10.

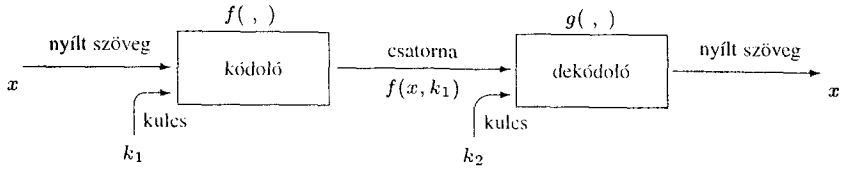
Nyilvános kulcsú titkosírások

A meglepetés kulcsa a gyorsaság ötvözése titoktartással.
CARL VON CLAUSEWITZ

Itt rövid ízelítőt adunk a kriptográfia, a titkos információcsere elemeiből. Az illetéktelen hozzáféréssel szemben biztonságos kommunikáció sokáig elsősorban a politikusok és a katonák ügye volt. Ma, az elektronikus hálózatok világméretű térhódításával alapvető, mindennapi igénnyé vált a biztonságos és gyors információcsere. Ennek megfelelően a kriptográfia komoly tudományággá nőtt; se szeri, se száma a protokolloknak és algoritmusoknak. Utóbbiak egy része az algoritmusok bonyolultságának elméletén alapul. A kriptográfusok sajátos, fordított szemlélettel nézik a bonyolultsági eredményeket, amennyiben elsősorban a hatékonyan nem kezelhető problémák érdekesek számukra. Ami a gyors algoritmusok tervezésén fáradozók számára átok, az számukra áldás lehet. Több olyan kriptográfiai módszer ismeretes, amelynek a biztonságossága, „feltörhetetlensége" egy algoritmikus probléma nehézségén alapul. Bizonyos mértékig ilyen módszer a gyakorlatban is igen népszerű RSA-algoritmus, amit vázolni fogunk, érdekes alkalmazását adva ezzel néhány eddig megismert fogalomnak és eredménynek.

10.1. Kriptográfia - a titkosírások tudománya

Először szólunk néhány szót a kriptográfiáról általában. A titkos információcsere következő egyszerű és általános modelljét fogjuk használni.



$$g(f(x, k_1), k_2) = x$$

Az x a kódolatlan – szokásos szakkifejezéssel: *nyílt* – üzenet, amit el szeretnénk juttatni a *vevőhöz*. Köztünk és a vevő között képzeljük el a *csatornát*, amin az üzenetnek át kell jutnia. A csatorna a kommunikációs rendszernek az a része, ahol az üzenethez illetéktelenek hozzáférhetnek. Ez ellen úgy védekezünk, hogy a csatornán az x üzenet *kódolt* változatát küldjük át, amit a másik oldalon a vevő megfejt, visszakapva az eredeti nyílt üzenetet. A kódolást, a titkosítást egy f eljárással (függvénnyel) végezzük. Az f egy k_1 *kulcs* segítségével végzi az átalakítást. Az f függvénynek a kódolandó szöveg és a k_1 információ a paraméterei. Az eredményül kapott rejtjelezett $f(x, k_1)$ üzenet megy át a csatornán. A vevő oldalán a titkos üzenet megfejtését, a *dekódolást* szintén egy kulccsal (jelölje ezt k_2) vezérelt g eljárás végzi. A g az y rejtjelezett üzenetből visszaadja a $g(y, k_2)$ nyílt szöveget. Az egymásnak megfelelő f, g eljárásokkal és k_1, k_2 kulcspárral szemben nyilvánvaló követelmény, hogy $g(f(x, k_1), k_2) = x$ teljesüljön minden x üzenetre. Jóval kevésbé formalizálható az a követelmény, hogy a rendszer biztonságos legyen, vagyis pusztán a csatornán átmenő kódolt $f(x, k_1)$ üzenetből ne lehessen kitalálni az x nyílt üzenetet.

