Kriptográfiai protokollok intelligens kártyákon

Borbás Gergely

Témavezető: Csirmaz László

Tartalomjegyzék

1. fejezet

Bevezető

Az utóbbi időben a technika rohamos fejlődése sok tekintetben megváltoztatta, felgyorsította megszokott világunkat. Az új körülmények a kereskedelem területén is éreztetik hatásukat, egyre inkább terjed az otthonról, az interneten keresztüli vásárlás igénye, de persze még sok más példát is találhatnánk arra, hogy miért vált kényelmetlenné a készpénz használata. Erre a kényelmetlenségre jelent megoldást a bankkártyák megjelenése. Természetesen a váltás nem csupán a kártyák használóinak az érdekeit szolgálja (persze az sem lebecsülendő: egy jól megtervezett és kiépített rendszer készpénzhez közeli rugalmasságot kínál magsfokú biztonság mellett), a bankok is szorgalmazzák a chipkártyák elterjedését, ami nem is csoda, hiszen sokat számít, hogy az emberek elköltésre váró pénze a banknál forog, dolgozik, ahelyett, hogy otthon, a pénztárcákban lapulva tétlenül várja, hogy használják.

Több szempontból is szükség van tehát a váltásra, az elektronikus fizetőeszközök bevezetésére. Ez persze nem olyan egyszerű, az adódó problémákat meg kell oldani, hogy az igényeket minden oldalról kielégíthessük. Lássuk, melyek azok a feladatok, amelyekkel meg kell birkóznunk, mik az igazi pénznek illetve a fizetésnek, pénzcserének azok a tulajdonságai, amiket modelleznünk kell, ha egy jól működő rendszert szeretnénk tervezni:

- Atomosság, ami azt jelenti, hogy a tranzakciónak egészként kell viselkednie, vagy az összes lépésnek végre kell hajtódnia, vagy egynek sem. Pontosabban bármi is történik tranzakció közben, a rendszer végül két állapot közül lehet egyben: vagy úgy, mintha rendben lefolyt volna miden művelet, vagy pedig úgy, mintha el sem kezdődött volna semmi. Ez modellezi azt, hogy ne történhessen meg, hogy fizetéskor például csak a pénz kerül át az egyik oldalról a másikra, amiért fizettünk, az nem.

- Konzisztencia, azaz a rendszernek egy értelmes, jól definiált állapotban kell lennie a tranzakció elkezdése előtt, és hasonlónak igaznak kell lennie a folyamat befejeződése után, tehát nem szabad, hogy csak „félig" kerüljön át a pénz, illetve elvesszen közben valahogy (itt már, ott még nincs meg).

- Izoláció: követelmény, hogy párhuzamos folyamatok ne kommunikálhassanak egymással, egy érme ne vehessen részt egyszerre két külöböző tranzakcióban.

- Tartósság: lényeges, hogy egy sikeresen véghezvitt tranzakció végeredménye megmaradjon, a rendszer szereplő ne tudjanak önkényesen az előző állapothoz visszatérni, ha egyszer fizettünk egy érmével, akkor ne tudjuk megtartani, vagy tagadni a fizetés tényét.

Ezek a követelmények rögzítenek sok olyan fontos tulajdonságot, főleg biztonsági szempontból, amelyet egy elektronikus – vagy bármely egyéb – fizetési rendszertől elvárunk. A készpénznek azonban ezek mellett is vannak olyan ismérvei, amelyeket esetleg szeretnénk, ha az új rendszerek is teljesítenének. Ezek közül egy a pénz felhasználójának az anonimitása, azaz a fizetésnél nem történik meg a fizető azonosítása, az értéket az érme önmagában képviseli, garantálja; utólag nem bizonyítható – természetesen ideális fizetés esetén –, hogy kié volt eredetileg a pénz. Egy másik tulajdonság a pénz továbbadhatósága, azaz az egyik fizetéskor kapott érmét változtatás nélkül azonnal újra, ugyanolyan értékben ismét föl lehet használni az eredeti célra. Ezekre a fizetésekre igaz az is, hogy ha egyazon érmével kétszer fizettünk, akkor nem lehet megállapítani, hogy ugyanaz az érme vett részt a két tranzakcióban. Ugyanez az összekapcsolhatatlanság természetesen igaz az emberekre is, két különböző fizetésről nem lehet eldönteni, hogy azonos volt-e a két fizető személy.

Természetesen a valóságban ezeket a célokat sokkal könnyebb elérni: az állam sok tartós, egyforma, ám lemásolhatatlan érmét gyárt, és máris megoldódnak a biztonság illetve az anonimitás problémái is. Az elektronikus kereskedelemben kézenfekvő, hogy az értéket az információ képviselje, egy jelsorozat legyen az érme, ez azonban rögtön két gondot is okoz: információt generálni sem túl nehéz, lemásolni azonban mindennél egyszerűbb. Valahogy garantálni kell hát, hogy a felhasználó ne tudjon saját maga olyan adathalmazt készíteni, amely érvényes fizetőeszköznek számít, illetve meg kell akadályozni, hogy a bank által generált információt több alkalommal is felhasználhassa. A kettő közül az utóbbi probléma tűnik nagyobbnak, a két kínálkozó megoldás egyike, hogy a bank *védi* az információ másolását; olyan eszközzel szereli föl a felhasználót, amely szükséges a fizetéshez és megakadályozza, hogy egy érmét kétszer költsenek el, ebben az esetben azonban van valaki, illetve valami, ami a felhasználó minden tranzakcióját figyeli, tehát veszélyben a kívánt anonimitás. A másik lehetőség, hogy a bank személyre szabott érméket gyárt, így a duplán elköltött érme által a vétkes személyazonossága kideríthető. Itt azonban ismét ellentmondásgyanús helyzet előtt állunk, hiszen azt szeretnénk, hogy az érme egyszeri elköltése semmilyen információt ne hordozzon arról, aki fizetett vele, míg két elköltés esetén minden ismertté váljon.

Az ilyen és ehhez hasonló problémákra kínál megoldást a matematika, pontosabban a kriptográfia használata. Modelljeinkben egymás mellett fogjuk használni a modern nyilvános kulcsú, illetve a klasszikus, szimmetrikus rendszereket, amelyek bizonyítottan – illetve legtöbbször csak sejtetten, bizonyos, nem igazolt, ám régóta vizsgált és igaznak tűnő feltételezésekre alapozva – nehézzé teszik a hamisítást, megvédik a felhasználó anonimitását, garantálják a felsosrolt „ACID” tulajdonságok teljesülését.

Először áttekintjük a manapság használatos rendszereket, osztályozzuk őket különböző szempontok alapján, megvizsgáljuk, mikor melyiket érdemes alkalmazni. A második fejezetben bevezetésre kerülnek a segédeszközeink, megismerkedünk azokkal a problémákkal, melyek nehézsége, bonyolultsága segítséget nyújt, hogy bizonyos dolgokat nehézzé és bonyolulttá tegyünk egy átlagos ember számára, mégis könnyű maradjon annak, aki ismer bizonyos plusz információt. Ezekre alapozva készítünk néhány kriptográfiai *makrót*, alapalgoritmust, melyekből a harmadik fejezetben felépítjük azokat a protokollokat, amelyek megvalósíthatják az elektronikus kereskedelmet. A negyedik fejezetben az addigi leegyszerűsített rendszerek egy lehetséges kiterjesztésének problémájával foglalkozunk, az érmék továbbadásának, az újrafizetésnek a lehetőségét vizsgáljuk.

Elöljáróban még néhány megjegyzés: a vázolt rendszerek szereplői természetes módon három csoportba oszthatók: először is van egy bank (általában $\mathcal{B}$-vel jelöljük), valaki ($\mathcal{U}$), akinek számlája van a banknál és szeretne fizetni valaki másnak ($\mathcal{S}$), aki szintén kapcsolatban van a bankkal. Ehhez a három szereplőhöz három protokoll tartozik, amely az elektronikus érme (C) életútja is egyben: C először $\mathcal{B}$-től $\mathcal{U}$-hoz kerül, mely művelet során $\mathcal{U}$ bankszámlája C ellenértékével csökken, majd $\mathcal{U}$ fizet $\mathcal{S}$-nél C-vel, megkapja azt a szolgáltatást, amit szeretne, végül pedig $\mathcal{S}$ visszajuttatja C-t $\mathcal{B}$-hez, amikor az érme értékének megfelelő összeg jóváíródik $\mathcal{S}$ számláján.

Természetesen mindig megköveteljük az említett biztonsági kritériumok teljesülését, gyakran ügyelünk a felhasználók és a fizetések párosíthatatlanságára, és csak utólag nézzük meg, hogy mi történik, ha szeretnénk, hogy C-fizetésre használhassa föl $\mathcal{S}$, ne kelljen visszaküldenie $\mathcal{B}$-nek és újat generáltatnia, ha ő is vásárolni akar.

Általában $\mathcal{B}$-t becsületesnek tételezzük föl, mint akinek nagyobb kárt okoz a kipattanó botrány, mint amennyi pénzt egy-egy felhasználót, boltot becsapva nyerhet. Amennyiben ez ellen is be szeretnénk biztosítani magunkat, szükségünk van egy negyedik személyre, aki szavatolja a bankokat. Továbbá nem foglalkozunk azzal sem, hogy hogyan történik meg a felhasználók bankbeli regisztrációja, könnyen látható, hogy tisztán kriptográfiai eszközökkel nem lehet garantálni az első találkozás hitelességét, a rendszerbeli azonosítót és az igazi személyazonosságot valahogyan külső eszközökkel kell összekötni.

1.1. Osztályozási szempontok

Az elektronikus fizetési lehetőségeket, módszereket (1) nyomán az alábbiak szerint csoportosíthatjuk:

Előre kifizetett pénz - fizetés utasításra: Az előbbi eset valójában az elektronikus pénz megvalósítása. A felhasználó valamikor a vásárlás előtt valódi értékeit elektronikus érmékre konvertálja még a fizetési tranzakció előtt (ekkor a megfelelő összeggel csökken a számlája a banknál), majd a boltos a kapott információt a bankhoz elküldve hozzájut pénzéhez.

A módszer biztonságtechnikai szempontból legkényesebb pontja annak megakadályozása, hogy a felhasználóhoz került könnyen másolható információt ne lehessen több alkalommal is felhasználni. Szintén életbevágóan fontos a hamisítás lehetetlenné tétele, ne lehessen a bank segítsége nélkül érvényes pénzt generálni.

Esetünkre jó példát szolgáltatnak a telefonkártyák.

A második lehetőségnél a tranzakció folyamán a felhasználó engedélyezi a kereskedőnek, hogy a banknál megterhelhesse a számláját, tulajdonképpen kitölt egy csekket, ami később a banknál beváltható.

A rendszer megvalósításánál arra kell legfőképp vigyázni, hogy a vásárló ne adhassa ki másnak magát, ne legyen hamisítható az aláírása.

Ilyen elven működnek egyes külföldön elterjedt kreditkártyarendszerek.

On-line - off-line: A megkülönböztetés alapja, hogy a fizetési protokoll két- vagy háromszemélyes-e, azaz szükség van-e a bank részéről a tranzakció hitelesítésére (ekkor nevezzük on-line rendszernek) avagy nincs (off-line fizetés). Az utóbbi eset nehézségei hasonlatosak az előző pontban említettekéhez, a bank segítsége mindenképpen megkönnyíti a csalások kiszűrését, ám nagy árat kell érte fizetni: egy harmadik, távoli, esetleg éppen nagyon elfoglalt szerverrel történő kapcsolat jelentősen megnöveli a tranzakció időigényét. On-line módon működnek a magyar pénzintézetek által kibocsátott bankkártyák.

Számláló - érmék: A rendszerek között az alapján is különbséget tehetünk, hogy a kártyán hogyan tárolódik a felhasználó pénze. Az első esetben egy számláló található a kártyán, aminek a pillanatnyi állása az éppen rendelkezésre álló mennyiséget jelzi, a pénz felvétele és a fizetés pedig ennek a számlálónak a megfelelő átállítását jelenti. Ez a módszer elegáns és hatásos: kevés tárolóhelyet igényel, bármekkora összeg kifizethető vele (természetesen a kártyán tárolt mennyiség behatárolja a lehetőségeket).

A második lehetőség elektronikus pénzérmék gyártása. Amikor a felhasználó pénzt vesz föl a banktól, akkor előre meghatározott értékű „információcsomagokat kap”, ezek tárolódnak a kártyán, kerülnek át fizetéskor a kereskedőhöz, majd jutnak vissza végül a bankhoz.

