Eötvös Loránd Tudományegyetem

Természettudományi Kar

Fejezetek a Bonyolultságelméletből

Szakdolgozat

Hrubi Nóra

Matematika Bsc

Matematikai elemző szakirány

Konzulens: Korándi József

Adjunktus

Budapest

2016

Tartalomjegyzék

1. Bevezetés

A szakdolgozatom témáját a Kódjátszma című film inspirálta, amit 2015 tavaszán mutattak be. A film megnézése után biztos voltam benne, hogy olyan témáról szeretnék írni, ami a kódok feltörésének matematikai módszereivel kapcsolatos. A pontos irányt azonban a konzulensem segítségével sikerült eldönteni, hogy a bonyolultságelmélet területén belül milyen témákat érintsek. Engem a kriptográfia rögtön megfogott, mert bára tanulmányaim során nem találkoztam vele, mégis úgy gondolom, hogy a titkosítás matematikája a hétköznapi élet szerves részét képezi.

Kriptográfia használata a mai világban már nélkülözhetetlen. Célja, hogy az egymástól távol élő emberek bizalmasan tudjanak üzenetet váltani. A történelem igazolta a titkosítás szükségességét, és az információ-technika fejlődésének köszönhetően ez a jövőben is nélkülözhetetlen lesz. A kódolt üzenetek használata kiterjedt minden híradástechnikai eszközre, mint a telefon, a rádió és később a számítógépek. Gondoljunk bele, hogy mi történne a bankokkal, ha illetéktelen kezekbe kerülnének az ügyfelek adatai, mert nem lennének titkosak. Bizalmas pénzügyi információk kiszivárgása komoly gazdasági problémát idézhetnek elő. A Kódjátszma című filmben is hasonló eset történt, ahol a hadi üzeneteket az ellenfél sikeresen fel tudta törni. Ilyen, és még ehhez hasonló érdekes problémákat fogok bemutatni és megoldani a dolgozatom során.

Dolgozatom felépítése deduktív, egy rövid történeti áttekintés után a kriptográfia megértéséhez szükséges alapfogalmak bemutatásával kezdem, majd egy konkrét titkosítási rendszert is bemutatok. Ezután konkrét példákon keresztülismertetem a modern kriptográfia problémáit. Végezetül egy konkrét alkalmazást mutatok be aszimmetrikus kulcsú titkosításra.

Dolgozatom megírásához elsősorban a konzulensem által ajánlott nyomtatott szakirodalmat és az Eötvös Loránd Tudományegyetemen dolgozó tanáraink könyvének interneten található digitális változatát használtam fel. Továbbá a téma mélyebb kutatása és megértése érdekében több angol nyelvű internetes szakirodalmat is feldolgoztam.

2. Bonyolultságelmélet alkalmazása: kriptográfia

2.1 Mi a kriptográfia?

A kriptográfia egy görög eredetű szó, ami a „kriptos”=eltitkolt, elrejtett és a „graphein”=írni szavakból állt össze, így magyarul titkosírásként fordították le. A kriptográfiához tartozik a matematika azon része, amely a titkosírásokkal, kódolással és azok előállításával, feltörésével foglalkozik. A 70-es évekig csak az üzenetek titkosításának módszereit értették alatta, viszont mára már jelentősen kibővült az egész információvédelem biztosítására, minden híradástechnikai eszközre, mint a telefon, rádióhullámok és a számítógépes hálózatok is. Az alapfeladat az információk illetéktelenek előli elrejtése, kódolása, illetve azok tárolása és továbbítása a megfelelő személy számára akár írásban, akár szóban. Cél, hogy csak az a fogadó fél tudja értelmezni az üzeneteket, aki ismeri a dekódolásukat, ugyanakkor mindenki más számára titkos, felfedhetetlen legyen. A titkosítás során az üzenetet rejtjelezik, és olyan rejtsorozattá alakítják, amelynek megfejtése lehetetlenül sok időbe tellene az illetéktelenek számára. Viszont a fogadó fél egy kulcs segítségével könnyen fel tudja fedni a rejtjelek jelentését, és a küldött üzenet ismert lesz számára.

Történetileg két részt különíthetünk jól el, az egyiket klasszikus kriptográfiának, a másikat pedig nyilvános kulcsú kriptográfiának nevezzük. A klasszikus kriptográfia története a 20. század közepéig tart. Ekkor még nem nagyon használtak fel matematikai módszereket, az emberek csak saját találékonyságukat és kreativitásukat felhasználva alakítottak ki különböző titkosításokat, amit nagy titokban tartottak egymás elől. Ezek általában az aktuális korhoz, történelmi eseményekhez igazodtak. Például az ókori görögök is használták a titkosítást. A saját ABC-jüket számokra cserélték, és minden betűnek egy szám felelt meg. Itt a kódolás még csak annyit jelentett, hogy az üzenet betűit számokra cserélték. Julius Caesar, a Római Császár is használta a titkosítást, ő azt találta ki, hogy a leírt latin betűit mind 3-mal (kulcs értéke: 3) előre csúsztatja az ABC-ben, így első olvasásra, az illetékteleneknek egy kódolt szöveget mutat, de néhány órás gondolkozás után, ezek könnyen megfejthetőek, hiszen minden betűnek egyetlen kódja van. Tovább nehezítve a kódolást, Caesar kitalálta a kulcsszavas titkosítást, ahol az előre megadott kulcsszót írta be a kulcs értékének a helyére, és onnan folytatta tovább az ABC kimaradt betűit. Ezt a behelyettesítéses módszert Caesar

általában katonai célokra használta fel. Most már egy számítógép segítségével ezeket könnyen meg lehet fejteni.

A nyilvános kulcsú kriptográfia annyiban tér el a klasszikus kriptográfiától, hogy itt ismert a titkosítási módszer és a titkosítási kulcsok, viszont az üzenet megoldásához szükséges kulcs, továbbra is titkos marad, amit csak az illetékes felek ismernek. A nyilvános kulcsú kriptográfia már matematikai módszereket használ fel, és a számítógépeknek is komoly kihívást jelent ezeket feltörni.

Továbbá szükségesnek tartom megemlíteni a szimmetrikus kulcsú kriptográfiát, ami egy másik felosztási irányból mutatja jól be a titkosítást. Ide sorolható az összes klasszikus módszer is, de mint ahogy a nevében is jól látható, ez a módszer szimmetrikus, vagyis mind a Küldő, mind a Címzett félnek ismernie kell a titkosításhoz használt kulcsot. Ez annyit jelent, hogy lényegében ugyanazzal a kulccsal titkosítunk, mint amivel fejtünk. Később ennek kiküszöbölésére sikerült létrehozni az aszimmetrikus kulcsú kriptográfiát, ahol a titkosítási kulccsal már nem tudjuk megfejteni az üzenetet. Tehát van egy kulcsunk az üzenet titkosítására, és van a Fogadó félnek is egy kulcsa, az üzenet megfejtésére. Az egyik legismertebb aszimmetrikus kulcsú kriptográfiai eljárás az RSA kódolás, ami az 1970-es években született.

2.2 Alapfogalmak

- **Nyílt szöveg** (plain text): Az eredeti érthető szöveget vagy üzenetet, amit védeni szeretnénk.
- **Titkosított szöveg** (cypher text): A titkosítással átalakított szöveg.
- **Titkosítás**: Eljárás, melynek során az algoritmus a nyílt szöveget átalakítja egy titkosított szöveggé kódolással vagy rejtjelezéssel.
- **Titkosító kulcs**: A titkosító algoritmus alkalmazása során a nyílt szöveg titkosítottá tételére titkosító kulcsot használunk.
- **Megfejtő kulcs**: A titkosított szöveget a megfejtő kulcs segítségével tudjuk visszaalakítani nyílt szöveggé. A kulcs egy paraméter, amit a titkosító eljárás során csak a küldő és fogadó fél ismer.
- **Megfejtés**: A fogadó félnek egyértelműen meg kell tudnia fejteni a nyílt szöveget a rejtjelezett üzenetből. Ezt a folyamatot dekódolásnak, vagy megfejtésnek hívjuk.

- **Feltörés**: A nyílt szöveg kulcs ismerete nélküli visszanyerését a titkosított szövegből feltörésnek hívjuk.

Az általunk vizsgált esetekben feltételezzük, hogy a titkosított szöveg ismert mindenki számára, viszont azt szeretnénk, hogy a jelentését csak a címzett tudja megfejteni. A megfejtő algoritmus egy függvényként is értelmezhető, amit csak az illetékes felek ismernek. Így általában ha a többi fél ismeri is a rejtjelező algoritmust, akkor se tudják megfejteni a kódot, mert szükségük lenne hozzá a kulcsra és a konkrét függvényre, amivel a nyílt szöveget megkapnák.

2.3 Klasszikus kriptográfiai probléma

Alapproblémánkat úgy tudnánk egyszerűen megfogalmazni, hogy van egy üzenet, amit a Feladó el akar juttatni a Címzettnek. Legyen az üzenetünk m, ami egy l hosszúságú 0-1 sorozat. Célunk olyan üzenet küldése, amit csak a fogadó fél ért meg, senki más. Ehhez először kódolnunk kell az üzenetet. A Feladó nem az m üzenetet küldi meg a Címzettnek, hanem a hozzá tartozó szintén l hosszú c kódját. Tehát a Feladó először a k kulcs alapján kódolja a küldeni szánt üzenetét, és csak ezt a kódot küldi meg a Címzettnek. Így a kapott kódot már vissza tudja fejteni a Címzett, mert ő ismeri a hozzá tartozó kulcsot. Számára ismert lesz az üzenet, viszont senki másnak nem. Az eredeti üzenet visszafejtéséhez a k kulcsot használ, ami lehet például szintén egy l hosszúságú 0-1 sorozat. Ezt a kulcsot kizárólag a Feladó, és a hozzá tartozó illetékesek tudhatják, biztosítva, hogy az üzenet más számára ne legyen ismert.

