TITOKMEGOSZTÁS ÉS TÖBBRÉSZTVEVŐS SZÁMÍTÁSOK

Szakdolgozat

Írta: Zentai Dániel
Matematika bsc szak
Alkalmazott matematikus szakirány

Témavezető:
Dr. Csirmaz László

Konzulens:
Dr. Sziklai Péter

Eötvös Loránd Tudományegyetem
Természettudományi Kar

Tartalomjegyzék

1. fejezet

Bevezetés

Egy kalózkapitány zsákmányolt kincseit elássa, és a rejtekhelyről készít egy térképet. A vagyonát három fiára szeretné hagyni hagyni, ezért három részre vágja a térképet és minden gyerekének ad egy-egy darabot, hogy csak közösen tudják megtalálni a kincset.
Egy bankban minden széfhez két kulcs tartozik. Az egyik kulcs a széf bérlőjénél van, a másik egy banki alkalmazottnál. Csak ezen két személy együttes jelenlétében lehetséges a széf kinyitása.
Számtalan ehhez hasonló szituációt el tudunk képzelni, a lényege azonban mindegyiknek ugyanaz: van egy titkunk, amihez csak több ember együttműködése révén juthatunk hozzá. Erre a problémára kínál megoldást a titokmegosztás, és a hozzá szorosan kapcsolódó többrésztvevős számítások. A titokmegosztás fogalmát egymástól függetlenül Shamir [10] és Blakley [3] vezette be 1979-ben. A többrésztvevős számítások fogalma Yao [11] nevéhez fűződik. Dolgozatom célja ezen két témakör bemutatása. A dolgozat írása során főként Maurer [8] cikkére támaszkodtam.
A 2. fejezetben ismertetem a dolgozatban előforduló alapfogalmakat, definíciókat.
A 3. fejezetben szerepelnek egyszerű példák titokmegosztásra, illetve a [8]-ban szereplő titokmegosztással kapcsolatos tételek, eljárások.
A 4. fejezetben az előző fejezet tételeinek, eljárásainak egy hibájára mutatok rá, és javítom ezt hibát.
Az 5. fejezetben többrésztvevős számítással kapcsolatos példákat, általános tételeket, számolási módszereket mutatok.
A 6. fejezetben az üzenetszórás szimulálásáról, és ennek helyességéről lesz szó.

2. fejezet

Alapfogalmak ismertetése

2.1. A támadás modellezése

Titokmegosztásnál és többrésztvevős számításnál úgy modellezzük a támadást, hogy külső fenyegetést tételezünk fel, egy támadót, aki a résztvevők bizonyos részhalmazait tudja támadni. Tekinthetünk erre a támadóra például úgy, mint egy hackerre, aki a résztvevők számítógépébe képes betörni.

Két fajtáját különböztethetjük meg a támadásnak. A *passzív* támadás azt jelenti, hogy a támadó betekintést nyerhet a résztvevők által birtokolt információkba, de ettől eltekintve a résztvevők továbbra is szabályszerűen vesznek részt a protokollban. A továbbiakban a passzív támadás alatt álló résztvevőket *tisztességes, de kíváncsi* résztvevőknek hívjuk.
Az *aktív* támadás azt jelenti, hogy a támadó teljesen átveszi az irányítást a résztvevők fölött. Ekkor az is megtörténhet, hogy a résztvevők nem követik a protokoll szabályait, és a számítás során téves információkat közölnek a többi résztvevővel. Az aktív támadás alatt álló résztvevőket a továbbiakban *tisztességtelen* résztvevőknek hívjuk.
Azokat a résztvevőket, akik sem aktív, sem passzív támadás alatt nem állnak *tisztességes* résztvevőknek hívjuk.

Természetes feltevés, hogy a résztvevők egy támadható halmazának bármely részhalmaza is támadható.

2.1.1. Definíció. *Legyen $\mathcal{P}$ véges halmaz.*
Azt mondjuk, hogy $\Pi \subset 2^{\mathcal{P}}$ egy struktúra, ha Π zárt a részhalmazképzésre, azaz $S \in \Pi$, és $S' \subset S$ esetén $S' \in \Pi$.

2.1.2. Definíció. *Legyen $\mathcal{P}$ véges halmaz.*
Azt mondjuk, hogy $\Pi \subset 2^{\mathcal{P}}$ egy anti-struktúra, ha Π zárt a tartalmazásra, azaz $S \in \Pi$, és $S' \supset S$ esetén $S' \in \Pi$.

2.1.3. Definíció. *Legyen Π_1 és Π_2 struktúra.*
Ekkor $\Pi_1 \sqcup \Pi_2$ jelöli a Π_1 és Π_2 struktúra elemenkénti unióját, azaz $\Pi_1 \sqcup \Pi_2 = \{S_1 \cup S_2 : S_1 \in \Pi_1, S_2 \in \Pi_2\}$.

2.1.1. Megjegyzés. *Egy struktúra megadásánál elegendő csak a legbővebb halmazokat felsorolni, hiszen ha Π egy struktúra, és $S \in \Pi$, akkor minden $S' \subset S$-re automatikusan teljesül $S' \in \Pi$.*

2.1.4. Definíció. *Legyen $\mathcal{P}$ véges halmaz, $\Sigma \subset 2^{\mathcal{P}}$, $\Delta \subset \Sigma$ struktúra.*
Egy (Σ, Δ)-támadó egy olyan támadó, aki a Σ-beli halmazokat csak passzívan, a Δ-beli halmazokat aktívan is támadhatja.

Mind a passzív, mind az aktív támadó lehet *statikus*, illetve *adaptív*. A statikus támadó csak a számítási protokoll kezdete előtt választhatja ki, hogy mely részhalmazokat támadja passzívan, és melyeket aktívan. Az adaptív támadó a protokoll futása közben is újra választhat a támadható részhalmazok közül.

2.2. A kommunikáció modellezése

Biztonság szempontjából a kommunikáció kétféle lehet.
Kriptográfiai biztonságról akkor beszélünk, ha a támadó számítási kapacitására megszabunk egy korlátot, vagy bizonyos feladatokról feltesszük, hogy a támadó nem ismer a megoldásukra hatékony algoritmust (például ezzel a feltevésssel élünk az RSA titkosítás esetén a prímfaktorizációval kapcsolatban.)
Információ-elméleti, vagy *tökéletes* biztonságról akkor beszélünk, ha a támadónak korlátlan számítási kapacitás áll rendelkezésre, mégsem tudhat meg semmit a kommunikációról.

A kommunikációs csatorna is kétféle lehet.
Szinkronizált a csatorna, ha az üzenetek küldése összehangolt, és minden üzenet megérkezik egy adott időkorláton belül. Ekkor a kommunikáció körökre van osztva, egy körön belül minden résztvevő küldhet üzenetet az összes többi résztvevőnek, és az üzenetek megérkeznek a következő kör kezdete előtt.