Nyilvánvaló hátránya a rendszernek, hogy csak olyan fizetéseket lehet lebonyolítani, amelyek összegét pontosan elő lehet állítani a kártyán lévő érmékből, illetve a fizetés végén egy újabb fizetési procedúrával a kereskedő „visszaadhat”, azonban ez jelentősen elbonyolítja a helyzetet. Nagy előny viszont, hogy érméket csak a bank tud generálni, a hamisításhoz a bank digitális aláírását kellene föltörni, míg egy számláló jogosulatlan átállítását sokkal könnyebbnek tűnik megoldani. így a rendszer kijátszásának egyetlen módja a használatban lévő érvényes pénz lemásolása, ez azonban viszonylag könnyen kivédhető azáltal, ha az érmékbe elkódoljuk a jogosult felhasználó személyazonosságát. Továbbá egyedül az érmés rendszereknek van meg az a jó tulajdonságuk, hogy egyfelől biztonságot nyújtanak a banknak, másfelől képesek megőrizni a felhasználó anonimitását, azaz a kereskedőnek átadott érméről nem lehet eldönteni, hogy melyik alkalommal, kinek készítette a bank. Hogy ez ne mondjon ellent az előbb említett személyre szabott fizetőeszköznek, el kell érnünk, hogy a vásárló azonosítóját csak akkor lehessen dekódolni, ha megpróbálta kijátszani a rendszer szabályait.

Csak szoftveres - megfigyelőt is tartalmazó rendszerek: A modellek megvalósításában természetszerűleg a felhasználóknak szükségük van valamilyen speciális eszközre, bankkáryákra vagy számítógépekre, amiken keresztül kapcsolatba léphetnek a bankkal illetve a kereskedőkkel. Egy fontos szempont a rendszerek osztályozásánál, hogy a felhaszáló által használt eszköz tartalmaz-e speciális, a bank által készített hardvert, amely fizikailag védett a külső manipulációk ellen, vagy a rendszer biztonsága csupán a felhasznált kriptográfiai protokollok erején múlik.

A speciális hardver (megfigyelő) használata mindenképpen megkönnyíti a hamisítás elleni védekezést: amennyiben a megfigyelő érvényesítése is szükséges egy tranzakciónál, úgy nem lehet egy adott érmét több alkalommal is elkölteni, hiszen másodszorra a megfigyelő egyszerűen megtagadja a pénz hitelesítését.

Az említett változatok ezen előny mellett azonban rendelkeznek két jelentős hátránnyal is: lényegesen drágábbak, mint azok a rendszerek, ahol nincs szükség megfigyelőre, az utóbbiak szoftvere bármilyen számítógépen futtatható, a banknak nem kell ellátnia a klienseit fizikai eszközökkel. A másik hátrány, hogy a felhasználók anonimitása veszélybe kerülhet, ha a fizetési tranzakció közben a megfigyelő szabadon kommunikálhat a protokoll másik résztvevőjével.

A csak kriptográfiai eszközöket használó rendszerek az olcsóság és az egyszerűség mellett egy nagy problémával is bírnak: az érméket többször is el lehet költeni, a bank csak abban bízhat, hogy utólag leleplezi a tettest. Ezért is fontos, hogy az ilyen érméknek, tranzakcióknak nem szabad nagy értéket képviselniük. Így elkerülhető, hogy egy különösen nagy értékű csalás után a bűnös a tett és a felfedezése közti időben külföldre meneküljön, személyazonosságot váltson.

2. fejezet

Kriptográfiai segédeszközök

2.1. Az RSA rendszer

A modern kriptográfia egyik leggyakrabban alkalmazott eszköze a Rivest, Shamir és Adleman által kidolgozott nyilvános kulcsú rendszer, amelyet (2)-ban publikáltak. A szerzők a faktorizálás nehézségére építenek, miután nem ismeretesek gyors algoritmusok, amelyek egy megfelelően nagy (mondjuk 150 jegyű) számot gyorsan prímtényezőire bontanának. Fontos szerepet kap még, hogy ezzel ellentétben viszonylag könnyen meg tudjuk mondani nagy bizonyossággal egy hasonló nagyságú számról, hogy prím-e vagy sem (Rabin-Miller algoritmus).

Feltételezésünk tehát a következő:

- Az x ismeretlenű $x^e \equiv y \pmod{n}$ egyenletet rögzített e, y és n számokra, ahol utóbbi összetett, n prímfelbontásának ismerete nélkül nem tudjuk hatékonyan megoldani.

Jelenleg ezt nem tudjuk bizonyítani, azonban a következő állítás teljesül, ami a faktorizálás és a gyökvonás kapcsolatát mutatja:

2.1.1. állítás. Legyenek p, q nem ismert prímek és $n = pq$. Ha létezik algoritmus, ami y bemenetre polinomiális idő alatt megad egy x-et, amire $x^2 \equiv y \pmod{n}$, akkor p és q polinomiális időben nagy valószínűséggel kiszámítható.

Bizonyítás. Elemi algebrai megfontolásokkal belátható, hogy egy $a \in \mathbb{Z}_p$ számnak $\bmod p$ vagy pontosan két négyzetgyöke van vagy egy sincs. Valóban: $b^2 \equiv a \pmod{p}$-ből $(p-b)^2 \equiv a \pmod{p}$ következik. $b^2 \equiv c^2 \pmod{p}$-ből pedig $(b-c)(b+c) \equiv 0 \pmod{p}$, azaz vagy $b \equiv c \pmod{p}$ vagy pedig $b \equiv -c \pmod{p}$. Így $a \in \mathbb{Z}_n$-re $b^2 \equiv a \pmod{n}$-nek ha van megoldása $\mathbb{Z}_n$-ben, akkor pontosan négy van. Hiszen $b^2 \equiv a \pmod{n}$ pontosan akkor, ha $b^2 \equiv a \pmod{p}$ és $b^2 \equiv a \pmod{q}$ egyszerre teljesül. A lehetséges négy megoldáspár $\bmod n$ négy különböző maradékosztályt határoz meg.

Ezekután válasszunk $x_1, x_2, \cdots, x_k \in \mathbb{Z}_n$ számokat véletlenszerűen. Legyen $y_i \equiv x_i^2 \pmod{n}$. Tápláljuk be az algoritmusnak az y_i számokat. Miután az y_i-knek van gyökük $\mathbb{Z}_n$-ben, ezért léteznek további gyökeik is: $z_1, z_2, \cdots, z_k$ illetve ezek inverzei. Ha az x_i-ket véletlenül választottuk, akkor az algorimus $\frac{1}{4}$ valószínűséggel fogja az $x_i, -x_i, z_i$ és $-z_i$ számok egyikét választani, azaz $\frac{1}{2}$ valószínűséggel megkapjuk z_i-t. Összességében tehát $(\frac{1}{2})^k$ valószínűséggel megismerünk egy z_i-t. Jelöljük ezt a továbbiakban z-vel, a hozzátarozó x_i-t pedig x-szel.

Mivel $z^2 \equiv x^2 \pmod{n}$, ezért $z^2 \equiv x^2 \pmod{p}$ és $z^2 \equiv x^2 \pmod{q}$ is fönnáll. A $z \equiv x \pmod{p}$ és $z \equiv x \pmod{q}$ esetből $z \equiv x \pmod{n}$ következik, a $z \equiv -x \pmod{p}$, $z \equiv -x \pmod{q}$-ból pedig $z \equiv -x \pmod{n}$. Azonban föltételezésünk szerint ezek a lehetőségek nem fordulhatnak elő, így az általánosság megszorítása nélkül föltehető, hogy $z \equiv x \pmod{p}$, valamint $z \equiv -x \pmod{q}$.

Ekkor

$$(z-x)(z+x) \equiv z^2 - x^2 \equiv 0 \pmod{n},$$

azaz

$$n \mid (z-x)(z+x),$$

úgy, hogy $p \mid z - x$, de $q \nmid z - x$, amiből az következik, hogy n és $z - x$ legnagyobb közös osztója p, amit a polinomiális futási idejű euklideszi algoritmussal meghatározhatunk, azaz megállapítottuk n prímfelbontását. ■

Jelöljük a rendszer szereplőit szokás szerint $\mathcal{A}$-val és $\mathcal{B}$-vel. Inicializálásképpen $\mathcal{A}$ választ két nagy prímszámot p-t és q-t, a későbbiekben minden számítást $\bmod n$ fogunk végezni, ahol $n = pq$. Természetesen ahhoz, hogy a rendszer biztonságos és jól használható legyen, szükséges, hogy bizonyos szempontokat figyelembe vegyünk p és q megválasztásakor, ilyen követelmény például, hogy $\varphi(n)$-nek ne legyen kicsi páratlan prímosztója, valamint, hogy p és q ne legyen túl közel egymáshoz.

$\mathcal{A}$ választ egy d számot $\mathbb{Z}^*_{\varphi(n)}$-ből – azaz d relatív prím kell, hogy legyen $(p-1)(q-1)$-hez –, ami a titkos kulcsa lesz. A nyilvános kulcsa az (n, e) pár lesz, ahol e-t az

$$e \equiv \frac{1}{d} \pmod{\varphi(n)}$$

definiálja. Természetesen p-t, q-t, d-t illetve $\varphi(n)$-t $\mathcal{A}$ nem hozza nyilvánosságra. Itt is vigyázni kell azonban, hogy helyesen válasszuk meg az értékeket: ismeretes például, hogy $d < n^{0.292}$ esetén a rendszer nem biztonságos (lásd (3)). d-vel ellentétben e értékét gyakran célszerű kicsinek választani, amivel a titkosítás sebességét növelhetjük meg.

A nyilvános kulcs segítségével $\mathcal{B}$ tud üzenetet küldeni $\mathcal{A}$-nak, ahol az üzenet egy $\mathbb{Z}_n$-beli m szám (amennyiben m túl hosszú lenne, akkor 0 és n közé eső darabokra vágjuk), $\mathcal{B}$ kiszámítja $z \equiv m^e \pmod{n}$-et és eljuttatja $\mathcal{A}$-hoz. Az üzenet visszafejtése a kódoláshoz hasolóan történik:

$$m \equiv z^d \equiv (m^e)^d \pmod{n},$$

hiszen

$$ed = 1 + r\varphi(n) = 1 + r(p-1)(q-1),$$

ahol $r \in \mathbb{N}$. Így $p \nmid m$-re a „kis” Fermat-tételből tudjuk, hogy $m^{p-1} \equiv 1 \pmod{p}$, emiatt:

$$(m^e)^d \equiv m^{ed} \equiv m(m^{p-1})^{r(q-1)} \equiv m \pmod{p},$$

míg $p \mid m$-re

$$m^{ed} \equiv 0 \equiv m \pmod{p}.$$

Tehát minden m-re igaz

$$m^{ed} \equiv m \pmod{p}.$$

Teljesen hasonlóan láthatjuk, hogy

$$m^{ed} \equiv m \pmod{q}$$

is teljesül, azaz

$$m^{ed} \equiv m \pmod{pq = n}.$$

2.2. A Diszkrét Logaritmus Probléma

Legyen q tetszőleges prím, jelölje G_q a q-ad rendű csoportot, amelyről így tudjuk, hogy izomorfia erejéig egyértelmű, továbbá azt is, hogy ciklikus, valamint minden eleme (az egység kivételével) generálja az egész csoportot. Jelöljük k-val q jegyeinek számát ($k = \lceil \log_{10} q \rceil$).

2.2.1. definíció. Diszkrét Logaritmus Problémának nevezzük az egyértelmű $\log_g h \in \mathbb{Z}$ kitevő megtalálását, amelyre g-t emelve h-t kapjuk. Egy (nem feltétlenül determinisztikus) algoritmus megoldja a Diszkrét Logaritmus Problémát, ha létezik $g_1(x)$ és $g_2(x)$ polinom, hogy G-n egyenletes eloszlás szerint generált $g \neq 1, h$ inputokra t idő alatt legalább p valószínűséggel megadja $\log_g h$-t,ahol $t < g_1(k)$ és $p > \frac{1}{g_2(k)}$.

2.2.2. állítás. Ha egy algoritmus megoldja a Diszkrét Logaritmus Problémát, akkor létezik olyan algoritmus is, ami tetszőlegesen nagy valószínűséggel számolja $\log_g h$-t.

Bizonyítás. Az eredeti algoritmust (r_1-et és r_2-t véletlenül választva) a g^{r_1}, h^{r_2} páron n-szer futtatva legalább $1-(1-p)^n$ valószínűséggel kapunk egy helyes eredményt. Az eredmények helyességét könnyen ellenőrizhetjük, mindössze egy hatványozást kell hozzá elvégezni, amit általában gyorsan meg tudunk tenni. A helyes eredményt $\frac{r_1}{r_2} \mod q$-val szorozva $\log_g h$-t kapjuk. ■

2.2.3. állítás. Ha létezik egy olyan algorimus bármilyen rögzített $g \neq 1$-re, aminek inputként h-t adva kiszámítja $\log_g h$-t, akkor létezik olyan is, ami megteszi ezt tetszőleges g, h-ra.

