Uniwersytet im. Adama Mickiewicza
w Poznaniu
Wydział Matematyki i Informatyki

Łukasz Nitschke

Bezpieczeństwo protokołów w środowisku o ograniczonym zaufaniu

Rozprawa doktorska
napisana pod kierunkiem
prof. dra hab.
Jerzego Jaworskiego

Poznań 2016

Część z wyników zaprezentowanych w rozprawie doktorskiej została uzyskana w wyniku badań realizowanych w ramach grantu „Technologie ochrony danych i komunikacji oparte na technikach kryptograficznych" 0 T00A 003 23.

Część z wyników zaprezentowanych w rozprawie doktorskiej została uzyskana w wyniku badań realizowanych w ramach grantu „Teoretyczne aspekty bezpieczeństwa informacji, komunikacji oraz protokołów kryptograficznych" N N206 2701 33.

Składam serdeczne podziękowania
panu profesorowi
Jerzemu Jaworskiemu
za okazaną życzliwość
oraz wskazówki udzielone mi podczas pisania pracy.

Spis treści

Wstęp

Niniejsza rozprawa poświęcona jest problemowi bezpieczeństwa obliczeń w środowisku o ograniczonym zaufaniu, jednemu z podstawowych zagadnień współczesnej kryptologii. Celem pracy jest przedstawienie matematycznych narzędzi służących zwiększeniu bezpieczeństwa obliczeń oraz wykorzystujących je konkretnych rozwiązań. Głównymi zastosowaniami, które zostały szczegółowo omówione w pracy są protokoły wyborów elektronicznych oraz systemy elektronicznego egzaminowania. Przykłady te pozwalają zrozumieć wagę problemu bezpieczeństwa obliczeń dla współczesnej informatyki oraz dla rozwoju społeczeństwa informacyjnego. Przeglądowi istniejących rozwiązań oraz współczesnych trendów towarzyszy w pracy prezentacja własnych, oryginalnych protokołów. Opisane propozycje są wynikiem wieloletnich badań, których rezultaty w większości zostały już opublikowane.

Pierwsza część pracy zawiera definicje podstawowych pojęć oraz opisy algorytmów związanych z bezpieczeństwem obliczeń. Prezentujemy metody wykorzystujące techniki anonimizujące oparte na sieciach mieszających. W szczególności zaproponowana została modyfikacja procedury weryfikującej działanie takich sieci pozwalająca na bezpieczne, dwukrotne ich wykorzystanie. Propozycja ta jest wsparta analizą konsekwencji płynących z ujawniania częściowych informacji w procesie weryfikacji.

W rozdziale trzecim opisane zostały protokoły wyborów elektronicznych oraz systemy głosowania dla małych grup wyborców. Podrozdział 3.3 zawiera propozycję nowego protokołu bazującego na kartach chipowych z ekranem. Ta część rozprawy została oparta na opublikowanej już pracy [51]. Inne oryginalne rozwiązanie (również opublikowane – patrz [52]), przedstawione w podrozdziale 3.4, dotyczy wyborów elektronicznych opartych na papierowych kartach do głosowania. Ostatnia część rozdziału poświęcona jest systemom wyborczym dla małych grup głosujących. Dla odpowiednio sformułowanych wymagań dla tego typu głosowania zaproponowane zostały nowe protokoły oparte na szyfrowaniu homomorficznym.

Rozdział czwarty poświęcony jest systemom e-egzaminów. W szczególności zaproponowane zostały nowe rozwiązania problemu egzaminów elektro-

nicznych posiadające odpowiednie własności bezpieczeństwa dzięki zastosowaniu zmodyfikowanych sieci mieszających oraz dwuetapowych procedurach sprawdzających (podrozdział 4.3). Przedstawione w rozdziale propozycje zostały już przygotowane do ich publikacji.

Rozdział 1

Bezpieczeństwo obliczeń

Pojęcie bezpieczeństwa obliczeń jest nierozłącznie związane z pojęciem bezpieczeństwa informacji, które jest kluczowym pojęciem kryptografii. Niemniej, mówiąc o bezpieczeństwie obliczeń należy również brać pod uwagę zagrożenia związane z środowiskiem, maszyną, na której są wykonywane owe obliczenia. Zagadnienie to nazywa się w literaturze problemem bezpiecznej platformy [55]. Mamy z nim do czynienia w wielu systemach korzystających z narzędzi kryptograficznych. Jako przykład mogą posłużyć systemy do realizacji wyborów elektronicznych przez Internet. W rozwiązaniach tego typu zakłada się, że wyborca przy pomocy komputera osobistego może oddać tajny, anonimowy głos. Oznacza to, iż komputer wyborcy powinien w pewien sposób zaszyfrować głos (wykonać pewne obliczenia), a następnie przesłać go do serwera Komisji Wyborczej (EA – ang. *election authority*). Z punktu widzenia bezpieczeństwa obliczeń, ten uproszczony opis systemu wyborczego powinien sprowokować następujące pytania. Jaką pewność ma wyborca, iż jego głos nie został zmieniony i w tej zmienionej postaci nie został przesłany do komisji? Czy wyborca może ufać swojemu, potencjalnie zawirusowanemu, komputerowi? Jak zabezpieczyć taki system? Podobne pytania mogą się pojawić podczas analizy bezpieczeństwa systemów zdalnego egzaminowania, czy też w przypadku systemów mobilnych agentów [53].

Poniższy rozdział zawiera definicje podstawowych pojęć oraz opis standardowych narzędzi związanych z zagadnieniem bezpieczeństwa informacji. Zaprezentowany zostaje również wspomniany wyżej problem bezpiecznej platformy.

1.1 Definicje podstawowych pojęć

Bezpieczeństwo obliczeń, podobnie jak bezpieczeństwo informacji, realizuje się poprzez osiągnięcie pewnych celów. Podstawowy zestaw tych celów obejmuje: poufność i integralność danych, uwierzytelnianie uczestników, oraz uwierzytelnianie źródeł pochodzenia danych. Własności te uzyskiwane są poprzez zastosowanie w odpowiedni sposób technik kryptograficznych obejmujących algorytmy kryptograficzne oraz protokoły kryptograficzne. Podstawowe algorytmy wykorzystywane dla celów bezpieczeństwa to: algorytmy szyfrujące (oraz deszyfrujące), funkcje haszujące, bezpieczne generatory pseudolosowe, podpisy cyfrowe. Protokoły kryptograficzne to często rozproszone algorytmy wykonywane przez wiele maszyn (zwanych uczestnikami), wymieniających między sobą odpowiednio sformatowane komunikaty. Opis protokołu zawiera zatem specyfikację odpowiednio uporządkowanych wiadomości wysyłanych przez poszczególnych uczestników. Utworzenie komunikatu zgodnego ze specyfikacją wymaga często skorzystania z jednego lub wielu algorytmów kryptograficznych.

Uwaga 1.1.1 *W dalszej części pracy pewne problemy obliczeniowe będą określane jako trudne obliczeniowo. Oznaczać będzie to, że dla danego problemu nie istnieje wielomianowy algorytm, tzn. taki w którym zależność między liczbą kroków prowadzących do otrzymania rezultatu a rozmiarem danych wejściowych jest ograniczona asymptotycznie przez pewien wielomian. W praktyce, problem trudny obliczeniowo to taki, dla którego nie istnieje efektywny algorytm pozwalający przy obecnym stanie techniki na rozwiązanie tego problemu. Analogicznie, jeżeli dany problem nie jest obliczeniowo trudny, to problem będziemy określać łatwym obliczeniowo lub będziemy mówić, iż dla danego problemu istnieje efektywny (obliczeniowo) algorytm. Bardziej formalne definicje powyższych pojęć można znaleźć w [47].*

Jednym z najistotniejszych i najczęściej wykorzystywanych narzędzi kryptograficznych są funkcje haszujące (zwane również jednokierunkowymi funkcjami haszującymi lub jednokierunkowymi funkcjami skrótu).

Definicja 1.1.1 *Funkcję $h : D \rightarrow P$ nazywamy* funkcją haszującą *jeżeli posiada następujące własności:*

1. *własność* kompresji – *dziedzina D funkcji h obejmuje wszystkie skończone ciągi binarne, natomiast wartościami $h \in P$ są ciągi binarne o określonej długości;*

2. łatwość obliczeń – *mając dane $x \in D$ wyznaczenie wartości $h(x)$ jest łatwe obliczeniowo (patrz uwaga 1.1.1);*

3. nieodwracalność *(*jednokierunkowość*) – dla danej wartości funkcji* $y \in P$ *znalezienie* $x \in D$ *takiego, że* $h(x) = y$ *jest obliczeniowo trudne (patrz uwaga 1.1.1).*

Od funkcji haszujących wymaga się również by posiadały własność bezkolizyjności, która polega na trudności znalezienia pary argumentów dających tę samą wartość (znalezienia kolizji). Własność ta pozwala między innymi na wykrywanie niepożądanych zmian w danych (zachowywanie integralności danych) oraz tworzenie trudno podrabialnych skrótów wiadomości.

Definicja 1.1.2 *Bezkolizyjność funkcji haszującej (odporność na kolizję) definiujemy na dwóch poziomach.*

1. Słaba bezkolizyjność *(ang.* weak collision resistance, *second preimage resistance) – mając dane* x *i* $h(x) = y$ *problem znalezienia* $x' \neq x$ *takiego, że* $h(x') = y$ *jest obliczeniowo trudny.*

2. Bezkolizyjność – *dla danej funkcji haszującej* h *znalezienie dwóch różnych argumentów* x *i* x' *takich, że* $h(x) = h(x')$ *jest problemem obliczeniowo trudnym.*

Warunek bezkolizyjności implikuje słabą bezkolizyjność.

Opisane w dalszej części algorytmy szyfrowania, czy też podpisu cyfrowego często wykorzystują (np. przy tworzeniu klucza, podczas szyfrowania, podpisywania wiadomości) wartości pseudolosowe. Wartości pseudolosowe tworzone są przy pomocy pseudolosowego generatora bitów.

Definicja 1.1.3 *Pseudolosowym generatorem bitów nazywamy algorytm deterministyczny, który otrzymawszy na wejściu* k*-bitową wartość* s *zwraca ciąg bitowy dowolnej długości, którego odróżnienie od ciągu losowego jest problemem obliczeniowo trudnym. Wartość wejściowa* s *nazywana jest zarodkiem generatora (ang. seed), natomiast wartość wyjściowa ciągiem pseudolosowym.*

Uwaga 1.1.2 *Aby spełniony być warunek nieodróżnialności ciągu pseudolosowego od ciągu losowego, zarodek powinien być wybierany z dużego zbioru, którego rozmiar uniemożliwia atak* brutalny *(ang.* brute force*). Ciąg pseudolosowy musi mieć również „dobre" własności statystyczne, które zapewnią nieodróżnialność w testach symulujących pewne eksperymenty losowe.*

Poufność przesyłanych lub przechowywanych danych zapewnia się zazwyczaj poprzez zastosowanie systemu szyfrującego, czyli zestawu funkcji szyfrującej i deszyfrującej. Obie te funkcje wykorzystują specjalny argument zwany

kluczem, który jest z kolei tworzony przy pomocy funkcji generującej klucze. W przypadku kryptosystemów symetrycznych ten sam klucz służy do szyfrowania i deszyfrowania wiadomości przesyłanej pomiędzy uczestnikami wymieniającymi w bezpieczny sposób wiadomości.

Definicja 1.1.4 *Systemem szyfrowania symetrycznego (kryptosystemem symetrycznym) nazywamy zestaw trzech funkcji.*

1. *Funkcji $\mathcal{G}$ będącej generatorem pseudolosowym pozwalającym tworzyć klucze kryptosystemu. Zbiór wszystkich możliwych kluczy nazywamy przestrzenią kluczy i oznaczamy K.*

2. *Funkcji szyfrującej $\mathcal{E} : K \times M \to C$, wyznaczanej przez pewną wartość klucza $k \in K$, przekształcającej wiadomość $m \in M$ na szyfrogram $c \in C$, $c = \mathcal{E}(k, m) = \mathcal{E}_k(m)$ i mającej własność nieodwracalności, tzn. bez znajomości k wyznaczenie m na podstawie c (takiego, że $\mathcal{E}_k(m) = c$) jest obliczeniowo trudne.*

3. *Funkcji deszyfrującej $\mathcal{D} : K \times C \to M$, wyznaczanej przez klucz $k \in K$, odwracającej proces szyfrowania $\mathcal{D}_k(\mathcal{E}_k(m)) = m$.*

W latach 70-tych został zdefiniowany, a następnie zrealizowany system szyfrowania asymetrycznego, wykorzystujący parę kluczy: prywatny i publiczny. Klucz publiczny $\widehat{pk}$ służy do szyfrowania danych oraz, jak wskazuje jego nazwa, może być publicznie dostępny. Odpowiadający mu klucz prywatny $\widehat{sk}$, służący do deszyfrowania pozostaje sekretem odbiorcy szyfrowanych wiadomości.

Definicja 1.1.5 *System szyfrowania asymetrycznego (kryptosystem asymetryczny) składa się z trzech funkcji:*

1. *Funkcji $\mathcal{G}$ będącej generatorem pseudolosowym pozwalającym tworzyć pary kluczy kryptosystemu: klucz $\widehat{pk}$ (zwany publicznym) służący do szyfrowania, oraz klucz $\widehat{sk}$ (zwany prywatnym) do deszyfrowania. Problem wyznaczenia klucza prywatnego na podstawie znajomości klucza publicznego jest obliczeniowo trudny.*

2. *Funkcji szyfrującej $\mathcal{E} : K \times M \to C$, wyznaczanej przez klucz publiczny $\widehat{pk} \in K$, przekształcającej wiadomość $m \in M$ na szyfrogram $c \in C$, $c = \mathcal{E}(\widehat{pk}, m) = \mathcal{E}_{\widehat{pk}}(m)$ i mającej własność nieodwracalności, tzn. bez znajomości klucza $\widehat{sk}$ wyznaczenie m na podstawie c jest obliczeniowo trudne.*

3. *Funkcji deszyfrującej* $\mathcal{D} : K \times C \to M$*, wyznaczanej przez klucz prywatny* $\widehat{sk} \in K$*, odwracającej proces szyfrowania* $\mathcal{D}_{\widehat{sk}}(\mathcal{E}_{\widehat{pk}}(m)) = m$.

W powyższej definicji założyliśmy, że przestrzeń kluczy publicznych jest taka sama jak przestrzeń kluczy prywatnych, choć ogólnie rzecz biorąc tak nie musi być. Dzięki zastosowaniu szyfrowania asymetrycznego możliwe jest rozwiązanie problemu dystrybucji kluczy – klucz publiczny może być ogólnie dostępny i rozprowadzany poprzez niezabezpieczone kanały komunikacyjne (np. Internet). Uzyskanie tak silnej własności możliwe jest dzięki zastosowaniu specjalnych struktur algebraicznych oraz oparciu bezpieczeństwa na założeniu dużej złożoności obliczeniowej pewnych problemów takich, jak faktoryzacja dużych liczb lub wyznaczanie logarytmu dyskretnego w $\mathbb{Z}_p^*$. Podobne struktury można wykorzystać do stworzenia schematu podpisu cyfrowego, którego głównym zastosowaniem jest uwierzytelnianie źródła danych oraz jednostek.

Definicja 1.1.6 *Schemat podpisu cyfrowego składa się z trzech funkcji.*

1. *Funkcji* $\mathcal{G}$ *będącej generatorem pseudolosowym pozwalającym tworzyć pary kluczy: służącego do podpisywania klucza prywatny* $\widehat{sk}$ *oraz klucza publiczny* $\widehat{pk}$ *pozwalający weryfikować podpis.*

2. *Zależnej od klucza prywatnego* $\widehat{sk}$ *funkcji tworzenia podpisu*

$$\mathcal{P} : K \times M \to S$$

takiej, że problem wyznaczenia podpisu

$$s = \mathcal{P}_{\widehat{sk}}(m)$$

bez znajomości $\widehat{sk}$ *jest obliczeniowo trudny.*

3. *Wyznaczanej przez klucz publiczny funkcji weryfikacji podpisu*

$$\mathcal{V} : K \times S \to \{0, 1\},$$

która na podstawie wiadomości m *i podpisu* s *decyduje, czy dany podpis jest prawidłowy.*

Zdefiniowane powyżej narzędzia mogą posłużyć do skonstruowania prostych protokołów kryptograficznych. Przykładem podstawowych protokołów kryptograficznych mogą być protokoły uwierzytelniania uczestnika oraz protokoły ustalania klucza.

Definicja 1.1.7 *Uwierzytelnianiem jednostek (ang. entity authentication) nazywamy protokół, w którym jedna ze stron zostaje zapewniona o tożsamości drugiej, poprzez uzyskanie weryfikowalnych przy pomocy obliczeń dowodów opartych na znajomości pewnych sekretów (kluczy). Dodatkowo protokół taki musi zagwarantować, iż żaden inny uczestnik (nie posiadający dostępu do tych sekretów) nie był w stanie podszyć się pod właściwego uczestnika protokołu.*

Definicja 1.1.8 *Ustalaniem klucza (ang. key establishment) nazywamy protokół kryptograficzny, w wyniku działania którego uczestnicy protokołu uzyskują wspólny sekret, klucz. Wymagamy przy tym, aby problem wyznaczenia takiego sekretu przez nieuprawnionego uczestnika był obliczeniowo trudny.*

Uczestników protokołów kryptograficznych oznacza się przeważnie kolejnymi wielkimi literami alfabetu $A, B, C, \ldots$ lub odnosi się do nich poprzez imiona Alice, Bob, Dave. Operację przesłania wiadomości m przez Alice do Boba możemy zapisać w uproszony sposób przedstawiony poniżej.

$$A \longrightarrow B : m$$

Uwaga 1.1.3 *W celu zapewnienia bezpieczeństwa przesyłanej informacji można ją zabezpieczyć szyfrując, podpisując lub wykorzystując inne narzędzie kryptograficzne. Mówimy wtedy, że komunikacja odbywa się przez kanał zabezpieczony. W tej pracy będziemy korzystali następujących typów zabezpieczonych kanałów komunikacyjnych.*

- *Prywatny kanał komunikacyjny – wiadomości są szyfrowane kluczem publicznym odbiorcy lub kluczem sesyjnym znanym odbiorcy w celu zapewnienia ich tajności.*

$$A \xrightarrow{prv} B : m \;\equiv\; A \longrightarrow B : \mathcal{E}(m)$$

- *Uwierzytelniony kanał komunikacyjny – wiadomości są zaopatrywane w podpis nadawcy w celu uwierzytelnienia źródła danych.*

$$A \xrightarrow{auth} B : m \;\equiv\; A \longrightarrow B : m, \mathcal{P}(m)$$

- *Uwierzytelniony kanał prywatny – połączenie obu powyższych.*

$$A \xrightarrow[prv]{auth} B : m \;\equiv\; A \longrightarrow B : \mathcal{E}(m, \mathcal{P}(m))$$

Bezpieczeństwo protokołów i algorytmów bada się analizując scenariusze, algorytmy pozwalające osobie nieupoważnionej (adwersarzowi) naruszyć poufność, integralność lub inną cechę bezpieczeństwa.

Uwaga 1.1.4 *Z bezpieczeństwem protokołów i algorytmów kryptograficznych związane jest nierozłącznie pojęcie adwersarza. Jest to nielegalny uczestnik protokołu, który w sposób bierny (podsłuchuje) lub aktywny (podszywa się pod legalnych uczestników, przesyła komunikaty) próbuje naruszyć bezpieczeństwo systemu. Możemy rozróżnić dwa poziomy bezpieczeństwa protokołów i algorytmów kryptograficznych. Z jednej strony, mogą być one bezpieczne wobec ograniczonego obliczeniowo adwersarza, tzn. takiego który potrafi rozwiązywać dowolne problemy łatwe obliczeniowo (1.1.1). Z drugiej strony, algorytm lub protokół może oferować wyższy (a właściwie najwyższy) poziom bezpieczeństwa jeżeli jest odporny na ataki adwersarza nieograniczonego obliczeniowo, tzn. takiego który potrafi rozwiązywać dowolne problemy. Protokoły i algorytmy oferujące najwyższy poziom bezpieczeństwa nazywamy bezwarunkowo bezpiecznymi, doskonale bezpiecznymi lub bezpiecznymi z punktu widzenia teorii informacji [28]. W większości zastosowań pierwszy poziom bezpieczeństwa jest uważany za wystarczający.*

1.2 Narzędzia kryptograficzne

1.2.1 Schematy zobowiązań

W niektórych zastosowaniach wymagana jest gwarancja, iż pewna tajna informacja (sekret) pozostawała niezmieniona do czasu ujawnienia. W tym celu tworzona jest specjalna wartość zwana zobowiązaniem. Funkcja tworząca zobowiązania ma własność nieodwracalności (w podobnym sensie jak w definicji 1.1.1), tzn. uzyskanie jakiejkolwiek informacji na temat sekretu na podstawie wartości zobowiązania jest trudne obliczeniowo. Ponadto twórca zobowiązania może w dowolnym momencie odkryć zobowiązanie. Polega to na ujawnieniu sekretu lub pewnych danych pozwalających go poznać. Z drugiej strony, podczas ujawniania sprawdzający zobowiązanie uzyskuje pewność, iż ujawniony sekret nie został zmieniony od czasu przekazania mu wartości zobowiązania. Zobowiązanie ma charakter wiążący, tzn. twórca zobowiązania, nie jest w stanie w efektywny sposób stworzyć dowodu, iż dane zobowiązanie dotyczy innej niż oryginalna informacja.

Definicja 1.2.1 *Schematem zobowiązania nazywamy zestaw dwóch protokołów wykonywanych przez twórcę (nadawcę S – ang.* sender*) zobowiązania oraz weryfikującego (odbiorcę R – ang.* receiver*) realizujących opisane poniżej cele.*

1. *Protokół tworzenia zobowiązania, w wyniku którego R poznaje wartość zobowiązania c dla pewnej wiadomości m. Zobowiązanie c jest wartością funkcji tworzącej zobowiązania* commit *na m oraz innych danych wejściowych, które nie są w pełni znane odbiorcy R. O funkcji tworzącej zobowiązanie zakładamy, że ma dwie własności:*

   - ukrywania *(ang.* hiding*) – wyznaczenie m na podstawie c jest problemem trudnym obliczeniowo;*
   - wiązania *(ang.* binding*) – znalezienie $m' \neq m$ oraz danych wejściowych dających to samo c jest trudne obliczeniowo.*

2. *Protokół odkrywania zobowiązania, którego celem jest dostarczenie odbiorcy wartości pozwalających na poznanie m oraz zweryfikowanie zobowiązania. Podczas odkrywania zobowiązania wykorzystywana jest funkcja odkrywająca decommit przyjmująca jako argument zobowiązanie c.*

W zależności od długości wiadomości m wyróżniamy dwa typy schematów zobowiązań: zobowiązania bitowe (ang. *bit commitment*), oraz zobowiązania wielo-bitowe (ang. *string commitment*). Dodatkowo (patrz [19]) możemy podzielić schematy na doskonale wiążące (ang. *perfectly binding*) oraz doskonale ukrywające (ang. *prefectly hiding*). W przypadku schematów zobowiązań doskonale wiążących mamy gwarancję, iż nawet nieograniczony obliczeniowo twórca zobowiązania nie jest w stanie naruszyć własności wiążącej zobowiązania i spreparować dowód na to, iż dane zobowiązanie dotyczy danych innych niż użyte w momencie tworzenia. Za nieograniczonego obliczeniowo będziemy uznawać uczestnika posiadającego zasoby pozwalające rozwiązywać problemy trudne obliczeniowo (w rozsądnym czasie). Własność doskonałego ukrywania oznacza, iż wartość zobowiązania nie zdradza żadnych informacji o tajnej informacji nawet dla posiadającego dostęp do nieograniczonej mocy obliczeniowej.

1.2.2 Kryptografia asymetryczna

Pierwszym, a przez to przełomowym systemem szyfrowania asymetrycznego był schemat RSA [61]. Podobnie jak w przypadku wielu innych systemów asymetrycznych obliczenia odbywają się w arytmetyce modularnej. Moduł n jest wyznaczany jako iloczyn tajnych liczb pierwszych p i q (w praktyce, ze względów bezpieczeństwa p i q powinny być dużymi liczbami 2048 lub 4096-bitowymi). Kluczem publicznym $\widehat{pk}$ jest para (e, n), gdzie e wybierane jest takie, że $NWD(e, \varphi(n)) = 1$ (φ oznacza funkcję Eulera, zatem w tym przypadku: $\varphi(n) = (p-1)(q-1)$). Kluczem prywatnym $\widehat{sk}$ jest element d, który

jest odwrotny do e w $Z^*_{\varphi(n)}$ ($e \cdot d = 1 \bmod \varphi(n)$). Szyfrowanie wiadomości m i deszyfrowanie kryptogramu c polega na podnoszeniu do odpowiedniej potęgi, a poprawność algorytmu wynika z twierdzenia Eulera o liczbach względnie pierwszych.

- $c = \mathcal{E}_{\widehat{pk}}(m) = m^e \bmod n$
- $m = \mathcal{D}_{\widehat{sk}}(c) = c^d \bmod n$

Innym często stosowanym kryptosystemem asymetrycznym jest schemat ElGamala. Popularność tego rozwiązania wynika z jego względnej prostoty oraz z własności, które zostaną przedstawione w dalszej części tej pracy. Niech p będzie liczbą pierwszą, g generatorem w $\mathbb{Z}_p$, natomiast x losowym elementem w $\mathbb{Z}_p$. Definiujemy klucz publiczny i prywatny systemu ElGamala jako parę $(\widehat{pk}, \widehat{sk}) = ((p, g, y), (x))$, gdzie $y = g^x \bmod p$, natomiast funkcje szyfrujące i deszyfrujące w sposób następujący:

- $\mathcal{E}_{\widehat{pk}}(m, k) = (a, b) = (m \cdot y^k \bmod p, g^k \bmod p)$, gdzie k jest wartością pseudolosową,
- $\mathcal{D}_{\widehat{sk}}((a, b)) = a \cdot b^{-x} \bmod p$.

W wyniku szyfrowania powstaje para liczb, która jest zależna od wartości pseudolosowej k. W konsekwencji tej samej wiadomości m mogą odpowiadać dwa różne szyfrogramy (a, b) i (a', b'), powstałe przy użyciu różnych wartości k i k'.

Nierozróżnialność Dzięki opisanej powyżej własności schemat ElGamala spełnia kryterium *bezpieczeństwa semantycznego* (ang. *semantic security*). Pojęcie to zostało wprowadzone w pracy [29], której autorzy pokazali również równoważność tego pojęcia z *nierozróżnialnością w ataku z wybranym tekstem jawnym* (IND-CPA – ang. *ciphertext indistinguishability with chosen plaintext*). Pojęcie IND-CPA jest częściej stosowane, jako że jego definicja jest bardziej naturalna. Mówi się, iż dany kryptosystem jest IND-CPA jeżeli adwersarz o wielomianowej mocy obliczeniowej nie jest w stanie odgadnąć z prawdopodobieństwem znacząco odbiegającym od $\frac{1}{2}$ czy dany szyfrogram c zawiera wiadomość m_1, czy też m_2. Wiadomości m_1 i m_2 są wybierane przez adwersarza, który przed poznaniem c może wygenerować dowolną (wielomianową) ilość szyfrogramów – adwersarz ma dostęp do wyroczni szyfrującej. Wyrocznia szyfrująca to maszyna wykonująca algorytm szyfrowania, która zwraca szyfrogram. Jeżeli realizowany algorytm szyfrowania wykorzystuje wartości losowe to nie są one znane użytkownikowi wyroczni. Opisana powyżej „gra" odbywa się przy ustalonej parze kluczy (adwersarz zna $\widehat{pk}$). Więcej

szczegółów oraz formalne definicje nierozróżnialności można znaleźć w pracy [28]. Przykładem kryptosystemu, który nie spełnia IND-CPA jest RSA. Wynika to z faktu, iż funkcja szyfrująca jest deterministyczna. W celu zapewnienia własności IND-CPA można zmodyfikować algorytm RSA dodając do szyfrowanych wiadomości losowe uzupełnienie (ang. padding).

1.2.3 Dowody z wiedzą zerową

Jednym z ciekawszych mechanizmów stosowanych m. in. w uwierzytelnianiu jednostek są dowody z wiedzą zerową. Dowody z wiedzą zerową są szczególnym przypadkiem protokołów typu *wyzwanie-odpowiedź* (ang. *challange-response*), w których jeden z uczestników P dowodzi, iż zna pewną wartość, odpowiadając na losowe zapytanie (ang. *challenge*) sprawdzającego V. Przy założeniu trudności pewnego problemu, na którym bazuje protokół, wiadomości ujawniane przez dowodzącego nie zdradzają żadnych informacji o tajnej wartości. Przykładem protokołu o wiedzy zerowej może, być dowód równości logarytmów dyskretnych dla dwóch par generator-wartość w $\mathbb{Z}_p$: (g, y) i (g', y'). Zadaniem strony dowodzącej P jest wykazanie, iż $log_g(y) = log_{g'}(y')$. Przebieg dowodu jest następujący [13].

1. P wybiera losowe $s \in \mathbb{Z}_p$, a następnie wylicza $(a, b) = (g^s, g'^s)$ i przesyła do V.
$$P \rightarrow V : (a, b)$$

2. V wybiera losowe $c \in \mathbb{Z}_p$ i odsyła jako wyzwanie.
$$V \rightarrow P : c$$

3. P oblicza i odsyła $r = s + cx$ $(x = log_g(y) = log_{g'}(y'))$.
$$P \rightarrow V : r$$

4. V sprawdza, czy $g^r = a \cdot y^c$, oraz czy $g'^r = b \cdot y'^c$

Powyższy dowód można w prosty sposób przerobić na dowód znajomości logarytmu dyskretnego, w szczególności klucza prywatnego x w kryptosystemie ElGamala ($y = g^x \bmod p$). W podobny sposób można przeprowadzić dowód pokazujący, bez ujawniania klucza prywatnego, że dany szyfrogram algorytmu ElGamala dopowiada danemu tekstowi jawnemu.

Nieinteraktywne dowody z wiedzą zerową Inna, dość przydatna modyfikacja protokołu może polegać na wyeliminowaniu wszelkiej interakcji pomiędzy P i V, poprzez zastąpienie kroku 2, czyli wyboru wartości c, wyliczeniem jej przez P, jako $c = h(r, a, b)$, gdzie h oznacza funkcję haszującą. Zastosowanie funkcji haszującej uniemożliwia P manipulowanie wartością c. Dzięki temu P może samodzielnie (bez interakcji i wymiany komunikatów) stworzyć dowód, przesłać go do V, który może go również samodzielnie (bez interakcji) zweryfikować. Dowód zawiera wszystkie wartość wyliczane podczas jego przeprowadzania, czyli w poprzednim przykładzie: (a, b, c, r). Tego typu dowód nazywamy nieinterakcyjnym dowodem o wiedzy zerowej (NIZK – ang. *non-iteractive zero-knowledge*), jego weryfikacja wymaga sprawdzenia poprawności kolejnych obliczeń, oraz dokonania końcowej weryfikacji z kroku 4. Zaprezentowana technika przekształcania dowodów o wiedzy zerowej w dowody nieinterakcyjne jest uniwersalna i można ją stosować również w przypadku innego dowodu o wiedzy zerowej [24].

Niech para (a, b) oznacza zaszyfrowaną algorytmem ElGamala wiadomość m, przy wykorzystaniu klucza publicznego (p, g, y). Dowód typu NIZK poprawności deszyfrowania wiadomości w kryptosystemie ElGamala można skonstruować w następujący sposób.

1. P wybiera liczbę losową $s \in \mathbb{Z}_p$, po czym wylicza $c = h(a, b, p, g, y)$ oraz parę $(e, f) = (b^s, g^s)$. Następnie wyliczane jest $r = s + cx$. Wartości r, e, f stanowią dowód poprawności deszyfrowania.

2. V w podobny sposób co P wylicza c, oraz $a' = a \cdot m^{-1}$, a następnie sprawdza, czy:
$$b^r = e \cdot a'^c, g^r = f \cdot y^c.$$

1.2.4 Schematy podziału sekretu

Schemat podziału sekretu to zestaw dwóch protokołów, z których pierwszy pozwala uczestnikowi zwanemu *dealerem* D na podział sekretu m na n wartości zwanych *cieniami*. Wartości te mogą zostać rozdane n uczestnikom zwanymi *graczami* ($P_1, \ldots, P_n$ – ang. *player*). Drugi protokół schematu pozwala k spośród wszystkich n posiadaczy cieni wyliczyć sekret początkowy m. Jeżeli przyjmiemy, że $k = n$, to schemat podziału sekretu możemy zrealizować trywialnie dzieląc m na $s_1, s_2, \ldots, s_n$ takich, że $m = s_1 \otimes s_2 \otimes \cdots \otimes s_n$ ($\otimes$ oznacza operację bitową XOR). Rozwiązanie problemu podziału sekretu dla $k < n$ pojawiło się w roku 1979 w pracy [68]. Pomysł (k, n)-progowego podziału sekretu opierał się na interpolacji Lagrange'a. Protokół takiego podziału ma następujący przebieg.

1. D dla sekretu m losowo wyznacza wielomian f rzędu $k-1$ nad ciałem skończonym $\mathbb{F}_{q^l}$ $(m < q^l)$ postaci:

$$f(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + \cdots + a_1 x + m.$$

2. D do każdego z graczy przesyła wartości wielomianu w punktach $1, \ldots, n$ para $(i, f(i))$ jest cieniem sekretu,

$$D \to P_i : (i, f(i)).$$

W celu odzyskania sekretu k spośród n graczy wyznacza wielomian interpolacyjny Lagrange'a na podstawie posiadanych par argument-wartość, a następnie wyznaczają wartość tego wielomianu w punkcie 0. Schemat (k, n) progowy podobnie jak schemat oparty na operacji bitowej XOR jest bezpieczny bezwarunkowo (1.1.4).

1.2.5 Ślepe podpisy cyfrowe

Ślepe podpisy cyfrowe są rodzajem podpisów cyfrowych, w których wiadomość przed podpisaniem jest „zaciemniana" (szyfrowana) przez odbiorcę podpisu tak, że w rezultacie podpisujący nie widzi co podpisuje. Podobnie jak tradycyjny podpis, podpis ślepy może zostać publicznie zweryfikowany. Jako przykład może posłużyć protokół oparty na algorytmie RSA wprowadzony w pracy [11]. Protokół ten składa się z następujących kroków. Odbiorca wybiera wartość losową r, i zaciemnia wiadomość m wyliczając $r^e m \mod n$, gdzie (e, n) jest kluczem publicznym podpisującego. Zaciemniona wiadomość jest wysyłana do podpisującego, który wylicza $s' = (r^e m)^d \mod n$. Wartość s' jest odsyłana drugiej stronie. Podpis $s = m^d \mod n$ jest otrzymywany poprzez wyliczenie $s' \cdot r^{-1} \mod n$.

1.2.6 Szyfrowanie homomorficzne

Jedną z ciekawszych własności szyfrów RSA i ElGamala jest fakt, iż przemnożenie szyfrogramów stworzonych przy użyciu tego samego klucza daje szyfrogram przemnożonych tekstów jawnych. Własność tego typu nazywać będziemy $(\cdot, \cdot)$-*homomorfizmem*.

$$\mathcal{E}_{\widehat{pk}}(m_1) \cdot \mathcal{E}_{\widehat{pk}}(m_2) = \mathcal{E}_{\widehat{pk}}(m_1 \cdot m_2)$$

Zaletą tej własności jest możliwość wykonywania obliczeń, w tym przypadku mnożenia, na zaszyfrowanych danych. W kontekście wyborów elektronicznych bardziej praktyczny jest tak zwany $(\cdot, +)$-*homomorfizm*, który pozwala

sumować zaszyfrowane liczby. Prosta modyfikacja systemu ElGamala pozwala na uzyskanie tej cechy przy pewnych założeniach. Zmiana polega na zastąpieniu szyfrogramu potęgą, której wykładnikiem jest szyfrowana wiadomość, a podstawą generator h pewnej podgrupy $\mathbb{Z}_p$.

$$\hat{\mathcal{E}}_{\widehat{pk}}(m) = (h^m y^k, g^k) \pmod{p}$$

Deszyfrowanie odbywa się w analogiczny sposób jak w tradycyjnym schemacie ElGamala. Pozwala nam to uzyskać $h^m \bmod p$. Aby na tej podstawie wyznaczyć m musimy założyć, iż m jest małą liczbą lub h generuje małą podgrupę rzędu c w $\mathbb{Z}_p$ (otrzymamy $m \bmod c$). W tym drugim przypadku mówimy o $(\cdot, +_c)$-*homomorfiźmie*.

Kryptosystem Pailliera

Jeżeli potrzebny jest kryptosystem, który oferuje $(\cdot, +)$-*homomorfizm* bez żadnych dodatkowych ograniczeń, możemy skorzystać z schematu Paillera [56]. Podobnie jak w przypadku RSA, wykorzystywana jest liczba $n = pq$, będąca iloczynem dwóch dużych liczb pierwszych. Niech g będzie generatorem $\mathbb{Z}^*_{n^2}$ takim, że $g = 1 \bmod n$, $\lambda = NWW(p-1, q-1)$ oraz $L(x) = (x-1)/n$. Role kluczy publicznego i prywatnego pełnią odpowiednio: (g, n) i λ. Funkcje szyfrująca i deszyfrująca mają następującą postać.

