így Whitfield Diffie, Martin Hellman és kicsit később Ralph Merkle. Olyan függvényeket kerestek, amelyet ők egyirányú függvénynek neveztek, azaz olyanokat, amelyeknél az egyik irányban gyorsan tudunk számolni, a másik irányba pedig nincs esélyünk. Egy kicsit pontosabban, visszafelé csak valamiféle extra információ segítségével tudunk haladni. Valójában ötleteik az aszimetrikus titkosítás alapjait teremtette meg. Kidolgozták a Diffie-Hellman-Merkle-féle kulcsmegosztási rendszert, amely működött is, csak nem tökéletesen. Tovább folytatták kutatásaikat a Stanford Egyetemen, keresték azt a bizonyos egyirányú függvényt, amely valósággá változtatja az aszimmetrikus kódot. Eltökélt munkájukat leginkább Martin Hellman egy mondata jellemzi. „Isten megjutalmazza a bolondokat.”

A számelmélet egy nagyon híres művelője G. H. Hardy (1877–1947) így írt munkásságáról: „Soha nem csináltam semmi „hasznosat”. Az ész egyetlen felfedezése sem változtatott a világon, és nem valószínű, hogy valaha is változtatni fog rajta közvetve vagy közvetlenül, jól vagy rosszul, akár kis mértékben is. Részt vettem más matematikusok képzésében, ugyanolyan matematikusokéban, mint amilyen én vagyok, és munkájuk – vagy legalábbis amennyi ebből az én segítségemnek tulajdonítható – mind ez ideig épp olyan haszontalan volt, mint a sajátom. A matematikus életének értéke, bármiféle gyakorlati norma szerint ítéljük is meg, a nullával egyenlő; és mindenképpen jelentéktelen a matematikán kívül.”

G. H. Hardy

A kiváló matematikus sok megjegyzésével lehetne vitatkozni, de fél évszázad távlatából ez nem lenne túl bölcs dolog. Amiért mégis ide kívánkoztak ezek a sorok annak az az érdekes tény az oka, hogy a nyilvános kulcsú kódolás ötletének megvalósítása a számelmélethez, a „matematika királynőjéhez”, vezette a kutatókat, immáron gyakorlati hasznot húzva Hardy „haszontalan” tudományából.

1. RSA

A nyilvános kulcsú kódolás megvalósítása érdekében tovább folytak a kutatások. Az új ötlet a prímek világából származik.

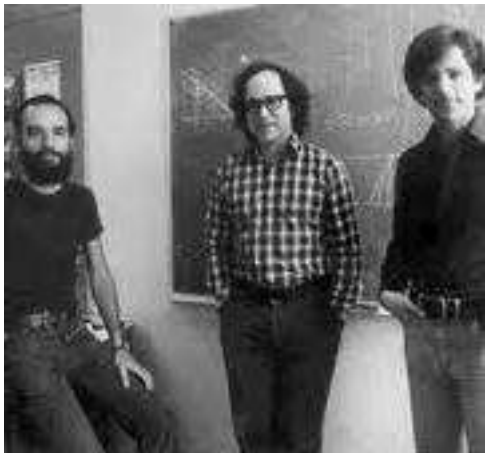
Úgy gondolhatnánk, hogy egy összetett szám prímtényezőkre bontása egyszerű dolog. Ez az elképzelés igaznak is bizonyul, ha a szám nem túlságosan nagy. A matematikai részben szereplő bonyolultságelméleti kitérő azonban rámutat, hogy ez az elképzelés nem tartható, ha a szám elegendően nagy, azaz nem ismerünk olyan gyors algoritmust, amely a prímtényezőkre bontást gyorsan végzi.

Ifj. Hendrik W. Lenstra ezt tréfásan úgy fogalmazta: „Tegyük fel, hogy a takarítónő tévedésből kidobta a p és q számokat, de a pq szorzat megmaradt. Hogyan nyerhetjük vissza a tényezőket? Csakis a matematika vereségeként érzékelhetjük, hogy ennek legreményteljesebb módja a szeméttelep átguberálása és memohipnotikus technikák alkalmazása.”

Ted Rivest, Adi Shamir és Leonard Adleman az MIT számítástechnikai laboratóriumában dolgozott és ismerték Diffie, Hellman és Merkle munkásságát és szerették volna megalkotni a mások által megálmodott egyirányú függvényt.

Egy húsvéti jól sikerült ünnepség után, 1977 áprilisában talált rá Rivest a megoldásra, aki munkatársaival együtt jelentette meg a cikket, amely új utakat nyitott a kriptográfiában. Neveik kezdőbetűiből a módszert RSA titkosítási rendszernek nevezzük. A kis Fermat-tétel segítségével sikerül eldugni a kulcsot az avatatlan szemek elől. A titkosítási módszert a következőkben részletezzük.

Ted Rivest, Adi Shamir és Leonard Adleman



Legyenek p és q különböző prímszámok, általában száz vagy több jegyű decimális számot választunk. Jelöljük a szokásos módon $\phi(n)$-nel az n-nél nem nagyobb, n-hez relatív prím pozitív egészek számát.

Ekkor, ha $n = pq$, akkor a

$$\phi(n) = (p-1)(q-1)$$

egyenlőség teljesül. Az n számot modulusnak nevezzük a továbbiakban. Válasszunk a következő lépésben egy $d > 1$ számot úgy, hogy $(d, \phi(n) = 1$ és határozzuk meg azt az e számot, melyre $1 < e < \phi(n)$ egyenlőtlenség teljesül és amely kielégíti az

$$ed \equiv 1 \pmod{\phi(n)}$$

kongruenciát.

Ezen értékek megválasztása után a titkosítandó szöveget kódoljuk és az így kapott T értéket titkosítjuk. A titkosított C szöveget a

$$C = T^e \pmod{n}$$

egyenlőség határozza meg.

(Megjegyezzük, hogy a kódolásnál leginkább decimális számokat használunk. A kapott számsorozatot blokkokra tagoljuk és ezeket külön-külön titkosítjuk. Általában olyan i hosszúságú blokkokat használunk, melyre teljesül a $10^{i-1} < n < 10^i$ egyenlőség.)

A titkosítás elvégzése után a fejtés problematikájával foglalkozunk. A következő tétel mutatja meg a fejtéshez vezető utat.

8.1. Tétel. A fenti jelöléseket használva, teljesül a

$$T \equiv C^d \pmod{n}$$

kongruencia. Ha a fejtés egyértelmű, akkor $T = C^d, \pmod{n}$.

A tételt Euler tételének felhasználásával bizonyítjuk.

Bizonyítás. Az előzőekben definiált d választása miatt létezik olyan pozitív egész j, melyre az

$$ed = j\phi(n) + 1$$

egyenlőség teljesül.

Tegyük fel először, hogy sem p, sem q nem osztja T-t. Euler tétele szerint érvényes a

$$T^{\phi(n)} \equiv 1 \pmod{n}$$

kongruencia, amelyből a

$$T^{ed-1} \equiv 1 \pmod{n}$$

kifejezés adódik. Így teljesülnek a

$$C^d \equiv (T^e)^d \equiv T \pmod{n}$$

kongruenciák.

Ha p és q közül pontosan az egyik, mondjuk p osztja T-t, akkor

$$T^{q-1} \equiv 1 \pmod{q}.$$

Ebből a

$$T^{\phi(n)} \equiv 1 \pmod{q}, \quad T^{j\phi(n)} \equiv 1 \pmod{q}, \quad T^{ed} \equiv T \pmod{q}$$

kongruenciák érvényessége következik. Mivel az utolsó kongruencia nyilvánvalóan $\pmod{p}$ is teljesül, így a Tétel állítását kapjuk.

Hasonlóan bizonyítható az az eset, amikor mind p, mint q osztható T-vel. □

Az általunk bizonyított tétel, tehát megmutatja, hogy ha titkosított szöveget d-edik hatványra emeljük, majd $\pmod{n}$ redukáljuk, akkor a megfejtendő szöveget kapjuk.

Megjegyezzük, hogy a rendszer tervezésénél szükségünk van arra, hogy eldöntsük e és $\phi(n)$ relatív prímségét. Ez Eulkideszi algoritmus használatával lineáris idő alatt sikerül. Másik feladatunk, meghatározni a d értékét úgy, hogy az

$$ed \equiv 1 \pmod{\phi(n)}$$

kongruencia teljesüljön. Könnyű látni, hogy ilyen d fejtési kitevő létezik. Mivel $(e, \phi(n)) = 1$, így léteznek olyan olyan x_0 és y_0 egészek, hogy

$$ex_0 + \phi(n)y_0 = 1.$$

Ebből az egyenletből azt kapjuk, hogy

$$ex_0 \equiv 1 \pmod{\phi(n)}$$

azaz x_0 éppen az általunk keresett d értékének felel meg.

Egy egyszerű példán megmutatjuk hogyan is működik az RSA. Válassza Alice a $p = 11$ és $q = 23$ prímszámokat, ekkor a modulusa $n = 253$-nak adódik, esetünkben $\phi(n) = (p-1)(q-1) = 220$. Legyen $e = 17$ a titkosítási kitevő, amiből meghatározható a $d = 13$ fejtési kitevő.

Bob hasonló választásai a következők $p = 13, q = 19$, így $n = 247$, illetve $\phi(n) = 216, e = 65$ és $d = 113$.

Képzeljük el, hogy Alice el szeretne küldeni egy üzenetet, jelen esetben ez a TITOK szó, Bobnak. Alkalmazza minden betűre az RSA ismert módszerét, azaz meghatározza a

$$C = (T^e, (\mathrm{mod}\ n))$$

kifejezés értékét. A T betű ASCII kódja 84, Bob publikus titkosító kulcsa 65, még modulusa 247, ami szintén publikus. Így Alice kiszámítja a

$$84^{65} \equiv 145 \pmod{247}$$

értéket, és T titkosított képe 145-nek adódik. Hasonlóan járunk el minden betű esetén. Az eredményeket a következő táblázat részletezi.

ASCII kód
84
73
84
79
75

Most, hogy a módszer bemutattuk, felmerül a kérdés, hogy mit publikálunk belőle és mi az, amit titokban kell tartanunk. Az n modulust és az e titkosítási kitevőt publikálhatjuk ennél a rendszernél. A $p, q, \varphi(n)$ és d számok képezik a titkos csapóajtót, amely azt jelenti, hogy bármelyiknek az ismerete feltöri a módszert.

Az RSA algoritmust szabadalom védte az USA-ban egészen 2000-ig. Ma már bárki készíthet licenszdíj fizetése nélkül RSA algoritmust használó hardver, illetve szoftvereszközt. Philiph R. Zimmermann által kifejlesztett PGP (Pretty Good Privacy) mindenki által használható titkosságot biztosító eszköz, amely felhasználja az RSA által nyújtott lehetőségeket (lásd [17], http://www.pgpi.org/).

Az RSA biztonságosan megvalósítja Diffie és Hellman álmát a kulcscserét is, mivel az algoritmusban e és d szerepe felcserélhető. Az RSA napjaink legismertebb aszimmetrikus kriptográfiai módszere, amely a DES-hez hasonlóan blokkos titkosító.

Gyakorlati alkalmazásra jelenleg az $1024 - 3072$ bites modulusokat tekintjük biztonságosnak. Ha az RSA kulcsainak hosszáról beszélünk, akkor ez alatt mindig n hosszát értjük. A biztonság érdekében ajánlott közel azonos méretű prímszámokból előállítani a kulcsot. További, gyakorlati szempontból lényeges dolgot a következő fejezetben tárgyalunk.

Lényeges megjegyezni, hogy az RSA feltörhető, amennyiben az n számot faktoraira tudjuk bontani. Ezt természetesen meg is tudjuk tenni, csak elegendő idő és számítási kapacitás kell hozzá, ami a jelenlegi matematikai ismereteink és számítási kapacitásaink mellett olyan óriási igény, ami nem teszi lehetővé ésszerű időkorlátok mellett a fejtést.

Érdekességként említjük meg, hogy Az 1977-ben az RSA bejelentése után Martin Gardner, amerikai matematikus és népszerű tudományt népszerűsítő művek szerzője, a Scientific American című folyóirat általa vezetett Matematikai játékok rovatában közzétette „Egy új-fajta sifrírozás, amelynek feltöréséhez évmilliók kellenek” című cikket.