Bizonyítás. Jelöljük g_1-gyel az eredeti algoritmusunk „alapszámát”. Ekkor $r_1 = \log_{g_1} g$ és $r_2 = \log_{g_1} h$-val $\log_g h = \frac{r_2}{r_1}$. ■

Természetesen a DLP megoldásának a nehézsége a csoport realizációján múlik, például $\mathbb{Z}_q^+$-ban a feladat triviális, csupán egy q szerint vett osztási maradékot kell számolni, melyet az Euklideszi algoritmussal polinomális időben tehetünk meg. Azonban G_q-nak (egy $p = nq+1$ alakú prímre) $\mathbb{Z}_p^*$ egy q elemű részcsoportját választva nem ismeretes hatékony algoritmus. (Amennyiben külön nem hangsúlyozzuk, a továbbiakban G_q-val ezt a csoportot jelöljük.) Ezzel kapcsolatos a következő

2.2.4. feltételezés. *Nincs olyan algoritmus, amely tetszőleges, konkrét G_q-ra megoldaná a Diszkrét Logaritmus Problémát.*

Feltételezésünk eddig ugyan nem nyert bizonyítást, így az előbbiekben vázolt speciális esetről sem mondhatunk semmi bizonyosat, ám az eddig fölfedezett algritmusok elegendően lassúak ahhoz, hogy a későbbiekben nyugodtan támaszkodhassunk a DLP nehézségére. Ahhoz, hogy G_q-t a későbbiekben sikeresen használhassuk, feltétlenül szükséges, hogy (a logaritmusszámítással ellentétben) hatékonyan tudjunk különböző manipulációkat végezni G_q-val kapcsolatban: egy számról eldönteni, hogy eleme-e a csoportnak; elem inverzét, adott hatványát, valamint két elem szorzatát számítani; a csoportból egyenletes vagy közel egyenletes eloszlás szerint egy elemet kiválasztani (közel egyenletes eloszlás alatt azt értjük, hogy néhány -polinomiális számú- elem választása nem megengedett, például, amikor garantáltan különböző elemekre van szükségünk), valamint megállapítani két elemről, hogy egyenlőek-e. Könnyen látszik, hogy ezekre a feladatokra léteznek gyors algoritmusok az említett G_q-ban.

Gyakorlati szempontokat figyelembe véve G_q definiálásánál p-t egy legalább 512, q-t pedig minimálisan 140 bit hosszúnak szokás választani. Az ilyen csoportoknál az említett műveletek ismert számítási módszereinek sebességei megfelelnek a kívánalmaknak.

2.3. A hash függvény

Az egyik alapvető kriptográfiai segédeszköz, amely szinte minden alkalmazásben jelentős szerephez jut, a továbbiakban $\mathcal{H}$-val jelölt hash függvény. Formálisan $\{0,1\}^*$-ból $\{0,1\}^n$-be képez valamilyen rögzített n-re. Legfőbb kritériuma, hogy *egyirányú* legyen, ezt kétféleképen is szokták definiálni:

2.3.1. definíció. (gyenge változat) Ne létezzen olyan n-ban polinomiális futási idejű algoritmus, ami x inputra talál olyan $y \neq x$-t, hogy $\mathcal{H}(x) = \mathcal{H}(y)$.

2.3.2. definíció. (erős változat) Ne létezzen olyan n-ban polinomiális futási idejű algoritmus, ami talál olyan x-et és y-t, hogy $x \neq y$ és $\mathcal{H}(x) = \mathcal{H}(y)$.

A későbbiekben a második tipusú függvényeket nevezzük hash függvényeknek. Nyilvánvaló, hogy k-t elegendő nagyra kell megválasztanunk, hogy ne lehessen például egyszerű kereséssel gyorsan találnunk megfelelő (x, y) párt. A gyakorlatban általában ezért k-t legalább 160-ra szokták választani. Bizonyítás nélkül megemlítjük az alábbi lemmát:

2.3.3. lemma. *(Születésnap-támadás) Egyenletesen véletlenül választva x, y párokat $O(2^{\frac{k}{2}})$ próba után $\frac{1}{2}$-nél nagyobb eséllyel találunk x, y különböző számokat, amikre $\mathcal{H}(x) = \mathcal{H}(y)$.*

Bizonyítás. Az állítás a jól ismert *Születésnap-paradoxonhoz* kapcsolódik, mint a neve is mutatja; bizonyítása egyszerű, ám hosszas számolás. ■

Alkalmazásainkban gyakran a hash függvény értékkészlete gyakran nem a teljes $\{0, 1\}^n$ lesz, hanem például G_q, valamint az argumentumok számát sem fogjuk egyben korlátozni, ezt a problémát könnyen megoldhatjuk az argumentumok konkatenálásával.

Bár léteznek „elméleti” hash függvények is, amelyek bizonyíthatóan jól viselkedek, ezeket a gyakorlatban ritkán használják viszonylagos lassúságuk miatt; bizonyos „bitkevergetésen” alapuló módszerek (például az SHA-1) a tapasztalatok alapján megfelelő eredményt adnak és nagyságrendekkel gyorsabbak.

2.4. Személyazonosság igazolása

Az eddig megismert módszerek segítségével könnyen gyárthatunk egy olyan protokollt, mely használatával az egyik szereplő ($\mathcal{P}$) meggyőzheti a másikat ($\mathcal{V}$) arról, hogy ő tényleg az, akinek állítja magát. Természetesen szükségünk van bizonyos előkészületekre: $\mathcal{P}$ és $\mathcal{V}$ először megállapodik egy bizonyos G_q-ban, a protokoll végrehajtása folyamán ezen G_q egyes elemei lesznek az egymáshoz intézett üzenetek. Második lépésként választanak egy $g \in G_q \setminus \{1\}$ elemet; majd $\mathcal{P}$ legenerálja a *titkos kulcsát*, azaz véletlenül választ egy $x \in \mathbb{Z}_q$

számot, ami ismerete fogja őt azonosítani. Kiszámítja $h = g^x$-t (természetesen $\mod p$), majd h-t ismerteti $\mathcal{V}$-vel is, a (g, h) pár lesz az ő *nyilvános kulcsa*. A nyilvános kulcsot $\mathcal{V}$ eltárolja az adatbázisában azzal az információval, hogy $\mathcal{P}$-hez tartozik. Az eddigi lépések nem tartoztak szorosan a protokollhoz. Természetesen annak az ellenőrzését is meg kell oldani – és erre nem vagyunk képesek tisztán matematikai eszközökkel –, hogy $\mathcal{P}$ tényleg az-e, akinek kiadja magát.

Az előkészületek után lássuk mi történik, amikor valamikor később a már regisztrált $\mathcal{P}$ kapcsolatba akar lépni $\mathcal{V}$-vel, és meg akarja győzni arról, hogy ő tényleg $\mathcal{P}$:

1. $\mathcal{P}$ választ egy véletlen számot: $w \in \mathbb{Z}_q$, kiszámítja $a = g^w$-t és elküldi $\mathcal{V}$-nek.

2. $\mathcal{V}$ generál egy $c \in \mathbb{Z}_q$-t és elküldi $\mathcal{P}$-nek.

3. $\mathcal{P}$ válasza $r = cx + w \pmod q$.

$\mathcal{V}$ elfogadja $\mathcal{P}$ igazolását, ha $g^r = h^c a$. A fenti protokoll teljesíti azt a három kritérimot, amit elvárunk tőle:

- Ha $\mathcal{P}$ követi a protokoll szabályait, akkor $\mathcal{V}$ elhiszi, hogy valóban az, akinek állítja magát (tényleg tudja x-et).

- x ismerete nélkül $\mathcal{P}$ nem tudja meggyőzni $\mathcal{V}$-t, csak elenyésző valószínűséggel.

- Tetszőleges megfigyelő a protokoll polinomiálisan sok végrehajtásából nem tud elegendő információt nyerni ahhoz, hogy nem elhanyagolható valószínűséggel el tudja játszani $\mathcal{P}$ szerepét.

Az első tulajdonság nyilvánvaló, hiszen így konstruáltuk meg a protokollt, a második kettő pedig a DLP föltételezésből következik.

2.5. A Schnorr-féle aláírás

A fönti személyazonosságot igazoló protokollból könnyen készítetünk digitális aláírást is.

Adott a következő szituáció: $\mathcal{U}$ szeretne hitelesíteni egy m adatsorozatot, azaz bizonyítékot szeretne arra kapni (amit majd később fölhasználhat), hogy az adott pillanatban m milyen értéket tartalmazott. Bizonyítékként elfogadjuk $\mathcal{B}$ állítását.

Első lépésként $\mathcal{B}$ generál egy nyilvános (g, h) kulcsot a hozzátartozó $x = \log_g h$ titkos kulccsal. A (g, h) párt nyilvánosságra hozza egy előre rögzített $\mathcal{H}$ hash-függvénnyel együtt.

Amikor $\mathcal{U}$ szeretne egy hitelességet igazoló aláírást kapni egy m számra, akkor ezt megteheti úgy, hogy a már ismert identifikációs protokollt játsza le $\mathcal{B}$-vel, azzal a különbséggel, hogy c-t nem véletlenül választjuk:

1. $\mathcal{B}$ generál egy véletlen $w \in \mathbb{Z}_q$-t, majd közli $\mathcal{U}$-val $a = g^w$-t.

2. $\mathcal{U}$ elárulja $\mathcal{B}$-nek $c = \mathcal{H}(m, a)$-t.

3. $\mathcal{B}$ kiszámítja c^x-t és elküldi $\mathcal{U}$-nak.

$\mathcal{U}$ hitelesnek tekinti az aláírást, ha $g^r = h^c a$.

A (c, r) pár az m üzenet Schnorr-aláírása, amennyiben $c = \mathcal{H}(m, g^r h^{-c})$. $\mathcal{H}$ használata garantálja egy későbbi ellenőrző számára, hogy a protokoll lépései a megfelelő sorrendben hajtódtak végre, hiszen például $c = \mathcal{H}(m)$ választással r-et véletlenszerűen választva $a = \frac{g^r}{h^c}$-vel $\mathcal{U}$ „érvényes" alírást kaphatna.

Sajnos rossz szerkezetű $\mathcal{H}$-val előfordulhat, hogy néhány m_i és a hozzájuk tartozó (c_i, r_i)-k ismeretében $\mathcal{U}$ $\mathcal{B}$ közreműködése nélkül is tud valamely m-re algebrai úton egy helyes (c, r) aláírást produkálni. Ezért is fontos, hogy megbízható, alapos vizsgálatoknak alávetett hash-függvényeket alkalmazzunk, amelyek a lehetőségekhez képest ellenállnak az efféle támadásoknak.

A módszer egyik hiányossága az, hogy $\mathcal{B}$ – ugyan m-et nem, de – m aláírását megismeri a protokoll végrehajtásakor. Ez a későbbiekben ott kap szerepet, hogy bizonyos rendszerekben m fogja játszani az érme szerepét, és mivel említettük, hogy egyik célunk az, hogy a pénzt anonim módon költhessük el, $\mathcal{B}$ könnyedén kikeresheti az adatbázisából, hogy az adott tranzakcióban szereplő aláírást melyik felhasználó számára készítette, a fönti célt ez teljesen meghiúsítja. Szükség van tehát egy olyan algoritmusra, amellyel $\mathcal{B}$ *vakon* tudja aláírni m-et, azaz később ne tudja eldöneni, hogy egy adott (c, r) párt a protokoll melyik végrehajtásánál szolgáltatta.

Az ehhez megfelelő módozatot Schnorr publikálta (5)-ben.

1. $\mathcal{B}$ generál egy véletlen $w \in \mathbb{Z}_q$-t, majd közli $\mathcal{U}$-val $a = g^w$-t.

2. $\mathcal{U}$ véletlenül választja az $u, v \in \mathbb{Z}_q$ számokat. a'-t $a' = ag^v h^{-u}$, míg c'-t $c' = \mathcal{H}(m, a')$ definiálja. $\mathcal{U}$ elküldi $\mathcal{B}$-nek $c = c' - v$-t.

3. $\mathcal{B}$ c-re a szokásos $r = w + xc$-et válaszolja.

4. $r' = r + v$ és $\mathcal{U}$ ellenőriz, amennyiben $\mathcal{B}$ helyesen hajtotta végre a számításokat $g^{r'} = a'h^{c'}$-nek kell teljesülnie.

2.5.1. állítás. (c', r') hiteles aláírás m-en.