- $\mathcal{E}_{(g,n)}(m, r) = g^m \cdot r^n \bmod n^2$, gdzie r jest wartością losową z $\mathbb{Z}^*_n$.
- $\mathcal{D}_{(\lambda)}(c) = \frac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n$

1.2.7 Szyfrowanie progowe

Własności algebraiczne kryptosystemu ElGamala umożliwiają stosunkowo łatwe zrealizowanie *kryptosystemu progowego* (ang. *threshold cryptosystem*) bazującego na schemacie dzielenia sekretu Shamira [22]. Kryptosystem progowy jest protokołem, w którym k spośród n uczestników jest w stanie odszyfrować wiadomość m zaszyfrowaną kluczem publicznym $y = g^x \bmod p$, gdzie x jest odpowiednim kluczem prywatnym. Aby tego dokonać uczestnicy muszą otrzymać własne klucze prywatne (U_i otrzymuje x_i), które są cieniami klucza prywatnego x. Odszyfrowanie w tym systemie progowym kryptogramu ElGamala postaci (a, b) wymaga obliczenia:

$$m = \frac{a}{\prod b^{x_i L_i(0)}},$$

$L_i(0)$ oznacza współczynniki Lagrange'a dla wartości 0 (patrz 1.2.4). Nietrudno zauważyć, iż powyżej zaprezentowany kryptosystem wymaga istnienia zaufanego uczestnika (dealera), który tworzy klucz prywatny oraz dystrybuuje jego cienie. Inne podejście, nie wymagające istnienia zaufanego użytkownika zostało zaproponowane w pracy [58]. Idea tego rozwiązania polega na tym, iż każdy z uczestników wybiera własny element klucza x_i, a następnie dostarcza jego cienie pozostałym uczestnikom. Dzięki temu dowolny podzbiór uczestników o mocy k jest w stanie odzyskać poszczególne klucze prywatne, a właściwie ich iloczyn, czyli globalny klucz prywatny x.

1.2.8 Kleptografia

Autorzy [74, 75] zaobserwowali w latach 90-tych, że wykorzystanie urządzeń mających własność czarnych skrzynek niesie ze sobą poważne niebezpieczeństwa. Urządzenia te charakteryzują się zabezpieczeniami utrudniającymi w znaczący sposób fizyczny dostęp do ich wnętrza. Przez to poznanie ich konstrukcji, stanu pamięci oraz oprogramowania jest trudne i kosztowne. Za przykład urządzeń o charakterystyce czarnej skrzynki mogą posłużyć karty chipowe. Okazuje się, iż zastosowanie tego typu urządzeń do realizacji celów kryptograficznych może mieć katastrofalne konsekwencje dla użytkowników. Wynika to faktu, iż twórca takich urządzeń może umieścić w nich złośliwy kod, który będzie powodował niezauważalny wyciek danych. Innymi słowy urządzenie może się zachowywać zgodnie ze specyfikacją, i jednocześnie być źródłem cennych informacji dla swojego twórcy. Tego typu kleptograficzne ataki wykorzystują przede wszystkim wszelkie wartości losowe, które mogą zostać zastąpione asymetrycznie zaszyfrowanymi tajemnicami. Wymaga to zapisania w urządzeniu klucza publicznego, dla którego odpowiedni klucz prywatny znany jest tylko nieuczciwemu twórcy.

Przykładem schematu podpisu cyfrowego podatnego na ataki kleptograficzne może być RSA. Z tego względu, iż algorytm podpisywania RSA nie wykorzystuje wartości losowych może się wydawać, że mamy do czynienia ze schematem odpornym na zagrożenia kleptograficzne. Tak nie jest, co zostało pokazane w pracy [74], gdzie wykorzystano własności algorytmu generowania kluczy dla RSA. Algorytm ten można „złośliwie" zmodyfikować w taki sposób, aby poprzez odpowiednie dobieranie q otrzymać publiczny moduł $n = pq$, który na najbardziej znaczących bitach zawiera zaszyfrowaną liczbę p. Sposób uniknięcia tego zagrożenia został pokazany w pracy [76] poprzez wprowadzenie weryfikowalnego generowania kluczy RSA. Wymaga ono jednak założenia, iż wszyscy użytkownicy urządzeń współdzielą ogólnie uznany publiczny wykładnik e, przeważnie $e = 2^{16} + 1$.

1.3 Problem bezpiecznej platformy

Wiele złożonych systemów informatycznych charakteryzuje się tym, iż część z funkcjonalności systemu realizowana jest przez program działający w środowisku, które nie może być w pełni kontrolowane przez twórcę systemu lub przez dostawcę usług świadczonych przez ten system. Bardzo często owe operacje wykonywane w niezaufanym środowisku są istotne z punktu widzenia bezpieczeństwa i wymagana jest gwarancja, iż przebiegły poprawnie bez ingerencji środowiska lub innego programu działającego w tymże środowisku uruchomieniowym. Z sytuacją tego typu mamy do czynienia w przypadku wyborów elektronicznych przez Internet. Komisja wyborcza nie może mieć pewności, iż głos wyborcy przed zaszyfrowaniem i wysłaniem nie został zmieniony przez złowrogie oprogramowanie, którym zainfekowany jest komputer osobisty wyborcy. Podobna sytuacja ma miejsce w przypadku systemów mobilnych agentów [53], które w założeniu mają być autonomicznymi programami działającymi na rzecz swoich użytkowników przemieszczając się po platformach świadczących pewne usługi (również komercyjne).

Jednym z klasycznych scenariuszy wykorzystania mobilnych agentów jest dokonywanie przez nich zakupów w imieniu klienta. W tym celu agenci wyposażeni w preferencje klienta oraz algorytm podejmowania decyzji przemieszczają się po platformach różnych sklepów i zapoznają się z ofertą. Nietrudno zauważyć, iż platforma mogłaby spróbować wykorzystać swoją przewagę, przeanalizować zasoby agenta i zaproponować minimalną ofertę spełniającą jego oczekiwania. Przykład ten pokazuje konieczność zachowania tajności zasobów oraz wyników działania algorytmów agentów, jako warunków bezpieczeństwa obliczeń. Dla problemu bezpieczeństwa mobilnych agentów software'owych zaproponowano kilka rozwiązań, które wykorzystują między innymi opisane poniżej narzędzia teoretyczne.

Szyfrowanie funkcji Wyliczenie we wrogim środowisku krytycznej dla bezpieczeństwa wartości funkcji f może odbywać się w zaszyfrowanej postaci. Naturalną metodą szyfrowania funkcji jest złożenie jej z inną, odwracalną funkcją g [65, 66]. Odszyfrowanie funkcji lub obliczanie wartości f dla pewnego argumentu można dokonać stosując funkcję g^{-1} ($g^{-1} \circ g \circ f = f$). Funkcje wymierne, czyli będące ilorazem wielomianów, są przykładem klasy funkcji, które pozwalają na efektywne składanie i odwracanie. Bezpieczeństwo tego rozwiązania wynika z faktu, że problem znajdowania f mając dane $g \circ f$ jest uważany za obliczeniowo trudny. Autorzy prac [65, 66] proponują specjalny schemat podpisów cyfrowych zwanych *nieodłączalnymi podpisami cyfrowymi* (ang. *undetachable digital signatures*), które przy pewnych ograniczeniach pozwalają na bezpieczne podpisywanie przez mobilnych agentów wiadomo-

ści i dokumentów. Realizację tej propozycji opartą na schemacie RSA można znaleźć w pracy [7].

Sieci boolowskie Dla każdej funkcji n-argumentowej f, która może być przedstawiona jako sieć boolowska istnieją protokoły, które umożliwiają rozproszone obliczenia i zapewniają, iż żaden z uczestników nie poznaje argumentów wejściowych innych niż te, które pochodzą od nich samych [70, 3]. Rezultat obliczeń może być zarówno współdzielony przez uczestników protokołu, jak i każdy z uczestników może poznać tylko część wyniku. Podstawą tego typu protokołów jest idea *zaciemnionych sieci boolowskich* (ang. *garbled circuits*) wprowadzona przez Yao [73]. Wadą tego typu sieci są trudności związane z implementacją tego rozwiązania – przetwarzanie jednej bramki logicznej wymaga odczytania wartości w tablicy.

Szczególnie ważne, w kontekście tej rozprawy doktorskiej, jest zastosowanie szyfrowania homomorficznego opartego na trzecim rozwiązaniu.

Homomorficzne pierścienie Teoretycznie tajność obliczeń może zostać osiągnięta poprzez zastosowanie dwóch pierścieni R_1 oraz R_2 [65, 66] i wykorzystaniu homomorfizmu $\mathcal{E} : R_1 \to R_2$, który ma własność transformacji szyfrującej – trudno jest odwrócić $\mathcal{E}$ bez znajomości pewnej tajnej informacji (patrz 1.2.6). Funkcja $\mathcal{E}$ z definicji zachowuje operacje pierścienia: $\mathcal{E}(x+y) = \mathcal{E}(x) + \mathcal{E}(y)$, $\mathcal{E}(x \cdot y) = \mathcal{E}(x) \cdot \mathcal{E}(y)$, co oznacza, że wszystkie obliczenia można przeprowadzać na zaszyfrowanych danych (w pierścieniu R_2). Niestety, trudno jest znaleźć funkcję $\mathcal{E}$, dla której istnieją dwa efektywne algorytmy *plus* i *mult* pozwalające wykonywać operacje dodawania i mnożenia na zaszyfrowanych danych:

- $plus(\mathcal{E}(a), \mathcal{E}(b)) = \mathcal{E}(a+b)$
- $mult(\mathcal{E}(a), \mathcal{E}(b)) = \mathcal{E}(a \cdot b)$.

W szczególnym przypadku, wielomianów postaci:

$$p(x_1, \ldots, x_n) = \sum a_{i_1,\ldots,i_n} x_1^{i_1} \cdot \ldots \cdot x_n^{i_n},$$

możemy jednak wykorzystać zmodyfikowaną parę algorytmów *plus* i *mixmult*. Drugi z algorytmów pozwala pomnożyć element zaszyfrowany z niezaszyfrowanym: $mixmult(\mathcal{E}(a), b) = \mathcal{E}(a \cdot b)$. Aby zrealizować *mixmult* w przypadku algorytmów szyfrujących opisanych w 1.2.6 należy podnieść szyfrogram $\mathcal{E}(a)$ do potęgi b. Należy zwrócić uwagę, iż po zastąpieniu algorytmu *mult* przez *mixmult* nie jest możliwe wykonanie obliczeń na zaszyfrowanych danych. Możliwe jest jednak stworzenie przez jedną stronę wielomianu, zaszyfrowanie jego współczynników oraz przesłanie go drugiej innemu uczestnikowi.

Ten może z kolei obliczyć zaszyfrowaną wartość tego wielomianu dla sobie znanego argumentu x, a następnie ją odesłać.

Rozdział 2

Sieci mieszające

Poniższy rozdział poświęcony jest sieciom mieszającym, jednemu z najważniejszych narzędzi pozwalających zapewnić odpowiedni poziom anonimowości dla uczestników wielu protokołów kryptologicznych. Zaprezentowane zostaną metody weryfikujące działanie sieci mające na celu również zagwarantowanie integralności wiadomości przesyłanych przez sieć. Szczególna uwaga zostanie poświęcona procedurze częściowego sprawdzania RPC (opisanej szczegółowo w punkcie 2.3). Procedura ta polega na ujawnieniu połowy przejść wykonywanych przez serwery mieszające, co w oczywisty sposób osłabia anonimowość sieci.

Początek rozdziału będzie poświęcony charakterystyce sieci mieszających oraz przeglądowi różnych metod zapewniania integralności wiadomości. W dalszej części zaprezentowana zostanie procedura sprawdzająca RPC oraz zostanie omówiony wpływ jej zastosowania na anonimowość sieci. Końcowa część rozdziału zawiera propozycje autorskich dwuetapowych procedur sprawdzających typu RPC wraz z omówieniem ich wpływu na zapewnienie anonimowości i integralności sieci mieszającej. Procedury tego typu będą wykorzystywane w protokołach wyborów elektronicznych oraz e-egzaminów opisanych w kolejnych rozdziałach.

2.1 Charakterystyka sieci mieszających

Jednym z kluczowych problemów współczesnych systemów informatycznych jest zapewnienie anonimowości przesyłanych lub przetwarzanych danych. Cel ten pozwalają osiągnąć sieci mieszające (ang. *mixing networks* lub *mixnets*). Sieci te pozwalają na przemieszanie list zaszyfrowanych wiadomości w rozproszony sposób przez λ zaufanych uczestników (serwerów mieszających). Każdy kolejny serwer M_i, $i = 1, 2, \ldots, \lambda$, permutuje i przekształca elemen-

ty listy. Lista wynikowa jest przekazywana następnemu serwerowi mieszającemu poprzez uwierzytelniony kanał publiczny (BB, ang.bulletin board). Oznacza to, że listy wejściowe i wyjściowe zaszyfrowanych wiadomości są publicznie dostępne wraz z podpisem BB poświadczających ich autentyczność. Przekształcenia wykonywane przez pojedynczy serwer mają za zadanie ukrycie związków pomiędzy elementami list wejściowych i wyjściowych. Z tego względu zakładamy, że wyznaczenie tajnej permutacji pojedynczego serwera, a w konsekwencji globalnej permutacji całej sieci, jest obliczeniowo trudne.

Na wejściu sieci mieszającej mamy $N = 2n$ przesyłanych wiadomości, które utożsamiać możemy z $2n$ oznaczonymi zewnętrznie kulami o pewnej zawartości – możemy przyjąć, że te zewnętrzne oznaczenia określają nadawcę. Zadaniem sieci mieszającej jest, przy zachowaniu niezmienionej zawartości kul (zachowanie integralności wiadomości, podrozdział 2.2), „wymazanie" tych oznaczeń zewnętrznych. Zastosowanie nieznanej losowej permutacji (globalnej permutacji całej sieci) – złożenia losowych permutacji używanych przez poszczególne serwery powinno gwarantować realizację tego ostatniego celu – innymi słowy, chcemy by obserwator zewnętrzny (atakujący) nie był w stanie określić nadawcy żadnej wiadomości. Ponieważ nie możemy niestety wykluczyć prób oszustwa przez serwery, potrzebne są procedury weryfikujące poprawność działania sieci mieszających, gwarantujące nam zachowanie integralności przesyłanych wiadomości. Przykładem takiej procedury jest omawiana w podrozdziale 2.3 RPC. Kosztem jej zastosowania może być wyciek pewnych informacji na temat losowej permutacji omówiony w podrozdziale 2.4.

Aby przeprowadzić rozproszone mieszanie uczestnicy protokołu wykorzystują dwie funkcje.

- $onion(m, k)$ – tworzy (używając pewną wartość losową k) początkową, zaszyfrowaną formę wiadomości m, tzw. cebulę (ang. onion), która przechodzi przez serwery mieszające;

- $trans_i(c, k)$ – funkcja transformująca i-tego serwera, która przekształca szyfrogram c w c' w taki sposób, że c' i c są szyfrogramami tej samej wiadomości, ale stwierdzenie tego faktu bez znajomości wartości losowej k jest obliczeniowo trudne.

Należy zwrócić uwagę na fakt, iż przeważnie wyżej wymienione funkcje są zależne od kluczy publicznych i prywatnych $(\widehat{pk}_i, \widehat{sk}_i)$ posiadanych przez serwery mieszające. W zależności od doboru funkcji $onion$ i $trans$ możemy wyróżnić dwa główne rodzaje sieci mieszających.

- Sieci częściowo deszyfrowujące (ang. *partially decrypting mixnets*), które charakteryzuje fakt, iż serwery mieszające częściowo deszyfrują elementy listy. Ostatni serwer dokonuje odszyfrowania, w wyniku którego powstaje lista tekstów jawnych wiadomości. Poniżej zostały przedstawione popularne realizacje sieci częściowo deszyfrujących oparte na różnych kryptosystemach asymetrycznych.

  ▷ Sieci mieszające Chauma, które zostały zaproponowane w 1981 roku w pracy [14], a następnie wykorzystane w systemie wyborczym [12]. Cebule są tworzone poprzez wielokrotne szyfrowanie algorytmem RSA (patrz 1.2.2) z dopełnianiem wartościami losowymi r_i, $i = 1, 2, \ldots, \lambda$,

  $$onion(m, (r_1, \ldots, r_\lambda)) = \mathcal{E}_{\widehat{pk}_1}(r_1, \mathcal{E}_{\widehat{pk}_2}(r_2, \ldots \mathcal{E}_{\widehat{pk}_\lambda}(r_\lambda, m) \ldots)).$$

  Zapis $\mathcal{E}_{\widehat{pk}_i}(r_i, m')$ oznacza szyfrowanie przy pomocy algorytmu RSA kluczem $\widehat{pk}_i$ złączenia wartości losowej r_i oraz m'. Przetwarzanie wiadomości wymaga odszyfrowania kluczem prywatnym i-tego serwera kolejnej warstwy cebuli. Zastosowanie algorytmu RSA z dopełnianiem wartościami losowymi powoduje wydłużanie się cebuli wraz z kolejnym szyfrowaniem oraz skracanie się cebul podczas przetwarzania ich przez serwery.

  ▷ Sieć mieszająca oparta na kryptosystemie ElGamala, która charakteryzuje się niezmienną długością cebul, jest realizowana przy pomocy opisanych poniżej funkcji.

  * $onion(m, k) = \mathcal{E}_y(m) = (m \cdot y^k, g^k)$ (gdzie $y = y_1 \cdot y_2 \cdot \ldots \cdot y_\lambda$),
  * $trans_i((a, b), k) = (a(y_{i+1}y_{i+2}\ldots y_\lambda)^k / b^{x_i}, b \cdot g^k)$,

  gdzie $trans_i$ jest funkcją transformującą, a (x_i, y_i) jest parą asymetrycznych kluczy serwera M_i $(i = 1, 2, \ldots, n)$. Para (a, b) jest szyfrogramem kryptosystemu ElGamala (patrz 1.2.2).

- Sieci przeszyfrowujące (ang. *re-encrypting mixnets*), które bazując na własnościach funkcji szyfrującej pozwalają bez znajomości tajnego klucza w losowy sposób zmienić postać szyfrogramu pozostawiając jego zawartość niezmienioną. Tego typu funkcję przeszyfrowującą można wykorzystać jako funkcję $trans$. Przykładem realizacji tego typu sieci jest sieć przeszyfrowująca oparta o algorytm ElGamala [57]. Kluczem publicznym wykorzystywanym zarówno przy tworzeniu cebul jak i przy ich przetwarzaniu przez kolejne serwery jest $y = \prod y_i$. Wiadomości przetworzone przez sieć mieszająca pozostają przez cały proces mieszania

w postaci zaszyfrowanej, zabezpieczonej kluczem prywatnym $x = \prod x_i$. Z tego względu ostatnim etapem działania sieci mieszającej jest odszyfrowywanie wiadomości wyjściowych przez współpracujące serwery mieszające. Proces ten może wymagać udziału wszystkich serwerów lub jedynie pewnej ich podgrupy (patrz szyfrowanie progowe 1.2.7).

$$onion(m, k) = \mathcal{E}_y(m, k)$$

$$trans(c, k) = c \cdot \mathcal{E}_y(1, k') = \mathcal{E}_y(m, k) \cdot \mathcal{E}_y(1, k') = \mathcal{E}_y(m \cdot 1, k + k')$$

W dalszej części pracy będziemy zakładać, że losowość permutacji generowanych przez uczciwe serwery mieszające nie budzi wątpliwości. Będziemy te permutacje traktować jako w pełni losowe. W praktyce niewłaściwe wykorzystanie generatorów pseudolosowych może również stanowić zagrożenie dla anonimowości sieci mieszającej.

2.2 Zapewnianie integralności wiadomości

Sieć mieszająca może być użyta w wielu ważnych zastosowaniach takich jak wybory elektroniczne, pod warunkiem zagwarantowania integralności procesu mieszania i przetwarzania wiadomości przez serwery. Zapewnienie tej własności zwanej *odpornością* (ang. *robustness*) nie jest problemem trywialnym. Wynika to z własności funkcji *trans*, która zaciera związki pomiędzy cebulami wejściowymi i wyjściowymi, umożliwiając tym samym podmianę cebul. Odporność może zostać zapewniona poprzez dodatkowe sprawdzanie „uczciwości" serwerów.

David Chaum w pionierskiej pracy [14] zaproponował osiągnięcie tego celu poprzez dwukrotne użycie tej samej sieci mieszającej. Za pierwszym razem przesyłany jest klucz publiczny $\widehat{pk}_i$, a następnie szyfrogram (zawierający właściwą wiadomość m_i) powstały przy użyciu odpowiadającego mu klucza prywatnego $\mathcal{E}_{sk_i}(m_i)$. W rezultacie każdy przesyłający wiadomość może po pierwsze sprawdzić, czy wybrany przez niego klucz publiczny jest na wyjściu z sieci mieszającej, po drugie może zweryfikować $\mathcal{E}_{\widehat{sk}_i}(m_i)$. Podejście to ma jednak pewne wady. Nie da się go przenieść na inne typy sieci mieszających, gdyż wykorzystuje cechę algorytmu szyfrowania RSA, która pozwala odwrócić role klucza publicznego i prywatnego. Poza tym, autorzy pracy [59] wykazali, że atakujący może w drugim etapie dla zadanej cebuli wejściowej z pierwszego etapu spreparować odpowiednią cebulę, która pozwoli namierzyć jej pozycję wyjściową, a tym samym pozwoli naruszyć anonimowość jednego z uczestników protokołu. Ta ostania wada wynika z tego, że weryfikacja sieci

ma charakter indywidualny – każdy z uczestników weryfikuje jedynie przesyłanie własnej wiadomości, a zatem inni uczestnicy nie mogą weryfikować pozostałych ścieżek pokonywanych przez wiadomości innych użytkowników (w tym nieuczciwych). Nie jest zatem możliwe zweryfikowanie całego procesu przetwarzania przez wykwalifikowaną do tego organizację lub organizacje. Przykład ten pokazuje potrzebę uniwersalnych metod weryfikacyjnych, tzn. takich, w których możliwa jest weryfikacja całego procesu przez jednego lub wielu z osobna uczestników.

Propozycja uniwersalnej metody weryfikacji pojawiła się w pracy [43]. Pomysł dotyczył sieci mieszających opartych o algorytm ElGamala (punkt 1.2.2) i polegał na zastosowaniu dowodu typu zapytanie-odpowiedź (ang. *challenge-response*). W tym celu wymagany jest drugi przebieg sieci mieszającej z zastosowaniem niezależnie dobranych permutacji. Listy cebul wejściowych i wyjściowych dla danego serwera w obu przebiegach są publicznie znane, w przeciwieństwie do permutacji π pierwszego przebiegu oraz ciągu wartości losowych (k_j) wykorzystywanych przy przeszyfrowywaniu. Nie są znane również: permutacja π' wykorzystana w drugim przebiegu oraz wartości losowe (k'_j). Weryfikacja sieci polega na odpytaniu każdego z serwerów poprzez wylosowanie dla niego bitu. Jeżeli bit ma wartość 0 serwer jest zobowiązany ujawnić π' oraz (k'_j), w przeciwnym wypadku $\pi' \circ \pi^{-1}$ oraz $(k_j - k'_j)$. Jak widać nigdy nie jest wymagane ujawnienie π, ani elementów ciągu (k_j), co pozwala zachować anonimowość pierwszego przebiegu. Niestety wymaganie dodatkowego przebiegu uznane jest za zbyt dużą niedogodność, której pozbawiona jest kolejna propozycja zapewnienia integralności sieci mieszającej.

Zaproponowana w kolejnej dekadzie metoda weryfikacji częściowego, losowego sprawdzania [38] (ang. *randomized partial checking* – RPC) pozwoliła na wyeliminowanie dodatkowego „kosztu" związanego z wielokrotnym wykonaniem protokołu mieszania w celu weryfikacji sieci. Metoda ta polega na ujawnieniu przez każdy z serwerów połowy przejść, w taki sposób, iż mamy gwarancję nie ujawnienia pełnej ścieżki dla żadnej z wiadomości. Dodatkową zaletą tej metody jest jej uniwersalność pozwalająca na zastosowanie niezależnie od typu sieci oraz wykorzystanego systemu szyfrującego. Dzięki tym cechom procedura ta znalazła szerokie zastosowanie praktyczne, będzie również wykorzystywana, rozwijana i analizowana w dalszej części tej pracy.

Warto w tym miejscu wspomnieć pomysł na dowód poprawności mieszania przy wykorzystaniu dowodów z wiedzą zerową (opisanych w punkcie 1.2.3) oraz tzw. sieci sortujących (lub sieci motylkowych – ang. *butterfly*) [1, 37]. Sieci te charakteryzują się tym, iż każdy z serwerów może przekształcić daną pozycję wejściową na jedną z dwóch (lub kilku) pozycji wyjściowych. Jednocześnie sieć jest na tyle głęboka, iż każda z pozycji wejściowych sieci

może osiągnąć dowolną pozycję wyjściową. Serwery po przetworzeniu wiadomości publikują nieinteraktywne dowody z wiedzą zerową dla każdej pozycji wejściowej, iż przeszła ona na jedną z dwóch możliwych pozycji wyjściowych (dowód alternatywy). Nie jest zatem wymagane zapytanie (ang. *challenge*) uczestnika weryfikującego. Co więcej, weryfikacja obejmuje każde przejście. Stanowi to niewątpliwą zaletę tej metody sprawdzania mieszania. Z drugiej strony, duża ilość publikowanych dowodów, które są stosunkowo długie oraz fakt, iż ilość serwerów mieszających zależy od ilości wiadomości znacznie ograniczają praktyczne zastosowania.

2.3 Procedura częściowego sprawdzania

Randomized Partial Checking (RPC) jest przykładem prostej i efektywnej metody weryfikacji [38]. W ramach RPC serwery mieszające są zobligowane do ujawnienia losowo wybranej połowy powiązań pomiędzy elementami list wejściowych i wyjściowych w specjalny sposób. Serwery są łączone w pary, a następnie są zmuszane do odkrycia połowy swoich przejść. Dla pierwszego serwera w parze wskazywana jest losowa połowa pozycji wyjściowych, dla których jest zobowiązany udowodnić poprawność transformacji. Drugi serwer w parze ma dokonać tego samego, ale dla pozostałych (dopełniających) pozycji, które w jego przypadku są pozycjami wejściowymi. Dzięki podziałowi zbioru pozycji dla pierwszego i drugiego serwera mamy gwarancję, iż ujawnione w ramach pary przejścia nie tworzą ścieżki długości 2. Jest to kluczowa cecha procedury RPC, pozwalająca (niezależnie od liczby serwerów i wiadomości) zagwarantować nieujawnianie pełnych ścieżek pokonywanych przez wiadomości podczas weryfikowania odporności sieci mieszającej.

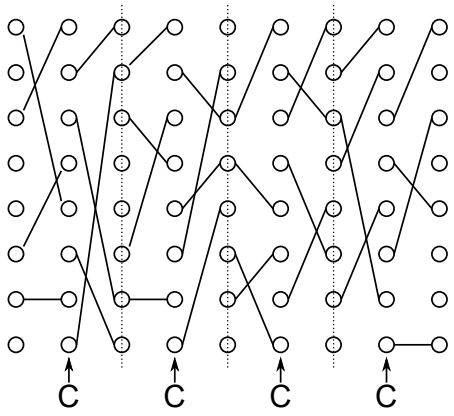
Dokładniej, procedura RPC składa się z następujących kroków [38].

1. *(Przed mieszaniem)* Serwery $M_1, M_2, \ldots, M_\lambda$ publikują zobowiązania do swoich permutacji

   $$pcommit(\pi_i) = (commit(\pi_i(1)), \ldots, commit(\pi_i(N))),$$

   gdzie π_i oznacza permutację N-elementowego zbioru $\{1, 2, \ldots, N\}$ wykorzystywaną przez M_i,

2. *(Po mieszaniu)* Serwery ustalają wartość $r = r_1 \oplus r_2 \oplus ... \oplus r_\lambda$ ($\oplus$ oznacza bitową operację XOR). Wartość r jest wybierana w uczciwy sposób, tzn. tak aby żaden z uczestników nie był w stanie zdeterminować r. W tym celu każdy serwer M_i publikuje zobowiązanie do wartości r_i przed jej ujawnieniem. W kolejnym kroku na podstawie wartości r i aktualnego



Rysunek 2.1: Procedura RPC na przykładzie 4 par serwerów (literą C zostały oznaczone wejścia i wyjścia przekształceń, dla których obowiązuje zasada komplementarności)

stanu BB wyznaczane są podzbiory $\frac{N}{2}$-elementowe przejść do ujawnienia dla poszczególnych par serwerów. Wybrany dla danej pary podzbiór określa pozycje wyjściowe do ujawnienia przez pierwszy serwer w parze. Dopełnienie tego zbioru będzie wyznaczało pozycje wejściowe do ujawnienia przez drugi serwer w parze. Aby udowodnić prawidłowość wskazanego przejścia j-tej pozycji listy wejściowej, serwer M_i publikuje wartość $validator(i, j)$, która może się składać z $decommit(\pi_i(j)), k_{ij}$, gdzie k_{ij} jest wartością losową wykorzystywaną w i-tej transformacji j-tego wejścia listy.

Autorzy procedury Randomized Partial Checking [38] zauważają, iż w przeciwieństwie do prostej metody weryfikacyjnej polegającej na niezależnym wyborze połowy przejść do ujawnienia, ich metoda oparta na na wyborach zależnych w parach serwerów nie wymaga logarytmicznej (względem liczby wiadomości) liczby serwerów mieszających dla zapewnienia anonimowości. Dzięki nieujawnianiu pełnych ścieżek przez pary serwerów nie jest konieczne zwiększanie liczby serwerów w celu zmniejszenia prawdopodobieństwa ujawnienia pełnej ścieżki dla jakiejkolwiek wiadomości. Pomimo tego, iż w pracy [31] wykazany został pewien wyciek informacji o permutowanych wiadomościach przez procedurę RPC, autorzy konkludują, że nawet przy stałej liczbie serwerów mieszających wyciek ten jest pomijalny.

2.4 Wpływ procedury częściowego sprawdzania na poziom anonimowości

2.4.1 Definicje anonimowości

Sformalizowanie pojęcia bezpieczeństwa sieci mieszających oraz innych systemów anonimizujących wymaga ścisłych definicji anonimowości. W kontekście technik anonimizujących, które zacierają powiązania pomiędzy nadawanymi i odbieranymi wiadomościami poprzez wykonywanie utajnionego mieszania wiadomości, pojęcie anonimowości jest ściśle związane z pojęciem losowej permutacji. Losową permutacją Π nazywamy element ze zbioru S_N, gdzie S_N oznacza zbiór wszystkich permutacji N elementów, wybrany zgodnie z pewnym zadanym na S_N rozkładem prawdopodobieństwa. Zostało zaproponowanych wiele definicji anonimowości, zarówno intuicyjnych jak i bardziej formalnych [20, 16, 5, 32]. Załóżmy, iż pewien system anonimizujący został oparty o permutację losową Π, będącą wektorem zmiennych losowych

$$\Pi = (\Pi(1), \Pi(2), \ldots, \Pi(N)) = (P_1, P_2, \ldots, P_N),$$

wówczas w literaturze najczęściej spotykamy się z dwoma podanymi niżej definicjami.

Definicja 2.4.1 (Anonimowość i-tego wejścia systemu) *Niech w_i oznacza i-ty element wejścia. Definiujemy poziom anonimowość i-tego elementu jako entropię rozkładu brzegowego P_i*

$$S(w_i) = -\sum_{j=1}^{N} \Pr\{\Pi(i) = j\} \cdot \ln\left(\Pr\{\Pi(i) = j\}\right).$$

Definicja 2.4.2 (Słaba anonimowość) *Poziom słabej anonimowość systemu definiujemy jako średnią entropię rozkładów brzegowych P_i*

$$S = \frac{1}{N} \sum_{i=1}^{N} S(w_i).$$

Chociaż podana wyżej definicja jest naturalna, nie uwzględnia ona zależności pomiędzy elementami wyjściowymi (zmienne losowe $P_1, P_2, \ldots, P_N$ są w oczywisty sposób silnie skorelowane). Zauważmy, że zgodnie z tą definicją system oparty o losowe cykliczne przesunięcie charakteryzować się będzie dużą anonimowością. Z drugiej strony wiedza obserwatora o charakterze tej permutacji może uczynić system całkowicie bezużytecznym. Innymi słowy, w tej definicji nie została wzięta pod uwagę częściowa wiedza obserwatora

na temat związków pomiędzy elementami wejściowymi i wyjściowymi. Z tego względu w literaturze rozpatrywane jest również pojęcie silnej anonimowości, oparte o metrykę całkowitego wahania (odległość pełnego wahania – ang. *total variation distance*) [39].

Definicja 2.4.3 (Metryka całkowitego wahania) *Niech X i Y oznaczają zmienne losowe o wartościach w dyskretnym zbiorze $\mathcal{W}$, a $\mu = \mathcal{L}(X)$ i $\nu = \mathcal{L}(X)$ będą ich rozkładami prawdopodobieństwa. Metryka całkowitego wahania dla rozkładów μ, ν zdefiniowana jest jako*

$$d_{TV}(\mu, \nu) = \frac{1}{2} \sum_{z \in \mathcal{W}} |\Pr\{X = z\} - \Pr\{Y = z\}|.$$

Metryka całkowitego wahania jest z reguły definiowana, w sposób bardziej ogólny, przy pomocy kresu górnego różnic prawdopodobieństw po zdarzeniach σ-algebry $\mathcal{F}$

$$d_{TV}(\mu, \nu) = \sup_{A \in \mathcal{F}} |\Pr\{X \in A\} - \Pr\{Y \in A\}|.$$

Prosty dowód równoważności podanych wyżej definicji metryki całkowitego wahania w przypadku dyskretnym można znaleźć w wykładzie [49]. Podobnie jak monografii [39] będziemy używać uproszczonego zapisu dla

$$d_{TV}(X, Y) = d_{TV}(\mathcal{L}(X), \mathcal{L}(Y))$$

lub zapisu hybrydowego

$$d_{TV}(X, \mathcal{L}(Y)) = d_{TV}(\mathcal{L}(X), \mathcal{L}(Y)).$$

Niech Π oznacza permutację losową reprezentującą wiedzę obserwatora po wykonaniu pewnego protokołu anonimizującego, natomiast Π_{ap} oznacza permutację losową reprezentującą wiedzę a priori obserwatora.

Definicja 2.4.4 (Silna anonimowość) *Poziom silnej anonimowości jest określony odległością pomiędzy rozkładami zmiennych losowych Π oraz Π_{ap} zdefiniowaną przy pomocy metryki całkowitego wahania:*

$$d_{TV}(\Pi, \Pi_{ap}).$$

Wiedza a priori obserwatora reprezentuje jego wiedzę częściową. Jeśli obserwator nie posiada wiedzy częściowej, wtedy $\Pi_{ap} = \Pi_U$ (Π_U jest losową permutacją z rozkładem jednostajnym).

Istnieje zasadnicza różnica w interpretacji wartości słabej oraz silnej anonimowości. Dla systemów optymalnych wartość S anonimowości słabej będzie względnie duża. Jednocześnie współczynnik określający anonimowość silną tego systemu będzie względnie mały i bliski 0. Przykładem takiego optymalnego systemu anonimizującego jest system, w którym zmienna losowa Π ma rozkład jednostajny. Wtedy, zakładając brak wiedzy a-priori o systemie, współczynnik określający anonimowość silną wynosi 0. Natomiast wartość anonimowości słabej S osiągnie dla Π wartości maksymalne, gdyż rozkłady prawdopodobieństw pozycji wyjściowych permutacji dla poszczególnych pozycji wejściowych również będą jednostajne, a zatem dające maksymalną entropię. Zatem możemy mówić, iż dany system oferuje optymalną słabą anonimowość, jeżeli jej poziom jest bliski $\log_2(N)$. Natomiast o optymalności w kontekście silnej anonimowości systemu możemy mówić, jeżeli jej poziom jest bliski 0.

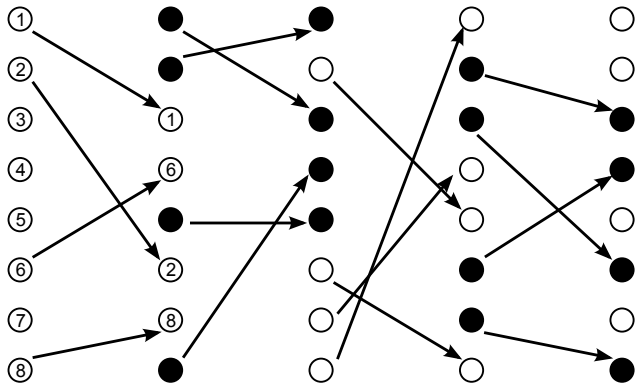
2.4.2 Konsekwencje stosowania częściowego sprawdzania

Twórcy procedury RPC analizując jej wpływ na anonimowość sieci mieszającej rozpatrywali przede wszystkim możliwość ujawnienia pełnych ścieżek przetwarzania jakichkolwiek wiadomości [38]. Oprócz wersji właściwej RPC, przestrzegającej zasadę komplementarności, rozpatrywana była procedura, w której ujawniane były dowolne przejścia (niezależnie dla każdego serwera). Przy założeniu niezależności wyboru połowy przejść do ujawnienia przez każdy serwer można stosunkowo łatwo wykazać [38], iż liczba rund lub serwerów mieszających (rund może być więcej niż serwerów jeżeli pozwolimy na wielokrotne przetwarzania cebul przez serwery), która gwarantuje, że z dużym prawdopodobieństwem żadna pełna ścieżka nie zostanie ujawniona, zależy logarytmicznie od ilości przetwarzanych wiadomości. Natomiast wprowadzenie zasady komplementarności przy wyborze przejść zapewnia z definicji „przerywanie" ujawnianych ścieżek. Analiza osłabienia anonimowości w tym przypadku wymaga uwzględnienia wycieku danych innego typu [31]. Opis wpływu procedury RPC na anonimowość został przedstawiony w dalszej części rozdziału.

