

Wykład 6. Obliczenia Wielopodmiotowe

Wykładowca: *Stefan Dziembowski*

Skryba: *Krzysztof Juchimiuk i Tomasz Jadachowski*

Streszczenie.

Na wykładzie zajmować będziemy się wielopodmiotowymi obliczeniami. Najpierw zdefiniujemy bezpieczeństwo takich obliczeń, a następnie pokażemy pewne proste implementacje protokołów. Oto ogólny plan wykładu:

Plan

1. Przykłady problemów, którymi będziemy się zajmować.
2. Definicje i modele.
3. Podstawowe wyniki.
4. Implementacje.

Pierwsze prace

Obliczenia Wielopodmiotowe (MPC)

Poczynając od:

Yao Protocols for secure computations (1992).

Goldreich, Micall, Widgerson How to play any mental game – a completeness theorem for protocols with honest majority (1986).

1 Przykłady problemów, którymi będziemy się zajmować.

Przykład 1: problem z miłością Alicja i Bob

- chcą sprawdzić czy kochają się nawzajem
- trzymają swoje uczucia w największym sekrecie.

Przykład 1: bardziej formalnie

Alicja ma prywatne wejście a i Bob ma prywatne wejście b :

$$a := \begin{cases} 1 & \text{jeśli Alicja kocha Boba} \\ 0 & \text{wpp} \end{cases} \quad b := \begin{cases} 1 & \text{jeśli Bob kocha Alicję} \\ 0 & \text{wpp} \end{cases} \quad (1)$$

Celem Alicji i Boba jest obliczanie funkcji $f(a, b) := a \wedge b$ dopóki ich wejścia są trzymane w największym sekrecie:

- jeśli $f(a, b) = 1$ to $a = b = 1$, więc prywatność nie jest możliwa (oboje się kochają).
- Ale: jeśli $a = 0$ to $f(a, 0) = f(a, 1) = 0$, więc Alicja nie wie czy Bob ją kocha czy nie!

Można przyjąć, że Alicja i Bob mają jakieś napisy. Trzeba sprawdzić czy te napisy są równe. Można mieć listy i sprawdzać pasujące do siebie pary.

Przykład 2: rzucanie monetą

Alicja i Bob rzucają monetą.

Bardziej precyzyjnie:

- Alicja i Bob zaczynają bez żadnych wejść
- Chcą uzyskać bit r :

$$r := \begin{cases} 0 & \text{z prawdopodobieństwem } \frac{1}{2} \\ 1 & \text{z prawdopodobieństwem } \frac{1}{2} \end{cases} \quad (2)$$

Funkcja nie bierze wejścia i zwraca bit. Jak to zaimplementować za pomocą zobowiązań bitowych?

Przykład: Głosowanie elektroniczne

Przypuśćmy, że mamy grupę osób $\{P_1, \dots, P_n\}$. Każda osoba P_i ma prywatne wejście

$$x_i := \begin{cases} 1 & \text{jeśli głosowała na „tak”} \\ 0 & \text{wpp.} \end{cases} \quad (3)$$

Wszystkie osoby, przy zachowaniu prywatności głosowania chcą obliczyć funkcję

$$f(x_1, \dots, x_n) := x_1 + \dots + x_n.$$

2 Definicje i modele.

2.1 Cel

Rozważamy grupę osób P_1, \dots, P_n i ustaloną (publicznie znaną) funkcję

$$f : D_1 \times \dots \times D_n \rightarrow C_1 \times \dots \times C_n$$

Każda osoba P_i trzyma prywatne wejście $x_i \in D_i$.

Niech

$$(y_1, \dots, y_n) := f(x_1, \dots, x_n)$$

Cel konstrukcja protokołu Π_f takiego, że w wyniku każdy gracz P_i pozna wartość y_i

Co więcej, protokół powinien być bezpieczny!

(Rozważamy protokół odporny na ataki, gdzie gracze mogą zajmować się wymianą informacji)

2.2 Co to jest bezpieczeństwo?

W dalszej części zechcemy sformalizować pojęcie *bezpieczeństwa*. Jak się okazuje pojęcie to może mieć różne znaczenia. Wszystko zależy od modelu...