Bizonyítás. Könnyen látható, hogy:

$$g^{r'} = g^{r+v} = ah^c g^v = ah^{c'} h^{-u} g^v = a'h^{c'},$$

azaz $a' = h^{-c'} g^{r'}$, vagyis $c' = \mathcal{H}(m, h^{-c'} g^{r'})$, tehát (c', r') tényleg m aláírása.
■

2.5.2. állítás. $\mathcal{B}$ egy (m, c', r') hármasból nem tud következtetni, hogy mely alkalommal készült.

Bizonyítás. A protokollok során $\mathcal{B}$ a w_i, a_i, c_i, r_i számokat látja. Bármelyik (c_i, r_i) párhoz pontosan egy (u_i, v_i) pár található, amire $c' = c_i - u_i$ illetve $r' = r_i + v_i$. Mivel $a_i = h^{-c_i} g^{r_i}$ és $w_i = \log_g a$, ezért igaz, hogy bármely (w_i, a_i, c_i, r_i) olyan négyeshez, amit $\mathcal{B}$ a protokoll egy helyes végrehajtása során eltárolhat, létezik egy megfelelő (u_i, v_i) pár. Tehát (m, c', r') nem tartalmaz olyan információt, ami megkülönböztetné az ő generálását a többiétől. ■

A Schnorr-féle aláírással kapcsolatban a következő feltételezéssel élünk:

2.5.3. feltételezés. *Tetszőleges $n > 0$-ra, ha $\mathcal{U}$ (polinomiálisan limitált számítási kapacitással) $\mathcal{B}$-vel n alkalommal végrehajtja a protokollt – és természetesen $\mathcal{B}$ követi az előírásokat –, akkor $\mathcal{U}$ csak elhanyagolhatóan kis valószínűséggel tud legalább $n+1$ különböző* üzenet – helyes aláírás *pár birtokába jutni.*

A feltételezés ezidáig még nem nyert bizonyítást, ám az eddig fölfedezett támadási módszerek sikertelennek bizonyultak a Schnorr-aláírás ellen.

3. fejezet

Protokollok

3.1. Chaum eredeti rendszere

Az elektronikus fizetési rendszer egyik első megvalósítását (4)-ben olvashatjuk, ahol a közimert RSA titkosírás adja az alapokat. Lássuk, hogyan épül föl a rendszer:

Az előkészületi szakaszban a bank generál egy RSA rendszert (ahol $n = pq$, p és q prímek, $de \equiv 1 \pmod{\varphi(n)}$) az (n, e) nyilvános kulccsal. Természetesen p, q és d titokban marad. Gyakorlati okokból, mivel általában a felhaszálók számítási kapacitására vannak erősebb felső korlátok, érdemes e-t kicsinek választani, legyen például 3, ekkor d-t $\frac{1}{3}$-dal jelöljük. A bank választ még egy $f : \mathbb{Z}_n \to \mathbb{Z}_n$ -fel jelölt egyirányú (hash) függvényt is.

Az egységnyi értékű érme egy, a bank által aláírt, $(m, f(m)^{\frac{1}{3}})$ pár lesz, ahol $m \in \mathbb{Z}_n$. A felhasználó (a továbbiakban $\mathcal{U}$) tehát választ egy tetszőleges m-et, elküldi a banknak (akit $\mathcal{B}$-vel jelölünk), aki kiszámítja $f(m)^{\frac{1}{3}}$-t, amiről bárki könnyedén ellenőrizheti, hogy ${f(m)^{\frac{1}{3}}}^3 = f(m)$, azaz a bank valóban hitelesítette a pénzt. A hamisítás lehetősége kizárt, hiszen f egyirányú függvény, azaz nem lehet visszafelé, $f(m)^{\frac{1}{3}}$-t választva először, hozzá megfelelő m-et generálni.

Azonban ez az ötlet segít abban, hogy $\mathcal{U}$ anonimitását megőrizzük a fizetési protokoll folyamán: $\mathcal{U}$ nem $f(m)$-et küldi el aláírásra a banknak, hanem $r^3 f(m)$-et, ahol $r \in \mathbb{Z}_n$. A kapott $rf(m)^{\frac{1}{3}}$-ból pedig könnyen számíthatja $f(m)^{\frac{1}{3}}$-t. Így viszont fölmerül a probléma, hogy $\mathcal{U}$ a kapott érmét újra elköltheti (már amennyiben nem akarunk on-line ellenőrzést a fizetési protokoll közben), tehát valahogyan el kell kódolni $\mathcal{U}$ személyazonosságát az érmékben, úgy, hogy az csak a második elköltésnél derüljön ki.

Ezt a következőképpen oldhatjuk meg:

Először $\mathcal{B}$ választ egy k biztonsági paramétert, illetve f mellett egy másik, hasonlóan egyirányú függvényt, g-t. $\mathcal{U}$ azonosítóját a banknál u-val, a neki generált érme sorszámát pedig v-vel fogjuk jelölni, ezeket $\mathcal{B}$ eltárolja a protokoll során.

1. $\mathcal{U}$ véletlenül kiválasztja $\mathbb{Z}_n$-ből az a_i, c_i, d_i és r_i értékeket, $1 \leq i \leq k$-ra. Ezekből képzi az

$$x_i = g(a_i, c_i) \tag{3.1}$$

$$y_i = g(a_i \oplus (u \parallel (v+i)), d_i) \tag{3.2}$$

számokat, ahol $\oplus$ a bitenkénti összeadás (XOR) művelete, $\parallel$ pedig a konkatenációé. Majd elküldi a

$$B_i = r_i^3 f(x_i, y_i) \tag{3.3}$$

sorozatot a banknak.

2. $\mathcal{B}$ kiválaszt az $\{1, \ldots, k\}$ halmazból $\frac{k}{2}$ elemet, amiket közöl $\mathcal{U}$-val. Az egyszerűség kedvéért tegyük föl, hogy $\mathcal{B}$ a $\{\frac{k}{2}+1, \ldots, k\}$ számokat választotta.

3. $\mathcal{U}$ elküldi a kapott indexhalmaznak megfelelő a_i, c_i, d_i, r_i értékeket $\mathcal{B}$-nek.

4. $\mathcal{B}$ ellenőrzi, hogy az általa kért négyesek teljesítik-e az (3.1), (3.2) és (3.3) egyenleteket, megjegyezzük, hogy $\mathcal{B}$ birtokában van u-nak és v-nek. Amennyiben az ellenőrzés sikeres, úgy $\mathcal{B}$ föltételezi, hogy a meg nem vizsgált B_i-k is helyesen lettek generálva.

5. $1 \leq i \leq \frac{k}{2}$-re $\mathcal{B}$ kiszámítja $B_i^{\frac{1}{3}}$-t és elküldi a

$$B \equiv \prod_{1 \leq i \leq \frac{k}{2}} B_i^{\frac{1}{3}} \pmod n$$

számot $\mathcal{U}$-nak, valamint megnöveli v-t k-val.

6. Az $\mathcal{U}$ által a későbbiekben felhasználható érme

$$C \equiv B \prod_{1 \leq i \leq \frac{k}{2}} r_i \equiv \prod_{1 \leq i \leq \frac{k}{2}} f(x_i, y_i)^{\frac{1}{3}} \tag{3.4}$$

lesz, azzal a változtatással, hogy $\mathcal{U}$ átindexeli a változóit, úgy, hogy az $f(x_i, y_i)$ értékek monoton növő sorozatot alkossanak. Természetesen ő is megnöveli a számlálóját – v-t – k-val.

A fizetési protokoll a következőképpen alakul:

1. $\mathcal{U}$ elküldi C-t a kereskedőnek – $\mathcal{S}$-nek.

2. $\mathcal{S}$ véletlenszerűen kiválasztja a $z_i \in \{0, 1\}$ számokat, $1 \leq i \leq \frac{k}{2}$-re, és elküldi őket $\mathcal{U}$-nak.

3. (a) Amennyiben $z_i = 0$, $\mathcal{U}$ elárulja a_i-t és c_i-t, valamint y_i-t,

   (b) míg $z_i = 1$-re $a_i \oplus (u \parallel (v + i))$, d_i, illetve x_i lesz $\mathcal{U}$ válasza.

4. $\mathcal{S}$ ellenőrzi, hogy teljesülnek-e az (3.1), (3.2) valamint (3.4) egyenlőségek. Pozitív eredmény esetén elfogadja $\mathcal{U}$ pénzérméjét. A kapott C érme mellett eltárolja a protokollban szereplő z_i értékeket, valamint $\mathcal{U}$ előző lépésbeli válaszát.

Egy bizonyos, előre meghatározott időpontban $\mathcal{S}$ visszaküldi az összes addig begyűjtött C érmét $\mathcal{B}$-nek a hozzájuk tartozó tárolt segédinformációkkal együtt. $\mathcal{B}$ a szokásos ellenőrzési lépések végrehajtása után jóváírja $\mathcal{S}$ számláján a megfelelő összeget.

3.1.1. lemma. *$\mathcal{B}$ és $\mathcal{S}$ együttes erővel sem tudják párosítani a pénzfelvételi és a fizetési protokollokat úgy, hogy a használt érmék (és ezáltal a felhasználók) megegyezzenek. $\mathcal{U}$ anonimitása garantált amennyiben követte a protokoll szabályait.*

Bizonyítás. A $\mathcal{B}$ által megfigyelt a_i, c_i, d_i, r_i számok nem szerepelnek később a protokollban. Ezeken kívül $\mathcal{B}$ még $B_i = r_i^3 f(x_i, y_i)$-t ismeri $1 \leq i \leq \frac{k}{2}$-re, azonban az érmében szereplő $f(x_i, y_i)$-k csak elhanyagolható valószínűséggel nem invertálhatók (ha f megfelelően van megválasztva), így minden n-nel relatív B_i-re létezik pontosan egy r_i, amivel $B_i = r_i^3 f(x_i, y_i)$ teljesül. Tehát $\mathcal{B}$ nem tud segíteni, hogy egy fizetésnél $\mathcal{U}$-t felismerjék. $\mathcal{S}$-nek pedig a meglévő információk g egyirányúsága miatt nem elegendőek ahhoz, hogy x_i-ből a_i-t, s így $a_i \oplus a_i \oplus (u \parallel (v+i)) = u \parallel (v+i)$-t kiszámítsa. ■

A felhasználó anonimitásánál is fontosabb kérdés a rendszer *biztonsága*, azaz $\mathcal{U}$ ne tudjon úgy vásárolni, hogy nem fizeti ki érte a kívánt összeget. Ezt $\mathcal{U}$ kétféleképpen valósíthatná meg: pénzhamisítással illetve a meglévő érme két- esetleg többszörös elköltésével. Így az alábbi biztonsági kritériumok teljsítését követeljük meg a rendszertől:

- $\mathcal{U}$ (esetelegesen más felhasználókkal vagy boltokkal együttműködve) ne tudjon $\mathcal{B}$ segítsége nélkül érvényes érmét készíteni.

- $\mathcal{B}$ vegye észre, ha $\mathcal{U}$ ugyanazt az érmét többször is elkölti és ekkor kinyomozhassa $\mathcal{U}$ azonosítóját, még akkor is, ha ez lehetetlen lenne, ha $\mathcal{U}$ csak egyszer használja föl az érmét.