Ha a Feladó rendelkezik egy m üzenettel, egy k kulccsal, akkor a kód az alábbi formulával írható fel:

$$c = f(m,k) \quad (l \text{ hosszúságú 0-1 sorozat})$$

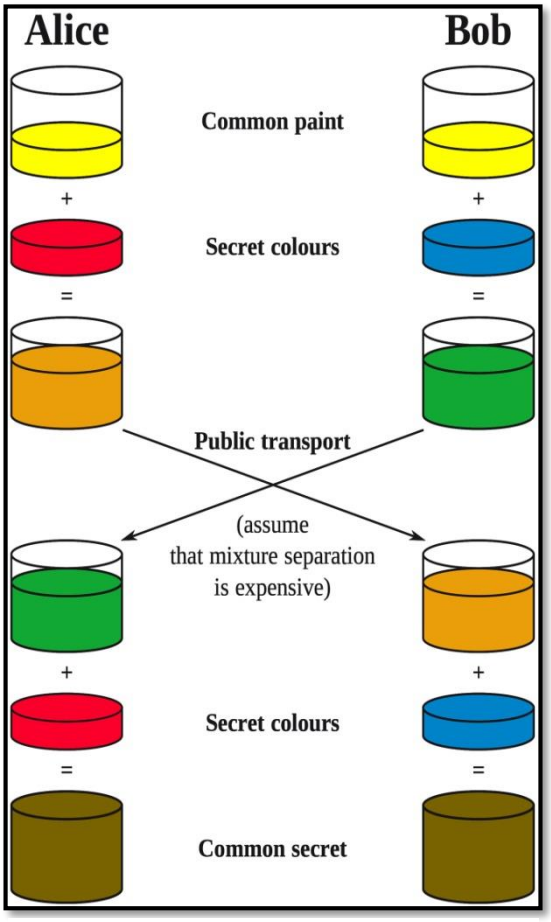
A c kód kiszámításához először fel kell tennünk, hogy minden rögzített k -ra $f(\cdot, k)$ bijektíven képezi le $\{0,1\}^l$ -t önmagára. Ekkor létezik $f^{-1}(\cdot, k)$, így ha a Címzett ismeri a k kulcsot, akkor segítségével könnyen visszanyerheti az m üzenetet. Egy gyakran használt f függvény az: $f(m,k) = m \oplus k$ (bitenkénti összeadás modulo 2), ami látható módon ekkor $f^{-1}(\cdot, k) = f(\cdot, k)$.

A következő részben az első publikált nyilvános kulcsú módszert fogom bemutatni, ami egy praktikus eljárás a titkos kulcsok nyilvános cseréjére.

2.4 Diffie – Hellman kulcscsere

Whitfield Diffie és Martin Hellman 1970-es években hozta létre a nyilvános kulcscserélő algoritmust. Ezzel az eljárással, egy olyan algoritmust akartak létrehozni, ahol nyilvános a titkosítási módszer, és nincs szükség előre egyeztetett információkra. Tehát mindenki számára ismert a nyilvános kulcs a titkosításhoz. Viszont a titkos kulcsot csak az illetékesek ismerik a visszafejtéshez. Titkos kulcs ismerete nélkül nem fejthető vissza az üzenet, de a nyilvános kulcsból se lehet visszafejteni a titkos kulcsot, még ha ismerik is az létrehozásához szükséges algoritmust.

Az általános algoritmus alapgondolatát színek keverésével is szemléltethetjük. Tegyük fel, hogy van két fél, legyen Alice és Bob, akik szeretnének egy azonos színű festéket létrehozni, amelyet más nem ismer, de távol vannak egymástól. Megegyeznek egy tetszőlegesen választható kiinduló színben. Ez nyilvános mindenki számára. Például legyen mindkettőjük kiválasztott színe a citromsárga. A második szinten mind a ketten kiválasztanak egy titkos színt, amit senki más nem ismerik. Legyen ez Alice-nál piros, Bobnál zöld. Most jön a folyamat lényege, ahol Alice és Bob a nyilvános színeiket összekeverik a titkos színeikkel. Tehát Alice a citromsárgát pirossal, ami narancs keverék lesz, és Bob a citromsárgát zölddel, ami a kék keverék lesz. Ezek után nyilvánosan kicserélik a két színkeveréket. Tehát Alice-hoz kerül a kék keverék, Bobhoz pedig a narancs színkeverék. Utolsó lépésként pedig mind a ketten ismét hozzáadják a titkos színeiket, amivel megkapják a végleges színelegyet, ami azonos színű lesz mind a kettejüknek.



Diffie-Hellman kulcscsere[3]

Egy harmadik fél számára lehetetlen lenne meghatározni a közös titkos színt, mert ő csak a keverékeket látja, és a narancs, illetve zöld színekből nem tudjuk megállapítani a piros, illetve kék pontos árnyalatát, még a sárga ismeretében sem.

Matematikailag ez a következőképpen valósítható meg: Legyen p egy prímszám, q pedig relatív prímpárja a p-nek. Például Alice és Bob a következő értékekben egyeznek meg, amik nyilvánosak mindenki számára:

$p = 23$ $q = 5$ (ami relatív prímpárja a 23-nak)

Alice ezek után választ egy titkos számot, $a = 6$.

Ezek után Bobnak elküldi azt az A számot, melyre

$$A \equiv q^a \bmod p \text{ és } 0 \leq A < p$$

(Erre a továbbiakban az $A = q^a \bmod p$ jelölést használom)

$$A = 5^6 \bmod 23 = 8$$

Bob is kiválaszt egy titkos egész számot, $b = 15$,

majd elküldi Alice-nak a $B = q^b \bmod p$ számot:

$$B = 5^{15} \bmod 23 = 19$$

Alice kiszámolja a $s = B^a \bmod p$, ahol

$$s = 19^6 \bmod 23 = 2$$

(s értéke titkos, csak Alice és Bob ismerik).

Bob is kiszámítja az s értékét, ami$s = A^b \bmod p$, ahol

$$s = 8^{15} \bmod 23 = 2$$

Alice és Bob közös titkos száma az $s = 2$.

Mind a ketten ugyan azt a számot kapták meg, mert közös a modulus értéke (p). Ezt a közös kulcsot fogják használni a kommunikáció során.

Összegezve általánosan:

$$A^b \bmod p = q^{ab} \bmod p = q^{ba} \bmod p = B^a \bmod p$$

Tehát:

$$(q^a \bmod p)^b \bmod p = (q^b \bmod p)^a \bmod p$$

Érdemes megjegyezni, hogy csak az a, b és $q^{ab} \bmod p (= q^{ba} \bmod p)$ a titkosak, minden más érték, p,q,$q^a \bmod p$, és $q^b \bmod p$ nyilvánosak mindenki számára.

Alice-nak és Bobnak egyszer kell csak kiszámolniuk a közös titkos kulcs értékét, ennek birtokában később a kommunikációjukat már egy nyilvános csatornán keresztül bonyolíthatjuk.

Az egyszerűbb számolás érdekében én kis számokat használtam, és rövid egyéni kulcsokat, de az algoritmus biztonságos használatához természetesen sokkal nagyobb értékekre lenne szükség. Például ha egy legalább 600 számjegyű p prímet választunk ki, akkor még a leggyorsabb modern számítógépek sem fogják tudni p, q és $q^a \bmod p$ ismeretében a-t kiszámolni. Ezt a problémát diszkrét logaritmus problémának hívják. A q^a kiszámítása mod p–ben gyors, azaz ismert a moduláris hatványozás, de a-t mégse lehet hatékonyan kiszámolni nagy számokra. Ehhez nem kell, hogy q értéke nagy legyen, gyakorlatban általában ez egy kis szám (például 2,3,..).

Az algoritmus kettőnél több fél esetén is jól működik. Tetszőleges számú felhasználó is részt vehet, közös megállapodások végrehajtásával és a megfelelő adatok cseréjével.

Számelméleti alapként szolgált a Diffie-Hellman kulcscsere a nyilvános kulcsú titkosításhoz.

3. Nyilvános kulcsú kriptográfia

3.1 Nyilvános kulcsú kriptográfia

A következő általam bemutatott rendszer a nyilvános kulcsú kriptográfia lesz, ami több módon is javítja a klasszikus kriptográfiai módszereket. A rendszer a mindennapi élet egyszerűbb titkosítási problémáira is ad elektronikus megoldást. Vegyük például az elektronikus postát. A hagyományos postán jelen van az ügyfél, így nem okoz problémát egy aláírás, vagy a boríték megírása, feladása. Viszont a mi célunk, ezen eszközök megoldása elektronikus úton.