Martin Gardner

Ebben elmagyarázta az olvasóknak a nyilvános kulcsú titkosírás működését és közölt egy n modulust, amellyel egy szöveget titkosított. Esetében n =114 381 625 757 888 867 669 235 779 976 146 612 010 218 296 721 242 362 562 561 842 935 706 935 245 733 897 830 597 123 563 958 705 058 989 075 147 599 290 026 879 543 541.

A feladat az volt, hogy az olvasók faktorizálják n-et és fejtsék meg a szöveget. Száz dollár díjat ajánlott az első megfejtőnek. Gardner azt ajánlotta, hogy az RSA megértése érdekében forduljanak az MIT számítástechnikai

laboratóriumához. El tudjuk képzelni Rivest, Shamir és Adleman meglepetését, amikor több mint háromezer levél érkezett hozzájuk.

Gardner feladatát csak tizenhét év múlva oldották meg. 1994. április 26-án egy hatszáz önkéntesből álló csoport bejelentette, hogy az N faktorai a következők, $q =$ 3 490 529 510 847 650 949 147 849 619 903 898 133 417 764 638 493 387 843 990 820 577, és $p =$ 32 769 132 993 266 709 549 961 988 190 834 461 413 177 642 967 992 942 539 798 299 533.

A történet teljességéhez hozzátartozik, hogy a megfejtett szöveg a következő volt: „The magic words are squeamish ossifrage.” (A varázsszó finnyás fakó-keselyű.)

A faktorizációs feladatot megosztották a számítógéphálózaton, minden szabadkapacitást kihasználva. Megjegyezzük, hogy a Mersenne prímeket ma is hasonló módon keresik a kíváncsiak.

A 17 év elég rövid időnek tűnik, de látnunk kell, hogy Gardner problémájában csak egy 10^{129} nagyságrendű modulust használtunk, ami sokkal kisebb a mostanában ajánlott modulusoknál, amelyeknél milliárd években kell gondolkodnunk.

Természetesen ez csak egy kedves történet, nem bankokról vagy hadititkokról van benne szó, de jól illusztrálja azt a megjegyzésünket, hogy mit is jelent az időkorlát.

A betolakodónak az RSA feltöréshez szüksége lenne egy gyors faktorizáló eljárásra. Ez egyelőre nem áll rendelkezésünkre. A módszer jól működik, bár elég bizonytalan lábakon áll abban az értelemben, hogy nincs bizonyítva, hogy nem létezik polinomiális idejű faktorizáló algoritmus. Sőt az sincs bizonyítva, hogy nincs olyan módszer, amely feltöri az RSA-t, de nem faktorizáció segítségével.

Az hogy a titkosítási és fejtési kulcs felcserélhető, lehetővé teszi, hogy a Diffie és Hellman által kigondolt módszert realizáljuk. Tegyük fel, hogy Alice és Bob akar közös kulcsot választani, ehhez nyilvánosan megegyeznek egy n modulus és egy g generátor elem választásában, amelyek 195-512 bit hosszúak. A generátor elem fogalom azt jelenti, hogy az n-nél kisebb számoknak elő kell állniuk $g^x \pmod n$ alakban. Ezután mindketten választanak egy-egy $n-1$-től kisebb véletlen számot, amelyeket titokban tartanak, legyenek ezek x, y. Ezután a következő lépések történnek:

1. Alice elküldi Bobnak a $k_1 = (g^x, (\mathrm{mod}\ n))$ értéket,
2. Bob válaszában a $k_2 = (g^y, (\mathrm{mod}\ n))$-t küldi vissza,
3. Alice kiszámolja $k_3 = (g^y, (\mathrm{mod}\ n))^x \pmod n$,
4. Bob kiszámolja $k_4 = (g^x, (\mathrm{mod}\ n))^y \pmod n$,
5. A közös kulcsuk a $(g^{xy}, (\mathrm{mod}\ n))$.

Tudjuk, hogy az $a^x = y$ egyenletet a valós számok körében a logarimus függvény felhasználásával meg tudjuk oldani. Egészen más a helyzet az RSA bemutatásánál megismert moduláris hatványozásnál. A művelet elvégzése után csekély esélyünk van az alapban lévő szám megtalálására.

Általánosabban megfogalmazva a fenti okfejtést a következőket mondjuk. Legyen g és y egy-egy eleme a G véges csoportnak. Ekkor bármely olyan $x \in G$ elemet, melyre érvényes az $g^x = y$ egyenlőség, az y diszkrét logaritmusának nevezzük a g alapra nézve. Nyilvánvaló, hogy minden $y \in G$ elemnek akkor és csak akkor van

diszkrét logaritmusa a g alapra nézve, ha G ciklikus csoport és g egy generátor eleme. A diszkrét logaritmus probléma az előzőekben ismertetett NP osztályhoz tartozik.

Az előzőekben megismert Diffie-Hellman módszer egy módosított változata az ElGamal módszer, melyet 1984-ben Taher Elgmal egyiptomi kriptográfus publikált.

Legyen q és az $F^*(q)$ g generátora minden fél számára ismert. Ekkor minden A felhasználó titkosan választ egy $0 < m_A < q - 1$ egészt és publikálja g^{m_A}-t (ami, mint tudjuk egy eleme $F(q)$-nak).

Amennyiben egy w üzenetet kívánunk küldeni A-nak, azt a következő formában küldjük el (g^k, wg^{km_A}), ahol k egy tetszőleges, a küldő által választott, pozitív egész. Nyilvánvaló, hogy a diszkrét logaritmus probléma nehézsége miatt az illegális betolakodó nem juthat értékes információhoz. Az üzenetet megkapó A, viszont ki tudja számolni g^{-km_A} értéket, amelynek felhasználásával w könnyedén adódik.

Az RSA felfedezése komoly fegyvertény, az emberi gondolkodás és közös munka egy szép eredménye. Ugyanakkor történetében is érdekes dolog. A teljesség kedvéért megemlítenénk, hogy a brit kormány szerint a nyilvános kulcsú kriptográfiát Cheltenhamben, a Government Communications Headquartersben (CGCH), a második világháború után létrehozott, szigorúan titkos intézményben találták ki először.

Utólag tudjuk, hogy brit tudósok, James Ellis, Clifford Cocks és Malcolm Williamson 1975-re a nyilvános kulcsú kriptográfia összes alapvető tételét kidolgozta, de hallgatniuk kellett róla. A nyilvános, feltáró előadásra csak 1997-ben kerülhetett sor. A feltalálók közül csak ketten vehettek részt az előadáson, mert James Ellis egy hónappal előtte meghalt.

Az események jól szemléltetik, hogy a tudományok egy nagyon egzotikus határán járunk, ahol az új felfedezéseket igyekszenek titokban tartani, mert a titkos tudás lépéselőnyt jelent sok szempontból az adott kormány számára. Az emberi sorsok, sajnos ezekben az esetekben másodlagos szemponttá válnak.

2. Gyakorlati megjegyzések

Először is szembe kell néznünk azzal, hogy 100 jegyű prímeket kell keresnünk, ami a keresés és a prímség eldöntésének problematikáját is jelenti. A nagy prímek keresése jelenleg is folyik és az is lehetséges, hogy nem is publikálják a titkosítás miatt.

A keresés úgy folyik, hogy véletlenszerűen választunk egy megfelelő méretű (a jelenlegi problémánál 100 jegyű) páratlan számot, majd valamely prímteszt segítségével, amelyeket a következő részben ismertetünk, eldöntjük, hogy az illető természetes szám prím-e vagy sem. Nemleges válasz esetén a rákövetkező páratlan szám vizsgálatával foglalkozunk.

A Prímszám tétel szerint körülbelül

$$10^{100}/\ln 10^{100} - 10^{99}/\ln 10^{99}$$

100 jegyű prím van. Ebből az következik, hogy egy tetszőlegesen választott páratlan szám esetén a sikeres teszt megvalósulásának valószínűsége 0,00868.

A következő probléma a d kiválasztása. Amikor p és q kiválasztása megtörtént a d szám kiválasztása következik. Fontos, hogy d ne legyen túl kicsi, hisz akkor könnyen megtalálható és ez a feltöréshez vezet. A kiválasztott d számot az Euklideszi algoritmussal teszteljük. Amennyiben jól választottunk d kielégíti a $(d, \phi(n)) = 1$ feltételt, akkor megtaláltuk a megfelelőt és az Euklideszi algoritmus egyenleteiből a szükséges e szám rögtön leolvasható.

Mind a titkosításhoz, mind a fejtéshez szükségünk van az $(a^r \pmod n)$ típusú kifejezés meghatározására. Ezt sokkal gyorsabban elvégezhetjük, ha az a önmagával való beszorzása és az utána következő redukálás helyett az úgynevezett szukcesszív négyzetreemelések módszerét követjük és minden művelet után a kapott számot redukáljuk $\mathrm{modulo}\, n$.

Ezt úgy végezzük, hogy először tekintjük r binér reprezentációját, esetünkben

$$r = \sum_{j=0}^{k} x_j 2^j, \quad x_j = 0{,}1; \quad k = [log_2 r] + 1.$$

Ismételt négyzetreemelések elvégzésével könnyen meghatározhatjuk az

$$(a^{2^j} \pmod{n}), \quad 0 \leq j \leq k$$

értékeket. Ebből a számunkra szükséges $(a^r \bmod n)$ kifejezés könnyen kiszámítható. A számítás elvégzéséhez legfeljebb $k-1$ szorzás és redukció elvégzése szükséges.

Lássunk egy példát a szukcesszív négyzetreemelésre. Határozzuk meg a $7^{83} \pmod{61}$ kifejezés értékét. Fermat tételéből következik, hogy

$$7^{60} \equiv 1 \pmod{61}.$$

Így számunkra elegendő kiszámítani a $7^{23} \pmod{61}$ kifejezés értékét. A szukcesszív négyzetreemeléshez előre legyártjuk a 7 azon hatványait, ahol a kitevő 2^j alakú és a végeredmény $\pmod{61}$ vesszük. Az eredményeket egy táblázatban összegezzük:

0	1	2	3
7	49	22	57

Ekkor a kívánt eredményt a következő egyszerű számítással kapjuk meg, amennyiben tudjuk, hogy a 23 kettes számrendszerbeli alakja 10111

$$7^{83} \pmod{61} \equiv (16(22(49 \cdot 7)) \pmod{61}) = 17.$$

Nézzük meg hogyan működik ez titkosítás közben. Kódoljuk a betűket a sorszámukkal, eltérően az előzőektől, ahol ASCII kódot használtunk. Titkosítsuk az SA betűpárt, amelynek az 1901 szám felel meg, illetve az UN betűpárt, melynek a 2114 felel meg és tegyük fel, hogy a titkosítási kitevő 17. A következő táblázat mutatja a szukcesszív négyzetreemelés egymás után következő lépéseit:

	1901	21
	582	16
	418	17
	25	22
3	625	4
7	1281	16

További odafigyelés szükséges, ha igényes, nehezen fejthető rendszert akarunk megvalósítani. El kell kerülnünk, hogy p és q közel legyen egymáshoz. Ha ugyanis p és q közeliek, akkor $(p-q)/2$ kicsi és $(p+q)/2$ nem sokkal nagyobb, mint $\sqrt{n}$. Ráadásul a

$$\frac{(p+q)^2}{4} - n = \frac{(p-q)^2}{4}$$

egyenlőség baloldala teljes négyzet. Ennek segítségével n faktorizációját úgy valósíthatjuk meg, ha olyan x-eket tesztelünk amelyekre $x > \sqrt{n}$ és az eljárást addig folytatjuk, amíg $x^2 - n$ nem lesz teljes négyzet. Ha ezt a teljes négyzetet y-nak nevezzük, akkor $p = x + y$ és $q = x - y$ egyenlőségek megadják a keresett faktorizációt.

A tervezésnél figyelnünk kell $\phi(n)$ viselkedésére is. Ha ugyanis $p-1$ és $q-1$ legnagyobb közös osztója nagy, akkor legkisebb közös többszörösük, jelöljük ezt u-val, kicsi $\phi(n)$-hez képest. Ekkor e bármely inverze $\text{modulo}\, u$ fejtési kitevőnek minősül. Ebben az esetben d megtalálása sokkal könnyebb, így a rendszer megalkotásánál ügyelnünk kell arra, hogy $(p-1, q-1)$ ne legyen túl nagy.