Ennek tükrében:

3.1.2. lemma. *Az érméket nem lehet hamisítani.*

Bizonyítás. Az aláírt érme $(m, f(m)^{\frac{1}{3}})$ alakú. f egyirányúsága miatt egy ilyen generálásához először m-et kell rögzíteni, azaz rögzített $f(m)$-re kellene kiszámolni $f(m)^{\frac{1}{3}}$ értékét, ami pedig ellentmond annak a feltételezésünknek, miszerint az RSA rendszer biztonságos. ■

3.1.3. lemma. *Ha $\mathcal{U}$ az érme készítésénél a k darab B_i szám ϵ-ad részét rosszul adja meg a banknak (azaz nem kódoja bele személyazonosságát), ezt a bank $1 - e^{ck\epsilon}$ valószínűséggel észleli valamilyen c pozitív konstanssal.*

Bizonyítás. A lemma bizonyítása elemi számolás, melynek részletezésétől eltekintünk. ■

3.1.4. lemma. *Ha $\mathcal{S}$ ugyanazt az érmét kétszer költi el, akkor nagy valószínűséggel tette bizonyítható.*

Bizonyítás. Amikor a boltok a kapott érméket visszaküldik a banknak, $\mathcal{B}$ ellenőrzi, hogy nem kapott-e olyan érmét, amit egyszer már elköltöttek valahol. Amennyiben ez az eset állna fönn – föltéve, hogy $\mathcal{S}_1$ és $\mathcal{S}_2$ követték a fizetési protokoll szabályait –, a két alkalommal használt z_1 és z_2 bináris sorozatok $1 - \frac{1}{2^{\frac{k}{2}}}$ valószínűséggel legalább egy j indexű helyen megegyeznek. Ez azt jelenti, hogy erre a j-re az egyik fizetésnél a_j, a másodiknál $a_j \oplus (u \parallel (v + j))$ került $\mathcal{B}$ birtokába, amikből könnyen kiszámíthatja $u \parallel (v + j) = a_j \oplus a_j \oplus (u \parallel (v + j))$-t. Amennyiben a z_i-k megegyeznének, a bolt(ok) hibájának gyanúja merül föl, például, hogy $\mathcal{S}_2$ $\mathcal{U}$-val összejátszva lemásolta az $\mathcal{S}_1$ és $\mathcal{U}$ közti fizetést, ekkor $\mathcal{B}$ nem tudja eldönteni, melyik boltos mond igazat. Ezt megelőzheti az a változtatás, ha a boltok a z-t nem teljesen véletlenül választják, hanem úgy, hogy valamilyen módon tartalmazza az azonosítóikat. Például lehet $z = \mathcal{H}(S, spec, rand)$, ahol $\mathcal{H}$ egy alkalmas hash-függvény, S $\mathcal{S}$ azonosítója, *rand* egy véletlenül választott érték, *spec*-ben pedig esetlegesen egyéb adatokat lehet eltárolni, mint mondjuk a fizetés időpontja. ■

A rendszer jelenlegi formájában azonban még $\mathcal{B}$-nek lehetősége nyílik, hogy $\mathcal{U}$ nevében generáljon érméket. Szükséges tehát, hogy egy érméről $\mathcal{U}$ bizonyíthassa, hogy tényleg az ő közreműködésével készült.

Ennek biztosítására $\mathcal{U}$-nak rendelkeznie kell egy külső, megbízható szervezet által garantált digitális aláírással. Az érmék generálása a következőképpen módosul: a folyamat első lépésében $\mathcal{U}$ az a_i, c_i, d_i, r_i számok mellett még a véletlen z_i, z'_i-ket is megválasztja. A pénzérmébe $\mathcal{U}$ az eddigi u helyett (y_i definiálásánál) $u_i = (u \parallel z_i \parallel z'_i)$-t kódolja bele. A későbbiekben a B_i értékek mellett $\mathcal{U}$ elküldi még $\mathcal{B}$-nek $F = g(z_1, z'_1) \parallel g(z_2, z'_2) \parallel \ldots \parallel g(z_k, z'_k)$-t, amit hitelesít a digitális aláírásával. Az ellenőrzés folyamán $\mathcal{B}$ a kért i indexekre z_i-t és z'_i is megkapja, így megbizonyosodhat róla, hogy F tényleg a megfelelő adatok felhasználásával készült.

$\mathcal{B}$ akkor állíthatja bizonyosan, hogy $\mathcal{U}$ ugyanazt az érmét többször költötte el, ha legalább $\frac{k}{2} + 1$ darab z_i-t és z'_i-t ismer.

A rendszer legfőbb hibája a „cut-and-choose" technika használata, a protokollok során a résztvevő feleknek túl sok információt kell kicserélniük, az érme „túl nagy" ahhoz, hogy praktikus legyen.

3.2. Brands változata

3.2.1. Az első próbálkozás

Az alábbiakban egy olyan rendszert vázolunk illetve vizsgálunk meg közelebbről, amely többszempontból is újdonságot jelent: a szóban forgó pénzmennyiséget nem egységnyi értékű érmék alkotják, hanem egy számláló állása jelöli a felhasználó rendelkezésére álló összeget, valamint a kártyán található egy olyan – bontásvédett – rész (megfigyelő), amely nem esik a birtokos ellenőrzése alá. Utóbbi jelenléte már a számláló miatt is szükségszerű, hiszen nélküle a felhasználó szabadon változtathatná a kártyán lévő pénz mennyiségét. Továbbá sikerül megszabadulni az előző rendszer *cut-and-choose* technikájától, ezt arra használtuk, hogy meggyőződjünk róla, hogy a felhasználó személyazonossága ténylegesen bele van írva az érmébe. Itt valami olyasmi történik, hogy a felhasználó érméjét egy egyenesnek képzelhetjük a síkon – az egyenes meredeksége lesz a felhasználó azonosítója –, erről az egyenesről fog a bolt egy pontot kérdezni. Azt, hogy a pont eleme-e az egyenesnek, könnyedén ellenőrizhető, míg egy pont ismeretéből nem következik az egyenes meredeksége. Amennyiben azonban kétszer költi el az érmét a felhasználó, a két ismert pontból a személyazonosság kiszámítható. Látni fogjuk majd, hogy nem lesz szükség a *cut-and-choose*-ra, az felhasználó csak olyan érmét tud készíteni, amelyben el van tárolva az azonosítója.

Az eddigiekhez képest a protokollok is lényeges változásokon mennek keresztül: a pénzfelvételi eljárás két részre bomlik, itt is található egy – a pénzfelvételihez hasonló – protokoll, amely során a felhasználó a banktól egy egyedi adatsorozatot kap, ezeket az érmékhez hasonlóan csak egyszer szabad fölhasználni, fontos különbség azonban, hogy itt az egyes aláírásokhoz nem tartozik konkrét pénzmennyiség. Ennek következtében szükség van egy metódusra, amely során a bank utasítja a megfigyelőt, hogy növelje a számláló állását. A fizetésbe be kell vonni a megfigyelőt, ami itt is a számláló pontos változtatásáért felelős. A boltos számlájára való jóváírás gyakorlatilag nem változik: a fizetéskor megismert adatok a bankhoz kerülnek, aki az ellenőrzés sikeressége esetén átutalja a megfelelő összeget.

A felhasználó anonimitása védelmével kapcsolatban is új problémák merülnek föl: a megfigyelő, amit valójában a bank képviselőjének tekinthetünk, részt vesz mind a pénzkivételi, mind a fizetési protokollban, így óvakodni kell, nehogy kompromittálhassa a felhasználót. Biztonsági szempontból két folyamatot kell figyelemmel kisérni: a *be-* illetve *kiáramlást*, amelyek a protokollok során a megfigyelő által megfigyelt illetve kiadott olyan adatokat jelentik, amelyek segítségével a felhasználó tranzakcói összeköthetőekké válnak. Ezért az ehhez hasonló rendszerek egyik lényeges vonása, hogy nem szabad a megfigyelőnek – a továbbiakban $\mathcal{O}$-val jelöljük – közvetlenül kommunikálnia a külvilággal, az összes $\mathcal{O}$ által érintett adatcsere a felhasználón keresztül kell, hogy megtörténjen, így védve az illetéktelen információátadást. A két probléma közül a beáramlást tekinthetjük a kevésbé veszélyesnek, hiszen a megfigyelő tudomására jutott adatokat addig nem lehet felhasználni, amíg az a felhasználó birtokában van; tehát csak a kártya elvesztése, ellopása esetén sérülhet a vásárló inkognitója, illetve természetesen, ha a bank a érvényességi idő letelte után megköveteli a lejárt kártya visszaszolgáltatását.

A rendszerben használt számok ismét egy olyan G_q csoport elemei, amelyben a DLP teljesül. A számok összeszorzása pedig a csoportművelet végrehajtását jelenti.

Inicializálásképpen $\mathcal{B}$ kiválaszt véletlenül két elemet – x-et és y-t – $\mathbb{Z}_n$-ből, ezek a titkos kulcsai lesznek; illetve egy nyilvános kulcsot – g_0-t – G_q-ból, természetesen úgy, hogy g_0 ne legyen G_q egységeleme. Az x-hez és y-hoz tartozó nyilvános kulcsok másik fele $h = g_0^x$ és $g_1 = g_0^y$ lesz. Továbbá rögzítésre kerül egy hash függvény is, amelyet $\mathcal{H}$-val jelölünk.

Amikor egy fölhasználó – $\mathcal{U}$ – számlát nyit a banknál, $\mathcal{B}$ generál egy titkosan kezelt $x_u \in \mathbb{Z}_q$ számot, melyet az $\mathcal{U}$ rendelkezésére bocsátott $\mathcal{O}$-ban eltárol. $\mathcal{U}$ bankbeli azonosítóját $I_u \equiv x + yx_u \pmod q$ jelöli. Az I_u értéket a bank $\mathcal{U}$ való életbeli azonosítójával és számlája állásával együtt eltárolja. $\mathcal{O}$ x_u mellett g_1-et ismeri, természetesen a számlálót, amit C_u-val jelölünk, ez jelképezi $\mathcal{U}$ pénzét, valamint egy másik számlálót, T_u-t, ami a pénzkivételek számát méri ($\mathcal{B}$ ezt is tárolja). $\mathcal{U}$ megkapja $h_u = g_1^{x_u}$-t, a (g_0, g_1, h) hármassal egyetemben.

Bizonyos esetekben $\mathcal{U}$-nak szüksége van pénzre, azaz szeretné, ha a kártyán lévő számláló értéke megnövelődne, hiszen különben nem tud vásárolni. Ekkor kapcsolatba lép a bankkal és a következő protokoll játszódik le:

1. $\mathcal{U}$ közli $\mathcal{O}$-val azt a t összeget, amit ki szeretne venni a bankból.
2. $\mathcal{O}$ kiszámítja $v = \mathcal{H}(x_u, T_u, t)$-t, és $v' = \mathcal{H}(v)$-t, utóbbit megmondja $\mathcal{U}$-nak.
3. $\mathcal{U}$ fölveszi a kapcsolatot $\mathcal{B}$-vel, azonosítja magát, és bejelenti azt a t összeget, amit ki szeretne venni a bankból.
4. $\mathcal{B}$ kiszámítja $v = \mathcal{H}(x_u, T_u, t)$-t és közli $\mathcal{U}$-val.
5. $\mathcal{U}$ ellenőrzi, hogy $v' = \mathcal{H}(v)$, és átadja v-t $\mathcal{O}$-nak.
6. Amennyiben $\mathcal{U}$ a helyes v értéket küldte, $\mathcal{O}$ hozzáad C_u-hoz t-t, T_u-hoz pedig egyet.
7. $\mathcal{B}$ is elvégzi a szükséges számlálómódosításokat.

Megjegyzésre érdemes, hogy a protokoll során csak klasszikus kriptográfiai eszközöket alkalmaztunk, nyilvános kulcsú módszerek alkalmazására nem is volt szükség, hiszen $\mathcal{O}$-nak és $\mathcal{B}$-nek volt egy közös titkos kulcsa (x_u), aminek segítségével tudtak megbízhatóan kommunikálni. A protokoll kezdő és befejező lépései tulajdnképpen off-line módon futhatnak, nincs szükség kapcsolatra $\mathcal{B}$ és $\mathcal{U}$ között, így összességében nagyon kevés on-line adatforgalomra van szükség. $\mathcal{O}$ és $\mathcal{B}$ szerepét értelemszerűen felcserélve a számáló csökkentésére is lehetőség nyílik, így a kártyáról be is lehet tenni pénzt a bankba, nem csak kivenni.

3.2.1. állítás. Az $\mathcal{O}$ által tárolt T_u érték nem lehet magasabb a $\mathcal{B}$ által tároltnál, ha követték a protokoll szabályait.

Bizonyítás. Nyilvánvaló, hiszen $\mathcal{U}$ nem tudja x_u-t, azaz nem képes a $\mathcal{H}(x_u, x, y)$ függvény értékét kiszámítani egyetlen (x, y) pontban sem. $\mathcal{H}$ hash függvény, ezért x_u-ra sem tud következtetni v-ből, valamint T_u használata megakadályozza, hogy egy régebben megismert v értéket újra felhasználva kétszer növeltethesse $\mathcal{O}$-val C_u-t. (Természetesen az lehetséges, hogy $\mathcal{U}$ csak a protokoll $\mathcal{B}$-vel zajló szakaszát hajtja végre, ekkor sikerül úgy pénzt kivennie a bankból, hogy nem kapja meg. Ebben az esetben ráadásul még a T_u számlálók is elcsúsznak és $\mathcal{U}$ nem tud több pénzt kivenni.)

3.2.2. állítás. A protokoll sikeresen kiszűri a $\mathcal{B}$-től $\mathcal{O}$-ba áramló nemkívánatos információt.

Bizonyítás. $\mathcal{B}$-től csak a v szám jut el $\mathcal{O}$-ba, azonban az első két lépés során $\mathcal{U}$ megbizonyosodik róla, hogy $\mathcal{O}$ ezt a rendelkezésére álló adatok alapján egyedül is ki tudta számítani. (A protokoll alatt $\mathcal{O}$-ból nem jut ki semmilyen információ, így a kiáramlás nem okoz problémát.)