W ogólności założymy, że *protokół* jest atakowany przez *przeciwnika*

2.3 Przeciwnik

Część z graczy P_1, \dots, P_n może oszukiwać, tzn. mogą próbować ominąć protokół. Takich graczy nazwiemy *skorumpowani*. Założymy, że tacy gracze mogą działać grupowo.

Aby to zamodelować założymy, że istnieją pojedynczy przeciwnicy A , którzy mogą skorumpować pewnych graczy. Przeciwnik A jest modelowany jako probabilistyczna Maszyna Turing'a.

Teraz będziemy mogli się przekonać co może zrobić przeciwnik.

2.4 Przeciwnik aktywny/pasywny

Jeśli gracz P_i został raz skorumpowany, to przeciwnik przejmuje nad nim pełną kontrolę.

przeciwnik pasywny Ma dostęp do wszystkich wewnętrznych danych gracza P_i i wszystkie wiadomości przychodzących do P_i .

przeciwnik aktywny Ma dodatkowo pełną kontrolę akcji wykonywanych przez P_i . (np. może uruchamiać jakieś programy)

Model pasywny jest mniej praktyczny. Jakkolwiek jest często rozważany w literaturze z powodów: **metodycznych** Protokół opierający się na ochronie pasywnej często może być zastępowany jakimś modelem aktywnym.

praktycznych W wielu aplikacjach pasywna ochrona jest lepsza niż żadna.

2.5 Moc przeciwnika

Zależnie od konfiguracji mamy następujące typy przeciwników:

[model obliczeniowy] ograniczona moc obliczeniowa (zwykle przez zrandomizowany czas wielomianowy). Protokoły są bezpieczne o ile zadanie jest trudne obliczeniowo.

[model teorio-informacyjny] o nieograniczonej mocy obliczeniowej (bezpieczeństwo teorio-informacyjne)

[model niestandardowy] nieograniczona moc obliczeniowa, ale za to ma inne ograniczenia (hałaśliwe kanały, pamięć ograniczona kryptograficznie)

W przypadku modelu obliczeniowego potrzebujemy ochrony opartej na kilku nieudowodnialnych założeniach (np. istnienie jednokierunkowej permutacji)

W innych przypadkach istnieją dowody bez jakichkolwiek nieudowodnionych założeń.

2.6 Możliwe korupcje

W większości przypadków musimy założyć, że przeciwnik nie może skorumpować dowolnie wielu graczy.

Często, w celu zapewnienia protokołom bezpieczeństwa zakładamy, że przeciwnik jest w stanie skorumpować co najwyżej $t < n$ graczy. Nazywa się to *atak progowy* (*threshold adversary*)

Bardziej ogólnie: możemy wyspecyfikować strukturę przeciwnika następująco:

Niech F będzie rodziną podzbiorów zbioru

$$\{P_1, \dots, P_n\}.$$

Przeciwnik może skorumpować zbiór X tylko jeśli $X \in F$. Przykładowo, jeśli mamy grupę 10 polityków $\{P_1, \dots, P_{10}\}$ i 10 naukowców $\{S_1, \dots, S_{10}\}$ i niech $F = \{\{P_1, P_{i_2}, P_{i_3}, P_{i_4}\}, \{S_{i_1}\}\}$ to przeciwnik może skorumpować co najwyżej czterech polityków i jednego naukowca.

2.7 Przystosowujący się przeciwnik

Są przynajmniej dwie opcje do rozważenia.

[nieprzystosowujący się przeciwnik] Przeciwnik musi zdecydować kogo skorumpuje zanim rozpocznie się wykonanie protokołu.

[przystosowujący się przeciwnik] Przeciwnik korumpuje graczy jeden po drugim podczas wykonania protokołu. Na początku nie korumpuje nikogo, potem jednego (popatrzy co się dzieje, podsłucha), potem kolejnego itd. Czasami przeciwnik może „odkorumpować” gracza, aby skorumpować kogoś innego.

2.8 Kanały komunikacyjne

Zwykle zakładamy, że istnieją połączenia pomiędzy parami graczy. Zależnie od modelu zakładamy, że

[model obliczeniowy] przeciwnik może podsłuchiwać komunikację międzygrupową.