Az előzőekben említett problémák elkerülése véget általában úgynevezett erős prímeket használunk melyeknek a tulajdonságai a következők:

1. a választott p prím nagy, legalább 400-500 bit hosszú
2. $p-1$ legnagyobb prímosztója nagy,
3. $p-2$ legnagyobb prímosztója nagy,
4. $p+1$ legnagyobb prímosztója nagy.

Ugyanakkor R. Rivest és R. Silverman kutatásai [16] azt bizonyítják, hogy bizonyos új típusú faktorizációs eljárások (Lenstra elliptikus görbén alapuló eljárása) erős prímek esetén is hatásosak lehetnek. Tehát az erős prímek sem oldanak meg minden problémát, de ennek ellenére az RSA megbízható titkosságot biztosít napjainkban.

A fejezetben gyakorlati kérdéseket tárgyalunk, ejtsünk néhány szót a mindennapi gyakorlatról is. Bár a szukcesszív négyzetreemelés sokat gyorsít a kívánt matematikai műveletek végrehajtásán, az RSA gyorsasága továbbra sem megfelelő a hétköznapokban.

Éppen ezért a gyakorlati megvalósítások során nem szokták az egész üzenetet nyilvános kulcsú algoritmussal kódolni, főleg akkor nem, ha az üzenet hosszú. Hanem hagyományos szimmetrikus algoritmusokból, amelyek sok százszor gyorsabbak, mint az RSA és nyilvános kulcsú megoldásokból álló, úgynevezett hibrid kriptorendszereket alkalmaznak. A szokásos eljárás az, hogy az üzenetet egy gyorsabb titkos kulcsú algoritmussal, az ehhez használt, véletlenszerűen generált, kulcsot pedig a nyilvános kulcsú módszerrel titkosítják, és a kettőt együtt küldik el. Az ilyen alkalmankénti, egyszer használt kulcsot viszonykulcsnak (session key) nevezzük.

Természetesen a nyilvános kulcsú algoritmus ebben az esetben csak a kulcsot védi, tehát a kulcselosztásban segít. Ezért a hibrid rendszer megvalósításakor fontos figyelni a használt szimmetrikus algoritmusra, mert ha ez törhető, akkor hiába védtük erős módszerrel a kulcsunkat.

3. Digitális aláírás

A digitális aláírás egy nagyon gyakran használt fogalom mostanában, ugyanakkor kevesen értik, hogy pontosan mit is rejt ez a fogalom. A digitális aláírást többek közt azzal a céllal hozták létre, hogy kiváltsa a hagyományos kézzel írott aláírásokat, ugyanakkor megfeleljen a mai kor informatikai követelményeinek. Maga a digitális aláírás egy szám, ami erősen függ az aláíró fél privát kulcsától (ami szintén egy szám), az aláírandó üzenettől, valamint egyéb, nyilvános paraméterektől. Fontos szempont, hogy egy digitális aláírás verifikálható legyen, azaz egy elfogulatlan harmadik fél az aláíró fél privát kulcsa nélkül is kétségtelenül igazolhassa, hogy az aláírást valóban az adott entitás készítette.

Az aszimetrikus kriptográfiai módszerek kiválóan alkalmazhatók digitális aláírás előállítására. Ebben az esetben minden félnek van egy privát és egy nyilvános kulcsa. Az aláíró fél mindvégig titokban tartja a privát kulcsát, azt soha, semmilyen körülmények közt nem hozza nyilvánosságra a saját biztonsága érdekében. Ezzel ellentétben, a nyilvános kulcsát közzéteheti bárki más számára. Sok esetben szükség is van erre, mert a nyilvános kulcs segítségével validálható egy digitális aláírás egy adott üzenetre és aláíró személyre vonatkozóan.

Rendkívül fontos szempont, hogy amennyiben A megszerzi B digitális aláírását egy adott üzenetre vonatkozóan, akkor ezt felhasználva A ne legyen képes egyéb üzeneteket B aláírásával ellátni.

A digitális aláírásnak manapság számos alkalmazási területe létezik.

1. Az Adatintegritás (annak biztosítása, hogy az adatok nem lettek megváltoztatva megbízhatatlan felek által),
2. Adatok forrásának verifikációja (annak bizonyítása, hogy az adatok valóban onnan származnak ahonnan kell),
3. Megtagadás elleni védelem (annak biztosítása, hogy egy adott fél ne tudja letagadni az általa generált aláírásokat).

A digitális aláírási sémák előállításánál felhasználhatjuk a most megismert RSA módszert, a diszkrét logaritmuson alapuló technikákat vagy a később megismerendő elliptikus görbéket.

Lássunk most egy RSA módszeren alapuló digitális aláírást. Ez a módszer igen egyszerű, messze nem a legbiztonságosabb, de jól megérthető a módszer lényege. Maradunk a már megismert Alice és Bob elnevezésű felekné. Alice a d_A titkos kulcsával kiszámolja az $C = m^{d_A} \pmod n$ értékét, ahol m az üzenetét jelenti. Ezután elküldi Bobnak, aki miután megkapta Alice nyilvános e_A kulcsával megfejti

$$C^{e_A} \pmod n = (m^{d_A})^{e_A} \pmod n = m.$$

Ha eredményül az üzenetet kapja meg, azaz m értelmes szöveg, biztos lehet benne, hogy az üzenet Alicetól jött. Itt titkosítás nem történik, hiszen a nyilvános kulcs ismeretében bárki megfejtheti az üzenetet.

Az üzenet teljes titkosítása nyilvános kulcsú módszerekkel elég problémás, hiszen ez olykor nagyon időigényes. Még a különböző gyorsítási módszerekkel is meglehetősen lassú az RSA. Emiatt nem az egész üzenetet szokás titkosítani, hanem annak csak egy egyedi kivonatát. Ezt a kivonatot üzenetpecsétnek nevezzük (message digest, MD). Ezek közül a legismertebbek az SHA-1 vagy MD5. Ezek nagyon egzotikus algoritmusok, arra képesek, hogy tetszőleges hosszúságú bitsorozatból egy fix hosszúságú bitsorozatot ad. (Ennek hossza SHA-1 esetén 160 bit, MD5 esetén 128 bit). Ez a viszonylag rövid bitsorozat képviseli a továbbiakban a dokumentum tartalmát.

Ebben az esetben az aláírás folyamata a következőképpen alakul. Kiszámoljuk az $s = (MD(m))^d \pmod n$ úgynevezett ellenőrző összeget és elküldjük az $\{m, s\}$ párt, így gyorsítva az aláírás folyamatát. Megjegyezzük, hogy az $\{e, n\}$ pár publikus, tehát felhasználható az ellenőrzésre.

4. Feladatok

1. Legyenek $p = 2609$ és $q = 3023$ adott prímek. Határozzuk meg az RSA használatához szükséges többi paramétert!

2. Határozzuk meg $2109^{157} \pmod{2773}$ értékét a megismert szukcesszív négyzetreemelés segítségével!

3. Legyenek $n = 2773$ és $e = 17$, titkosítsuk a SZAUNA szót, a kódoláshoz használjuk a betűk abc-ben elfoglalt helyének sorszámát. (Például S helyett 19, A helyett 01 szerepel)

4. Határozzuk meg az eredeti két betűt, ha a titkosított szöveg 1281. Azt tudjuk, hogy $e = 17, n = 2773$ és $\phi(n) = 2668$.

5. Tegyük fel, hogy elrontottuk n választását, amely nem két, hanem három prím szorzata, $n = 7 \cdot 13 \cdot 19$. Határozzuk meg az RSA működtetéséhez szükséges paramétereket és nézzük meg milyen d esetén tudjuk fejteni a szöveget.

9. fejezet - Prímtesztek és faktorizációs eljárások

1. Prímtesztek

Az előzőekben láthattuk, hogy a módszer jósága két nagy prím megfelelő megválasztásán múlik. Nem tudunk azonban olyan algoritmust, amely tetszőleges pozitív egész esetén polinomiális időn belül el tudja dönteni, hogy az illető szám prím-e vagy sem. Szükségünk lenne tehát egy olyan algoritmusra, amely nagyon kis valószínűséggel hibázik. Nyilvánvalóan egy matematikus sem örül annak, ha egy szám „nagy valószínűséggel prím", de ezekben az esetekben érdemes más prímtesztek használata, esetleg többlet gépidő bevetése a cél érdekében.

Mielőtt nekilátnánk a prímtesztelésnek néhány számot előre érdemes kizárni, jelesül azokat, amelyeket nyilvánvalóan tudunk, hogy nem prímek. Ilyen könnyen ellenőrizhető módszer van az A halmaz elemeivel való osztásnál, ahol $A = 2,3,5$, illetve könnyen kizárhatók a négyzetszámok is. Általában néhány előre rögzített prímmel való osztást is tesztelünk, mielőtt a módszereket elkezdjük alkalmazni. Az előzetes szűrésre azért van szükség, mert a következő teszteknek erőteljes erőforrás szükségletük van.

1.1. Euler–Fermat tételen alapuló prímteszt

A matematikai részben ismertetett Euler–Fermat tétel egy triviális következménye, hogy ha $(w, m) = 1$ és

$$w^{m-1} \not\equiv 1 \pmod{m},$$

akkor az m szám összetett.

Ez megfelelne prímtesztnek, hiszen csak a kongruencia igaz vagy hamis volta eldöntené a prímség problematikáját. Az a probléma azonban, hogy vannak olyan összetett számok is amelyek átmennek a teszten.

Tetszőleges w-hez létezik olyan m összetett szám van, melyre teljesül, hogy $(w, m) = 1$ és

$$w^{m-1} \equiv 1 \pmod{m}$$

kongruencia teljesül. Ezeket az m számokat w alapú pszeudoprímeknek nevezzük.

Ilyen például a 91, amely 3 alapú pszeudoprím, ugyanis könnyen belátható, hogy $3^{90} \equiv 1 \pmod{91}$, ugyanakkor $91 = 3 \cdot 17$.

A következő tétel a bevezetőben említett valószínűségi állítást tartalmazza. Ha valamely w-re teljesül a teszt, akkor azt mondjuk, hogy w tanúskodik m prímsége mellett.

9.1. Tétel. Az összes vagy legfeljebb a fele az olyan w egészeknek, melyre

$$1 \leq w < m, (w, m) = 1,$$

tanúskodik m prímsége mellett.

Erre a tételre már alapozhatunk egy egyszerű prímkereső módszert.

Először az adott m-hez véletlenszerűen választunk egy w egészt, melyre $1 \leq w < m$. Ezután Euklideszi algoritmussal meghatározzuk w és m legnagyobb közös osztóját. Ha $(w, m) > 1$, akkor m összetett. Ha nem teljesül az egyenlőtlenség, akkor nekiláthatunk a tesztelésnek.

Kiszámítjuk az $u = (w^{m-1}, \pmod{m})$ kifejezés értékét. Ha $u \neq 1$, akkor az m szám összetett. Ha $u = 1$, akkor w tanúskodik m prímsége mellett.

Ha találunk k tanút, akkor annak valószínűsége, hogy m összetett legfeljebb 2^{-k}, kivéve azt a 9.1 Tételben említett szerencsétlen esetet, amikor is az összes választott b tanú lesz, ugyanakkor a szám mégis összetett.

Az ilyen prímeket Carmichael-számoknak nevezzük. A legkisebb ilyen tulajdonságú szám az $561 = 3 \cdot 11 \cdot 17$. W. R. Alford, A. Granville és C. Pomerance (lásd[2]) igazolta, hogy a pszeudoprímekhez hasonlóan ezek is végtelen sokan vannak. Ha az n szám Carmichael-szám, akkor n négyzetmentes, van legalább három prímfaktora, illetve ha p egy prímosztója, akkor $p-1$ osztója $n-1$-nek. Láthatjuk, hogy ez a módszer további finomításra szorul.

1.2. Solovay–Strassen prímteszt

A továbbiak megértéséhez néhány matematikai fogalmat kell megismernünk. A Legendre-szimbólumot, melyet $\left(\frac{a}{p}\right)$-vel jelölünk, tetszőleges a egész és p páratlan prímszám esetében, a következőképpen értelmezzük:

$$\left(\frac{a}{p}\right) = \begin{cases} 0, & \text{ha } a \equiv 0 \pmod{p} \\ 1, & \text{ha } a \not\equiv 0 \pmod{p} \text{ és } \exists x \in \mathbb{Z}, a \equiv x^2 \pmod{p} \\ -1, & \text{ha } a \not\equiv 0 \pmod{p} \text{ és } \nexists x \in \mathbb{Z} \text{ melyre } a \equiv x^2 \pmod{p} \end{cases}$$

Ha $\left(\frac{a}{p}\right) = 1$ akkor a-t kvadratikus maradéknak nevezzük$\pmod{p}$.