A fizetéshez szükség van még „fizetési engedélyek" beszerzésére a banktól, ami tulajdonképpen a bank által hitelesített tikos-nyilvános kulcspár lesz azzal a tulajdonsággal, hogy a használatához szükséges információ egy részét $\mathcal{O}$ fogja tárolni, így $\mathcal{O}$-t föl kell törni ahhoz, hogy kétszer lehessen ugyanazt az összeget elkölteni. Azonban még ha ez be is következne, $\mathcal{U}$ azonosítható a tett elkövetése után.

Lássuk tehát a protokollt:

1. $\mathcal{O}$ generál egy véletlen $w_u \in \mathbb{Z}_q$ számot, tárolja, és az $a_u = g_1^{w_u}$-t elküldi $\mathcal{U}$-nak.

2. $\mathcal{U}$ kiválasztja az $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$ véletlen elemeket $\mathbb{Z}_q$-ből, majd kiszámítja a $h'_u = h_u g_0^{\alpha_1}$, $a'_u = a_u g_1^{\alpha_2} g_0^{\alpha_3}$ értékeket. A h'_u, a'_u, $\alpha_{1\ldots3}$ számok hosszú távon tárolódnak, használatukra a fizetés során is sor kerül majd, α_4 és α_5 csak ebben a protokollban kerül felhasználásra.

3. $\mathcal{U}$ kapcsolatba lép $\mathcal{B}$-vel, azonosítja magát

4. $\mathcal{B}$ meghatároz egy véletlen $w \in \mathbb{Z}_q$-t, és $a = g_0^w$-t elküldi $\mathcal{U}$-nak.

5. $\mathcal{U}$ kiszámítja $c' = \mathcal{H}(h'_u, a'_u, a g_0^{\alpha_4}(hh_u)^{\alpha_5})$-et, tárolja, és $c \equiv c' + \alpha_5 \pmod q$-öt eljuttatja $\mathcal{B}$-nek.

6. $\mathcal{B}$ válasza $r \equiv cI_u + w \pmod q$ lesz.

7. $\mathcal{U}$ ellenőrzi, hogy $g_0^r(hh_u)^{-c} = a$ fennáll-e, amennyiben igen, eltárolja $r' \equiv r + c'\alpha_1 + \alpha_4 \pmod q$-t.

Természetesen a protokoll elején és végén itt sincs szükség on-line kapcsolatra $\mathcal{B}$ és $\mathcal{U}$ között. $\mathcal{B}$ és $\mathcal{O}$ között nincs kapcsolat, így információáramlás sem léphet föl.

A Schnorr-féle aláíráshoz hasonlóan itt is belátható, hogy a fizetési protokollban ténylegesen használt „vesszőzött" értékekről $\mathcal{B}$ nem szerez információt.

A fizetés a következő forgatókönyv szerint zajlik:

1. *spec* egy olyan sorozatot jelöl, amiben a tranzakció bizonyos adatai tárolódnak, például a számla összege, az idő, valamint $\mathcal{S}$ egyedi azonosítója $\mathcal{B}$-nél. $\mathcal{U}$ elküldi a h'_u, a'_u számokat, valamint *spec*-et $\mathcal{O}$-nak.

2. $\mathcal{O}$ ellenőrzi, hogy a kifizetendő pénzre van-e elég keret, valamint, hogy tárolja-e a w_u értéket. Amennyiben a feltételek teljesülnek, kiszámolja $d = \mathcal{H}(h'_u, a'_u, spec)$-t és $r_{u1} \equiv dx_u + w_u \pmod q$-t. Elküldi r_{u1}-et $\mathcal{U}$-nak, majd csökkenti C_u-t a megfelelő összeggel és törli a memóriából w_u-t.

3. $\mathcal{U}$ is kiszámolja $d = \mathcal{H}(h'_u, a'_u, spec)$-t, ellenőrzi, hogy tényleg igaz-e $g_1^{r_{u1}}(h'_u)^{-d} = a_u$. $r'_{u1} \equiv r_{u1} + \alpha_2 \pmod q$ lesz, illetve $r_{u2} \equiv d\alpha_1 + \alpha_3 \pmod q$.

4. $\mathcal{U}$ elküldi a h'_u, c', r', a'_u, r'_{u1}, r_{u2} adatokat $\mathcal{S}$-nek.

5. $\mathcal{S}$ kiszámítja $d = \mathcal{H}(h'_u, a'_u, spec)$-t, és ellenőrzi, hogy teljesül-e $c' = \mathcal{H}(h'_u, a'_u, g_0^{r'}(hh'_u)^{-c'})$ és $g_1^{r'_{u1}} g_0^{r_{u2}} (h'_u)^{-d} = a'_u$. Ha az egyenlőségek igazak, elfogadja a fizetést.

Befejezésképpen $\mathcal{S}$ jóváíratja $\mathcal{B}$-nél a beérkezett pénzt, ezt szokás szerint teheti valamilyen rendszerességgel, egy csomagban elküldve a még elintézetlen vásárlások anyagát.

1. $\mathcal{S}$ elküldi $\mathcal{B}$-nek a fizetéskor használt $(h'_u, c', r', a'_u, r'_{u1}, r_{u2}, spec)$ hetest.

2. $\mathcal{B}$ ellenőrzi, hogy szerepel-e az adatbázisában a (h'_u, c', r') hármas:

   (a) Nem: ezt az igazolást még nem használták; $\mathcal{B}$ is megvizsgálja az $\mathcal{S}$ által is ellenőrzött egyenlőségeket, ha teljesülnek, akkor elfogadja $\mathcal{S}$ adatait, jóváírja a számláján a megfelelő összeget és eltárolja a kapott információt.

   (b) Igen: ez azt jelenti, hogy ugyanezzel a kulccsal már fizettek egyszer. Ekkor két lehetőség adódik:

      i. *spec* értéke is megegyezik a két esetben. Ekkor $\mathcal{B}$ $\mathcal{S}$-et teszi felelőssé a problémáért, hiszen a *spec*-ek egyenlőségéből adódóan mindkétszer $\mathcal{S}$ fogadta el az érmét, ráadásul a *spec*-be nyomtatot időpecsét hibakorlátján belül. $\mathcal{B}$ természetesen nem utalja át $\mathcal{S}$ számlájára a fizetés összegét.

      ii. Ha *spec* értéke különböző a bizonyítvány két felhaszálásakor ($spec_1 \neq spec_2$), akkor – nagy valószínűséggel – $d_1 = \mathcal{H}(h'_u, a'_u, spec_1)$ és $d_2 = \mathcal{H}(h'_u, a'_u, spec_2)$ értéke is különböző lesz, így

$$
\begin{aligned}
r'_{u1_1} &\equiv d_1 x_u + w_u + \alpha_2 \pmod{q} \\
r'_{u1_2} &\equiv d_2 x_u + w_u + \alpha_2 \pmod{q}
\end{aligned}
$$

miatt

$$x_u \equiv \frac{r_{u1_1} - r_{u1_2}}{d_1 - d_2} \pmod q,$$

azaz

$$I_u \equiv x + y x_u \pmod q,$$

vagyis $\mathcal{U}$ azonosítója kiszámítható a két tranzakció adataiból.

Sajnos a protokoll a kitűzött céljaink egyikét nem valósítja meg, nevezetesen $\mathcal{O}$ fizetéskor megkapja – többek közt – *spec*-et, így, ha valakinek $\mathcal{O}$ a birtokába jut és ki tudja nyerni belőle az esetlegesen eltárolt adatokat, akkor máris tudomást szerezhet $\mathcal{U}$ vásárlásairól.

A rendszer ezen hibáján azon az áron javíthatunk, hogy feladjuk a számláló kényelmes használatát és áttérünk az elektronikus érmékre. Ez azt jelenti, hogy egy-egy $\mathcal{B}$ által hitelesített kulcspár nem csupán a fizetés érvényességének igazolására szolgál, hanem maga hordozza az értéket, minden egyes $(h'_u, c', r', a'_u, r'_{u1}, r_{u2})$ hatos előre rögzített értéket képvisel.

Ez a következőkben változtatja meg protokolljainkat. Először is nyilván feleslegessé válik az $\mathcal{O}$-ban lévő T_u számláló, ezáltal maga az a módszer is, amellyel T_u értékét lehet növelni. Másodszor pedig: immár – mivel minden érme illetve igazolás ugyanannyit ér – nem szükséges, hogy $\mathcal{O}$ tudomást szerezzen a fizetés részleteiről, anélkül is hitelesítheti az érmét. Így tehát:

Érmegenerálás:

1. $\mathcal{O}$ kitalál egy véletlen $w_u \in \mathbb{Z}_q$ számot, tárolja, és az $a_u = g_1^{w_u}$-t elküldi $\mathcal{U}$-nak.

2. $\mathcal{U}$ kiválasztja az $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6)$ véletlen elemeket $\mathbb{Z}_q$-ből, majd kiszámítja a $h'_u = h_u g_0^{\alpha_1}$, $a'_u = a_u g_1^{\alpha_2} g_0^{\alpha_3} h_u^{\alpha_6}$ értékeket. h'_u, a'_u, $\alpha_{1 \ldots 3}$ mellett most az új α_6 számot is megjegyzi $\mathcal{U}$ későbbi használatra. α_4

és α_5 továbbra is csak ebben a protokollban kerül felhasználásra. A számlálós változathoz képest itt a'_u definiálásánál lép be egy $h_u^{\alpha_6}$-os szorzótényező.

3. $\mathcal{U}$ kapcsolatba lép $\mathcal{B}$-vel, azonosítja magát

4. $\mathcal{B}$ meghatároz egy véletlen $w \in \mathbb{Z}_q$-t, és $a = g_0^w$-t elküldi $\mathcal{U}$-nak.

5. $\mathcal{U}$ kiszámítja $c' = \mathcal{H}(h'_u, a'_u, ag_0^{\alpha_4}(hh_u)^{\alpha_5})$-et, tárolja, és $c \equiv c' + \alpha_5 \pmod q$-t eljuttatja $\mathcal{B}$-nek.

6. $\mathcal{B}$ válasza $r \equiv cI_u + w \pmod q$ lesz.

7. $\mathcal{U}$ ellenőrzi, hogy $g_0^r(hh_u)^{-c} = a$ fennáll-e, amennyiben igen, eltárolja $r' \equiv r + c'\alpha_1 + \alpha_4 \pmod q$-t.

Illetve a fizetés:

1. *spec* itt is a tranzakció tulajdonságait írja le: $\mathcal{S}$ azonosítója $\mathcal{B}$-nél, egy egyedi szám, mely megkülönbözteti a többi $\mathcal{S}$-et érintő kifizetéstől (mint például az időpecsét), stb. $\mathcal{U}$ kiszámolja $d = \mathcal{H}(h'_u, a'_u, spec)$-t, és most csak ennek az eltranszformált változatát $d' \equiv d + \alpha_6 \pmod q$-t küldi el $\mathcal{O}$-nak, $\mathcal{H}$ argumentumait nem.

2. $\mathcal{O}$ ellenőrzi, hogy tárolja-e a w_u értéket. Amennyiben igen, kiszámolja $r_{u1}^+ \equiv d'x_u + w_u \pmod q$-t, elküldi $\mathcal{U}$-nak, majd törli a memóriából w_u-t.

3. $\mathcal{U}$ ellenőrzi, hogy tényleg igaz-e $g_1^{r_{u1}^+}(h'_u)^{-d'} = a_u$. $r'_{u1} \equiv r_{u1}^+ + \alpha_2 \pmod q$ lesz, illetve $r_{u2} \equiv d\alpha_1 + \alpha_3 \pmod q$.

4. $\mathcal{U}$ elküldi a h'_u, c', r', a'_u, r'_{u1}, r_{u2} adatokat $\mathcal{S}$-nek.

5. $\mathcal{S}$ kiszámítja $d = \mathcal{H}(h'_u, a'_u, spec)$-t, és ellenőrzi, hogy teljesül-e $c' = \mathcal{H}(h'_u, a'_u, g_0^{r'}(hh'_u)^{-c'})$ és $g_1^{r'_{u1}} g_0^{r_{u2}} (h'_u)^{-d} = a'_u$. Ha az egyenlőségek igazak, elfogadja a fizetést.

A jóváírás a boltos számlájára nem változik.

Az előbbiekhez hasonló módon itt is csak akkor lehet kétszer fizetni egy igazolással, ha $\mathcal{O}$ védelmét $\mathcal{U}$-nak sikerül föltörnie, azonban ugyanúgy kiszámítható a személyazonossága, mint a számlálós változatban.

