

# Historia Alicji i Boba

Stefan Dziembowski

<http://www.mimuw.edu.pl/~std>

Instytut Informatyki  
Uniwersytet Warszawski

# Zarys historii kryptografii

Kryptografia rozwija się od czasów starożytnych.

Początkowo celem było zapewnienie bezpieczeństwa komunikacji.

# Zarys historii kryptografii

Kryptografia rozwija się od czasów starożytnych.

Początkowo celem było zapewnienie bezpieczeństwa komunikacji.

Obecnie kryptografia przeżywa burzliwy rozwój. W dużej mierze związany z zastosowaniami w e-gospodarce.

# Tradycyjny cel kryptografii [1/2]

bezpieczna komunikacja.

# Tradycyjny cel kryptografii [1/2]

bezpieczna komunikacja.

Zwykle zakładamy, że komunikujące się strony to

Alicja  Bob

# Tradycyjny cel kryptografii [1/2]

bezpieczna komunikacja.

Zwykle zakładamy, że komunikujące się strony to

Alicja  Bob

Komunikaty przesyłane pomiędzy nimi  
przeciwnik **Ewa** może

- podsłuchiwać,
- modyfikować i fabrykować.

# Tradycyjny cel kryptografii [2/2]

Stworzenie protokołów zabezpieczających przed podsłuchem:

szyfrowanie

oraz przed modyfikowaniem i fabrykowaniem:

autentykacja

# Co to jest *protokół*

Protokół dla Alicji i Boba to para programów komputerowych:

- program dla Alicji
- program dla Boba.

W każdym z tych programów Alicja może użyć komend:  $wyslij(Bob, x)$  i  $odbierz(Bob, x)$ . (podobnie w przypadku Boba).

Za pomocą tych komend odbywa się komunikacja między Alicją i Bobem.

# Co jeszcze mogą robić Alicja i Bob?

Ogólnie: Alicja i Bob

mogą wykonywać wspólnie jakieś zadanie  
nie ufając sobie nawzajem.

# Co jeszcze mogą robić Alicja i Bob?

Ogólnie: Alicja i Bob

mogą wykonywać wspólnie jakieś zadanie  
nie ufając sobie nawzajem.

(Zakładamy, że Alicja i Bob mają do dyspozycji  
bezpieczne łącze).

# Co jeszcze mogą robić Alicja i Bob?

Ogólnie: Alicja i Bob

mogą wykonywać wspólnie jakieś zadanie  
nie ufając sobie nawzajem.

(Zakładamy, że Alicja i Bob mają do dyspozycji  
bezpieczne łącze).

Ten dział kryptografii nazywa się

**bezpieczne obliczenia wielopodmiotowe**

(ang. secure multiparty computations).

# Plan

1. Wstęp ✓.
2. Przykłady bezpieczne obliczeń wielopodmiotowych.
3. Definicje i przegląd modeli.
4. Implementacje.
5. Ogólna dyskusja o założeniach potrzebnych do dowiedzenia bezpieczeństwa.

# Przykłady

# Przykład 1: Problem miłosny[1/3]

Alicja i Bob chcą sprawdzić, czy się nawzajem kochają nie ujawniając sobie swoich uczuć.

# Przykład 1: Problem miłosny[1/3]

Alicja i Bob chcą sprawdzić, czy się nawzajem kochają nie ujawniając sobie swoich uczuć.

Niech:

$$a := \begin{cases} 1 & \text{jeśli Alicja kocha Boba} \\ 0 & \text{jeśli Alicja nie kocha Boba} \end{cases}$$

$$b := \begin{cases} 1 & \text{jeśli Bob kocha Alicję} \\ 0 & \text{jeśli Bob nie kocha Alicji} \end{cases}$$

# Przykład 1: Problem miłosny[2/3]

Celem Alicji i Boba jest poznanie wartości funkcji

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

bez ujawniania sobie nawzajem wartości  $a$  i  $b$ .

- Jeśli  $f(a, b) = 1$ , to  $a = b = 1$  i o żadnej tajemnicy nie ma mowy.

# Przykład 1: Problem miłosny[2/3]

Celem Alicji i Boba jest poznanie wartości funkcji

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

bez ujawniania sobie nawzajem wartości  $a$  i  $b$ .

- Jeśli  $f(a, b) = 1$ , to  $a = b = 1$  i o żadnej tajemnicy nie ma mowy.
- Ale, jeśli  $a = 0$ , to  $f(a, b) = 0$ , niezależnie od wartości  $b$ . Zatem Alicja nie będzie wiedzieć, czy Bob ją kocha, czy nie.

# Przykład 1: Problem miłosny[3/3]

Jak to zrobić?

- Metoda tradycyjna:

# Przykład 1: Problem miłosny[3/3]

Jak to zrobić?

- Metoda tradycyjna:

Użyć zaufanej strony trzeciej.

# Przykład 1: Problem miłosny[3/3]

## Jak to zrobić?

- Metoda tradycyjna:  
Użyć zaufanej strony trzeciej.
- Metoda kryptograficzna:

# Przykład 1: Problem miłosny[3/3]

## Jak to zrobić?

- Metoda tradycyjna:  
Użyć zaufanej strony trzeciej.
- Metoda kryptograficzna:  
Skonstruować protokół!  
Taki protokół istnieje.

## Przykład 2: Rozwód

Alicja i Bob muszą zdecydować o podziale majątku.