[model teorio-informacyjny] kanały są bezpieczne

Czasami również przyjmujemy, że istnieje rozgłaszanie kanałowe (dostępne dla każdego gracza)

Co więcej sieć może być synchroniczna, albo nie (zwykle zakładamy, że jest).

Przeciwnik zna wejścia i wyjścia i nic więcej.

2.9 Bezpieczeństwo – nieformalnie

Nieformalnie mówiąc bezpieczeństwo wymaga spełnienia dwóch wymagań. **Prywatność** Przeciwnik nie może dowiedzieć się niczego o wejściach uczciwych graczy (nie jest to wymagane jeśli chodzi o wejścia/wyjścia skorumpowanych graczy).

Poprawność Przeciwnik nie ma wpływu na wyjścia uczciwych graczy (inaczej niż przez zastąpienie wejść skorumpowanych graczy)

Nota: W przypadku gdy $f(a, b) = a \wedge b$ jeśli $a = 1$ to Alicja dostaje pełną informację o b .

Skorumpowana Alicja może zawsze kłamać (i mówić, że kocha Boba pomimo, że tak nie jest).

2.10 Model idealny

Aby zrozumieć zjawisko wprowadzimy *model idealny*.

W idealnym modelu zakładamy istnienie „niekorumpowalnej” zaufanej grupy T . Obliczenia są wykonywane przez T :

- gracze wysyłają swoje wejścia x_1, \dots, x_n do T
- T oblicza $(y_1, \dots, y_2) = f(x_1, \dots, x_n)$
- T wysyła każde y_i do P_i . Każde P_i przetwarza y_i

Jeżeli gracz jest skorumpowany, to może podmienić wejścia na coś innego, ale nic więcej nie może zrobić. Protokół Π bezpiecznie oblicza f , jeśli wszystko to co przeciwnik może osiągnąć atakując Π , może zostać zasymulowane w modelu idealnym.

2.11 Definicja Bezpieczeństwa

Powiemy, że protokół Π bezpiecznie oblicza f , jeśli (nieformalnie):
cokolwiek przeciwnik może osiągnąć atakując Π może to również osiągnąć w modelu idealnym.
Aby być bardziej formalnym możemy potrzebować definicji pojęć *symulatora* i *środowiska*.
Aby zapoznać się ze szczegółami zobacz [?].

2.12 Bezpieczeństwo doskonałe/niedoskonałe

Jeli pracujemy w teorio-informacyjnym modelu możemy również

- wymagać doskonałego bezpieczeństwa – tzn. bezpieczeństwa utrzymywanego z prawdopodobieństwem 1, lub
- zgodzić się na niedoskonałe bezpieczeństwo – tzn. istnieje *pomijalne* prawdopodobieństwo błędu.
Precyzując, Dla danej partii wejść i parametru bezpieczeństwa k ,
oraz:
dla dowolnego c prawdopodobieństwo błędu jest mniejsze niż k^{-c} ,
przy dostatecznie dużym k .

2.13 Funkcje losowe

Funkcja f może być

losowa

Żeby to pokazać założymy, że funkcja f pobiera ekstra wejście r wybrane jednoznacznie jako losowe z pewnej dziedziny R . Tzn. Funkcja f jest typu:

$$f : D_1 \times \dots \times D_n \times R \rightarrow C_1 \times \dots \times C_n$$

Przykład. Protokół rzucania monetą:

$$f : \{\#\} \times \{\#\} \times \{0, 1\} \rightarrow \{0, 1\} \times \{0, 1\}$$

gdzie $f(\#, \#, r) = (r, r)$

2.14 Systemy reaktywne

Odwołanie w którym zaufana grupa T oblicza funkcję. Możemy rozważać bardziej ogólne modele w których T wykonuje kilka procesów on-line. Obliczenie jest wykonywane fazowo.

Przykład

Faza 1 jeden z graczy przechowuje sekret

Faza 2 sekret jest ujawniany graczom.

3 Podstawowe wyniki.

3.1 Fundamentalne pytanie

Dla przeciwnika (liczącego efektywnie)

$$f : D_1 \times \dots \times D_n \times \rightarrow C_1 \times \dots \times C_n$$

Czy istnieje efektywny wielopodmiotowy protokół bezpieczeństwa obliczający f ?