A Vernam-kódoló

A teljes x nyílt üzenet és a $k_1 = k_2 = d$ kulcsok egyforma hosszú bitsorozatok, és $f(x, d) := x \oplus d$. Itt $\oplus$ a bitenkénti kizáró vagy (XOR) műveletet jelenti, d pedig egy „véletlen" bitsorozat. Nyilvánvaló, hogy $g = f$ használható dekódoló függvényként, hiszen egyfelől $d \oplus d$ sorozat csupa nullából áll, másfelől az $\oplus$ művelet asszociatív. A Vernam-kódoló egy régi, sokat használt eljárás. Hátránya, hogy a meglehetősen terjedelmes d kulcsot a kommunikáció megkezdése előtt valahogy (pl. futárral) biztonságosan el kell juttatni a másik félhez.

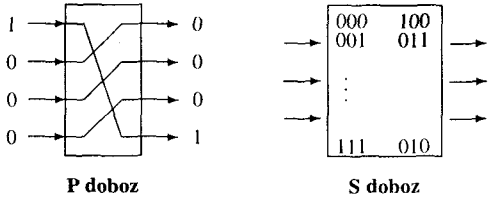
A továbbiakban olyan rendszerekkel foglalkozunk, melyekben az f kódoló függvény az üzenet rögzített (mondjuk k) hosszúságú darabjait kódolja el, és a titkos üzenet ezen elkódolt darabok egymásutánja. Hasonlóképpen működik a g dekódoló.

A DES rendszer

Röviden vázoljuk az IBM által kifejlesztett, alkalmas paraméterekkel az USA-ban

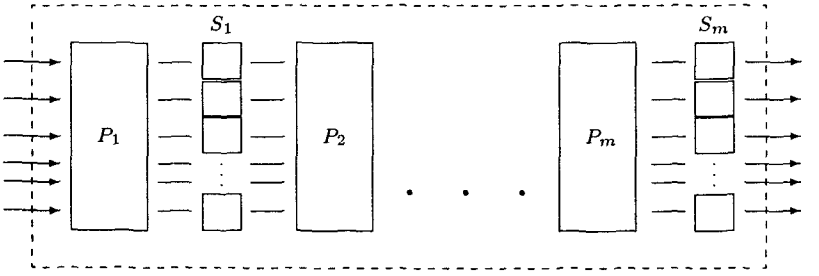
szabványosított DES (Data Encryption Standard) rendszer elemeit. Egyik kódoló-dekódoló alapeszköz a *P-doboz*. Egy P-doboz az $x_1x_2\cdots x_k$ input bitsorozatot egy rögzített σ permutációval az $x_{\sigma(1)}x_{\sigma(2)}\cdots x_{\sigma(k)}$ bitsorozattá alakítja.

A másik alapeszköz az *S-doboz*. Egy S-doboz kulccsal vezérelhető. A kulcs egy beállítása mellett a doboz egy $s : \{0,1\}^k \to \{0,1\}^k$ kölcsönösen egyértelmű (azaz invertálható) leképezést valósít meg. A következő ábra egy 4 bites ($k = 4$) P-dobozt és egy 3 bites ($k = 3$) S-dobozt mutat.



A P-doboz itt ciklikusan balra lépteti az input biteket; az S-doboznál pedig a benne lakozó s függvényt igyekeztünk szemléltetni. Tehát például $s(000) = 100$, $s(001) = 011$.

A DES ajánlás 64 bites nyílt üzenetek 64 bites rejtjelezett üzenetté való kódolását javasolja. E célra 64 bites P-dobozokat és 4 vagy 8 bites S-dobozok 16 illetve 8 magasságú tornyait használja. Ilyen kódoló egységeket kell sorban egymás után fűzni. Jelölje $P_1, \ldots P_m$ a használt P-dobozokat, $S_1, \ldots S_m$ az S-dobozok tornyait. A 64 bites x üzenetből a szintén 64 bites $S_m(P_m(\ldots S_1(P_1(x))\ldots)$ rejtjelezett üzenetet kapjuk.