A Legendre-szimbólum segítségével definiálhatjuk a számunkra fontos Jacobi-szimbólumot. Legyen $3 \leq n$ és $n = p_1^{e_1} p_2^{e_2} \cdots {}_k^{e_k}$, ekkor

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{e_1} \left(\frac{a}{p_2}\right)^{e_2} \cdots \left(\frac{a}{p_k}\right)^{e_k}.$$

9.2. Tétel. Ha m páratlan prím, akkor minden w esetén

$$w^{\frac{m-1}{2}} \equiv \left(\frac{w}{m}\right) \pmod{m}.$$

Az előzőekben említett pszeudoprímnek itt is van megfelelője. Ha egy m összetett szám kielégíti az előző kongruenciát valamely m-hez relatív prím w esetén, akkor Euler pszeudoprímnek nevezzük.

9.3. Tétel. Ha egy m szám Euler pszeudoprím a w alapra nézve, akkor pszeudoprím is a w alapra nézve.

Ez a prímteszt mégis egy „erősebb” prímteszt, ugyanis a Carmichael-számnak nincsen megfelelője, azaz érvényes a következő tétel.

9.4. Tétel. Ha m egy páratlan összetett szám, akkor legfeljebb a fele a w egészeknek, melyekre $1 < w < m$ és $(w, m) = 1$, elégíti ki az előző 9.2. Tételben szereplő kongruenciát.

Az előző részben ismertetett algoritmussal megegyező lépésekkel itt egy jól használható algoritmushoz jutunk. Ha k számú w tanúskodik m prímsége mellett, akkor legfeljebb 2^{-k} annak valószínűsége, hogy m nem prím. Ezen becslést nem lehet tovább javítani, ugyanis létezik olyan m egész, amely az alapok felére nézve Euler pszeudoprím.

Az így felépített prímtesztet Solovay-Strassen tesztnek nevezzük.

További előnyét jelenti a módszernek, hogy a kongruenciában szereplő Jacobi szimbólumok a következő tétel segítségével könnyen meghatározhatók.

Tétel. (Gauss kvadratikus reciprocitási tétele). Ha p és q egymástól különböző páratlan prímek, akkor

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

1.3. Miller–Rabin prímteszt

A fentebb említett prímtesztelő algoritmusok nagyon lassan segítik csak a munkánkat. A következő, prímszámokra vonatkozó tétel egy gyakorlatban jobban használható módszert ad a kezünkbe.

> **9.5. Tétel.** Legyen p egy páratlan prím és $p-1=2^s r$ ahol r páratlan. Ha $(a,n)=1$ és $1<a<n$ akkor
>
> $$a^r \equiv 1 \pmod{n} \quad \text{vagy} \quad a^{2^{s'}r} \equiv -1 \pmod{n}$$
>
> teljesül valamely $0<s'<s$.

A 9.5. Tételre alapozott tesztet Miller–Rabin prímtesztnek nevezzük és a következőkben részletezzük.

Előkészítő lépések
: Választunk egy tetszőleges n páratlan számot és egy $1<a<n$ természetes számot. Ha $(a,n)\neq 1$, akkor n összetett, ha $(a,n)=1$, akkor előállítjuk $(n-1)$-et a következő alakban $n-1=2^s r$, ahol r páratlan.

Szukcesszív gyökvonás
: Teszteljük az $a^{2^s r} \equiv 1 \pmod{n}$ kongruencia teljesülését (ami egy Fermat tesztlépésnek is felfogható), majd „szukcesszív gyökvonások” sorozatába kezdünk.

Teszt
: Az első gyökvonás után három lehetőség van.

- Ha $a^{2^{s-1}r} \not\equiv \pm 1 \pmod{n}$, akkor n összetett.
- Ha $a^{2^{s-1}r} \equiv 1 \pmod{n}$, akkor tovább folytatjuk a gyökvonást.
- Ha $a^{2^{s-1}r} \equiv -1 \pmod{n}$ teljesül, akkor azt mondjuk, hogy a tanúskodik n prímsége mellett.

További gyökvonások
: Amíg az előző algoritmus folytatása lehetséges további gyökvonásokat végzünk.

Teszt vége
: Végül, ha a gyökvonások végén $a^r \equiv 1 \pmod{n}$ kongruencia teljesül, akkor is azt mondjuk, hogy a tanúskodik n prímsége mellett.

Ha egy összetett szám „átmegy” az előző teszt lépéseken, akkor azt erős pszeudoprímnek nevezzük.

> **9.6. Tétel.** Ha egy n szám erős pszeudoprím az a alapra nézve, akkor Euler pszeudoprím is az a alapra nézve.

Az előzőektől eltérő módon, egy n összetett szám esetén, a Miller–Rabin prímtesztben választható a értékek legfeljebb az $1/4$-e tanúskodik n prímsége mellett. Ez azt jelenti, hogy k teszt elvégzése után $(\frac{1}{4})^k$ annak valószínűsége, hogy nem prímre leltünk.

Megemlítjük, hogy $n<25109$ esetében egyetlen összetett szám sem megy át a Miller–Rabin prímteszten, ha a $\{2,5,7,13\}$ halmaz elemeit, mint választandó a értékeket végigpróbáljuk.

1.4. AKS algoritmus

Az algoritmus egy determinisztikus prímteszt, melyet 3 indiai matematikus, Manindra Agrawal, Neeraj Kayal és Nitin Saxena 2002-ben, majd 2004-ben újra publikált (lásd [1]). Az első olyan eljárás, amely determinisztikus, futási ideje polinomiális és nem alapszik semmilyen hipotézisre. A megjelenést követően Lenstra és Pomerance 2005-ben (lásd [9]]) megjelent dolgozatában tovább javította eredeti algoritmus futási idejét.

Az algoritmus implementálása azóta is számos nyitott kérdést rejt magában. Az algoritmus egy régóta ismert azonosságra épül, mely szerint az n szám akkor és csakis akkor prím, ha fennáll a következő összefüggés

$$(x-a)^n \equiv (x^n - a) \pmod{n},$$

ahol a maradékos osztást a polinom együtthatóin kell elvégezni.

A könnyebb érthetőség kedvéért mutatunk egy egyszerű példát.

9.7. Példa. Igazoljuk, hogy az 5 prímszám!

Ekkor

$$(x-1)^5 = x^5 + 5x^4 + 10x^3 + 10x^2 + 5x + 1 \equiv (x^5 - 1) \pmod{5},$$

a polinom minden együtthatójának 5-tel való osztási maradéka 0, kivéve az első és utolsó együtthatókat.

A tétel értelmezéséhez szükség van az rend fogalmára.

9.8. Definíció. Valamely r természetes szám esetében azt a legkisebb t természetes számot, melyre az

$$n^t \equiv 1 \pmod{r}$$

kongruencia teljesül az n elem rendjének nevezzük és $ord_r(n)$-el jelöljük.

Az AKS módszer a következő tételen alapszik.

9.9. Tétel. Adott $n \geq 2$ szám esetében legyen r egy pozitív egész és $r < n$ és $ord_r(n) > \log_2(n)$. Az n szám akkor és csakis akkor prím, ha teljesülnek az alábbi feltételek:

1. n nem teljes hatvány,
2. n-nek nincs r-nél kisebb vagy vele egyenlő prímtényezője,
3. $(x-a)^n \equiv x^n - a \pmod{x^r - 1, n}$, bármely a egész szám esetében, ahol $1 \leq a \leq \sqrt{r}\log n$.

A tételben szereplő $(x-a)^n \equiv x^n - a \pmod{x^r - 1, n}$ kongruencia azt jelenti, hogy meghatározzuk az adott polinomok $x^r - 1$ polinommal való osztási maradékát, majd az együtthatókat $\pmod{n}$ vesszük.

2. Egész számok faktorizációja

Az általunk megismert RSA módszer azon alapszik, hogy az egész számok faktorizációja nehéz matematikai feladatnak számít, azaz nem ismert hatékony algoritmus ezen faktorok meghatározására. Ebben a fejezetben néhány olyan algoritmust mutatunk meg, amivel van esélyünk a faktorizációra. Ez másképpen azt is jelenti, hogy az RSA tervezőinek figyelnie kell a rendszer kidolgozásánál e lehetséges törési lehetőségekre.

2.1. Fermat-féle faktorizáció

Elsőként olyan esetet tekintünk, ami azokban az esetekben használatos, amikor a faktorizálandó n számot felírhatjuk két négyzetszám különbségeként, és az egyik négyzetszám kicsi.

9.10. Tétel. Legyen n egy pozitív páratlan egész. Ekkor létezik kölcsönösen egyértelmű hozzárendelés az n természetes szám ab alakú faktorizációja, ahol $a \geq b > 0$, és $t^2 - s^2$ alakban való felírása között, ahol t és s nem negatív egészek.

Bizonyítás. A tétel egyik irányú állítása nyilvánvalóan következik, az n következő felírásából:

$$n = ab = \left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2.$$

Megfordítva, az

$$n = t^2 - s^2 = (t+s)(t-s)$$

egyenlőségből következik az állítás. □

Ha $n = ab$ és a és b közel vannak egymáshoz, akkor $\frac{a-b}{2}$ „kicsi”, így t közel van $\sqrt{n}$-hez. Nyilvánvaló, hogy a „kicsi” jelző nem túlságosan jól definiált, és szokatlan is a matematikai irodalomban, de néhány próbálkozás után jól érthető a fogalom.

Azaz t megtalálásához a próbálkozást a $[\sqrt{n}] + 1$ esettel kezdjük, majd egyesével növeljük a számokat, és közben figyeljük, hogy mikor teljesül a $t^2 - n = s^2$ egyenlőség. Egy példán keresztül még érthetőbb lesz a módszerünk.

9.11. Példa. Faktorizáljuk a 200819-et!

Esetünkben a $[\sqrt{200819}] + 1 = 449$. Ekkor $449^2 - 200819 = 782$, amely nem teljes négyzet. A következő próbálkozásunk a $t = 450$ eset, amikor is $450^2 - 200819 = 1681 = 41^2$. Ekkor sikerült felbontani a faktorizálandó számot két négyzetszám különbségére és így

$$200819 = 450^2 - 41^2 = (450+41)(450-41) = 491 \cdot 409.$$

Nyilvánvaló tehát, hogy az RSA tervezésénél nem érdemes túl közeli prímszámokból kiindulni. A Fermat-féle módszer egy módosított változata, azonban ilyen esetekben is segíthet.

Ebben az esetben válasszunk egy kis k értéket és válasszuk t-t a következőknek, $[kn]+1, [kn]+2, [kn]+3$ …A t választása után vizsgáljuk meg a $t^2 - kn = s^2$ egyenlőség teljesülését. Ekkor $(t+s)(t-s) = kn$ és így $t+s$-nek van nem triviális közös osztója n-el, azaz a $(t+s, n)$ meghatározása megadja a kívánt végeredményt. Mint az előzőekből már kiderült az Euklidészi algoritmus ezt az eredményt könnyen megadja.

9.12. Példa. Faktorizáljuk a 141467-et!

Gyorsan kiderül, hogy az alapmódszerrel nem jutunk gyorsan célba, hiszen a 377-el kell kezdenünk a próbálkozást.

Legyen most $k = 3$ és próbálkozzunk a $[\sqrt{3n}] + 1$, $[\sqrt{3n}] + 2 \ldots$ értékekkel, azaz $t = 652, 653, \ldots$ Rövid próbálkozás után azt kapjuk, hogy

$$655^2 - 3 \cdot 141467 = 68^2.$$

Ezek után az Euklidészi algoritmussal kiszámítjuk, hogy $(655 + 68, 141467) = 241$. A $141467 = 241 \cdot 587$ egyenlőség adja a feladat megoldását.

Figyelmesen megnézve az eredményt azt látjuk, hogy az egyik faktor a másik háromszorosa, ami indokolhatja a $k = 3$ választás jogosságát.

Az előző módszereknek megadható egy általánosítása is. Ha bármely faktorizációs problémánál meg tudunk adni egy

$t^2 \equiv s^2 \pmod{n}$

kongruenciát, ahol $t \not\equiv \pm s \pmod{n}$, akkor azonnal meg tudjuk adni n egy faktorát a $(t+s,n)$ vagy a $(t-s,n)$ kiszámításával.