Na przykładzie procedury częściowego sprawdzania widać wyraźnie konflikt pomiędzy wymogiem anonimowości a wymogiem zachowania integralności procesu mieszania. Z jednej strony chcemy zachować jak największą tajność permutacji wykonywanych przez serwery mieszające, z drugiej, narzucamy częściową jawność. Powyższy przykład pokazuje również, iż odpowiednio modyfikując proces ujawniania (np. stosując zasadę komplementarności) mo-

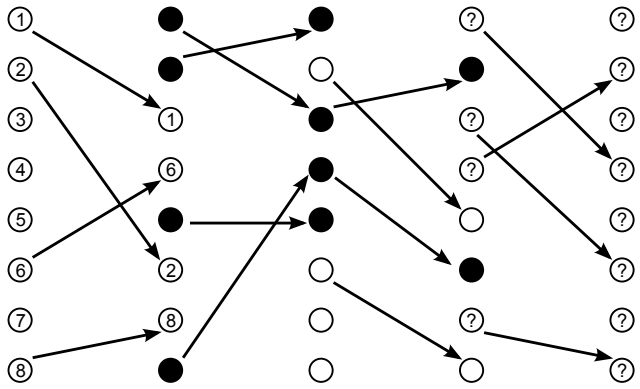
żemy zmniejszyć jego negatywny wpływ na anonimowość sieci mieszającej.

Pomimo tego, iż procedura RPC nie ujawnia pełnych ścieżek dla przetworzonych cebul, możemy zaobserwować pewien wyciek informacji o rozkładzie wiadomości na wyjściu sieci [31]. Jest to spowodowane faktem, iż ujawnienie połowy przejść pierwszego serwera (oraz komplementarnej połowy przejść serwera drugiego) powoduje podzielenie cebul na dwie grupy. Cebule, które znajdują się na pozycjach wskazanych do ujawnienia dla pierwszego serwera będziemy nazywali białymi, a pozostałe czarnymi. Po ujawnieniu przejść przez pierwszą parę serwerów dokładnie wiadomo na jakich pozycjach znajdują się cebule białe i czarne. Ujawnienie przejść przez kolejne pary serwerów powoduje, iż informacja o położeniu cebul białych i czarnych stopniowo zanika. Możemy jednak określić prawdopodobieństwo wystąpienia białej cebuli na danej pozycji. Okazuje się, że przy założeniu, iż na samym początku mieliśmy taką samą liczbę cebul białych i czarnych, na połowie ze wszystkich pozycji wyjściowych systemu prawdopodobieństwo wystąpienia cebuli białej jest takie same, równe $p \geqslant \frac{1}{2}$. Na pozostałych pozycjach to prawdopodobieństwo wynosi $1 - p = q$.



Rysunek 2.2: Podział na czarne i białe – informacja obserwatora pozostaje niezmieniona

Jako przykład możemy przyjrzeć się wiedzy o białych i czarnych cebulach po ujawnieniu przejść przez drugą parę serwerów. Przejścia trzeciego serwera są wskazywane niezależnie, zatem możliwe jest, choć mało prawdopodobne, że przy wyborze przejść do ujawnienia wskazane zostaną jedynie pozycje cebul białych. Sytuacja taka została zaprezentowana na rysunku 2.2. W tym przypadku informacja obserwatora pozostaje niezmieniona. Bardziej prawdopodobna jest jednak skrajnie inna sytuacja, w której pozycje listy wejściowej tego serwera wskazane przez RPC mogą dotyczyć takiej samej liczby



Rysunek 2.3: Podział na czarne i białe – informacja obserwatora zostaje prawie całkowicie zatarta

białych i czarnych. Oznacza to, iż białe i czarne są wymieszane w równych proporcjach, a więc informacja o rozmieszczeniu nawet koloru cebul zostaje kompletnie stracona (patrz rysunek 2.3).

Dalsze rozważania dotyczące ewolucji parametrów związanych z rozkładami brzegowymi (anonimowością słabą) oraz ich ewentualnego wpływu na zbieżność rozkładu łącznego (anonimowością silną) zawiera Dodatek A.

2.4.3 Opis modelu

Niech $\Pi_1, \Pi_2, \ldots, \Pi_\lambda$ oznaczają losowe permutacje N-elementowe poszczególnych serwerów mieszających $M_1, M_2, \ldots, M_\lambda$. Dalej będziemy zakładać, że λ jest parzyste (system składa się z $\frac{\lambda}{2}$ bloków), $N = 2n$, a $\Upsilon_j = \Pi_{2j-1} \circ \Pi_{2j}$ oznacza złożenie permutacji pary kolejnych serwerów mieszających lub inaczej mówiąc permutację bloku procedury RPC. Dla uproszczenia zapisu analizy założymy (bez utraty ogólności), iż procedura RPC będzie zawsze wskazywać do ujawnienia pierwszą połowę pozycji wejściowych pierwszego serwera pary (bloku). W tym celu zdefiniujemy dwa zbiory pozycji $H_1 = \{1, 2, \ldots, n\}$ oraz $H_2 = \{n+1, n+2, \ldots, 2n\}$. Podczas analizy będzie nas interesował rozkład losowego wektora binarnego $X^{(i)} = (X_1^{(i)}, X_2^{(i)}, \ldots, X_{2n}^{(i)}) \in B_n$, który będzie reprezentował rozmieszczenie białych i czarnych elementów na danym etapie, po i-tym bloku permutacji mieszających. $B_n \subset \{0,1\}^{2n}$ jest zbiorem wszystkich ciągów binarnych o n zerach i n jedynkach. Możemy przyjąć konwencję, iż 0 będzie reprezentować kulę czarną, a 1 kulę białą. Obserwacja tego, ile pozycji wejściowych z pierwszej połówki jest przekształcanych przez Υ_i na pozycje również z pierwszej połówki (liczba ta ma oczywiście

rozkład hipergeometryczny $\mathcal{H}(2n, n, n)$) daje pewną informację o rozkładzie wektorów binarnych, w szczególności pozwala wyznaczyć rozkłady brzegowe $\Pr\{X_j^{(i)} = 1\}$. Należy uściślić, iż wartościami zmiennej losowej o rozkładzie hipergeometrycznym $\mathcal{H}(N, m, n)$ jest, używając modelu urnowego, ilość kul wyróżnionych wśród n wybranych spośród N. Przy czym, z N wszystkich kul w urnie m jest wyróżnionych.

Obserwator analizujący proces mieszania oraz przejścia ujawniane w procesie częściowego sprawdzania może od samego początku zidentyfikować dwie grupy po n wiadomości (w postaci zaszyfrowanych cebul), tzw białe i czarne, dla których zna pozycje wejściowe do sieci. Wiadomości wewnątrz grupy są dla obserwatora nierozróżnialne. W miarę postępu procesu mieszania (po i-tym bloku serwerów) obserwator może ustalić n pozycji uprzywilejowanych, na których prawdopodobieństwo wystąpienia 1 (cebuli białej) jest większe i wynosi p_i. Wie również, że dla pozostałych pozycji to prawdopodobieństwo wynosi $1 - p_i$ oraz potrafi wyliczyć p_i śledząc tak zwany proces „stapiania" (przemieszania białych i czarnych w proporcjach wyznaczanych przez rozkład hipergeometryczny) opisany w Dodatku A.3.

Prawdopodobieństwo, że wśród n pozycji uprzywilejowanych mamy dokładnie k jedynek (to jednocześnie oznacza, że a na pozostałych n pozycjach mamy dokładnie $n - k$ jedynek) nie zależy od konkretnego położenia tych jedynek, tzn., mamy $\binom{n}{k} \times \binom{n}{n-k} = \binom{n}{k}^2$ jednakowo prawdopodobnych wektorów binarnych. W naszym procesie może się zdarzyć, szczególnie na początku, że istnieją stałe K_1, K_2, takie, że dla $k \leqslant K_1$ lub $k \geqslant K_2$ takie wektory mają zerowe prawdopodobieństwo. W Dodatku A.2 pokazujemy, że takie ograniczenia liczbowe (związane z istnieniem stałej K_1 lub K_2) określone parametrem δ (Dodatek A.2) szybko zanikają.

Niech $\Pi^{(i)} = \Upsilon_1 \circ \Upsilon_2 \cdots \circ \Upsilon_i$. W myśl definicji 2.4.4 silnej anonimowości to właśnie odległości rozkładów prawdopodobieństw tych permutacji od rozkładu jednostajnego określają wiedzę obserwatora. Prawdopodobieństwo wystąpienia danej permutacji zależy wprost od prawdopodobieństwa wystąpienia odpowiadającego jej wektora binarnego, tzn. jest równe temu prawdopodobieństwu przemnożonemu przez $(n!\, n!)^{-1}$. Zatem w przypadku wiedzy obserwatora wynikającej ze śledzenia cebul białych i czarnych analizę silnej anonimowości (rozkładu prawdopodobieństw permutacji) można sprowadzić do badania rozkładu prawdopodobieństw takich wektorów binarnych (patrz podpunkt 2.4.4).

2.4.4 Przegląd znanych wyników

Autorzy pracy [31] przeprowadzili analizę wycieku informacji w procedurze częściowego sprawdzania. W pracy tej mieszanie poprzez losowe permutacje, o których obserwator posiada pewną częściową wiedzę, modelowane jest poprzez proces Markowa M, którego stany są $2n$-elementowymi permutacjami z powtórzeniami n elementów czarnych oraz n elementów białych. Możemy je utożsamić z wprowadzonymi powyżej ciągami binarnymi. W każdym kroku procesu stan ulega „przemieszaniu" poprzez losową permutację kolejnego bloku serwerów Υ_i. Nie trudno zauważyć prawdziwość następującego lematu (patrz końcówka poprzedniego podpunktu).

Lemat 2.4.1 ([31])

$$d_{TV}(\Pi^{(i)}, \mu_{S_N}) = d_{TV}(X^{(i)}, \mu_{B_n})$$

Szybkość osiągania przez proces M rozkładu stacjonarnego, który w tym przypadku jest rozkładem jednostajnym, została zbadana przy pomocy metody zwanej *sprzęganiem* (ang. *coupling*). Aby ją przybliżyć rozpatrzmy „podwójny" proces Markowa (Y_t, Y_t^*) na przestrzeni stanów $S \times S$ o macierzy przejścia $\overline{\mathcal{P}}$. Każdy z tych procesów rozpatrywany z osobna jest kopią łańcucha Markowa $\mathcal{Y}_t$, który określa macierz przejścia $\mathcal{P}$. Dodatkowo, jeżeli procesy (Y_t, Y_t^*) osiągną ten sam stan, to dalej poruszają się razem:

$$\overline{\mathcal{P}}((x,x)(y,y')) = \begin{cases} P(x,y) & y = y' \\ 0 & y \neq y' \,. \end{cases}$$

Konstrukcję takiej pary nazwiemy sprzęganiem łańcuchów lub couplingiem ([49]). Wykorzystanie metody sprzęgania polega na stworzeniu takiej zależności pomiędzy procesami w parze, aby spowodować przyjmowanie przez nie podobnych stanów (bliskich w sensie pewnej metryki). Jeżeli uda się stworzyć takie sprzęganie, to można oszacować odległość rozkładu procesu mieszania od rozkładu stacjonarnego π_S.

Lemat 2.4.2 (Lemat o sprzęganiu, [6]) *Dla dowolnego sprzęgania łańcuchów (Y_t, Y_t^*) procesu $M = (\mathcal{Y}_t)$, przy założeniu, że Y_0^* jest wybrane zgodnie z rozkładem stacjonarnym π_S, zachodzi*

$$d_{TV}(\mathcal{Y}_t, \pi_S) \leqslant \Pr\{Y_t \neq Y_t^*\} \,.$$

W przypadku badanego procesu rozkład stacjonarny jest rozkładem jednostajnym nad przestrzenią stanów B_n ($\pi_S = \mu_{B_n}$). Z kolejnego lematu można wyprowadzić skuteczną metodę analizy szybkości zbiegania się procesu

zwaną *path couplingiem*. Lemat wykorzystuje pojęcie *szybkości mieszania* (ang. *mixing time*) procesu M

$$\tau_M(\varepsilon) = \min\{T : \forall y \in S, \forall t \geqslant T : d_{TV}(\mathcal{L}_y(\mathcal{Y}_t), \mu_S) < \varepsilon\},$$

($\mathcal{L}_y(\mathcal{Y}_t)$ oznacza rozkład $\mathcal{Y}_t$ przy $\mathcal{Y}_0 = y$) oraz funkcję Δ odległości stanów zdefiniowanej, w przypadku analizy RPC, jako minimalną liczbę transpozycji potrzebnych do przejścia z jednego stanu do drugiego

$$\Delta : B_n \times B_n \to \{1, 2, \ldots, n\}.$$

Maksymalna wartość D funkcji Δ wynosi w przypadku opisywanego procesu n.

Lemat 2.4.3 (Path Coupling, [31]) *Niech Δ oznacza funkcję odległości zdefiniowaną na przestrzeni stanów procesu M, a D maksymalną odległość (przekątną przestrzeni stanów w sensie Δ). Załóżmy że istnieje sprzęganie (Y_t, Y_t^*) dla procesu $M = (\mathcal{Y}_t)$ taki, że dla pewnego rzeczywistego $\beta < 1$ zachodzi:*

$$E[\Delta(Y_{t+1}, Y_{t+1}^*)] \leqslant \beta$$

dla każdego (Y_t, Y_t^*) spełniającego

$$\Delta(Y_t, Y_t^*) = 1.$$

Wtedy

$$\tau_M(\varepsilon) \leqslant \left\lceil \frac{\ln(D\varepsilon^{-1})}{\ln(\beta^{-1})} \right\rceil.$$

Kluczowa część pracy poświęcona jest konstrukcji sprzęgania łańcuchów, których stany różnią się na dwóch pozycjach. Odległość między stanami jest zdefiniowana jako liczba transpozycji, a zatem w tym przypadku odległość wynosi 1. Sprzęganie łańcuchów polega na stosowaniu w jednym procesie permutacji transformującej Υ, która jest wybierana losowo, oraz jej modyfikacji $\Upsilon \circ (u, v)$ poprzez transpozycję (u, v) w drugim procesie. Transpozycja ta jest dobierana w taki sposób, aby powodować z dużym prawdopodobieństwem ujednolicenie stanów w kolejnym lub w drugim w kolejności kroku. Jednocześnie transpozycja (u, v) nie może zaburzać rozkładów prawdopodobieństw wektorów, zatem nie może być pomiędzy pozycjami należącymi do przejść wskazanych do ujawnienia oraz tych niewskazanych. W analizie stosowane jest uproszczenie zakładające, iż zawsze wskazywana do ujawnienia jest pierwsza połowa pozycji wejściowych. W takim modelu u, v muszą należeć do tej samej połowy. Możliwość skutecznego doboru pozycji u, v wynika

z własności rozkładu hipergeometrycznego, iż żadna z połów nie może być zdominowana przez 0 lub 1. Poniżej zamieszczony fakt możemy sformułować na postawie oszacowań dla rozkładu hipergeometrycznego zawartych w monografii [39].

Fakt 2.4.1 *Niech X będzie zmienną losową o rozkładzie hipergoemetrycznym $\mathcal{H}(2n, n, n)$ ($EX = \frac{n}{2}$). Wtedy dla $t \geqslant 0$*

$$\Pr\{|X - EX| \geqslant tn\} \leqslant 2\exp(-2t^2 n)\,,$$

oraz dla każdego $\frac{1}{2} < c < 1$ zachodzi

$$\Pr\left\{|X - EX| \geqslant n^c\right\} \leqslant 2\exp\left(-2n^{2c-1}\right).$$

Z powyższych rozważań wynika, iż nawet przy stałej liczbie serwerów wyciek informacji wynikający z analizy grup wiadomości białych i czarnych jest mały, a tym samym nie powoduje wyraźnego naruszenia anonimowości, co zostało formalnie udowodnione w głównym twierdzeniu pracy.

Twierdzenie 2.4.1 ([31]) *Istnieje $t = O(1)$ takie, że*

$$d_{TV}(\Pi^{(t)}, \mu_{S_N}) = O\left(\frac{1}{n}\right).$$

Więcej informacji na temat *couplingu*, *mixing time* oraz metody *path couplingu* można znaleźć w pracy [6].

W badaniach związanych z rozprawą doktorską podjęto próbę alternatywnego dowodu twierdzenia 2.4.1. Wyniki przeprowadzonych symulacji wskazywały na bliskość rozpatrywanego wyżej rozkładu łącznego i zaproponowanego w Dodatku A rozkładu zmiennych losowych symetrycznie zależnych (rozkładu łącznego „wymienialnych" zmiennych losowych – ang. *exchangeable random variables*), otrzymanego za pomocą niezależnych zmiennych losowych w odpowiedniej przestrzeni warunkowej.

Definicja 2.4.5 (Zmienne losowe symetrycznie zależne, [23]) *Zmienne losowe $X_1, X_2, \ldots, X_n$ nazywamy symetrycznie zależnymi, jeżeli wszystkie $n!$ permutacji $(X_{k_1}, X_{k_2}, \ldots, X_{k_n})$ mają te same n–wymiarowe rozkłady prawdopodobieństwa.*

Próba polegała na zbadaniu szybkości zbieżności do zera wartości parametru ε, opisującego ewolucję rozkładu brzegowego, oraz określeniu kryteriów pozwalających na zastosowanie uzyskanych wyników w badaniu rozkładu łącznego i jego związku z zaproponowanym rozkładem generowanym przez zmienne losowe niezależne. Jedno z takich kryteriów dotyczy zaniku ograniczeń ilościowych i opisane jest zmianami wartości pewnego parametru δ (patrz Dodatek A.2). Należy w tym miejscu dodać, że wspomniana próba nie doprowadziła do prostszego pełnego dowodu.

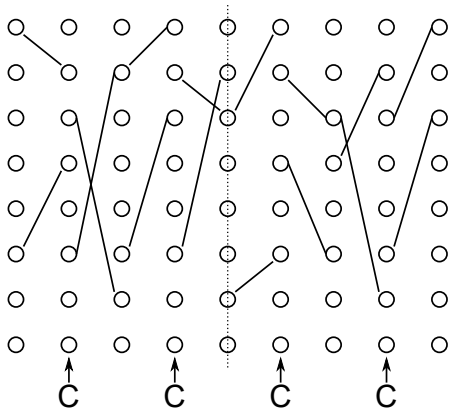
2.5 Wariant dwuetapowy procedury częściowego sprawdzania

W niektórych zastosowaniach możemy chcieć użyć tą samą sieć mieszającą dwukrotnie. Przez tą samą sieć rozumiemy sieć składającą się z tych samych serwerów mieszających, które realizują te same permutacje w obu etapach. Dla zapewnienia własności odporności dwuetapowego procesu mieszania potrzebna jest dwuetapowa procedura weryfikacji. Wymóg ten jest szczególnie istotny w przypadku gdy, pierwsze i drugie użycie sieci jest oddalone znacznie w czasie, a wykrycie nieprawidłowości z tak dużym opóźnieniem mogłoby spowodować podważenie uczciwości całego procesu oraz brak możliwości powtórzenia całego protokołu. Przykładem takiego dwukrotnego wykorzystania sieci mieszającej jest protokół wyborów elektronicznych opisany w podpunkcie 3.4, w którym pierwsze użycie sieci ma miejsce na wiele miesięcy przed wyborami i ma na celu utworzenie specjalnych kart do głosowania. Sieć jest uruchamiana po raz drugi w okresie wyborów. Wykrycie błędów w procesie wytwarzania kart po wyborach miałby oczywiście katastrofalne skutki. Jeżeli sieć miałaby zostać wykorzystana dwukrotnie do przesłania wiadomości i odpowiedzi na nią w krótkim odstępie czasu, wtedy możemy przyjąć, iż weryfikacji nie trzeba wykonywać po każdym etapie „na raty". Wystarczy jedynie przeprowadzić zwykłe RPC po drugim wykorzystaniu sieci.

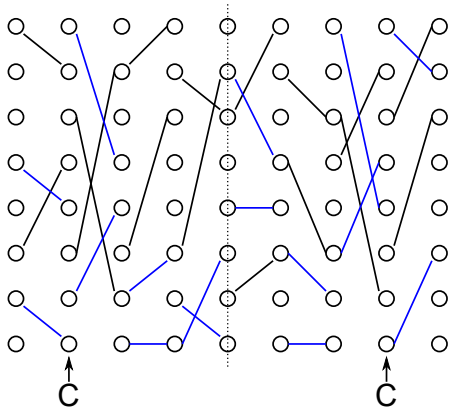
Możemy rozważyć proste rozdzielenie RPC wprost na dwa etapy, tzn. ujawnienie w pierwszym etapie 1/4 przejść, a następnie w drugim, kolejnej 1/4 spośród pozostałych 3/4. Niestety takie podejście nie gwarantuje bezpieczeństwa, gdyż przed drugim etapem sprawdzania wrogi serwer byłby w stanie wskazać przejścia, które nie będą testowane zgodnie z zasadą komplementarności. Z tego względu, zaproponowana została inna dwuetapowa procedura RPC2, w której serwery zgrupowane są w czwórki składające się z dwóch par.

1. *(Pierwszy etap – pierwsze przejście sieci mieszającej)* W każdej parze 1/4 przejść jest odkrywana w sposób podobny do zwykłej procedury RPC – z zachowaniem zasady komplementarności, tzn. pierwszy serwer odkrywa losową 1/4 swoich przejść, a drugi serwer 1/3 przejść ze zbioru dopełniającego poprzednio wskazane pozycje wyjściowe.

2. *(Drugi etap – drugie przejście sieci mieszającej)* W każdej czwórce wybierana jest jedna para, w której ujawnianie kolejnej ćwiartki przejść odbywa się w podobny sposób jak w tradycyjnym RPC (z zachowaniem zasady komplementarności). Serwery w drugiej parze ujawniają ćwiartki przejść niezależnie.



Rysunek 2.4: Pierwszy etap (dwa bloki, 8 serwerów)



Rysunek 2.5: Drugi etap (przejścia wskazane w drugim etapie są zaznaczone kolorem niebieskim)

Opisana powyżej procedura pozwala zapewnić nieujawnianie pełnych ścieżek. Najdłuższe ścieżki jakie mogą zostać ujawnione mają długość co najwyżej 6 (pod warunkiem, iż ilość serwerów mieszających to co najmniej 8). Sytuacja taka ma miejsce, gdy procedura RPC2 wskaże przejścia do ujawnienia przez ósemkę serwerów w sposób opisany przez sekwencję:

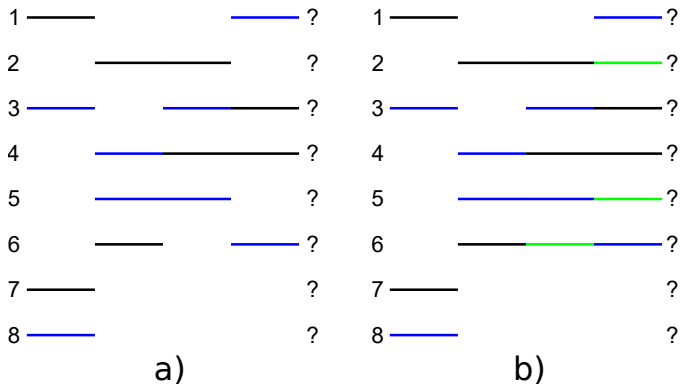
$$(N, K, N, N, N, N, N, K).$$

W powyższym ciągu N oznacza niezależny wybór ścieżek do ujawnienia, natomiast K – komplementarny. W tym przypadku pomiędzy serwerem drugim i siódmym może zostać ujawniona ścieżka długości 6 dla pewnej wiadomości. Wynika to z faktu, iż jedynie między N i K mamy gwarancję ucinania ścieżek.

Można ujawniane ścieżki skrócić upraszczając nieco całą procedurę. Zmiana polega na zastąpieniu wyboru par serwerów dokonywanego po drugim etapie mieszania. Zamiast wybierać parę w każdej czwórce serwerów, dla której ma zostać zachowana zasada komplementarności wybierane jest to, czy zasada ta będzie dotyczyć par parzystych czy nieparzystych. W tym przypadku dla dowolnej ósemki kolejnych serwerów możliwe są tylko dwie sekwencje wyborów (N, K, N, N, N, K, N, N) lub (N, N, N, K, N, N, N, K). Zatem, dzięki tej modyfikacji ujawnianie w procesie weryfikacji ścieżki nie będą dłuższe niż 4.

2.5.1 Zastępowalność bloku RPC przez blok RPC2

Aby móc skorzystać z wyników dotyczących anonimowości standardowej procedury losowo-częściowego sprawdzania w kontekście zaprezentowanej procedury dwuetapowej należy wykazać, iż przejścia ujawnianie przez blok tej procedury ujawniają mniej informacji niż RPC. W skrajnym przypadku może dojść do ujawnienia n pełnych ścieżek długości 3 w ramach jednego bloku. Jest to jednak mało prawdopodobne, raczej będziemy mieli do czynienia z sytuacją podobną do zaprezentowanej na poniższym rysunku (2.6). W przykładzie zastosowano uproszczenie polegające na ustaleniu wszystkich permutacji serwerów jako permutacje identycznościowe.



Rysunek 2.6: Uproszczony schemat (wszystkie permutacje są identycznościami) ujawniania przejść w procedurze RPC2 – niebieskie w pierwszym etapie, czarne w drugim etapie, zielone ujawniane aby uzyskać sytuację analogiczną ze zwykłym RPC (komplementarność obowiązuje w pierwszej parze)

Schemat pokazuje, jakie przejścia należy dodatkowo ujawnić (zaznaczone kolorem zielonym), aby otrzymać tę samą ilość informacji odnośnie relacji po-

między pozycjami wejściowymi i wyjściowymi bloku co w procedurze RPC. Zakładamy oczywiście, iż obserwator nie może przyjąć, iż wszystkie przejścia będę identycznościowe, i tak na przykład czwarta pozycja wyjściowa może być połączona ścieżką z pozycjami $2, 4, 5$ lub 6. Podobnie rozumując widzimy, że pierwsza pozycja wyjściowa może być jedynie połączona z $1, 3, 7$ lub 8, a pozostałe możliwości są wyczerpane przez pozycje wyjściowe numer $2, 4, 5, 6$. Przy okazji możemy również zaobserwować jakie warunki musiałyby być spełnione, aby ścieżki długości 3 powstały w ramach procedury RPC2. Widać, że przejścia ujawnione przez pierwszy serwer nie mogą leżeć na tej samej ścieżce co przejścia ujawnione przez ostatni serwer. Podobne wymogi dotyczą przejść serwera trzeciego. Zatem, z dużym prawdopodobieństwem konieczne będzie ujawnienie dodatkowych przejść, aby uzyskać blok o podobnym „wycieku" informacji co w RPC. W konsekwencji możemy przyjąć, iż przy zastosowaniu tej samej liczby bloków w procedurze RPC2, co w procedurze RPC możemy osiągnąć nie gorszy poziom anonimowości.

2.5.2 Analiza integralności mieszanych wiadomości

Zanim przejdziemy do omówienia wymagań odnośnie poprawności procedur weryfikacji dwuetapowej przyjrzymy się temu ile wynosi prawdopodobieństwo niewykrycia naruszenia integralności cebul w zwykłej procedurze RPC. Będziemy rozpatrywali przypadek ogólniejszy, zależny od wartości dodatkowego parametru φ, w którym ujawniane jest φN przejść ($\varphi \in \left(0, \frac{1}{2}\right\rangle$). Dla tradycyjnej procedury częściowego sprawdzania parametr ten wynosi $\frac{1}{2}$.

Niech $A_1, A_2, \ldots, A_\lambda$ oznaczają zbiory przejść wylosowane do ujawnienia w ramach procedury sprawdzającej dla kolejnych serwerów mieszających M_j ($|A_j| = \varphi N$). Możemy przyjąć, iż przejścia są reprezentowane przez numery pozycji wejściowych. Zbiory $B_1, B_2, \ldots, B_\lambda$ będą reprezentować pozycje, których integralność została naruszona przez poszczególne serwery. Prawdopodobieństwo niewykrycia oszustwa serwera M_j, przy założeniu losowego jednostajnego wyboru A_j, możemy oszacować:

$$\Pr\{A_j \cap B_j = \emptyset\} = \frac{\binom{N-|B_j|}{\varphi N}}{\binom{N}{\varphi N}} = \frac{(N-|B_j|)!}{(N-\varphi N-|B_j|)!\,n!} \cdot \frac{(N-\varphi N)!\,n!}{N!}$$

$$= \frac{(N-\varphi N)(N-\varphi N-1)\ldots(N-\varphi N-|B_j|+1)}{N(N-1)\ldots(N-|B_j|+1)}$$

$$= \prod_{i=0}^{|B_j|-1}\left(1-\frac{\varphi N}{N-i}\right) = \prod_{i=0}^{|B_j|-1}\left(1-\frac{\varphi}{1-\frac{i}{N}}\right) \leqslant (1-\varphi)^{|B_j|}\ .$$

Podobne oszacowanie możemy uzyskać w sytuacji kiedy pozycje naruszające integralność są rozłożone pomiędzy wiele serwerów,

$$\beta = |B_1| + |B_2| + \cdots + |B_\lambda|.$$

Wtedy, biorąc pod uwagę potencjalne związki pomiędzy przejściami wskazanymi przez serwery należące do tej samej pary (oraz niezależność wyborów przejść dla serwerów należących do różnych par, np. serwerów parzystych lub nieparzystych), możemy wywnioskować:

$$\Pr\left\{\bigcap_j \{A_j \cap B_j = \emptyset\}\right\} \leqslant \Pr\left\{\bigcap_{j\,:\,2\nmid j} \{A_j \cap B_j = \emptyset\}\right\}$$

$$= \prod_{\substack{j=1 \\ 2\nmid j}}^{\lambda-1} \prod_{i=0}^{|B_j|-1} \left(1 - \frac{\varphi}{1-\frac{i}{N}}\right) \leqslant (1-\varphi)^{\beta_1},$$

$$\Pr\left\{\bigcap_j \{A_j \cap B_j = \emptyset\}\right\} \leqslant \Pr\left\{\bigcap_{2|j} \{A_j \cap B_j = \emptyset\}\right\}$$

$$= \prod_{\substack{j=2 \\ 2|j}}^{\lambda} \prod_{i=0}^{|B_j|-1} \left(1 - \frac{\varphi}{1-\frac{i}{N}}\right) \leqslant (1-\varphi)^{\beta_2},$$

gdzie

$$\beta_1 = |B_1| + |B_3| + \cdots + |B_{\lambda-1}|, \quad \beta_2 = |B_2| + |B_4| + \cdots + |B_\lambda|$$

$$(\beta_1 + \beta_2 = \beta).$$

Zatem

$$\Pr\left\{\bigcap_j \{A_j \cap B_j = \emptyset\}\right\} \leqslant (1-\varphi)^{\max\{\beta_1,\beta_2\}} \leqslant (1-\varphi)^{\frac{\beta}{2}}.$$

Procedura dwuetapowego RPC2 powinna gwarantować w pierwszym etapie taką samą wykrywalność naruszeń integralności co procedura RPC z parametrem $\varphi = \frac{1}{4}$. Analiza drugiego etapu jest o wiele trudniejsza, gdyż przejścia ujawnione w pierwszym etapie mogą pozwalać na wytypowanie przejść, których zmanipulowanie w drugim etapie może być trudniejsze do wykrycia.

Dla analizy procedur dwuetapowego sprawdzania wprowadzimy dodatkowe oznaczenia. Niech

$$A_1^{(1)}, A_2^{(1)}, \ldots, A_\lambda^{(1)}, \quad A_1^{(2)}, A_2^{(2)}, \ldots, A_\lambda^{(2)}, \quad (A_j^{(1)} \cap A_j^{(2)} = \emptyset)$$

oznaczają zbiory przejść wskazane do ujawnienia w pierwszym i drugim etapie procedury sprawdzającej ($|A_j^{(1)}| = |A_j^{(2)}| = \frac{n}{2}$).

Przy naiwnym rozdzieleniu procedury RPC na dwa etapy, w których wskazywane są kolejne $\frac{1}{4}$ przejść do ujawnienia, wystarczy po pierwszym etapie przyjąć jako $B_j^{(2)}$ podzbiory pozycji komplementarnych w stosunku do ujawnionych w pierwszym etapie:

$$B_j^{(2)} \subseteq \begin{cases} \Pi_{j-1}(A_{j-1}^{(1)}) & 2 \mid j \\ \Pi_{j+1}^{-1}(A_{j+1}^{(1)}) & 2 \nmid j \,. \end{cases}$$

Taki dobór zbiorów $B_j^{(2)}$ powoduje, że niezależnie od wyboru $A_j^{(2)}$ prawdopodobieństwo niewykrycia

$$\Pr\left\{\bigcap_j \{A_j^{(2)} \cap B_j^{(2)} = \emptyset\}\right\} = 1\,.$$

W podobny sposób możemy wykazać niepoprawność konstrukcji RPC2 zaproponowanej powyżej. Jeżeli ograniczymy się do jednego serwera i dobierzemy zbiór $B_j^{(2)}$ zgodnie z wyżej zapisaną zależnością, wtedy prawdopodobieństwo niewykrycia oszustwa jest równe prawdopodobieństwu, że dana para zachowa w drugim etapie zasadę komplementarności:

$$\Pr\{A_j^{(2)} \cap B_j^{(2)} = \emptyset\} = \frac{1}{2}.$$

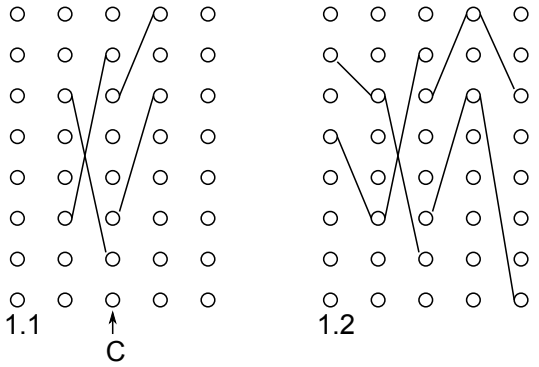
Sytuacji nie poprawia zastosowanie uproszczenia RPC2 polegającego na jednokrotnym rzucie monetą po drugim etapie w celu wskazania, czy komplementarność ma być zachowana w parzystych, czy w nieparzystych parach. W takim przypadku, jeśli oszukujący będą jedynie wśród tylko parzystych lub tylko nieparzystych serwerów, to prawdopodobieństwo niewykrycia będzie stałe (niemalejące wraz ze wzrostem β)

$$\Pr\left\{\bigcap_j \{A_j^{(2)} \cap B_j^{(2)} = \emptyset\}\right\} = \frac{1}{2}.$$

2.5.3 Ulepszona wersja procedury dwuetapowego, częściowego sprawdzania

Przedstawiona wyżej luka procedury RPC2 powoduje, iż z dużym prawdopodobieństwem ($\frac{1}{2}$) nieuczciwy serwer może skutecznie naruszyć integralność dużej frakcji wiadomości. Jest to szczególnie istotna luka z punktu widzenia wyborów elektronicznych, w przypadku których istnieje pokusa zmodyfikowania wielu głosów w celu unieważnienia wyborów, wypaczenia lub wręcz

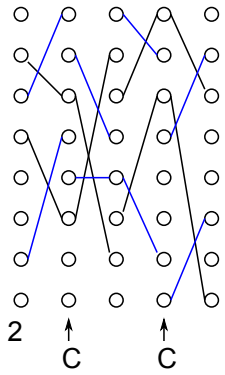
zmiany ich wyniku. Należy jednak zaznaczyć, iż protokoły wyborcze mogą stosować dodatkowe środki przeciwdziałające takim naruszeniom integralności oraz niwelujące opisaną powyżej wadę (patrz podpunkt 3.4.4). W przypadku egzaminów elektronicznych opisany problem jest mniej istotny, gdyż potencjalne oszustwa w tym przypadku mają raczej charakter „punktowy" (dotyczą konkretnych wiadomości), a nie grupowy. W celu zapewnienia integralności w każdym zastosowaniu sieci mieszających podwójnego użycia zaproponowana została poniżej wersja druga dwuetapowego RPC2 – RPC2*.