Aby zdecydować o tym kto dostanie samochód postanawiają rzucić monetą.

## Przykład 2: Rozwód

Alicja i Bob muszą zdecydować o podziale majątku.

Aby zdecydować o tym kto dostanie samochód postanawiają rzucić monetą.

Problem: jak to zrobić bez spotykania się (kontaktując się wyłącznie przez telefon)?

## Przykład 2: Rozwód

Alicja i Bob muszą zdecydować o podziale majątku.

Aby zdecydować o tym kto dostanie samochód postanawiają rzucić monetą.

Problem: jak to zrobić bez spotykania się (kontaktując się wyłącznie przez telefon)?

Rozwiązanie: użyć protokołu kryptograficznego.  
Taki protokół istnieje!

# Ogólniejszy przykład

Czy można grać w karty przez telefon?

# Ogólniejszy przykład

Czy można grać w karty przez telefon?

Tak

(istnieje protokół kryptograficzny).

# Większe grupy uczestników

Ogólniej, zamiast pary

$\{\text{Alicja, Bob}\}$

możemy rozważać większe grupy uczestników:

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

Z których każdy  $P_i$  ma swój prywatny argument  $x_i$ .

# Przykład 3: Problem miłosny

Ten problem można uogólnić na grupę osób

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

# Przykład 3: Problem miłosny

Ten problem można uogólnić na grupę osób

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

w następujący sposób:

- Każda osoba  $P_i$  ma swoją prywatną listę osób które kocha.
- Wynikiem działania protokołu jest lista pasujących par.

Taki protokół istnieje.

# Przykład 4: Problem milionerów

Grupa  $n$  milionerów chce sprawdzić który jest najwięcej zarabia.

Nie chcą przy tym zdradzić wysokości swoich zarobków.

# Przykład 5: Głosowanie[1/2]

Każdy uczestnik  $P_i$  określa wartość swojego argumentu w następujący sposób:

$$x_i := \begin{cases} 1 & \text{jeśli } P_i \text{ jest za wejściem RP do UE} \\ 0 & \text{jeśli } P_i \text{ jest przeciwny wejściu RP do UE} \end{cases}$$

Obliczana funkcja jest sumą argumentów:

$$f(x_1, \dots, x_n) := \sum_{i=1}^n x_i.$$

# Przykład 5: Głosowanie[2/2]

Protokoły kryptograficzne protokoły do głosowania elektronicznego istnieją i mogą być użyte w praktyce.

# Przykład 5: Głosowanie[2/2]

Protokoły kryptograficzne protokoły do głosowania elektronicznego istnieją i mogą być użyte w praktyce.

Pewnym problemem jest zaewnienie niepokwitowalności

# Przykład 6: Szpitale [1/3]

Szpitale są zobowiązane do zachowania tajności danych swoich pacjentów.

Dla potrzeb medycznych potrzeba przeprowadzać badania statystyczne na tych danych.

# Przykład 6: Szpitale [1/3]

Szpitale są zobowiązane do zachowania tajemnicy danych swoich pacjentów.

Dla potrzeb medycznych potrzeba przeprowadzać badania statystyczne na tych danych.

Na przykład: chcemy wiedzieć ilu pacjentów chorujących na raka leczyło się psychiatrycznie.  
(Mogli się leczyć w różnych szpitalach.)

# Przykład 6: Szpitale [2/3]

Problem ten rozwiązuje się obecnie za pomocą zaufanej strony trzeciej.

## Przykład 6: Szpitale [2/3]

Problem ten rozwiązuje się obecnie za pomocą zaufanej strony trzeciej.

Roziązanie to wiąże się z następującymi niedogodnościami:

- jest kosztowne,
- i podatne na oszustwa.

# Przykład 6: Szpitale [3/3]

Czy da się to zrobić bez zaufanej strony trzeciej?

W zasadzie:

Tak.

## Przykład 6: Szpitale [3/3]

Czy da się to zrobić bez zaufanej strony trzeciej?

W zasadzie:

Tak.

W praktyce:

Nie ...

Problemem są bardzo duże rozmiary danych.

# Przykład 7: $(n, t)$ -podział sekretu

Ustalmy grupę  $n$  osób:  $\{P_1, \dots, P_n\}$  oraz parametr  $t < n$ .

Założmy, że znamy tajną liczbę  $s$ .

# Przykład 7: $(n, t)$ -podział sekretu

Ustalmy grupę  $n$  osób:  $\{P_1, \dots, P_n\}$  oraz parametr  $t < n$ .

Założmy, że znamy tajną liczbę  $s$ . Chcemy żeby:

- dowolna grupa maksymalnie  $t$  osób nie miała żadnej wiedzy na temat  $s$ , oraz
- dowolna grupa co najmniej  $t + 1$  osób potrafiła obliczyć  $s$ .

Istnieje odpowiedni protokół (pokażemy go za chwilę).

# Prywatne pozyskiwanie danych

Założmy, że mamy bazę danych reprezentowaną jako wektor bitów:  $(x_1, \dots, x_n)$ .

Chcemy poznać jedną ze współrzędnych w taki sposób, że baza danych nie będzie wiedziała o którą współrzędną nam chodzi.

# Prywatne pozyskiwanie danych

Założmy, że mamy bazę danych reprezentowaną jako wektor bitów:  $(x_1, \dots, x_n)$ .