Mint ahogy a postán keresztül való levelezés estén, itt is legalább 2 szereplőre van szükség a rendszer működéséhez. Jelöljük a szereplők számát P-vel, ahol most $P = 2$. Minden szereplőhöz tartozik két kulcs, az egyik a nyilvános kulcs, ami mindenki számára elérhető, a másik pedig a titkos kulcs, amit csak saját maga ismer. Legyen a j-edik szereplőhöz tartozó nyilvános kulcs k_j, a titkos kulcsa pedig s_j. Továbbá az üzenet kiszámításához szükségünk van egy kódoló/dekódoló függvényre, amit mindenki ismer. Így ha van egy m üzenetünk, és rendelkezünk a hozzá szükséges (titkos vagy nyilvános) k kulccsal, akkor a következő függvény $f(m, k$ segítségével könnyen kiszámítjuk az üzenetünket. (Legyen $x, k \in$ H halmaznak, ahol H egy egyszerűen megadható halmaz legyen, mint például: $\{0,1\}^l$, vagy akár a modulo h maradékosztályok halmaza) Továbbá feltételezzük, hogy az m üzenetünk tartalmazza a küldési dátumot, valamint a Feladó és a Címzett neveit. Az alábbi összefüggés teljesül minden $m \in H$-ra és minden $1 \leq j \leq P$ esetén:

$$f\big(f\big(m, \mathrm{k_j}\big), \mathrm{s_j}\big) = f\big(f(m, \mathrm{s_j}\big), \mathrm{k_j}) = m$$

Tehát ezzel a rendszerrel küldhet az egyik szereplő a másik szereplőnek üzenetet. Jelen esetben az müzenetet szeretné a j-edik szereplő elküldeni az $(j + 1)$-edik szereplőnek. Ekkor az üzenet helyett a rendszer az

$$n = f(f(m, \mathrm{s_j}), \mathrm{k_{j+1}})$$

üzenetet fogja elküldeni a $(j + 1)$-edik szereplőnek. Ebből az eredményből a $(j + 1)$-edik szereplő már ki tudja számolni az eredeti üzenetét az alábbi képlettel:

$$m = f(f(n, \mathrm{s_{j+1}}), k_\mathrm{l})$$

A rendszer használhatóságához további bonyolultsági feltételek teljesülésére van szükségünk:

(1) Ha ismerjük az $m$üzenetet és a hozzá tartozó $k$nyilvános kulcsot, akkor az $f(m, k)$ függvény „gyorsan” kiszámítható

(2) Adott$m$és $k$melletttetszőleges számús$_\mathrm{i}$, $\mathrm{i} \in \{1,2, \dots, \mathrm{P}\}$, $\mathrm{i} \neq$ jtitkos kulcsot ismerünk, akkor az $f(f(m, k_\mathrm{j}), \mathrm{s_j}) = m$ nem számítható ki „gyorsan” (nem számolható ki polinomiális időn belül).

Az (1)-es feltételben biztosítjuk, hogy a rendszerben lévő függvény kiszámítása polinomiális időn belül megtörténjen, így a Címzett képes lesz megfejteni a neki szánt üzenetét

A (2)-es feltétel pedig biztosítja, a rendszer biztonságos működését. Pl. ha egy $m$üzenet a j-edik szereplő nyilvános kulcsával lett kódolva, (de ő elvesztette az eredeti kulcsát) akkor az ő s_j kulcsa nélkül kódolt üzenetből a résztvevők semmilyen együttműködéssel sem fogják tudni újra létrehozni az eredeti szöveget.

Állítás: A$(j + 1)$-edik szereplőnek szóló üzenetet csak ő maga tudja megfejteni.

Bizonyítás: Tegyük fel, hogy a $\mathrm{sz_1}, \mathrm{sz_2}, \dots, \mathrm{sz_i}$ az illetéktelen szereplők egy csoportja megtudja az üzenetet és még azt is, hogy ki küldte ezt az üzenetet és kinek. Ekkor az

$$m = f(f(m, \mathrm{s_j}), \mathrm{k_{j+1}})$$

üzenetet eredményesen ki tudják számolni. Tehát az illetéktelen szereplők csoportja aj-edik szereplővel együtt ki tudja számolni az $n = f(m, \mathrm{k_{j+1}})$-t, amiből ki tudják számolni az $m = f(n, \mathrm{s_{j+1}})$-t. Ha az$o = f(m, \mathrm{k_i})$ üzenetet ismerjük, akkor ismerjük az

$$n = f(m, \mathrm{k_{j+1}}) = f(f(o, \mathrm{s_j}), \mathrm{k_{j+1}})$$

üzenetet is. Így ha az $\mathrm{sz_1}, \mathrm{sz_2}, \dots, \mathrm{sz_i}$ eljárást alkalmazzuk, akkor végül ki fogják tudni számolni az o-t. De ha már kiszámolták az o-t, akkor a j-edik szereplő segítségével ki fogják tudni számolni m-et is, ahol $m = f(o, \mathrm{s_{j+1}})$. Ez viszont ellentmondása a (2)-es feltételnek (ha a$(j + 1)$-edik szereplő nem volt a $\mathrm{sz_1}, \mathrm{sz_2}, \dots, \mathrm{sz_i}$ szereplők között).

Néhány további állítás, bizonyítás nélkül:

Állítás: Ha az üzenetet a szereplők egymás után küldik tovább egy előre meghatározott sorrendben. A j-edik szereplő küldi tovább az üzenet a$(j+1)$-edik szereplőnek, így azt az üzenetet senki nem tudja meghamisítani, vagyis a $(j+1)$-edik szereplő biztos lehet abban, hogy a kapott üzenetet a j-edik szereplő küldte.

Állítás: A$(j+1)$-edik szereplő bizonyítani tudja, ha szükséges, hogy milyen üzenetet kapott a j-ediktől, viszont a titkos kulcsok (s_{j+1}, s_j) elemeit nem kell felfedni hozzájuk.

Állítás: A$(j+1)$-edik szereplő nem tudja az üzenetet megváltoztatni, vagy más nevében másnak tovább küldeni.

A két feltételnek eleget tevő rendszer létezése továbbra sem biztos. A rendszert csak számelméleti bonyolultsági feltételek megadásával sikerült megadni, ami így biztonságosnak mondható. A következő fejezetben egy igen népszerű, gyakran használt rendszert mutatok be.

3.2Aszimmetrikus kulcsú titkosítás

A szimmetrikus kulcsú titkosítás során két számítógépen történik a titkosítás, ahol azonos a kódoló és dekódoló értéke. Viszont az aszimmetrikus kulcsú titkosítás során nem azonos a két kulcs értéke, és nem is lehetséges belátható időn belül egyikből kiszámolni a másik értékét.

Jelöljük a két számítógépet A-val és B-vel (Alice és Bob). Azoknál a rendszereknél, ahol egynél több résztvevő van, ott a nyilvános kulcsokat egy kulcstárban helyezik el. A kulcstárhoz mindenki hozzá tud férni, így azt bárki olvashatja. Tehát ha Alice egy üzenetet szeretne küldeni Bobnak, akkor Alice a nyilvános kulcstárból ki tudja olvasni a Bobnak küldendő üzenetek kódoló kulcsát, majd ennek felhasználásával tudja a kódolt üzenetet elküldeni Bobnak. Bob megkapja a titkosított üzenetet, majd annak dekódolásával megfejti az üzenetet. A dekódoló kulcs titkos, amit csak a B fél ismer.

Tehát a titkosítás minden nyilvános kulcsa elérhető egy központi kulcstárban, így nagyon könnyen kódolhatunk bármit. Viszont a dekódoló kulcsot nem ismerjük, és nem is tudjuk kiszámolni a nyilvános kulcsok segítségével sem, így ez egy gyakorlatilag nem megfejthető üzenet lesz a kívülállók számára.

További előnye az aszimmetrikus kulcsú titkosításnak, hogy nincs szükséges előre megbeszélt információkra. Nem kell a Feladónak előre egyeztetnie egy titkos kulcsot a Címzettel. A titkos kommunikációhoz csak egy dologra van szükségük. A Feladónak meg kell nézni a nyilvános kulcstárban a Címzetthez tartozó nyilvános kulcsot, és ezzel a nyilvános kódoló kulccsal rejtjelezni az üzenetet. Fontos megjegyezni, hogy a Feladó valóban a Címzett nyilvános kulcsát használja, és ne másét, mert különben az nem fogja tudni megfejteni. Ezzel leegyszerűsíti a nyilvános kulcsú kriptográfia a kulcscsere-problémát, hiszen nem egy titkos csatornát használ, hanem egy nyílt csatornát. Ami lehetőséget ad arra, hogy a Feladó eljuttathassa a nyilvános kulcsát a Címzettnek.

3.3RSA titkosítási rendszer

Ebben az alfejezetben az RSA titkosítási rendszert fogom bemutatni. Az RSA rövidítés Rivest-Shamir-Adleman, a3 alkotó neveinek kezdőbetűiből állt össze 1977-78-ban. Ez az egyik leggyakrabban használt nyilvános kulcsú algoritmus, és itt szerepelt először nyilvános kulcs.

Alapötletünk, hogy legyen két elég nagy *n* számjegyű prímünk, jelöljük őket a-val és b-vel. E két nagy prímszám szorzata legyen $h = a \cdot b$. A rendszer ezt ah számot fogja nyilvánossá tenni. Továbbá egy k számot is nyilvánossá tesz, amit úgy hozunk létre, hogy az $(a-1)$-hez és $(b-1)$ -hez viszonyítva is relatív prímet adjon, és k-ra az is igaz legyen, hogy $1 \leq k \leq h$. Relatív prímet úgy is konstruálhatunk, hogy 0 és $(a-1)\cdot(b-1)$ között keresünk véletlen módszerekkel egy prímet. A megtalált prímet ezek után ellenőrizni kell, hogy relatív prím-e$(a-1)\cdot(b-1)$-hez, amihez az euklideszi algoritmust használjuk. Ha a megtalált prím nem relatív prím, akkor újabb prímet keresünk az adott intervallumon, amíg meg nem találunk egy megfelelő relatív prímet $(a-1)\cdot(b-1)$-hez. A folyamat véges, mert tudjuk,hogy n számjegyű prímek esetén, a kereséshez log(n) ismétléssel elég nagy valószínűséggel fogunk találni egy megfelelő $k$számot. További számolással megadhatunk az euklideszi algoritmus felhasználásával egy olyan $1 \leq s_j \leq h-1$ számot, amelyre a következő feltétel teljesül:

$$k_j s_j \equiv 1 \, mod(a-1)\cdot(b-1).$$

A j-edik szereplőhöz tartozó titkos kulcsot s_j –vel, a nyilvános kulcsot pedig k_j-vel jelöljük. Az üzenet hosszáról feltesszük, hogy egy $h$hosszú természetes szám, különben ha nem így lenne, akkor feldarabolással h hosszúságú darabokra vágjuk az üzenetet, a maradékot pedig feltöltjük annyi karakterrel, hogy h hosszúságú legyen. Üzenetünk kódolásához egy függvényt használunk, amit a következőképpen adunk meg:

$$f(m,k) \equiv m^k (mod\ h),$$

ahol $0 \leq f(m,k) < h$.

A feltöréshez viszont a következő függvényt használjuk:

$$f(m,s) = m^{\mathrm{s}} (mod\ h\ ),$$

ahol $0 \leq f(m,s) < h$.