Rysunek 2.7: Pierwszy etap – w przypadku, gdy dwie kolejne permutacje mieszające wykonują inne serwery etap ten musi zostać wykonany w dwóch krokach

1. *(Pierwszy etap – pierwsze przejście sieci mieszającej)* Serwery są grupowane w pary par (czwórki). Pierwsza i druga para ujawnia komplementarne ścieżki długości 2. Liczba ścieżek ujawnianych przez każdą parę równa jest $\frac{1}{4}$ liczby wszystkich przesyłanych wiadomości. W rezultacie ujawnione ścieżki urywają się pomiędzy drugim i trzecim serwerem czwórki serwerów. W przypadku gdy dwa kolejne miksy są realizowane przez inne serwery należy ujawnianie ścieżek wykonać w dwóch krokach– serwer pierwszy musi poczekać, aż serwer drugi ujawni wskazane przejścia. Podobna sytuacja ma miejsce w drugiej parze – serwer czwarty musi poczekać na trzeci (patrz rysunek 2.7).

2. *(Drugi etap – drugie przejście sieci mieszającej)* Po drugim przejściu przez serwery mieszające oprócz wskazanych do ujawnienia w poprzednim etapie przejść wybierane jest kolejna $\frac{1}{4}$ przejść ($\frac{1}{3}$ pozostałych). Wskazanie przejść odbywa się podobnie jak w tradycyjnym RPC, z zachowaniem zasady komplementarności w parach (patrz rysunek 2.8).



Rysunek 2.8: Drugi etap (przejścia wybrane w drugim etapie oznaczone są kolorem niebieskim)

Zaproponowana w powyższym kształcie procedura dwuetapowej weryfikacji cechuje się tym, że po pierwszym etapie odsłaniania przejść nie można wskazać wśród pozostałych takich, które są mniej lub bardziej uprzywilejowane w drugim etapie. Innymi słowy przejścia ujawnione w pierwszym etapie nie mają wpływu na prawdopodobieństwa ujawnienia pozostałych przejść w etapie drugim, oczywiście poza tym, że eliminują $\frac{1}{4}$ przejść do wyboru w etapie drugim. Zatem przy analizie prawdopodobieństwa skutecznego oszustwa w etapie drugim możemy w każdej parze pominąć przejścia ujawnione w etapie pierwszym. Prowadzi to do sytuacji analogicznej do procedury RPC z parametrem $\varphi = \frac{1}{3}$.

Rozdział 3

Wybory elektroniczne

Poniższy rozdział poświęcony jest problemowi realizacji wyborów elektronicznych. Pierwsza część zawiera przegląd stanu wiedzy o e-wyborach ze szczególnym uwzględnieniem zdalnych wyborów przez Internet. Część ta składa się z opisu wymogów jakie muszą spełnić tego typu protokoły (patrz podrozdział 3.1) oraz przedstawienia przykładowych protokołów najczęściej przywoływanych w literaturze (podrozdział 3.2). Kolejne trzy podrozdziały zawierają propozycje autora rozprawy nowych protokołów wyborczych: protokołu opartego na zaufanym hardware (3.3, [51]), protokołu wykorzystującego papierowe karty do głosowania (3.4, [52]) oraz systemu głosowania dla małych grup wyborczych (3.5).

3.1 Charakterystyka systemów głosowania elektronicznego

3.1.1 Pożądane cechy protokołu wyborów elektronicznych

Przed protokołami realizującymi wybory elektroniczne stawia się szereg wymogów w postaci własności, które tego typu system powinien posiadać. Podstawowy zestaw cech bezpiecznego sytemu wyborczego obejmuje poniżej opisane własności. Należy zwrócić uwagę na fakt, iż nie zawsze zapewnienie wszystkich tych cech jest możliwe, a niespełnienie którejś z nich może być motywowane ograniczeniami technicznymi lub kompromisem na rzecz wydajności.

Anonimowość, poufność – głos wyborcy, a w niektórych przypadkach także to kto brał udział w wyborach pozostaje tajne. Co więcej, żadna

podgrupa uczestników wyborów (wyborcy lub zaufani uczestnicy) nie powinna być w stanie ustalić na kogo dany wyborca głosował.

Weryfikowalność – wyborca powinien mieć możliwość sprawdzenia poprawności każdego etapu protokołu wyborczego. Powinien również móc zweryfikować proces zliczania i przetwarzania wszystkich głosów (*weryfikowalność globalna*) oraz sprawdzić czy jego głos został zaliczony do odpowiedniej puli głosów oraz czy został prawidłowo przetworzony (*weryfikowalność indywidualna*). Zwykle weryfikowalność indywidualna sprowadza się do sprawdzenia czy dana pozycja (przeważnie zaszyfrowany głos wyborcy) występuje w publicznym katalogu. Globalna weryfikowalność natomiast wymaga przeważnie wykonania pewnych mniej lub bardziej skomplikowanych obliczeń. W konsekwencji, przeciętny wyborca może oddelegować globalną weryfikowalność do ekspertów lub pozarządowych organizacji kontrolnych. Weryfikowalność pociąga za sobą zachowanie integralności wyborów – żaden z uczestników nie jest w stanie wprowadzić nielegalnych głosów oraz wszystkie prawidłowe głosy są zliczone. Przeważnie wyborca jest w stanie stwierdzić, czy jego głos w postaci zaszyfrowanej został zaliczony do puli wszystkich głosów. Głosy zaszyfrowane są następnie przetwarzane i zliczane. Na tym etapie w większości wypadków wyborca musi polegać na globalnej weryfikacji dokonywanej przez odpowiednie organizacje i osoby (posiadające odpowiednie kompetencje). Mówimy wtedy o *weryfikowalności pośredniej*. Istnieje jednakże kilka protokołów dających zwykłemu wyborcy możliwość sprawdzenia również etapu przetwarzania głosów, czyli *weryfikowalność bezpośrednią* [51].

Niemożność sprzedawania głosów – wyborca nie ma możliwości udowodnienia osobie trzeciej (kupującemu głos) na kogo głosował (*brak transferowalnego potwierdzenia*, ang. *receipt-freeness*). Istnieje również mocniejsza wersja tej własności, tzw. *odporność na wywierane naciski* (ang. *coertion resistance*), która zakłada, iż nawet przy aktywnym udziale osoby trzeciej (wywierającej nacisk na wyborcę), oznaczającym wcielanie się w wyborcę lub interakcję z nim (poza samym aktem oddawania głosu), wyborca jest w stanie oszukać wywierającego nacisk [21].

Uczciwość (ang. *fairness*) – żadna z zaufanych stron trzecich w czasie przeprowadzania wyborów nie zna cząstkowych wyników.

Legalność (ang. *eligibility*) – tylko uprawnieni wyborcy mogą oddać jeden głos.

3.1.2 Uczestnicy

Powstało wiele propozycji realizacji wyborów elektronicznych, różniących się wykorzystywanymi technologiami, oferowanymi własnościami, oraz rolami uczestników biorących udział w protokole wyborczym. Można jednak wyróżnić pewien wspólny zestaw ról jakie odgrywają uczestnicy protokołów wyborczych.

Wyborca (V) – uczestnik, który może oddać głos. W wielu przypadkach jest on również zobowiązany do przeprowadzenia procesu rejestracji, weryfikacji procesu przetwarzania i zliczania głosów.

Organizator lub organizatorzy wyborów – (EA – ang. *election authority*) uczestnik zapewniający całą infrastrukturę wyborów elektronicznych: serwery zbierające i przetwarzające głosy, maszyny do oddawania głosów. W niektórych protokołach występuje jeden w pełni zaufany EA, w innych z kolei występuje dwóch lub więcej organizatorów, którzy nie wymagają bezwarunkowego zaufania. W drugim przypadku przy założeniu, iż organizatorzy nie współpracują ze sobą mamy zagwarantowaną uczciwość wyborów.

Tablica ogłoszeń – (BB – ang. *bulletin board*) uczestnik ten dostarcza usługi sieciowej umożliwiającej publikowanie uwierzytelnionych (podpisanych) wiadomości. Pełni rolę uwierzytelnionego publicznego kanału informacyjnego.

Organizacja monitorująca wybory – w niektórych protokołach odgrywa rolę audytora, sprawdzającego poprawność wyborów. Protokoły te przeważnie proponują procedurę weryfikacyjną wymagającą pewnych kompetencji matematycznych lub informatycznych, których nie można wymagać od przeciętnego wyborcy. W takim przypadku wyborca może zostać wyręczony przez odpowiednią organizację pożytku publicznego.

3.2 Przegląd istniejących rozwiązań

3.2.1 Różne podejścia do rozwiązywania problemów związanych z wyborami elektronicznymi

Model kontroli środowiska oraz maszyny. Dotychczas zostały zaproponowane trzy podejścia do problemu elektronicznych wyborów [15, 55].

- *Głosowanie w lokalach wyborczych* – specjalne maszyny do głosowania wyposażone w dedykowane oprogramowanie są instalowane w kabinach wyborczych. Wyborcy mogą oddawać głosy w interakcji z tymi urządzeniami, oraz, w niektórych przypadkach, mogą otrzymać potwierdzenie umożliwiające weryfikację. W tym przypadku zakładamy, iż maszyna wyborcza oraz środowisko (lokal wyborczy) są pod kontrolą organizatora. Co więcej, pewne kroki protokołu mogą być wykonane przez urzędników lub mężów zaufania znajdujących się w lokalu wyborczym (np. wyborca może zostać przez nich osobiście uwierzytelniony).

- *Głosowanie za pomocą publicznych terminali* – głosowanie odbywa się za pośrednictwem publicznie dostępnych terminali (np. rozbudowanych bankomatów, lub dedykowanych, państwowych komputerów). W tym scenariuszu zakładamy, iż jedynie maszyna może być kontrolowana.

- *Głosowanie przez Internet* – wykonywane przez aplikację typu klient-serwer, której część kliencka uruchomiona jest na komputerze osobistym wyborcy (PC)/telefonie komórkowym/PDA/karcie chipowej. Strona serwerowa jest zapewniana przez zaufane strony trzecie. Zakładamy, że ani maszyna, ani środowisko nie mogą być kontrolowane. Głosowanie przez Internet jest szczególnym przypadkiem głosowania zdalnego, które obejmuje również głosowanie przez pocztę lub telefon.

Problem bezpiecznej platformy – środki zaradcze. Głosowanie przez Internet jest w oczywisty sposób najbardziej wymagającym z punktu widzenia bezpieczeństwa. W tym typie głosowania elektronicznego trzeba się zmierzyć z problemem bezpiecznej platformy (patrz 1.3). Jako możliwe środki zaradcze wymienia się mechanizmy opisane poniżej [55].

- Zaufany terminal – zamknięte urządzenie zarówno ze względu na oprogramowanie jak i sprzęt posiadające aplikację umożliwiającą uczestnictwo w wyborach.

- Czysty system operacyjny – specjalnie przygotowany system operacyjny uruchamiany z płyty LiveCD, który może zostać uruchomiony na komputerze wyborcy. Tak przygotowana dystrybucja systemu operacyjnego powinna posiadać jedynie te elementy i aplikacje, które umożliwiają uczestnictwo w wyborach.

- Zaufane urządzenie – niewielkie urządzenie o ograniczonych możliwościach zapewniające duże bezpieczeństwo oraz własność czarnej skrzynki. Urządzenie tego typu (np. karta chipowa) może realizować najbardziej newralgiczne kroki protokołu po stronie wyborcy, a następnie

zwracać komputerowi wyborcy jedynie zaszyfrowane dane, tak aby nie mogły zostać zmienione.

- Karty do głosowania (karty kodów, ang. *code sheets*) – mogące zawierać dodatkowe informacje, np. specjalne kody reprezentujące kandydatów, które wprowadzone do komputera nie zdradzają informacji o wyborze dokonanym przez użytkownika.

Pierwsze dwa podejścia do problemu wyborów przez Internet uważa się za niepraktyczne lub zbyt idealistyczne. Wynika to z faktu, iż obie propozycje wymagają zaufania pokładanego w twórcy software'u i hardware'u, oraz radzenia sobie z produkcją systemów imitujących zachowanie legalnych, a wykonujących niedozwolone operacje. Co więcej, dedykowane terminale byłyby stosunkowo drogie. Z tego względu jedynie dwa ostatnie podejścia są poważnie brane pod uwagę.

Zastosowanie kart kodów powoduje całkowitą asymetrię w sensie obliczeniowym – żadne obliczenia nie są wykonywane po stronie wyborcy. Zostaje to osiągnięte poprzez danie wyborcom kart zawierających unikalne kody reprezentujące poszczególnych kandydatów (inny zestaw kodów dla każdej karty). Każdy kod kandydata może posiadać odpowiadający mu kod weryfikujący. Komputer osobisty jest używany do przekazania wprowadzonego kodu do komisji wyborczej, która odpowiada odpowiednim kodem weryfikacyjnym, który jest wyświetlany i może być porównany z kodem weryfikacyjnym znajdującym się na karcie. W konsekwencji, uczciwy organizator wyborów może ochronić je przez atakami hakerskimi, natomiast nieuczciwy organizator może wpłynąć na rezultat wyborów lub naruszyć tajemnicę wyborców. Przykładem rozwiązania opartego na kartach kodów jest system SureVote, który został opisany w [69]. W celu zapewnienia poprawności wyborów zdalnych można również zastosować testowe karty do głosowania. Głosy oddawane przy pomocy tych kart powinny być nieodróżnialne od zwykłych głosów dla uczestników obsługujących wybory. Dzięki temu nie są oni w stanie określić, dla których głosów będą musieli odkryć w szczegółowy sposób proces przetwarzania. Pojawiają się również próby rozwiązania problemu braku zaufania do maszyny poprzez zastosowanie zaufanych urządzeń ([46, 45, 17]), jednakże muszą się one zmierzyć z problemem nieuczciwych producentów i kleptografii [74].

Rodzaje wyborów. Twórcy rozwiązań dla wyborów elektronicznych muszą brać pod uwagę różne rodzaje wyborów organizowane w krajach o różnej tradycji wyborczej. Biorąc ten aspekt pod wagę można wyróżnić następujące rodzaje wyborów.

- Referendum (1-z-2) – wyborca wybiera jedną z dwóch możliwości („tak", „nie") zatem wybór może być zakodowany przy pomocy jednego bitu.
- Wybory 1-z-n – wyborca może wybrać jedną z n opcji, głos jest zatem liczbą z zakresu $1..n$.
- Wybory m-z-n – wyborca wybiera m kandydatów spośród n dostępnych, zatem wybór jest reprezentowany jako podzbiór m-elementowy:

$$\{v_1, v_2, \ldots, v_m\},\ v_i \in \{1, 2, \ldots, n\}.$$

- Wybory m-z-n uporządkowane (pojedynczy głos przechodni, STV – ang. *Single transferable vote*) – wyborca wybiera m spośród n kandydatów, a następnie ustawia w kolejności odpowiadających jego preferencjom. Głos jest zatem kodowany jako ciąg $(v_1, v_2, \ldots, v_m)$, gdzie $v_i \in \{1, 2, \ldots, n\}$.
- Głos otwarty (ang. *wite-in vote*) – niektóre ordynacje wyborcze wymagają, aby wyborca miał możliwość dopisania nazwiska kandydata i oddania na niego głosu.

3.2.2 Obecny stan badań związanych z wyborami zdalnymi

Dotychczas, największy wysiłek badaczy był poświęcony rozwijaniu protokołów przeznaczonych do głosowania w lokalach wyborczych. Jest to najmniej wymagające podejście z perspektywy bezpieczeństwa, spośród trzech powyżej zaprezentowanych podejść. W tym przypadku możemy zakładać pełną kontrolę nad maszyną i środowiskiem (urzędnicy mogą być obecni w lokalu wyborczym). Zostało zaproponowanych kilka dobrze zaprojektowanych rozwiązań, m.in.: system wyborczy Chauma [12], system wyborczy Neffaa [48], W-voting [44], Scratch & Vote (S&V) [2], Prêt à Voter (PaV) [64], Punchscan (PS) [33] and Punchscan połączony z Prêt à Voter Voter (PS+PaV) [71]. Maszyny wykorzystywane w lokalach wyborczych tworzą karty do głosowania (w wersji cyfrowej lub papierowej).

Karta zawiera zaszyfrowany głos lub wszystkie możliwe głosy w formie zaszyfrowanej, wraz z innymi wartościami (identyfikatorami, dowodami o wiedzy zerowej, wartościami losowymi, kluczami). Niektóre z wartości, które dowodzą prawidłowej konstrukcji karty mogą być ukryte. Karta, dla której takie wartości zostały ujawnione (np. poprzez dodrukowanie [12] lub niezniszczenie [2]) jest unieważniana i może zostać zweryfikowana. Procedura weryfikacyjna może wymagać programu komputerowego lub sprzętu (np. skanera).

W praktyce, weryfikacja ma być przeprowadzana przez pozarządowe organizacje kontrolne, które zbierają te karty w pobliżu lokali wyborczych lub w innych miejscach publicznych. W rezultacie, pozostała karta jest uważana za prawidłowo skonstruowaną i może zostać użyta do oddania ważnego głosu. Karta ta (bez danych weryfikujących) nie może dostarczyć dowodu dla sprzedaży głosu (brak potwierdzenia). Zaszyfrowane głosy są publikowane i przetwarzane przez organizatorów wyborów w celu uzyskania ostatecznego wyniku. Każdy wyborca może sprawdzić, czy jego głos został zliczony. Prawidłowość zliczania może być publicznie zweryfikowana. Interaktywne testowanie kart wraz z weryfikacją przetwarzania głosów daje dużą gwarancję, iż oszustwo nie jest możliwe.

Wśród wymienionych rozwiązań można wyróżnić protokoły z uprzednio drukowanymi kartami – wydruk karty nie wymaga interakcji z wyborcą (S&V, PaV, PS, PS+PaV). Karta w tym przypadku zawiera kody dla wszystkich możliwych wyborów. Pozostałe protokoły wymagają podjęcia decyzji podczas tworzenia karty do głosowania, z tego względu są uważane za nienadające się do zdalnych wyborów. Jest to konsekwencja następujących obserwacji.

- Maszyny do głosowania nie mogą zostać zastąpione przez komputery osobiste wyborców. Wynika to z faktu, iż komputer PC nie jest odporny na ingerencję fizyczną – pewne informacje, które powinny być ukryte przed wyborcą mogą wyciekać, pozwalając na sprzedawanie głosów. Przykładowo, probabilistyczne testowanie kart do głosowania opiera się na założeniu, iż wyborca nie poznaje danych weryfikacyjnych właściwej karty do głosowania.

- Zmniejszenie rozmiarów urządzeń do głosowania w lokalach wyborczych tak, aby uczynić je przenaszalnymi, jest trudne technicznie i kosztowne (zintegrowana drukarka i inne aspekty implementacyjne).

Protokoły, w których karty są drukowane przed wyborami nie wymagają maszyn do głosowania w lokalu wyborczym – karty mogą zostać rozdystrybuowane wcześniej. Głos jest oddawany ręcznie, oraz zapisywany (digitalizowany) przez urzędnika obsługującego wybory. Większość z tych protokołów opiera się na jednej zaufanej stronie trzeciej odpowiedzialnej za zbieranie głosów oraz wielu zaufanych stronach trzecich drukujących karty. Model ten zakłada *rozproszenie zaufania* pomiędzy uczestników tworzących karty a uczestnika zbierającego głosy. Rozproszenie zaufania jest własnością, która daje wyborcy pewność, iż nie istnieje jedna zaufana strona trzecia (lub ich podzbiór), która może naruszyć poufność lub integralność sytemu. Model ten wydaje się odpowiedni dla wyborów przez Internet, jeżeli założymy, że

wyborca otrzymuje kartę do głosowania (lub dwie do interaktywnego testowania) przed wyborami. Następnie wysyła odpowiednie wartości z kart przez Internet mając pewność, że żaden z zaufanych uczestników nie jest w stanie naruszyć ich prywatności. Te wymagania są spełnione przez system Prêt à Voter, który realizuje rozproszone drukowanie – proces drukowania jest przeprowadzany przez dwóch lub więcej zaufanych uczestników [64, 72]. Pomimo tego, iż rozproszenie zaufania nie jest zapewnione przez rozwiązania oparte na schemacie Punchscan, oferują one bezwarunkową prywatność (przy założeniu, że wykorzystane zostaną doskonale ukrywające zobowiązania – ang. perfectly hiding commitments – patrz 1.2), tzn. taką która jest zachowana nawet przy nieograniczonym obliczeniowo adwersarzu (1.1.4). Warto również zauważyć, iż systemy inspirowane schematem Punchscan mają najprostsze karty do głosowania (szczególnie PS+PaV), bez żadnych szyfrogramów oraz długich ciągów znaków.

3.2.3 Protokoły oparte na ślepych podpisach cyfrowych

Ślepe podpisy cyfrowe zostały wprowadzone przez Chauma [11] jako środek uwierzytelniania wiadomości bez poznania treści tej wiadomości [11]. Ta metoda podpisu była pierwotnie używana w systemach elektronicznych pieniędzy, ale została zaadoptowana również przez protokoły wyborcze. Tego typu protokoły składają się z trzech faz:

- *rejestracja* — wykorzystując ślepe podpisu cyfrowe (patrz 1.2.5) wyborca uzyskuje żeton do głosowania (ang. *token* – podpisany na ślepo identyfikator) od Komisji Wyborczej (*EC* – ang. *Election Committee*) (przez uwierzytelniony kanał),
- *głosowanie* — wyborca wysyła swój głos oraz żeton do Komisji Zliczającej (*CC* – *Counting Committee*) przez kanał anonimowy i poufny,
- *zliczanie* — głosy są zliczanie oraz publikowane wraz z wynikami przez *CC*.

Podczas fazy rejestracyjnej sprawdzana jest tożsamość wyborcy oraz jego prawa wyborcze. Następnie głos jest przekazywany do *CC* w sposób zapewniający tajność oraz anonimowość. Głosy są zliczane a następnie publikowane w odpowiedniej formie. Dzięki temu możliwe jest zapewnienie globalnej weryfikowalności oraz, w przypadku większości protokołów (np. FOO [25], Radwin [60], JL [40]) weryfikowalności indywidualnej. Jednak w przypadku protokołów opartych na ślepych podpisach zapewnienie indywidualnej weryfikowalności oznacza zrezygnowanie z niemożności sprzedawania głosów. Ta

sprzeczność pomiędzy własnościami jest dość oczywista, gdyż z jednej strony wyborca wymaga cyfrowego dowodu w celu weryfikacji. Z drugiej strony, wyborca nie powinien móc udowodnić osobie trzeciej na kogo głosował.

Problem jednoczesnego zapewnienia braku potwierdzenia oraz weryfikowalności został przezwyciężony w protokole OKA [54]. Cel osiągnięto poprzez rozdzielenie funkcjonalności CC pomiędzy dwie zaufane strony trzecie. Jedna z nich otrzymuje i publikuje zobowiązanie dla głosów i żetonów, podczas gdy druga otrzymuje wartości odkrywające. Wartości te są wykorzystywane przy publikacji list spermutowanych głosów wraz dowodami o wiedzy zerowej ich poprawności w odniesieniu do zobowiązań opublikowanych przez pierwszą zaufaną stronę trzecią. Jest to nieco inny typ weryfikowalności – wyborca widzi opublikowane własne zobowiązania, tym samym upewnia się, że głosy w swej masie zostały prawidłowo odkryte i zliczone. Zatem, wyborca może ufać, że oprogramowanie, którego używa wysłało prawidłowe dane. W tym przypadku mamy zatem do czynienia z weryfikowalnością pośrednią.

3.2.4 Protokoły oparte na sieciach mieszających

Sieci mieszające są jednym z najczęściej wykorzystywanych przez twórców protokołów wyborczych narzędzi krytograficznych. Jedną z pierwszych tego typu propozycji, a przy okazji jednym z najwyżej ocenianych, i najpoważniej rozważanych rozwiązań jest protokół Davida Chauma oparty o kryptografię wizualną [12]. Protokół wykorzystuje specjalne maszyny do głosowania, które potrafią drukować dwuwarstwowo na folii. Wybór jest dokonywany w interakcji z maszyną a następnie nadrukowywany na dwie warstwy w taki sposób, iż jest widoczny jedynie po nałożeniu obu warstw – efekt nałożenia warstw przypomina operację XOR. Wyborca następnie wybiera, którą warstwę chce zabrać jako potwierdzenie. Na wybranej warstwie zapisywana jest cebula zawierająca dane pozwalające odtworzyć drugą warstwę, która jest niszczona przed oddaniem głosu. Wybrana warstwa zostaje zeskanowana przez urzędnika w lokalu wyborczym i opublikowana na stronie wyborów. Pozwala to wyborcy upewnić się, iż jego głos został policzony. Rozwiązanie to mimo swojej elegancji i wielu przemyślanych detali posiada trudną do wyeliminowania wadę – wymaga wykorzystania niestandardowych i wyrafinowanych drukarek. Należy jednak zaznaczyć, iż jest to jedno z nielicznych rozwiązań pozwalających na wpisanie dowolnego głosu, również dopisanie własnego kandydata. Własność ta jest wymagana przez niektóre ordynacje wyborcze.

Prêt à Voter

Jednym z ciekawszych systemów wyborczych z perspektywy tej pracy jest system Prêt à Voter (PaV)[64], który jest właściwie modyfikacją systemu o tej samej nazwie opisanego wcześniej w pracy [63]. Jednym z głównych założeń modyfikacji było wprowadzenie rozproszonego tworzenia kart do głosowania.

Założenia wstępne Zbiór uczestników organizujących wybory jest podzielony na trzy podzbiory:

- R – zbiór uczestników rejestrujących (ang. registrars), posiadających wspólny klucz publiczny krytosystemu progowego ElGamala postaci (p, g, y_R),
- T – zbiór uczestników zliczających głosy (ang. tellers), których publicznym kluczem progowym jest (p, g, y_T),
- C – zbiór urzędników (ang. clerks), który stanowi sieć mieszającą.

Rozproszone generowanie kart

1. Dla każdej z kart indeksowanej tutaj parametrem i do głosowania C_0 tworzy parę „uwikłanych" cebul (ang. *entangled onions*), zawierające początkowe wartości losowe q_i^0, r_i^0, s_i^0, a następnie przekazuje urzędnikowi C_1.
$$(y_R^{q_i^0} \cdot z^{-s_i^0}, g^{q_i^0}), (y_R^{r_i^0} \cdot z^{-s_i^0}, g^{r_i^0})$$
2. C_1 przeszyfrowuje otrzymane cebule wykorzystując losowe wartości q_i^1, r_i^1, s_i^1.
$$(y_R^{q_i^0} y_R^{q_i^1} \cdot z^{-s_i^0} z^{-s_i^1}, g^{q_i^0} g^{q_i^1}), (y_R^{r_i^0} y_R^{r_i^1} \cdot z^{-s_i^0} z^{-s_i^1}, g^{r_i^0} g^{r_i^1})$$
3. C_2 przeszyfrowuje w analogiczny sposób wykorzystując losowe wartości q_i^2, r_i^2, s_i^2, itd.

W rezultacie dla każdej karty powstaje para cebul:
$$(y_R^{q_i} \cdot z^{-s_i}, g^{q_i}), (y_R^{r_i} \cdot z^{-s_i}, g^{r_i}),$$
gdzie:
$$q_i = \sum_{j=0}^{n} q_i^j, \qquad r_i = \sum_{j=0}^{n} r_i^j, \qquad s_i = \sum_{j=0}^{n} s_i^j.$$

Wartości s_i^j są wybierane z liczb skupionych wokół 0. Rozkład prawdopodobieństw wyboru tych liczb jest dwumianowy, natomiast odchylenie standardowe jest określone przez parametr δ. W konsekwencji wartością oczekiwaną dla s_i będzie 0, a odchylenie standardowe wynosi $\delta\sqrt{n}$. Wartość s_i będzie wykorzystywana jako zarodek funkcji determinującej permutację kandydatów na karcie wyborczej.

Tworzenie kart do głosowania na żądanie i oddawanie głosu Obie utworzone w poprzednim kroku cebule, które zawierają ten sam zarodek s_i, zostają nadrukowane na lewą i prawą stronę karty do głosowania. Wyborca udaje się, z taką „niekompletną" kartą do zasłoniętego miejsca oddawania głosu, w którym znajduje się maszyna przekazująca zeskanowaną kartę uczestnikom rejestrującym, którzy kolektywnie deszyfrują wartość s_i, która jest przekazana maszynie w celu wydrukowania odpowiednio uporządkowanej listy kandydatów na jednej z połówek karty. Lista ta jest potrzebna do zaznaczenia znakiem „X" odpowiedniego pola na drugiej części karty, po czym powinna zostać od niej oderwana i zniszczona. Część zawierająca cebulę zaszyfrowaną kluczem uczestników zliczających wraz zaznaczonym kandydatem na pozycji t_i jest cyfrowo rejestrowana przez urzędnika w lokalu wyborczym. Papierowa wersja karty pozostaje w rękach wyborcy jako potwierdzenie. Wartość t_i oraz oraz cebula przeznaczona dla serwerów zliczających są publikowane na tablicy ogłoszeń BB.

Zliczanie głosów Zliczanie polega na progowym odszyfrowaniu przez uczestników zliczających (lub ich odpowiedni podzbiór). Odszyfrowywanie jest poprzedzone przemieszaniem głosów przez tych uczestników działających jako sieć mieszająca. Autorzy zwracają uwagę na to, że jeżeli założymy, iż zarodek s_i wyznacza przesunięcie cykliczne, to możemy „wciągnąć" wartość t_i do cebuli, tak aby zawierała $z^{t_i - s_i}$. Odpowiednio dobrany parametr δ gwarantuje, iż możliwe jest znalezienie logarytmu dyskretnego, a tym samym odszyfrowanego głosu.

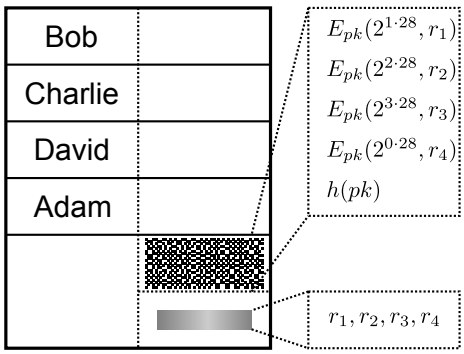
3.2.5 Protokoły oparte na szyfrowaniu homomorficznym

Pierwszą pracą opisującą system wyborczy bazujący na własnościach homomorficznych funkcji szyfrujących była praca Josha Benaloh [4] zawierająca wiele nowatorskich pomysłów, jak choćby połączenie schematów dzielenia sekretów z asymetrycznymi funkcjami szyfrującymi. Dzięki temu możliwe było zrealizowanie sumowania głosów poprzez sumowanie ich cieni. Praca ta stała

się inspiracją dla wielu innych protokołów wyborczych opartych o własności homomorficzne, których jednym z ciekawszych przykładów jest protokół opisany poniżej.

Scratch & Vote

Protokół Scratch & Vote (S&V) [2] jest przykładem nowoczesnego protokołu wyborczego opartego na szyfrowaniu homomorficznym. Charakteryzuje się specjalnymi papierowymi kartami do głosowania, które zawierają spermutowaną listę kandydatów, pola do zakreślania głosu, dwuwymiarowy kod kreskowy oraz tzw. zdrapkę. Karty te są w pełni weryfikowalne bez aktywnego udziału urzędników przeprowadzających wybory.



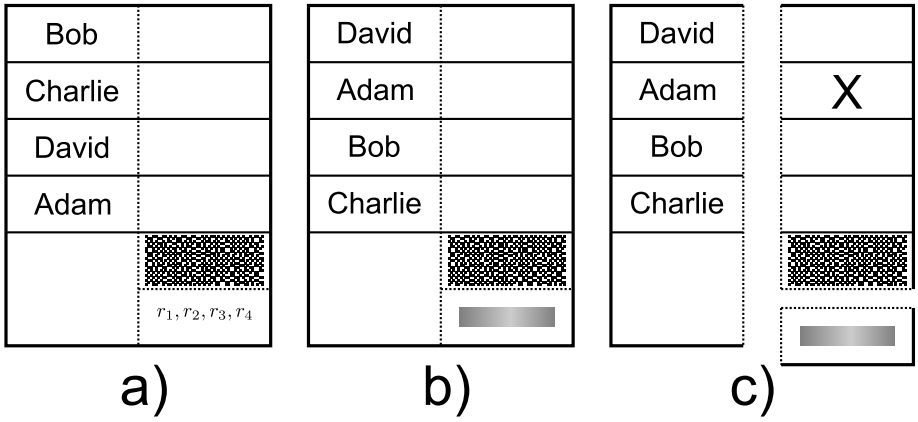
Rysunek 3.1: Karta do głosowania w protokole Scratch & Vote

Głosowanie z perspektywy wyborcy

1. Pobranie karty. Wyborca V wybiera dwie karty do głosowania. Karty są odpowiednio złożone, aby urzędnik wydający karty VO nie widział uporządkowania kandydatów na liście. Wyborca wskazuje kartę, która posłuży mu do oddania głosu. Druga karta będzie służyła jako karta weryfikująca.

2. Audyt (opcjonalny). V usuwa zdrapkę zasłaniającą specjalne wartości weryfikacyjne z karty weryfikacyjne. Tak przygotowana karta może zostać zweryfikowana przez samego wyborcę (przy wykorzystaniu odpowiedniego oprogramowania i skanera) lub przez odpowiednią organizację monitorującą wybory.

3. Wybór. V zaznacza na karcie kandydata, a następnie dzieli kartę na pół. Lewa część karty z nazwiskami kandydatów trafia do specjalnego



Rysunek 3.2: a) karta weryfikacyjna, b) karta do głosowania, c) po oddaniu głosu lewa kolumna karty pozostaje w rękach wyborcy, górna cześć prawej kolumny trafia do urny, zdrapka jest niszczona

pojemnika (z którego V może wyjąć inną połówkę by udać, iż oddał głos na kogoś innego).

4. Oddawanie głosu. V dostarcza EO (urzędnik wyborczy, ang. *election officer*) prawą stronę karty. EO upewnia się, iż zdrapka jest nienaruszona, a następnie odrywa ją i niszczy. Karta zostaje zeskanowana.

5. Weryfikacja indywidualna. V sprawdza czy na stronie wyborów (BB) został opublikowany obraz jego karty do głosowania.

Przygotowanie wyborów. Przed wyborami zostaje ustalona zostaje lista l kandydatów, para kluczy $(\widehat{pk}, \widehat{sk})$ (ilość bitów klucza powinna umożliwiać implementację l liczników). Głos na j-tego kandydata jest kodowany jako $E_{\widehat{pk}}(2^{(j-1)\cdot M})$ (patrz rysunek 3.1 $M = 28$). Zatem parametry globalne to: $(\widehat{pk}, M, (cand_1, cand_2, ..., cand_l))$.

Przygotowanie kart. Dla każdej karty organizator wyborów wybiera kolejność kandydatów na karcie. W tej samej kolejności zostają zapisane w kodzie kreskowym zaszyfrowane głosy na tych kandydatów wraz z haszem klucza publicznego $E_{pk}(2^{28}, r_1), E_{pk}(2^{56}, r_2), E_{pk}(2^{84}, r_3), E_{pk}(2^0, r_4), h(pk)$ r_i są wartościami losowymi, które są zapisywane w części karty zakrytej zdrapką. Dodatkowo na stronie wyborów dostępne są nieinteraktywne dowody poprawności kart (patrz nieinteraktywne dowody z wiedzą zerową 1.2.3). Pojedynczy dowód potwierdza iż dany ciąg szyfrogramów zawartych w kodzie

kreskowym 2D szyfruje wszystkie możliwe głosy (odpowiednie potęgi liczby 2). Dowody te pozwalają na zweryfikowanie właściwych kart do głosowania, które są pozbawione części ze zdrapką.

Zliczanie głosów. Zaszyfrowane głosy są pobierane z kodu kreskowego, a następnie są umieszczane w liczniku homomorficznym. Liczniki te są następnie sumowane i deszyfrowane. Rezultat może zostać w łatwy sposób zweryfikowany.

Ograniczenia protokołu. Wykorzystanie liczników homomorficznych, dla zliczania głosów oddawanych na kandydatów wymaga jednak aby długość klucza była dłuższa niż liczba kandydatów przemnożona przez liczbę bitów potrzebną na zapisanie maksymalnej liczby głosów. Dla przykładu kryptosystem wykorzystujący klucz o długość 2048 bitów może pomieścić 128 liczników 16-bitowych. W przypadku niektórych typów wyborów zależność ta może prowadzić do zwiększania rozmiaru grupy, w której wykonywane są obliczenia, a to z kolei może doprowadzić do przekroczenia pojemności dwuwymiarowych kodów kreskowych. Kody te posiadają ograniczoną pojemność – niektóre źródła podają, iż jest to jedynie 106 bajtów na centymetr kwadratowy powierzchni [10]. Można łatwo zauważyć, że nawet przy 50 kandydatach oraz zalecanym kluczu szyfrującym długości 2048 dwuwymiarowy kod kreskowy przestaje być praktyczny.