Chcemy poznać jedną ze współrzędnych w taki sposób, że baza danych nie będzie wiedziała o którą współrzędną nam chodzi.

Trywialne rozwiązanie: Baza danych wysyła nam całą swoją zawartość.

# Prywatne pozyskiwanie danych

Założmy, że mamy bazę danych reprezentowaną jako wektor bitów:  $(x_1, \dots, x_n)$ .

Chcemy poznać jedną ze współrzędnych w taki sposób, że baza danych nie będzie wiedziała o którą współrzędną nam chodzi.

Trywialne rozwiązanie: Baza danych wysyła nam całą swoją zawartość.

Istnieje protokół kryptograficzny, który ma złożoność komunikacyjną znacznie niższą niż  $n$ .

# Definicje i modele

# Definicje

Co to znaczy, że  
dany protokół dla  $n$  graczy bezpiecznie  
oblicza funkcję  $f$  ?

# Definicje

Co to znaczy, że  
dany protokół dla  $n$  graczy bezpiecznie  
oblicza funkcję  $f$  ?

Nieformalnie: Oszukujący gracz nie może  
zaszkodzić bardziej niż jakby funkcja  $f$  była  
obliczona przez zaufaną stronę trzecia.

# Definicje

Co to znaczy, że  
dany protokół dla  $n$  graczy bezpiecznie  
oblicza funkcję  $f$  ?

Nieformalnie: Oszukujący gracz nie może  
zaszkodzić bardziej niż jakby funkcja  $f$  była  
obliczona przez zaufaną stronę trzecia.

W szczególności. Alicja może fałszywie  
zadeklarować, że kocha Boba. (Tego nie da się  
uniknąć.)

# Modele

- Z reguły przyjmuje się, że oszuści działają w koalicjach. Jaki jest maksymalny rozmiar takiej koalicji?

# Modele

- Z reguły przyjmuje się, że oszuści działają w koalicjach. Jaki jest maksymalny rozmiar takiej koalicji?
- Czy oszuści działają zgodnie z protokołem (pasywnie), czy niezgodnie (aktywnie)?

# Modele

- Z reguły przyjmuje się, że oszuści działają w koalicjach. Jaki jest maksymalny rozmiar takiej koalicji?
- Czy oszuści działają zgodnie z protokołem (pasywnie), czy niezgodnie (aktywnie)?
- Jaka mocą obliczeniową dysponują oszuści?

# Modele

- Z reguły przyjmuje się, że oszuści działają w koalicjach. Jaki jest maksymalny rozmiar takiej koalicji?
- Czy oszuści działają zgodnie z protokołem (pasywnie), czy niezgodnie (aktywnie)?
- Jaka mocą obliczeniową dysponują oszuści?
- Czy kanały między uczestnikami są bezpieczne?



- 
- 
- 



# Jak to zrobić?



- 
- 
- 
- 
- 
- 
- 
- 
- 
-

# Losowanie bitu przez telefon [1/2]

Alicja i Bob chcą zalogować bit  $r \in \{0, 1\}$  przez telefon.

## Pierwszy pomysł

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .

# Losowanie bitu przez telefon [1/2]

Alicja i Bob chcą załosować bit  $r \in \{0, 1\}$  przez telefon.

## Pierwszy pomysł

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Strony wysyłają sobie nawzajem to co wylosowały.

# Losowanie bitu przez telefon [1/2]

Alicja i Bob chcą zlosować bit  $r \in \{0, 1\}$  przez telefon.

## Pierwszy pomysł

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Strony wysyłają sobie nawzajem to co wylosowały.
3. Wynikiem losowania jest  $a \oplus b$ .

$$(a \oplus b := a + b \text{ mod } 2)$$

# Losowanie bitu przez telefon [2/2]

Zauważmy, że wystarczy, żeby co najmniej jedna z liczb  $a$  i  $b$  była wybrana losowo, żeby

$$a \oplus b = \begin{cases} 0 & \text{z prawdopodobieństwem } \frac{1}{2} \\ 1 & \text{z prawdopodobieństwem } \frac{1}{2} \end{cases}$$

# Losowanie bitu przez telefon [2/2]

Zauważmy, że wystarczy, żeby co najmniej jedna z liczb  $a$  i  $b$  była wybrana losowo, żeby

$$a \oplus b = \begin{cases} 0 & \text{z prawdopodobieństwem } \frac{1}{2} \\ 1 & \text{z prawdopodobieństwem } \frac{1}{2} \end{cases}$$

Protokół ma jednak pewien problem:

# Losowanie bitu przez telefon [2/2]

Zauważmy, że wystarczy, żeby co najmniej jedna z liczb  $a$  i  $b$  była wybrana losowo, żeby

$$a \oplus b = \begin{cases} 0 & \text{z prawdopodobieństwem } \frac{1}{2} \\ 1 & \text{z prawdopodobieństwem } \frac{1}{2} \end{cases}$$

Protokół ma jednak pewien problem:

jak zagwarantować, żeby strony wysłały sobie bity  $a$  i  $b$  dokładnie w tym samym momencie?

# Losowanie bitu przez telefon [2/2]

Zauważmy, że wystarczy, żeby co najmniej jedna z liczb  $a$  i  $b$  była wybrana losowo, żeby

$$a \oplus b = \begin{cases} 0 & \text{z prawdopodobieństwem } \frac{1}{2} \\ 1 & \text{z prawdopodobieństwem } \frac{1}{2} \end{cases}$$

Protokół ma jednak pewien problem:

jak zagwarantować, żeby strony wysłały sobie bity  $a$  i  $b$  dokładnie w tym samym momencie?

w praktyce (internet, telefon) nie da się ...

