

# Wykład 7. Bezwarunkowo Bezpieczne Protokoły MPC

Wykładowca: *Stefan Dziembowski*

Skryba: *Barbara Domagała i Karol Ostowski*

*Streszczenie.*

W niniejszym wykładzie omówione są dwa protokoły realizujące MPC. Opieramy się na [2] i [1]. Pierwszy z nich jest prosty (łatwy do zrozumienia, przez co wartościowy ze względów dydaktycznych), lecz mało efektywny obliczeniowo przez co niepraktyczny, drugi natomiast trochę bardziej skomplikowany lecz efektywniejszy. Oba są bezpieczne nawet przy założeniu że przeciwnik dysponuje nieograniczoną mocą obliczeniową.

## 1 Wstęp

Na dzisiejszym wykładzie poznamy dwa protokoły MPC bezpieczne względem przeciwnika o nieograniczonej mocy obliczeniowej.

Pierwszy z nich jest prosty i bardzo ogólny, ale ze względu na złożoność w większości przypadków niepraktyczny. Drugi natomiast jest bardziej skomplikowany, ale trochę bardziej praktyczny (wydajniejszy).

Oba protokoły działają według zasady opisanej na poprzednim wykładzie, to znaczy obliczaną funkcję reprezentujemy jako obwód arytmetyczny (w skład którego wchodzi bramki mnożenia i dodawania) nad jakimś skończonym ciałem  $F$ , a następnie:

1. każdy z graczy dzieli swoje wejście pomiędzy pozostałych
2. gracze obliczają wartość obwodu bramka-po-bramce
3. gracze rekonstruuują wynik.

## 2 Protokół [2]

### 2.1 Podstawowe definicje

Niech  $P$  będzie skończonym zbiorem (uczestników)

Zbiór  $\Pi \subseteq 2^P$  nazywany *strukturą* dla  $P$  jeśli jest on zamknięty na operację brania podzbiorów, czyli jeśli  $S \in \Pi$  oraz  $S' \subseteq S$  implikuje  $S' \in \Pi$ .

Łatwo widać że każda struktura jest jednoznacznie określona przez listę jej zbiorów maksymalnych (ze względu na porządek inkluzji). Przez *rozmiar*  $|\Pi|$  *struktury*  $\Pi$  rozumiemy długość tej listy.

Binarna operacja  $\sqcup$  na strukturach zdefiniowana jest następująco:

$$\Pi_1 \sqcup \Pi_2 := \{S_1 \cup S_2 : S_1 \in \Pi_1, S_2 \in \Pi_2\}$$

Przykładem struktur są *struktury progowe* zdefiniowane (dla ustalonego  $t$ ) jako:

$$\Pi := \{S \subseteq P : |S| \leq t\}$$

Zauważmy że w takim wypadku zbiorów maksymalnych będzie  $\binom{n}{t}$  czyli wykładnicza ilość względem  $n$  (pesymistycznie).

## 2.2 Podział sekretu

Na poprzednim wykładzie poznaliśmy schemat Shamira podziału sekretu. Działał on w ten sposób, że dowolna grupa co najwyżej  $t$  uczestników nie miała żadnej wiedzy na temat sekretu  $s$ , natomiast każda większa grupa mogła sekret zrekonstruować (schemat ten był oparty na własnościach wielomianów).

Aby rzecz uogólnić możemy ustalić dowolną *strukturę sekretności*  $\Sigma$  i wymagać, żeby:

- żadna grupa  $X \in \Sigma$  nie miała żadnej informacji na temat  $s$ .
- każda grupa  $X \notin \Sigma$  potrafiła obliczyć  $s$ .

(w literaturze z reguły definiuje się *strukturę dostępu* (ang.: *access structure*)  $\Gamma = 2^P - \Sigma$ ).

Jak widać schemat Shamira działa dla progowych struktur sekretności.

Istnieją wydajne schematy podziału sekretu dla rozmiatych struktur.