Az IBM Lucifer elnevezésű titkosítási rendszere hasonló elrendezésben 128 bites P-dobozokat és 4 bites S-dobozok (32 magasságú) tornyait használja.

10.2. Nyilvános kulcsú kriptográfia

A régi vágású kriptográfiai rendszerek komoly gondja a kulcsok kezelése. A Vernam-kódoló jól mutatja ezt. A rendszer használata előtt az adónak és vevőnek meg kell egyeznie a használt d sorozatokat illetően. Az ehhez szükséges információcserének is biztonságosnak kellene lennie, hiszen ha valaki megneszeli a d sorozatokat, akkor könnyen meg tudja fejteni az $x \oplus d$ alakú üzeneteket. A nyilvános kulcsú rendszerek elegáns megoldást adnak a kulcsok kezelésére. Működésükhöz, mint látni fogjuk, nincs szükség előzetes titkos információcserére.

Legyenek a kommunikációs hálózat résztvevői $A_1, \ldots, A_n$. A_i -nek egy titkos D_i és egy nyilvános E_i kulcsa van. Az E_i kulcs mindenki számára ismert, mondjuk benne van a telefonkönyvben. Szintén nyilvános (és minden résztvevő számára ugyanaz) az f kódoló függvény, melyre minden i index és x input szó (üzenet) esetén teljesül, hogy

$$f(f(x, E_i), D_i) = f(f(x, D_i), E_i) = x.$$

E kikötések azt fogalmazzák meg, hogy f használható dekódoló függvényként is; ezenfelül (E_i, D_i) és (D_i, E_i) összetartozó kulcspárok abban az értelemben, hogy ha az egyik kulcsot használjuk kódolásra, akkor a másik használható dekódolásra. Teljesülnie kell még két fontos további követelménynek:

$\mathbf{K}_1$: Adott x, E_i bemenetre $f(x, E_i)$ hatékonyan számítható; adott x, D_i bemenetre $f(x, D_i)$ hatékonyan számítható.
$\mathbf{K}_2$: Pusztán x és E_i ismeretében $f(x, D_i)$ nem számítható hatékonyan; ugyanígy, ha csak x és D_i ismert, akkor $f(x, E_i)$ nem számítható ki hatékonyan.

A $\mathbf{K}_1$ feltétel azt jelenti, hogy a kódolás, illetve dekódolás gyorsan végezhető. Ez a feltétel viszonylag egyszerűen teljesíthető. Jóval több gond van a $\mathbf{K}_2$ feltétellel, ami azt célozza, hogy a rendszer biztonságos, nehezen feltörhető titkos kommunikációt tegyen lehetővé. A $\mathbf{K}_2$ tulajdonságot az ismert rendszerek úgy igyekeznek elérni, hogy adott x, E_i esetén $f(x, D_i)$ meghatározása egy nehéz (pl. ismereteink szerint polinom időben nem kezelhető) algoritmikus probléma megoldását jelentse. Ugyanez vonatkozik természetesen az x, D_i párokra és az $f(x, E_i)$ értékekre is.

Tegyük fel, hogy van egy ilyen f függvényünk, A_i rendelkezik a D_i titkos kulccsal, és a telefonkönyvben megtalálhatók az E_i nyilvános kulcsok. Nézzük, miként oldható meg két kommunikációs alapfeladat ebben a keretben.

1. *A_1 titkos üzenetet küld A_2-nek.* A_1 az x nyílt szövegből az A_2 nyilvános kulcsával az $y = f(x, E_2)$ rejtjelezett üzenetet képezi, és ezt küldi el a csatornán.

A_2 ezt a saját titkos kulcsával dekódolhatja: $x = f(y, D_2)$. A $\mathbf{K}_2$ feltétel biztosítja, hogy az A_2-n kívül más ezt az y üzenetet nem tudja megfejteni.