# Protokół zobowiązania bitowego

Założmy, że Alicja wybrała bit  $a$ .

**Zobowiązanie** Alicja wysyła do Boba jakąś informację  $x$ .  
Na podstawie  $x$  Bob nie potrafi obliczyć  $a$ .

**Otworzenie** Alicja wysyła do Boba informację  $y$ , taką, że  
Bob może obliczyć  $a := B(x, y)$ .

Alicja nie jest w stanie podać  $y'$  takiego, że  $B(x, y) = 1 - a$ .

# Protokół zobowiązania bitowego

Założmy, że Alicja wybrała bit  $a$ .

**Zobowiązanie** Alicja wysyła do Boba jakąś informację  $x$ .  
Na podstawie  $x$  Bob nie potrafi obliczyć  $a$ .

**Otworzenie** Alicja wysyła do Boba informację  $y$ , taką, że  
Bob może obliczyć  $a := B(x, y)$ .

Alicja nie jest w stanie podać  $y'$  takiego, że  $B(x, y) = 1 - a$ .

Intuicja:  $x$  jest pudełkiem w którym Alicja zamyka  $a$ , liczba  
 $y$  jest kluczem do tego pudełka.

# Przykład zobowiązania bitowego

$E$  — funkcja szyfrowania w ustalonym kryptosystemie klucza publicznego.

$a$  — bit Alicji.

**Zobowiązanie** Alicja wybiera klucz publiczny  $e$  oraz losową wiadomość  $m$ . Niech  $c := E(p, m)$  i niech  $m_1$  oznacza ostatni bit  $m$ .

Alicja wysyła  $e, c$  i  $m_1 \oplus a$  do Boba.

# Przykład zobowiązania bitowego

$E$  — funkcja szyfrowania w ustalonym kryptosystemie klucza publicznego.

$a$  — bit Alicji.

**Zobowiązanie** Alicja wybiera klucz publiczny  $e$  oraz losową wiadomość  $m$ . Niech  $c := E(p, m)$  i niech  $m_1$  oznacza ostatni bit  $m$ .

Alicja wysyła  $e, c$  i  $m_1 \oplus a$  do Boba.

**Otworzenie** Alicja wysyła  $m$  do Boba. Bob sprawdza, czy  $c = E(p, m)$ . Jeśli tak, to oblicza  $a$ .

# Poprawiony protokół

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .

# Poprawiony protokół

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Alicja zobowiązuje się do bitu  $a$ .

# Poprawiony protokół

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Alicja zobowiązuje się do bitu  $a$ .
3. Bob wysyła do Alicji bit  $b$

# Poprawiony protokół

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Alicja zobowiązuje się do bitu  $a$ .
3. Bob wysyła do Alicji bit  $b$
4. Alicja otwiera zobowiązanie do  $a$

# Poprawiony protokół

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Alicja zobowiązuje się do bitu  $a$ .
3. Bob wysyła do Alicji bit  $b$
4. Alicja otwiera zobowiązanie do  $a$
5. Wynikiem losowania jest  $a \oplus b$ .

# Poprawiony protokół

1. Alicja losuje bit  $a \in \{0, 1\}$ .  
Bob losuje bit  $b \in \{0, 1\}$ .
2. Alicja zobowiązuje się do bitu  $a$ .
3. Bob wysyła do Alicji bit  $b$
4. Alicja otwiera zobowiązanie do  $a$
5. Wynikiem losowania jest  $a \oplus b$ .

Jeśli Alicja odmówiła otworzenia zobowiązania, to automatycznie przegrywa.