Odpowiedź zależy od modelu ...

3.2 Model obliczeniowy

Czy możemy skonstruować protokół dla dowolnego f ?

liczba t skorumpowanych graczy	pasywny	aktywny
$t < n/2$	Tak	Tak
$t \geq n/2$	Tak	Tak

jeśli $t \geq n/2$ i przeciwnik jest aktywny to może przerwać wykonanie w dowolnym momencie:-).

1. Alicja losuje b i zobowiązuje się do b
2. Bob losuje b' i zobowiązuje się do b'
3. Alicja otwiera b

Weryfikacją jest $b' \oplus b$. Skorumpowana Alicja może zastopować wykonanie protokołu.

3.3 Model teorio-informacyjny

Model bez kanału rozgłoszeniowego:

liczba t skorumpowanych graczy	pasywny	aktywny
$t < n/3$	TAK	TAK
$n/3 \leq t < n/2$	TAK	NIE
$n/2 \leq t$	NIE	NIE

Jeśli $n/3 \leq t \leq n/2$ i przeciwnik jest aktywny to można skonstruować protokół bezpieczeństwa zakładający istnienie kanału rozgłoszeniowego.

3.4 Uogólnienie

Powyższe wyniki mogą być uogólnione na dowolną strukturę przeciwnika następująco. Niech P będzie wejściowym zbiorem graczy. Przez F oznaczamy strukturę przeciwnika.

Wymaganie „, $t < n/p''$ może być przetłumaczone na:

Q_2 Dla każdych $A, B \in F$ mamy $A \cup B \neq P$.

Wymaganie „, $t < n/3''$ może być przetłumaczone na: Q_3 Dla każdych $A, B, C \in F$ mamy $A \cup B \cup C \neq P$. Dokładny opis jest w [?].

3.5 Dlaczego potrzebujemy większości

Zamierzamy udowodnić, że w modelu teorio-informacyjnym nie jest możliwe bezpieczne obliczenie

$$f(a, b) = a \wedge b$$

Przypuśćmy, że mamy protokół Π dla Alicji i Boba, który bezpiecznie oblicza f . Dla uproszczenia, załóżmy, że działa on bezbłędnie.

3.6 Transkrypt

	Alicja	Bob
wejście	a	b
losowe wejście	r_A	r_B
	$m_{A1} \rightarrow$	
	$\leftarrow m_{B1}$	
	\vdots	
wyjście	y	y

Definiujemy transkrypt wykonania protokołu Π jako $T := (m_{A1}, m_{B1}, m_{A2}, \dots, y)$.

3.7 Zakończenie dowodu

Def. Powiemy, że transkrypt T jest zgodny i $a = x$ jeśli da wynik z wykonania P_i z $a = x$. Przypuśćmy, że $a = 0$. W tym przypadku $y = f(a, b) = 0$. Dlatego

- Jeśli $b = 0$ to Bob nie powinien posiadać informacji o a .
Zatem T musi być zgodny równocześnie z $a = 0$ i $a = 1$
- Jeśli $b = 1$ to T musi być zgodny tylko z $a = 1$
(w innym przypadku $y = 0$ przy $a = b = 1$).

Alicja może sprawdzić, który to przypadek, od kiedy ma nieskończoną moc obliczeniową. Tak więc, ona może poznać b .

3.8 Kilka uwag na temat dowodu

Dowód działa przy założeniu, że Π działa bezbłędnie.

Jakkolwiek, jest możliwe uogólnienie dla przypadku niedoskonałego bezpieczeństwa. To można uogólnić dla przypadku wielu graczy:

wystarczy założyć, że istnieją dwa zbiory A i B w strukturze przeciwnika, takie że $A \cup B = P$

4 Implementacje

4.1 Plan

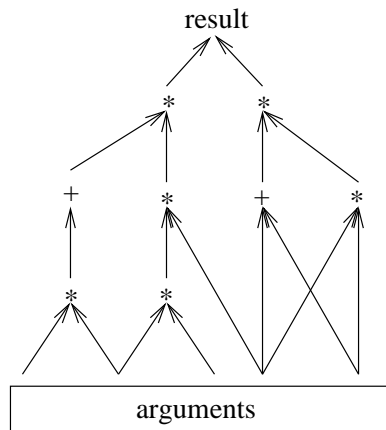
Pokażemy kilka przykładów implementacji protokołów.