Aszinkronizált a csatorna, ha az üzenetküldés nincs összehangolva, és az üzenetek érkezése sincs időkorlátok közé szorítva.
Üzenetszórásról beszélünk, ha az üzenetet úgy küldjük el, hogy azt az összes résztvevő megkapja egyszerre. Az üzenetszórás nem mindig valósítható meg. Ilyenkor az üzenetszórást szimulálni szokás. Az üzenetszórás szimulálását később részletesen tárgyaljuk.

3. fejezet

Titokmegosztás

Arra, hogy egy többrésztvevős számításban a biztonságot garantálni tudjuk, *titokmegosztás*t használunk. A résztvevők véges $\mathcal{P} = \{p_1, \ldots, p_n\}$ halmazának elemei között szeretnénk egy titkot szétosztani úgy, hogy bármely k résztvevő együtt vissza tudja állítani a titkot, de semelyik $k-1$ résztvevő se tudjon meg többet a titokról, mint amit az inputjából, és a végeredményből megtudhat. A titok tipikusan valamely $\mathbb{F}$ véges test elemei közül kerül ki. A titokrészeket szétosztó résztvevőt *osztó*nak nevezzük.

3.0.1. Definíció. *Legyenek $S_1, \ldots, S_n$ véges halmazok, ahol S_i a p_i résztvevőnek kioszthatótitkok halmaza, és legyen $S = S_1 \times \cdots \times S_n$.*
S elemeit titokmegosztásnak hívjuk.

3.0.2. Definíció. *Az olyan titokmegosztási sémát, melyben bármely k résztvevő vissza tudja állítani a titkot, de semelyik $k-1$ résztvevőnek sincs semmilyen plusz információja a titokról, k-küszöb sémának nevezzük.*

3.0.3. Definíció. *Egy $\mathcal{P}' \subseteq \mathcal{P}$ halmaz kvalifikált, ha a benne szereplő résztvevők vissza tudják állítani a titkot.*
$\mathcal{P}' \subseteq \mathcal{P}$ ignorált, ha nem kvalifikált.

3.0.4. Definíció. *Legyen Γ a kvalifikált részhalmazokból, Σ az ignorált részhalmazokból álló struktúta.*
Γ-t elérési struktúrának, Σ-t biztonsági struktúrának hívjuk.

3.0.1. Megjegyzés. *Itt az elérési struktúra csak egy elnevezés. Γ valójában egy anti-struktúra, hiszen ha a résztvevőknek egy részhalmaza vissza tudja állítani a titkot, akkor minden ennél bővebb részhalmaz is vissza tudja állítani.*

3.1. Példák

3.1.1. Algebrai titokmegosztás

Titokmegosztásra először Shamir dolgozott ki algoritmust 1979-ben [10].
Shamir titokmegosztásának bemutatása előtt áttekintjük a Lagrange-interpoláció algoritmusát:

Legyen $\mathbb{F}$ egy véges test, $|\mathbb{F}| \geq k+1$, $C \subset \mathbb{F}$, $|C| = k+1$ $f(x) \in \mathbb{F}[x]$ egy k-adfokú egyváltozós polinom. Ha ennek a polinomnak ismerjük $k+1$ helyen a helyettesítési értékét, akkor Lagrange-interpolációval meg tudjuk határozni magát a polinomot, mégpedig

$$f(x) = \sum_{i \in C} f(i) l_i(x)$$

, ahol $l_i(x)$ egy k-adfokú polinom, melyre $l_i(j) = 0$, ha $i \neq j$, és $l_i(j) = 1$, ha $i = j$. Formálisan

$$l_i(x) = \prod_{j \in C, j \neq i} \frac{x-j}{i-j}$$

.

Ezek után Shamir algoritmusa a következő:
Tekintsünk egy $\mathbb{F}$ véges testet, és egy $f(x) \in \mathbb{F}[x]$ polinomot. Legyen $f(x) = a_k x^k + \ldots a_1 x + a_0$, ahol $f(0) = a_0$ a titok, $a_1, \ldots, a_n$-et pedig azonos (tipikusan egyenletes) eloszlás szerint, egymástól függetlenül választjuk $\mathbb{F}$-ből. Mivel az f polinom k-adfokú, bármely $k+1$ résztvevő Lagrange-interpolációval rekonstruálni tudja f-et, és ezáltal a polinom konstans tagját, azaz titkot is, de semelyik k résztvevő sem tudhat meg semmit a titokról, azaz ez az algoritmus egy $(k+1)$-küszöb sémát ad.

3.1.2. Geometriai titokmegosztás

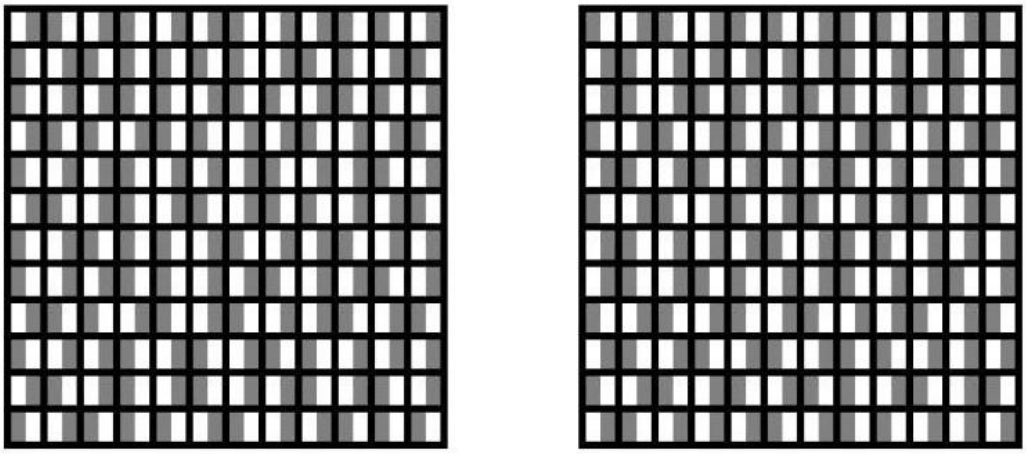
Shamirtól függetlenül szintén 1979-ben Blakley is kidolgozott egy módszert titokmegosztásra [3].

Legyen most a titok egy tetszőleges Q pont az n-dimenziós euklideszi térben. A titokrészek legyenek a Q pont koordinátái, azaz kapja minden résztvevő a Q pont egy koordinátáját. Ekkor n résztvevő együtt meg tudja határozni a Q

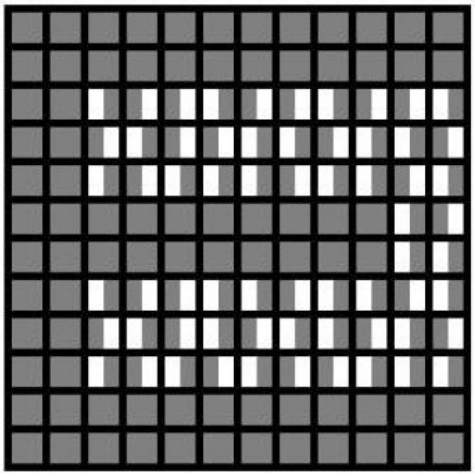
pont összes koordinátáját, de bármely $n-1$ résztvevő csak az n-dimenziós tér egy olyan $n-1$-dimenziós alterét ismeri, amin a Q pont található, így magát a Q pontot nem tudja visszaállítani semelyik $n-1$ résztvevő. Ez az eljárás egy n-küszöb séma.