Euler-Fermat tétele miatt a kódolás inverze a dekódolás.

Az Euler-Fermat tétel szerint:

$$k_{\mathrm{j}} s_{\mathrm{j}} = 1 + \varphi(h)t = 1 + t(a-1)(b-1),$$

ahol t természetes szám.

Így ha $(m,a) = 1$, akkor

$$f\big(f\big(m,k_{\mathrm{j}}\big)\mathrm{s_j}\big) \equiv (m^{\mathrm{k_j}})^{\mathrm{s_j}} = m^{\mathrm{k_j}\ \ \mathrm{s_j}} = m(m^{\mathrm{a-1}})^{\mathrm{t(b-1)}} \equiv m\ (mod\ a).$$

Másrészt ha $a|m$, akkor:

$$m^{\mathrm{k_j}\ \ \mathrm{s_j}} \equiv 0 \equiv m(mod\ a).$$

Így tehát minden m-re felírhatjuk az alábbi összefüggést:

$$m^{\mathrm{k_j}\ \ \mathrm{s_j}} \equiv m\ (mod\ a)$$

Hasonlóan felírhatjuk

$$m^{\mathrm{k_j}\ \ \mathrm{s_j}} \equiv 0 \equiv m\ (mod\ b),$$

ezért$m^{\mathrm{k_j}\ \ \mathrm{s_j}} = m(mod\ h)$.

Feltevésünkből tudjuk, hogy minden m szám 0 és h-1 között található, így az első és utolsó számunk is biztos, hogy ebbe az intervallumban található meg. Ebből viszont azt is következtethetjük, hogy ezek egyenlők, azaz:

$$f\big(f\big(m,k_j\big)s_{\mathrm{j}}\big) = f\big(f\big(m,\mathrm{s_j}\big)k_{\mathrm{j}}\big) = m$$

Az (1)-es feltétel könnyen ellenőrizhető, mert adott az m üzenet, $\mathrm{k_j}$ nyilvános kulcs és a h szám, ekkor az $m^{\mathrm{k_j}}$-nak$h$-val vett osztási maradéka polinomiálisan kiszámítható. A (2)-es feltételnek is teljesülnie kell, ám az csak egy gyengébb változatban áll fent:

(2*) Adott az $m$üzenet és a k_j nyilvános kulcs, így viszont nincs elég adatunk $f(m, s_j)$ polinomiális időben történő kiszámításához. Ezt a feltevést jelenleg még nem tudjuk bizonyítani, és ebben erősít meg minket a számelmélet is.

A fent leírt RSA kódunk viszonylag egyszerűnek mondható, de mégis néhány dolgot meg kell vizsgálnunk vele kapcsolatban. Elsődleges problémánk a „posta", ahol ismert az a, b és a titkos kulcs, s_j is. Ezek ismeretében viszont fel tudja törni az összes üzenetet. Továbbá a legnagyobb problémát az okozza, hogy a„posta" bármelyik résztvevőjét is nézzük, fel fogja tudni törni bármely más résztvevőtől küldött üzenetet.

Vizsgáljuk meg azt az esetet, amikor a $(j+1)$-edik résztvevő megkapja az l-edik résztvevő üzenetét, méghozzá az $o = f(f(m, s_j) \cdot k_l)$-t, és jelölje $n = f(m, s_j)$ az üzenetet. A $(j+1)$-edik résztvevő tehát ismeri az $o$és $n$üzenetet, amiből már ki tudja számolni a számára küldött üzenetet. Először is ki kell számolni $(k_{j+1} \cdot s_{j+1} - 1)$ szorzattá bontását (legyen $u \cdot v$), ahol az $(u, k_l) = 1$, és v minden prímosztója, k_l -nek is. A számolást először euklideszi algoritmussal kell kezdeni, ami kiszámítja a két szám k_l és $(k_{j+1} \cdot s_{j+1} - 1)$ számok legnagyobb közös osztóját, v_1 -et. Ezek után kiszámoljuk a v_2 -t, ami a k_l és $(k_{j+1} \cdot s_{j+1} - 1)/v_1$ legnagyobb közös osztója lesz, majd a v_3 -at, ami a k_l és $(k_{j+1} \cdot s_{j+1} - 1)/(v_1 \cdot v_2)$ legnagyobb közös osztója lesz, és így tovább. Az eljárás véges időben véget ér, amíg $v_t = 1$, ehhez viszont legfeljebb $t = \lceil \log(k_{j+1} \cdot s_{j+1} - 1) \rceil$ lépésre van szükségünk. Ezzel sikeresen szorzattá bontottuk $(k_{j+1} \cdot s_{j+1} - 1)$ számunk, ahol

$$v = v_1, v_2 \dots v_t \qquad \text{és} \qquad u = (k_{j+1} \cdot s_{j+1} - 1)/v.$$

Számításaink alapján további következtetéseket vonhatunk le. Először is tudjuk, hogy $(\varphi(h), k_l) = 1$, amiből következik $(\varphi(h), v) = 1$. Másfelől $\varphi(h) | k_{j+1} \cdot s_{j+1} - 1 = u \cdot v$, tehát akkor: $\varphi(h) | u$. Ha $(u, k_l) = 1$, akkor a $(j+1)$-edik résztvevő ki fog tudni számolni az euklideszi algoritmus felhasználásával olyan s és t természetes számokat, melyekre a következő egyenlet igaz: $sk_l = tu + 1$. Ennek függvényében viszont tudjuk, hogy:

$$o^s \equiv n^{sk_l} \equiv n^{1+tu} \equiv n(n^u)^t \equiv n \pmod{h},$$

és ekkor:

$$m \equiv n^{k_j} \equiv o^{s\, k_j}$$

Ezek alapján a $(j+1)$-edik résztvevő ki fogja tudni számolni az m üzenetet.

3.3.1 RSA, Digitális aláírás

A digitális aláírás egy nyilvános kulcsú algoritmus. Eddig mindig csak az üzenetünk titkosításával foglalkoztunk, és nem foglalkoztunk a Feladóval vagy a Címzettel. Elsődleges céljaa digitális aláírásnak az volt, hogy felváltsa a hagyományos kézzel írott aláírást, hogy az informatika világában ez helyettesíthető legyen. A digitális aláírással tudjuk igazolni az elektronikusan küldött üzenet készítőjét és az üzenet tartalmát. Gyakran összekeverik az elektronikus aláírással, vagy nem is tudják van-e különbség a kettő között. Az elektronikus aláírás viszont nem más, mint a hagyományos aláírásunk elektronizált változata. A digitális aláírás egy szám, ami jelentősen függ az üzenettől, annak tartalmától és létrehozójától. Ezek függvényében hoz létre valamilyen matematikai algoritmust a digitális aláírás. Így biztosítjuk, hogy a Címzett biztos lehessen benne, hogy az adott aláírás kitől származik az üzenet végén.

A digitális aláírás előállítására aszimmetrikus kriptográfiai módszereket alkalmaznak. Ebben az esetben mindenkinek van egy nyilvános és egy titkos kulcsa. Gyakran így biztosítják a küldőnek a biztos és érvényes üzenetküldést egy nem biztonságos csatornán keresztül, hogy a küldött üzenetet valójában a Feladó küldte, és nem egy illetéktelen személy. Az aláíró félnek titokban kell tartania a titkos kulcsát, amit a saját biztonsága érdekében jó ha nem ad ki másnak. Viszont a nyilvános kulcsot közzé teheti, gyakran erre még szükség is van, hogy sok esetben ezzel érvényesíthető egy digitális aláírás, az üzenetre és az aláíró személyre hivatkozva. A digitális aláírás egyenértékű sok esetben a hagyományos kézi aláírással, de ha megfelelően hozzák létre a digitális aláírást, akkor nehezebb egy digitális aláírást hamisítani, mint a kézzel írott típust. A digitális aláírás kriptográfiai elemeket használ annak érdekében, hogy minél hatékonyabb legyen. A digitális aláírásnak biztosítania kell azt is, hogy az aláíró fél ne tudja letagadni a saját aláírását, amíg tudja, hogy a titkos kulcsot csak ő ismeri.
Sok területen alkalmazzák a digitális aláírást, mint például pénzügyi tranzakciók lebonyolításánál, elektronikus levelezésnél, szerződéseknél és sok más esetekben, ahol fontos elkerülni a hamisíthatóságot.

Néhány alapkövetelmény a digitális aláírással szemben:

- Hiteles aláírás: biztosítja az aláírás a dokumentum olvasóit, hogy a dokumentumban szereplő adatok valóban onnan származnak, ahonnan arra számítanak
- Hamisíthatatlan: az aláírás biztosítja, hogy valóban az a személy írta alá, és nem más

- Más dokumentumon nem használható fel az aláírás: a digitális aláírás nem helyezhető át másik dokumentumra
- Adatintegritás: biztosítja az olvasóknak, hogy az adatok nem lettek megváltoztatva illetéktelenek által
- Letagadhatatlan: az aláíró fél nem tagadhatja le későbbiekben, hogy ő írta alá

A digitális aláíró sémák általában három algoritmusból állnak.
Az első a kulcsot létrehozó algoritmus, ami először véletlenszerűen kiválaszt egy titkos kulcsot az előre megadott feltételű titkos kulcsok közül. Az algoritmus kimenetele a titkos kulcs és a hozzá megfelelő nyilvános kulcs lesz.
A második az aláírást létrehozó algoritmus. Az adott üzenet és a titkos kulcs tekintetében létrehoz egy aláírást.
A harmadik pedig az aláírást ellenőrző algoritmus. Itt már van üzenetünk, nyilvános kulcsunk és aláírásunk. Ezeket felhasználva ellenőrizzük, hogy bármelyik visszautasítja-e az üzenet igényét hitelességre vagy elfogadja őt.
Hozzunk létre egy digitális aláírást a RSA algoritmus segítségével. Ezzel az algoritmussal nagyon könnyen megérthető a módszer lényege. Az előző fejezetben használt rövidítéseket fogom itt is használni. Feladó generál először is két nagy prímet$a$ -t és b-t, méghozzá oly módon, hogy ki tudja számolni a $C = m^s (mod\ h)$ értéket, ahol s a titkos kulcs, és m az üzenet. Ezt az üzenetet elküldi a Címzettnek. Ha a Címzett tudja már a Feladó nyilvános k kulcsát, akkor ennek segítségével könnyen megfejti az üzenetet.