My będziemy używać następującego (bardzo niewydajnego ale prostego) schematu. Nasze rozwiązanie będzie w pewnym sensie dualne do schematu podziału sekretu

*k-z-k* który teraz krótko przypomnimy. *k-z-k* oznacza że mamy próg  $t = k - 1$  (gdzie  $k$  jest liczbą uczestników), innymi słowy potrzebna jest wiedza wszystkich graczy do zrekonstruowania sekretu. Aby podzielić sekret  $s$  pomiędzy  $k$  graczy wybieramy losowe  $s_1, \dots, s_k$ , takie że:

$$s_1 + \dots + s_k = s$$

$i$ -ty gracz otrzymuje  $s_i$  – to co gracz otrzymuje (w tym wypadku  $s_i$  nazywamy jego udziałem).

Teraz przystąpimy do opisu naszego rozwiązania. Niech  $T_1, \dots, T_k$  będą maksymalnymi zbiorami danej struktury sekretności  $\Sigma$ . Aby podzielić sekret  $s$ :

1. dzielimy  $s$  udziałem *k-z-k* (dzielimy w „pamięci” – to znaczy nie dajemy jeszcze niczego graczom). Niech  $s_1, \dots, s_k$  będą udziałami.
2. dla każdego  $i$  wysyłamy  $s_i$  do każdego z graczy w  $\overline{T_i}^1$ .

**Sekretność** Jeśli mamy zbiór  $T \in \Sigma$ , to jest on podzbiorem jakiegoś  $T_i$ . Zatem żaden z graczy w  $T$  nie zna  $s_i$ .

---

<sup>1</sup> $\overline{A}$  – uzupełnienie zbioru  $A$

Rekonstrukcja Jeśli mamy zbiór  $T \notin \Sigma$ , to dla dowolnego  $i$  musi on zawierać element, który nie jest w  $T_i$ , zatem ktoś w  $T_i$  zna  $s_i$ .

Zauważmy, że np. dla struktur progowych złożoność tego protokołu jest wykładnicza...

## 2.3 Uogólnione struktury przeciwnika

Przypomnijmy, że w przypadku obliczeń MPC zakładaliśmy, że przeciwnik ma prawo skorumpować maksymalnie  $t$  graczy.

Ogólniej, można po prostu wyspecyfikować *strukturę przeciwnika*  $\Delta$ , to znaczy rodzinę zbiorów które wolno przeciwnikowi skorumpować.

Jak wspominaliśmy na poprzednim wykładzie: bezwarunkowo bezpieczne obliczenia wielopodmiotowe są możliwe jeśli:

- $P \notin \Delta \sqcup \Delta$  („warunek  $\mathcal{Q}_2$ ”) — w przypadku przeciwnika pasywnego
- $P \notin \Delta \sqcup \Delta \sqcup \Delta$  („warunek  $\mathcal{Q}_3$ ”) — w przypadku przeciwnika aktywnego

( $P$  jest zbiorem wszystkich uczestników).

Jeszcze ogólniej możemy rozważać dwie struktury:  $\Delta \subseteq \Sigma \subseteq 2^P$  i powiedzieć, że  $(\Sigma, \Delta)$ -przeciwnik, to taki, który, może skormupować niektórych graczy  $A$  aktywnie<sup>2</sup>, a niektórych graczy  $B$  pasywnie<sup>3</sup>, tak długo póki:

$$A \in \Delta \text{ oraz } (A \cup B) \in \Sigma.$$

Oczywiście jeśli ktoś jest skorumpowany aktywnie to można go uznawać także za skorumpowanego pasywnie.