3.1.3. Optikai titokmegosztás

Következő példánk egy 2-küszöb séma. Felosztunk két átlátszó fóliát kis téglalapokra, majd az egyik fólián minden kis téglalapot kiszínezünk úgy, hogy az egyik oldala fekete legyen, a másik pedig átlátszó maradjon. Mindig $1/2$ valószínűséggel színezzük a bal oldalát, $1/2$ valószínűséggel pedig a jobb oldalát feketére. A másik fóliát úgy színezzük ki, hogy a két fóliát egymással fedésbe hozva a titok (egy kép, vagy egy szöveg) rajzolódjon ki. Azaz azokat a téglalapokat, ahol a titoknak ki kell rajzolódnia, ellentétesen színezzük, mint az első fólián, a többi téglalapot pedig ugyanúgy, mint az első fólián. Mindkét fólia színezése csak a másik fóliától függ, önmagában minden kis téglalapot egymástól függetlenül, azonos eloszlás szerint színeztünk ki. A következő két ábra egy egyszerű példát mutat optikai titokmegosztásra.



3.1.1. ábra. A két fólia külön



3.1.2. ábra. A két fólia egymással fedésbe hozva

3.2. Biztonság passzív támadás ellen

Ebben a fejezetben olyan protokollokat tekintünk át, amelyek passzív támadás ellen biztonságosak. Mivel csak passzív támadást feltételezünk, ebben a fejezetben $\Delta = \emptyset$.
A titokrészek szétosztásához egy építőelem a *k-ból k titokmegosztás*. Legyen a szétosztandó titok $s \in \mathbb{F}$, ahol $\mathbb{F}$ egy véges test.

K-ból k titokmegosztás
(1) Válasszunk k elemet, $s_1, \ldots, s_n$-et egymástól függetlenül, azonos (tipikusan egyenletes) eloszlás szerint $\mathbb{F}$-ből. (2) Legyen $s' = \sum_{i=1}^{k} s_i$. Amennyiben $s' \neq s$, újra választjuk $s_1, \ldots, s_n$-et. Ha $s' = s$, akkor az i. titokrészt s_i-nek választjuk.

Ekkor k résztvevő együtt vissza tudja állítani a titkot, hiszen a titok épp az általuk birtokolt titokrészek összege. Azonban bármely legfeljebb $k - 1$ résztvevő csak egymástól és a titoktól statisztikailag független valószínűségi változókat ismer, melyek összege szintén független a titoktól.

Titokmegosztás $\Sigma = \{T_1, \ldots, T_k\}$ biztonsági struktúrában
(1) Részekre osztjuk a titkot k-ból k titokmegosztással.
(2) Szétosztjuk a titkot a résztvevők között úgy, hogy p_j akkor kapja meg az s_i titokrészt, ha $p_j \in \overline{T_i}$.

Ezzel a titokmegosztási sémával titokrészek lineáris függvényeit biztonságosan ki lehet számolni, hiszen a linearitás miatt ezeket a függvényeket a résztvevők lokálisan ki tudják számítani, így nincs szükség kommunikációra. A biztonság garantált, hiszen minden $T_i \in \Sigma$ halmazra teljesül $s_i \notin T_i$, így egyetlen Σ-beli halmaz sem tudja visszaállítani a titkot. Ugyanakkor ha Γ elérési struktúra, akkor minden $S \in \Gamma$ minden i-re tartalmaz $p \notin T_i$ résztvevőt. Ez a p résztvevő ismeri az s_i titokrészt, így az S-beli résztvevők ismerik az összes titokrészt.

3.2.1. Definíció. *Legyenek $s = \sum_{i=1}^{k} s_i$ és $t = \sum_{j=1}^{k} t_j$ titkok.*
Ekkor s és t szorzatát a következőképp definiálhatjuk:

$$st = (\sum_{i=1}^{k} s_i)(\sum_{j=1}^{k} t_i) = \sum_{i=1}^{k}\sum_{j=1}^{k} s_i t_j.$$

Szorzás protokoll
(1) Osszuk fel az $\{(i,j) : 1 \leq i,j \leq n\}$ halmazt az $U_1, \ldots, U_n$ halmazokra úgy, hogy $(i,j) \in U_m \Leftrightarrow p_m \in \overline{T_i} \cap \overline{T_j}$.
(2) Osszuk szét az $s = \sum_{i=1}^{k} s_i$ és a $t = \sum_{j=1}^{k} t_j$ titkokat.
(3) Minden $1 \leq m \leq n$-re p_m kiszámolja a $v_m = \sum_{(i,j)\in U_m} s_i t_j$ értéket, és ezt megosztja az összes többi résztvevővel.
(4) Az összes résztvevő kiszámolja a $v = \sum_{m=1}^{n} v_m$ értéket.

3.3. Biztonság aktív támadás ellen

A passzív támadáshoz képest ebben az esetben két fontos változtatást kell eszközölnünk. A résztvevők továbbra is a titokrészek lineáris függvényeit és szorzatát számolják ki, de az alkalmazott protokollokat ellenállóvá kell tennünk egy aktív támadóval szemben is, ezen kívül azt a problémát is orvosolnunk kell, ha esetleg az osztó is tisztességtelen.

Az eddigi titokmegosztási sémák használhatatlanok aktív támadók jelenlétében, mert nincs semmi jele annak, ha egy résztvevő nem a valós eredményt közli a többiekkel számítás közben, és ezáltal a végeredmény sem lesz helyes. Ezt a

problémát oldja meg az *ellenőrizhető titokmegosztás*.

3.3.1. Definíció. *Az ellenőrizhető titokmegosztási séma egy* ***(Megoszt, Visszaállít)*** *protokoll pár, ahol a következők teljesülnek egy* (Σ, Δ)*-támadó jelenlétében:*

1. *Ha a* ***Megoszt*** *protokoll sikeresen lefut, akkor a* ***Visszaállít*** *protokoll a támadó minden lehetséges stratégiája esetén ugyanazt a fix értéket fogja visszaadni.*

2. *Ha az osztó megbízható a* ***Megoszt*** *protokoll futása alatt, akkor a* ***Visszaállít*** *protokoll mindig az osztó inputját adja vissza.*

3. *Ha az osztó megbízható, akkor a támadó semmilyen plusz információt nem tudhat meg a titokról.*

3.3.1. Tétel. *[8] Pontosan akkor lehet tökéletesen biztonságos egy ellenőrizhető titokmegosztás egy* (Σ, Δ)*-támadó ellen, ha*

$$\mathcal{P} \notin \Sigma \sqcup \Delta \sqcup \Delta$$

, azaz ha a résztvevők teljes halmazát nem lehet lefedni semelyik kettő tisztességtelen, illetve egy tisztességes, de kíváncsi halmaz uniójával.