$$C^k (mod\ h) = (m^s)^k (mod\ h) = m$$

Ha a fenti képlet kiszámolásával az üzenetet vagy egy értelmes szöveget kap vissza a Címzett, akkor biztos lehet abban, hogy ez a Feladótól jött. Mint ahogy írtam már, ez nem túl biztonságos módszer, mert nyilvános kulcs ismeretében bárki, aki hozzájut a C üzenethez, meg tudhatja m-et, így ezt nem nevezhetjük még titkosításnak. Az egész üzenet titkosítása viszonylag nehéz feladat lenne, és időigényes. Ezért csak egy részét titkosítjuk az üzenetnek, egy egyedi kivonatát.
Ezt a kivonatot üzenetpecsétnek (message digest, MD), vagy hash-függvénynek nevezzük. Ezzel sokkal hatékonyabbak leszünk, mert rövidebb lesz az aláírás, így időt takaríthatunk meg az üzenetpecsét használatával, ami a gyakorlatban is gyorsabb lesz. Ezek az algoritmusok azért népszerűek, mert a tetszőleges hosszúságú bitsorozatból fix hosszúságú bitsorozatot adnak vissza. Így az üzenetünk lényegét egy ilyen rövid bitsorozat fogja képviselni.

A dokumentum hitelességét a digitális aláírás igazolja. A Feladó a dokumentumból képez egy ellenőrző számot, $MD(m)$-et.Az $MD(x)$ függvény mindenki, így a Címzett számára is ismert. (Például: betűk, számjegyek, írásjelek kódjainak összege). Ha már megvan ez az egyedi azonosító, akkor innentől kezdve nagyon gyorsan alakul a folyamat. Először is kiszámoljuk az S ellenőrző összeget, $S = (MD(m))^s (mod\ h)$. A Feladó ezt az S értéket küldi meg a Címzettnek az üzenettel együtt, vagyis az (m, S) párt küldi el. A Címzett ellenőrizni tudja a dokumentumot,ehhez csak ismét ki kell számítania az üzenet egyedi számát, az$MD(m)$-et és a Feladó által küldött Sérték alapján vissza kell fejtenie az egyedi azonosítót: $Z = S^k\ (mod\ h)$. A dokumentum abban az esetben lesz hiteles, ha $Z = MD(m)$, és ekkor az is biztos, hogy a dokumentum a Feladótól származik, és nem illetéktelentől. Vagyis a dokumentum változatlanul jutott át a csatornán. Ebben az esetben *m*-et nem titkosítottuk, vagyis a módszer ilyen formában csak akkor alkalmazható, ha az üzenet nem tartalmaz érzékeny információt, csak a küldő személyében szeretnénk biztosak lenni, és abban, hogy nem változtatták meg az üzenetet.

3.4Feladatok

1. Titkosítsa az alábbi nyílt szöveget Caesar titkosítással, ahol a kulcs értéke 5.

Továbbá feltételezzük, hogy az angol ABC 26 betűjét használhatjuk.

A KOCKA EL VAN VETVE

A titkosító tábla használatához az angol ABC betűit használjuk:

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
F	G	H	I	J	K	L	M	N	O	P	Q	R

13	14	15	16	17	18	19	20	21	22	23	24	25
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
S	T	U	V	W	X	Y	Z	A	B	C	D	E

(Az angol ABC betűihez tartozó számkódot 0-val kezdem, mert ha a kulcs=0, akkor nincs titkosítás)

Ha a kulcs 5, akkor a következő titkosított szöveget kapjuk:

F PTHPF JQ AFS AJVAJ

Azaz minden egyes betűt az angol ABC-ben 5-tel eltolja. Így a kódolt üzenetet könnyen visszafejthetjük az eredetire, ha a kulcs inverzét, azaz a kulcs értéke: (-5) –öt használjuk. Ha az üzenet kulcsát nem ismerjük, akkor az üzenet feltöréséhez minden lehetséges kulcsot ki kell próbálni, ami 26 betű esetén, 26 kulcsot jelent.

2. Titkosítsa az alábbi nyílt szöveget Keyword (kulcsszavas) Caesar titkosítással, ahol a kulcs értéke (6, MATK).

A KOCKA EL VAN VETVE

A kulcshoz tartozó megfelelő titkosító tábla ekkor a következőképpen néz ki:

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
U	V	W	X	Y	Z	M	A	T	K	B	C	D

13	14	15	16	17	18	19	20	21	22	23	24	25
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
E	F	G	H	I	J	L	N	O	P	Q	R	S

Ekkor a következő titkosított szöveget kapjuk meg:

U BFWBU YC OUE OYLOY

Ennek feltöréséhez az összes kulcs kipróbálása nem működik, mert az 26!=403291461126605635584000000. De lehet próbálkozni betű-pár, betű-hármas, szó gyakoriságok vizsgálatával.

3. Titkosítsa ismét A KOCKA EL VAN VETVE üzenetet RSA kódolással, kulcsok pedig a megadott értékek legyenek!

   Adott a nyilvános kulcs: $(65\,537,\ 7\,238\,359)$

   Adott a titkos-kulcs: (1 332 809)

   Az előzőekhez hasonlóan itt is először a megadott üzenet betűihez hozzárendeljük a számkódokat:

A	K O C K A	E L	V A N	V E T V E
0	10 14 2 10 0	4 11	21 0 13	21 4 19 21 4

   A karaktersorozatot 26-os számrendszerbeli számnak fogjuk tekinteni.

   A számokat átírjuk a 26^k számrendszerbe, ahol $k = \lceil \log_{26} n \rceil$ és $n = 7\,238\,359$.

   Ekkor $k = 4$

   Tehát 4-es csoportokat kell készíteni, és 26^4 számrendszerbe a következő számokat kapjuk:

$$44\,876 \qquad 196\,050 \qquad 369\,455 \qquad 83\,698$$

   Amit így számoltunk ki:

$$44\,876 = 2 \cdot 26^3 + 14 \cdot 26^2 + 10 \cdot 26^1 + 0 \cdot 26^0$$
$$196\,050 = 11 \cdot 26^3 + 4 \cdot 26^2 + 0 \cdot 26^1 + 10 \cdot 26^0$$
$$369\,455 = 21 \cdot 26^3 + 0 \cdot 26^2 + 13 \cdot 26^1 + 21 \cdot 26^0$$
$$83\,698 = 4 \cdot 26^3 + 19 \cdot 26^2 + 21 \cdot 26^1 + 4 \cdot 26^0$$

   Mind a 4 számot titkosítjuk a nyilvános-kulccsal:
   $(65\,537\ , 7\,238\,359)$

e	44 876	196 050	369 455	83 698
$c = e^{65\,537}\ mod(7\,238\,359)$	1 987 760	6 194 439	5 897 773	6 518 210

Ekkor az eredmény:

$$1\,987\,760 \quad 6\,194\,439 \quad 5\,897\,773 \quad 6\,518\,210$$

A kapott számokat $26^{k+1} = 26^5$ -beli számoknak tekintjük, és át írjuk a 26-os számrendszerbe. Az eredményt pedig 5-ös csoportokba formáljuk:

$$8\ 12\ 2\ 9\ 4 \quad 17\ 9\ 11\ 14\ 13 \quad 11\ 13\ 14\ 23\ 12 \quad 8\ 8\ 22\ 6\ 14$$

Amit az alábbi módon kaptunk meg:

$$1\,987\,760 = 4 \cdot 26^4 + 9 \cdot 26^3 + 2 \cdot 26^2 + 12 \cdot 26^1 + 8 \cdot 26^0$$
$$6\,194\,439 = 13 \cdot 26^4 + 14 \cdot 26^3 + 11 \cdot 26^2 + 9 \cdot 26^1 + 17 \cdot 26^0$$
$$5\,897\,773 = 12 \cdot 26^4 + 23 \cdot 26^3 + 14 \cdot 26^2 + 13 \cdot 26^1 + 11 \cdot 26^0$$
$$6\,518\,210 = 14 \cdot 26^4 + 6 \cdot 26^3 + 22 \cdot 26^2 + 8 \cdot 26^1 + 8 \cdot 26^0$$

Ekkor a számkódok alapján könnyen megtaláljuk minden egyes számhoz tartozó betűt, amivel a titkosított karaktersorozatot megkapjuk:

IMCJE RJLON LNOXM IIWGO

Visszafejtés:

A titkosított számkód-sorozat:

$$8\ 12\ 2\ 9\ 4 \quad 17\ 9\ 11\ 14\ 13 \quad 11\ 13\ 14\ 23\ 12 \quad 8\ 8\ 22\ 6\ 14$$

A számkódokat átírjuk 26^5-es számrendszerbe, aminek az eredménye az alábbiak:

$$1\,987\,760 \quad 6\,194\,439 \quad 6\,518\,210$$

Ezután minden egyes számra alkalmazzuk a titkos-kulcsot (1 332 809) és a nyilvános kulcsot (7 238 359):

Jelöljük *c*-vel a titkosított számot:

c	1 987 760	6 194 439		6 518 210
$c^{1\,332\,809}\ mod(7\,238\,359)$	44 876	196 050	369 455	83 698

A számolással megkapott számokat átírjuk 26-os számrendszerbe, amivel ha mindent jól csináltunk, visszakapjuk az eredeti számkód-sorozatot:

$$0\ 10\ 14\ 2\ 10\ 0\ 4\ 11\ 21\ 0\ 13\ 21\ 4\ 19\ 21\ 4$$

Tehát behelyettesítve a számkódokhoz tartozó betűkkel:

A KOCKA EL VAN VETVE

4. Készítse el az alábbi üzenethez tartozó digitális aláírást!

„Julius Caesar híres szállóigéje: A kocka el van vetve.”