2.2. Pollard-féle ρ heurisztikus módszer

A címben említett módszert John Pollard publikálta 1975-ben [13]. Egy tetszőleges egész szám prímosztóinak a meghatározására lehet alkalmazni, ahol az egész szám nem lehet prímhatvány, a prímosztók pedig lehetőleg kicsik kell, hogy legyenek.

Az algoritmus, amelyet Monte–Carlo módszernek is neveznek, a következőképpen működik, feltételezve, hogy egy n szám prímosztóit szeretnénk meghatározni,

- választunk egy egész együtthatós polinomot, amely lehetőleg elég egyszerű legyen a további számolásokhoz (például az $f(x) = x^2 + 1$ ilyen választásnak bizonyul),
- választunk egy x_0 kezdőpontot vagy véletlenszerűen generálunk egyet (például $x_0 = 1$ vagy $x_0 = 2$).
- a következő iterációkat számoljuk ki,

  $x_1 = f(x_0) \pmod{n}$, $x_2 = f(f(x_0)) \pmod{n}$, ...

  azaz $x_{j+1} = f(x_j) \pmod{n}$ $j = 1,2,\ldots$
- az x_i értékeket összehasonlítjuk, és olyanokat keresünk, amelyek különböző osztályokba tartoznak $\pmod{n}$, de ugyanabba $\pmod{r}$. Azaz teszteljük az $(x_i - x_j, n)$ értékeket mindaddig, amíg n valódi osztóját nem kapjuk.

Megjegyezzük, hogy bizonyos számú iteráció elvégzése után után ismétlődést fogunk tapasztalni.

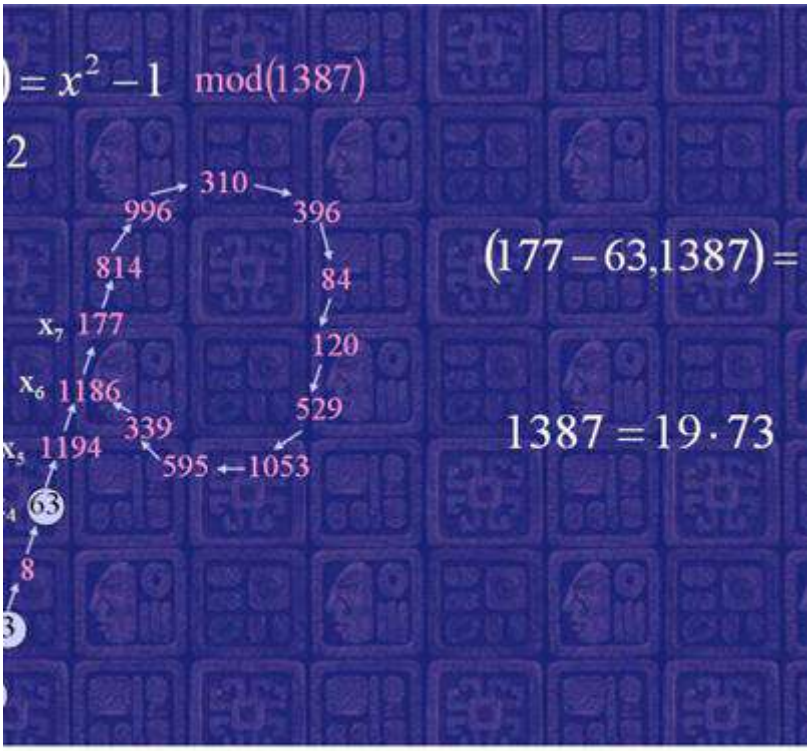
Az f polinomról feltételezzük , hogy Z/nZ önmagára való leképezését eléggé „véletlenszerűen” végzi, azaz lehetőleg minden maradék előforduljon változatos sorrendben. Egy példán illusztráljuk az eddigieket.

9.13. Példa. Faktorizáljuk az 1387-et a Pollard-féle ρ módszer segítségével!

Használjuk az $f(x) = x^2 - 1$ polinomot és az $x_0 = 2$ pontot. Az iterációk elvégzését a következő táblázat tartalmazza. Figyeljük meg, hogy a 17-edik iteráció elvégzése után visszakapjuk az x_6 pontot, azaz ezt a ciklust fogja ezek után az iteráció ismételni. Az önmagába záródó furcsa hurokról kapta a ρ-módszer nevet a módszer.

2	x_{10}	
3	x_{11}	
8	x_{12}	
63	x_{13}	
1194	x_{14}	
1186	x_{15}	
177	x_{16}	
811	x_{17}	
996	x_{18}	

Az $(x_7 - x_4, n) = (177 - 63,1387) = 19$ egyenlőségből kapjuk, hogy 1387 egyik faktora 19, így az $1387 = 19 \cdot 73$ egyenlőség adódik.



Nyilvánvalóan érdekes számunkra, hogy meddig kell keresnünk az $(x_i - x_j, n)$ értékek között, amíg egy nem triviális eredményt kapunk. Amennyiben r egy nem triviális osztója n-nek, az érdekel bennünket, hogy tekintetbe véve Z/nZ összes önmagára való leképezését és az összes lehetséges x_0 értéket, átlagban hányadik i értékhez van olyan j, úgy, hogy $x_i \equiv x_j \pmod r$. Úgyis feltehetjük a kérdést, hogy hányadik iterációtól kezdődik el a fentebb említett ismétlődés. N. Koblitz ezzel kapcsolatban a következőt igazolta [7].

9.14. Tétel. Legyen S egy r-elemű halmaz, az f leképezés S-nek önmagára való leképezése és $x_0 \in S$. Tekintsük az $x_{i+1} = f(x_i)\ i = 0,1,2,\cdots$ iterációt, legyen $\lambda > 0$ egy tetszőleges valós szám és $l = 1 + [\sqrt{2\lambda r}]$. Ekkor azon f, x_0 párok aránya, amelyre az $x_0, x_1, \ldots x_l$ különböző és ahol f befutja S önmagára való összes kölcsönösen egyértelmű leképezését és x_0 felveszi S összes lehetséges értékét, kisebb, mint $e^{-\lambda}$.

2.3. A kvadratikus szita módszere

A kvadratikus szita módszerét Carl Pomerance publikálta először (lásd [15]).

Carl Pomerance

Az egyik leggyorsabb faktorizációs algoritmusnak számít. A faktorizálandó n számra egyetlen kitétel van, mégpedig az, hogy egyik prímosztója se legyen nagyobb, mint $\sqrt{n}$. Az algoritmus megkeresi azokat az x és y egész számokat, melyekre fennállnak a következők:

$$x^2 \equiv y^2 \pmod{n},$$
$$x \not\equiv y \pmod{n},$$
$$x \not\equiv (-y) \pmod{n}.$$

Az így kapott x és y értékekből kapjuk n egy faktorát az $(x-y,n)$ kiszámításával.

Az algoritmus egy $f(x) = (m+x)^2 - n$ polinomot használ, ahol $m = [\sqrt{n}]$ és $x = 0, \pm 1, \pm 2, \ldots$ Az $f(x)$ értékek kiszámítása mellett az algoritmus meghatározza azok faktorizációs felbontását is.

Az algoritmus a továbbiakban megállapít egy B küszöbértéket és egy S listát, mely tartalmazni fogja azokat a p prímeket, amelyekre teljesülnek a következő tulajdonságok, $\left(\frac{n}{p}\right) = 1$ és $p \leq B$. Esetünkben $\left(\frac{n}{p}\right)$ a Legendre szimbólumot jelenti.

A kiszámolt $f(x)$ értékek közül csak azokat fogjuk eltárolni, amelyek faktorizációs felbontásában nincs egyetlen egy olyan prímtényező sem, mely ne szerepelne az S listában. A szakirodalom [15] az ilyen tulajdonságú elemeket B-sima tulajdonságúnak nevezi, jobb híján megtartjuk ezt az elnevezést. A B érték meghatározására a javasolt érték

$$B = e^{\sqrt{\ln n \ln\ln n}}.$$

Ha az S lista elemszáma k és $f(x_i)$ faktorizációs felbontása

$$\prod_{j=1}^{k} {p_j}^{e_{i,j}},$$

alakú, akkor a meghatározott $f(x_i)$-k száma legalább eggyel több kell hogy legyen, mint k.

Minden egyes prímtényezős felbontásban az e_i kitevőkhöz hozzárendelhetünk k dimenziós vektort a következő módon:

$$v_i = (v_{i1}, v_{i2}, \ldots v_{ik}),$$

ahol $v_{ij} \equiv e_{ij} \pmod{2},\ j = 1,2,\ldots k.$

Ezek után azokat a vektorokat kell kiválogatnunk, amelyeknek az összege 0-at eredményez $\pmod 2$. A módszer kitalálója ezzel biztosítja, hogy ha ezeket az $f(x_i)$ értékeket összeszorozzuk, akkor teljes négyzetet kapjunk, esetünkben y^2. A hozzátartozó $m + x_i$ értékeket összeszorozva megkapjuk x^2-et is. Ezek után már csak a feltételeket kell ellenőriznünk. A következő példa jól illusztrálja a módszert.

9.15. Példa. Határozzuk meg az $n = 25387$ osztóit.

Legyen $m = \sqrt{n} = 159$ és $S = \{-1, 2, 3, 23, 41, 43, 47\}$. Alkalmazzuk az $f(x) = (m+x)^2 - n$ függvényt a fenti módon. A létrejövő prímtényezős felbontást és a v_i vektorokat a táblázatban közöljük.

i	x	$f(x)$	$m+x$	$f(x)$ felbontása	v_i
1	-2	-738	157	$(-1)\cdot 2\cdot 3^2\cdot 41$	(1100100)
2	-6	-1978	153	$-1\cdot 2\cdot 23\cdot 43$	(1101010)
3	10	3174	169	$2\cdot 3\cdot 23^2$	(0110000)
4	-11	-3483	148	$(-1)\cdot 3^4\cdot 43$	(1000010)
5	17	5589	176	$3^5\cdot 23$	(0011000)
6	-29	-8487	130	$(-1)\cdot 3^2\cdot 23\cdot 41$	(1001100)
7	32	11094	191	$2\cdot 3\cdot 43^2$	(0110000)
8	80	31734	239	$2\cdot 3^2\cdot 41\cdot 43$	(0100110)

Könnyen ellenőrizhetjük, hogy $v_4+v_5+v_6+v_7+v_8=0$. A megfelelő $f(x)$ felbontások összeszorzása után teljes négyzetet kapunk, amelyet y^2-el jelölünk, így

$$y^2=((-1)\cdot 3^4\cdot 43)\cdot(3^5\cdot 23)\cdot((-1)\cdot 3^2\cdot 23)\cdot(2\cdot 3\cdot 43^2)\cdot(2\cdot 3^2\cdot 41\cdot 43).$$

Ekkor a következő x és y értékeket kapjuk,

$$y\equiv(-1)\cdot 2\cdot 3^7\cdot 23\dot{4}1\cdot 43^2\equiv 22426 \pmod{25387},$$
$$x\equiv 130\cdot 148\cdot 176\cdot 191\cdot 239\equiv 22426 \pmod{25387}.$$

Ebben az esetben azt kapjuk, hogy $x\equiv y \pmod n$, ami azt jelenti, hogy x és y nem felel meg céljainknak. Keressünk tehát új x és y értékeket.

Esetünkben $v_3+v_7=0$, így a megfelelő $f(x)$ értékek kiválasztása után azt kapjuk, hogy

$$y^2=(2\cdot 3\cdot 23^2)\cdot(2\cdot 3\cdot 43^2)=2^2\cdot 3^2\dot{2}\dot{3}^2\cdot 43^2.$$

Ekkor könnyen adódik, hogy

$$y\equiv 2\cdot 3\cdot 23\cdot 43\equiv 5934 \pmod{25387},$$
$$x\equiv 169\cdot 191\equiv 6892 \pmod{25387}.$$

Esetünkben teljesül, hogy

$$x\not\equiv y \pmod n,$$
$$x\not\equiv (-y) \pmod n.$$

Ezek után az Euklidészi algoritmus használatával meghatározzuk a módszerben ismertetett legnagyobb közös osztókat, $(x-y,n)$, illetve $(x+y,n)$ értékeket, és így megkapjuk n faktorait, esetünkben

$$(6892-5934,25387)=479\ (6892+5934,25387)=53.$$

3. Feladatok

1.