3.3.2. Tétel. *[8] Pontosan akkor szimulálható üzenetszórás biztonságosan egy* (Σ, Δ)*-támadó ellen, ha*

$$\mathcal{P} \notin \Delta \sqcup \Delta \sqcup \Delta$$

, azaz ha a résztvevők teljes halmazát nem lehet lefedni semelyik három tisztességtelen halmaz uniójával.

Megoszt protokoll
(1) Osszuk szét a titkot passzív támadás ellen biztonságos titokmegosztással. (2) Minden s_i titokrészre a $\overline{T_i}$-beli résztvevők páronként ellenőrzik, hogy mindegyikük ugyanazt az s_i értéket kapta -e. Ha eltérést találnak, azt üzenetszórással jelzik. (3) Az osztó üzenetszórással elküldi az összes résztvevőnek az összes olyan s_i titokrészt, amelyre a résztvevők hibát jeleztek. Amennyiben az osztó megtagadja az üzenetszórást, a protokoll leáll.

Itt (2)-re azért van szükség, hogy az osztót is tudjuk ellenőrizni. Ugyanis a passzív támadással ellentétben, aktív támadásnál előfordulhat, hogy az osztó tisztességtelen, és nem ugyanazt az s_i titokrészt küldi az összes $\overline{T_i}$-beli résztvevőnek.

Visszaállít protokoll
(1) Minden résztvevő elküldi az általa ismert titokrészeket az összes többi résztvevőnek. (2) Minden résztvevő lokálisan kiszámolja az $s = \sum_{i=1}^{k} s_i$ értéket. (3) Legyen v_j az az érték, amit a $p_j \in \overline{T_i}$ résztvevő küld el a többi résztvevőnek. A titok az a v értek lesz, amire létezik olyan $A \in \Delta$, hogy $v_j = v$ minden $p_j \in \overline{T_i} - A$-ra.

A titokrészek lineáris függvényeinek számolásakor ebben az esetben sincs probléma, hiszen a linearitás miatt ezeket a függvényeket a résztvevők lokálisan ki tudják számítani, így nincs szükség kommunikációra. A **Szorzás** protokollt azonban meg kell változtatnunk úgy, hogy tisztességtelen résztvevők jelenlétében is biztonságos maradjon. Azaz kezelnünk kell azt a problémát, amikor egy, vagy több résztvevő nem valós információkat közöl a többi résztvevővel. Az aktív támadás ellen is biztonságos **Szorzás** protokoll a következő:

Szorzás protokoll

(1) Ellenőrizhető titokmegosztással megosztjuk az $s = \sum_{j=1}^{k} s_i$ és a $t = \sum_{j=1}^{k} t_j$ titkokat.

(2) Minden p_m résztvevő kiszámolja azon $s_i t_j$ titokrészeket, melyekre $p_m \in \overline{T_i} \cap \overline{T_j}$, majd megosztja ezeket ellenőrizhető titokmegosztással.

(3) Minden (i, j) párra legyen $\{p_{m_1}, \ldots, p_{m_r}\}$ azon résztvevők halmaza, akik a $s_i t_j$ titokrészt számolták ki.

(4) A résztvevők kiszámolják a p_{m_1} és p_{m_i} ,$(i = 2, \ldots, r)$ által küldött $s_i t_j$ titokrészek különbségét. Ha minden esetben 0 az eredmény, elfogadják $s_i t_j$-t, különben rekonstruálják s_i-t és t_j-t, p_1 megkapja $s_i t_j$-t, a többi résztvevő pedig 0-t.

(5) Minden résztvevő kiszámolja $st = (\sum_{i=1}^{k} s_i)(\sum_{j=1}^{k} t_i) = \sum_{i=1}^{k} \sum_{j=1}^{k} s_i t_j$-t.

4. fejezet

Az előző tételek hiányossága

A [8]-ban közölt tételek, illetve protokollok némelyikének van egy hiányossága, ugyanis sérülhet a titokmegosztás biztonsága bizonyos értelemben. Ugyan a támadó továbbra sem juthat hozzá illetéktelenül semmilyen információhoz, de azt elérheti, hogy tisztességes résztvevők egy halmaza olyan információhoz jusson, amire a résztvevők lennének jogosultak.

Az s_i titokrész páronkénti ellenőrzésekor egy tisztességtelen résztvevő mondhatja azt, hogy ő egy $s_i' \neq s_i$ titokrészt kapott, még ha az osztó valóban tisztességes is volt, és minden $\overline{T_i}$-beli résztvevőnek ugyanazt a titokrészt adta. Ekkor a résztvevők jelzik az eltérést az osztónak, aki üzenetszórással elküldi mindenkinek s_i-t. Ekkor pedig a T_i-beli résztvevők is megtudják s_i-t.

Ha a T_i- beli résztvevők egyike sem tisztességes, akkor nem sérül a biztonság, ugyanis vagy azért volt szükség üzenetszórásra, mert az osztó volt tisztességtelen, vagy azért, mert legalább egy tisztességtelen résztvevő már megtudta s_i-t, és mindkét esetben az összes tisztességtelen, illetve tisztességes, de kíváncsi résztvevő megtudhatja s_i-t, hiszen ezek a résztvevők együttműködhetnek.
Ha viszont T_i-ben voltak tisztességes résztvevők, akkor ezen résztvevők plusz információhoz jutottak az üzenetszórás miatt. Ahhoz, hogy ne sérülhessen ilyen módon a biztonság, az előbbi tételekben, illetve protokollokban némi változást kell eszközölnünk.

Egyrészt az 3.2.1. Tételt kell módosítanunk úgy, hogy az s_i titokrész szétosztásából kihagyott T_i halmazon kívül is biztonságosan működhessen a titokmegosztás, azaz azt szeretnénk, hogy a $\overline{T_i}$ halmazra is teljesüljenek a tétel feltételei. Ehhez az kell, hogy $\overline{T_i}$-t ne lehessen lefedni semelyik kettő tisztességtelen és egy tisztességes, de kíváncsi halmaz uniójával.

Mivel $T_i \in \Sigma$, az 3.2.1. Tételben szereplő feltételt a következőképp kell módosítanunk a **Megoszt** protokoll esetében:

4.0.3. Tétel. *Pontosan akkor lehet tökéletesen biztonságos egy ellenőrizhető titokmegosztás egy (Σ, Δ)-támadó ellen, ha*

$$\mathcal{P} \notin \Sigma \sqcup \Sigma \sqcup \Delta \sqcup \Delta$$

, azaz ha a résztvevők teljes halmazát nem lehet lefedni semelyik kettő tisztességtelen, illetve semelyik kettő tisztességes, de kíváncsi halmaz uniójával.

Ezenkívül módosítanunk kell a **Megoszt** protokollt, hogy ha a résztvevők hibát jeleznek (akár valós, akár valótlan hibát), az osztó üzenetszórása által az s_i titokrész ne juthasson el a $\overline{T_i}$ halmaz résztvevőihez. Ez a következő módosítással érhetjük el.