# $(n, t)$ -podział sekretu

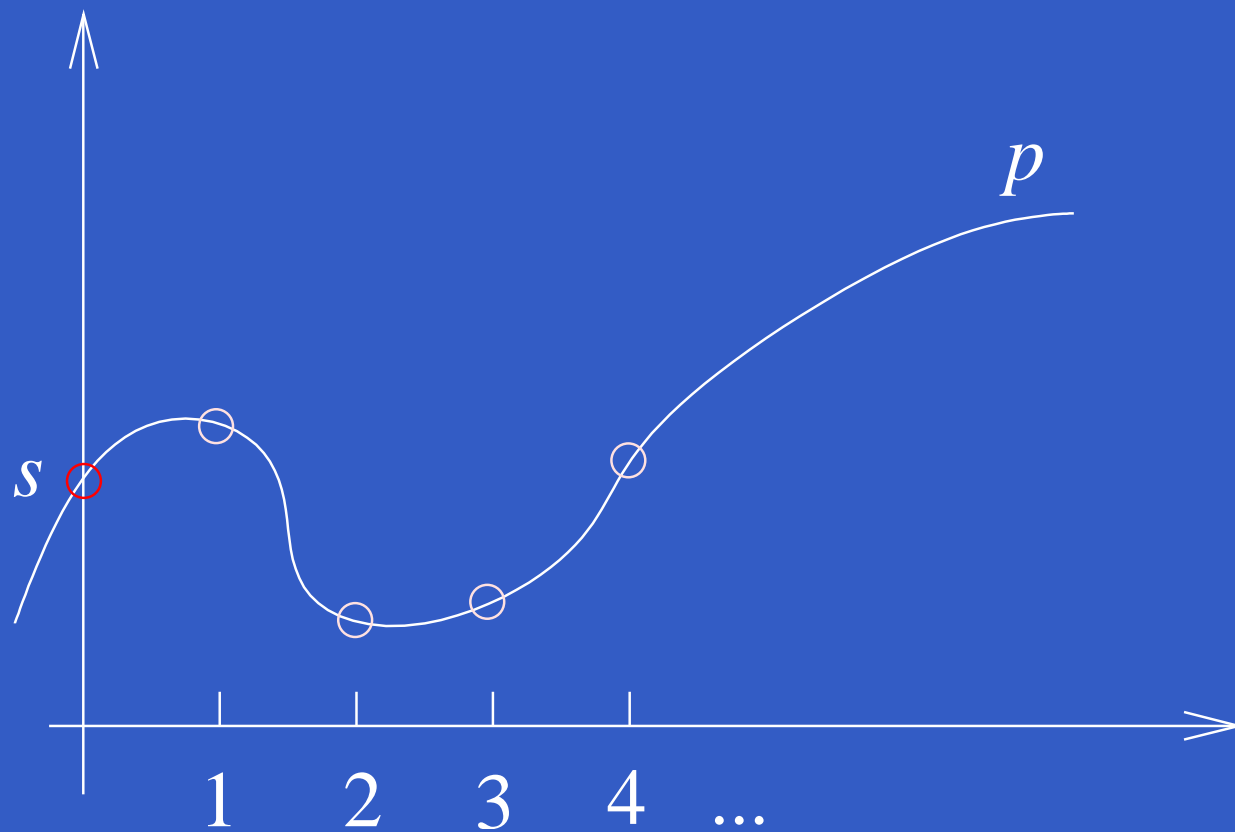
Ustalmy ciało  $F$ . Aby  $(n, t)$ -podzielić sekret  $s \in \{0, 1\}$ .

- Wybieramy losowy wielomian  $p$  stopnia maksymalnie  $t$  taki, że

$$p(0) = s.$$

- Do każdego gracza  $P_i$  wysyłamy  $p(i)$ .

# $(n, t)$ -podział sekretu: intuicja



# $(n, t)$ -podział sekretu: rekonstrukcja

Jeśli grupa graczy  $P_{i_1}, \dots, P_{i_{t+1}}$  chce zrekonstruować sekret wówczas:

- każdy gracz  $P_{i_j}$  oznajmia pozostałym liczbę,  $p(i_j)$ , którą otrzymał.
- na podstawie tych wartości gracze interpolują wartość wielomianu  $p$  dla argumentu 0.

# $(n, t)$ -podział sekretu: prywatność

Z właściwości wielomianów wynika, że dowolna grupa co najwyżej  $t$  graczy nie ma żadnej informacji na temat sekretu.

Nieformalnie mówiąc: jeśli znamy tylko  $t$  punktów wielomianu, to nic nie wiemy na temat jego wartości w zerze.

# $(n, t)$ -podział sekretu: prywatność

Z właściwości wielomianów wynika, że dowolna grupa co najwyżej  $t$  graczy nie ma żadnej informacji na temat sekretu.

Nieformalnie mówiąc: jeśli znamy tylko  $t$  punktów wielomianu, to nic nie wiemy na temat jego wartości w zerze.

Pytanie do zastanowienia się:

Jak to zformalizować?

# $(n, t)$ -podział sekretu: addytywność

Powyższy protokół ma dodatkową własność:

addytywność.

# $(n, t)$ -podział sekretu: addytywność

Powyższy protokół ma dodatkową własność:

addytywność.

Założmy, że podzielone zostały dwa sekrety:  
 $s_1$  i  $s_2$  za pomocą wielomianów  $p_1$  i  $p_2$ .

# $(n, t)$ -podział sekretu: addytywność

Powyższy protokół ma dodatkową własność:

addytywność.

Założmy, że podzielone zostały dwa sekrety:  
 $s_1$  i  $s_2$  za pomocą wielomianów  $p_1$  i  $p_2$ .

Jeśli każdy gracz  $P_i$  podstawia  $x_i := p_1(i) + p_2(i)$ ,  
to wartości  $x_1, \dots, x_n$  utworzą  $(n, t)$ -podział  
sekretu  $s_1 + s_2$ .

# $(n, t)$ -podział sekretu: addytywność

Powyższy protokół ma dodatkową własność:

addytywność.

Założmy, że podzielone zostały dwa sekrety:  
 $s_1$  i  $s_2$  za pomocą wielomianów  $p_1$  i  $p_2$ .

Jeśli każdy gracz  $P_i$  podstawia  $x_i := p_1(i) + p_2(i)$ ,  
to wartości  $x_1, \dots, x_n$  utworzą  $(n, t)$ -podział  
sekretu  $s_1 + s_2$ .

Pytanie dodatkowe: Co z mnożeniem?

# Protokół głosowania

Addytywności możemy użyć przy konstrukcji następującego protokołu do głosowania w grupie  $n$  graczy:

- Każdy gracz  $P_i$  ( $n, n - 1$ )-dzieli swój głos  $x_i$  pomiędzy pozostałych graczy (za pomocą wielomianu  $p_i$ ).