Határozzuk meg a Fermat faktorizációs módszer segítségével 517 egyik prímosztóját!

2.

Határozzuk meg 2041 osztóit a Fermat faktorizációs eljárás módosított változatával!

3.

Határozzuk meg 25661 egyik prímosztóját a Pollard-féle heurisztikus ρ-módszerrel. Használjuk az $f(x) = x^2 + 1$ polinomot és az $x_0 = 2$ pontot!

4.

Határozzuk meg 4087 valamely prímosztóját a Pollard-féle heurisztikus ρ-módszerrel! Használjuk az $f(x) = x^2 + x + 1$ polinomot és az $x_0 = 2$ pontot.

5.

Döntsük el a megismert eljárások alapján, hogy 2701 prím-e vagy sem!

6.

Határozzuk meg a legkisebb pszeudoprímet az 5 alapra nézve!

7.

Mutassuk meg, hogy 65 erős pszeudoprím a 8 és 18 alapokra nézve, de nem az a 18 alapra nézve, amely 8 és 18 szorzata $\pmod{65}$!

8.

Igazoljuk, hogy 17 prím az AKS algoritmus felhasználásával!

9.

Igazoljuk, hogy az 1729 Carmichael szám!

10.

Határozzuk meg 20473 faktorait kvadratikus szita használatával!

10. fejezet - Elliptikus görbék

Egy ideje egyre több helyen találkozhatunk az ECC betűhármassal, amely egy nyilvános kulcsú kriptorendszert jelöl. A betűszó az angol Elliptic Curve Cryptosystem elnevezés rövidítéséből ered, amely elliptikus görbéken megvalósított titkosítást jelent. Nagy előnyének említi a szakirodalom, hogy az RSA-nál kisebb méretű kulcsokkal érhető el ugyanakkora biztonság, sokkal gyorsabb a működése, kisebb a tárigénye a kulcsok tárolásához.

Az elliptikus görbék története a 17. század végéig nyúlik vissza, amikor Isaac Newton (1642-1727) és Gottfried Wilhelm Leibniz (1646-1716) egymástól függetlenül kidolgozták a differenciál- és integrálszámítás elméletét.

Isaac Newton

Gottfried Wilhelm Leibniz

Az új matematikai eszközöket a kor tudósai előszeretettel alkalmazták olyan fizikai problémák megoldására, amelyek geometriai megfontolásokat igényeltek. Általában olyan görbék meghatározása volt a feladat, amelyeket egy adott „részecske” bizonyos kényszererők hatására leír. Jakob Bernoulli (1654-1705) a következő problémát vetette fel. Melyik az a görbe, amely mentén leguruló test egyenlő időközök alatt egyenlő utakat tesz meg. E probléma vizsgálata során jutott el az $(x^2 + y^2)2 = 2a^2(x^2 - y^2)$ görbéhez, amelyet egy „elfordított nyolcashoz” hasonlította és lemniscusnak nevezte el, amely görögül szalagot jelent. A fenti egyenlettel meghatározott görbét, Bernoulli-féle lemniszkátának szokás hívni. Az ívhosszának tanulmányozása az

$$\int_0^1 \frac{1}{\sqrt{1-x^4}} dx$$

integrálra vezet és ezt elliptikus integrálnak nevezik, mivel a probléma rokon az ellipszis ívhosszának kiszámításánál felmerülő problémával. Az ilyen típusú függvények inverzeit elliptikus görbéknek nevezzük.

Az elkezdett munkát Giulio Carlo Fagnano (1682-1766) olasz matematikus folytatta, később Leonhard Euler (1707-1783) munkája alapozta meg az elliptikus integrálok elméletét. A további fejlődést az elliptikus görbék elméletében Adrien-Marie Legendre (1752-1833), Niels Henrik Abel (1802-1829) és Carl Gustav Jakob Jacobi munkássága jelentette.

Az 1980-as évek közepén Neal Koblitz (University of Washington) és Victor Miller (IBM) javasolták, egymástól függetlenül, az elliptikus görbék kriptográfiai alkalmazását.

Azért, hogy kedvet kapjunk a következő matematikai fejtegetésekhez nézzük meg, az előzőekben ismertetett három kriptorendszer általánosan, vagy szabvány szerint használt kulcsméreteit. Az egy sorban lévő kulcsméretek közel azonos biztonságot nyújtanak ([18]).

nodulus	AES	RSA modulus	RSA
12	56	512	
61	80	1024	
56	128	3072	1
84	192	7680	2
12	256	15630	3

A táblázat egyértelműen visszatükrözi a bevezető sorokban már említett tényt, miszerint az elliptikus görbéken alapuló titkosítási rendszer sokkal kisebb kulcsmérettel is megfelelő biztonságot nyújt.

Az elliptikus görbék elméletének kezdeteiről már tettünk említést, jól láthatóan egy bonyolultabb elméletről van szó, mint amit az RSA vagy az AES kifejtésénél láthattunk. A következőkben némi matematikai hátteret nyújtunk a megértéshez.

1. Az elliptikus görbe fogalma

Az elliptikus görbék elméletének alapos megismeréséhez számos kiváló könyvet tudunk ajánlani (lásd például [3], [8]). Jelen fejezetben azonban nem törekszünk kimerítő alaposságra, csak a témánk megértéséhez szükséges elméleti háttér ismertetésére.

10.1. Definíció. Legyen K egy olyan test, melynek a karakterisztikája nem kettő, nem három és legyen

$$y^2 = x^3 + ax + b, \quad a, b \in K$$

egy olyan harmadfokú polinom, amelynek nincsenek többszörös gyökei. Egy K test feletti elliptikus görbe olyan $P(x, y)$ pontok halmaza, ahol az $x, y \in K$ koordináták kielégítik az

$$y^2 = x^3 + ax + b$$

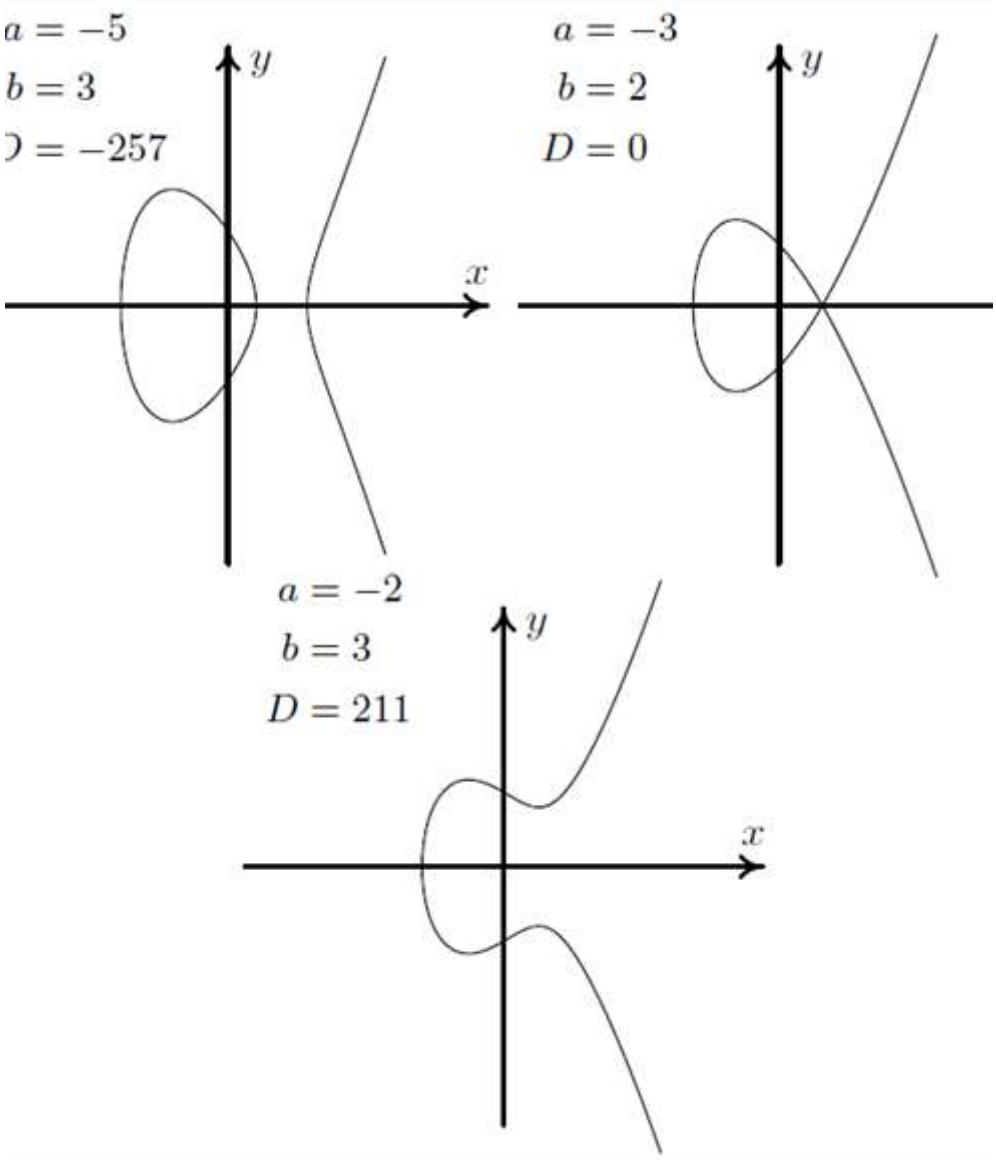
egyenletet, és hozzátartozik a görbéhez egy úgynevezett O-val jelölt „végtelen távoli pont”.

Az elliptikus görbe diszkriminánsán a $D = -16(4a^3 + 27b^2)$ kifejezést értjük. A diszkrimináns nem nulla, ha $x^3 + ax + b$ polinomnak három különböző gyöke van, mint esetünkben.

Abban az esetben, ha a K test a valós számtest, a diszkriminánsnak bizonyos geometriai interpretációját is megadhatjuk. Ha $D \neq 0$, akkor az elliptikus görbe nem szinguláris (a görbe génusza 1). Ha $D = 0$ és $a = 0$, akkor a görbének a szinguláris pontban egy érintője van (cusp singularity). Ezt az esetet úgy jellemezhetnénk, hogy a görbe egy csúcsban végződik. Ha $D = 0$ és $a \neq 0$, akkor a görbe szinguláris pontját csomópontnak (node) mondjuk, amelybe két különböző érintő húzható. Ebben az esetben a görbe elmetszi magát, ahogy a lentebbi példában is látható. A szinguláris esetekkel a továbbiakban nem foglalkozunk, de megemlítjük, hogy ezekben az esetekben a görbe génusza 0.

A következőkben három elliptikus görbét ábrázolunk a diszkrimináns különböző értékeinél. Kriptográfiai céljainknak kizárólag a szingularitással nem rendelkező elliptikus görbék felelnek meg, azaz esetünkben $D \neq 0$.

10.1. ábra. Elliptikus görbék különböző diszkriminánsok esetén



2. Műveletek a görbe pontjaival

Az előbbiekben tárgyalt, szingularitással nem rendelkező görbéken a következőkben műveleteket definiálunk.

1.

Egy P pont additív inverze

Egy $P(x, y)$ pont additív inverze az a $-P$ pont, amely a pont x tengelyre tükrözött képe, amely szintén rajta lesz a görbén, és a koordinátái $(x, -y)$.

2.

Két különböző pont összeadása

Legyen a görbe két különböző pontja P *és* Q! E két pont összegét jelölje R, vagyis $R = P + Q$. A műveletet a következőképpen kell elvégezni:

1. Kössük össze, a P *és* Q pontot egy egyenessel!

2. Az egyenes egy harmadik pontban metszi a görbét, ez a pont lesz az általunk $-R$-el jelölt.

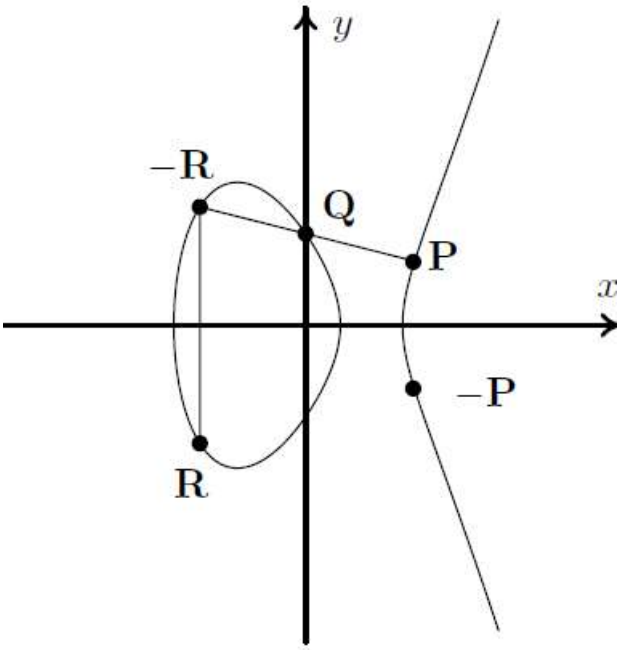
3. E pont x tengelyre tükrözött képe az előző szabály szerint szintén rajta lesz a görbén és ez lesz az R pont.