A Megoszt protokoll
(1) Osszuk szét a titkot passzív támadás ellen biztonságos titokmegosztással. (2) Minden s_i titokrészre a $\overline{T_i}$-beli résztvevők páronként ellenőrzik, hogy mindegyikük ugyanazt az s_i értéket kapta -e. Ha eltérést találnak, azt üzenetszórással jelzik. (3) Az osztó szimulált üzenetszórással elküldi az összes $\overline{T_i}$-beli résztvevőnek az s_i titokrészt, amennyiben a résztvevők hibát jeleztek s_i szétosztásakor. Amennyiben az osztó megtagadja az üzenetszórást, a protokoll leáll.

Mivel ebben a módosított **Megoszt** protokollban olyan szimulált üzenetszórásra van szükségünk, amely minden $\Sigma = \{T_1, \ldots, T_k\}$-beli halmazon kívül biztonságosan működik, módosítanunk kell a 3.2.2. Tételt is.

4.0.4. Tétel. *Pontosan akkor szimulálható üzenetszórás biztonságosan egy (Σ, Δ)-támadó ellen, ha*

$$\mathcal{P} \notin \Sigma \sqcup \Delta \sqcup \Delta \sqcup \Delta$$

, azaz ha a résztvevők teljes halmazát nem lehet lefedni semelyik három tisztességtelen, illetve egy tisztességes, de kíváncsi halmaz uniójával.

Maga a **Szorzás** protokoll nem szorul módosításra, a módosítást már megtettük a protokoll (1) lépésében szereplő **Megoszt** protokollban. Azonban szintén a protokoll (1) lépésében csak az olyan p_m résztvevők tudják kiszámítani az $s_i t_j$ szorzatot, melyekre $p_m \notin \overline{T_i} \cap \overline{T_j}$. Tehát az ellenőrizhető titokmegosztásnak a $\overline{T_i} \cap \overline{T_j}$ halmazon is biztonságosnak kell lennie, azaz az 3.2.1. Tétel feltételeinek

teljesülnie kell a $\overline{T_i} \cap \overline{T_j} = \overline{T_i \cup T_j}$ halmazra. Mivel $T_i \cup T_j \in \Sigma \sqcup \Sigma$, az 5. Tételben szereplő feltételt a következőképp kell módosítanunk a **Szorzás** protokoll esetében:

4.0.5. Tétel. *Pontosan akkor lehet tökéletesen biztonságos egy ellenőrizhető titokmegosztás egy* (Σ, Δ)*-támadó ellen, ha*

$$\mathcal{P} \notin \Sigma \sqcup \Sigma \sqcup \Sigma \sqcup \Delta \sqcup \Delta$$

, azaz ha a résztvevők teljes halmazát nem lehet lefedni semelyik kettő tisztességtelen, illetve semelyik három tisztességes, de kíváncsi halmaz uniójával.

5. fejezet

Többrésztvevős számítások

Többrésztvevős számítás alatt azt értjük, hogy n résztvevő egy $f : \mathbb{F} \to \mathbb{F}$ függvény értékét kiszámítja biztonságosan, ahol $\mathbb{F}$ egy véges test. Konkrétabban, ha a résztvevők véges halmaza $\mathcal{P} = \{p_1, \ldots, p_n\}$ az inputok $x_1, \ldots, x_n \in \mathbb{F}$, ahol p_i ismeri az x_i inputot, akkor a résztvevők kiszámítják egy adott $f : \mathbb{F} \to \mathbb{F}$ függvényre $f(x_1, \ldots, x_n) = (y_1, \ldots, y_n)$-t. A biztonság itt azt jelenti, hogy garantáljuk a végeredmény helyességét, illetve azt, hogy egyik résztvevő sem tudhat meg semmilyen plusz információt a kommunikációból. Azaz minden résztvevő csak azt tudhatja, amit az általa ismert inputból és a végeredményből ki tud találni, és ennél többet még akkor sem tudhat, ha a résztvevők között nem mindenki tisztességes.

Mivel a számolásokat a résztvevők egy $\mathbb{F}$ véges test fölött végzik, elegendő, ha csak a következő műveleteket használják. Egyrészt szükség van valamilyen titokmegosztási sémára, mellyel minden résztvevő szétosztja a birtokában lévő inputot. Ezenkívül szükség van a szétosztott titokrészek összeadására, összeszorzására, illetve konstanssal való szorzására. Ezekkel a műveletekkel minden $\mathbb{F}$-beli polinom előállítható. Arra, hogy miért elég csak polinomokkal számolni, a következő tétel ad magyarázatot.

5.0.6. Tétel. *Véges test fölött minden függvény polinom.*

5.1. Példák

5.1.1. Milliomosok problémája

Egy példa többrésztvevős számításra a Yao-féle *milliomosok problémája* [11]: Két milliomos rá szeretne jönni, hogy melyikük a gazdagabb, mindezt anélkül, hogy bármelyikük megtudná, hogy a másiknak mennyi pénze van. Ebben az esetben mindkét résztvevő csak a saját vagyonát ismerheti, illetve azt, hogy melyikük a gazdagabb. Ez egy kétrésztvevős számítás, ahol az input két egész szám, a végeredmény pedig annak az igazságértéke, hogy az első szám volt -e a nagyobb. A következőt tehetik a milliomosok:

Legyen a két milliomos A és B. A-nak van i millió forintja, B-nek j milliója. Tegyük fel, hogy $0 < i, j < 100$, i és j egészek. Jelölje egy x üzenet A nyilvános kulcsával való titkosítását $y = E_A(x)$, ennek visszafejtését $x = D_A(y)$.

1. B generál egy N jegyű véletlen egész számot, ahol N páros pozitív egész. Legyen ez a szám x. B elküldi A-nak a $k - j + 1$ értéket, ahol $k = E_A(x)$.

2. A kiszámolja $y_u = D_A(k - j + u)$-t, $u = 1, \ldots, 100$

3. A generál egy $N/2$ jegyű p prímszámot, és kiszámolja $z_u = y_u \ (mod\ p)$-t minden u-ra. Ezt ismétli, amíg olyan z_u-kat nem kap, melyeknek páronkénti különbsége legalább 2.