2. *A_1 aláírt titkos levelet küld A_2-nek.* Ekkor A_1 az $y = f(f(x, D_1), E_2)$ üzenetet küldi el. Más szóval A_1 először kódolja x-et a D_1 titkos kulcsával, majd ennek az eredményét az A_2 nyilvános kulcsával. A fogadó oldalon A_2 így veheti le a két lakatot: $x = f(f(y, D_2), E_1)$. Az első lépésben a birtokában levő D_2 kulccsal visszakapja az $f(x, D_1)$ szót, ami aztán az E_1 nyilvános kulccsal kezelhető. Az aláírást az jelenti, hogy a $\mathbf{K}_2$ feltevés szerint a második lépésben csak az E_1 kulcs fogja nyílt szöveggé alakítani $f(y, D_2)$-t.

10.3. A Rivest–Shamir–Adleman- (RSA-) kód

Az előbb körvonalazott nyilvános kulcsú kriptográfiai séma egy nevezetes megvalósítását szeretnénk bemutatni . Mielőtt nekivágnánk, kis kitérőt teszünk. Szükségünk lesz az *euklideszi algoritmus* egyik változatára, amivel egészek legnagyobb közös osztóját lehet gyorsan meghatározni.

Legyenek $a_0 \geq a_1$ legfeljebb k bit hosszúságú természetes számok. Képezzük az a_i egészeket maradékos osztással a következők szerint:

$$\begin{aligned} a_0 &= q_1 a_1 + a_2, \\ a_1 &= q_2 a_2 + a_3, \\ &\vdots \\ a_{n-2} &= q_{n-1} a_{n-1} + a_n, \\ a_{n-1} &= q_n a_n + a_{n+1}, \end{aligned}$$

úgy, hogy $|a_{i+1}| \leq |a_i/2|$ teljesüljön minden $i > 0$ indexre. A sorozatot addig képezzük, amíg nullát nem kapunk; vagyis $a_{n+1} = 0$. Ez szükségképpen bekövetkezik, mégpedig gyorsan. Ha a két kezdő számunk legfeljebb k bit hosszúságú, akkor $n \leq k$, hiszen minden lépésben – kivéve esetleg az utolsót – legalább egy bittel rövidebb egészet kapunk. Az a_{i+1} az a_i és a_{i-1} számokból maradékos osztással kapható; a q_i vagy $\lfloor a_{i-1}/a_i \rfloor$ lesz, vagy pedig $\lceil a_{i-1}/a_i \rceil$. Ezek közül valamelyikkel biztosan teljesül az $|a_{i+1}| \leq |a_i/2|$ egyenlőtlenség.

A sorozat végétől az eleje felé lépdelve látható, hogy $|a_n|$ osztója mindegyik a_i-nek, így a_1-nek és a_0-nak is. Ha pedig a b pozitív egész osztója a_0-nak és a_1-nek is, akkor a sorozat elejétől indulva kapjuk, hogy b osztója mindegyik a_i-nek, ezért $|a_n|$-nek is. Vagyis $|a_n|$ éppen a_0 és a_1 legnagyobb közös osztója. A legnagyobb közös osztó tehát polinom időben kiszámítható.

Feladat: Mutassuk meg, hogy polinom időben kaphatunk olyan c_0 és c_1 egészeket, melyekre $c_0 a_0 + c_1 a_1 = lnko(a_0, a_1)$, továbbá $|c_0| \leq a_1$ és $|c_1| \leq a_0$. Ezek

alapján elmondhatjuk, hogy a_0 és a_1 legnagyobb közös osztója hatékonyan kifejezhető a két szám egész együtthatós lineáris kombinációjaként, mégpedig nem túl nagy együtthatókkal. (Az euklideszi sorozat elejéről indulva sorra fejezzük ki az a_i számokat a_0 és a_1 segítségével.)