Adott a nyilvános kulcs: $(53201, 64829)$

Adott a titkos-kulcs: (28721)

MD5 hash generátor által elkészített hash függvénye az üzenetnek:

8e38b361d94eda1c8390d89d4ca3b4a3

Előre megegyeznek, hogy az üzenet egyedi azonosítója, a hash függvény számainak összege lesz, tehát $MD(m)$ = 94

Ezután kiszámolja a Feladó az üzenethez tartozó digitális aláírást:

$$S = (MD(m))^s(modh) = 94^{28\,721}\ (mod 64\,829)$$

$$S = 58746$$

A Feladó az (m, S) párt küldi el a Címzettnek:

(„Julius Caesar híres szállóigéje: A kocka el van vetve.”, 94)

Visszafejtés:

A Címzett is kiszámolja az üzenethez tartozó egyedi azonosítót, az $MD(m)$-et és az S érték alapján visszafejti az egyedi azonosítót:

$$Z = S^k(modh) = 58\,746^{53\,201} mod(64\,829) = 94$$

Ha $Z = MD(m)$ értékével, akkor hiteles a dokumentum.

4. Interaktív bizonyítások

4.1 Prímtesztelés

A korábbi feladatok végrehajtásához szükségünk volt prímszámokra, de például honnan tudjuk eldönteni, hogy a 123456 szám prím-e? Erről a számról természetesen még könnyen el tudjuk dönteni, hogy nem, mivel páros és 2-nél nagyobb. De mit csináljunk akkor, ha már egy látszatra nehezebb számról akarjuk eldönteni, például a 1234567-ről? Ezt már nem tudjuk ránézésre megmondani. Ekkor megpróbáljuk először 2-vel, aztán 3, 4, 5… számokkal elosztani. Egy idő után sikeresen megkapjuk a prímtényezős felbontását, ami a 1234567=127·9721.
De mi történik akkor, ha még nagyobb számról kell eldöntenünk, hogy prím-e? Legyen ez a szám például:

1111 222 233 334 444 555 566 667 777 888 899 967

Ahhoz hogy erről a számról eldöntsük, hogy prím-e, a szám négyzetgyökéig minden számra, de legalább minden prímre ki kellene próbálnunk, hogy osztható-e vele. Mivel a szám nagyobb, mint 10^{36}, így a négyzetgyöke is nagyobb lesz mint 10^{18}. Viszont ennyi számot ember biztosan nem, de még a legmodernebb számítógépek segítségével se fogunk tudni végigpróbálni.

Fermat –teszt: A probléma megoldásának egyik módszerét a „kis” Fermat-tétel biztosítja. A „kis” Fermat-tételből következik, hogy ha$p$ prím, akkor $p \mid 2^p - 2$. Továbbá ha feltesszük a p-ről, hogy páratlan szám, akkor $p \mid 2^{p-1} - 1$. (Így a $p = 2$esetet sikeresen kizártuk már.)
Nézzük meg, hogy összetett n-ekre teljesült-e $n \mid 2^{n-1} - 1$ reláció. Vizsgáljuk először csak páros n-ekre. Ha $n$páros, akkor a reláció úgy néz ki, hogy egy páros szám osztója egy páratlannak, ami persze nem lehet. Elegendő ezek után már csak a páratlan n -ekre megvizsgálni a relációt. Néhány példa:

$$9 \nmid 2^8 - 1 = 255, \qquad 15 \nmid 2^{14} - 1 = 16383, \qquad 21 \nmid 2^{20} - 1 = 1048575$$

A fenti 3 példánkkal ellenőriztük, hogy a $n \mid 2^{n-1} - 1$ relációnk jól működik, tehát eldönthető vele egy $n$páratlan számról, hogyaz vajon nem prím-e. Viszont további problémák lehetnek a relációval kapcsolatba.
A $n \mid 2^{n-1} - 1$ reláció azt sugallja, hogy könnyen ellenőrizhetjük egy számról, hogy az prím-e vagy sem. Eddig nem vettük figyelembe azt a problémát, hogy egy nagy hatvány

kiszámításának műveletigénye igen magas. Mivel egy 2^{n-1} kiszámításához $(n-2)$ szorzást kell elvégeznünk. Ha $n$egy 100-jegyű szám, akkor ez egy 10^{100}darab lépést jelent, aminek kiszámítása nagyon sok időt vesz igénybe.

A fenti problémát viszont egy nagyon ügyes trükkel megoldhatjuk. Legyen ez a nagy hatványkitevővel rendelkező szám a 2^{24}. Elkezdjük kiszámolni először a $2^3 = 8$, aminek a négyzetre emelésével $2^6 = 64$-et kapjuk meg. Újabb négyzetre emeléssel 2^{12} =4096 –ot, majd ugyan ezt folytatva megkapjuk a 2^{24} =16777216-ot. Így a 23 szorzás helyet elegendő volt csak 5 szorzást elvégeznünk. Persze a módszerünk azért működött jól, mert a 24 osztói között nagy 2-hatvány is van, így a 2^{24}-hez egy kis számból kiindulva könnyen eljutottunk. Viszont most mutassuk meg más számokra is, hogy valójában jól működik ez a módszer. Legyen ez a szám 2^{29},amit például a következőképpen kapunk meg:

$$2^3 = 8, \qquad 2^6 = 64, \qquad 2^7 = 128, \qquad 2^{14} = 16384,$$
$$2^{28} = 268435456, \qquad 2^{29} = 536870912$$

Lépésszám:$[log_2 n]$

A nagy hatvány kiszámolása után további nehézséget jelent számunkra a nagy számok kezelése. Ha van egy $n$számunk, ami 100-jegyű, akkor annak nem csak a 2^{n-1} száma lesz nagy, hanem maga a végeredmény számjegyei száma sok. Illetve egy ekkora számot ellenőrizni se tudunk, hogy osztható-e *n*-nel. A problémát a következő megoldással tudjuk kikerülni: Minden lépésben, mikor a számot n-nel akarjuk leosztani, az adott maradékkal számolunk. Nézzünk egy példát erre a gyakorlatban:

Ellenőrizni akarjuk, hogy a 25 osztója-e a 2^{24-1}számnak. A 2^{24} számot már könnyen ki tudjuk számolni. Először a $2^3 = 8$-at számoljuk mi, majd ezt négyzetre emelve $2^6 = 64$-et kapunk. Itt viszont nem emeljük ismét a négyzetére, hanem megnézzük a 25-tel való osztási maradékát, ami 14. Így most már ezzel a maradékkal helyettesítjük a 2^6 számot. Következő lépésben tehát nem a 2^6-t fogjuk négyzetre emelni, hanem a maradékot, $14^2 = 196$. Megnézzük ennek is a 25-tel való osztási maradékát, ami 21, és ezzel helyettesítjük tovább. Tehát a 2^{12} négyzetre emelése helyett elég nekünk már csak a maradékot négyzetre emelnünk, a 21^2. Ezzel sokkal kevesebbet számolást kell csak végrehajtanunk. Mivel $21^2 = 441$, osztási maradéka pedig 16. Így a relációba visszahelyettesítve megkapjuk, hogy 16-1=15 nem osztható 25-tel, azaz ezzel azt kaptuk, hogy a 25 nem prímszám. A fenti módszer ennél nagyobb n-ekre is jól működik. Ahhoz hogy a számaink mindig kicsik maradjanak, mindig a maradékokkal kell tovább számolnunk. Persze ha egy 100-jegyű

számunk van, akkor annak a négyzete 199- vagy 200-jegyű, amit egy ember számára szinte lehetetlen kiszámolni „kézzel”, de egy számítógépnek ez nem okoz problémát.

Prímteszteléssel kapcsolatos harmadik problémánk pedig a pszeudoprímek. A fenti

$$n|\ 2^{n-1}-1$$

reláció arra sejtet, hogy segítségével könnyen el tudjuk dönteni egy számról, hogy vajon prím-e vagy sem. De mennyire vehetjük biztosnak, ha egy $n$számra igaz a $n|\ 2^{n-1}-1$ reláció, akkor valóban prím? Fermat tételéből ez nem következik. Méghozzá egyből egy ellenpéldát is tudunk mutatni, a $341|\ 2^{341-1}-1$ reláció teljesül rá, miközben a 341=11·31 nem prím. Ezeket a számokat, amik a „kis” Fermat-tétel szerint prímként viselkednek pszeudoprímeknek vagy álprímeknek nevezzük.

Nincs sok ilyen szám, de arra mégis elegendő, hogy ne tudjuk teljes biztonsággal kezelni a prímtesztet. Ennek kiküszöbölésére a „Kis” Fermat-tételt használjuk fel, miszerint: Tetszőleges $p$prímszám és $a$egész szám esetén $p|\ a^p-a$. Továbbá a tételt gyakran úgy is kimondják, hogy ha $p$prímszám,$a$pedig p-vel nem osztható egész szám, akkor $p|\ a^{p-1}-1$. Ezt az ötletet felhasználva, máris látjuk, hogy a 341 egy álprím, mert valójában

$$341 \nmid\ 3^{340}-1\,.$$

A teszt jól működik minden *n* pozitív egész számra, ahol$1 \le a \le n-1$.

Vannak azonban olyan $n$pozitív összetett számok is, amelyekhez nem létezik Fermat-tanú. Az ilyen számokat Carmichael-számoknak hívják. Ezek a számok eléggé ritkán fordulnak elő, de ahhoz mégis elég gyakran, hogy a Fermat-tesztben ne bízhassunk feltétel nélkül. A legkisebb Carmichael-szám az561. Tehát van olyan $n$összetett egész szám, amelyhez képest relatív prím a szám esetén sem akad fenn a Fermat-teszten:

$$n|\ a^{n-1}-1\,.$$

Azaz minden olyan $a$esetén, amelyre$lnko(n,a)=1$.