4. A elküldi B-nek a $p, z_1, \ldots, z_i, z_i + 1, \ldots, z_{100} + 1 \ (mod\ p)$ számokat.

5. B megnézi p-t nem számítva a j. számot. Ha ez egyenlő x-szel $(mod\ p)$, akkor $i \geq j$, különben $i < j$.

5.1.2. Ebédlő kriptográfusok

Egy másik példa többrésztvevős számításra a Chaum-féle *ebédlő kriptográfusok problémája* [4]: Három kriptográfus elmegy ebédelni egy étterembe, majd amikor fizetni szeretnének, a pincér közli velük, hogy a számla már ki van egyenlítve. Két lehetőség van. Vagy az NSA (az Amerikai Nemzetbiztonsági Hivatal) fizette ki az ebédjüket, vagy hármuk közül valaki. Ki szeretnék deríteni, hogy mi is történt, de úgy, hogy ha hármuk közül rendezte valaki a számlát, akkor ne derüljön ki, hogy melyikük. Ez egy háromrésztvevős számítás, ahol a számítás előtt minden résztvevő csak azt tudja, hogy ő maga fizetett -e, a

számítás után pedig mindhárman tudják, hogy az NSA fizetett -e, vagy hármuk közül valaki. A következőt kell tenniük:

Leülnek körben egy asztalhoz, és feldobnak egy-egy pénzérmét. Mindegyikük úgy helyezi el az érmét, hogy csak ő és a tőle jobbra ülő kollégája láthassa azt. A fejnek megfeleltetik a 0, az írásnak az 1 értéket. Összeadják az általuk látott érméknek megfelelő értéket modulo 2, és az, aki fizetett, még 1-et hozzáad az eredményhez modulo 2. Mindhárman megmondják a kapott eredményt, majd ezeket is összeadják modulo 2. Mivel minden 1-est és minden 0-t pontosan két ember lát, ezért ha ezeket összeadják modulo 2 a végeredmény 0 lesz, kivéve ha egyikük még hozzáadott az eredményhez 1-et modulo 2. Így ha a végeredmény 0, akkor az NSA fizetett, ha pedig 1, akkor hármuk közül valaki.

5.2. Általános tételek

Ahhoz, hogy egy többrésztvevős számítás biztonságos lehessen, korlátoznunk kell a támadható résztvevők számát. A következő tételek felső korlátot adnak a passzívan, illetve az aktívan támadható résztvevők számára.

5.2.1. Tétel. *[6] Legyen t_a az aktívan, t_p a passzívan támadható résztvevők száma.*
Egy n résztvevős számítás pontosan akkor lehet kriptográfiailag biztonságos, ha $t_p < n$, és $t_a < n/2$.

5.2.2. Tétel. *[2],[5] Legyen t_a az aktívan, t_p a passzívan támadható résztvevők száma.*
Egy n résztvevős számítás pontosan akkor lehet tökéletesen biztonságos, ha $t_p < n/2$, és $t_a < n/3$.

5.2.3. Tétel. *[9],[1] Legyen t_a az aktívan támadható résztvevők száma.*
Egy n résztvevős számítás üzenetszórásos csatornával pontosan akkor lehet tökéletesen biztonságos, ha $t_a < n/2$.

5.2.4. Tétel. *[8] Pontosan akkor lehet tökéletesen biztonságos egy többrésztvevős számítás egy (Σ, Δ)-támadó ellen, ha*

$$\mathcal{P} \notin \Sigma \sqcup \Sigma \sqcup \Delta$$

, azaz ha a résztvevők teljes halmazát nem lehet lefedni semelyik kettő tisztességes, de kíváncsi, illetve egy tisztességtelen halmaz uniójával.

5.3. A szükséges műveletek

Ebben a fejezetben áttekintjük, hogy hogyan végezhetjük el biztonságosan a többrésztvevős számításhoz szükséges műveleteket. Mint már említettük a szükséges műveletek a titokrészek összeadása, a konstanssal való szorzás, illetve a titokrészek szorzása. Tökéletes biztonságot szeretnénk elérni, így az 5.2.2 Tétel alapján fel kell tennünk, hogy $t_p < n/2$, valamint $t_a < n/3$, ahol n az összes résztvevő száma, t_p a tisztességes, de kíváncsi résztvevők száma, t_a pedig a tisztességtelen résztvevők száma.

Az inputok szétosztása. Az összes p_i résztvevő titokmegosztással szétosztja a birtokában lévő $x_i \in \mathbb{F}$ inputot.

Összeadás. Legyen a és b két szétosztandó input, $a_1, \ldots, a_n$ és $b_1, \ldots, b_n$ rendre a hozzájuk tartozó titokrészek. Ekkor mindegyik p_i résztvevő lokálisan kiszámolja $a_i + b_i$-t. Az $a_1 + b_1, \ldots, a_n + b_n$ titokrészek pedig egyértelműen meghatározzák az $a + b$ titkot.

Konstanssal való szorzás. Legyen a egy szétosztandó input, $a_1, \ldots, a_n$ a hozzá tartozó titokrészek, $d \in \mathbb{F}$ egy konstans. Ekkor mindegyik p_i résztvevő lokálisan kiszámolja da_i-t. A $da_1, \ldots, da_n$ titokrészek pedig egyértelműen meghatározzák az $a + b$ titkot.

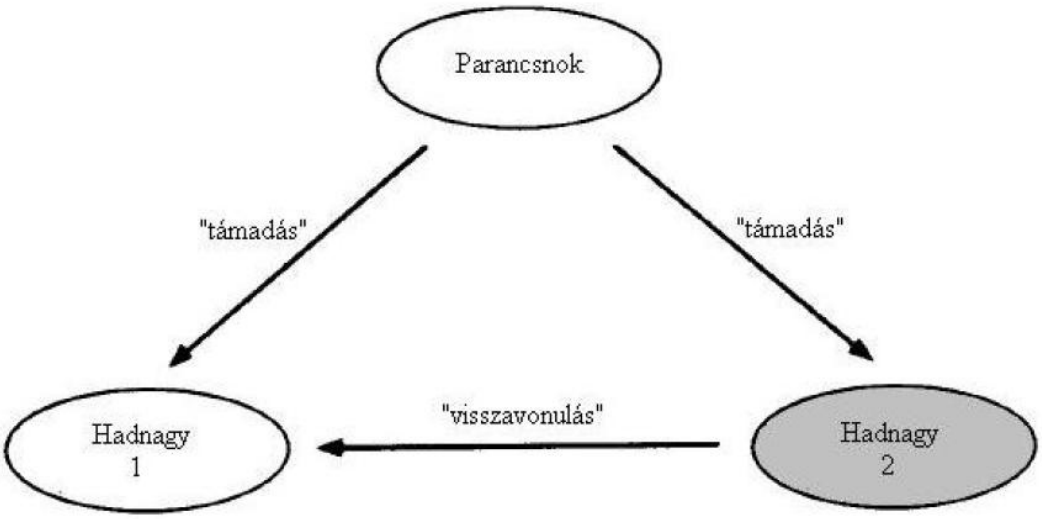
Szorzás. Legyen a és b két szétosztandó input, $a_1, \ldots, a_n$ és $b_1, \ldots, b_n$ rendre a hozzájuk tartozó titokrészek. Ekkor mindegyik p_i résztvevő lokálisan kiszámolja $c_i = a_i b_i$-t, majd p_i szétosztja c_i-t. Legyenek a c_i-hez tartozó titokrészek $c_{i1}, \ldots, c_{in}$, ahol a p_j résztvevő megkapja a c_{ij} titokrészt. Ezután a $c_{i1}, \ldots, c_{in}$ titokrészek egyértelműen meghatározzák c_j-t, a $c_1, \ldots, c_n$ titokrészek pedig egyértelműen meghatározzák a $c = ab$ titkot.