Megjegyezzük még, hogy a legkisebb nemnegatív maradékokat használó euklideszi sorozat (az $|a_{i+1}| \leq |a_i/2|$ követelmény helyett $0 \leq a_{i+1} < a_i$ szerepel) is elég gyorsan eléri a nullát; ennek az igazolása azonban valamivel bonyolultabb.

Ezután ismertetjük a Rivest–Shamir–Adleman- (széles körben használt rövidítéssel RSA-) kriptorendszert. Ahogy korábban megállapodtunk, legyenek $A_1, \ldots, A_n$ a kommunikációs hálózat résztvevői. Először A_i generál két nagy prímszámot, p_i-t és q_i-t (mondjuk 500 bináris jegyűeket). Ez a 9.4.3. pontban látottak szerint hatékonyan megtehető véletlen választásokkal és prímtesztelléssel. Legyen $m_i = p_i q_i$. Ezen felül A_i választ egy véletlen e_i természetes számot, melyre $1 < e_i < m_i$ és $lnko(e_i, \phi(m_i)) = 1$. Itt ϕ az Euler-függvényt jelenti. Tudjuk, hogy $\phi(m_i) = (p_i - 1)(q_i - 1)$. Végül kiszámít egy olyan $0 < d_i < m_i$ egészet, melyre $e_i d_i \equiv 1 \pmod{(p_i - 1)(q_i - 1)}$. Az e_i ellenőrzése, majd pedig d_i számítása az euklideszi algoritmussal történhet. Az utóbbihoz olyan c_0 és c_1 egészeket keresünk, melyekre $c_0 e_i + c_1 \phi(m_i) = 1$, $|c_0| \leq \phi(m_i)$ és $|c_1| \leq e_i$ (lásd az euklideszi algoritmussal kapcsolatos feladatot). A $d_i := c_0 \pmod{\phi(m_i)}$ választás nyilvánvalóan jó lesz.

Az A_i titkos kulcsa a $D_i = (d_i, m_i)$, nyilvános kulcsa pedig az $E_i = (e_i, m_i)$ pár. Az E_i kulcsot közzéteszi, a D_i-t, pontosabban d_i-t pedig megtartja magának.

Feltesszük, hogy a nyílt üzenet olyan x egészek sorozata, melyekre $0 \leq x < m_i$ és $lnko(x, m_i) = 1$ teljesül. Az első feltétel pusztán azt jelenti, hogy az üzenetet blokkonként akarjuk kezelni. A második feltétel nem támaszt komoly megszorítást; m_i-hez képest elenyésző azoknak a $[0, m_i]$ intervallumba eső x egészeknek a száma, amelyek oszthatók p_i-vel vagy q_i-vel.

Az f kódoló/dekódoló függvénynek három argumentuma van. Ezekből az utolsó kettő együtt alkotja a kulcsot. Az f a nyílt szöveg egy x blokkját alakítja át rejtjelezett blokká:

$$f(x, e_i, m_i) := x^{e_i} \pmod{m_i}.$$

Így néz ki az x-nek az A_i nyilvános kulcsával kódolt változata. Feltesszük, hogy az eredmény a legkisebb nemnegatív maradék: $0 \leq f(x, e_i, m_i) < m_i$. Ahogy korábban utaltunk rá, ugyanezzel a függvénnyel végezzük a dekódolást is. A_i az általa kapott titkos üzenet y blokkját ($0 \leq y < m_i$) az alábbi módon fejtheti meg:

$$f(y, d_i, m_i) := y^{d_i} \pmod{m_i}.$$

Nézzük ezután az előző pontban megfogalmazott követelményeket! Tudjuk, hogy $e_i d_i = 1 + \phi(m_i) q$ teljesül alkalmas q egészre. Ezt, és a számelméletből jól

ismert Euler–Fermat-tételt használva

$$f(f(x,e_i,m_i),d_i,m_i) = (x^{e_i})^{d_i} \equiv x^{e_i d_i} \equiv x \cdot (x^{\phi(m_i)})^q \equiv x \cdot 1^q \equiv x \pmod{m_i},$$

vagyis a rejtjelezés után a dekódolás tényleg visszaadja az eredeti üzenetet.