# Protokół głosowania

Addytywności możemy użyć przy konstrukcji następującego protokołu do głosowania w grupie  $n$  graczy:

- Każdy gracz  $P_i$  ( $n, n - 1$ )-dzieli swój głos  $x_i$  pomiędzy pozostałych graczy (za pomocą wielomianu  $p_i$ ).
- Gracze wspólnie rekonstruują sekret  $x_1 + \dots + x_n$  interpolując wielomian  $p_1 + \dots + p_n$ .

# Problem z protokołem głosowania

Jeśli wszyscy gracze zachowują się zgodnie z protokołem, to wszystko jest OK.

# Problem z protokołem głosowania

Jeśli wszyscy gracze zachowują się zgodnie z protokołem, to wszystko jest OK.

Ale jeśli któryś z graczy zacznie aktywnie oszukiwać, wówczas może on zaburzyć wynik.

# Problem z protokołem głosowania

Jeśli wszyscy gracze zachowują się zgodnie z protokołem, to wszystko jest OK.

Ale jeśli któryś z graczy zacznie aktywnie oszukiwać, wówczas może on zaburzyć wynik.

Protokół zabezpieczający przed aktywnymi oszustami istnieje, ale jest znacznie bardziej skomplikowany!

# Problem z protokołem głosowania

Jeśli wszyscy gracze zachowują się zgodnie z protokołem, to wszystko jest OK.

Ale jeśli któryś z graczy zacznie aktywnie oszukiwać, wówczas może on zaburzyć wynik.

Protokół zabezpieczający przed aktywnymi oszustami istnieje, ale jest znacznie bardziej skomplikowany!

Jest to przykład ogólnego zjawiska

# Modularność [1/2]

Protokoły można konstruować modularnie:

1. Konstrujemy pod-protokół do wykonania konkretnego pod-zadania.
2. Używamy tego pod-protokołu do skonstruowania głównego protokołu.

# Modularność [1/2]

Protokoły można konstruować modularnie:

1. Konstrujemy pod-protokół do wykonania konkretnego pod-zadania.
2. Używamy tego pod-protokołu do skonstruowania głównego protokołu.

Przykład: Używamy protokołu zobowiązania bitowego do konstrukcji protokołu losowania bitu.

# Modularność [2/2]

Okazuje się, że w zasadzie dowolny protokół można skonstruować korzystając z jednego protokołu.

# Modularność [2/2]

Okazuje się, że w zasadzie dowolny protokół można skonstruować korzystając z jednego protokołu. Ten protokół, to

transfer utajniony (ang. oblivious transfer).

# Transfer utajniony

Protokół transferu utajnionego działa w następujący sposób:

- argument Alicji: parę liczb  $(x_0, x_1)$
- argument Boba: bit  $s \in \{0, 1\}$ .

W wyniku wykonania protokołu.

- Alicja nie dowiaduje się niczego.
- Bob poznaje liczbę  $x_s = \text{OT}(x_0, x_1, s)$  (i nic więcej).

# Problem miłosny

Jak Alicja i Bob mogą obliczyć koniunkcję swoich argumentów  $a \wedge b$  (w modelu z pasywnymi oszustwami)?

# Problem miłosny

Jak Alicja i Bob mogą obliczyć koniunkcję swoich argumentów  $a \wedge b$  (w modelu z pasywnymi oszustwami)?

- Alicja i Bob wykonują protokół transferu utajnionego z argumentem  $(0, a)$  dla Alicji i  $b$  dla Boba. Niech  $y = \text{OT}(0, a, b)$  będzie wynikiem.
- Bob wysyła  $y$  do Alicji. Oboje uznają  $y$  za wynik.

# Implementacja transferu utajnionego

$(x_0, x_1)$  – argument Alicji,  $s$  — argument Boba.

- Alicja wybiera klucz publiczny  $e$  i prywatny  $d$  i wysyła  $e$  go do Boba.

# Implementacja transferu utajnionego

$(x_0, x_1)$  – argument Alicji,  $s$  — argument Boba.

- Alicja wybiera klucz publiczny  $e$  i prywatny  $d$  i wysyła  $e$  go do Boba.
- Bob wybiera losowo  $m$  i  $c$ , podstawia  $z_s := E(e, m)$  oraz  $z_{1-c} := c$  i wysyła  $(z_0, z_1)$  do Alicji.

# Implementacja transferu utajnionego

$(x_0, x_1)$  – argument Alicji,  $s$  — argument Boba.

- Alicja wybiera klucz publiczny  $e$  i prywatny  $d$  i wysyła  $e$  go do Boba.
- Bob wybiera losowo  $m$  i  $c$ , podstawia  $z_s := E(e, m)$  oraz  $z_{1-c} := c$  i wysyła  $(z_0, z_1)$  do Alicji.
- Alicja odszyfrowuje  $z_0$  i  $z_1$ . Niech  $w_0, w_1$  będą ostatnimi bitami odszyfrowanych wiadomości. Alicja wysyła  $(w_0 \oplus x_0, w_1 \oplus x_1)$  Bobowi.

# Implementacja transferu utajnionego

$(x_0, x_1)$  – argument Alicji,  $s$  — argument Boba.