3.

Egy P pont kétszerezése

A $2P$ pont meghatározása az b) pontban ismertetett módon történik, azzal a különbséggel, hogy a két pont összekötése helyett, a P pontba húzott érintő jelöli ki $-2P$ pontot. Ebből a $2P$ előállítása az a) pontban ismertetett módon zajlik.

10.2. ábra. Műveletek



Érdemes észrevenni, hogy az összeadás előzőleg ismertetett művelete egy eset kivételével az ábrázolt görbe egy pontját állítja elő. Az egyetlen eset az, amikor az (x, y) és$(x, -y)$ pontok összeadását végezzük. Ekkor az előzőleg definiált végtelen távoli pontot kapjuk, amely a definíció értelmében hozzátartozik az elliptikus görbéhez. A görbék pontjainak ilyen típusú összeadását Carl Gustav Jacob Jacobi javasolta először 1835-ben.

Megjegyezzük, hogy az összeadás könnyen elvégezhető algebrai úton is, hiszen egyenesek és az elliptikus görbe metszéspontjait kell számolnunk.

Az additív inverzet már láthattuk, a többi eset a következő.

1.

Két különböző pont összeadása

Ha $P(x_1, y_1)$ és $Q(x_2, y_2)$ pontok nem egymás ellentettjei, akkor a $R = P + Q$ pont koordinátáit megadhatjuk az $s = (y_1 - y_2)/(x_1 - x_2)$ kifejezés felhasználásával,

$$x_R = s^2 - x_1 - x_2,$$
$$y_R = s(x_1 - x_R) - y_1.$$

2.

Egy adott pont kétszerezése

Az előző jelöléseket használva, az $R = 2P$ pont koordinátái a következőképpen adhatók meg

$$x_R = \left(\frac{3{x_1}^2 + a}{2y_1}\right)^2 - 2x_1,$$
$$y_R = -y_1 + \left(\frac{3{x_1}^2 + a}{2y_1}(x_1 - x_R)\right).$$

Megmutatható, hogy a pontok a végtelen távoli ponttal együtt Abel csoportot alkotnak, ahol a zérus szerepét a végtelen távoli pont játsza.

A végtelen távoli pontot eddig a képzelőerőnkre bíztuk, a pontosabb megértés érdekében néhány szóban kitérünk az úgynevezett projektív síkra.

Projektív sík alatt (X, Y, Z) számhármasok ekvivalencia osztályait értjük (nem minden komponens nulla), ahol két számhármast ekvivalensnek mondunk, ha az egyik a másikból skalárral való szorzással származtatható. Egy ilyen ekvivalencia osztályt projektív pontnak nevezünk. Ha $Z \neq 0$ akkor egy és csak egy olyan pont van, amely ekvivalens az $(x, y, 1)$ számhármassal. Könnyen látható, hogy ebben az esetben a projektív sík pontjait meg lehet feleltetni az általunk jól ismert „szokásos” sík pontjainak. A $Z = 0$ esetben kapott pontok alkotják a végtelen távoli egyenest.Az $x = \frac{X}{Z}$ és $y = \frac{Y}{Z}$ helyettesítést elvégezve az általunk tanulmányozott elliptikus görbén az

$$Y^2 Z = X^3 + aXZ^2 + bZ^3$$

egyenlethez jutunk. A $Z = 0$ helyettesítés elvégzése után az $X = 0$ értéket kapjuk. Azaz egyetlen olyan pont van az elliptikus görbén, amelynek a Z koordinátája nulla, a $(0,1,0)$ ekvivalenciaosztály. Ezt a pontot végtelen távoli pontnak nevezzük és O-val jelöljük.

3. Elliptikus görbe a racionális számok teste felett

Amennyiben a definícióban adott K test a racionális számok teste, azaz az a és b együtthatók racionális számok és $x, y \in \mathbb{Q}$, még többet tudunk a görbéről. Louis Mordell 1921-ben igazolta a következő tételt.

> **10.2. Tétel.** Egy racionális számok teste felett értelmezett elliptikus görbén a racionális pontok egy végesen generált Abel csoportot alkotnak.

A tétel kibontásához egy új fogalomra van szükségünk.

> **10.3. Definíció.** Akkor mondjuk, hogy egy P pont rendje N egy elliptikus görbén, ha N a legkisebb olyan természetes szám, melyre $NP = O$.

Megjegyezzük, hogy természetesen nem szükségszerű ilyen N létezése. A matematikusokat és kriptográfusokat erőteljesen érdekli az a kérdés, hogy egy adott elliptikus görbén található-e véges rendű pont. Különösen fontos kérdés ez a racionális számok teste felett értelmezett elliptikus görbék esetén.

A Mordell tételben említett Abel csoport struktúráját is ismerjük. A csoport egy végesen generált torziós részcsoportból (a véges rendű pontok) és véges számú végtelen rendű elem részcsoportjából áll. Ez számunkra azt jelenti, hogy létezik r végtelen rendű $P_1, P_2, \ldots P_r$ pont és $Q_1, Q_2, \ldots Q_s$ prímhatvány rendű pont úgy, hogy az elliptikus görbe minden racionális P pontja felírható

$$P = n_1 P_1 + n_2 P_2 \cdots n_r P_r + m_1 Q_1 + m_2 Q_2 \cdots m_s Q_s$$

alakban, ahol $n_i \in \mathbb{Z}$ és $m_i \in \mathbb{Z}/{p_i}^{e_i}\mathbb{Z}$.

A végtelen rendű elemek számát az elliptikus görbe rangjának nevezzük.

4. Elliptikus görbe véges test felett

Legyen F_q egy véges test és tekintsük az E elliptikus görbénket ezen test felett. Könnyű látni, hogy egy ilyen elliptikus görbének legfeljebb $2q+1$ pontja lehet. Ami azt jelenti, hogy $2q$ darab (x,y) pár és az előzőekben definiált végtelen távoli pont. Megfigyelhetjük, hogy minden lehetséges x-hez legfeljebb kettő y érték tartozik.

Az, hogy mennyi pont van ténylegesen az F_q feletti elliptikus görbén általában nem tudjuk. Helmut Hasse (1898-1979) következő tétele egy becslést ad a pontok számára.

> **10.4. Tétel (Hasse).** Legyen N az F_q-pontok száma az F_q véges test feletti E elliptikus görbén. Ekkor
>
> $$|N-(q+1)| \leq 2\sqrt{q}$$

A könnyebb érthetőség miatt a következőkben tekintsük a vizsgált elliptikus görbét a $\mathbb{Z}_p$ test fölött. Használjuk fel a moduláris aritmetika ismert szabályait a következőkben. Legyen

$$y^2 \equiv x^3 + ax + b \pmod{p},$$

ahol p prím.

Egyszerűen úgy fogalmazhatnánk ebben az esetben, hogy ha az egyenlet mindkét oldala ugyanazt a maradékot adja p-vel történő osztás után, akkor a $P(x,y)$ pont a görbe pontjai közé tartozik. A pontokra és az előzőekben említett diszkriminánsra legyenek érvényesek a következők

1. $0 \leq x \leq p-1$ és $0 \leq y \leq p-1$
2. valamint $(4a^3+27^2) \pmod{p} \neq 0$.

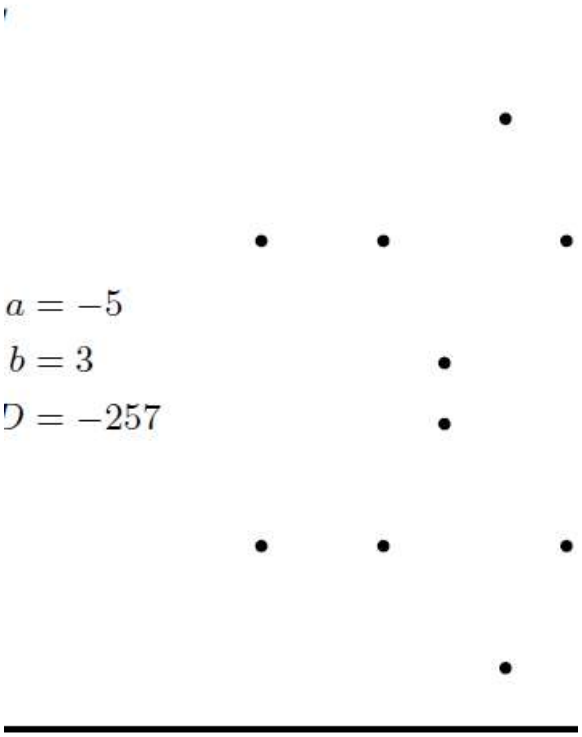
Első látásra nehézkesnek tűnik a munka egy ilyen véges test felett értelmezett elliptikus görbével, de ha kicsit jobban szemügyre vesszük sok figyelemre méltó tulajdonságát fedezhetjük fel. A következők segítenek céljaink megvalósításában,

1. a valós számokkal való számolás lassú és pontatlan, a moduláris aritmetika gyors és pontos, csak egész számokkal dolgozik,
2. a „valós” görbének végtelen sok pontja van, a modulárisnak jóval kevesebb,
3. a moduláris aritmetikában behatárolható a számok értelmezési tartomány, mert a műveletek operandusa (a) és eredménye mindig 0 és $p-1$ közé esik,
4. a moduláris aritmetika alkalmazása megnöveli a kriptográfiai megoldások számát.

Az ilyen görbék másképpen néznek ki, mint az általunk jól ismert valós görbék. A szimmetria ugyanakkor továbbra is megmarad, csak sok esetben nem az x tengelyre vonatkozóan. A következő ábra $p = 11$, $a = 1$, $b = 0$ és paraméterek által definiált „görbét” mutatja (lásd [18]). Megfigyelhetjük, hogy:

1. 11 darab pontja van a görbének,
2. ebből 1 darab az origóban (mert $b = 0$),
3. 10 darab viszonylag véletlenszerűen, de az $y = 5{,}5$-re szimmetrikusan helyezkedik el, ezért
4. minden x értékhez továbbra is kettő y tartozik.

10.3. ábra. Az $E : y^2 \equiv x^3 + 1x + 0 \pmod{11}$ „görbe”



A pontosság kedvéért közöljük fenti görbe pontjait, melyek a következők: $(0{,}0), (5{,}3), (7{,}3), (8{,}5), (9{,}1), (10{,}3), (5{,}8), (7{,}8), (8{,}6), (9{,}10), (10{,}8)$.

Az E görbe pontjainak száma (amit egyébként a görbe kardinalitásának vagy rendjének is hívnak és általában $\#E(p)$-vel jelölik) most csak véletlenül egyenlő p-vel. Azokat a görbéket, amelyek pontjainak száma megegyezik p-vel, rendhagyó görbéknek (anomalous curve) nevezzük és gyakorlatilag az összes szabvány tiltja használatukat, mert létezik hatékony támadási módszer az ilyen görbét használó ECC-rendszer ellen.

5. Műveletek a görbe pontjaival

1.

Additív inverz

A valós számokon értelmezett görbe esetén a $P(x,y)$ pont additív inverze az $(x,-y)$ pont volt. Most is ugyanez a helyzet, csak figyelembe vesszük a modulust is. Az $(x,-y \pmod p)$ kifejezés alatt a továbbiakban az $(x,p-y \pmod p)$ pontot értjük. Ha megnézzük az előző ábra megoldásait, láthatjuk, hogy az egymással szemben lévő pontok y koordinátáinak összege mindig $p=11$. Például $(5,3)$ és $(5,8) \rightarrow 3+8=11$. Tehát egy $R=-P$ pont kiszámolása a következőképpen történik $x_R = x_P$ és $y_R = -y_P \pmod p$.