3.2.6 Protokoły niekorzystające z algorytmów kryptograficznych

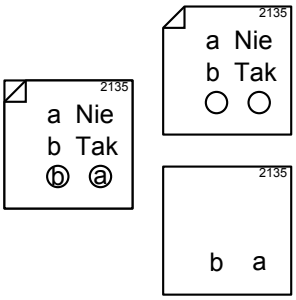
Jednym z istotnych trendów ostatnich lat jest pojawienie się propozycji realizacji bezpiecznych wyborów przy znacznym ograniczeniu liczby stosowanych krypto-narzędzi. Oprócz poniżej opisanego protokołu Punchscan [33], szeroko był omawiany również protokół ThreeBallot [62]. W przeciwieństwie do rozwiązań opartych o zaawansowaną kryptografię, tego typu systemy miały być prostsze, a przez to łatwiej weryfikowalne i zrozumiałe dla przeciętnego wyborcy. Szczególnie ten ostatni fakt był wcześniej niedoceniany jako przeszkoda dla szerszej akceptacji elektronicznych wyborów.

Punchscan

Protokół wyborczy Punchscan opiera się na specjalnych, dwuwarstwowych kartach do głosowania, które są przygotowywane przez w pełni zaufaną, centralną jednostkę. Ilość przygotowanych kart powinna być dwa razy większa od

przewidywanej liczby wyborców. Wynika to z faktu, iż połowa kart zostanie unieważniona na etapie sprawdzania procesu tworzenia kart.

Karty do głosowania. Karty składają się z dwóch warstw, z których każda realizuje przesunięcie cykliczne modulo ilość możliwych wyborów. Warstwa górna zawiera otwory, przez które widoczne są symbole wydrukowane na warstwie dolnej. W przypadku referendów, gdy mamy dwa możliwe wybory: „Tak" lub „Nie", poszczególne warstwy reprezentują przesunięcie o 1 lub 0 pozycji, natomiast karta powstała z nałożenia kart umożliwia wykonanie złożenia tych dwóch przesunięć (reprezentuje operację XOR).

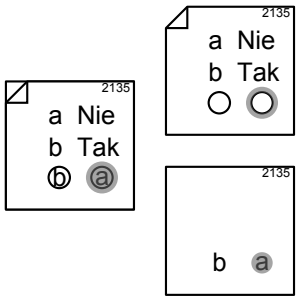


Rysunek 3.3: Karta do głosowania w protokole Punchscan (po lewej stronie nałożone na siebie warstwy karty, po prawej stronie warstwa górna, poniżej warstwa dolna)

Oddanie głosu. Wyborca zaznacza na karcie do głosowania swój wybór specjalnym markerem (o grubości większej niż średnica otworu), który powoduje oznaczenie obu warstw. Wyborca następnie decyduje, którą warstwę chce zachować jako potwierdzenie. Warstwa ta jest skanowana przez urzędnika lub maszynę zbierającą głosy, natomiast druga warstwa jest niszczona. Zeskanowane warstwy są publikowane na tablicy ogłoszeń.

Przygotowanie kart. Zarządzający procesem głosowania przygotowuje i publikuje trzy tabele.

- Tabelę kart, która zawiera dane wszystkich kart do głosowania (identyfikator id, przesunięcie na górnej L_T i dolnej L_B warstwie oraz dodatkową kolumnę, w której zapisany będzie numer otworu zaznaczonego przez wyborcę e). Wszystkie dane są ukryte przy pomocy zobowiązania bitowego.



Rysunek 3.4: Dokonywanie wyboru w protokole Punchscan

- Tabelę deszyfrowania, której rekordy są spermutowane w stosunku do tabeli kart. Indeksy kart zawarte są w pierwszej kolumnie, kolumny D_1 oraz D_2, zawierają przesunięcia, których złożenie jest równoważne złożeniu przesunięć warstw karty. Kolumna rid zawiera informację o relacji pomiędzy rekordami tabeli deszyfrowania, a tabeli rezultatów. Kolumna p dostarcza miejsce, w które zostaną wpisane wyniki częściowe deszyfrowania.
- Tabela z wynikami, zawierająca identyfikatory poszczególnych wyników rid oraz kolumnę v, w której wpisany zostanie odszyfrowany głos.

id	L_T	L_B	e
1	0	0	
2	0	1	
3	1	0	
4	1	1	
5	0	1	
6	1	0	

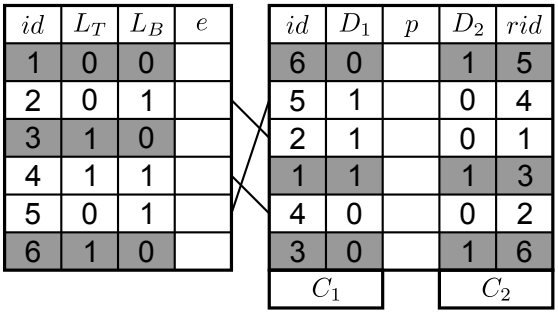
dane ukryte za zobowiązaniem bitowym

id	D_1	p	D_2	rid
6	0		1	5
5	1		0	4
2	1		0	1
1	1		1	3
4	0		0	2
3	0		1	6
C_1			C_2	

rid	v
1	
2	
3	
4	
5	
6	

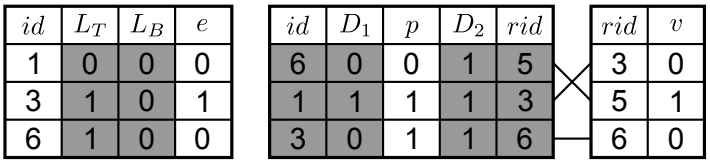
Rysunek 3.5: Publiczne tabele wykorzystywane w protokole Punchscan: (od lewej) tabela kart, tabela deszyfrowania, tabela wyników

Weryfikacja kart. Dla losowej połowy kart odsłaniane są wpisy, które teraz mogą zostać publicznie zweryfikowane. Karty te zostają unieważnione i nie mogą zostać wydane wyborcom. Losowego wskazania dokonują organizacje odpowiedzialne za audyt.



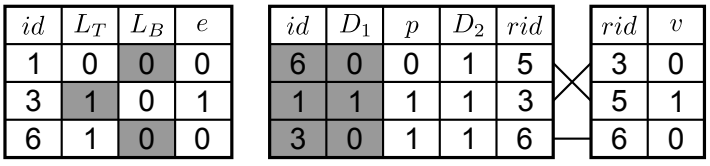
Rysunek 3.6: Odsłanianie połowy kart

Opublikowanie wyników. Po oddaniu głosu przez wyborcę, wybrana przez niego warstwa karty trafia do systemu, co ma odzwierciedlenie w tabeli kart. Zostają odkryte zobowiązanie dla wybranej warstwy, oraz wpisany zostaje numer zaznaczonego otworu. Jednostka zarządzająca wyborami dokonuje odszyfrowania głosu, wynik zapisany zostaje w tabeli wyników, natomiast wynik częściowy w kolumnie p tabeli deszyfrowania.



Rysunek 3.7: Wyniki wyborów opublikowane w protokole Punchscan

Weryfikacja. Proces weryfikowania deszyfrowania polega na wskazaniu przez audytora losowej części tabeli deszyfrowania do ujawnienia. W zależności od wskazania wyborcy weryfikowane jest $e \otimes D_1{=}p$ lub $p \otimes D_2{=}v$.



Rysunek 3.8: Weryfikacja deszyfrowania w protokole Punchscan

Połączenie Prêt à Voter i Punchscan. Dosyć ciekawe połączenie zalet protokołów Prêt à Voter oraz Punchscan zostało zaproponowane w pracy [71]. W rezultacie powstał system, który posiada proste, jednowarstwowe karty do głosowania wzorowane na tych z protokołu Prêt à Voter. Z drugiej strony rozwiązanie to nie korzysta z sieci mieszającej, a jedynie z tablic inspirowanych protokołem Punchscan.

Bezwarunkowa prywatność. Należy zwrócić uwagę na fakt, iż rozwiązania oparte na protokole Punchscan, które wykorzystują zobowiązania bitowe mogą oferować bezwarunkową prywatność. Oznacza to, iż po zakończeniu wyborów anonimowość i prywatność głosów jest zagwarantowana na zawsze i nie mogą zostać naruszone nawet przez posiadającego nieograniczoną moc obliczeniową adwersarza (refremark:unbounded). Własność ta występuje pod warunkiem wykorzystania doskonale kryjących zobowiązań (patrz strona 13).

3.3 Protokół oparty na kartach chipowych z ekranem

3.3.1 Założenia wstępne

Protokół wykorzystuje ślepe podpisy oraz sieci mieszające (opisane w podrozdziałach 1.2.5, 2.1). Kluczowe operacje po stronie klienta wykonywane są przez niewielkie urządzenie wyposażone w ekran. Protokół wprowadza nowatorski mechanizm negocjacji wartości anonimowego identyfikatora, tak aby zapewnić odporność rozwiązania na zagrożenia kleptograficzne. Cechą wyróżniającą proponowane rozwiązanie jest jego bezpośrednia weryfikowalność.

Model komunikacyjny

Kluczowe cechy bezpieczeństwa proponowanego rozwiązania są zapewnione poprzez pewne bezpośrednie interakcje pomiędzy użytkownikiem a urządzeniem V^{dev} – urządzenie przesyła użytkownikowi opis głosu lub identyfikator głosu wyświetlając te dane na ekranie. Z drugiej strony, wyborca powinien potwierdzić dokonany przez siebie wybór. W wersji podstawowej wyborca może po prostu pozostawić kartę w czytniku przez określony czas po tym jak głos został wyświetlony. Można również wyobrazić sobie urządzenie wyposażone w przyciski: „ok" i „anuluj", małą klawiaturę numeryczną lub nawet czytnik odcisków palców. Ta ostatnia możliwość wydaje się szczególnie kusząca, gdyż wiązałaby jeszcze silniej obywatela z urządzeniem, a tym samym zwiększała bezpieczeństwo całego sytemu.

Komunikacja pomiędzy V^{dev} a serwerami komitetów wyborczych odbywa się poprzez program pośredniczący (potencjalnie niezaufany) uruchomiony na komputerze wyborcy (V^{pc}) – to on udostępnia interfejs sieciowy urządzeniu. W komunikacji pomiędzy użytkownikiem a urządzeniem również pośredniczy to oprogramowanie i terminal. Rolę terminala może pełnić komputer osobisty lub jakiś publicznie dostępny terminal.

Komunikację przez kanał uwierzytelniony, który jest realizowany poprzez przesłanie m wraz z podpisem $\mathcal{P}_A(m)$ będziemy oznaczać: $A \xrightarrow{auth} B : m$ (patrz uwaga 1.1.3).

3.3.2 Opis protokołu

Uczestnicy oraz ustawienia początkowe. W protokole biorą udział niżej wymienieni uczestnicy. Należy zwrócić uwagę na fakt, iż wyborca, jego komputer osobisty oraz zaufane urządzenie są traktowani jako osobni aktorzy.

- Wyborca (V_i), jego komputer V^{pc}, oraz urządzenie V_i^{dev}, które zawiera:
  - ▷ $\widehat{pk}_i, \widehat{sk}_i$ (równoważne zapisowi $\widehat{pk}_{V_i}, \widehat{sk}_{V_i}$) – para kluczy asymetrycznych wykorzystywanych tylko na potrzeby tego protokołu, w celu zapewnienia uwierzytelnionego kanału komunikacyjnego w fazie rejestracji (klucze te nie mogą zostać wykorzystane na życzenie użytkownika dla zwyczajnego podpisywania),
  - ▷ $\widehat{pk}_{EC}$ – klucz publiczny EC, $\widehat{pk}_A$ – klucz publiczny sieci mieszającej.
- Komisja Wyborcza (EC – ang. Election Committee):
  - ▷ posiada $\widehat{pk}_{EC}, \widehat{sk}_{EC}$, parę kluczy asymetrycznych;
  - ▷ dostarcza na ślepo podpisanych żetonów uprawnionym do głosowania obywatelom.
- Tablica Ogłoszeń (BB – ang. Bulletin Board):
  - ▷ BB zapewnia uwierzytelniony kanał publiczny;
  - ▷ posiada $\widehat{pk}_{BB}, \widehat{sk}_{BB}$, parę kluczy asymetrycznych.
- Sieć mieszająca $(A_1, A_2, ..., A_\lambda)$:
  - ▷ zapewnia anonimowy kanał;
  - ▷ każdy serwer posiada parę kluczy asymetrycznych ElGamala, klucze te tworzą globalny klucz publiczny $\widehat{pk}_A$ sieci mieszającej; klucz ten jest wykorzystywany do tworzenia cebul.

Przygotowanie wyborów W celu wygenerowania parametrów wyborów oraz udostępnienia ich zaufanym urządzeniom wyborców wykonywane są następujące kroki.

1. EC generuje identyfikator wyborów E oraz parę kluczy asymetrycznych RSA dla wyborów $(\widehat{sk}_E, \widehat{pk}_E) = (d_E, (e_E, n_E))$.

2. EC publikuje klucz wyborów.

$$EC \xrightarrow{auth} BB : E, \widehat{pk}_E$$

3. Urządzenie pobiera parametry wyborów z BB.

$$V_i^{dev} \longleftarrow BB : E, \widehat{pk}_E, \mathcal{P}_{EC}(E, \widehat{pk}_E)$$

V_i^{dev}		BB		EC
				$E, \widehat{pk}_E, \widehat{sk}_E$ publikuje: $E, \widehat{pk}_E$
			$\longleftarrow$ $E, \widehat{pk}_E$ $\mathcal{P}_{EC}(E, \widehat{pk}_E)$	
	$\longleftarrow$ $E, \widehat{pk}_E$ $\mathcal{P}_{EC}(E, \widehat{pk}_E)$			

Tabela 3.1: Przygotowanie wyborów

Rejestracja Przed właściwym głosowaniem wyborca jest uwierzytelniany, a następnie otrzymuje żeton pozwalający mu w anonimowy sposób oddać głos. Służą temu następujące kroki protokołu.

1. V_i^{dev}:

   (a) Wybiera losowo wartość r, oraz część anonimowego identyfikatora wyborcy ID_i'. Obie wartości są elementami zbioru $\mathbb{Z}_{n_E}^*$.

   (b) Zaciemnia $b' = r^{e_E} \cdot ID_i' \mod n_E$, i zachowuje (ID_i', r).

$$V_i^{dev} \longrightarrow V_i^{pc} : h(b')$$

2. V_i^{pc} wybiera drugą część anonimowego identyfikatora wyborcy $ID_i'' \in \mathbb{Z}_{n_E}^*$.

$$V_i^{pc} \longrightarrow V_i^{dev} : ID_i''$$

3. V_i^{dev} wylicza $b = r^{e_E} \cdot ID_i' \cdot ID_i'' \mod n_E$, i zapisuje $ID_i = ID_i' \cdot ID_i'' \mod n_E$

$$V_i^{dev} \xrightarrow{auth} V_i^{pc} : b$$

4. V_i^{pc} weryfikuje $h(b \cdot (ID_i'')^{-1} \mod n_E) = h(b')$. Jeśli wartość jest poprawna, to b jest przekazane EC.

$$V_i^{pc} \xrightarrow{auth} EC : b$$

5. EC:
   (a) Sprawdza tożsamość V_i oraz jego uprawnienia wyborcze.
   (b) $EC \longrightarrow BB : (b, V_i, \mathcal{P}_{V_i}(b))$
   (c) Podpisuje na ślepo: $s = b^{d_E} \mod n_E$
   (d) $EC \longrightarrow V^{dev} : s$

6. V_i^{dev} uzyskuje żeton $t_i = s \cdot r^{-1} \mod n_E$. t_i jest weryfikowane.

V_i^{dev}		V_i^{pc}		EC
ID_i', r				
$b' = r^{e_E} \cdot ID_i'$				
	$\longrightarrow$			
	$h(b')$			
		ID_i''		
	$\longleftarrow$			
	ID_i''			
$b = b' \cdot ID_i''$				
$ID_i = ID_i' \cdot ID_i''$				
	$\xrightarrow{auth}$		$\xrightarrow{auth}$	
	b		b	
				$s = b^{d_E}$
	$\longleftarrow$		$\longleftarrow$	
	s		s	
$t_i = s \cdot r^{-1}$				

Tabela 3.2: Rejestracja

Głosowanie Oddanie głosu przez wyborcę odbywa się następujący sposób.

1. V_i : Wybiera głos v_i i wysyła do urządzenia.
2. V_i^{dev} :
   (a) Wyświetla v_i, i prosi o potwierdzenie.
   (b) Wylicza i wyświetla cebulę $onion(ID_i, t_i, v_i)$.
   (c) $V_i^{dev} \xrightarrow{auth} BB : onion(ID_i, t_i, v_i)$

V_i^{dev}		BB		EC
v_i, ID_i, t_i	$\xrightarrow{auth}$ $onion(ID_i, t_i, v_i)$			
		...mieszanie... publikuje wyniki	$\longleftarrow$ $\mathcal{P}_{\widehat{sk}_E}(E)$	

Tabela 3.3: Głosowanie oraz publikowanie wyników wyborów

Weryfikacja

1. Globalna weryfikacja wymaga przeprowadzenia procedury Randomized Partial Checking, przeliczenia opublikowanych głosów przez odpowiedni program o otwartym kodzie źródłowym oraz porównania liczby zarejestrowanych wyborców z liczbą oddanych głosów.
2. Indywidualna weryfikacja (etap 1):
   (a) Wyborca sprawdza, czy cebula zawierająca jego głos została opublikowana na BB. Cebula lub jej kilka początkowych bajtów są wyświetlane przez urządzenie.
3. Indywidualna weryfikacja (etap 2):
   (a) V_i instaluje $\mathcal{P}_{\widehat{sk}_E}(E)$ w urządzeniu.
   (b) V_i^{dev} weryfikuje podpis (sprawdza czy $\mathcal{V}_{\widehat{pk}_E}(\mathcal{P}_E(ID_i))$ daje wynik pozytywny). Jeśli klucz jest prawidłowy, wyświetlone zostaje ID_i.
4. V_i odnajduje swoją parę (ID_i, v_i) na liście oraz weryfikuje bezpośrednio v_i.

3.3.3 Poziom bezpieczeństwa

Proponowany protokół został zaprojektowany z myślą o implementacji na urządzeniach odpornych na fizyczną ingerencję, czyli posiadających cechę tzw. czarnej skrzynki. Poza tym, urządzenia te powinny być wyposażone w ekran. Między innymi dzięki tym własnościom urządzenia protokół może zaoferować wiele atrakcyjnych cech (oprócz globalnej i indywidualnej weryfikowalności).

Po pierwsze, trudno jest zorganizować sprzedawanie głosów na dużą skalę, gdyż dane weryfikacyjne nie są dostępne przed opublikowaniem wyników wyborów. Urządzenie ukrywa te dane, wykorzystując własność czarnej skrzynki, aż do końca wyborów. W rezultacie, wyborca otrzymuje dowód, który umożliwia opóźnioną weryfikację. Dowód ten jest bezwartościowy dla zautomatyzowanego sprzedawania głosów, gdyż żadne dane nie mogą przekonać kupującego głosy – wyborca może przesłać dane weryfikacyjne lub zdjęcie urządzenia, które mogą zostać na tym etapie łatwo sfałszowane. Należy też zwrócić uwagę na to, iż algorytmy rozpoznawania obrazu mogą zostać oszukane, a zatem jedyną niezawodną metodą przekonania potencjalnego kupującego jest przesłanie urządzenia do ręcznej weryfikacji. Jednakże, takie nielegalne zachowanie byłoby trudne do ukrycia i zorganizowania na dużą skalę, szczególnie jeśli urządzenie jest na przykład dowodem osobistym.

Po drugie, etap rejestracji nie wymaga podjęcia decyzji (własność *niezależnej rejestracji*). Oznacza to, iż faza ta może mieć miejsce na długo przed wyborami. Dzięki temu elektroniczne wybory mogą być łatwo zintegrowane z tradycyjnymi wyborami – Komisja Wyborcza może oznaczyć e-wyborców na tradycyjnych listach wyborczych (patrz schemat estoński [18]). Własność ta nie występowała w schemacie FOO [25], który wymaga zobowiązania do głosu w fazie rejestracji. Łatwa integracja z tradycyjnymi wyborami została uznana za kluczową cechę w badaniach nad możliwymi implementacjami [15]. Dodatkowo podczas fazy rejestracji, która wymaga uwierzytelnienia wyborcy można sprawdzić jego uprawnienia wyborcze i zapewnić legalność.

Co więcej, podczas generowania anonimowego identyfikatora stosowany jest protokół interakcyjnego generowania kluczy, który pozwala komputerowi osobistemu wyborcy zapewnić losowość tworzonej wartości przez zaufane urządzenie. Dzięki temu można wyeliminować zagrożenia kleptograficzne wynikające z faktu generowania wartości losowej przez urządzenie posiadające własności czarnej skrzynki. Jednocześnie mamy gwarancję, iż oprogramowanie komputera osobistego nie jest w stanie poznać identyfikatora.

Fakt 3.3.1 *Bezpieczeństwo interakcyjnego generowania kluczy można sprowadzić do bezpieczeństwa algorytmu RSA (patrz 1.2.5), który jest wykorzystywany do uzyskania ślepego podpisu.*

Dowód Załóżmy, iż urządzenie wyborcy jest zmodyfikowane w „złośliwy" sposób i chce spowodować wyciek informacji poprzez anonimowy identyfikator wyborcy. W tym celu urządzenie musi być w stanie narzucić ostateczny kształt identyfikatora ID_i. Przyjmijmy, iż identyfikator zawierający poufne dane właściciela ma postać $\hat{ID}_i$. Na początku protokołu negocjacyjnego urządzenie wybiera własny składnik identyfikatora ID'_i, zaciemnia go $(b' = r^{e_E} \cdot ID'_i)$, a następnie przesyła hasz tej wartości $(h(b'))$. W odpowiedzi urządzenie otrzymuje ID''. Co więcej, oprogramowanie komputera osobistego poznało $h(b')$, zatem nie można na tym etapie zmienić b'. Zadanie urządzenia polega na znalezieniu r' takiego, że $r'^{e_E} \hat{ID}_i ID_i''^{-1} = b'$ $(r'^{e_E} = b' \hat{ID}_i^{-1} ID_i'')$. Innymi słowy, urządzenie powinno być w stanie odszyfrować szyfrogram RSA nie znając klucza d_E, co jest równoważne ze złamaniem RSA. □

W przypadku proponowanego protokołu głos jest związany z żetonem (własność *powiązanie żeton-głos*) — może zostać zweryfikowane, czy głos został przesłany przez właściciela żetonu. Ta własność jest osiągnięta przez schemat FOO [25], lecz wymaga podjęcia decyzji wyborczej na etapie rejestracji. Powiązanie głos-żeton jest zapewnione przez odporną sieć mieszającą, dającą gwarancję, że treść przesyłanych wiadomości nie może być nielegalnie zmieniona podczas mieszania. Zastosowanie sieci mieszających ma również inną zaletę – żadna z zaufanych stron trzecich nie jest uprzywilejowana w tym sensie, że nie może ustalić częściowych wyników wyborów (w czasie gdy one trwają). Tę ważną własność nazywa się *uczciwością* (ang. *fairness*). Pomimo, iż FOO [25] and JL [40] zapewniają uczciwość, wymagają jednak zrezygnowania z niezależnej rejestracji lub przeprowadzania skomplikowanych wielostronnych protokołów. Poniżej sformułowany fakt wynika wprost z własności bezpiecznych sieci mieszających.

Fakt 3.3.2 *Powiązanie głos-żeton oraz anonimowość, poufność i integralność oddawanych głosów jest zapewniona przez sieć mieszającą.*

Poza wymienionymi zaletami należy zwrócić uwagę, na fakt, iż zaufanie pokładane w komputerze osobistym użytkownika zostało zminimalizowane. To zaufanie zostało właściwie przeniesione na urządzenie, które jest odporne na ataki z zewnątrz. Urządzenie to posiada parę specjalnych kluczy asymetrycznych (poza kluczami wykorzystywanymi dla zwykłych podpisów) przeznaczonych do zastosowania tylko i wyłącznie podczas wyborów elektronicznych. W konsekwencji, niemożliwe jest zasymulowanie protokołu poza urządzeniem wykorzystując je jako podpisująca czarną skrzynkę. Co więcej wybór dokonany przez obywatela jest wyświetlany przez urządzenie upewniając go, że głos zostanie wysłane w postaci niezmienionej. Zostało zauważone przez Schneiera i Shostacka [67], że dodanie ekranu do karty chipowej pozwala

wyeliminować całą klasę ataków typu terminal-karta. Te ataki wykorzystują fakt, iż karta chipowa jest urządzeniem o ograniczonym wejściu i wyjściu. W konsekwencji, musi polegać całkowicie na terminalu (komputerze osobistym wyposażonym w czytnik), co do tego, iż przekazuje jej dane (numery PIN, to na kogo głosuje) od użytkownika w sposób prawidłowy, oraz, że dane wysyłane z karty nie podlegają nielegalnym zmianom. Należy podkreślić, iż nadal możliwa jest zmiana głosu przez komputer osobisty – wyborca może jednak ten fakt odkryć dzięki informacjom wyświetlanym na ekranie urządzenia. Co więcej, jest to wada wszystkich rozwiązań opartych na urządzeniach, które nie mają możliwości bezpośredniego wprowadzania danych. Wadę tę można wyeliminować poprzez zastosowanie dodatkowych środków np. karty kodów [69] .

3.3.4 Analiza poprawności protokołu

Z punktu widzenia wyborcy, głównymi celami, które chce osiągnąć są: oddanie głosu, oraz możliwość weryfikacji procesu zbierania i zliczania głosów. Lista celów, która będzie wyznaczała poprawność protokołu dla organizatora wyborów zawierać będzie: możliwość zweryfikowania praw wyborczych użytkownika systemu, możliwość odszyfrowania i zliczenia głosów oraz możliwość zweryfikowana poprawności procesu przetwarzania głosów.

Oddanie głosu wymaga przeprowadzenia procesu rejestracji podczas, którego wyborca jest uwierzytelniany - komunikacja odbywa się przez kanał uwierzytelniony, wiadomości przychodzące od wyborcy zaopatrzone są w podpis cyfrowy. Dzięki temu organizator wyborów może sprawdzić uprawnienia wyborcze użytkownika, a tym samym spełnić wymóg legalności wyborów, oraz osiągnąć jeden ze swoich celów. Wyborca posiadający uprawnienia otrzymuje żeton do głosowania będący podpisanym „na ślepo” anonimowym identyfikatorem wyborcy. Wartość ta jest wykorzystywana później, podczas oddawania głosu.

Wyborca oddaje głos w postaci zaszyfrowanej cebuli, która oprócz głosu zawiera również żeton oraz anonimowy identyfikator. Zaszyfrowany głos jest publikowany na *BB*, a zatem wyborca może sprawdzić, czy dotarł on w postacie niezmienionej (weryfikacja zbierania głosów). Głosy są następnie przetwarzane przez sieć mieszającą, czego wynikiem jest przemieszana lista odszyfrowanych głosów. Organizator wyborów weryfikuje przebieg procesu mieszania (stosując np. procedurę RPC - patrz 2.1) oraz sprawdza poprawność żetonów do głosowania, tzn. sprawdza, czy rzeczywiście są one jego podpisami pod identyfikatorami wyborców. Po zweryfikowaniu procesu przetwarzania głosów, organizator może zliczyć odszyfrowane głosy i opublikować wyniki. Należy zwrócić uwagę na to, iż każdy może na własną rękę przeliczyć

głosy i zweryfikować wynik. Dodatkowo, po opublikowaniu rezultatu wyborów publikowana jest wartość, która stanowi dowód zakończenia wyborów. Po wprowadzeniu do urządzenia wartość ta jest weryfikowana jako podpis wykonany przy użyciu klucza prywatnego wyborów pod identyfikatorem wyborów. Jeżeli weryfikacja jest jest pomyślna, to urządzenie wyświetla identyfikator wyborów. W konsekwencji, wyborca może odnaleźć swój odszyfrowany głos, a następnie sprawdzić, czy jest poprawny (weryfikacja bezpośrednia głosu).

	Nowy	FOO	OKA	Radwin	Radwin'	JL
Legalność	Tak	Tak	Tak	Tak	Tak	Tak
Uczciwość	Tak	Tak	Tak	Nie	Nie	Tak
Niezal. rejestr.	Tak	Nie	Tak	Tak	Tak	Nie
Powiąz. żeton-głos	Tak	Tak	Tak	Nie	Nie	Tak
Brak potwi.	Tak	Nie	Tak	Nie	Tak	Nie
Weryfik. indyw.	Tak	Tak	Tak*	Tak	Nie	Tak

Tabela 3.4: Porównanie protokołów opartych na ślepych podpisach opisanych w 3.2.3 (* pośrednia weryfikowalność)

3.4 Protokół oparty na papierowych kartach do głosowania

3.4.1 Założenia wstępne

Cele nam przyświecające przy tworzeniu tego protokołu były następujące [52].

Rozproszenie zaufania Schemat, podobnie jak Prêt à Voter (patrz 3.2.4), zapewnia rozproszenie zaufania, jednakże nie wymaga rozproszonego drukowania i związanych z tym komplikacji (np. korzystania ze zdrapek). W rezultacie, osiągnięcie tej własności jest mniej kłopotliwe i tańsze.

Weryfikowalność Każdy etap protokołu podlega weryfikacji przez wyborcę lub przez niezależną organizację.

Brak potwierdzenia Brak potwierdzenia jest gwarantowany w ten sam sposób jak w przypadku Prêt à Voter lub Scratch & Vote – wyborca otrzymuje dwie karty, z których jedna (wyborca wskazuje która) zostaje całkowicie odsłonięta i może zostać gruntownie przebadana. Pozytywny wynik takiego losowego sprawdzenia sprawia, że druga karta

może być uważana za prawidłowo skonstruowaną i może zostać użyta do oddania ważnego głosu. Kluczowym atrybutem tej karty jest fakt, iż wyborca nie posiada cyfrowego (matematycznie weryfikowalnego) dowodu jej autentyczności. Dzięki temu, jak pokażemy dalej trudno jest zorganizować sprzedawanie głosów.

Prosta karta do głosowania Karta nie zawiera żadnych wartości asymetrycznie zaszyfrowanych (długich ciągów znaków) i jest równoważna karcie z protokołu PaV+PS (patrz 3.2.6). Wyborca nie musi wprowadzać długich łańcuchów znaków, co znacząco zwiększa przyjazność dla użytkownika systemu w porównaniu z Prêt à Voter (3.2.4).

Wiarygodność Całkowicie odsłonięte karty, mogą zostać wykorzystane do wprowadzenia przez zwykłych wyborców głosów testowych pozwalających na sprawdzenie poprawności i uczciwości sytemu. Taka możliwość może przekonać sceptyków co do poprawności procesu zliczania.

Praktyczność System zapewnia możliwość łatwej i niedrogiej integracji z tradycyjnymi wyborami. Po stronie wyborcy nie jest wymagany żaden dodatkowy sprzęt, a zdalni wyborcy są rejestrowani przed rozpoczęciem wyborów. Ma to na celu zapobieżenie podwójnemu głosowaniu. Wyborcy, którzy zagłosowali zdalnie mogą również oddać głos w sposób tradycyjny. Tym samym anulują głos oddany drogą elektroniczną. Tego typu podejście zostało zastosowane w Estonii [18] i w pewnym stopniu rozwiązuje problemy wyborców, którzy tracą zaufanie do sytemu e-głosowania oraz wyborców, na których wywierana jest presja.

Odporność na zawirusowany PC Protokół zakłada redukcję zaufania pokładanego w komputer osobisty wyborcy poprzez zapewnienie tego, że komputer nie jest w stanie poznać i zmienić wyboru dokonanego przez użytkownika.

Główną ideą proponowanego rozwiązania jest szyfrowanie głosu wyborcy poprzez wykonanie prostego przesunięcia cyklicznego. Jednocześnie ta trywialna operacja może być odwrócona jedynie przez grupę współpracujących serwerów, które wykonują rozproszone obliczenia. W ten sposób zaufanie do serwerów realizujących e-wybory ulega rozproszeniu i, w przeciwieństwie do prostych kart kodów (patrz system SureVote [69]), proces deszyfrowania głosów może być publicznie kontrolowany. Przyjęty model zakłada również rozproszone tworzenie kart do głosowania. Karta może zostać interaktywnie przetestowana, przy pomocy podobnych technik jak w przypadku protokołów dla głosowania w lokalach wyborczych. Dodatkowo, rozwiązanie to jest

przyjazne dla użytkownika i pozwala mu wprowadzać głosy testujące. Ta ostatnia cecha może okazać się jednym z głównych argumentów dla pozyskania społecznej akceptacji. Przekonanie społeczeństwa do idei wyborów przez Internet wydaje się być jednym z większych problemów stojących na drodze szerszego zastosowania.

3.4.2 Sieci mieszająco-obliczeniowe

Jeśli połączymy ideę homomorficznego szyfrowania (1.2.6) z sieciami mieszającymi (2.1) otrzymamy protokół realizujący rozproszone obliczenie, który zaciera powiązania pomiędzy danymi wejściowymi i wyjściowymi. Obliczenia wykonywane przez taką sieć mogą zostać wykorzystane do tego by anonimowo odwrócić operację dodawania modulo (przesunięcie cykliczne) wykonywaną przez wyborcę. Otrzymujemy dwie nowe funkcje definiujące sieć mieszającą:

- $\hat{onion}_k(m) = \hat{\mathcal{E}}_y(m)$, gdzie $y = y_1 \cdot y_2 \cdot ... \cdot y_\lambda$,
- $\hat{trans}_{i,k,l_{ij}}((a,b)) = (a \cdot h^{l_{ij}} \cdot (y_{i+1}y_{i+2}...y_\lambda)^k / b^{x_i}, b \cdot g^k)$, l_{ij} jest wartością dodawaną danej transformacji i-tego serwera na j-tej pozycji.

W powyższych definicjach funkcji para (y_i, x_i) reprezentuje klucz publiczny i prywatny systemu ElGamala serwera A_i, natomiast g, p są współdzielonymi przez wszystkie serwery mieszające parametrami kryptosystemu. Wartość h jest generatorem podgrupy $\mathbb{Z}_p^*$ o odpowiednim rozmiarze pozwalającym realizować dodawanie modulo odpowiednia liczba (1.2.6).

Dla sprawdzenia dwuetapowego procesu tworzenia kart i deszyfrowania głosów wykorzystywana jest dwuetapowa procedura Randomized Partial Checking (patrz 2.5). Z tą różnicą, iż podczas odkrywania przejść wskazanych przez procedurę weryfikującą ujawniane są również wartości l_{ij}.

3.4.3 Opis protokołu i uzasadnienie jego poprawności

Uczestnicy

Poza wyborcami w protokole biorą udział następujący uczestnicy.

- EC_1 (*Komisja Wyborcza 1* – ang. *Election Committee 1*) -– świadczy usługę głosowania on-line, której użytkownicy są uwierzytelniani przy pomocy standardowych mechanizmów. Komunikacja z EC_1 może się odbywać poprzez uwierzytelniony kanał prywatny.
- EC_2 (*Komisja Wyborcza 2* – ang. *Election Committee 2*) -– drukuje papierowe karty do głosowania wykorzystywane do zakodowania głosów

wyborców. Zakładamy, iż EC_1 oraz EC_2 są w konflikcie interesów (np. są kontrolowane przez partię rządzącą i opozycyjną).

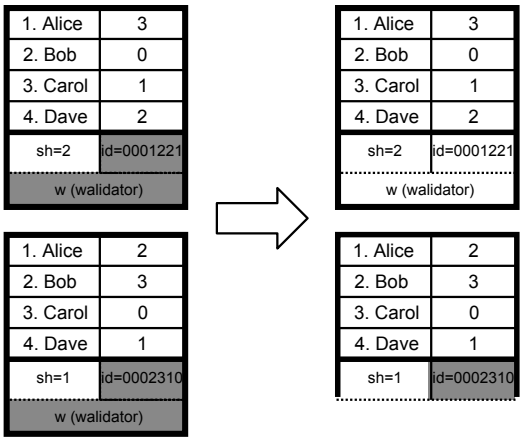
- $A_1, A_2, ..., A_\lambda$ – serwery biorące udział w tworzeniu kart oraz w procesie odszyfrowywania głosów, zapewniają rozproszenie zaufania i audyt.
- BB – tablica ogłoszeń (ang. bulletin board), zapewnia uwierzytelniony publiczny kanał komunikacyjny – pozwala na publikowanie podpisanych wiadomości i zapewnia dostęp do tych informacji.

Protokół z perspektywy wyborcy

1. *(Rejestracja)* Procedura rejestracyjna przypomina aplikację o internetowe konto bankowe. Obywatel wypełnia formularz aplikacyjny doręcza go osobiście w lokalnym urzędzie, gdzie sprawdzana jest jego tożsamość w tradycyjny sposób (na podstawie podpisu i dowodu osobistego). Po upłynięciu określonego czasu wyborca otrzymuje zestaw do głosowania od EC_1. Zestaw zawiera środki umożliwiające zdalny dostęp do internetowego serwisu wyborczego. Metody uwierzytelniania przypominają te wykorzystywane w e-bankowości (PINy, hasła, hasła jednorazowe, tokeny). Token zintegrowany z dowodem osobistym lub dowód osobisty umożliwiający wykonywanie podpisów cyfrowych wydaje się być optymalnym rozwiązaniem, gdyż dowód osobisty jest ściśle związany z wyborcą.