6. fejezet

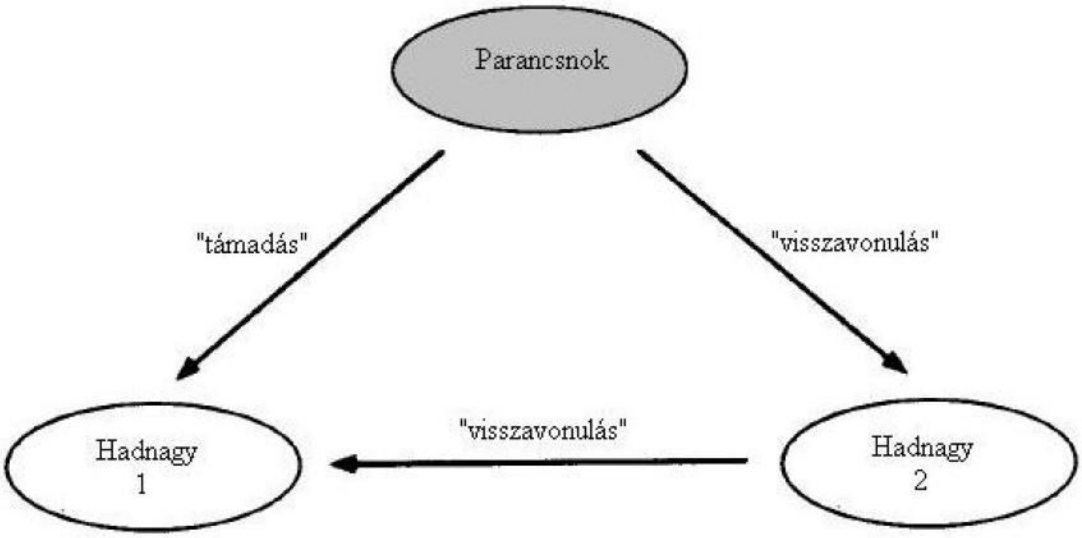
Üzenetszórás

6.1. A bizánci generálisok problémája

A *bizánci generálisok problémája* [7] a következő. A bizánci hadsereg két hadnagya parancsot vár a parancsnoktól. A parancsnok kétféle utasítást adhat: támadás, vagy visszavonulás. Miután a parancsnok kiadta az utasítást, a két hadnagy megbeszéli egymással, hogy mi a teendő. Probléma akkor van, ha a parancsnok, illetve a két hadnagy közül valaki nem tisztességes. A problémát a következő két ábra szemlélteti.



6.1.1. ábra. Az egyik hadnagy tisztességtelen



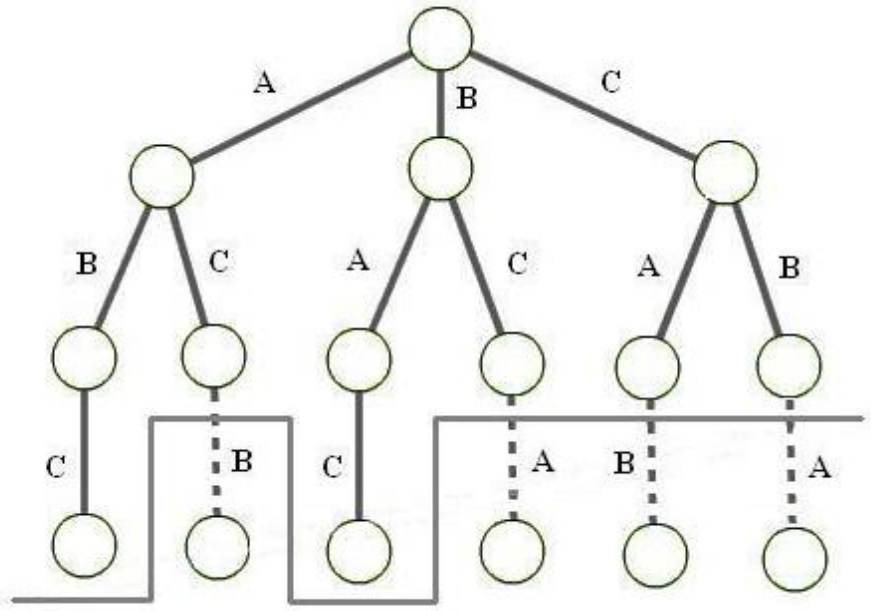
6.1.2. ábra. A parancsnok tisztességtelen

Tegyük fel, hogy Hadnagy 1 tisztességes. Ekkor ha Hadnagy 2 tisztességtelen, és a Parancsnok mindkét hadnagynak a "támadás" utasítást adta, akkor Hadnagy 2 a "visszavonulás" utasítást fogja továbbadni Hadnagy 1-nek. Ha a Parancsnok a tisztességtelen, és Hadnagy 1-nek a "támadás", Hadnagy 2-nek pedig a "visszavonulás utasítást adta", Hadnagy 2 ebben az esetben is a "visszavonulás" utasítást fogja továbbadni Hadnagy 1-nek. Hadnagy 1 mindkét esetben ugyanazt fogja tapasztalni, mégpedig hogy a Parancsnok azt mondta, hogy "támadás", Hadnagy 2 pedig azt, hogy "visszavonulás". Így Hadnagy 1 nem fogja tudni, hogy melyikük hazudott, azaz nem tudja kiszűrni a tisztességtelen résztvevőt. Ezt a problémát hivatott megoldani a *szimulált üzenetszórás*.

6.2. Szimulált üzenetszórás

Szimulált üzenetszórásnál [7] az üzenetszóráshoz hasonlóan az a célunk, hogy minden résztvevőnek eljuttassuk ugyanazt az üzenetet úgy, hogy a bizánci generálisok problémáját elkerüljük. Ez csak abban az esetben lehetséges, ha a kommunikációs csatorna szinkronizált. Az üzenetszórás szimulálását a következő algoritmussal valósíthatjuk meg.

Minden résztvevő konstruál egy fát, melynek éleit megcímkézi a résztvevők $\mathcal{P} = \{p_1, \ldots, p_n\}$ halmazának elemeivel úgy, hogy a gyökértől a levelekig a $p_1, \ldots, p_n$ elemek mindegyike pontosan egyszer forduljon elő az összes lehetséges sorrendben, azaz a fa első szintjén az élek $p_1, \ldots, p_n$ címkét kapnak, a második szinten a p_i él alatt már csak a $p_1, \ldots, p_{i-1}, p_{i+1}, \ldots, p_n$ címkék szerepelnek, és így tovább. Ezután a fának törlik néhány csúcsát és élét, mégpedig úgy, hogy a fa gyökerétől indulva haladnak lefelé az éleken addig, amíg egy olyan élt nem találnak, amit az eddigiekhez hozzávéve a kapott $\{p_{i_1}, \ldots, p_{i_l}\}$ élhalmaz már nem lehet tisztességtelen, azaz teljesül rá $\{p_{i_1}, \ldots, p_{i_l}\} \notin \Delta$. Az első ilyen él alatt lévő összes többi élt a hozzájuk tartozó csúcsokkal együtt törlik. Ezt megismétlik az összes lehetséges úttal. A következő ábrán egy egyszerű példával szemléltetjük a fa konstrukcióját.