- Alicja wybiera klucz publiczny  $e$  i prywatny  $d$  i wysyła  $e$  go do Boba.
- Bob wybiera losowo  $m$  i  $c$ , podstawia  $z_s := E(e, m)$  oraz  $z_{1-c} := c$  i wysyła  $(z_0, z_1)$  do Alicji.
- Alicja odszyfrowuje  $z_0$  i  $z_1$ . Niech  $w_0, w_1$  będą ostatnimi bitami odszyfrowanych wiadomości. Alicja wysyła  $(w_0 \oplus x_0, w_1 \oplus x_1)$  Bobowi.
- Bob zna  $w_s$ , więc może obliczyć  $x_s$ .

# Jak policzyć dowolną funkcję

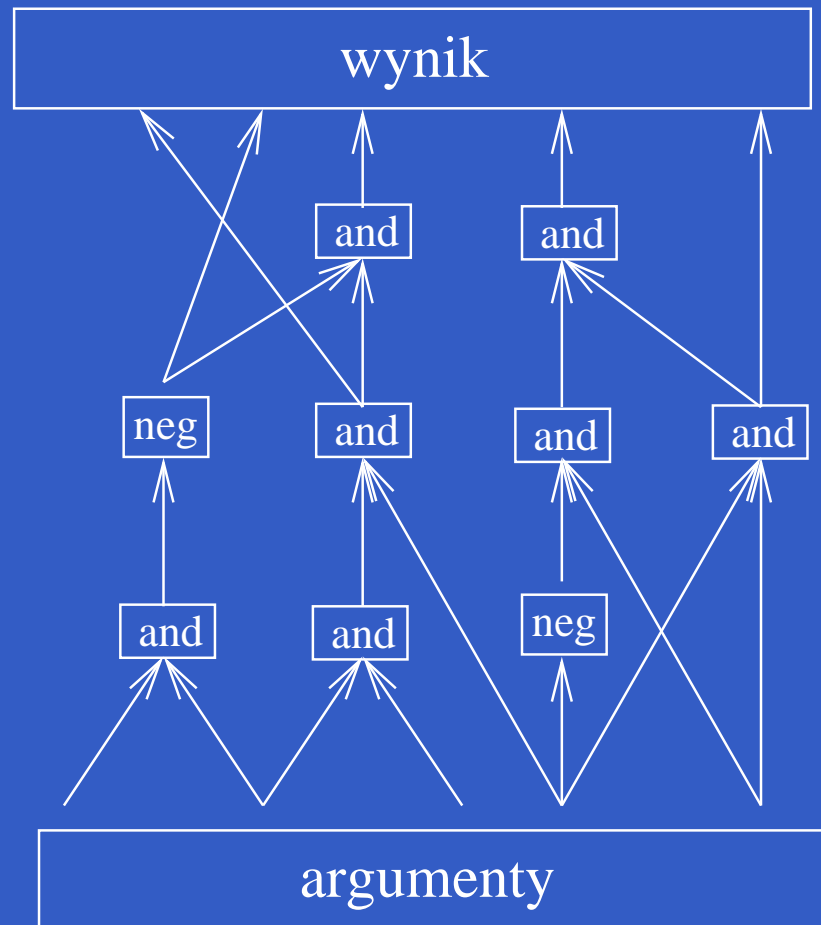
Dowolną wydajnie obliczalną  $n$ -argumentową funkcję można bezpiecznie obliczyć za pomocą protokołu dla  $n$  graczy ( $n = 2, 3, \dots$ ).

# Jak policzyć dowolną funkcję

Dowolną wydajnie obliczalną  $n$ -argumentową funkcję można bezpiecznie obliczyć za pomocą protokołu dla  $n$  graczy ( $n = 2, 3, \dots$ ).

Idea: przedstawiamy funkcję jako obwód logiczny.

# Obwód logiczny



# Idea działania protokołu

- Na początku każdy uczestnik dzieli swój argument między pozostałych uczestników.
- Tworzymy pod-protokoły, które na podstawie podziałów sekretów  $x$  i  $y$  pozwala graczom uzyskać podział
  - $x \wedge y$
  - negacji  $x$ .
- Aby uzyskać wynik gracze używają protokołu rekonstrukcji sekretu.

# Protokół dla 2 graczy

Sekret  $x \in \{0, 1\}$  jest dzielony między Alicję i Boba przez wybranie losowej pary bitów  $(x_A, x_B)$  takich, że  $x_A \oplus x_B = s$ . Bit  $x_A$  jest przydzielany Alicji, a bit  $x_B$  jest przydzielany Bobowi.

# Protokół dla 2 graczy

Sekret  $x \in \{0, 1\}$  jest dzielony między Alicję i Boba przez wybranie losowej pary bitów  $(x_A, x_B)$  takich, że  $x_A \oplus x_B = s$ . Bit  $x_A$  jest przydzielany Alicji, a bit  $x_B$  jest przydzielany Bobowi.

Rekonstrukcja jest trywialna.

# Protokół dla 2 graczy

Sekret  $x \in \{0, 1\}$  jest dzielony między Alicję i Boba przez wybranie losowej pary bitów  $(x_A, x_B)$  takich, że  $x_A \oplus x_B = s$ . Bit  $x_A$  jest przydzielany Alicji, a bit  $x_B$  jest przydzielany Bobowi.

Rekonstrukcja jest trywialna.

Negacja też jest prosta: Alicja o prostu neguje swój bit, a Bob nic nie robi.

# Protokół dla 2 graczy: mnożenie

Założmy, że mamy 2 sekrety  $x$  i  $y$ . Alicja zna bity  $x_A$  i  $y_A$  a Bob zna bity  $x_B$  i  $y_B$  takie, że  $x = x_A \oplus x_B$  i  $y = y_A \oplus y_B$ .