Zaczynamy od przypadku 2 grup graczy i następnie przechodzimy do przypadku wielu grup graczy. Dla uproszczenia zakładamy że:

- funkcja f jest deterministyczna i
- wyjścia wszystkich graczy są takie same.

Zakładamy, że wejścia wszystkich grup pochodzą z pewnej skończonej dziedziny F . Funkcja f jest reprezentowana jako wartości logiczne nad F .

4.2 Przykład – obwód arytmetyczny



5 Podział sekretu (secret sharing)

Przedstawimy teraz protokół umożliwiający podział sekretu wśród grupy graczy, w taki sposób by tylko dostatecznie duży ich podzbiór pozwalał na odtworzenie sekretu. Dokładniej każdy gracz P_j dostaje pewną wiadomość s_j (udział) taką że:

- Mały podzbiór graczy nie ma szans dowiedzieć się nic o sekrecie s_j
- Duży podzbiór graczy może zrekonstruować sekret s_j .

6 Ogólny paradygmat obliczeń wielopodmiotowych

Przedstawimy teraz ogólny schemat działania protokołów wielopodmiotowych obliczeń. Możemy wyróżnić trzy istotne fazy działania:

- Każdy gracz P_i dzieli swój sekret, którym są jego dane wejściowe s_i .
- Gracze obliczają obwód arytmetyczny bramka po bramce, zachowując poniższą regułę:
 - Po obliczeniu każdej bramki, gracze mają poprawne udziały potrzebne do obliczenia rezultatu
- Na koniec gracze rekonstruują rezultat.

Jak widać idea jest prosta, wystarczy teraz że pokażemy:

- jak dzielić i rekonstruować sekret
- jak dodawać rozdzielone sekrety (to jest zazwyczaj proste)
- jak mnożyć rozdzielone sekrety (to jest trudniejsze!)

7 Model obliczeń

Bardzo często konstruując protokoły wielopodmiotowe robi się to w dwóch fazach:

- Najpierw konstruujemy protokół pasywnie bezpieczny.
- Następnie modyfikujemy go by był też bezpieczny dla aktywnego wroga.

W dalszej części wykładu, dla przejrzystości, będziemy rozpatrywać tylko pasywne dwustronne (two-party) protokoły.

8 Oblivious transfer

8.1 Nota historyczna

Bardzo ważną rolę w obliczeniach wielopodmiotowych mają protokoły oblivious transfer. Pierwsze protokoły oblivious transfer przedstawił M. O. Rabin w pracy *How to exchange secrets by Oblivious*. Jego definicja była inna od przedstawionej poniżej aczkolwiek równoważna.

8.2 Definicja

Przedstawimy definicję podstawowego protokołu oblivious transfer, mianowicie:

1-out-of- k oblivious transfer

jest to protokół wykonywany pomiędzy dwoma graczami: nadawcą S (zwanym dalej nadawcaem) i odbiorcą R (zwanym dalej odbiorcaem).

Danymi wejściowymi nadawcy jest sekwencja x_1, \dots, x_k elementów z jakiejś ustalonej dziedziny. Natomiast danymi wejściowymi odbiorcy jest liczba $i \in \{1, \dots, k\}$. Danymi wyjściowymi nadawcy jest zbiór pusty, a odbiorcy jest element x_i .

Dane wyjściowe odbiorcy oznaczają będziemy przez $\text{OT}_1^k(x_1, \dots, x_k, i) = x_i$.

8.3 Przykład: obliczenie koniunkcji

Używając protokołu oblivious transfer możemy skonstruować protokół obliczający $f(a, b) = a \wedge b$. Mianowicie:

- Alicja jest nadawcaem z danymi wejściowymi, parą bitów $(0, a)$. Bob zaś działa jako odbiorca z danymi wejściowymi, bitem $i = b$.
- Bob wysyła do Alicji bit który otrzyma i oboje uznają to jako wynik.