6.2.1. ábra. A fa konstrukciója $\mathcal{P} = \{A, B, C\}$ és $\Delta = \{\{A, B\}, \{C\}\}$ esetén

A fa konsrukciója után a résztvevők kitöltik a fa csúcsait. A fa gyökeréhez mindegyikük hozzárendeli azt az értéket, amit ő kapott az osztótól. Ha az osztó tisztességes, ez az érték minden résztvevő esetében ugyanaz lesz. Első lépésben minden résztvevő elküldi ezt az értéket az összes többi résztvevőnek, majd kitölti a második szintet. Hasonlóan a k. lépésben minden résztvevő elküldi az összes többi résztvevőnek a k. szinthez tartozó értékeket, majd kitölti a $k + 1$. szintet. A kitöltés úgy történik, hogy minden résztvevő azt az értéket rendeli a v csúcshoz, ami tudomása szerint a p_i résztvevő fájában az u csúcshoz rendelt érték, ahol u és v a p_i-vel címkézett él két végpontja, és u a v csúcs szülője.

A csúcsok kitöltése után az összes csúcsokhoz rendelt értéket kitörlik a leveleken kívül. Ezután rekonstruálják a fát. Ha egy v csúcsnak az összes gyereke ki van már töltve, a v csúcsot is ki tudjuk tölteni. Legyenek a v csúcs és a gyerekei között futó élek a $p_{i_1}, \ldots, p_{i_l}$ résztvevőkkel megcímkézve. Felosztjuk a $\{p_{i_1}, \ldots, p_{i_l}\}$ halmazt az $A_1, \ldots, A_k$ halmazokra úgy, hogy azok a résztvevők kerülnek egy halmazba, akik ugyanazt az értéket mondták a v csúcs alatti szint kitöltésekor. Ha pontosan egy olyan j van, melyre $A_j \notin \Delta$, akkor azt az értéket rendeljük a v csúcshoz, amit az A_j-beli résztvevők mondtak. Ha nincs ilyen j, vagy több is van, akkor a v csúcshoz a $*$ karaktert rendeljük. Ezt az eljárást ismételjük a gyökérig.

Azt, hogy ez az eljárás valóban szimulálja az üzenetszórást, az alábbi tételben fogalmazzuk meg.

6.2.1. Tétel. *Minden tisztességes résztvevő a saját maga által konstruált fának a gyökerében ugyanazt az értéket kapja.*

A tétel bizonyítása előtt bebizonyítunk két lemmát.

6.2.1. Lemma. *Ha A egy tisztességes résztvevő, akkor minden résztvevő fájában az A-val címkézett él v végpontjában ugyanaz az érték szerepel, mint A fájában ugyanezen él egy szinttel feljebb lévő végpontjában, azaz a v csúcs szülőjében.*

Bizonyítás. Ha A a legalsó szinten van, akkor tisztességes volta miatt A mindenkinek ugyanazt az értéket mondja, így a fa konstrukciója miatt valóban igaz az állítás. Ha A nem a legalsó szinten van, tegyük fel hogy A egy x értéket rendelt a v csúcshoz. Ekkor a fa konstrukciója miatt az összes tisztességes résztvevő ugyanezt az x értéket rendelte a v csúcs A-val címkézett élén lévő gyerekéhez. Azok a résztvevők, akik a fa rekonstruálásánál nem az x értéket küldték a többi résztvevőnek , tisztességtelenek, azaz Δ-beli résztvevők. Akik nincsenek az A-val címkézett él alatti részfában, szintén Δ-beli résztvevők, így azon résztvevők, akik az x értéket küldték a többi résztvevőnek, nem lehetnek Δ-beli résztvevők a 3.2.2. Tételben szereplő feltétel miatt, miszerint $\mathcal{P} \notin \Delta \sqcup \Delta \sqcup \Delta$, azaz ezek a résztvevők nem lehetnek tisztességtelenek. Így a fa rekonstrukciós szabályai miatt valóban igaz a lemma.

□

6.2.2. Lemma. *Ha egy v csúcsból kiindulva minden ágon van tisztességes résztvevő, akkor a v csúcshoz minden résztvevő ugyanazt az értéket rendeli.*

Bizonyítás. Ha v minden gyereke tisztességes, a lemma triviálisan igaz. Ha v-nek van egy w tisztességtelen gyereke, akkor ez alatt a csúcs alatt minden ágon kell lennie tisztességes résztvevőnek. Ekkor a w csúcshoz minden résztvevő ugyanazt az értéket rendeli. Emiatt a v csúcs összes gyerekéhez minden résztvevő ugyanazt az értéket rendeli, így ez a 6.2.1. Lemma miatt a v csúcsra is teljesül.

□

A 6.2.1. Tétel bizonyítása. Azt szeretnénk megmutatni, hogy minden tisztességes résztvevő a saját maga által konstruált fának a gyökerében ugyanazt az értéket kapja. Valóban, a fa konstruálásakor addig tartottuk meg az éleket,

amíg találtunk egy olyat, amivel együtt a fa gyökerétől idáig vezető út már nem lehet Δ-ban. Emiatt a fa gyökeréből kiindulva minden ágon van tisztességes résztvevő, így a 6.2.2. Lemma miatt a fa gyökeréhez minden résztvevő ugyanazt az értéket rendeli.

□

Köszönetnyilvánítás

Szeretnék köszönetet mondani témavezetőmnek, Csirmaz Lászlónak a dolgozat írása során nyújtott segítségéért, türelméért, és hasznos tanácsaiért.

Irodalomjegyzék

[1] D. Beaver. Secure multi-party protocols and zero-knowledge proof systems tolerating a faulty minority. *Journal of Cryprology vol.4, no. 2*, 1991

[2] M. Ben-Or, S. Goldwasser, A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. *20th ACM Symposium on the Theory of Computing*, 1988

[3] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of the National Computer Conference*, 1979

[4] D. Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *Journal of Cryptology*, 1988

[5] D. Chaum, C. Crépeau, I. Damgård. Multi-party unconditionally secure protocols (extended abstract). *20th ACM Symposium on the Theory of Computing*, 1988

[6] O. Goldreich, S. Micali, A. Wigderson. How to play any mental game - a completeness theorem for protocols with honest majority. *19th ACM Symposium on the Theory of Computing*, 1987

[7] L. Lamport, R. Shostak, M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 1982

[8] U. Maurer. Secure Multi-Party Computation Made Simple. *Security in Communication Networks (SCN'02)*, 2003

[9] T. Rabin, M. Ben-Or. Verifiable secret-sharing and multiparty protocols with honest majority. *21st ACM Symposium on the Theory of Computing*, 1989

[10] A. Shamir. How to share a secret. *Communications of the ACM*, 1979

[11] A. C. Yao. Protocols for secure computations. *23rd IEEE Symposium on the Foundations of Computer Science*, 1982