2. *(Uzyskanie kart do głosowania)* Obywatel, który chce oddać głos zdalnie jest zobligowany odwiedzić lokalny urząd w celu otrzymania karty do głosowania (po sprawdzeniu tożsamości). Czynność ta może zostać wykonana w rozsądnie długim okresie czasu przed wyborami. Zakłada się również, iż wyborca wcześniej uzyskał dostęp do serwisu wyborczego (patrz: rejestracja). Wyborca wybiera dwie karty, a następnie decyduje, która z nich ma zostać zweryfikowana. Część weryfikacyjna wybranej karty zostaje całkowicie odsłonięta. Taka karta jest rejestrowana przez urzędnika (skanowany jest jej identyfikator), po czym może zostać zweryfikowana przez wyborcę lub niezależną organizację. Druga karta zostaje pozbawiana zakrytej części weryfikacyjnej, która jest niszczona na oczach wyborcy. Część główna tej karty może posłużyć do oddania ważnego głosu.

3. *(Ręczne szyfrowanie)* Każda karta posiada wartość przesunięcia sh oraz identyfikator id. Przekształcenie cykliczne sh jest reprezentowane poprzez tabelę, której pierwsza kolumna zawiera listę nazwisk kandydatów



Rysunek 3.9: Verification of paper ballots

w odpowiedniej kolejności. Druga kolumna zawiera zaszyfrowane głosy – kandydat numer v jest kodowany poprzez $v + sh \bmod c$ (c jest liczbą kandydatów).

1. Alice	3
2. Bob	0
3. Carol	1
4. Dave	2
sh=1	id=0001221

Rysunek 3.10: Karta do głosowania. Wartość przesunięcia sh nie musi być jawnie zapisana.

4. *(Oddanie głosu)* W określonym przedziale czasowym (np. tydzień przed wyborami tradycyjnymi), kiedy zbierane są głosy zdalne, wyborca, korzystając z serwisu wyborczego (EC_1) wprowadza id oraz zaszyfrowany głos.

5. *(Publikowanie zaszyfrowanych głosów)* EC_1 publikuje nazwiska wyborców wraz z zaszyfrowanymi głosami na BB (bez identyfikatorów id).

6. *(Weryfikacja)* Wyborca sprawdza, czy jego zaszyfrowany głos trafił na tablicę ogłoszeń w formie niezmienionej.

7. *(Publikacja wyników wyborów)* Głosy zostają odszyfrowane przez zaufane strony trzecie $A_1, A_2, \ldots, A_\lambda$ a następnie opublikowane.

Czynności protokołu

Tworzenie kart do głosowania. Stworzenie kart do głosowania wymaga wysłania n par częściowo deszyfrowanych cebul. Pierwsza cebula ($c_{0,j}$ w parze zawiera identyfikator pozycji wejściowej, podczas gdy druga ($\hat{c}_{0,j}$) wykorzystuje homomorficzne szyfrowanie do akumulacji wartości sh, która wyznacza cykliczne przesunięcie w tabeli na karcie do głosowania.

$$c_{0,j} = \mathcal{E}_y(\pi_0(j)),\ y = y_1 \cdot y_2 \cdot ... \cdot y_\lambda \cdot y_{EC_2},\ \hat{c}_{0,j} = (1,1),\ j = 1, .., n$$

$$EC_1 \longrightarrow BB : (c_{0,j}), (\hat{c}_{0,j})$$

Para cebul $(c_{0,j}, \hat{c}_{0,j})$ jest następnie przetwarzana przez kolejne serwery mieszające A_i ($c_{i,j} = trans_{i,k_{i,j}}(c_{i-1,j})$, $\hat{c}_{i,j} = \hat{trans}_{i,k_{i,j},l_{i,j}}(\hat{c}_{i-1,j})$), $i = 1, 2, ..., \lambda$ i przekazywana kolejnemu serwerowi za pośrednictwem tablicy ogłoszeń. Należy zwrócić uwagę na fakt, iż identyfikatory i wartości sh pozostają nieznane wszystkim uczestnikom, gdyż są cały czas zaszyfrowane kluczem publicznym EC_2.

Drukowanie kart do głosowania. Każda para na liście wyjściowej sieci jest deszyfrowana przez EC_2, a wynikowe identyfikatory i wartości przesunięcia cyklicznego są drukowane na karcie do głosowania (sh, id). Wartość przesunięcia służy do utworzenia drugiej kolumny tabeli. Każda karta jest uzupełniana ukrytymi wartościami: odpowiadającą parą cebul wyjściowych ($c_{\lambda,k}$, $\hat{c}_{\lambda,k}$) oraz nieinterakcyjnym dowodem o wiedzy zerowej poprawności odszyfrowania $c_{\lambda,k}$ ($nizk(id, c_{\lambda,k})$) i $\hat{c}_{\lambda,k}$ ($nizk(h^v, \hat{c}_{\lambda,k})$).

Dystrybucja i sprawdzanie kart. Każdy wyborca V osobiście uzyskuje dwie papierowe karty do głosowania. Następnie wskazuje jedną z nich, która może zostać poddana szczegółowej weryfikacji. Dla tej karty wyborca poznaje ukryte wartości weryfikacyjne, w przeciwieństwie do drugiej karty, dla której ta część karty zostaje zniszczona. Identyfikator karty wskazanej zostaje zeskanowany przez urzędnika, pozycja wyjściowa sieci mieszającej odpowiadająca tej karcie zostaje oznaczona przez EC_2 jako nieważna na BB. Wartości, które zawiera ta karta mogą zostać zweryfikowane. Należy zwrócić uwagę na fakt, iż weryfikacja wymaga dostępu do danych dostępnych online na BB. Druga karta udostępnia wyborcy id, sh, które pozwalają na oddanie ważnego głosu.

Głosowanie. Każdy wyborca V szyfruje swój głos na kandydata numer v przy pomocy wartości z drugiej kolumny, która jest równa $v + sh \bmod c$. Wartość ta jest przesyłana razem z id do EC_1 poprzez uwierzytelniony kanał.

$$V \xrightarrow{auth} EC_1 : id, v + sh \bmod c$$

Komisja wyborcza publikuje: pozycję na liście wejściowej $p = \pi_0^{-1}(id)$, identyfikator wyborcy, jego zaszyfrowany głos, cebulę

$$\hat{c}'_{0,p} = o\hat{n}\hat{i}on_{k'_{0,j}}(v + sh \mod c),$$

oraz $k'_{0,j}$ jako dowód poprawności cebuli. Wartość id karty nie jest publikowana.

$$EC_1 \longrightarrow BB : p, V, v + sh \mod c, \hat{c}'_{0,p}, k'_{0,j}$$

Odszyfrowywanie i zliczanie głosów. Zaszyfrowane głosy umieszczone na tablicy ogłoszeń zostają wprowadzone do tej samej sieci mieszającej, która korzysta z tych samych wartości l_{ij} oraz z tych samych permutacji. Wartości l_ij zamiast być dodawane są odejmowane przez A_i ($\hat{c}'_{\pi_i(j)} = \hat{t}_{i,k'_{i,j},-l_{i,j}}(\hat{c}'_{i-1,j})$). Po drugim etapie przetwarzania sieć mieszająca zwraca odszyfrowane głosy (publikowane na BB). Cebule, które opuszczają sieć na pozycjach oznaczonych jako nieważne, są śledzone wstecz. Oznacza to, iż łatwo można wykryć korzystanie z odkrytych kart do głosowania.

3.4.4 Poziom bezpieczeństwa

Poufność i rozproszenie zaufania. Poufność systemu jest zachowana głównie dzięki własnościom częściowo deszyfrującej sieci mieszającej bazującej na asymetrycznym kryptosystemie ElGamala. Operacje przeprowadzane przez sieć mieszająco-obliczeniową pozwalają utworzyć wartości przesunięć cyklicznych wykorzystywanych przez wyborcę do zakodowania głosu operacją dodawania modulo (one-time pad).

Fakt 3.4.1 *Bezpieczeństwo sieci mieszająco-obliczeniowej, a w szczególności obliczeń na zaszyfrowanych elementach grupy cyklicznej opiera się na bezpieczeństwie semantycznym schematu ElGamala (patrz* bezpieczeństwo semantyczne *w 1.2.2).*

Należy podkreślić fakt, iż bezpieczeństwo wyborów jest zachowane jeżeli zaufane strony trzecie EC_1 i EC_2 są w konflikcie interesów, np. są kontrolowane przez partię rządzącą i opozycyjną. W przeciwnym wypadku byłyby w stanie naruszyć prywatność obywateli i próbować wprowadzić fałszywe

głosy. Jest to spowodowane faktem, iż jedna z komisji (EC_1) kontroluje wejście do sieci mieszającej, podczas gdy druga (EC_2) kontroluje jej wyjście. Dokładniej rzecz biorąc EC_2 poznaje wartości wyjściowe (przesunięcia sh oraz identyfikatory), natomiast EC_1 wie jak, na podstawie identyfikatorów kart, rozmieścić zaszyfrowane głosy na wejściu do drugiego etapu przetwarzania przez sieć mieszającą. Zatem, zsumowana wiedza obu komisji pozwala odtworzyć permutację wykonywaną przez sieć mieszającą.

Poprawność częściowego sprawdzania. Obliczająca sieć mieszająca jest weryfikowana poprzez dwuetapowe RPC2 bez ujawniania pełnych ścieżek. Dzięki temu zachowana jest anonimowość przy jednoczesnym zapewnieniu nienaruszalności głosów. Należy jednak zaznaczyć, iż poziom bezpieczeństwa zależy od dobranej metody dwuetapowego sprawdzania (patrz 2.5). Nieuczciwy serwer miksujący może wykorzystać własności homomorficzne kryptosystemu używanego do tworzenia cebul i spowodować przesunięcie głosów z kandydata X na kandydata znajdującego się o dowolną liczbę pozycji dalej na liście. Jeżeli zależy mu na wsparciu kandydata Y, który znajduje się o k pozycji dalej na liście niż popularny kandydat X, to jest w stanie zmodyfikować głosy i przesunąć je o k pozycji. Jeżeli te zmiany będą dokonane po drugim etapie na odpowiednich przejściach to z prawdopodobieństwem $\frac{1}{2}$ nie zostaną wykryte przez przez RPC2 (patrz 2.5.2). By tego uniknąć można wykorzystać dodatkową metodę weryfikacyjną tego protokołu, która pozwala wprowadzić pewną liczbę głosów testowych (maksymalnie mogą one stanowić połowę wszystkich głosów), dla których ujawniane są pełne ścieżki. Zauważmy, że jeżeli y oznacza frakcję głosów testowych pośród wszystkich N, natomiast x frakcję głosów wypaczonych przez pewien serwer, to prawdopodobieństwo niewykrycia oszustwa jest bardzo małe, co wynika z poniższego oszacowania:

$$\frac{\binom{N-xN}{yN}}{\binom{N}{yN}} = \frac{((1-x)N)_{yN}}{(N)_{yN}} \leqslant (1-x)^{yN} \sim e^{-xyN}.$$

Zamiast stosowania RPC2 z dodatkowymi głosami testowymi możemy użyć udoskonaloną procedurę weryfikacyjną RPC2* (2.5.3).

Niemożność sprzedawania głosów i odporność na wywieranie wpływu na wyborców. Zapewnienie weryfikowalności i niemożności sprzedawania głosów jest jednym z największych wyzwań z jakimi trzeba się zmierzyć projektując protokół wyborczy. W proponowanym protokole wyborca otrzymuje dwie papierowe karty do głosowania. Następnie spośród nich wybiera jedną, dla której ujawnione będą wartości weryfikujące. Karta ta zostaje oznaczona w systemie jako nieważna i może zostać zweryfikowana przez

odpowiednią organizację pozarządową. W efekcie, wyborca wierzy, iż posiada prawidłowo skonstruowaną kartę, której wartości weryfikacyjne zostały zniszczone. Dzięki temu nie jest w stanie udowodnić autentyczności tej karty nikomu innemu (kto nie był obecny podczas wyboru kart), a więc nie może sprzedać swojego głosu. Brak cyfrowego, łatwo weryfikowalnego dowodu poprawności karty oraz fakt, iż można stworzyć fałszywe karty czyni proceder skupywania głosów bezcelowym. Nieuczciwy kandydat może wymusić na wyborcy absencję podczas wyborów lub oddanie losowego głosu, lecz wyborca może zawsze oddać swój głos w lokalu wyborczym.

Weryfikowalność każdego kroku. Aby zweryfikować poprawność oraz integralność wyborów należy przebadać dokładnie następujące kroki: rozproszone tworzenie kart, deszyfrowanie kart i drukowanie, oddawanie głosów, rozmieszczenie głosów na listach wejściowych, rozproszone deszyfrowanie głosów. Dowód poprawności rozproszonego tworzenia kart oraz deszyfrowania głosów opiera się na zmodyfikowanej dwuetapowej procedurze RPC. Dowód może zostać publicznie zweryfikowany lecz wymagana jest od sprawdzającego pewna ponadprzeciętna kompetencja informatyczno-matematyczna. W konsekwencji, weryfikacja będzie oddelegowana do niezależnych organizacji i ekspertów. Drukowanie, deszyfrowanie oraz rozmieszczenie głosów na listach wejściowych jest weryfikowane przez przez interaktywne testowanie kart – wybieranie jednej kart z dwóch jako karty testowej. Pozycja na wyjściu sieci, która odpowiada karcie testowej zostaje oznaczona. Dzięki temu głosy testowe mogą zostać później zidentyfikowane, a proces ich przetwarzania szczegółowo zweryfikowany. W ten sposób sprawdzanie całego procesu przetwarzania głosów zostaje wzmocnione. Wyborcy sami mogą wprowadzać głosy testowe lub przekazać przekazać karty testowe niezależnym organizacjom. Karty testowe można uważać za czynnik zwiększający zaufanie do systemu, gdyż wyborca może głos testowy zweryfikować bezpośrednio.

Odporność na nowe ataki na RPC. W pracy [42] zostały opisane ataki umożliwiające śledzenie wiadomości przesyłanych przez wybranych użytkowników sieci mieszających pod warunkiem, iż w procederze bierze udział pewna podgrupa nadawców wiadomości oraz co najmniej jeden nieuczciwy serwer mieszający. Ataki te zakładają, iż zbiór możliwych wiadomości jest stosunkowo duży, pozwalający gwarantować (z dużym prawdopodobieństwem) ich niepowtarzalność. W opisanym protokole moc zbioru wiadomości jest równa ilości kandydatów, która jest przeważnie znacząco większa od ilości głosujących użytkowników. Zatem wiadomości będą z dużym prawdopodobieństwem się powtarzać, co uniemożliwia ich śledzenie metodami z pracy [42].

3.4.5 Porównanie z innymi rozwiązaniami

Tabela zamieszczona poniżej zawiera porównanie cech protokołów z uprzednio drukowanymi kartami, w tym cechy proponowanego protokołu.

	Nowy	PaV	S&V	PS	PS+ PaV	Sure Vote
Weryfikowalność	Tak	Tak	Tak	Tak	Tak	Nie*
Rozproszenie zaufania	Tak+	Tak	Nie**	Nie	Nie	Nie*
Prosta karta	Tak	zawiera szyfr. zdrapka	zwiera szyfr.	Dwie warstwy	Tak	Tak
Bezwarunkowa poufność***	Nie	Nie	Nie	Tak	Tak	Nie

Tabela 3.5: Porównanie protokołów z uprzednio drukowanymi kartami opisanych w podrozdziałach 3.2.4 i 3.2.5 (* dokumentacja nie dostarcza przekonywujących dowodów, ** uczestnik drukujący poznaje zbyt wiele na podstawie publikowanych wartości weryfikujących, + nie potrzebne rozproszone drukowanie, *** poufność zachowana nawet przy nieograniczonym obliczeniowo adwersarzu 1.1.4)

3.4.6 Propozycje uogólnień

Karty kodów. Proponowany schemat na pewno zyskałby na atrakcyjności dzięki natychmiastowym zapewnieniu, iż głos oddany przez wyborcę dotarł w niezmienionej postaci do Komisji Wyborczej. Aby osiągnąć ten cel można zastosować kody weryfikacyjne umieszczane przez EC_1 w cebuli niosącej identyfikator podczas tworzenia kart do głosowania. Kody te byłyby nadrukowywane przy nazwiskach kandydatów i służyły do potwierdzenia otrzymania głosu przez EC_1. Jako kody weryfikacyjne można zastosować obcięte podpisy cyfrowe EC_1 – EC_1 mogłoby wtedy weryfikować karty online. Jeśli ilość kandydatów przekracza pojemność cebuli, można przesłać zarodek generujący kody.

Wybory m-z-n. W podstawowej wersji rozwiązanie oparte na papierowych kartach do głosowania umożliwia wybór 1 z n kandydatów. Jednakże, można łatwo rozszerzyć sposób głosowania, poprzez dodanie wielu przesunięć cyklicznych, na wybory m-z-n lub uporządkowane wybory m-z-n. Wymaga to

zwiększenia liczby homomorficznych cebul przetwarzanych przez serwery mieszające oraz nieco inny układ karty do głosowania. Karta w takim przypadku musiałaby zawierać listę kandydatów oraz kolumny zawierające numery odpowiadające przesunięciom cyklicznym kodującym głosy na poszczególnych kandydatów.

3.5 Systemy głosowania elektronicznego dla małych grup wyborców

3.5.1 Wprowadzenie

Systemy elektronicznego dokonywania wyborów nie są jedynie przydatne w przypadku wyborów narodowych. Również instytucje publiczne, czy też duże i średnie firmy w wielu sytuacjach dokonują ważnych wyborów w sformalizowany sposób poprzez ustalone gremia. Zatem systemy wyborcze mogą znaleźć o wiele szersze zastosowanie, również w sytuacjach, które nie wymagają tak restrykcyjnego zestawu własności bezpieczeństwa jak np. wybory parlamentarne. Możemy z powodzeniem zaprojektować system, który sprawdzi się w przypadku obrad senatu uczelni, choć np. nie gwarantuje niemożności sprzedawania głosów. Poniżej zostanie zaprezentowany opis cech oraz propozycje systemu wyborczego dla wyborów małej skali.

3.5.2 Charakterystyka systemu głosowania dla małych grup

Można traktować głosowania wykonywane przez małe grupy jako szczególny przypadek normalnych wyborów organizowanych na dużą skalę. Wydaje się jednak, iż tego typu wybory posiadają swoją specyfikę wynikającą z faktu, iż przeważnie głosujący znajdują się w jednym pomieszczeniu, mogą posiadać dedykowany sprzęt (przydzielany anonimowo). Co więcej, tego typu głosowania wymagają szybkiego zliczania głosów oraz możliwości sprawnego przejścia do kolejnego głosowania, tak aby możliwe było zrealizowanie kilku głosowań w krótkim odcinku czasu. Poniżej zaprezentowana została lista postulatów dotyczących charakterystyki praktycznego systemu wyborczego dla małych grup.

Kanał komunikacyjny System musi być w jak największym stopniu wolny od papierowych kart do głosowania, których dystrybucja spowalnia cały proces wyborczy. Do realizacji wyborów wykorzystywane są urządzenia

elektroniczne komunikujące się z serwerem bezprzewodowo, bez pośrednictwa sieci Internet. Do publikowania informacji odnośnie przebiegu wyborów (listy kandydatów, wynik wyborów, dane weryfikacyjne) system wykorzystuje widoczny dla wszystkich wyborców telebim, ekran rzutnika lub ekrany zainstalowane na stanowiskach wyborców.

Rodzaj wyborów Standardowo system powinien przede wszystkim obsługiwać trzy opcje wyboru („tak", „nie", „wstrzymuję się"). Należy jednak przyjąć, iż powinny być również wspierane wybory m-z-n. Jednocześnie możemy założyć relatywnie małą liczbę wyborców – z reguły nie przekraczającej 150.

Weryfikacja System powinien dawać możliwość weryfikowania procesu zliczania i oddawania głosów. W tym celu publikuje on odpowiednio zakodowane głosy wyborców na ekranie publicznym. Weryfikacja może się odbywać częściowo przy wykorzystaniu specjalnego oprogramowania o otwartym kodzie źródłowym.

Super-wyborcy Atrakcyjną cechą systemu byłaby możliwość dana pewnej grupie wyborców (tzw. super-wyborców) pozwalająca odtajnić głosy oddane przez wyborców, ale jak pokażemy dalej, bez naruszania anonimowości głosowania.

Poufność, anonimowość Poufność wyborów jest zapewniona dzięki bezpiecznemu szyfrowaniu głosów wyborców. Anonimowość może być zapewniona dzięki temu, iż wybory sami losują urządzenia wchodząc na salę obrad lub poprzez odpowiednie przetwarzanie głosów i wykorzystanie super-wyborców. W pierwszym przypadku przez cały czas trwania obrad nie jest znany sposób przydzielenia wyborców do urządzeń. Zwiększenie poczucia wpływu wyborcy na proces szyfrowania można uzyskać poprzez umożliwienie generowania przez niego liczby losowej wykorzystywanej później w procesie szyfrowania głosu. Dzięki temu może również sprawdzić, czy odpowiednio zakodowany głos został uwzględniony przez system (został wyświetlony na publicznym ekranie i jest zapisany w protokole przebiegu głosowania).

3.5.3 Rozwiązanie oparte o własności homomorficzne funkcji szyfrujących

Poniżej zostanie zaprezentowane rozwiązanie oparte o własności homomorficzne funkcji szyfrującej algorytmu ElGamala (opisane w podrozdziale 1.2.6).

W pierwszej kolejności zostanie zaprezentowane rozwiązanie realizujące podstawowe wymagania wyborów dla małych grup. Następnie zaprezentowane zostanie udoskonalenie pozwalające zmniejszyć ilość obliczeń oraz publikowanych danych. W dalszej kolejności pokazane zostanie w jaki sposób można wprowadzić super-wyborców. Zaprezentowany system będzie umożliwiał wybór typu 1-z-n. Do tego celu wykorzystane zostaną urządzenia wyposażone w ekran. Będą one wyświetlały listę kandydatów oraz będą pozwalały przesuwać tę listę i zaznaczyć wybranego kandydata. Możliwe będzie również przesunięcie o losową liczbę pozycji numeru wybranego kandydata w celu zakodowania głosu (patrz rys. 3.11). Czynność tą można wykonać w sposób wygodny i intuicyjny wykorzystując współczesny tablet oraz aplikację wyświetlającą dwukolumnową listę przewijaną kandydatów oraz numerów ich pozycji (zarówno kandydaci jak i numery powtarzają się cyklicznie). Po wybraniu kandydata kolumna z nazwiskami zostaje zablokowana. W następnym kroku użytkownik przesuwa prawą kolumnę z numerami o losową ilość pozycji i zatwierdza. W rezultacie zostaje wyświetlone zaszyfrowane przesunięcie c_d.

Oznaczenia

- Uczestnicy:
  - ▷ EA – zaufana strona trzecia, serwer przeprowadzający wybory,
  - ▷ V – wyborca,
  - ▷ $Dev(V)$ – urządzenie wyborcy;
- m – liczba kandydatów, wyświetlanych przez urządzenie podczas głosowania,
- $p, g, y; x$ – parametry kryptosystemu ElGamala (patrz 1.2.2).

Przygotowanie głosowania

1. EA wybiera p, g, x, y, h takie, że g jest generatorem w $\mathbb{Z}_p^*$, natomiast h generuje podgrupę rzędu m i jest wykorzystywany by realizować szyfrowanie homomorficzne (1.2.6).
2. $EA \rightarrow Dev(V) : p, g, y, h$

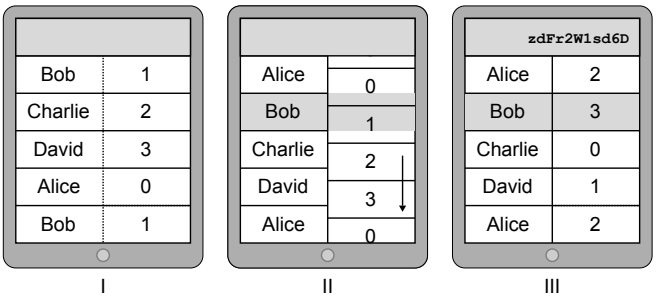
Głosowanie Poniżej przedstawimy jak wygląda głosowanie z punktu widzenia jednego wyborcy.

1. V wybiera kandydata v, zatwierdza, a następnie przesuwa listę numerów przyporządkowanych kandydatom o d pozycji, powodując jego zakodowanie. Przesunięcie jest cykliczne modulo liczba kandydatów (patrz rysunek poniżej).

2. Urządzenie $Dev(V)$ tworzy szyfrogram ElGamala $c_d = (h^d y^k, g^k)$ kodujący przesunięcie d, gdzie k jest losową wartością wybieraną przez urządzenie wykorzystywaną przez algorytm szyfrujący (patrz 1.2.2).

3. Urządzenie $Dev(V)$ przesyła do EA zaszyfrowany głos $c = d + v \mod m$ oraz szyfrogram zawierający przesunięcie (na rysunku poniżej, w kroku III jest wyświetlany u góry ekranu).

$$Dev(V) \rightarrow EA : c, c_d$$



Rysunek 3.11: Proces kodowania głosu przez wyborcę: I – przeglądanie listy kandydatów, II – wybór kandydata i rozpoczęcie procesu kodowania, III – zakodowany głos (u góry ekranu wartość c_d)

4. EA publikuje na tablicy pary (c, c_d) wszystkich wyborców, którzy oddali głos. Dzięki temu wyborca może sprawdzić, czy jego głos został uwzględniony. Szyfrogram c_d jest z dużym prawdopodobieństwem unikalny, dzięki czemu oddający głos może w sposób jednoznaczny zidentyfikować swój głos.

Zliczanie EA odszyfrowuje i publikuje oddane głosy v obok par (c, c_d). W tym celu EA zamienia szyfrogram c_d zawierający przesunięcie na szyfrogram zawierający głos c_v.

$$c_v = (h^{-c} \cdot h^d y^k, g^k) = (h^{-v} y^k, g^k)$$

Dodatkowo EA może opublikować dowody z wiedzą zerową pokazujące, iż deszyfracja przebiegła prawidłowo. Proces deszyfracji homomorficznej wersji algorytmu ElGamala przebiega w sposób opisany w podrozdziale 1.2.6, natomiast tworzenie i weryfikowanie dowodów poprawnego deszyfrowania zostało przedstawione w podrozdziale 1.2.3. Głosujący mogą zapisać sobie wartości (c, c_d) jako potwierdzenie oddania głosu. Może ono zostać później wykorzystanie, jeśli wyborca będzie chciał zgłosić zastrzeżenia co do przebiegu wyborów.

Uwagi

- Można dodatkowo zanonimizować szyfrogramy c_v przepuszczając je przez sieć mieszającą lub odszyfrować dwuetapowo jak w protokole Punchscan (patrz 3.2.6).
- Wartości y^k, g^k mogą być przygotowane wcześniej, tak aby w czasie głosowania urządzenie nie musiało wykonywać złożonych obliczeń (potęgowań o dużych wykładnikach). Jest to istotne szczególnie wtedy, gdy urządzenie posiada ograniczoną moc obliczeniową.

Wykorzystanie liczników homomorficznych

Zaprezentowane powyżej rozwiązanie wymaga publikowania wielu dowodów poprawności deszyfrowania (patrz 1.2.3). Ich pełna weryfikacja wymaga sprawdzenia każdego indywidualnego głosu. Okazuje się, iż możemy zmodyfikować protokół tak, aby umożliwiał zsumowanie zaszyfrowanych głosów c_v poprzez ich przemnożenie. Następnie wystarczy odszyfrować wynik wyborów oraz opublikować dowód poprawności tej operacji. Wykorzystane zostaną własności homomorficzne funkcji szyfrującej oraz możliwość zrealizowania wielu liczników w jednym szyfrogramie (podobnie jak w protokole 3.2.5). Modyfikacja wymaga zmiany etapu przygotowania protokołu.

Dodatkowe oznaczenia

- w – liczba wyborców, $b = w + 1$ – podstawa systemu pozycyjnego wykorzystywanego do szyfrowania wielu liczników.

Przygotowanie

1. EA wybiera parametry kryptosystemu ElGamala takie, że istnieje generator h podgrupy $\mathbb{Z}_p^*$, który jest rzędu b^m,
2. $EA \rightarrow Dev(V) : p, g, y, h$

Głosowanie Głosowanie przebiega w taki, podobnie jak poprzednio. Jedyna różnica polega na sposobie wyliczenia c_d : $c_d = (h^{(b^{-d})} \cdot y^k, g^k)$.

Zliczanie Aby możliwe było sumowanie zaszyfrowanych głosów c_v należy zmienić sposób wyliczania c_v.

$$c_v = ((h^{(b^{-d})} \cdot y^k)^{b^c}, (g^k)^{b^c}) = (h^{(b^v)} \cdot y^{k'}, g^{k'})$$

Należy zwrócić uwagę, iż sumowanie głosów odbywa się w wielu licznikach jednocześnie. Odczytanie wyników sumowania wymaga spojrzenia na zapis sumy głosów w systemie pozycyjnym o podstawie b. Głos oddany na kandydata v, będzie zaszyfrowaną zmodyfikowanym (1.2.6) algorytmem ElGamala wartością b^v. Po wymnożeniu tego typu szyfrogramów otrzymamy szyfrogram postaci:

$$(h^{b^{v_1}+b^{v_2}+\cdots+b^{v_n}} \cdot y^{k''}, g^{k''}),$$

z którego w standardowy sposób możemy „odzyskać":

$$b^{v_1} + b^{v_2} + \cdots + b^{v_n}.$$

Wartość powyższa jest równa:

$$n_0 b^0 + n_1 b^1 + \cdots + n_{m-1} + b^{m-1},$$

gdzie $n_0, n_1, \ldots, n_{m-1}$ to kolejne cyfry wyniku sumowania głosów zapisanego w systemie pozycyjnym o podstawie b. Wartości te są jednocześnie wynikami wyborów, tzn. liczbami głosów oddanych na poszczególnych kandydatów lub na poszczególne opcje.

Wykorzystanie liczników homomorficznych i super-wyborców

Poprzednie wersje protokołu zapewniały anonimowość dzięki anonimowemu przydzieleniu urządzeń do głosowania. Modyfikacja zaprezentowana powyżej pozwala zapewnić anonimowość wyborców nawet, jeżeli znany jest sposób przydzielenia urządzeń. Jest to możliwe jeżeli odszyfrowanie sumy głosów wymaga współpracy wielu uczestników protokołu wyborczego, a tym samym nie istnieje wyróżniony uczestnik, który mógłby odszyfrowywać pojedyncze głosy. Grupę mającą możliwość deszyfrowania będzie stanowiła pewna grupa wyróżnionych wyborców – tzw. super-wyborców.

Dodatkowe oznaczenia

- $\hat{V}$ – super-wyborca.

Przygotowanie głosowania

1. EA wybiera parametry kryptosystemu ElGamala p, g, h takie, że g jest generatorem w $\mathbb{Z}_p^*$, natomiast h generuje podgrupę rzędu b^m,

2. Super-wyborcy tworzą przy współpracy klucz publiczny y, oraz cienie klucza prywatnego, które pozostają sekretami poszczególnych super-wyborców (patrz 1.2.7). Klucz publiczny wraz z pozostałymi parametrami wyborów jest dystrybuowany przez EA.

3. $EA \rightarrow Dev(V) : p, g, y, h$

Pozostałe etapy wyborów przebiegają w ten sam sposób, poza jednym krokiem. Wyborca oprócz zakodowanego głosu przesyła również identyfikator id_V. Identyfikator może być tożsamy z identyfikatorem urządzenia $Dev(V)$ lub może zawierać dane identyfikujące tożsamość głosującego. Na telebimie są zatem publikowane trójki: id_V, c, c_v. Po zsumowaniu zaszyfrowanych głosów super-wyborcy przystępują do deszyfrowania wyniku wyborów. W tym celu zaszyfrowany wynik wyborów jest sekwencyjne przetwarzany przez kolejnych super-wyborców, a wyniki ich działań są publikowane dla późniejszej weryfikacji.

Odtajnienie wyborów Dzięki zastosowaniu szyfrowania progowego (patrz 1.2.7) przez super-wyborców można, jeżeli zajdzie taka potrzeba, odtajnić całe wybory. Może to mieć miejsce, jeżeli uczestnicy chcą sprawdzić prawidłowość działania sytemu lub jeżeli jest to wymagane z przyczyn proceduralnych. Proces odtajnienia wyborów można zrealizować na dwa sposoby. Super-wyborcy mogą odtworzyć i ujawnić klucz prywatny x, co pozwala automatycznie odszyfrować poszczególne głosy. Inna możliwość polega na odszyfrowaniu poszczególnych głosów w sposób kolaboracyjny (przez współpracujących super-wyborców), tak jak to ma miejsce w przypadku deszyfrowania wyniku wyborów. W drugim przypadku można również selektywnie odszyfrować tylko poszczególne głosy nie naruszając tajności pozostałych. Jeżeli identyfikatorem głosującego id_V jest identyfikator urządzenia to odtajnienie głosowania nie powoduje naruszenia anonimowości wyborów – znane są wszystkie głosy i to z jakiego urządzenia zostały oddane, lecz nie wiadomo kto z danego urządzenia korzysta w czasie głosowania.

3.5.4 Wymagania obliczeniowe protokołu

Protokół w zmodyfikowanej wersji wykorzystuje generator podgrupy h, który może być dosyć dużego rzędu. W przypadku głosowania z udziałem 150

wyborców na jedną z trzech opcji potrzebujemy h, który jest rzędu 151^3. Obliczenie logarytmu dyskretnego o podstawie h metodą brutalną lub poprzez skatalogowanie wszystkich elementów generowanej przez h grupy wymaga skatalogowania 3375000 wartości (ok. 1GB). Jest to znacznie mniej niż gdybyśmy kodowali głosy przy pomocy zapisu binarnego, tak jak ma to miejsce w protokole Scrach & Vote (3.2.5). Zapis binarny wymagałby poświęcenia 8 bitów na każdy licznik, a zatem h byłoby rzędu 256^3. W tym przypadku należałoby skatalogować ok. 4GB danych.

Ilość danych, które należy skatalogować rośnie bardzo szybko wraz ze wzrostem liczby kandydatów. W konsekwencji realizacja głosowania typu 1-z-n lub m-z-n może okazać się trudna obliczeniowo (patrz tabela poniżej). Możemy jednak tego typu wybory przeprowadzić jako serię wyborów typu 1-z-2 lub 1-z-3 ma kolejnych kandydatów. Dodatkowo można zweryfikować, czy liczba głosów pozytywnych zgadza się z liczbą wyborców. W przypadku nieprawidłowości można zsumować głosy poszczególnych wyborców w postaci zaszyfrowanej i sprawdzić, czy oddali odpowiednią ilość głosów pozytywnych. Co więcej proces ten może się odbywać bez odkrywania poszczególnych głosów danego wyborcy.

Ilość opcji	Il. głosujących 4	8	16	32
4	625	6 561	83 521	1 185 921
5	3 125	59 049	1 419 857	39 135 393
6	15 625	531 441	24 137 569	**1 291 467 969**
7	78 125	4 782 969	410 338 673	$4,26E+10$
8	390 625	43 046 721	**6 975 757 441**	$1,41E+12$
9	1 953 125	387 420 489	$1,19E+11$	$4,64E+13$
10	9 765 625	**3 486 784 401**	$2,02E+12$	$1,53E+15$
11	48 828 125	$3,14E+10$	$3,43E+13$	$5,05E+16$
12	244 140 625	$2,82E+11$	$5,83E+14$	$1,67E+18$
13	1 220 703 125	$2,54E+12$	$9,90E+15$	$5,50E+19$
14	**6 103 515 625**	$2,29E+13$	$1,68E+17$	$1,82E+21$

Tabela 3.6: Rozmiar katalogu wartość w zależności od liczb głosujących oraz opcji do wyboru (wartości brzegowe, takie dla których wybory $1-z-n$ przestają być możliwe, zostały wytłuszczone)

Rozdział 4

Egzaminy elektroniczne

Ostatnie lata przyniosły gwałtowny wzrost zainteresowania wszelkimi systemami związanymi ze zdalnym nauczaniem (tzw. e-learningiem). Ciekawym zjawiskiem obserwowanym w ostatnim czasie jest pojawienie się platform MOOC (ang. Massive Online Open Course) umożliwiających udostępnienie kursów na poziomie uniwersyteckim bardzo dużym grupom odbiorców. W roku 2011 na Uniwersytecie Stanforda zorganizowano kilka kursów tego typu z dziedzin techniczno-informatycznych, które zebrały powyżej 100 000 słuchaczy. Równocześnie, problem bezpiecznego testowania wiedzy na odległość nie doczekał się dotychczas przekonywującego rozwiązania. Fakt ten wynika z trudności spełnienia wymagań stawianych tego typu systemom. Twórcy systemów służących do elektronicznego przeprowadzania egzaminów powinni zapewnić bezpieczeństwo co najmniej na poziomie tradycyjnych, „papierowych" egzaminów, jednocześnie oferując korzyści rozwiązania elektronicznego (oszczędność kosztów i czasu, wygoda, itp.). Nietrudno jest zauważyć, iż sprawdzenie tożsamości studenta lub zapewnienie tego, że nie będzie ściągał jest o wiele prostsze w przypadku tradycyjnych egzaminów niż dla ich elektronicznych, zdalnych odpowiedników.