łatwo zauważyć że ten protokół działa, ponieważ:

$$\text{OT}((0, a), b) = a \wedge b.$$

8.4 Implementacja

Niech (x_1, \dots, x_k) — dane wejściowe nadawcy, i — dane wejściowe odbiorcy.

- nadawca generuje parę kluczy(RSA) prywatny i publiczny (e, d) i wysyła e do odbiorcy.
- odbiorca przygotowuje k kryptogramów c_1, \dots, c_k .
 - kryptogram $c_i = E(e, m)$, gdzie m jest wybraną losowo przez odbiorcy wiadomością
 - pozostałe kryptogramy to losowe ciągi

odbiorca wysyła c_1, \dots, c_k do nadawcy.

- nadawca odszyfrowuje wszystkie kryptogramy. Niech m_1, \dots, m_k będą najmniej znaczącymi bitami(LSB) odszyfrowanych wiadomości. nadawca wysyła do odbiorcy $m_1 \oplus x_1, \dots, m_k \oplus x_k$.
- odbiorca odszyfrowuje m_i .

Oczywiście zamiast RSA można wziąć dowolną funkcję jednokierunkową z zapadką.

9 2-podmiotowe obliczenia

Pokażemy teraz jak zaimplementować 2-podmiotowe obliczenia. Przyjmijmy za ciało $F = Z_2$, a graczami będą Alicja i Bob.

9.1 Podział sekretu

Użyjemy bardzo prostego mechanizmu podziału sekretu. Mianowicie, by podzielić sekretny bit s , tworzymy losową parę bitów (s_A, s_B) taką że $s_A \oplus s_B = s$. Wysyłamy następnie s_A do Alicji i s_B do Boba. Rekonstrukcja jest trywialna.

9.2 Dodawanie

Dodawanie jest także proste, jeśli

- x jest podzielone na (x_A, x_B) i
- y jest podzielone na (y_A, y_B)

to: $(x_A \oplus x_B, y_A \oplus y_B)$ przedstawia podział $x \oplus y$.

9.3 Mnożenie

Niech x będzie podzielone na (x_A, x_B) , a y na (y_A, y_B) . Chcemy otrzymać podział $z = xy$. Robimy to w dwóch krokach.

1. Najpierw Alicja wybiera losowy bit c , który będzie jej częścią(udziałem). Bob musi zaś obliczyć $c \oplus z$.

2. Następnie Alicja i Bob wykonują protokół 1-out-of-4 oblivious transfer. Alicja jest nadawcą z danymi wejściowymi

$$\left(\overbrace{c + x_A y_A}^1, \overbrace{c + x_A (y_A + 1)}^2, \overbrace{c + (x_A + 1) y_A}^3, \overbrace{c + (x_A + 1) (y_A + 1)}^4 \right).$$

Bob to odbiorca z danymi wejściowymi

$$(1 + 2x_B + y_B) = \begin{cases} 1 & \text{if } (x_B, y_B) = (0, 0) \\ 2 & \text{if } (x_B, y_B) = (0, 1) \\ 3 & \text{if } (x_B, y_B) = (1, 0) \\ 4 & \text{if } (x_B, y_B) = (1, 1) \end{cases}$$

Z powyższych równań wynika, że dane wyjściowe Boba to: $c + (x_A + x_B)(y_A + y_B) = c + xy$.

Oczywiście można skonstruować podobną metodę dla operacji na wielu bitach.

9.4 Aktywne bezpieczeństwo

Nie będziemy precyzyjnie opisywać jak modyfikować podany wyżej protokół, lecz przedstawimy samą ideę. W celu ulepszenia protokołu, by był także aktywnie bezpieczny, musimy użyć mechanizmu zobowiązania się i protokołów z wiedzą zerową. Dokładniej gracze muszą zobowiązać się do swoich danych wejściowych, a także dowieść za pomocą protokołu wiedzy zerowej, że wykonują ściśle protokół.