## 2.4 Główne rezultaty

Będziemy się teraz zajmować trzema różnymi protokołami, mianowicie:

- kanał rozgłaszający
- VSS – opisany dalej
- MPC

**Twierdzenie 1** *Symulacja kanału rozgłaszającego jest możliwa względem  $(\Sigma, \Delta)$ -przeciwnika wtedy i tylko wtedy gdy*

$$P \notin \Delta \sqcup \Delta \sqcup \Delta. \tag{1}$$

<sup>2</sup>przypominamy że korumpowanie aktywne polega na tym że przeciwnik ma pełną kontrolę nad zachowaniami gracza

<sup>3</sup>dla przypomnienia – korumpowanie pasywne oznacza że przeciwnik zna wszystkie udziały gracza, gracz jednak nadal postępuje zgodnie z protokołem

Tęgo twierdzenia nie będziemy dowodzić. Twierdzenie to dotyczy sytuacji gdy mamy zbiór graczy połączonych bezpiecznymi łączami i chcą oni symulować względnie bezpieczny kanał rozgłaszania. Jak widać w takim przypadku istotne są tylko korupcje aktywne ( $\Sigma$  nie występuje w warunku).

Następne twierdzenie dotyczy mocniejszego pojęcia, mianowicie schematu *Weryfikowalnego Podziału Sekretu* (ang.: *Verifiable Secret Sharing (VSS)*). Jest to schemat podziału sekretu bezpieczny względem  $(\Sigma, \Delta)$ -przeciwnika. Konkretnie, protokół składa się z dwóch faz:

Podział W czasie tej fazy dealer  $D$  dzieli swój sekret  $s$  pomiędzy graczy.

Rekonstrukcja W czasie tej fazy gracze rekonstruują  $s$ .

Przy czym spełnione są następujące warunki:

- Niezależnie od tego, czy dealer jest uczciwy czy nie: po wykonaniu podziału gracze zawsze zrekonstruują ten sam sekret (to znaczy, *dealer* jest zobowiązany do swojego sekretu niezależnie od zachowania przeciwnika).
- Jeśli dealer jest uczciwy w czasie podziału, to gracze zawsze zrekonstruują  $s$ .
- Tak długo jak dealer jest uczciwy, przeciwnik nie ma żadnej informacji na temat  $s$ .

**Twierdzenie 2** *Idealny<sup>4</sup> VSS bezpieczny względem  $(\Sigma, \Delta)$ -przeciwnika jest możliwy wtedy i tylko wtedy gdy:*

$$P \notin \Sigma \sqcup \Delta \sqcup \Delta. \quad (2)$$

**Twierdzenie 3** *Idealne MPC bezpieczne względem  $(\Sigma, \Delta)$ -przeciwnika jest możliwe wtedy i tylko wtedy gdy:*

$$P \notin \Sigma \sqcup \Sigma \sqcup \Delta. \quad (3)$$

Uwaga: Łatwo zauważyć, że:

- Istnienie MPC  $\Rightarrow$  istnienie VSS  $\Rightarrow$  istnienie kanału rozgłaszającego
- Oraz na odwrót: (3)  $\Rightarrow$  (2)  $\Rightarrow$  kanał rozgłaszający  $(\Delta \sqcup \Delta \sqcup \Delta)$ .

## 2.5 Implementacja VSS

Ustalmy jakieś struktury  $\Sigma$  i  $\Pi$ . Możemy założyć, że dysponujemy kanałem rozgłaszającym.

**Podział sekretu  $s$ :**

1. Dealer  $D$  dzieli  $s$  jak w Rozdziale 2.2 (z tym samym  $\Sigma$ ).

To oczywiście nie ujawnia nic przeciwnikowi (o ile  $D$  jest uczciwy).

---

<sup>4</sup>chodzi o przypadek z przeciwnikiem o nieskończonej mocy obliczeniowej, oraz o zerowe prawdopodobieństwo błędu

2. Teraz każdy z graczy otrzymał jakiś sekret, przy rekonstrukcji przekupieni mogą potencjalnie oszukiwać. Aby temu zapobiec wykonujemy następujące operacje: Dla każdego udziału  $s_i$  i każdej pary graczy  $(P_a, P_b)$ , takich że  $P_a, P_b \in \overline{T}_i$ <sup>5</sup> wykonaj:

- $P_a$  i  $P_b$  wysyłają do siebie nawzajem to co otrzymali jako  $s_i$ .
- Jeśli coś się nie zgadza, to gracz wysyła protest *za pomocą kanału do rozgłaszania*.