3.2.3. állítás. Ha $\mathcal{U}$ és $\mathcal{S}$ követi a protokollok szabályait illetve $\mathcal{U}$ elfogadta $\mathcal{B}$ és $\mathcal{O}$ válaszait az érmegenerálás és a hitelesítés folyamán, akkor $\mathcal{S}$ elfogadja az érmét függetlenül attól, hogy $\mathcal{B}$ és $\mathcal{O}$ esetleg nem követte az előírásokat pontosan.

Bizonyítás. $\mathcal{S}$ pontosan akkor fogadja el $\mathcal{U}$ fizetését, ha

$$c' = \mathcal{H}(h'_u, a'_u, g_0^{r'}(hh'_i)^{-c'}),$$

valamint

$$a'_u = g_1^{r'_{u1}} g_0^{r_{u2}} (h'_u)^{-d}.$$

Fölhasználva az

$$\begin{aligned} a'_u &= a_u g_1^{\alpha_2} g_0^{\alpha_3} h_u^{\alpha_6} \\ c' &= \mathcal{H}(h'_u, a'_u, as) \\ s &= g_0^{\alpha_4} (hh_u)^{\alpha_5} \end{aligned}$$

definíciókat

$$\begin{aligned} g_0^{r'}(hh'_u)^{-c'} &= a g_0^{\alpha_4} (hh_u)^{\alpha_5} \\ a_u g_1^{\alpha_2} g_0^{\alpha_3} h_u^{\alpha_6} &= g_1^{r'_{u1}} g_0^{r_{u2}} (h'_u)^{-d} \end{aligned}$$

teljesülése elégséges.

$$
\begin{aligned}
r' &\equiv r + c'\alpha_1 + \alpha_4 \pmod q \\
c' &\equiv c - \alpha_5 \pmod q \\
h' &= h_u g_0^{\alpha_1}
\end{aligned}
$$

felhasználásával az egyszerűsítések elvégzése után

$$
g_0^r (hh_u)^{-c} = a \tag{3.5}
$$

adódik, ez pedig azért teljesül, mert a feltétel szerint az érmegeneráláskor $\mathcal{U}$ elfogadta $\mathcal{B}$ válaszát, ami pedig pont (3.5) ellenőrzését jelentette. Hasonlóan

$$
\begin{aligned}
r_{u2} &\equiv d\alpha_1 + \alpha_3 \pmod q \\
d &\equiv d' - \alpha_6 \pmod q \\
r'_{u1} &\equiv r^+_{u1} + \alpha_2 \pmod q \\
h' &= h_u g_0^{\alpha_1}
\end{aligned}
$$

miatt

$$
a_u = g_1^{r^+_{u1}} (h'_u)^{-d'},
$$

ez pedig ekvivalens azzal, hogy $\mathcal{U}$ elfogadta $\mathcal{O}$ válaszát fizetéskor. ■

Belátható, hogy a számlálós változattal ellentétben itt már $\mathcal{U}$ pénzfelvételét és a fizetést nem lehet kapcsolatba hozni, azaz a vásárlás anonimitása nem sérül. Nevezzük $\mathcal{B}$ által megfigyelhető adatoknak az olyan (a, c, r) hármasokat, amikre $g_0^r (hh_u)^{-c} = a$ teljesül. Hasonlóképpen definiáljuk az $\mathcal{O}$ illetve $\mathcal{S}$ által megfigyelhető információt is (a_u, d', r^+_{u1}) és $(h'_u, c', r', a'_u, r'_{u1}, r_{u2})$ formájában, $\mathcal{O}$ hármasaira $g_1^{r^+_{u1}} h_u^{-d'} = a_u$-nak, $\mathcal{S}$ hatosaira pedig $g_1^{r'_{u1}} g_0^{r_{u2}} (h'_u)^{-d} = a'_u$-nek és $c' = \mathcal{H}(h'_u, a'_u, g_0^{r'} (hh'_u)^{-c'})$-nek kell teljesülnie. Ezek a megkötések szükségesek ahhoz, hogy az említett adatokat a megfelelő személyek a protokoll egy-egy sikeres végrehajtása során rögzíthessék.

3.2.4. állítás. Bármely $((a, c, r), (a_u, d', r_{u1}^+), (h_u', c', r', a_u', r_{u1}', r_{u2}))$ $\mathcal{B}$, $\mathcal{O}$ és $\mathcal{S}$ által megfigyelhető adatokra és $\mathcal{U}$-ra létezik pontosan egy olyan $(\alpha_1, \ldots, \alpha_6)$ hatos, amely felhasználásával $\mathcal{B}$, $\mathcal{O}$ és $\mathcal{S}$ megfigyelései összhangba hozhatók, azaz nem lehet megállapítani, hogy egy adott igazoláshoz melyik fizetés tartozik.

Bizonyítás. Tetszőleges $((a, c, r), (a_u, d', r_{u1}^+), (h_u', c', r', a_u', r_{u1}', r_{u2}), h_u)$ esetén legyen

$$\begin{aligned}
\alpha_1 &= \log_{g_0} \frac{h_u'}{h_u} \\
\alpha_2 &\equiv r_{u1}' - r_{u1}^+ \pmod q \\
\alpha_3 &\equiv r_{u2} - d\alpha_1 \pmod q \\
\alpha_4 &\equiv r' - r - c'\alpha_1 \pmod q \\
\alpha_5 &\equiv c - c' \pmod q \\
\alpha_6 &\equiv d' - d \pmod q.
\end{aligned}$$

Ezek a választások mind a két protokollban teljesítik az összes megkövetelt kapcsolatot a változók között, legtöbbjüket azért, mert az α_i-ket így definiáltuk. Ellenőrizni csak a

$$\begin{aligned}
c' &= \mathcal{H}(h_u', a_u', g_0^{r'}(hh_u')^{-c'}) \\
a_u' &= g_1^{r_{u1}'} g_0^{r_{u2}} (h_u')^{-d}
\end{aligned}$$

egyenleteket kell, ezek teljesülése azonban könnyen belátható a megfigyelhető adatokra vonatkozó feltételeink segítségével. ■

A rendszer biztonságára itt egy új követelményt is bevezetünk a korábban említettek mellett:

- Érmét többször csak abban az esetben lehessen elkölteni, ha $\mathcal{U}$-nak sikerül föltörnie $\mathcal{O}$-t.

Belátható, hogy a eddigi föltételezéseink mellett (DLP, a Schorr-féle aláírás megbízhatósága) a rendszer biztonságos, amennyiben $\mathcal{B}$ nem engedélyezi a két különböző $\mathcal{U}$-nak, hogy egyszerre, párhuzamosan futtassák az érmegenerálási protokollt.

3.2.2. Párhuzamos támadás és az ellene való védekezés

Az előbbiekben említettük, hogy Brands rendszerének biztonságához szükséges, hogy egyszerre csak egy felhasználó vehessen pénzt ki a bankból. Lássuk, hogy az első ránézésre biztonságos rendszert hogyan lehet kijátszani, ha megengedjük a párhuzamos érmegenerálást, a protokollok szereplőit $\mathcal{U}_1$-gyel és $\mathcal{U}_2$-vel jelölve (a megfelelő megfigyelők $\mathcal{O}_1$ és $\mathcal{O}_2$ lesznek). $\mathcal{U}_1$ és $\mathcal{U}_2$ képesek lesznek egy olyan érmét szerezni a banktól, amely többszörös elköltéséből egyikük személyazonosságát sem lehet kiszámítani, bár ehhez szükségük lesz $\mathcal{O}_1$ és $\mathcal{O}_2$ föltörésére.

A támadást lehetőségét csak vázoljuk, azaz a protokollban nem törődünk az $\mathcal{U}_1$ és $\mathcal{U}_2$ által a $\mathcal{B}$-nek átadott adatok védelmével, torzításával. Föltesszük, hogy $\mathcal{O}_1$-t és $\mathcal{O}_2$-t sikerült föltörni, azaz az ő szerepüket is $\mathcal{U}_1$ illetve $\mathcal{U}_2$ fogja játszani:

1. $\mathcal{B}$ a véletlenül választott w_1 és w_2 számokra $a_1 = g_0^{w_1}$-t és $a_2 = g_0^{w_2}$-t küldi $\mathcal{U}_1$-nek illetve $\mathcal{U}_2$-nak.

2. $\mathcal{U}_1$ és $\mathcal{U}_2$ közösen választanak tetszőleges x_u-t, $h'_u = g_1^{x_u}$ lesz. A véletlenül választott w_{u1} és w_{u2} értékekkel $a'_u = g_0^{w_{u1}} g_1^{w_{u2}}$, $c' = \mathcal{H}(h'_u, a'_u, a_1 a_2)$.

3. $\mathcal{U}_1$ kiszámítja

$$c_{u1} \equiv c' \frac{x_{u2} - x_u}{x_{u2} - x_{u1}} \pmod q$$

   értékét, és elküldi $\mathcal{B}$-nek. Hasonlóan tesz $\mathcal{U}_2$ a

$$c_{u2} \equiv c' \frac{x_u - x_{u1}}{x_{u2} - x_{u1}} \pmod q$$

számmal.

4. $\mathcal{B}$ válasza természetesen $r'_{u1} \equiv c_{u1}I_{u1} + w_1 \pmod q$ és $r'_{u2} \equiv c_{u2}I_{u2} + w_2 \pmod q$ lesz.

5. $\mathcal{U}_1$ és $\mathcal{U}_2$ elfogadják $\mathcal{B}$ válaszát, ha $a_1 = g_0^{r'_{u1}}(hh_{u1})^{-c_{u1}}$ és $a_2 = g_0^{r'_{u2}}(hh_{u2})^{-c_{u2}}$. Ekkor $r' \equiv r'_{u1} + r'_{u2} \pmod q$ lesz a megfelelő válasz a közös érméjükhöz.

3.2.5. lemma. *A kapott (h'_u, a'_u, c', r') érme „érvényes".*

Bizonyítás. Könnyen látszik, hogy

$$\begin{aligned} c_{u1} + c_{u2} &\equiv c' \pmod q \\ c_{u1}x_{u1} + c_{u2}x_{u2} &\equiv c'x_u \pmod q. \end{aligned}$$

Ezért

$$a_1a_2 = g_0^{r'_{u1}}(hh_{u1})^{-c_{u1}}g_0^{r'_{u2}}(hh_{u2})^{-c_{u2}} = g_0^{r'}(hh'_u)^{-c'},$$

azaz

$$c' = \mathcal{H}(h'_u, a'_u, g_0^{r'}(hh'_u)^{-c'}).$$

Az $\mathcal{S}$-fel folytatott protokollokban $\mathcal{O}_1$ és $\mathcal{O}_2$ feltörése miatt $\mathcal{U}_1$ és $\mathcal{U}_2$ könnyen érvényesítheti a hamis érmét, az ott használt (r'_{u1}, r_{u2}) párnak $(w_{u2}+x_ud, w_{u1})$ párt megadva teljesül, amit $\mathcal{S}$ ellenőriz:

$$g_1^{w_{u2}+x_ud}g_0^{w_{u1}}h_u^{-d} = a'_u$$

Amikor $\mathcal{B}$ észleli, hogy az érmét többször költötték el, és megpróbálja leleplezni a tettest, a kapott adatokból

$$\frac{r'_{u1_1} - r'_{u1_2}}{d_1 - d_2} \equiv x_u \pmod q$$

segítségével x_u-t tudja meghatározni, így $\mathcal{U}_1$ és $\mathcal{U}_2$ személyazonossága rejtve marad. ■

Ez – és más párhuzamos támadások – ellen nyújt védelmet (6) ötlete, mely fölcseréli g és h szerepét, azaz a bank rögzített nyilvános kulcsa lesz h és a felhasználók egyedi kulcsai lesznek g_u alakúak. Azaz: $\mathcal{B}$ nyilvános kulcsai: (g_0, g_1, h), ahol $g_0 = h^y, g_1 = h^x$. Egy tetszőleges $\mathcal{U}$ felhasználó titkos azonosítószáma x_u, ezt fogja majd $\mathcal{O}$ használni, $\mathcal{B}$ pedig az $I_u = yx_u + x$-et. $\mathcal{U}$ nyilvános kulcsa $g_u = h^{I_u} = g_0^{x_u} g_1$ lesz.

A pénzfelvétel:

1. $\mathcal{O}$ egy véletlen w_u-ra $(\in \mathbb{Z}_q)$ $a_u = g_0^{w_u}$-t közli $\mathcal{U}$-val.

2. $\mathcal{B}$ egy hasonlóan véletlen $w \in \mathbb{Z}_q$-ra $a = g_u^w$-nel teszi ugyanezt.