Annak, hogy az 561 Carmichael-szám a következőképpen történhet a bizonyítása:

$$561 = 3\cdot 11\cdot 17$$

Kínai maradéktétel és Fermat-teszt szerint:

$$a^2 \equiv 1 \quad (3) \quad \rightarrow \quad a^{560} \equiv (a^2)^{280} \equiv 1 \quad (3)$$
$$a^{10} \equiv 1 \quad (11) \quad \rightarrow \quad a^{560} \equiv (a^{10})^{56} \equiv 1 \quad (11)$$
$$a^{16} \equiv 1 \quad (17) \quad \rightarrow \quad a^{560} \equiv (a^{16})^{35} \equiv 1 \quad (17)$$

Tehát$a^{560} \equiv 1\ (561)$ minden a-ra, melyre $(a, 561) = 1$

4.2 Az utolsó sakklépés

Ebben a fejezetben egy konkrét példát fogok bemutatni. Továbbra is az a célunk hogy az üzenetünk ne kerüljön illetéktelen kezekbe, viszont ennél a feladatnál további kihívást is rejt az üzenetünk kódolásának bonyolultsága. 1970-es években találkoztak először ilyen problémákkal, innen számítják a modern kriptográfia kezdetét is.

A feladatunk úgy néz ki, hogy két ember sakkozik, de nem a szokásos módon, egy asztalnál, hanem egy elektronikus csatornán. Így nem látják egymást, de közlik a másikkal a lépésüket. Viszont a játszmát fel kell függeszteniük külső okok miatt, de később szeretnék folytatni. Itt találkozunk már az első problémával, hogy miután megtörtént az utolsó lépés, az egyiknek sokkal több gondolkodási ideje lesz a másikkal szemben, ami jogosulatlan előnyhöz juttatja az újrakezdéskor elsőként lépő játékost. Így az az ötletünk lenne először, hogy az utolsó sakklépést írjuk le egy papírra, és a szünet után az a lépés lesz meglépve. Elsőre jó ötletnek tűnik, de hogyan vigyük ezt véghez? Nem lehetünk abban biztosak, hogy a másik a leírt sakklépését nem fogja megváltoztatni, vagy ha ezt a papírt eljuttatnánk a másikhoz, nincs rá garancia, hogy ő tényleg nem fogja megnézni korábban. Ezért nekünk ezt az utolsó sakklépést üzenetként kell kezelnünk, és kódolnunk kell. A célunk egy kód kitalálása, és hozzá egy megfelelő kulcs, amivel konkrétan csak egy sakklépést definiálhatunk. Tehát az utolsó lépésnek kódolását elküldi a másik félnek még a szünet előtt, aki ezt nem fogja tudni feltörni, viszont a játszma folytatásakor megkapja hozzá a kulcsot, amivel meg tudja fejteni, hogy melyik sakklépésre is gondolt az ellenfél. A nehézséget az okozza, hogy egy megfelelően nehéz kulcsot találjunk ki. Figyelnünk kell arra, hogy az ellenfél kulcs nélkül ne tudja megfejteni az üzenetet, másrészt a lépés megváltoztatására se legyen lehetőség.

A probléma megoldására a számelmélet elemeit fogjuk használni. Minden lépés feleljen meg egy véletlenszerűen kiválasztott négyjegyű számnak. Például legyen az utolsó lépésünk Hb3, ekkor jelölje „13” H-t, „4” a b-t és a 3 pedig önmagát. Ekkor ennek a lépésnek eddig a „1343” négyjegyű szám felel meg. Ezt még nem nevezhetjük titkosításnak, de minden karaktert jelöltünk. Tehát az 1-es játékos a lépést megadó 4 számjegy után további számokat fog írni utána, méghozzá 196-ot, úgy hogy egy 200 jegyű $a = 1343\ldots$ prímszámot kapjon. Ezután készít egy másik 201 jegyű b prímszámot is. A két prímszám szorzata pedig legyen $C = a \cdot b$. Ekkor a szorzat 400 vagy 401 jegyű szám lesz, amit az 1-es játékos elküld a 2-es játékosnak. Ezzel elhárítottuk azt a problémát, hogy a 2-es játékos kulcs nélkül meg tudja fejteni a lépést, és másrészt az 1-es játékos sem fogja tudni megváltoztatni a lépését. A játék

folytatásakor a 2-es játékos megkapja az „a” és „b” két prímszámot, és a kisebb prímszám első 4 karaktere lesz az 1-es játékos lépése. A 2-es játékos közben leellenőrizheti azt is, hogy tényleg nem csalt a másik fél, valóban a két prím szorzata-e a Cszám.

Az 1-es játékos is biztos lehet, hogy az ellenfél nem találja ki a Cszámból a lépését, mert ehhez a 400 vagy 401 jegyű szám prímtényezőit kellene megtalálnia. Ehhez viszont még egy számítógépnek is rengetek időre és kapacitásra lenne szüksége. Továbbá az 1-es játékos sem teheti meg, hogy egy másik a’ és b’ prímpárt küld el, mert akkor nem jönne ki a $C = a \cdot b$, és lebukna a 2-es játékos előtt.

Tehát az 1-es játékos lépését a kisebb prímszám első 4 számjegye vitathatatlanul meghatározza. Úgy is értelmezhetjük ezt a megoldást, mintha a 4 számjegyet az utána lévő 196 számjeggyel és a másik 401 jegyű számmal becsomagolnánk valamiféle dobozba. A 2-es játékos számára pedig éppen ezzel a dobozzal sikerül elrejteni az üzenetünket, megfejteni pedig csak a kulcs birtokában fogja tudni. A feladat bonyolultsága abban rejlik, hogy megtaláljuk ennek a nagy számnak a prímtényezős felbontását.

Ezzel az egyszerű példával egy elektronikus „csomagolási lehetőséget” mutattam be. A mai világban a hagyományos leveleinket, üzeneteinket is elektronikus úton küldjük tovább. De természetesen más módszereket is használhatunk, az elektronikus jelszavak vagy az elektronikus aláírás problémájára. Szükség van olyan módszerekre is, ami megoldja a számítógépek, bankjegy automaták biztonságát. Ezeken kívül is számos alkalmazási terület létezik még, ahol az ilyen titkosító módszerekre szükség van. Ezek a módszerek általában valamilyen számelméleti tényeken alapulnak. A következő fejezetben egy ilyen módszert fogjuk jobban megvizsgálni.

4.3 Hogyan ellenőrizhető egy ismeretlen jelszó?

Vegyünk ismét egy példát a banki világban. Minden ügyfélhez tartozik egy név és egy jelszó, amik alapján a bankjegykiadó automaták meghatározzák az ügyfelet. Számelmélet segítségével próbálunk módszert találni arra a problémánkra, hogy a jelszavunk a bankban ismeretlen maradjon, banki dolgozók se férjenek hozzá ehhez az információhoz, még ha a számítógép memóriájában archiválják is. Tehát a banknak ellenőriznie kell, hogy valójában tudja-e az ügyfél a jelszavát, ugyanakkor maga a bank ne tudja a pontos jelszót. Ez első

hallásra lehetetlennek tűnhet, de a bonyolultságelmélet megoldást nyúlt erre. Az utolsó sakklépéshez hasonló módon ezt is könnyen kivitelezhetjük.

Először is legyen egy jelszavunk, ami 100 jegyű prímszám, amit jelöljünk a-val. Miután kiválaszt az ügyfél magának egy jelszót, kiválaszt hozzá egy másik prímszámot is, amelyet pedig jelöljünk b-vel. Jelöljük a két prímszám szorzatát ismétC-vel, $C = a \cdot b$. Az ügyfél a banknak ezt a C számot fogja csak megadni. Tehát amikor bemegy az ügyfél a bankba, és megadja a jelszavát, akkor a bank azt fogja ellenőrizni, hogy az a megadott szám osztója-e a rendszerben lévő C számnak. Ha igen, akkor a rendszer tovább is engedi az ügyfelet. A számítógép számára ez egy egyszerűen meghatározható feladat, egy 200 jegyű szám osztása egy 100 jegyű számmal. Viszont egy ember számára már nem.

Mi történik akkor, ha a banki rendszerben valaki meglátja ezt a C számot? Persze megpróbálhatja ennek tudatában kitalálni az ügyfél jelszavát, de ehhez a C szám egy 100 jegyű osztóját kellene megtalálnia. Így ő is abba a problémába esik, hogy egy nagy számot kellene prímtényezőkre felbontania, ami elég kilátástalan próbálkozás. Tehát a bank ismeri a C számot, ami végül is minden információt tartalmaz, de a prímtényezős felbontás elvégzésének bonyolultsága miatt azt mondhatjuk, hogy biztonságos kezekben vannak a jelszavaink a banki rendszerben.

4.4 Hogyan találjunk nagy prímeket?

Az előző két példánkban nagy prímekre volt szükségünk, de arra nem tértem ki, hogy is találjunk nagy prímeket. Azt tudjuk, hogy végtelen sok prímszámunk van. (Bizonyítás nélkül). De vajon abban mennyire lehetünk biztosak, hogy mint az utolsó sakklépéses példában, adva van az első négy számjegye, akkor találunk-e hozzá ilyen kezdetű prímszámot? Méghozzá nem is egy sima prímszámot, ami például 1163 számjegyekkel kezdődik, hanem legyen ez a prím 200 jegyű is.

A Maple program segítségével könnyen megkapjuk, mi az a legkisebb 200 jegyű prímszám, ami 1163-mal kezdődik:

116300000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000371

A legkisebb 200 jegyű szám, aminek az első négy számjegye 1163, az a $1163 \cdot 10^{196}$. Ez a számunk még nem prím, de a program segítségével megkapott számhoz nagyon közel áll már. Összesen $9 \cdot 10^{199}$ db 200-jegyű szám van, ebből viszont a prímszámtétel alapján tudjuk, hogy nagyságrendileg $1{,}95 \cdot 10^{197}$ db 200 jegyű prímszám van. Tehát a 200-jegyű számok között átlagosan minden 460-adik szám prím.