- Alicja losuje bity  $p, q$ . Wykonywane są protokoły:

$$\text{OT}(p, p \oplus x_A, y_B) = p \oplus x_A y_B$$

$$\text{OT}(q, q \oplus y_A, x_B) = q \oplus y_A x_B$$

# Protokół dla 2 graczy: mnożenie

Założmy, że mamy 2 sekrety  $x$  i  $y$ . Alicja zna bity  $x_A$  i  $y_A$  a Bob zna bity  $x_B$  i  $y_B$  takie, że  $x = x_A \oplus x_B$  i  $y = y_A \oplus y_B$ .

- Alicja losuje bity  $p, q$ . Wykonywane są protokoły:

$$\text{OT}(p, p \oplus x_A, y_B) = p \oplus x_A y_B$$

$$\text{OT}(q, q \oplus y_A, x_B) = q \oplus y_A x_B$$

- Podział  $z = xy$  tworzą:

# Protokół dla 2 graczy: mnożenie

Założmy, że mamy 2 sekrety  $x$  i  $y$ . Alicja zna bity  $x_A$  i  $y_A$  a Bob zna bity  $x_B$  i  $y_B$  takie, że  $x = x_A \oplus x_B$  i  $y = y_A \oplus y_B$ .

- Alicja losuje bity  $p, q$ . Wykonywane są protokoły:

$$\text{OT}(p, p \oplus x_A, y_B) = p \oplus x_A y_B$$

$$\text{OT}(q, q \oplus y_A, x_B) = q \oplus y_A x_B$$

- Podział  $z = xy$  tworzą:

- Część Alicji:  $z_A := x_A y_A \oplus p \oplus q$ .

# Protokół dla 2 graczy: mnożenie

Założmy, że mamy 2 sekrety  $x$  i  $y$ . Alicja zna bity  $x_A$  i  $y_A$  a Bob zna bity  $x_B$  i  $y_B$  takie, że  $x = x_A \oplus x_B$  i  $y = y_A \oplus y_B$ .

- Alicja losuje bity  $p, q$ . Wykonywane są protokoły:

$$\text{OT}(p, p \oplus x_A, y_B) = p \oplus x_A y_B$$

$$\text{OT}(q, q \oplus y_A, x_B) = q \oplus y_A x_B$$

- Podział  $z = xy$  tworzą:

- Część Alicji:  $z_A := x_A y_A \oplus p \oplus q$ .

- Część Boba:  $z_B := (p \oplus x_A y_B) \oplus (q \oplus y_A x_B) \oplus x_B y_B$ .

# Ogólna dyskusja o założeniach potrzebnych do dowiedzenia bezpieczeństwa

# Założenia złożonościowe

Praktycznie wszystkie współczesne szyfry opierają swe bezpieczeństwo na założeniu, że moc obliczeniowa przeciwnika jest ograniczona.

# Założenia złożonościowe

Praktycznie wszystkie współczesne szyfry opierają swe bezpieczeństwo na założeniu, że moc obliczeniowa przeciwnika jest ograniczona.

Jak pokazał Claude Shannon w latach 40tych takie założenie jest konieczne.

Inaczej mówiąc: nie ma praktycznych szyfrów, które byłyby bezpieczne bez takiego założenia.

# Nieudowodnione hipotezy

Co gorsza: nikt nie potrafi dowieść bezpieczeństwa  
żadnego praktycznego szyfru (w oparciu o założenie, że  
moc przeciwnika jest ograniczona).

# Nieudowodnione hipotezy

Co gorsza: nikt nie potrafi dowieść bezpieczeństwa żadnego praktycznego szyfru (w oparciu o założenie, że moc przeciwnika jest ograniczona).

W praktyce opieramy więc bezpieczeństwo na założeniu, że jakiś problem (np. faktoryzacja dużych liczb naturalnych) nie daje się rozwiązać za pomocą praktycznie dostępnych mocy obliczeniowych.

# Nieudowodnione hipotezy

Co gorsza: nikt nie potrafi dowieść bezpieczeństwa żadnego praktycznego szyfru (w oparciu o założenie, że moc przeciwnika jest ograniczona).

W praktyce opieramy więc bezpieczeństwo na założeniu, że jakiś problem (np. faktoryzacja dużych liczb naturalnych) nie daje się rozwiązać za pomocą praktycznie dostępnych mocy obliczeniowych.

Pokazanie bezpieczeństwa jakiegokolwiek szyfru implikowałoby rozwiązanie problemu „ $P=NP$ ”, który jest bardzo znanym problemem.

# Protokoły wielopodmiotowe

Nieco lepiej rzecz się ma z protokołami wielopodmiotowymi. Bez jakichkolwiek założeń możemy policzyć dowolną funkcję w gronie  $n$  graczy, pod warunkiem, że:

- Liczba oszustów nie przekracza  $n/3$  (jeśli oszukują aktywnie).
- Liczba oszustów nie przekracza  $n/2$  (jeśli oszukują pasywnie).

Jeśli oszustów jest więcej, to protokoły muszą opierać się na nieudowodnionych założeniach.

•  
•  
•

# Koniec

Te slajdy zostaną umieszczone na stronie  
<http://www.mimuw.edu.pl/■std>.