3. $\mathcal{U}$ vesz egy véletlen $(\alpha_1, \dots, \alpha_6) \in \mathbb{Z}_q^6$ hatost. Új nyilvános kulcsa $g'_u = g_u^{\alpha_5}$-lesz, majd kiszámítja $a' = g_u^{\alpha_3} h^{\alpha_4} a$-t, $a'_u = g_0^{\alpha_1} g_1^{\alpha_2} {g'_u}^{\alpha_6} a_u$-t és $c' = \mathcal{H}(g'_u, a'_u, a')$-t. $c \equiv c' + \alpha_4 \pmod q$-t közli a bankkal.

4. $\mathcal{B}$ válasza $r \equiv \frac{c}{I_u} + w \pmod q$ lesz.

5. $\mathcal{U}$ elfogad, ha $g_u^r h^{-c} = a$. Ebben az esetben $r' \equiv \frac{r+\alpha_3}{\alpha_5} \pmod q$.

A fizetés protokollja pedig a következő:

1. $\mathcal{U}$ kiszámítja $d = \mathcal{H}(spec, g'_u, a'_u)$-t, $d' \equiv t(d+\alpha_6) \pmod q$-t megmondja $\mathcal{O}$-nak.

2. $\mathcal{O}$ kiszámítja $r^+ \equiv d'x_u + w_u$-t, ha w_u-t megtalálja a memóriában és átadja $\mathcal{U}$-nak, majd elfelejti w_u-t.

3. Ha $g_0^{r^+} (\frac{g_u}{g_1})^{-d'} = a_u$, akkor $\mathcal{U}$ elfogadja $\mathcal{O}$ válaszát, kiszámítja $r_0 \equiv r^+ + \alpha_1 \pmod q$-et. $r_1 \equiv td + \alpha_6 + \alpha_2 \pmod q$ lesz. Az $\mathcal{S}$-nek átadott érme a $(g'_u, c', r', r_0, r_1, a'_u)$ hatos.

4. $\mathcal{S}$ is kiszámítja $d = \mathcal{H}(spec, g'_u, a'_u)$-t. Amennyiben

$$c' = \mathcal{H}(g'_u, a'_u, {g'_u}^{r'} h^{-c'})$$

és

$$a'_u = g_0^{r_0} g_1^{r_1} {g'_u}^{-d}$$

teljesül, a fizetés sikeres.

Ha bankhoz visszajuttatott érmék között lenne kettő, amelyekben (g'_u, c', r') megegyezik, akkor $\mathcal{B}$ a szokásos módon

$$\alpha_5 \equiv \frac{r_{1_0} - r_{1_1}}{d_0 - d_1} \pmod q$$

és

$$x_u \alpha_5 \equiv \frac{r_{0_0} - r_{0_1}}{d_0 - d_1} \pmod q$$

segítségével $\mathcal{U}$ azonosítója visszafejthető.

Erről a verzióról is az eddigiekhez hasonló módon beláthatjuk, hogy teljesül rá az eredeti verzió anonimitás-védelme, valamint a biztonsági tulajdonságai is, sőt, még az említett párhuzamos támadás ellen is védelmet nyújt.

Az előző változatban aláírásánál $\mathcal{B}$ válasza $r \equiv cI_u + w \pmod q$ volt, amire g_0-t emelve $I_u \equiv yx_u + x \pmod q$ miatt egy $g_0^{I_u} = hh_u$ tagot kaptunk, az ellenőrzésnél $g_0^r(hh_u)^{-c} = a$-nak kellett teljesülnie. Így amikor összeszoroztuk a két ellenőrző egyenletet,

$$g_0^{r_1}(hh_{u1})^{-c1} g_0^{r_2}(hh_{u2})^{-c2} = a_1 a_2$$

teljesüléséhez a keresett közös c'-nek a

$$h^{-c_1} h^{-c_2} = h^{-c'}$$

és

$$h_{u1}^{-c_1} h_{u2}^{-c_2} = {h'_u}^{-c'}$$

egyenletek mindegyikét ki kellett elégíteni, a két független egyenletből meg tudtuk határozni c_1-et és c_2-t.

Most ugyanezzel a módszerrel $g_u^r h^{-c} = a$ teljesülését várjuk el, így

$$g_{u1}^{r_1} h^{-c1} g_{u2}^{r_2} h^{-c2} = a_1 a_2$$

kell, ami

$$g_{u1}^{r_1} g_{u2}^{r_2} h^{-c1-c2} = a_1 a_2,$$

tehát $c' \equiv c_1 + c_2 \pmod q$ kellene ismét, újabb egyenletet azonban nem nyerünk c_1-re és c_2-re, azaz az egyiket bárhogy megválaszthatjuk. Ez azonban azt jelenti, hogy nem tudjuk kihasználni a párhuzamosságot, támadásunk tehát hatástalan, hiszen az egyfelhasználós támadások elleni biztonságot már beláttuk.

4. fejezet

Továbbíthatóság

Természetes követelményként merülhet föl, hogy az eddigi modellünkön változtassunk: miért is kellene az érméket rögtön fölhasználás után visszajuttatni a bankhoz, sokkal célszerűbb lenne, ha lehetne velük fizetni is, ezzel első ránézésre elég sok számítási kapacitást lehetne megspórolni.

Az új modellben tehát $\mathcal{U}_1$ készít egy C_1 érmét $\mathcal{B}$-vel együttműködve, $\mathcal{B}$-nél lévő számlája csökken az érme értékével. Fizetéskor C_1-ből legyártja C_2-t, hogy anonimitása sértetlen maradjon a vásárláskor, majd a C_2 érmét $\mathcal{U}_2$-nek adja. $\mathcal{U}_2$ a kapott érméből C_3-at csinál, amivel $\mathcal{U}_3$-nál fizethet, stb. Végül $\mathcal{U}_n$ úgy dönt, hogy nem akarja a kapott C_n érmét fölhasználni, ekkor C_{n+1}-et $\mathcal{B}$-hez visszajuttatva $\mathcal{B}$ jóváírja a számláján a megfelelő összeget.

A rendszerrel szemben azt a követelményt támasztjuk, hogy a $\mathcal{B}$-hez kerülő C_{n+1} érméből $\mathcal{B}$ ne legyen képes egyik $\mathcal{U}_i$ személyazonosságát sem kideríteni $1 \leq i \leq n-1$-re, akik előzőleg C_{n+1} elődeit használták. Természetesen $\mathcal{U}_n$ személyazonossága nem maradhat rejtve, hiszen az ő számlájára át kell utalni az érme értékét.

Az alábbiakban az előző fejezetben ismertetett rendszerből készítünk egy megfelelőt (7) általános módszere nyomán.

Kezdeti lépésként $\mathcal{B}$ két rendszert hoz létre, ugyanazt a G_q-t és $\mathcal{H}$-t használja, csak két különböző nyilvános kulcshármasa lesz: (g_0, g_1, h) illetve (g_0', g_1', h'). A rendszer felhasználói mindegyikének minkét helyen készít azonosítókat, ezek (x_u, I_u) és (x_u', I_u'). Az első rendszerben készített szokásos (g_u', a_u', c', r') érmét C, a másodikbelit C' jelöli. Látható, hogy egy érméről könnyű eldönteni, hogy melyik rendszerben készült, hiszen az érvényességhez $c' = \mathcal{H}(g', a_u', {g_u'}^{r'} h^{-c'})$-nek vagy $c' = \mathcal{H}(g', a_u', {g_u'}^{r'} {h'}^{-c'})$-nek teljesülnie kell, a másik egyenlőség nagy valószínűsséggel nem lesz igaz. A C érmék értéke a szokásos, míg a C' érmék önmagukban értéktelenek. A felhasználók tetszőleges időpontban készíthetnek C' érméket a bankkal, föltételezzük, hogy mindegyik $\mathcal{U}_i$-nek elegendően sok van a birtokában.

Amikor $\mathcal{U}_1$ szeretne egy valódi érmét kapni a banktól, teljesen hasonlóan jár el, mint az eredeti rendszerben, megkapja a C_1 érmét. Fizetéskor eddig $\mathcal{U}_2$ ellenőrizte $c' = \mathcal{H}(g_u', a_u', {g_u'}^{r'} h^{-c'})$-t, majd a $d = \mathcal{H}(spec, g_u', a_u')$ kérdésre az $r = (r_0, r_1)$ feleletet kapva ellenőrizte $g_0^{r_0} g_1^{r_1} = {g_u'}^d a_u'$-t, teljesülés esetén pedig elfogadta az érmét. Lássuk hogyan változik most mindez:

1. $\mathcal{U}_1$ átadja a $C_1 = (g_u', a_u', c', r')$ érmét $\mathcal{U}_2$-nak.

2. $\mathcal{U}_2$ ellenőrzi $c' = \mathcal{H}(g_u', a_u', {g_u'}^{r'} h^{-c'})$-t, majd választ egy $\mathcal{B}$-től régebben kért C_2' érmét. Az $\mathcal{U}_1$-nek küldendő d pedig $d_2 = \mathcal{H}(C_2', C, spec_2)$ lesz. $spec_2$ most nem tartalmazza $\mathcal{U}_2$ azonosítóját, szerepe az, hogy $\mathcal{U}_1$ – ha esetleg kétszer próbálná elkölteni C-t – ne kaphassa kétszer ugyanazt a d_2-t.

3. $\mathcal{U}_1$ d_2-re a már megszokott formájú r_2-t küldi válaszként.

4. $\mathcal{U}_2$ ellenőrzi, hogy helyes-e r_2, ha igen, akkor elfogadja az érmét.

Amikor uuk fizetni akar $\mathcal{U}_3$-nak a kapott érmével:

1. $\mathcal{U}_2$ elküldi $\mathcal{U}_3$-nak a $C_2 = (C_1, spec_2, r_2, C_2')$ érmét.

2. $\mathcal{U}_3$ $spec_2$ és r_2 segítségével ellenőrzi, hogy helyes-e a C érme, majd egy C_3' érme választása után a $d_3 = \mathcal{H}(C_3', C_2', spec_3)$ kérdésre
3. az r_3 feleletet kapja $\mathcal{U}_2$-tól, amivel
4. ellenőrzi C_2' helyességét.

$\mathcal{U}_n$ pedig visszajuttatja a kapott

$$C_{n-1} = (C_1, spec_2, r_2, C_2', \ldots, spec_{n-1}, r_{n-1}, C_{n-1}')$$

érmét kiegészítve $(spec_n, r_n, C_n')$-nel. Amennyiben $\mathcal{U}_n$ az érme megszerzésénél már tudja, hogy nem fogja továbbadni azt, akkor C_n' használata értelemszerűen elhagyható.

$\mathcal{B}$ ellenőrzi, hogy bármely két visszakapott érmében szerepel-e valahol megegyező C_1 vagy C_i', ha nem, akkor nem történt probléma, egyébként pedig a szokásos módon a tettes személyazonossága visszafejthető.

A rendszer egyik feltűnő problémája, hogy az érme minden lépésnél növekszik, ezzel arányosan nő az érvényesség ellenőrzésének ideje is. Chaum és Pedersen (8)-ban bizonyították, hogy sajnos ez szükségszerű következménye annak a követelménynek, hogy C_n-ből ne lehessen $\mathcal{U}_1, \ldots, \mathcal{U}_{n-1}$ személyazonosságát meghatározni.

Irodalomjegyzék

[1] B. Schoenmakers, „Basic Security of the ecash Payment System", Computer Security and Industrial Cryptography: State of Art and Evolution, ESAT Course, Leuven, Belgium. LNCS series, 1997.

[2] R.L. Rivest, A. Shamir, L. Adleman, „A Method for Obtaining Digital Signatures and Public-key Cryptosystems", Communications of the ACM, Vol. 21, No. 2, pp. 120-126, 1978.

[3] D. Boneh, G. Durfee, „Cryptanalysis of RSA with Private Key d Less than N^0.292", Eurocrypt '99, LNCS 1592, pp. 1-11, 1999.

[4] D. Chaum, A. Fiat, M. Naor, „Untraceable Electronic Cash", Crypto '88, LNCS 403, pp. 319-327, 1990.

[5] C.P. Schnorr, „Efficient Signature Generation by Smart Cards", Journal of Cryptology, pp. 161-174, 1991.

[6] B. Schoenmakers, „An Efficient Payment System Withstanding Parallel Attacks", Report CS-R9522, Centrum voor Wiskunde en Informatica, 1995.

[7] H. van Antwerpen, „Electronic Cash", Master's thesis, CWI, 1990.

[8] D. Chaum, T. B. Pedersen, „Transferred Cash Grows in Size", EuroCrypt '92, LNCS 658, pp. 390-417, 1993.