2.

Összeadás

Nyilvánvalóan nem működik az előzőekben ismertetett módszer, hogy „kössünk össze" két pontot, és keressük meg a harmadik metszéspontot, és nem jó a pont kétszerezéséhez kitalált módszer sem. Egyszerűnek látszik ugyanakkor a korábbi algebrai eredményeket egyszerűen átvétele a moduláris aritmetika szabályai szerint. Azaz

$$s = (y_P - y_Q)(x_P - x_Q)^{-1} \pmod p$$
$$x_R = s^2 - x_P - x_Q \pmod p$$
$$y_R = s(x_P - x_R) - y_P \pmod p$$

6. Diszkrét logaritmus

A knapsack rendszer bevezetésénél már említettük, hogy minden nyilvánoskulcsú kriptorendszer alapja egy olyan probléma, amit gyakorlatilag lehetetlen megoldani. Ez számunkra azt jelenti, hogy, hogy a megfejtéshez szükséges idő sokkal, de sokkal nagyobb, mint amennyi idő az információ megszerzéséhez rendelkezésre áll. Az elliptikus görbéken megvalósított titkosításnak (ECC) is egy ilyen probléma adja a biztonságát. A problémát a szakirodalom ECDLP (Elliptic Curve Discrete Logarithm Problem) jelöléssel használja, jelentése „diszkrét logaritmus elliptikus görbék feletti" kiszámolásának problémája.

1991-ben néhány kutató elkészítette az RSA algoritmus elliptikus görbén alapuló változatát is, de néhány évvel később többen is megmutatták, hogy az elliptikus RSA-nak (ECC-like RSA) nincs számottevő előnye a hagyományos RSA-val szemben. Az ECRSA problémája egyébként továbbra is a faktorizálás maradt.

Eddig lényegében két műveletet definiáltunk a görbén, a pontok összeadását és egy pont duplázását. Ha elképzeljük az általunk könnyen elkészíthető sorozatot $R, R+R, 2R+R, 3R+R$, rájövünk, hogy tulajdonképpen már szorozni is tudunk. Az így képzett $Q=nR$ pontot a pont skalár szorzatának nevezzük. Belátható, hogy az n természetes szám meghatározása a szorzat alapján nem egyszerű feladat főként, ha a görbét egy $\mathbb{Z}_p$ test felett értelmezzük.

> **10.5. Definíció.** Legyen E egy Z_p test feletti elliptikus görbe és R egy pont a görbén. Ekkor az E-n értelmezett diszkrét logaritmusos problémáról beszélünk (az R alapra vonatkozóan), ha adott egy $Q \in E$ pont és keressük azt az n természetes számot, melyre $nR=Q$ egyenlőség teljesül (ha ilyen n létezik). Ebben az esetben n diszkrét logaritmusa Q-nak a p bázis felett.

A diszkrét logaritmus előbb definiált szorzása és az elliptikus görbén értelmezett összeadás, lényegileg ugyanaz.

Megjegyezzük, hogy az ECDLP-n alapuló rendszerek többsége aláíró vagy kulcscserélő rendszer, mert gyors titkosításra ez a módszer is alkalmatlan. A következőkben néhány működő rendszert tekintünk át.

6.1. ECDH - Elliptic Curve Diffie - Hellman kulcscsere

Az eredeti Diffie-Hellman algoritmus a szimmetrikus titkosító rendszerek kulcsmegosztási problémáját oldotta meg. A két résztvevő ugyanazokat a műveleteket végezte el egyező nyilvános és különböző titkos paraméterekkel, de azonos eredményt kaptak, melyet kulcsként használhattak. Az ECDH is ugyan így működik,

csak nem moduláris hatványozást használ, hanem a fejezet eddigi részében megismert elliptikus görbe műveleteket.

10.6. Példa. Szemléltessük egy példán keresztül:

Alice és Bob megegyeznek egy E görbében és egy G pontban, utóbbit bázispontnak hívjuk. A továbbiakban eme paramétereket nyilvános rendszerparamétereknek tekintjük. Alice választ egy véletlen számot, (amely kisebb, mint a G pont rendje) és ugyan így tesz Bob is: Alice száma legyen a, Bobé legyen b. Mindketten titokban tartják választásukat. A kulcscsere következő lépésében Alice kiszámolja aG pontot, melyet elküld Bobnak, aki Alice műveletéhez hasonlóan kiszámolja bG pontot és elküldi Alicenek. Végül Alice a Bobtól kapott bG-t megszorozza a-val, így megkapja abG pontot, valamint Bob az Alicetől kapott aG pontot szorozza meg titkos b számával és eredményül ő is az abG pontot kapja. A közös pont valamely tulajdonsága (például x vagy y koordinátája vagy éppen $x + y$, x XOR y, stb.) használható kulcsként. A kíváncsi Eve-nek az abG pontot kellene kiszámolnia, de csak G, aG és bG pontokat ismeri, magukat a titkos a és b számokat nem. Az elliptikus Diffie-Hellman működését és lépéseit az alábbi egyszerű számpélda alapján követhetjük:

Nyilvános paraméterek	Legyen $E : y^2 \equiv x^3 + 5x + 8 \pmod{23}$ és $G(8, 10$
. titkos paraméterek:	Alice választ: $a = 7$
. Kulcselőkészítés:	Alice számol:
	$1G(8, 10)$
	$2G(13,4)$
	$3G(20, 9)$
	$4G(22, 18)$
	$5G(6, 1)$
	$6G(2, 18)$
	$7G(7, 15)$
	$\mathbf{aG = (7, 15)}$
.Kommunikáció:	Alice eredménye: $aG \rightarrow$
. Egyeztetett kulcs:	$1bG(20, 9)$
	$2bG(12, 18)$
	$3bG(7, 8)$
	$4bG(22, 5)$
	$5bG(8, 13)$
	$6bG(13, 4)$
	$7bG(6, 1)$
	$\mathbf{abG(6, 1)}$

	Bob választ: $b = 3$
	Bob számol:
	$1G(8, 10)$
	$2G(3, 4)$
	$3G(20, 9)$
	$\mathbf{bG = (20, 9)}$
	Bob eredménye: $\leftarrow bG$
	$1aG(7, 15)$
	$2aG(13, 19)$
	$3aG(6, 1)$
	$\mathbf{baG(6, 1)}$

6.2. ECElGamal-Elliptic Curve ElGamal titkosítás

Ahogy az eredeti ElGamal titkosítás a Diffie-Hellman algoritmus problémáján alapul, úgy építhető fel az elliptikus ElGamal is az ECDH-ra:

1.

Alice és Bob választ egy E görbét és egy G bázispontot.

2.

Mindketten választanak egy-egy véletlen a és b számot, mint titkos kulcsot.

3.

Alice elküldi az aG pontot, mint nyilvános kulcsot Bobnak.

4.

Bob elküldi a bG pontot, mint nyilvános kulcsot Alicenek.

5.

Ha Alice üzenni akar Bobnak, az üzenetet leképzi a görbe egy (vagy több) M pontjára, és generál egy véletlen k számot, mint viszonykulcsot. Elküldi Bobnak a $(kG, M + k(bG))$ üzenet párost.

6.

Bob a következőképpen olvassa el az üzenetet: a kapott küldemény első felét megszorozza saját titkos b számával, így bkG-t kap, amit egyszerűen kivon a küldemény második feléből.

6.3. Elliptikus görbén alapuló digitális aláírás, ECDSA-Elliptic Curve Digital Signature Algorithm

Ahhoz, hogy Alice egy M üzenetet aláírva el tudjon küldeni, következő paraméterek és eszközök szükségesek:

1.

egy elliptikus görbe $(\mathrm{mod}\ q)$ felett (nyilvános paraméter),

2.

egy G bázispont, melynek rendje n (nyilvános paraméter, $n \geq 160$ bit),

3.

egy véletlen d szám $(1 \leq d \leq n-1)$ és egy $Q = dG$ pont. Alice kulcspárja (d, Q), ahol d a titkos és Q a nyilvános kulcs.

7. Az aláírás algoritmusa

- Alice választ egy k számot 1 és $n-1$ között.
- Kiszámolja $kG = (x_1, y_1)$ pontot és $r = x_1 \pmod n$. Ha a pont x koordinátája zérus $(x_1 = 0)$, akkor új k számot választ. A pont x koordinátája lesz az aláírás egyik komponense, ezért jelöltük meg külön egy r betűvel.
- Kiszámolja k multiplikatív inverzét n-re $(k^{-1} \pmod n)$.
- Kiszámolja a küldendő üzenetpecsétjét, melyre a szabvány az SHA-1 algoritmust ajánlja. Legyen hát $e = \mathrm{SHA\text{-}1}(M)$ (számként értelmezve)!
- Az aláírás másik alkotóeleme: $s = k^{-1}(e + dr) \pmod n$. Abban szerencsétlen esetben, ha $s = 0$, akkor az egész algoritmust elölről kell kezdeni. Itt láthatjuk, hogy a 2. lépésben miért nem lehet $r = 0$, ekkor az aláírás nem tartalmazná a titkos kulcsot!
- Az M üzenethez és Alicehez tartozó aláírás: (r, s).

8. Feladatok

1.

Az $y^2 = x^3 - 36x$ egyenletű elliptikus görbén adottak a $P = (-3,9)$ és $Q = (-2,8)$ pontok. Határozzuk meg a $P + Q$ és $2P$ pontokat!

2.

Határozzuk meg a $P = (2,3)$ pont rendjét az $y^2 = x^3 + 1$ elliptikus görbén!

3.

Tekintsük az $Y^2 - 2Y = X^3 - X^2$ valós együtthatós elliptikus görbét és rajta a $P = (0,0)$ pontot. Határozzuk meg a $2P$ pont koordinátáit!

4.

Tekintsük az $Y^2 = X^3 + X + 5 \pmod{11}$ elliptikus görbét. A görbének egyik pontja a $P = (0,7)$.

Határozzuk meg a $2P, 3P, 4P, 5P, 6P, 7P, 8P, 9P, 10P$ pontokat.

5.

Az előző feladatban definiált elliptikus görbe segítségével küldjünk el egy üzenetet az ECElGamal módszer útmutatása szerint. Legyen a küldendő szöveg $M = (2,9)$, a bázispont legyen a $G = (0,7)$, $b = 3$ és $k = 6$.

Irodalomjegyzék

[1] M. Agrawal, N. Kayal, N. Saxena Primes is in P., Annals of Mathematics 160 (2004), 781–793.
[2] W. R. Alford, A. Granville, C. Pomerance, There are Infinitely Many Carmichael Numbers, Annals of Mathematics 140 (1994), 703–722.
[3] I. Blake, G. Seoussi, N. Smart, Elliptic curves in Cryptography, Cambridge University Press, 1999.
[4] Data Encryption Standard, Federal Information Processing Standards Publication, FIPS PUB 46-3, 1999. (http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf)
[5] D. Husemöller, Elliptic curves, Springer-Verlag, 1987.
[6] Iványi A. (szerk), Informatikai algoritmusok 1., ELTE, Eötvös Kiadó, 2004.
[7] N. Koblitz, A course in nuber theory and cryptography, Springer-Verlag, 1987.
[8] N. Koblitz, Introduction to Elliptic Curves and Modular Forms, Springer-Verlag, 1984.
[9] H. W. Lenstra, Jr., C. Pomerance, Primality Testing with Gaussian Periods, 2005.
[10] H. Lewis, C. Papadimitriou Elements of the Theory of Computation, Prentice-Hall, 1981.
[11] Jan C. A. Van Der Lubbe, Basic Methods of Cryptography, Cambridge University Press, 1998.
[12] A. Menezes, P. van Oorschot, and S. Vanstone Handbook of Applied Cryptography, CRC Press, 1996.
[13] J. M. Pollard, A Monte Carlo method for factorization, BIT Numerical Mathematics 15, (1975), 331–334.
[14] Márton Gyöngyvér, Kriptográfiai alapismeretek, Sciencia Kiadó Kolozsvár, 2008.
[15] C. Pomerance, A tale of two sieves, Notices Amer. Math. Soc. 43, (1996), 1473–1485.
[16] R. Rivest, R. Silverman, Are 'Strong' Primes Needed for RSA, Cryptology ePrint Archive: Report 2001/007.
[17] S. Singh, Kódkönyv, Park Könyvkiadó, 2007.
[18] Virasztó T., Titkosítás és adatrejtés, NetAcademia Kft., 2004.