Az utolsó sakklépésben az 1-es játékosnak tehát elég sok szám áll rendelkezésre, amiből véletlenszerűen ki kell választani egyet a megfelelő 4 számjegykezdésűekből, és tesztelnie kell, hogy a szám prím-e. Mint ahogyan már láttuk, minden 460-ik szám prím, így az 1-es játékosnak minden 460-adik próbálkozásra egy prímszámot fog kapni. Persze ez elég időigényes lenne az 1-es játékosnak, de a legkisebb számot mégsem érdemes választania, mert azt könnyen kitalálná az ellenfél. Viszont egy számítógép segítségével másodpercek alatt megkapunk egy ilyen tetszőleges 200 számjegyű prímet, aminek az első 4 karaktere adott:

116314671287655576327990970455966069082836547600666887381448935466247436041
989110468041110388689588057457155724800095696391740333854584185935354886223
237823175775598647396527011271770972783894654145 89

Tehát kódunkat prímekbe csomagoltuk, amivel a titkosítást könnyen megoldottuk, viszont ennek prímtényezős felbontása kilátástanul nehéz feladat, így biztosítjuk valójában a kulcsunk feltörhetetlenségét is.

5. A bonyolultságelméleti modell

5.1 Egy bonyolultságelméleti modell

Tekintsünk egy egyszerű bonyolultságelméleti feladatot, ami látszólag nem hasonlít a klasszikus problémánkra. Az emberek nagy része rendelkezik bankkártyával, és a készpénzfelvételhez a bankautomatákat használják. Az automaták használatához szükségük van a kártyatulajdonosoknak a kártyához tartozó jelszóhoz, amit elvileg csak ők ismernek. Kártyahasználatkor a saját egyedi jelszavukat kell megadniuk, amit a bank ellenőriz, hogy valójában a saját jelszavát adja meg az ügyfél. Ha igen, akkor a rendszer tovább engedi, és elvégezheti a kívánt tranzakciót. A kártyához tartozó jelszót egy külön borítékban küldi ki a bank az ügyfélnek, és a bankkártyára sincs ráírva, így ha az ügyfél nem mutatja meg másnak, akkor azt tényleg csak ő tudja. Viszont a probléma ott kezdődik, hogy ezt a jelszót a banknak is tudnia kell, és így visszaélhet vele a bank alkalmazottja. A kérdés az lenne, hogy vajon lehet-e olyan rendszert kialakítani, ami ezt a problémát kiküszöbölné? A banki alkalmazottak ismerik a jelszót ellenőrző programot, viszont megoldható-e, hogy ebből ne lehessen következtetni az eredeti jelszóra. Elsőre önellentmondásnak tűnő követelmény, de jobban megvizsgálva kiderül, hogy kielégíthető.

A fenti feladat például a következő megoldással is kivitelezhető lenne (a valóságban nem ezt a módszert használjuk): Az ügyfél felvesz $p$db pontot, és mindegyiket beszámozza 1-től p-ig. Ezen pontok alapján berajzol egy Hamilton-kört, és további éleket a gráfhoz. Az ügyfélnek innentől kezdve ez a Hamilton-kör lesz a saját jelszava, amit meg kell jegyeznie. Ezt a gráfot Hamilton-kör feltüntetése nélkül megmutatja a banknak is. A gráfhoz tartozó Hamilton-körnek csak a számlatulajdonos van a birtokában. Így a bank ellenőrizni tudja az ügyfelet a megadott gráf alapján, hogy valójában a hozzá tartozó Hamilton-kört adta meg az ügyfél. Viszont a bank már nem fogja tudni megkeresni ezt a Hamilton-kört, hiszen ennek megtalálása NP-nehéz.

A feladat megoldásához szükséges, hogy olyan rendszert alakítsunk ki, ami NP-nehéz. Mi itt most a Hamilton-kör problémát használtuk fel, de bármelyik más NP-nehéz probléma felhasználható.

A feladat sikeres megoldásához a további feltételeket is figyelembe kell vennünk:

Ahhoz hogy biztonságos legyen a rendszer, bíznunk kell a bankban lévő programban. A program ismeri a jelszavakat (gráfokat), viszont a hozzá tartozó Hamilton-kört nem, így amikor az ügyfél bejelentkezik és megadja a hozzá tartozó Hamilton-kört, a programnak csak ellenőriznie szabad a jelszót és nem tárolhatja el a memóriába. Továbbá a biztonsághoz szükséges, hogy a megadott jelszót senki se lássa vagy hallgassa le.

Foglalkoznunk kell azzal a kérdéssel is, hogy az ügyfél hány tovább i élet vegyen hozzá a gráfhoz, és azt milyen módon tegye. Célunk egy gráf, amiben a Hamilton-kör megtalálása nehéz legyen. De hogyan tudjuk ezt biztosítani? Első ötletként megpróbálhatunk egy gráfot véletlenszerűen kiválasztani, ám ha így választunk egy gráfot az összes p-pontú gráfok közül, akkor ebben vagy mi is elég könnyen találunk Hamilton-kört, vagy ha nehéz benne találni, akkor számunkra is. Túl nagy e vagy túl kicsi e esetén könnyű lenne a Hamilton-kör kereső feladata. Az első esetben könnyen találhatna ilyen kört, a második esetben könnyen bizonyíthatná, hogy nincs ilyen. Célszerű eljárás lehet kiválasztani egy C_p gráfot, és ehhez hozzávenni – például véletlenszerűen – további éleket, míg akkora gráfot nem kapunk, amiben már nehéz lesz megtalálni az eredeti C_p-t – ami természetesen Hamilton köre a gráfnak, hiszen pontokat nem vettünk hozzá a gráfhoz –, de még nem könnyű találni benne esetleg egy másik Hamilton-kört. A szakirodalom alapján, ha van egy gráfunk p ponttal, e éllel és ezekre a következő feltétel teljesül: $e = \left(\frac{1}{2}\right) p \cdot \log(p)$, akkor ebben a gráfban már elég nehéz a Hamilton-kör megtalálása.

NP-nehéz probléma egy gráf 3 színnel való színezhetősége is. Így ez is használható lenne jogosultság igazolásra, például a következő módon: megad valaki nekünk egy p pontú gráfot, és el kell döntenünk róla, hogy kiszínezhetők-e a csúcsai három színnel úgy, hogy bármely két szomszédos csúcs ne legyen azonos színű. Azonban ha valaki azt állítja, hogy 3 színnel sikerült színeznie az adott gráfot, akkor azt a konkrét színezést polinomiális időben le tudjuk ellenőrizni a következő módon: minden csúcsánál megnézzük a szomszédos csúcsok színeit, vagyis p csúcsnak legfeljebb p-1 szomszédját vizsgáljuk meg, azaz legfeljebb $p\cdot(p\text{-}1)$ lépésben tudjuk ellenőrizni, hogy jó-e a színezés.

6.2 P ≠NP probléma

A P=NP megoldatlan probléma a matematikában.

Fontos hogy egy nyilvános kódú üzenetet úgy kódoljunk, hogy annak kulcs nélküli visszakódolási algoritmusa NP nehéz legyen, viszont a Címzett olyan plusz információ - a kulcs - birtokába legyen, melynek segítségével polinomiális időben képes az üzenet dekódolására. A leggyakoribb módszerek azt használják fel, hogy egy nagy (például több ezer számjegyből álló) egész szám prímszámok szorzatára való bontására nem ismerünk olyan algoritmust, ami polinomiális időben megoldja a feladatot. A titkosítás során ezt az egész számot használjuk fel és a dekódolás csak akkor egyszerű, ha ismerjük a prímfelbontást. Vagyis csak az tudja a felbontást, akinek az üzenetet szánjuk. Ezzel az eljárással az a probléma, hogy nem bizonyított, hogy a prímfelbontás elkészítése NP-nehéz. Tehát előfordulhat, hogy valaki talál egy olyan felbontó algoritmust, ami polinomiális időben lefut. Ekkor a nyilvános kulcs ismeretében bárki, aki ismeri ezt az eljárást, el tudná olvasni a kódolt üzeneteket.

Köszönetnyilvánítás

Köszönöm Korándi József témavezetőmnek a szakdolgozat során adott hasznos ötleteit és türelmét. Az ő segítsége, jó tanácsai nélkül nem jöhetett volna létre ez a dolgozat. Mindig bátran fordulhattam hozzá kérdéseimmel.

Továbbá köszönöm szüleimnek a sok gondoskodást, ami elkísért a tanulmányaim során. Köszönöm a kitartó türelmüket és a mindennapi segítségüket, amivel mindvégig támaszt nyújtottak tanulmányaim során. Emellett köszönöm barátomnak és barátnőmnek, akik igyekeztek figyelmes észrevételeikkel segíteni.

Felhasznált irodalom

[1] https://hu.wikipedia.org/wiki/Kriptográfia

[2] https://en.wikipedia.org/wiki/Digital_signature

[3] https://en.wikipedia.org/wiki/Diffie-Hellman_key_exchange

[4] http://snailman.web.elte.hu/pub/diffie-hellman/diffie-hellman/dh-f2.htm

[5] Lovász László: Algoritmusok Bonyolultsága: http://www.cs.elte.hu/~kiraly/Algbony.pdf

[6]Lovász László, Pelikán József, Vesztergombi Katalin: Diszkrét Matematika, Typotex kiadó 2010

[7] http://orulunkvincent.blog.hu/2010/08/09/gyorshir_p_nem_egyenlo_np

NYILATKOZAT

Név: Hrubi Nóra
ELTE Természettudományi Kar, szak: Matematikai Bsc
NEPTUN azonosító: F2YWR3
Szakdolgozat címe: Fejezetek a bonyolultságelméletből

A **szakdolgozat** szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló munkám eredménye, saját szellemi termékem, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.

Budapest, 2016.05.27.

a hallgató aláírása