3.  $D$  ogłasza *za pomocą kanału do rozgłaszania*, wartości wszystkich  $s_i$  co do których były protesty.

Jeśli  $D$  tego nie zrobi, to uznawany jest za oszusta i gracze przyjmują jakiś ustalony z góry podział  $s$  (na przykład same zera). Gracze wiedzą które wartości powinien ogłosić  $D$ , ponieważ w pierwszym kroku wszystkie „oprotestowane”  $s_i$  zostały ogłoszone *za pomocą kanału do rozgłaszania*.

Zauważmy, że jeśli był protest w sprawie jakiegoś  $s_i$  to przeciwnik już to  $s_i$  zna (bo jeden z  $D, P_a, P_b$  musi być skorumpowany).

W wyniku tej procedury uczciwi gracze mogą być pewni, że ich wiedza jest spójna. Rekonstrukcja będzie podobna do tej w Rozdziale 2.2, trzeba tylko ustalić „prawdziwą” wartość każdego  $s_i$ .

W kroku pierwszym rekonstrukcji każdy z graczy wysyła do pozostałych „swoje” wartości  $s_i$ .

Ustalmy dowolnego gracza i dla każdego  $i$  rozważmy listę otrzymanych  $s_i$ . Potencjalnie nie wszystkie są równe (bo uszuści mogą kłamać). Każdy z graczy rekonstruuje  $s_i$  postępując następująco: Niech  $v_j$  dla  $j \in \overline{T}_i$  będzie wartością (odpowiadającą  $s_i$ ) otrzymaną od gracza  $p_j$ . Wybieramy tę (unikalną) wartość  $v$  dla której istnieje  $A \in \Delta$  dla której  $v_j = v$  dla wszystkich  $j \in \overline{T}_i - A$

## 2.6 Mnożenie

Dodawanie jest trywialne. Dla sekretów  $S$  i  $S'$  każdy z graczy ma odpowiadające im  $s_i$  i  $s'_i$  wystarczy że obliczy on lokalnie  $s_i + s'_i$  i będzie to udział odpowiadający sekretowi  $S + S'$ .

Mnożenie już takie proste niestety nie jest. Odbywa się według następującej procedury:

**Mnożenie – algorytm**<sup>6</sup>:

Sytuacja początkowa: dwie wartości  $S = \sum_{i=1}^k s_i$  i  $S' = \sum_{i=1}^k s'_i$  są dzielone przy użyciu VSS'a.

Chcemy osiągnąć:  $SS'$  jest dzielone przy pomocy VSS'a

1. Każdy z graczy  $p_m$  oblicza wszystkie te wartości  $s_i s'_j$  które jest w stanie obliczyć (czyli te dla których  $m \in \overline{T}_i \cap \overline{T}_j$  i dzieli je przy użyciu VSS'a.
2. Dla każdego  $(i, j)$ , niech  $(p_{m_1}, \dots, p_{m_r})$  będzie uporządkowaną listą graczy którzy obliczyli  $s_i s'_j$  w kroku 1 (gdzie  $r$  zależy od  $i$  i  $j$ ). Gracze (razem) obliczają<sup>7</sup> i

<sup>5</sup>gdzie  $T_1, \dots, T_k$  podobnie jak wcześniej oznaczają maksymalne podzbiory  $\Sigma$

<sup>6</sup>algorytm ten działa podobnie do algorytmu mnożenia przedstawionego na poprzednim wykładzie

<sup>7</sup>liczenie różnic wykonuje każdy z graczy lokalnie

otwierają  $r - 1$  różnic wartości dzielonych przez  $p_{m_1}$  i wartości dzielonych przez  $p_{m_i}$ , dla  $i = 2, \dots, r$ .