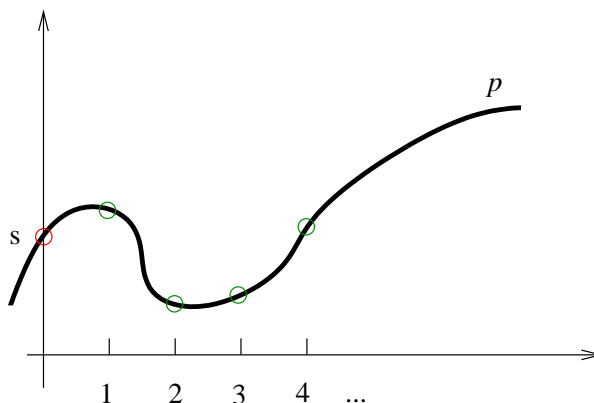
10 Wielopodmiotowe obliczenia (wróg pasywny)

Pokażemy teraz protokół wielopodmiotowych obliczeń (MPC), który działa w modelu, w którym wróg ma ograniczoną moc obliczeniową.

Przypuśćmy że wróg może skorumpować $t < n/2$ graczy. Do podziału sekretu użyjemy ideę z pracy Shamira *How to Share a Secret* (1979). Mianowicie, żeby podzielić sekret $s \in F$ wśród graczy P_1, \dots, P_n wykonujemy kolejno:

- wybieramy losowo wielomian p stopnia t , taki że $p(0) = s$
- wysyłamy $p(i)$ do P_i .

Poniżej znajduje się przykładowy wykres wielomianu p i zaznaczone na nim punkty - udziały poszczególnych graczy.



Jak łatwo zauważyć dowolny podzbiór graczy o mocy nie większej niż t nie ma żadnych informacji o s . Zaś każdy podzbiór o mocy większej niż t może zrekonstruować s przez interpolację p . W szczególności, jeśli $t = n - 1$ to istnieją wartości r_1, \dots, r_n takie że

$$p(0) = \sum_{i=1}^n r_i p(i).$$

a wektor, (r_1, \dots, r_n) jest nazywany wektorem rekombinacji.

10.1 Dodawanie

Dodawanie jest trywialne.

- Niech x_1, \dots, x_n będą udziałami x , a p_x odpowiednim wielomianem dla tego podziału.
- Niech y_1, \dots, y_n będą udziałami y , a p_y odpowiednim wielomianem dla tego podziału.

Wtedy by obliczyć $z = x + y$, wystarczy by każdy P_i dodał swoje udziały: $z_i := x_i + y_i$. W ten sposób zostaną utworzone udziały z_1, \dots, z_n będące punktami leżącymi na wielomianie $p_x + p_y$ a co za tym idzie udziałami $x + y$.

10.2 Mnożenie

Niestety idea, która działała dla dodawania jest nie prawidłowa dla mnożenia. Po pierwsze wielomian $p_x \cdot p_y$ ma za duży stopień $2t$, a po drugie $p_x \cdot p_y$ nie jest w pełni losowym wielomianem.

Niech x i y będą sekretne dzielone odpowiednio na x_1, \dots, x_n oraz y_1, \dots, y_n . Chcemy uzyskać podział $z = xy$, w tym celu:

- Każdy gracz P_i oblicza $z_i = x_i \cdot y_i$.
- Każdy gracz P_i sekretne dzieli z_i pomiędzy pozostałych graczy.

Niech z_{i1}, \dots, z_{in} będą uzyskanymi udziałami. Wtedy każde z_i jest liniową kombinacją z_{i1}, \dots, z_{in} . Warto zauważyć też że $(z_1, \dots, z_n) \cdot (r_1, \dots, r_n) = z$ (gdzie (r_1, \dots, r_n) jest wektorem rekombinacji).

	P_1	\dots	P_n
	z_1	\dots	z_n
P_1	z_{11}	\dots	z_{n1}
\vdots	\vdots	\vdots	\vdots
P_n	z_{1n}	\dots	z_{nn}

$$\begin{array}{l} P_1 \\ \text{Wynik mnożenia} \\ \vdots \\ P_n \end{array} \cdot \begin{bmatrix} z_{11} & \dots & z_{n1} \\ \vdots & \vdots & \vdots \\ z_{1n} & \dots & z_{nn} \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ \vdots \\ r_n \end{bmatrix}$$

utworzy poprawny sekretne podział z (stopnia t). To obliczenie może być wykonane lokalnie.

Literatura