A szorzás kommutativitásából közvetlenül adódik az E_i, D_i kulcsok felcserélhetőségét jelentő

$$f(f(x,e_i,m_i),d_i,m_i) = f(f(x,d_i,m_i),e_i,m_i) = x$$

azonosság. Ami a $\mathbf{K}_1$ feltételt illeti, f hatékonyan számítható gyors hatványozással. A $\mathbf{K}_2$ feltételt az „biztosítja", hogy az e-edik gyökvonásra mod m_i nem ismert polinom idejű módszer (randomizált sem), ha m_i összetett, és nem tudjuk a prímosztóit.

Feladat: Mutassuk meg, hogy p_i, q_i és e_i ismeretében d_i hatékonyan meghatározható. Tehát az m_i felbontását ismerők A_i titkos kulcsát könnyen kideríthetik.

Megjegyzés: Az RSA biztonságosságát illetően elmondhatjuk, hogy a rendszer eddig ellenállt a feltörési kísérleteknek, és szép, sikeres gyakorlati alkalmazásai vannak. Ugyanakkor *nincs bizonyítva*, hogy a feltörése algoritmikusan nehéz feladat lenne. Nem tudjuk egyfelől, hogy az egészek felbontása (m_i ismeretében p_i és q_i meghatározása) nehéz-e. Másfelől az sincs bizonyítva, hogy az RSA elleni sikeres támadáshoz tényleg kell tudni egészeket felbontani; pusztán arról van szó, hogy más esélyes támadási irányról nincs tudomásunk.

Vannak olyan, az RSA-nál bonyolultabb rendszerek, amelyek feltörése bizonyítottan legalább olyan nehéz, mint az egészek prímtényezős felbontásának a feladata.

Példa: Szeretnénk itt szemléltetni az RSA-rendszer működését. Az olvasó kényelméért eltekintünk a többszáz bites prímek használatától; beérjük öt bittel is. További egyszerűsítésként elhagyjuk az i indexet (A_i helyett A-t, p_i helyett p-t írunk, stb.).

Legyen $p = 11$ és $q = 17$. Ekkor az m modulus értéke $11 \cdot 17 = 187$. A $\phi(m)$ értéke $10 \cdot 16 = 160$. Legyen az e nyilvános kulcs 9, és határozzuk meg a d titkos kulcsot. E célból az euklideszi algoritmust hívjuk segítségül. Az $a_0 = 160$, $a_1 = 9$ bemenettel a számítás így alakul:

$160 = 18 \cdot 9 - 2$,
$9 = (-4) \cdot (-2) + 1$,
$-2 = (-2) \cdot 1 + 0$.

Innen látjuk egyrészt, hogy $lnko(\phi(m), e) = lnko(160, 9) = 1$ tényleg igaz. Másrészt az első egyenlőségből -2-t kifejezve és a másodikba helyettesítve $9 = (-4) \cdot$

$(160-18\cdot 9)+1$ adódik, amiből átrendezéssel kapjuk, hogy $-71\cdot 9+4\cdot 160=1$. A titkos kulcs tehát $d=89\equiv -71 \pmod{160}$.

Egy az A-nak szánt x üzenetblokk kódja az $f(x,9,187):=x^9 \pmod{187}$ szó lesz. Például az $x=3$ üzenetnek $3^9=3^4\cdot 3^4\cdot 3=81\cdot 81\cdot 3\equiv 16\cdot 3\equiv 48 \pmod{187}$ lesz a kódja. Az y rejtjelezett blokk dekódolása, visszafejtése az $f(y,89,187):=y^{89} \pmod{187}$ függvénnyel történik.