3. Jeśli wszystkie te otwarte wartości wynoszą 0, to udział od  $p_{m_1}$  jest używany jako udział odpowiadający  $s_i s'_j$ . W przeciwnym przypadku,  $s_i$  oraz  $s'_j$  są rekonstruowane<sup>8</sup> i udział odpowiadający  $s_i s'_j$  jest uzyskiwany przez  $p_1$  poprzez wzięcie  $s_i s'_j$  a wszystkich pozostałych graczy przez wzięcie udziału 0.
4. Gracze (lokalnie) obliczają sumę swoich udziałów odpowiadających wszystkim  $s_i s'_j$ , wynik jest udziałem odpowiadającym sekretowi  $SS'$

## 3 Protokół IC [1]

### 3.1 Model

Protokół z poprzedniego rozdziału jest bardzo niewydajny. Istnieje szereg rozmaitych znacznie wydajniejszych protokołów MPC. Przedstawimy tu protokół z pracy [1]. Krótkie podsumowanie właściwości przeciwnika w tym modelu:

- jest *progowy*: może skorumpować  $t < n/2$  graczy.
- jest aktywny
- jest adaptacyjny

Zakładamy istnienie kanału rozgłaszającego i pozwalamy na zaniedbywalne prawdopodobieństwo błędu.

### 3.2 Definicja IC

*Information checking* (IC) jest bezpieczną metodą autoryzacji danych. Schemat IC składa się z trzech protokołów:

- $Distr(D, INT, R, s)$ 
  - rozpoczynany przez dealera  $D$
  - $D$  wręcza sekret  $s$  do pośrednika  $INT$
  - $D$  przekazuje pewne dane pośrednikowi  $INT$  oraz odbiorcy  $R$
- $AuthVal(INT, R, s)$ 
  - rozpoczynany przez  $INT$
  - w tym protokole udział bierze tylko  $INT$  i  $R$
  - $INT$  zapewnia  $R$ , że  $s$  jest sekretem otrzymanym od dealera  $D$
- $RevealVal(INT, R, s')$

---

<sup>8</sup>nie ma tutaj niebezpieczeństwa ujawnienia tych wartości. Jeśli któraś z różnic była różna od zera to znaczy że któryś z posiadaczy tych udziałów jest przekupiony i przeciwnik już je zna tak czy inaczej

- rozpoczynany przez  $INT$
- w tym protokole udział bierze  $INT$  i  $R$
- $R$  dostaje  $s'$  od  $INT$ -a wraz z pewnymi danymi, następnie akceptuje  $s'$  lub odrzuca

### Właściwości IC

Schemat IC ma następujące właściwości:

#### Poprawność

1. Jeśli  $D$ ,  $INT$  i  $R$  nie są skorumpowani i  $s$  jest sekretem dealera, to  $R$  zaakceptuje  $s$  w fazie  $RevealVal$ .
2. Jeśli  $INT$  i  $R$  są uczciwi, to po fazach  $Distr$  i  $AuthVal$  pośrednik zna taką wartość  $s$ , którą  $R$  akceptuje.
3. Jeśli  $D$  i  $R$  nie są skorumpowani, to w fazie  $RevealVal$  gracz  $R$  odrzuci każdą wartość  $s'$  inną od  $s$ .

#### Sekretność

4. Informacja, którą  $D$  daje  $R$  w fazie  $Distr$ , przekazywana jest niezależnie od sekretu  $s$ . Dlatego, jeśli  $D$  i  $INT$  są uczciwi, i  $INT$  nie wykonał jeszcze fazy  $RevealVal$ , to  $R$  nie ma żadnej informacji na temat sekretu  $s$ .

**Definicja 4** Schemat IC to trójka protokołów ( $Distr$ ,  $AuthVal$ ,  $RevealVal$ ) spełniająca powyższe warunki 1 – 4.

## 3.3 Implementacja

Przedstawimy implementację protokołu, który spełnia definicję schematu IC. Idea tego protokołu polega na tym, że sekret  $s$  i informacja weryfikacyjna, leżą na prostej  $f$  oraz  $s = f(0)$ .