Poniższy rozdział zawiera charakterystykę oraz krótki opis istniejących rozwiązań e-egzaminów, który został zawarty w pracach przeglądowych [26, 27]. W podrozdziale 4.3 przedstawione zostały przygotowane do druku propozycje autora rozprawy nowych rozwiązań problemu zdalnego egzaminowania wykorzystujące mechanizm dwuetapowego Randomized Partial Checking (patrz 2.5).

4.1 Charakterystyka systemów egzaminów elektronicznych

System elektronicznego egzaminowania ma realizować te same cele co egzaminy tradycyjne. Działanie e-egzaminów opiera się na wykorzystaniu technologii informatycznych bez użycia papieru jako nośnika danych (pytań egzaminacyjnych, odpowiedzi, wyników). Przeprowadzenie egzaminu wymaga uczestnictwa wielu jednostek, zgrupowanych wedle odgrywanych przez nich ról. Poniżej opisane zostały podstawowe, ogólne role uczestników protokołów elektronicznego egzaminowania oraz ich wpływ na bezpieczeństwo systemu.

Student (S) – zdający egzamin. Pobiera zestaw pytań egzaminacyjnych, a następnie odsyła odpowiedzi na te pytania. Przeważnie po przeprowadzeniu egzaminu uzyskuje dostęp do otrzymanej oceny. Zakładamy, że może być nieuczciwy, to znaczy może próbować wykorzystać luki w samym systemie lub stosować nieuczciwe praktyki będące poza kontrolą systemu.

Nauczyciel (T) – (ang. *teacher*), sprawdzający egzaminy i wystawiający oceny. W niektórych systemach egzaminowania nauczyciel jest również odpowiedzialny za przygotowanie pytań. Zakładamy, iż może działać nieuczciwie – w zmowie z ocenianym uczniem.

Zarządzający egzaminem (EA) – (ang. *exam authority*), zaufany uczestnik zarządzający procesem egzaminowania. Pośredniczy w komunikacji pomiędzy studentem a nauczycielem. Zarządzających egzaminem może być wielu, mogą posiadać różny zakres kompetencji.

Sieć mieszająca (M) – realizuje anonimowy kanał informacyjny. Obejmuje wielu zaufanych uczestników oznaczanych $M_1, M_2, \ldots, M_\lambda$ (patrz 2.1).

4.1.1 Wymagania

Do porównywania i opisywania różnych systemów definiuje się pewien zestaw wymagań, które powinny one spełniać. W załączonych poniżej opisach zwróciliśmy również uwagę na różnice, występujące w definicjach pojęć w systemach proponowanych przez różnych autorów.

Autentyczność (ang. *authenticity*) – na każdym etapie egzaminu należy zapewnić autoryzację zarówno nauczycieli, studentów, jak i samego systemu. Nie można oczywiście ograniczyć się do samej autoryzacji. Należy

sprawdzić, czy dane osoby mogą pełnić swoją funkcję w systemie (nauczyciel musi mieć uprawnienia do przeprowadzania egzaminów z danego przedmiotu, student musi zostać dopuszczony do egzaminu i może do niego przystąpić tylko raz) [34, 41, 8].

Tajność (ang. *secrecy*) – pytania oraz poprawne odpowiedzi muszą być utrzymywane w tajemnicy przez cały czas trwania egzaminu. Nie można także dopuścić do ujawnienia odpowiedzi udzielonych przez studenta (także po zakończeniu procesu egzaminowania). Każdy student powinien mieć dostęp wyłącznie do ocen swoich egzaminów [34, 41]. Należy zaznaczyć, iż w niektórych sytuacjach oceny studentów nie muszą być im znane – mogą pozostać tajemnicą organizującego egzamin.

Integralność (ang. *integrity*) – zarówno pytania, jak i odpowiedzi powinny być sprawdzane pod kątem nieautoryzowanych modyfikacji, np. zmian w udzielonych odpowiedziach po zakończeniu egzaminu, modyfikacji pytań, etc.

Anonimowość (ang. *anonymity*) – w niektórych rozwiązaniach (np. opisywanym w [34]) wymagana jest anonimowość. Sprowadza się ona do wymagania, aby egzaminator nie wiedział kogo ocenia (np. studenci otrzymują pseudonimy na czas trwania egzaminu), a student nie wiedział kto jest jego egzaminatorem (przy założeniu, że mamy więcej niż jednego egzaminatora).

Rozproszenie zaufania (ang. *distribution of trust*) – anonimowość jest przeważnie zapewniona poprzez zaufanego zarządzającego egzaminem lub poprzez wielu zarządzających egzaminem. Mówimy o rozproszeniu zaufania jeżeli zarządzających egzaminem jest wielu oraz żaden z nich nie posiada wiedzy o tym, jakich studentów sprawdzali poszczególni nauczyciele [34].

Potwierdzenie (ang. *receipt*) – student, po zakończeniu egzaminu, otrzymuje potwierdzenie wysłania swoich odpowiedzi [34, 8].

Niewypieralność (ang. *non-repudiation*), czasami określana jako niezaprzeczalność – własność protokołu zapewniająca, że „wykonawca" pewnej czynności (np. przesłania pewnych odpowiedzi) nie może się jej wyprzeć, nawet jeżeli byłoby to w jego interesie.

Poprawność (ang. *correctness*) – oznacza sprawdzenie, czy student nie próbuje po raz drugi podejść do tego samego egzaminu [34].

Odporność na oszustwa (ang. *copy prevention and detection*), przy czym rozważane są różne ich rodzaje [41], na przykład:

- zdawanie egzaminu przez inną osobę (podszywanie się),
- konsultowanie odpowiedzi z innymi egzaminowanymi,
- używanie nieautoryzowanych materiałów,
- przechwytywanie lub zakłócanie komunikacji pomiędzy egzaminowanym a systemem.

Kontrola (ang. *monitoring*) – w niektórych przypadkach zapewnienie odporności na oszustwa można osiągnąć poprzez zastosowanie monitoringu, który będzie dostarczał materiału dowodowego dla nadużyć oraz pełnił rolę odstraszającą. W przypadku egzaminów zdalnych monitoring może polegać na strumieniowej transmisji wideo z kamery internetowej komputera osobistego [41].

Dostępność (ang. *accessibility*) – rozpatrywany typ egzaminów powinien być dostępny bez względu na lokalizację oraz w miarę możliwości niezależny od posiadanego sprzętu ([41]). Niektóre rozwiązania dają możliwość egzaminowania studenta znajdującego się w środowisku niekontrolowanym przez egzaminującego (np. w domu egzaminowanego), inne natomiast wymagają od egzaminowanego udania się do specjalnego centrum egzaminacyjnego (środowisko kontrolowane przez instytucję egzaminującą). Podejścia te oferują różny stopień dostępności.

4.2 Przegląd istniejących rozwiązań

Literatura poświęcona protokołom bezpiecznego egzaminowania przez Internet jest stosunkowo skromna w porównaniu z innymi działami kryptologii. Istnieje stosunkowo niewiele dokładnie opisanych protokołów. Poniższy podrozdział zawiera opis trzech przykładowych rozwiązań problemu zdalnego egzaminowania – począwszy od rozwiązań stosunkowo prostych do dość skomplikowanych systemów egzaminujących. W przypadku protokołów, którym twórcy nie nadali nazw będziemy stosować skróty pochodzące od pierwszych liter nazwisk ich autorów.

Poniżej opisany zestaw rozwiązań pomija protokół SeCOnE [41], który jest dość skomplikowany ze względu na liczbę typów uczestników oraz podziały kompetencyjne między nimi. Warto jednak podkreślić, że jego twórcy dużą wagę przywiązują do monitorowania zachowania ucznia zdającego

egzamin. Komputer ucznia powinien zostać wyposażony w kamerę internetową oraz mikrofon. Uczeń jest zobowiązany do zainstalowania specjalnego oprogramowania, które umożliwia udzielenie odpowiedzi na pytania, ale jednocześnie przesyła obraz wideo wraz ze zrzutami ekranów (po 2 na sekundę). Dodatkowo, blokuje wszystkie inne programy oraz swobodny dostęp do sieci. Dzięki zastosowaniu tak radykalnych środków możliwe jest sprawdzenie wiedzy ucznia w środowisku niekontrolowanym przez szkołę (np. w domu ucznia). Należy jednak pamiętać o zwiększonych wymogach dotyczących przepustowości łącza instytucji egzaminującej (ok. 700 MB na ucznia, na godzinę egzaminu) oraz przestrzeni dyskowej (przebieg egzaminu powinien być archiwizowany, gdyż może posłużyć jako dowód). Z punktu widzenia kryptologii protokół bazuje na infrastrukturze klucza publicznego, szyfrowaniu symetrycznym, oraz protokole Diffiego-Hellmana. System został zaprojektowany z myślą o egzaminach z matematyki i języka angielskiego dla uczniów gimnazjów i liceów. Podobnie jak protokół CHP (patrz podrozdział 4.2.1) został zaimplementowany.

4.2.1 Protokół CHP

Protokół CHP [9] jest bardzo prostą próbą zmierzenia się z problemem zdalnego egzaminowania. Twórcy rozwiązania przyjęli, iż studenci będą zdawać egzamin w pomieszczeniach kontrolowanych przez instytucję egzaminującą. Zdający będą odpowiadać na pytania za pomocą przenośnych terminali wyposażonych w łączność bezprzewodową (smartfony , tablety, etc...). Zakłada się, że terminal jest kontrolowany przez przeprowadzającego egzamin, a nie studenta. Dodatkowo każdy z użytkowników posiada parę kluczy (prywatny i publiczny), która umożliwia korzystanie z podpisów cyfrowych. Prostota protokołu, a w szczególności brak wielu egzaminujących jest przyczyną jednej z głównych wad rozwiązania – braku anonimowości.

Uczestnicy i podstawowe oznaczenia

- S – zdający egzamin,
- T – organizator i sprawdzający egzamin,
- g – generator dużej podgrupy w $\mathbb{Z}_p^*$ znany wszystkim uczestnikom protokołu,
- $cert_S, cest_T$ – certyfikaty kluczy publicznych S oraz T.

Rejestracja
Celem rejestracji jest uzyskanie klucza sesyjnego k, który będzie wykorzystywany w dalszych czynnościach protokołu.

1. S wybiera wartość losową r_S, a następnie wysyła:
$$S \rightarrow T : g^{r_S}.$$
2. T wybiera losowe r_T oblicza $k = (g^{r_S})^{r_T}$ i odsyła:
$$S \leftarrow T : g^{r_T}, \mathcal{E}_k(\mathcal{P}_T(g^{r_S}, g^{r_T})), cert_T.$$
3. S wyznacza $k = (g^{r_T})^{r_S}$, sprawdza jego prawidłowość weryfikując otrzymany podpis $\mathcal{P}_T(g^{r_S}, g^{r_T})$. Następnie opowiada (potwierdzając znajomość k):
$$S \rightarrow T : \mathcal{E}_k(\mathcal{P}_S(g^{r_S}, g^{r_T})), cert_S.$$

Pobranie pytań
W tym etapie protokołu zdający egzamin uzyskuje pytania egzaminacyjne i potwierdza ich otrzymanie.

1. T przesyła S pytania q egzaminu wraz z aktualnym znacznikiem czasu t:
$$S \leftarrow T : \mathcal{E}_k(\mathcal{Q}, t).$$
2. S odpowiada podpisem pod wartością funkcji haszującej h na otrzymanych danych:
$$S \rightarrow T : \mathcal{P}_S(h(\mathcal{Q}, t)).$$

Przesłanie odpowiedzi
Zdający egzamin przy pomocy aplikacji uruchomionej na swoim terminalu udziela odpowiedzi $\mathcal{A}$, otrzymuje potwierdzenie, a następnie przesyła odpowiedzi na serwer nauczyciela T.

1. S przesyła wartość funkcji haszującej, której argumentem są udzielone odpowiedzi $\mathcal{A}$:
$$S \rightarrow T : h(\mathcal{A}).$$
2. T odsyła potwierdzenie (w formie podpisu) udzielonych odpowiedzi oznakowane czasem t':
$$S \leftarrow T : t', \mathcal{P}_T(h(\mathcal{A}), t').$$
3. S przesyła odpowiedzi:
$$S \rightarrow T : \mathcal{E}_k(\mathcal{A}).$$
4. T weryfikuje otrzymany uprzednio hasz odpowiedzi.

4.2.2 Protokół CHD

Protokół CHD został opisany w pracy *A Secure E-Exam Management System* [8]. Jednym z głównych celów przyświecających jego autorom było zapewnienie anonimowości studenta wobec sprawdzającego go nauczyciela. Cel ten został osiągnięty dzięki wykorzystaniu zaufanej strony trzeciej, zwanej w tym przypadku menadżerem. Menadżer pośredniczy w komunikacji pomiędzy studentami i nauczycielami, przy okazji „zacierając" ślady o tym, które odpowiedzi egzaminacyjne pochodzą od danego studenta. W oczywisty sposób menadżer jest postawiony w roli, która wymaga pełnego zaufania do niego. Wartym uwagi walorem tego protokołu jest fakt, iż został zaimplementowany.

Założenia wstępne
Każdy z użytkowników posiada parę kluczy (prywatny i publiczny), która umożliwia realizację uwierzytelnionego i/lub prywatnego kanału komunikacji (patrz uwaga 1.1.3).

Uczestnicy

- S – student, egzaminowany;
- T – nauczyciel, egzaminator;
- EA – menadżer, zarządzający przebiegiem egzaminu, pośredniczy pomiędzy egzaminującym a egzaminowanym.

Przygotowanie egzaminu

1. T wyznacza identyfikator id egzaminu. Identyfikator zawiera następujące informacje o egzaminie: temat, kod tematu, semestr, data, ustalony przedział czasowy na udzielenie odpowiedzi, numer seryjny egzaminu.
2. T formułuje i przesyła pytania egzaminacyjne $\mathcal{T}$ uwierzytelnionym kanałem prywatnym:
$$T \xrightarrow[prv]{auth} EA : id, \mathcal{T}.$$
3. EA zapamiętuje dane $(id, \mathcal{T}, s_T)$ przesłane przez T, gdzie $s_T = \mathcal{P}_T(id, \mathcal{T})$ jest podpisem T przesyłanym w poprzednim kroku w celu realizacji kanału uwierzytelnionego.

Przeprowadzenie egzaminu

1. T publikuje identyfikator egzaminu id.
2. S poprzez uwierzytelniony kanał prosi EA o pytania egzaminu (podając id egzaminu).
3. EA weryfikuje, czy:
   (a) S jest uprawniony do zdawania egzaminu,
   (b) aktualna data i czas odpowiadają przedziałowi czasowemu egzaminu.
4. Jeżeli weryfikacja w punkcie 3 powiodła się, to
$$EA \xrightarrow{prv} S : id, \mathcal{T}, s_T.$$
5. S weryfikuje otrzymane dane, odpowiada na pytania egzaminacyjne i wysyła je. Dokładniej S:
   (a) weryfikuje podpis T pod $(id, \mathcal{T})$,
   (b) udziela i zapisuje swoje odpowiedzi $\mathcal{A}$ na pytania $\mathcal{T}$,
   (c) wyznacza losową liczbę ρ, która będzie pełniła rolę identyfikatora odpowiedzi dla studenta,
   (d) tworzy podpis $s_S = \mathcal{P}_S(s_T, \rho, \mathcal{A})$,
   (e) $S \xrightarrow{prv} EA : \mathcal{E}, id, s_T, \rho, \mathcal{A}, s_S$.
6. EA wykonuje wymienione poniżej czynności.
   (a) Sprawdza, czy data i czas otrzymania odpowiedzi od S mieszczą się w przedziale czasowym przypisanym egzaminowi.
   (b) Sprawdza, czy S nie przesyłał wcześniej odpowiedzi na pytania zawarte w danym egzaminie.
   (c) Sporządza podpis $s = \mathcal{P}_{EA}(id, \rho, \mathcal{A}, t)$ (t – aktualny czas), który będzie potwierdzeniem przesłania odpowiedzi $\mathcal{A}$ dla S.
   (d) $EA \longrightarrow S : id, \rho, t, s$.
   (e) EA wyznacza liczbę losową ρ' („zamaskowany” identyfikator odpowiedzi przeznaczony dla nauczyciela T), a następnie tworzy podpis $s' = \mathcal{P}_{EA}(s_T, \rho', \mathcal{A})$.
   (f) Wszystkie dane są przechowywane w bezpieczny sposób.

Ocenianie egzaminu

1. T uwierzytelnia się przed EA, a następnie wysyła prośbę o jeden zestaw odpowiedzi na pytania z egzaminu id.
2. $EA \xrightarrow{prv} T : \mathcal{T}, id, s_T, \mathcal{A}, \rho', s'$.
3. T ocenia odpowiedzi $\mathcal{A}$ na ocenę $\mathcal{G}$.
4. $T \xrightarrow[prv]{auth} EA : \mathcal{T}, id, s_T, \mathcal{A}, \rho', \mathcal{G}$.
5. EA znajduje ρ odpowiadające ρ' i przechowuje wszystkie dane w bezpieczny sposób.

Otrzymanie wyniku egzaminu

1. S uwierzytelnia się wobec EA, a następnie wysyła zapytanie o wyniki egzaminu o identyfikatorze odpowiedzi ρ.
2. EA sprawdza, czy ρ jest skojarzone z danym studentem.
3. $EA \xrightarrow[prv]{auth} S : \mathcal{T}, id, s_T, \rho, \mathcal{A}, \mathcal{G}, s_S, \mathcal{P}_T(\mathcal{T}, id, s_T, \mathcal{A}, \rho', \mathcal{G})$.
4. S weryfikuje otrzymane dane.

Ponowne ocenienie egzaminu
W przypadku wystąpienia wątpliwości student ma możliwość ubiegania się o ponowne sprawdzenie egzaminu. W tym celu wykonywane są następujące kroki.

1. Student S wybiera nowy anonimowy identyfikator ρ^*.
2. Uwierzytelnia się wobec EA i przesyła
$$S \xrightarrow{auth} EA : id, \rho, \rho^*.$$
3. EA weryfikuje i zapisuje otrzymane dane. Nauczyciel przy pomocy zmodyfikowanego protokołu oceniania egzaminu wystawia nową ocenę. Autorzy nie uściślają na czym polega modyfikacja protokołu oceniania.

4.2.3 Protokół HP

Twórcy protokołu HP [34, 35] położyli duży nacisk na zapewnienie rozproszenia zaufania do organizatorów elektronicznych egzaminów. Ich celem było zagwarantowanie, aby żaden z uczestników nie posiadał wiedzy pozwalającej odtworzyć połączenie pomiędzy zdającym egzamin studentem, a sprawdzającym go egzaminatorem. Jest to cecha unikalna na tle innych proponowanych rozwiązań. Została ona osiągnięta dzięki zastosowaniu zmodyfikowanych sieci mieszających, które umożliwiają przesyłanie w obie strony (patrz opisany poniżej anonimowy kanał zwrotny). Podobnie jak w poprzednich rozwiązaniach zakłada się, iż egzaminy odbywają się w kontrolowanych pomieszczeniach pilnowanych przez odpowiednie osoby.

Uczestnicy

- S – student, zdający egzamin;
- T – nauczyciel, sprawdzający egzamin;
- EA – pełni dwie zasadnicze role:
  - ▷ rejestruje użytkowników, zapewnia infrastrukturę klucza publicznego dla pozostałych uczestników, generuje parametry początkowe systemu;
  - ▷ zarządza przebiegiem egzaminu, wyznacza pseudonimy dla właściwych uczestników, wybiera nauczyciela sprawdzającego danego studenta, wpisuje ocenę do bazy danych.
- $\mathcal{M}$ – grupa n serwerów $M_1, M_2, \ldots, M_n$ realizująca dwie funkcje: anonimowy kanał zwrotny (sieć mieszająca) oraz usługa opóźnionego ujawniania anonimowych identyfikatorów.

Narzędzia

ElGamal

Zgodnie z opisem kryptosytemu ElGamala (patrz 1.2.2) niech p będzie modułem, natomiast q dużą liczbą pierwszą taką, że $q|p-1$. Wybierany jest również wspólny generator g rzędu q w $\mathbb{Z}_p^*$, który pozwoli użytkownikowi U wyznaczyć parę kluczy $(\widehat{pk}_U, \widehat{sk}_U)$ taką, że $\widehat{pk}_U = g^{\widehat{sk}_U} \mod p$. Dodatkowo identyfikatorami użytkowników U sytemu będą inne generatory $g_U \in \mathbb{Z}_p^*$ rzędu q. Szyfrogram utworzony kluczem U, zawierającym wiadomość m będziemy oznaczać $\mathcal{E}_U(m,k)$ lub $\mathcal{E}_{\widehat{pk}_U}(m,k)$, gdzie k jest wartością losową używaną

przez algorytm ElGamala. W miejscach, w których nie jest istotne podkreślanie losowego charakteru algorytmu ElGamala parametr k będzie pomijany, a zapis będzie skracany do $\mathcal{E}_U(m)$. Należy dodać, iż autorzy błędnie sugerują wykorzystanie g_U do stworzenia $\widehat{pk}_U$ – mechanizm zamiany szyfrogramów opisany poniżej wymaga ujednoliconego generatora g.

Własności algorytmu ElGamala pozwalają stworzyć cebulę $\mathcal{E}_{\mathcal{M}}(m)$ zawierającą pewną wiadomość zaszyfrowaną kluczami serwerów mieszających ($\widehat{pk}_{\mathcal{M}} = \widehat{pk}_{M_1} \cdot \ldots \cdot \widehat{pk}_{M_n}$). Cebula ta może być przeszyfrowywana lub zdeszyfrowana przez współdziałające serwery $\mathcal{M}$ (patrz podrozdział **??**).

Uwaga 4.2.1 *Zamiana cebuli $\mathcal{E}_{\mathcal{M}}(m,k)$ na szyfrogram (bez ujawniania m) $\mathcal{E}_U(m,k')$ wymaga, aby przed kolektywnym deszyfrowaniem cebuli przez $\mathcal{M}$ zaszyfrować ją dodatkowo $\mathcal{E}_U(\mathcal{E}_{\mathcal{M}}(m,k),k')$. Dzięki własnościom algorytmu ElGamala otrzymana w ten sposób cebula (z dodatkową warstwą) jest równoważna $\mathcal{E}_{\mathcal{M}}(\mathcal{E}_U(m,k'),k)$. Na koniec wymagane jest wspólne odszyfrowanie (bez mieszania) przez serwery $\mathcal{M}$ szyfrogramu zawartego w cebuli. Twórcy opisanego poniżej anonimowego kanału zwrotnego sugerują wykorzystanie w celu zamiany szyfrogramów algorytmu opisanego w artykule [36].*

Anonimowy kanał zwrotny

Zaproponowana w tym rozwiązaniu, a opisana szczegółowo w pracy [30], sieć mieszająca wykorzystana jest do anonimowego przekazywania informacji w dwie strony (anonimowy kanał zwrotny) i różni się nieco od tradycyjnej sieci mieszającej (patrz podrozdział **??**). Użytkownicy przesyłają nie jeden szyfrogram ElGamala (jak to ma zwykle miejsce w przypadku sieci mieszających), lecz wektor szyfrogramów (pary, trójki, itd.). Liczba szyfrogramów w wektorze jest stała dla wszystkich użytkowników. Serwery przeszyfrowują oddzielnie każdy element wektora, a następnie mieszają wektory, nie zmieniając kolejności szyfrogramów w wektorze. Anonimowy kanał zwrotny jest realizowany w następujący sposób. Załóżmy, że Alicja chce wysłać anonimowo wiadomość m do Boba, oraz chce by Bob mógł na nią odpowiedzieć wiadomością r. Ponadto Bob może przekazać innej osobie możliwość odpowiedzenia Alicji udostępniając podpis dostarczany w kroku 4 wraz z wiadomością. Cała komunikacja pomiędzy stronami odbywa się przez sieć mieszającą w poniżej opisanych krokach.

1. **Ustawienia sieci** – serwery wspólnie generują publiczne i prywatne parametry kryptosystemu ElGamala. Ponadto serwery ustalają wspólny klucz do podpisu elektronicznego.

2. **Przesyłanie wiadomości** – Alicja przesyła wiadomość m do sieci. Przez id_A oznaczmy identyfikator Alicji, $\widehat{pk}_A$ jest kluczem publicz-

nym Alicji (analogicznie dla Boba – id_B, $\widehat{pk}_B$). Alicja przesyła do sieci trójkę szyfrogramów powstałych przy użyciu klucza publicznego sieci mieszającej $(\mathcal{E}_{\mathcal{M}}((id_A, \widehat{pk}_A), k_1), \mathcal{E}_{\mathcal{M}}(m, k_2), \mathcal{E}_{\mathcal{M}}((id_B, \widehat{pk}_B), k_3))$ oraz dowód znajomości (patrz podpunkt 1.2.3) $(id_A, \widehat{pk}_A)$ i $(id_B, \widehat{pk}_B)$.

3. **Mieszanie wiadomości** – po uzbieraniu odpowiedniej liczby wiadomości, są one mieszane i przeszyfrowywane w turach przez kolejne serwery.

4. **Dostarczenie wiadomości** – wyjściem z sieci mieszającej jest lista trójek postaci $(\mathcal{E}_{\mathcal{M}}((id_A, \widehat{pk}_A), k'_1), \mathcal{E}_{\mathcal{M}}(m, k'_2), \mathcal{E}_{\mathcal{M}}((id_B, \widehat{pk}_B), k'_3))$. Serwery wspólnie deszyfrują trzeci szyfrogram uzyskując $id_B, \widehat{pk}_B$. Następnie transformują szyfrogram $\mathcal{E}_{\mathcal{M}}(m, k'_2)$ na $\mathcal{E}_B(m, k''_2)$ (patrz uwaga 4.2.1) oraz generują podpis s pod $\mathcal{E}_{\mathcal{M}}((id_A, \widehat{pk}_A), k'_2)$. Autorzy [30] nie wskazują konkretnego schematu podpisu grupowego do utworzenia podpisu s. Bob otrzymuje poniższą trójkę wartości.

$$\mathcal{E}_{\mathcal{M}}((id_A, \widehat{pk}_A), k'_1), \mathcal{E}_B(m, k'_2), s$$

5. **Odpowiedź.** Bob, jako posiadacz $\widehat{sk}_B$, może w oczywisty sposób odczytać wiadomość m. Jeżeli natomiast chciałby odpowiedzieć nadawcy wiadomością r, to musi przesłać poniższe wartości do sieci mieszającej

$$\mathcal{E}_{\mathcal{M}}((id_B, \widehat{pk}_B), k_4), \mathcal{E}_{\mathcal{M}}(r, k_5), \mathcal{E}_{\mathcal{M}}((id_A, \widehat{pk}_A), k'_1)$$

oraz s i dowód znajomości $(id_B, \widehat{pk}_B)$. Sieć sprawdza również poprawność podpisu s pod zaszyfrowanymi danymi Alicji. Od tego momentu protokół przebiega w ten sam sposób co przesyłanie oryginalnej wiadomości.

Usługa opóźnionego ujawniania (ang. timed-release service)

Protokół opóźnionego ujawniania ma za zadanie zagwarantować, iż tożsamość studentów zostanie ujawniona dopiero na etapie ujawniania ocen. Podczas rejestracji identyfikator g_U studenta $(U = S)$ lub sprawdzającego $(U = T)$ jest zmieniany w sposób losowy przez sieć serwerów $\mathcal{M}$. W rezultacie powstaje identyfikator anonimowy ${g_U}^{\Gamma}$, dla którego nikt nie zna odpowiadającego g_U oraz Γ (jest on zaszyfrowany kluczem U). W odpowiednim momencie współdziałające serwery M_i mogą ponownie wyznaczyć g_U^{Γ} (nie zaszyfrowane kluczem U) i ujawnić powiązania pomiędzy g_U i g_U^{Γ} (patrz krok 4 poniżej opisanego protokołu).

1. Dla $i = 1$ do n wykonuj poniższe kroki.

(a) M_i otrzymuje $\mathcal{E}_{M_i}(g_U)$ od uczestnika U i odszyfrowuje.

(b) M_i oblicza $\mathcal{E}_{\mathcal{M}}(g_U^{\gamma_i})$, gdzie γ_i jest wybierane losowo dla U.

(c) M_i publikuje $\mathcal{E}_{\mathcal{M}}(g_U^{\gamma_i})$, wraz z dowodem o wiedzy zerowej znajomości γ_i.

(d) M_i bezpiecznie zapisuje (γ_i, g_U, t), gdzie t oznacza czas kiedy γ_i będzie mogło zostać ujawnione.

2. Wyliczane jest $\mathcal{E}_{\mathcal{M}}(g_U^{\Gamma}) = \prod_{i=1}^{n} \mathcal{E}_{\mathcal{M}}(g_U^{\gamma_i})$, gdzie $\Gamma = \sum_{i=1}^{n} \gamma_i$.

3. Sieć mieszająca zamienia, nie ujawniając zaszyfrowanej wartości, $\mathcal{E}_{\mathcal{M}}(g_U^{\Gamma})$ na $\mathcal{E}_U(g_U^{\Gamma})$ (patrz uwaga 4.2.1) i dostarcza użytkownikowi U. Otrzymana przez U wartość g_U^{Γ} stanowi anonimowy identyfikator użytkownika na czas przeprowadzania egzaminu.

4. Jeśli nadejdzie czas t, wtedy ujawniane są dane pozwalające przypisać g_U do konkretnego, anonimowego identyfikatora g_U^{Γ}:

   (a) dla $i = 1$ do n:

      i. M_i ujawnia (γ_i, g_U).

W późniejszej wersji protokołu [35] sugerowane jest wykorzystanie schematu podziału sekretu opartego na wielomianie Lagrange'a (patrz podrozdział 1.2.4). Dzięki temu możliwe jest odtworzenie w kroku 4 anonimowego identyfikatora g_U^{Γ} przez pewną podgrupę serwerów $\mathcal{M}$.

Opis protokołu

Oznaczenia:

- $\mathcal{T}, \mathcal{Q}$ – temat i pytania egzaminu;
- $\mathcal{A}_S, \mathcal{G}_S$ – odpowiedzi na pytania i ocena zdającego egzamin S;
- $A \overset{mix}{\Longrightarrow} B : m_1$ – A wysyła do B wiadomość m_1 poprzez anonimowy kanał zwrotny;
- $A \overset{mix}{\Longleftarrow} B : m_2$ – B odpowiada wiadomością m_2 poprzez anonimowy kanał zwrotny.

Przygotowanie egzaminu

1. EA generuje wszystkie parametry kryptosystemu ElGamala: p, q, g.
2. Użytkownicy U wybierają parę kluczy $(\widehat{pk}_U, \widehat{sk}_U)$. Klucze mogą zostać wykorzystane w wielu egzaminach. Zakłada się istnienie ogólnodostępnej bazy danych z publicznymi kluczami studentów $(g_S, \widehat{pk}_S)$ oraz informacjami na temat egzaminów, które może zdawać lub sprawdzać dany użytkownik.
3. Przed każdym egzaminem EA wybiera odpowiednie $\overline{s}$ i $\overline{g}$, a następnie publikuje $(\overline{g}, \overline{h})$, gdzie $\overline{h} = \overline{g}^{\overline{s}} \mod p$.

Rejestracja

1. EA sprawdza uczestnika U w bazie danych, oblicza $\tilde{r} = g_U^{\overline{s}} \mod p$ i zachowuje $\tilde{r}$.
2. Jeśli U jest studentem:
   (a) dla każdego $i = 1, \ldots, n$:
   $$EA \longrightarrow M_i : \tilde{r}, g_S,$$
   (b) $\mathcal{M}$: oblicza (patrz usługa opóźnionego ujawniania) $r_1 = g_S^{\Gamma} \mod p$ i $r_2 = \tilde{r}^{\Gamma} \mod p$, gdzie $\Gamma = \sum_{i=1}^{n} \gamma_i$; każdy serwer przechowuje (t, γ_i, g_S), gdzie t oznacza czas ujawnienia γ_i,
   (c) $\mathcal{M} \longrightarrow U : (r_1, r_2)$.
3. Jeśli U jest sprawdzającym:
   (a) $EA \longrightarrow T : (\tilde{r}, g_T)$,
   (b) T oblicza: $r_1 = g_T^{\alpha} \mod p$, oraz $r_2 = \tilde{r}^{\alpha} \mod p$, gdzie α jest wartością losowo wybraną przez T.
4. U na podstawie dowodu o wiedzy zerowej równości logarytmów dyskretnych par (r_1, r_2) i $(\overline{g}, \overline{h})$ otrzymanego od EA stwierdza, iż uzyskał autoryzację.
5. U wylicza $p_1 = r_1^{\widehat{sk}_U} \mod p$.

W wyniku rejestracji U posiada (r_1, p_1, r_2), który służy jako jego pseudonim. Pseudonimy studentów będą zapisywane (a_1, b_1, a_2), natomiast nauczycieli (e_1, f_1, e_2). W przypadku nauczycieli nie jest wymagane odkrywanie tożsamości poprzez usługę opóźnionego ujawniania.

Egzamin

1. EA sprawdza tożsamość uczestników oraz to czy mają prawo podchodzić do danego egzaminu.

2. $S \overset{mix}{\Longrightarrow} EA : (a_1, b_1, a_2), \mathcal{T}$

3. $T \overset{mix}{\Longrightarrow} EA : (e_1, f_1, e_2), \mathcal{T}$

4. EA sprawdza tożsamości nadawców otrzymanych wiadomości oraz ich upoważnienia dotyczące $\mathcal{T}$. Weryfikacja wymaga sprawdzenia kongruencji $b_1^{\overline{s}} \equiv a_2 \pmod p$ lub $f_1^{\overline{s}} \equiv f_2 \pmod p$ oraz sprawdzenia, czy student nie przesłał już odpowiedzi. EA zapisuje w bezpieczny sposób:

$$\mathcal{T}; (a_1, b_1, a_2), \mathcal{E}_{\mathcal{M}}(id_S, \widehat{pk}_S); (e_1, f_1, e_2), \mathcal{E}_{\mathcal{M}}(id_T, \widehat{pk}_T).$$

5. EA przesyła studentowi S pytania $\mathcal{Q}$ wraz z podpisem i znacznikiem czasu t_1:

$$S \overset{mix}{\Longleftarrow} EA : \mathcal{Q}, t_1, \mathcal{P}_{EA}(\mathcal{Q}).$$

6. S rozwiązuje egzamin i przesyła zaszyfrowane odpowiedzi, część (a_1, b_1) identyfikatora anonimowego (a_1, b_1, a_2) oraz znacznik czasu t_2:

$$S \overset{mix}{\Longrightarrow} EA : (a_1, b_1), \mathcal{E}_{\mathcal{M}}(\mathcal{A}_S), t_2.$$

7. EA bezpiecznie przechowuje $(\mathcal{Q}, t_1, t_2, \mathcal{E}_{\mathcal{M}}(\mathcal{A}_S))$ oraz odsyła wartość funkcji haszującej jako potwierdzenie dla S:

$$S \overset{mix}{\Longleftarrow} EA : h(a_1, b_1, a_2, \mathcal{T}, \mathcal{Q}, t_1, t_2, \mathcal{E}_{\mathcal{M}}(\mathcal{A}_S)).$$

8. EA wybiera nauczyciela T do sprawdzania danego egzaminu. Następnie wybiera identyfikator zestawu odpowiedzi $id_{\mathcal{A}}$, który jest wykorzystana do przesłania zapytania przez sieć mieszającą składającego się z poniższych cebul:

$$\mathcal{E}_{\mathcal{M}}(id_{\mathcal{A}}, \widehat{pk}_{EA}), \mathcal{E}_M(\mathcal{A}_S), \mathcal{E}_{\mathcal{M}}(id_T, \widehat{pk}_T).$$

9. T ocenia otrzymane odpowiedzi $\mathcal{A}_S$ i odpowiada anonimowo:

$$\mathcal{E}_{\mathcal{M}}(id_T, \widehat{pk}_T), \mathcal{E}_{\mathcal{M}}(\mathcal{G}_S), \mathcal{E}_{\mathcal{M}}(id_{\mathcal{A}}, \widehat{pk}_{EA}).$$

Publikowanie ocen
Poznanie ocen poszczególnych studentów wymaga od EA odkrycia identyfikatora każdego studenta na podstawie identyfikatorów anonimowych powstałych na etapie rejestracji (a_1, b_1, a_2).

1. $EA \longrightarrow \mathcal{M} : a_2$

2. Jeśli upłynął czas t, wyznaczana jest dla S przez serwery $\mathcal{M}$ (patrz usługa opóźnionego ujawniania) opowiadająca wartość $\tilde{r}$, tzn. taka dla której $\tilde{r}^{\Gamma} = a_2$. Wartość ta $\tilde{r} = g_S^{\overline{s}}$ zostaje wysłana w zaszyfrowanej formie (kluczem $\widehat{pk}_{EA}$) do EA:

$$EA \longleftarrow \mathcal{M} : \mathcal{E}_{EA}(\tilde{r}).$$

3. EA wyznacza prawdziwą tożsamość studenta (identyfikator g_S) na podstawie $\tilde{r}$ i znajomości $\overline{s}$ ($\tilde{r} = g_S^{\overline{s}} \mod p$). Następnie zapisuje w bazie danych ocenę, $\mathcal{E}_{\mathcal{M}}(id_T, \widehat{pk}_T)$, wraz z podpisem i identyfikatorem odpowiedniego studenta.