Dla uproszczenia przyjmujemy, że jeśli gracz spodziewa się coś dostać, a nie otrzymuje niczego, to przyjmuje, że otrzymał pewną wartość domyślną.

Wprowadzamy pomocniczą definicję.

**Definicja 5** Wektor  $(x, y, z)$  jest  $1 - a - consistent$  ( $1 - a - c$ ), jeśli istnieje prosta  $w$  taka, że:

- $w(0) = x$
- $w(1) = y$
- $w(a) = z$

**Protokół  $Distr(D, INT, R, s)$**

1.  $D$  losuje  $a$  różne od 0 i 1.

2.  $D$  losuje  $y, z$ , takie, że  $(s, y, z)$  jest  $1 - a - c$ .
3.  $D$  losuje  $s', y', z'$ , takie, że  $(s', y', z')$  jest  $1 - a - c$ .
4.  $D$  wysyła  $s, s', y, y'$  do  $INT$ .
5.  $D$  wysyła  $z, z'$  do  $R$ .

**Protokół  $AuthVal(INT, R, s)$**

1.  $INT$  losuje  $d$ .
2.  $INT$  broadcastuje  $d, s' + ds, y' + dy$ .
3. Jeśli  $D$  zaobserwuje, że powyższe wartości są niepoprawne, to:
  - (a)  $D$  broadcastuje  $s, y$ .
  - (b) Koniec protokołu -  $INT$  skorumpowany.
  - (c)  $R$  oblicza sobie  $(s, y, z = f(a))$ , gdzie  $f$  obliczone na podstawie  $s$  i  $y$ .
4.  $R$  sprawdza, czy  $(s' + ds, y' + dy, z' + dz)$  jest  $1 - a - c$ .
5.  $R$  broadcastuje akceptację lub odrzucenie.
6. Jeśli  $D$  spostrzeże, że  $R$  zachowuje się niepoprawnie, to:
  - (a)  $D$  broadcastuje  $z, a$ .
  - (b) Wszyscy widzą, że  $R$  skorumpowany.
  - (c) Koniec protokołu.
  - (d)  $INT$  oblicza sobie  $(s, y, z)$ .
7. Jeśli  $R$  odrzucił i  $D$  nie dał na broadcast sprzeciwu, to  $D$  broadcastuje  $s$  i  $y$ .
8.  $R$  oblicza  $(s, y, z)$ .

**Protokół  $RevealVal(INT, R, s)$**

1.  $INT$  broadcastuje  $(s, y)$ .
2.  $R$  sprawdza, czy  $(s, y, z)$  jest  $1 - a - c$ .
3.  $R$  broadcastuje akceptację lub odrzucenie.

### 3.4 Dowód poprawności

**Twierdzenie 6** *Powyższa implementacja schematu ( $Distr, AuthVal, RevealVal$ ) jest schematem IC w sensie definicji 4.*

**Dowód:** Pokażemy, że każda z własności 1 – 4 jest spełniona.

1. Jeśli każdy gracz jest uczciwy, to  $R$  zaakceptuje  $s$ , a  $D$  nie wyśle nic przez kanał rozgłoszeniowy.
2. Jeśli  $D$  rozgłasza  $s$ ,  $y$  lub  $z$ ,  $a$  - własność jest trywialna. Wystarczy pokazać, że jeśli  $D$  na początku wyśle niepoprawne  $(s, y, z)$ , to  $R$  nie wyrazi akceptacji. Dla  $e$  różnego od  $d$ ,  $(s' + ds, y' + dy, z' + dz)$  oraz  $(s' + es, y' + ey, z' + ez)$  są  $1 - a - c$ . Wtedy również  $(s, y, z)$ , jako różnica i przeskalowanie, są  $1 - a - c$ . Ponieważ  $d$  było wybrane losowo, to dla  $(s, y, z)$ , które nie jest  $1 - a - c$ ,  $R$  wyrazi akceptację z prawdopodobieństwem co najwyżej  $1/|K|$ , gdzie  $K$  jest rozmiarem zbioru z którego losowaliśmy  $d$ .
3. Należy pokazać, że  $INT$  nie zna  $a$ . Pokażemy, że to zachodzi nawet jeśli  $D$  używa tego samego  $a$  przy każdym wołaniu tego protokołu. Zauważmy, że  $INT$  nie dowiaduje się żadnych informacji o  $a$  w fazach  $Distr$  i  $AuthVal$  - to co dostaje w fazie  $Distr$  jest niezależne od  $a$ . W fazie  $AuthVal$ , jeśli wysyła poprawne dane, to wie, że będą zaakceptowane. Jeśli wysyła niepoprawne, to wie, że  $D$  będzie się sprzeciwiał. Ma to miejsce nawet gdy wołamy  $AuthVal$  wiele razy. Jedyne, co wie  $INT$  o  $a$ , to, że jest ono różne od 0 i 1, oraz, że każda pozostała wartość jest tak samo prawdopodobna.  
Rozważmy teraz pozycję pośrednika  $INT$  tuż przed ujawnieniem pierwszej wartości  $s$ . Jeśli wysyła poprawne  $s$  i  $y$ , lub zmienia jedną z tych wartości, to z góry wie, jakie będzie zachowanie gracza  $R$  i niczego nowego się nie dowiaduje. Jeśli wysyła  $s'$  i  $y'$ , różne od  $s$  i  $y$ , to  $R$  wyrazi akceptację, jeśli  $(s', y', z)$  jest  $1 - a - c$ . Ponieważ  $(s, y, z)$  jest  $1 - a - c$ , wtedy również  $(s - s', y - y', 0)$  jest  $1 - a - c$ . Aby  $R$  zaakceptował niepoprawną wartość,  $INT$  musi więc odgadnąć  $a$ . Może to zrobić z prawdopodobieństwem  $1/(|K| - 2)$ , gdzie  $K$  jest rozmiarem zbioru, z którego losowaliśmy  $a$ . Więc jeśli  $R$  nie wyrazi akceptacji, to  $INT$  wie, że nie odgadł  $a$  i może je wykluczyć przy następnej próbie.  
Po  $l$  próbach,  $INT$  wykluczy  $|K| - l - 2$  kandydatów na  $a$ . Niepoprawna wartość zostanie zaakceptowana z prawdopodobieństwem  $l/|K| - l - 2$ . Jeśli  $l$  jest liniowe w stosunku do ilości graczy  $n$ , to prawdopodobieństwo błędu jest co najwyżej  $2^{-k+O(\log n)}$ .
4. Jeśli  $D$  i  $INT$  są uczciwi, a  $R$  skorumpowany, to musimy pokazać, że  $R$  nie pozna sekretu  $s$  przed fazą  $RevealVal$ . Zauważmy, że w fazie  $AuthVal$ ,  $R$  poznaje  $z$ ,  $z'$ ,  $d$ ,  $s' + ds$ ,  $y' + dy$ . Dodatkowo, jeśli  $D$  i  $INT$  są uczciwi,  $R$  wie, że  $(s' + ds, y' + dy, z' + dz)$  jest  $1 - a - c$  - może więc policzyć  $y' + dy$ , jest to więc dla niego informacja nadmiarowa i może zostać pominięta. Wiemy jednak, że  $z$ ,  $z'$ ,  $d$ ,  $s' + ds$  są niezależne od  $s$ .

□

## Literatura

- [1] Ronald Cramer, Ivan Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. Efficient multiparty computations secure against an adaptive adversary. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 311–326. Springer-Verlag, May 1999.
- [2] Ueli Maurer. Secure multi-party computation made simple. In G. Persiano, editor, *Third Conference on Security in Communication Networks (SCN'02)*, volume 2576 of *Lecture Notes in Computer Science*, pages 14–28. Springer-Verlag, 2003.