4.2.4 Porównanie cech opisanych protokołów

Poniższe zestawienie cech pokazuje pewną zależność pomiędzy ilością spełnionych wymogów bezpieczeństwa a poziomem skomplikowania protokołu. Rozwiązania, które okazały się spełniać najwięcej wymogów bezpieczeństwa wymagały zaangażowania wielu uczestników po stronie serwerowej, wykorzystania zaawansowanych i złożonych narzędzi kryptograficznych, a w konsekwencji zwiększenia ilości informacji i liczby komunikatów wymienianych przez sieć komputerową.

	Anonimowość	Rozproszenie zaufania	Zaufane środowisko (kontrola)	Potwierdzenie
CHP	Nie	Nie	Nie	Nie
CHD	Tak	Nie	Nie	Tak
HP	Tak	Tak	Nie	Tak
SeCOnE	Tak	Nie	Tak	Tak

Tabela 4.1: Zestawienie cech protokołów egzaminowania elektronicznego

4.3 Protokoły oparte na dwuetapowym mieszaniu

Poniższy podrozdział zawiera propozycję protokołu opartego o sieci mieszające częściowo deszyfrujące (oparte na algorytmie ElGamala – podrozdział 2.1). Zastosowanie tej procedury weryfikującej pozwala osiągnąć własności podobne do protokołu HP (4.2.3). W szczególności, protokół zapewnia anonimowość studentów i nauczycieli jednocześnie rozpraszając zaufanie pomiędzy uczestników realizujących przetwarzanie danych egzaminu. Warto również podkreślić, że własności te zostają osiągnięte przy znacznie mniejszym stopniu skomplikowania protokołu.

Poniżej zaprezentowane zostaną dwa warianty protokołu różniące się kierunkiem przepływu wiadomości podczas dwóch etapów ich przesyłania. W wariancie jednokierunkowym wiadomości będą przetwarzane przez sieć mieszającą dwa razy w tym samym kierunku, podczas gdy w wariancie dwukierunkowym w drugim etapie wiadomości będą przechodziły przez serwery mieszające w przeciwną stronę niż w etapie pierwszym.

Uczestnicy
Oba warianty protokołu oparte są na tym samym zestawie ról uczestników.

- $S_1, S_2, \ldots, S_n$ – studenci.
- $T_1, T_2, \ldots, T_m$ – nauczyciele, sprawdzający egzamin.
- EA – organizator egzaminu.
- $M_1, M_2, \ldots, M_\lambda$ – serwery mieszające tworzące sieć mieszającą $\mathcal{M}$.
- BB – tablica ogłoszeń wykorzystywana podczas przetwarzania wiadomości przez serwery mieszające. Umożliwia również publikowanie uwierzytelnionych wiadomości przez pozostałych uczestników protokołu.

4.3.1 Wariant jednokierunkowy

Opisany poniżej protokół posiada dwa podwarianty różniące się źródłem pochodzenia jednorazowego identyfikatora id_i studenta S_i. Identyfikator ten może być wybrany przez samego studenta lub nadany studentowi przez organizatora egzaminu. W dalszej części podrozdziału opisane zostaną cechy różniące podwarianty.

Dostarczenie pytań i zebranie odpowiedzi.

1. Studenci otrzymują pytania kanałem prywatnym. Zakładamy, że organizujący egzamin wie jaki zestaw pytań otrzymał dany student. Razem z pytaniami egzaminacyjnymi student S_i może również otrzymać jednorazowy identyfikator id_i.

2. Student S_i odszyfrowuje pytania, a następnie w określonym przedziale czasu formułuje na nie odpowiedzi $\mathcal{A}_i$ i szyfruje je kluczem publicznym sieci mieszającej $\mathcal{M}$. S_i szyfruje także swój identyfikator id_i, po czym publikuje je na BB w postaci cebul.

   $$S_i \longrightarrow BB : \mathcal{E}_{\mathcal{M}}(id_i), \mathcal{E}_{\mathcal{M}}(\mathcal{A}_i)$$

   Student wybiera identyfikator id_i, jeżeli nie został on mu narzucony przez EA. Zgodnie z opisem sieci mieszających opartych o algorytm ElGamala zaszyfrowanie kluczem sieci, będącym iloczynem kluczy poszczególnych serwerów mieszających, jest równoważne ze stworzeniem cebuli, która może być przez tę sieć przetwarzana.

Przetwarzanie odpowiedzi i ocena egzaminu

1. Sieć mieszająca $\mathcal{M}$ przetwarza cebule zawierające odpowiedzi studentów. Cebule są częściowo deszyfrowane, przy czym, w ostatnim kroku M_λ dodatkowo szyfruje je kluczem odpowiedniego nauczyciela. Możemy przyjąć konwencję, iż pierwsze $\frac{n}{m}$ pozycji wyjściowych sieci należy do nauczyciela T_1, kolejne $\frac{n}{m}$ pozycji do T_2, itd.

2. Sprawdzający T_j pobiera zestawy odpowiedzi, deszyfruje je, sprawdza, ocenia a następnie publikuje oceny na odpowiednich pozycjach na BB. Możemy przyjąć, iż sprawdzający mają przypisane kolejne grupy pozycji wyjściowych sieci (patrz rys. 4.1).

3. Po opublikowaniu wszystkich ocen na BB sieć mieszająca $\mathcal{M}$ przetwarza cebule zawierające identyfikatory. Cebule są częściowo deszyfrowane i przechodzą te same ścieżki co poprzednio cebule z odpowiedziami. W rezultacie na tablicy ogłoszeń BB obok zaszyfrowanych odpowiedzi i ocen pojawiają się identyfikatory.

4. Jeżeli identyfikator został wybrany przez studenta, to należy go ujawnić organizującemu egzamin przesyłając dowód, iż szyfrogram na wejściu do sieci $\mathcal{M}$ rzeczywiście zawierał ów identyfikator. Wymóg ten można spełnić stosując np. nieinteraktywne dowody z wiedzą zerową (patrz 1.2.3).

   $$S_i \longrightarrow EA : \mathcal{E}_{EA}(S_i, id_i, proof(id_i))$$

W zależności od źródła pochodzenia identyfikatora protokół może mieć różne własności. Jeżeli id_i pochodzi od EA, to nie jest potrzebny czwarty krok protokołu (przesłanie dowodu). Należy zaznaczyć, iż w tym przypadku EA poznaje na końcu permutację Π sieci $\mathcal{M}$. Dodatkowo, EA nie musi wcale przekazywać studentowi id_i w formie jawnej, może przekazać identyfikator zaszyty w cebuli.

W przypadku gdy identyfikator id_i jest wybierany przez S_i, konieczny jest czwarty krok protokołu, czyli dostarczenie dowodu autentyczności identyfikatora. Problem dowodu możemy dość prosto rozwiązać narzucając odpowiedni mechanizm tworzenia identyfikatorów. Identyfikator może być podpisem (lub fragmentem podpisu) studenta pod tematem egzaminu oraz pewną wartością losową k_i: $id_i = \mathcal{P}_{S_i}(\mathcal{T}, k_i)$. W takim przypadku jako dowód autentyczności id_i wystarczy przesłać k_i ($proof(id_i) = k_i$). Oczywiście można również wykorzystać w tym celu mechanizm dowodów z wiedzą zerową (patrz 1.2.3), przy czym będą one nieco bardziej skomplikowane w zastosowaniu.

4.3.2 Wariant dwukierunkowy

Opisany poniżej wariant dwukierunkowy posiada kilka podwariantów różniących się sposobem publikowania ocen przez sprawdzającego. W zależności od wybranej opcji można zapewnić inny poziom tajności wystawianych ocen. Ten aspekt wariantu dwukierunkowego e-egzaminów jest opisany bardziej szczegółowo w dalszej części podrozdziału.

Dostarczenie pytań i zebranie odpowiedzi

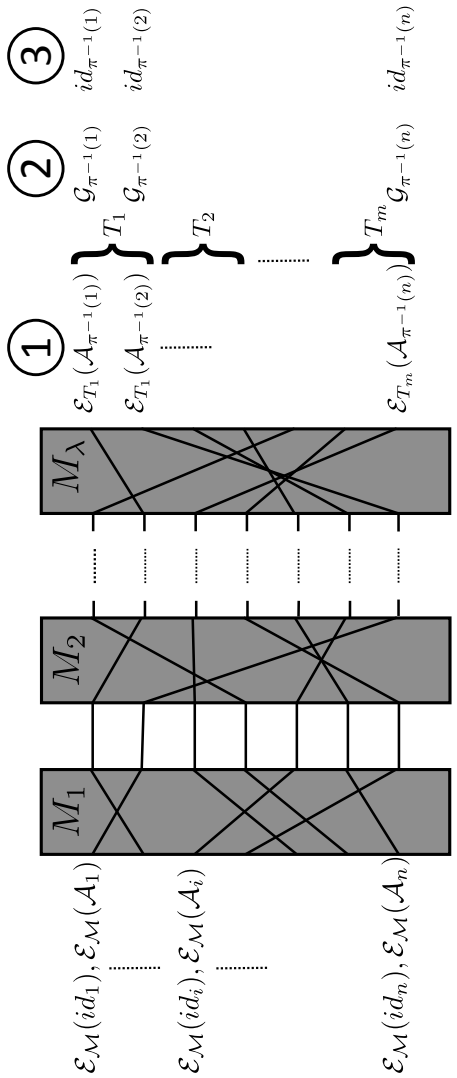
1. Podobnie jak w wariancie jednokierunkowym protokołu studenci otrzymują pytania kanałem prywatnym. Dodatkowo, na potrzebę danego egzaminu student S_i wybiera parę kluczy asymetrycznych kryptosystemu ElGamala – $(\widehat{pk}_i, \widehat{sk}_i)$

2. Student S_i udziela odpowiedzi $\mathcal{A}_i$ na zadane mu pytania, a następnie publikuje je w postaci zaszyfrowanej cebuli na BB.

$$S_i \longrightarrow BB : \mathcal{E}_{\mathcal{M}}(\mathcal{A}_i, \widehat{pk}_i)$$

Przetwarzanie odpowiedzi i ocena egzaminu

1. Cebule zawierające odpowiedzi są przetwarzane przez kolejne serwery mieszające. Podobnie jak w jednokierunkowej wersji protokołu ostatni serwer szyfruje odpowiedzi kluczem odpowiedniego nauczyciela.



Rysunek 4.1: Kolejne etapy wariantu jednokierunkowego: 1) przetwarzanie odpowiedzi, 2) publikowanie ocen, 3) przesyłanie identyfikatorów przez sieć mieszającą

2. Nauczyciele sprawdzają odpowiedzi, a następnie publikują oceny na odpowiednich pozycjach na BB. w od w od

3. Oceny w postaci zaszyfrowanej cebuli ($\mathcal{E}_{\mathcal{M}}(\mathcal{E}_{\widehat{pk}_i}(\mathcal{G}_i))$) są przetwarzane w odwrotnej kolejności przez serwery mieszające $(M_\lambda, M_{\lambda-1}, \dots)$. Pokonują te same ścieżki co cebule z odpowiedziami, lecz w odwrotnym kierunku. Tym razem ostatni serwer przetwarzający szyfruje ocenę dodatkowo kluczem odpowiedniego studenta i/lub kluczem organizatora egzaminów. W podwariancie protokołu dwukierunkowego publikowana i przetwarzana jest również cebula przeznaczona dla EA – $\mathcal{E}_{\mathcal{M}}(\mathcal{E}_{EA}(\mathcal{G}_i))$.

Niewątpliwą zaletą zaprezentowanego rozwiązania jest brak jakichkolwiek identyfikatorów. Poza tym, w zależności od wymogów danego egzaminu można zapewnić, iż zachodzi jedna z poniższych sytuacji.

- Tylko student poznaje otrzymaną ocenę. W tym przypadku nie są przesyłane żadne opcjonalne cebule.

- Tylko organizator egzaminu poznaje oceny studentów. Sytuacja tego typu zachodzi, jeżeli na etapie przesyłania odpowiedzi zamiast cebul zawierających $\mathcal{E}_{\widehat{pk}_i}(\mathcal{G}_i)$ przesyłane są te zawierające $\mathcal{E}_{EA}(\mathcal{G}_i)$.

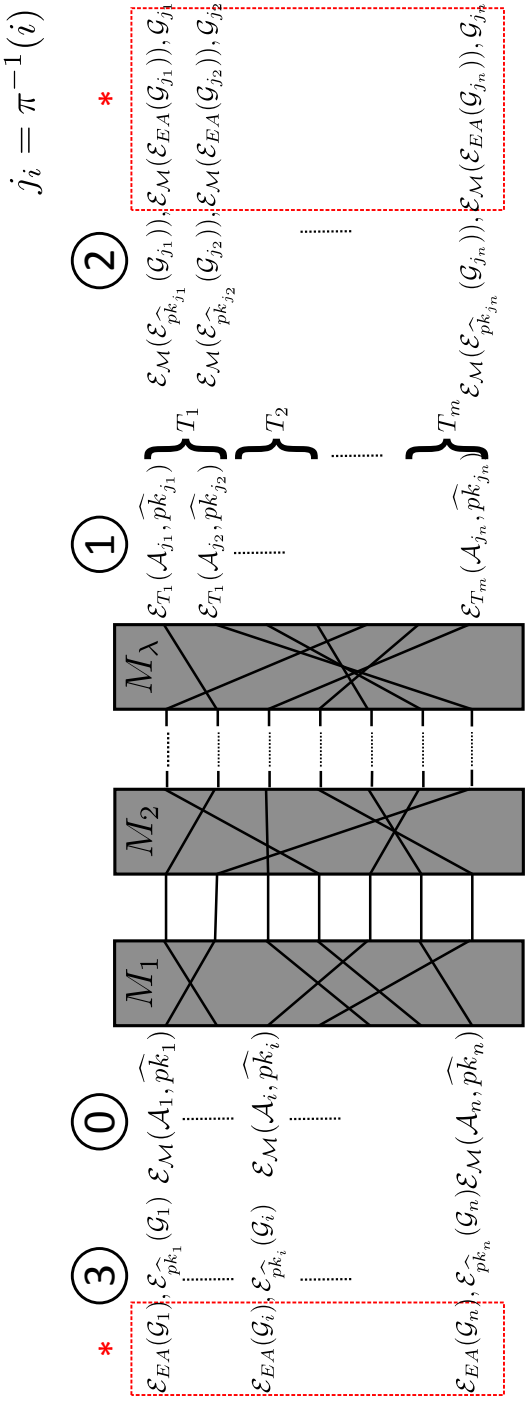
- Zarówno organizator jak i student poznają ocenę z egzaminu. Zapewnienie tej własności wymaga przesłania obu cebul wspomnianych w poprzednim punkcie.

- Powszechnie znane są anonimowe oceny wszystkich studentów, co pozwala policzyć własności statystyczne ocen (np. średnią lub medianę ocen). W tym celu nie jest wymagany przesyłanie ocen przez sieć mieszającą. Mogą one zostać opublikowane w postaci jawnej.

4.3.3 Weryfikacja i realizacja reklamacji

Weryfikacja ze strony studenta sprowadza się do sprawdzenia, czy jego odpowiedzi zostały wysłane i opublikowane na tablicy ogłoszeń w postaci zaszyfrowanej. W tym celu, urządzenie wykorzystywane przez studenta do wprowadzenia i wysłania odpowiedzi może wyświetlić szyfrogram odpowiedzi, który następnie można porównać z opublikowanym. Weryfikacja procesu przetwarzania odpowiedzi i ocen (dwukrotnego użycia sieci mieszającej) polega na przeprowadzeniu procedury 2-RPC opisanej w 2.5.

W przypadku, gdy student uważa, że jego test został nieprawidłowo oceniony istnieje możliwość prześledzenia procesu przetwarzania odpowiedzi.



Rysunek 4.2: Kolejne etapy wariantu dwukierunkowego: 0) przesłanie odpowiedzi przez studenta, 1) przetwarzanie odpowiedzi, 2) publikowanie ocen, 3) przesyłanie ocen przez sieć mieszającą; * oznacza opcjonalne dane przesyłane w podwariancie protokołu

W tej sytuacji wymagane jest ujawnienie przez studenta udzielonych odpowiedzi organizatorowi egzaminu. W obu wariantach student publikuje zaszyfrowane odpowiedzi na tablicy ogłoszeń BB. Autentyczność tych danych jest potwierdzona podpisem BB (tablica ogłoszeń realizuje uwierzytelniony kanał publiczny), dzięki czemu pozwalają one przeprowadzić procedurę reklamacyjną w następujących krokach.

1. S ujawnia odpowiedzi $\mathcal{A}'$, oraz dowód tego, iż opublikowany podczas egzaminu szyfrogram zawierał te odpowiedzi. Dowód może być dowodem z wiedzą zerową (patrz 1.2.3), lub jeżeli wykorzystywany jest schemat ElGamala, jako dowód może posłużyć pseudolosowa wartość k wykorzystywana podczas szyfrowania (patrz 1.2.2).

2. Jeżeli zaprezentowane przez studenta dane się zgadzają, wtedy jest powoływana specjalna komisja kwalifikacyjna, której zadaniem jest ponowne ocenienie egzaminu. Komisję mogą stanowić losowo wybrani nauczyciele, którzy są proszeni o niezależną ocenę pracy.

3. Jeśli ocena komisji odbiega od zaproponowanej w pierwszej kolejności przez nauczyciela, to istnieje możliwość odtworzenia ścieżki jaką przebyły odpowiedzi przez sieć mieszającą. W rezultacie możliwe jest sprawdzenie, czy odpowiedzi były prawidłowo przetwarzane, oraz czy jakiś z serwerów mieszających nie popełnił błędu lub nie dopuścił się nadużycia. Po ujawnieniu całej ścieżki możliwe jest wskazanie nauczyciela odpowiedzialnego za sprawdzenie egzaminu.

4. W przypadku, gdy istnieje taka potrzeba można nakazać nauczycielowi wykazanie poprawności procesu oceniania formularza z odpowiedziami.

4.3.4 Analiza bezpieczeństwa

Anonimowość i tajność. Anonimowość przesyłanych wiadomości jest zapewniona dzięki wykorzystaniu sieci mieszających oraz procedury weryfikacyjnej, która pozwala zachować anonimowość mimo ujawnienia połowy przejść permutacji poszczególnych serwerów mieszających. Należy zwrócić uwagę na fakt, iż dzięki dwukrotnemu wykorzystaniu sieci mieszającej oraz procedury dwuetapowego RPC (patrz 2.5) możliwe jest przesłanie anonimowo przez studenta formularza z odpowiedziami, oraz odesłania, również w sposób anonimowy, oceny egzaminu przez nauczyciela. Poufność przesyłanych danych jest zapewniona dzięki zastosowaniu szyfrowania ElGamala.

Rozproszenie zaufania i niezawodność. Protokół mieszania wiadomości przez wiele zaufanych serwerów w rozproszony sposób powoduje, iż organizator egzaminu nie jest w stanie ustalić czyje egzaminy zostały przydzielone do danego nauczyciela. Jednakże, w wariancie jednokierunkowym protokołu istnieje możliwość przekłamania wyników egzaminu, przy założeniu współpracy pomiędzy organizatorem a nauczycielem i pod warunkiem, iż organizator tworzy identyfikatory studentów. Możliwe jest w tym przypadku manipulowanie przez nieuczciwego sprawdzającego ocenami studentów, których prace zostały mu przydzielone. Tożsamość zdających egzamin może poznać, dzięki zmowie z nieuczciwym organizatorem. Jeżeli współpracujący uczestnicy działaliby na niekorzyść ocenianych studentów, to zawsze w ich obronie może zostać wszczęta procedura odwoławcza. Jednakże o wiele trudniejsze może być wykrycie takiej zmowy, która ma na celu nieuczciwe podniesienie oceny pewnych studentów przez sprawdzających egzamin. W tym przypadku wszczynanie procedury odwoławczej nie jest w interesie studenta.

Należy zwrócić uwagę na fakt, iż powyższe zarzuty dotyczą tylko wariantu jednokierunkowego protokołu, oraz pewnej podgrupy studentów, którzy zostali przypisani nieuczciwemu nauczycielowi. W drugim wariancie o wiele trudniej wyobrazić sobie współdziałanie podgrupy uczestników protokołu w celu zmanipulowania wyników. W oczywisty sposób, możliwe jest przekazywanie informacji o tożsamości studenta w odpowiednio spreparowanych odpowiedziach. Proceder ten może zostać wykryty, jeśli odpowiedzi nie trafią do odpowiedniego nauczyciela. Tego typu praktyki, podobnie jak przekazywanie przez organizatora prawidłowych odpowiedzi wybranym studentom przed egzaminem, stanowią nieodłączny problem wszelkich systemów egzaminacyjnych.

W wyjątkowej sytuacji może dojść do awarii jednego z serwerów mieszających. W celu zapewnienia niezawodności możemy przyjąć, iż przed rozpoczęciem egzaminu serwery przy pomocy schematu podziału sekretu dzielą się z pozostałymi wartościami pozwalającymi wygenerować permutacje (patrz szyfrowanie progowe 1.2.7). Dzięki temu możliwe jest zastąpienie dowolnego serwera przez pozostałe, jeżeli zajdzie taka potrzeba.

Podsumowanie

W ostatnich latach pojawiło się wiele ciekawych propozycji protokołów kryptograficznych odpowiadających na zapotrzebowania e-społeczeństwa. Jako przykłady mogą posłużyć protokoły wyborów elektronicznych (w szczególności wyborów przez Internet), czy też systemy elektronicznego egzaminowania. Proponowane dotychczas rozwiązania pozostawiały wiele do życzenia ze względu na bezpieczeństwo obliczeń. Celem tej rozprawy, oprócz przeglądu istniejących protokołów, była propozycja własnych, oryginalnych algorytmów i protokołów zapewniających wysoki poziom bezpieczeństwa obliczeń.

Jedną z zaproponowanych technik jest zmodyfikowana procedura RPC pozwalająca na podwójne wykorzystanie tej samej sieci mieszającej (patrz sekcja 2.5). Znalazła ona zastosowanie w protokole elektronicznych wyborów opartych o papierowe karty do głosowania (opisane w 3.3 oraz pracy [51]). Zastosowanie kart ze specjalnymi kodami pozwala wyborcy na oddanie głosu zdalnie poprzez jego komputer osobisty (który może być potencjalnie złośliwie zmodyfikowany), przy zachowaniu pełnego bezpieczeństwa głosu (integralności i poufności). Podwójne wykorzystanie sieci mieszającej okazało się również przydatne w systemach e-egzaminów. W podrozdziale 4.3 opisany został protokół pozwalający na zapewnienie jednoczesnej anonimowości egzaminowanych i egzaminujących. Dzięki dwukrotnym zastosowaniu sieci mieszającej jest on znacznie mniej skomplikowany niż inne protokoły oferujące ten sam poziom anonimowości i zaufania do organizatorów egzaminu. Analiza wpływu stosowania procedur częściowego sprawdzania na anonimowość sieci mieszającej przedstawiona jest w podrozdziale 2.4.

Kolejne oryginalne rozwiązanie, związane z problemem zdalnych wyborów (patrz podrozdział 3.3 oraz praca [51]), wymaga zastosowania prostych urządzeń w postaci kart chipowych z ekranem. Dzięki temu krytyczne operacje takie jak szyfrowanie głosu, czy też wybieranie anonimowego identyfikatora mogą być przeprowadzone w zaufanym środowisku urządzenia. Jako, że zastosowanie zaufanych urządzeń pociąga za sobą zagrożenie kleptografii (patrz 1.2.8), w pracy znalazła się propozycja protokołu umożliwiającego wygenerowanie wartości losowej przez dwie strony w taki sposób, iż jedna

ma gwarancję losowości tej wartości chociaż jej nie poznaje, natomiast druga strona może dysponować ową wartością. Ów protokół negocjacyjny, jako szczególny przykład protokołu interakcyjego generowania, może znaleźć zastosowanie w wielu innych sytuacjach [50].

Protokół wyborczy oparty na papierowych kartach do głosowania (opisany w podrozdziale 3.4) oprócz podwójnego wykorzystania sieci mieszających wprowadza też pewną modyfikację ich działania. Poza permutowaniem zaszyfrowanych wiadomości sieć wykonuje obliczanie sum w arytmetyce modularnej na zaszyfrowanych danych. W rezultacie powstaje nowy typ sieci mieszająco-obliczeniowej. Zaszyfrowane obliczenia w grupie cyklicznej są wykorzystywane również w nowatorskim protokole dla głosowań w małych grupach wyborczych. Ponieważ dotychczas nie została stworzona specyfikacja tego typu podgrupy systemów wyborczych, to opisowi protokołu w podrozdziale 3.5 towarzyszy propozycja wymogów stawianych tego typu systemom.

Dodatek A

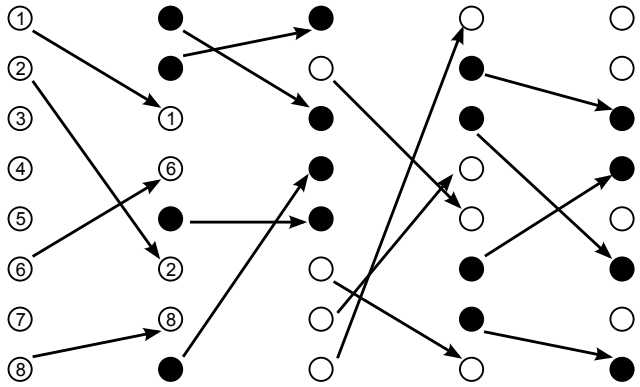
Analiza rozkładów brzegowych i łącznych w procedurze częściowego sprawdzania

Poniższy dodatek zawiera rozważania dotyczące słabej anonimowości oraz ewolucji rozkładów brzegowych w zależności od liczb wiadomości i serwerów sieci mieszającej. W dalszej części dodatku opisana została próba znalezienia związku pomiędzy słabą i silną anonimowością, tzn. pomiędzy rozkładami brzegowymi i łącznymi wektorów binarnych będących konsekwencją procedury częściowego sprawdzania, przy założeniu prawdziwości hipotezy o modelowaniu tych rozkładów łącznych za pomocą zmiennych losowych niezależnych.

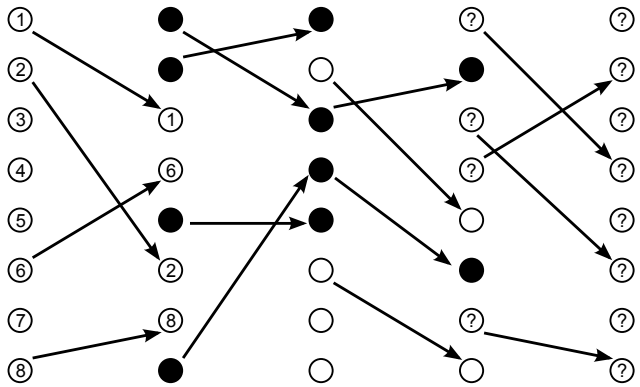
A.1 Rozkłady brzegowe i słaba anonimowość

Poniżej zaprezentowane zostały dwa skrajne przypadki mogące się pojawić przy procedurze częściowego sprawdzania związane z liczbą ścieżek kontynuowanych przez drugą parę serwerów. W ogólnym przypadku prawdopodobieństwo, że wskazane przejścia nie będą równomiernie rozłożone pomiędzy pozycjami z większym prawdopodobieństwem wystąpienia białej wiadomości a tymi pozycjami z mniejszym prawdopodobieństwem wystąpienia białej, jest stosunkowo małe. Zatem, zanikanie wiedzy o rozmieszczeniu białych i czarnych cebul jest dosyć szybkie i zależy od liczby wiadomości oraz liczby serwerów. Zauważmy jednak, iż w drugim przypadku (rysunek A.2) nadal pewne konfiguracje białych i czarnych nie są możliwe. Na przykład nie jest możliwe, aby wszystkie pozycje wyjściowe, do których prowadzi ujawnione przejście przez serwer czwarty (w przykładzie są to pozycje o numerach pa-

rzystych) były obsadzone przez wiadomości tylko z grupy białych lub tylko z grupy czarnych. Analogicznie nie jest możliwe, aby nieparzyste pozycje wyjściowe przypisane były wiadomościom tylko jednej grupy. Co więcej, dokładnie wiadomo ile może być białych i czarnych na pozycjach parzystych i nieparzystych.



Rysunek A.1: Podział na czarne i białe – informacja obserwatora pozostaje niezmieniona



Rysunek A.2: Podział na czarne i białe – informacja obserwatora zostaje prawie całkowicie zatarta

Opisany powyżej przykład pokazuje problem występowania ograniczeń ilościowych dla białych i czarnych na pozycjach wyjściowych kolejnych bloków serwerów. Parametr charakteryzujący te ograniczenia w i-tym kroku (po i blokach serwerów) oznaczymy δ_i. Jego ewolucja będzie opisywała proces zanikania ograniczeń ilościowych (δ_i przyjmuje wartości od n do 0). Wartość

n oznacza, iż jest możliwa tylko jedna konfiguracja, tzn taka gdzie w danej grupie pozycji wyjściowych możliwe są tylko białe lub tylko czarne (przykład opisywany przez rysunek A.1). Natomiast δ_i równe 0 oznacza, iż ograniczenia liczbowe nie występują. Bardziej formalny opis parametru δ_i oraz rozważania nad jego ewolucją zawiera punkt A.2.

Przy pomocy innego parametru ε_i będziemy opisywać wspomniany wcześniej proces przemieszania lub inaczej stapiania białych i czarnych w i-tym kroku. Możemy ten parametr interpretować jako odchylenie od rozkładu jednostajnego (również w sensie normy całkowitego wahania) rozkładów brzegowych, które przyjmują wartości $\{\frac{1}{2}+\varepsilon_i, \frac{1}{2}-\varepsilon_i\}$. W przykładach przytoczonych na rysunkach wartości ε wynoszą: $\frac{1}{2}$ (rysunek A.1) oraz 0 (rysunek A.2).

Z punktu widzenia analizy silnej anonimowości najważniejszy jest rozkład łączny, którego rozkłady brzegowe zależą od parametru ε_i. Należy zauważyć, iż wnioskowanie na temat rozkładu łącznego na podstawie ε_i ma sens jedynie, gdy nie ma ograniczeń ilościowych, tzn. $\delta_i = 0$. Poniżej zawarte zostały dwa przykłady ewolucji δ_i i ε_i w zależności od zmiennej losowej $\widetilde{\gamma}_i$, która wyraża proporcje w jakich zostaną przemieszane wiadomości obu grup.

$\widetilde{\gamma}_i$		$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$
i	0	1	2	3	4
ε_i	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$
δ_i	n	$\frac{3}{4}n$	$\frac{1}{2}n$	$\frac{1}{4}n$	0

a)

$\widetilde{\gamma}_i$		$\frac{1}{2}$	$\frac{1}{4}$
i	0	1	2
ε_i	$\frac{1}{2}$	0	0
δ_i	n	$\frac{1}{2}n$	$\frac{1}{4}n$

b)

Tabela A.1: Przykładowe ewolucje procesów δ i ε, w których jeden z parametrów osiąga 0 w zależności od zmiennej losowej $\widetilde{\gamma}_i$

Ewolucję rozkładów brzegowych, wyznaczanych przez parametr ε opisuje zależność

$$\Pr\{X_j^{(i+1)} = 1\} = \begin{cases} \frac{1}{n} \sum\limits_{k \in H_1} \Pr\{X_k^{(i)} = 1\} & j \in \Upsilon_{i+1}(H_1) \\ \frac{1}{n} \sum\limits_{k \in H_2} \Pr\{X_k^{(i)} = 1\} & j \in \Upsilon_{i+1}(H_2) \end{cases}.$$

Powyższy zapis oddaje charakter ewolucji rozkładów brzegowych jako proces uśredniania wartości w pewnych proporcjach. Proporcje te zależą od rozmiaru przekroju $H_1 \cap \Upsilon_i(H_1)$, i są dokładnie odwrotne dla obu grup pozycji wyjściowych (należących do obrazu permutacji poprzedniego bloku na pierwszej lub drugiej połowie pozycji wejściowych). Rozmiar tego przekroju jest

zmienną losową Z_i o rozkładzie hipergeometrycznym $\mathcal{H}(2n, n, n)$. Dodatkowo zdefiniujemy pomocnicze zmienne losowe:

$$\gamma_i = \frac{Z_i}{n}, \quad \widetilde{Z}_i = \min\{Z_i, n - Z_i\}, \quad \widetilde{\gamma}_i = \frac{\widetilde{Z}_i}{n}.$$

Zależność opisująca ewolucję w i-tym kroku pozwala zdefiniować dwa zbiory pozycji wyjściowych i-tego bloku, które charakteryzują się większym prawdopodobieństwem wystąpienia 1 niż 0 (zbiór $A_1^{(i)}$) oraz jego dopełnienie, tzn. zbiór pozycji o większym prawdopodobieństwie wystąpienia 0 niż 1 (zbiór $A_0^{(i)}$).

$$\{A_0^{(i)}, A_1^{(i)}\} = \{\Upsilon_i(H_1), \Upsilon_i(H_2)\}$$

Wykorzystując podział pozycji wyjściowych na $A_0^{(i)}$ i $A_1^{(i)}$ możemy zdefiniować podział wektorów na klasy $C_k^{(i)}$

$$C_k^{(i)} = \Big\{X \in B_n : \sum_{j \in A_1^{(i)}} X_j = k\Big\}.$$

W większości przypadków istotna będzie liczba jedynek na pozycjach uprzywilejowanych (należących do $A_1^{(i)}$), a nie numer i kroku który jest analizowany. Z tego względu łatwiej będzie założyć, że $A_1^{(i)} = H_1$ i odwołać się do ogólnej klasyfikacji wektorów

$$C_k = \Big\{X \in B_n : \sum_{j=1}^{n} X_j = k\Big\}.$$

Zauważmy, że klasy $C_k^{(i)}$ możemy odwzorować na C_k przy pomocy bijekcji Υ_i^{-1}.

Znalezienie zależności pomiędzy słabą anonimowością a rozkładami brzegowymi (parametrem ε) wymaga określenia rozkładów $\Pr\{\Pi(j) = k\}$ dla pozycji wejściowych w_j. Nie trudno zauważyć, że dla połowy pozycji wyjściowych prawdopodobieństwo to wynosi $\frac{1}{n}\left(\frac{1}{2} + \varepsilon\right)$, podobnie $\frac{1}{n}\left(\frac{1}{2} - \varepsilon\right)$ dla pozostałych. Zatem każda pozycja wejściowa charakteryzuje się tą samą anonimowością, co oznacza, iż średnia entropia (słaba anonimowość) jest równa entropii dowolnej pozycji wejściowej.

$$S = S(w_j) = n\frac{1}{n}\left(\frac{1}{2} + \varepsilon\right)\log\left(\frac{1}{\frac{1}{n}\left(\frac{1}{2} + \varepsilon\right)}\right) + n\frac{1}{n}\left(\frac{1}{2} - \varepsilon\right)\log\left(\frac{1}{\frac{1}{n}\left(\frac{1}{2} - \varepsilon\right)}\right) =$$

$$= \left(\frac{1}{2} + \varepsilon\right)\log\left(\frac{n}{\left(\frac{1}{2} + \varepsilon\right)}\right) + \left(\frac{1}{2} - \varepsilon\right)\log\left(\frac{n}{\left(\frac{1}{2} - \varepsilon\right)}\right)$$

$$= \left(\frac{1}{2}+\varepsilon\right)\left(\log(n) - \log\left(\frac{1}{2}+\varepsilon\right)\right) + \left(\frac{1}{2}-\varepsilon\right)\left(\log(n) - \log\left(\frac{1}{2}-\varepsilon\right)\right)$$

$$= \log(n) - \left(\left(\frac{1}{2}+\varepsilon\right)\log\left(\frac{1}{2}+\varepsilon\right) + \left(\frac{1}{2}-\varepsilon\right)\log\left(\frac{1}{2}-\varepsilon\right)\right)$$

$$= \log(n) + \left(\frac{1}{2}+\varepsilon\right)\log\left(\frac{1}{\frac{1}{2}+\varepsilon}\right) + \left(\frac{1}{2}-\varepsilon\right)\log\left(\frac{1}{\frac{1}{2}-\varepsilon}\right)$$

Jak widać z powyższej zależności słaba anonimowość jest równa $log(n)$ zwiększonemu o entropię binarnej zmiennej losowej X_j

$$S = log(n) + H(X_j).$$

Słaba anonimowość S osiąga największą wartość przy maksymalnej entropii X_j ($\varepsilon = 0$) wynoszącej $log(2)$

$$S_{max} = log(n) + log(2) = log(2n) = log(N).$$

Wartość minimalna jest osiągana dla minimalnego $H(X_j) = 0$ ($\varepsilon = \frac{1}{2}$)

$$S_{min} = log(n).$$

Z powyższych zależności wynika fakt, iż słaba anonimowość w przypadku naszego procesu dostarcza nam znikomą wiedzę o wycieku informacji.

A.2 Zanikanie ograniczeń ilościowych

Oprócz procesu „rozmywania" się wiedzy obserwatora procesu (wyrażonej przez parametr ε_i) należy rozważyć towarzyszący mu proces zaniku ograniczeń dotyczących minimalnej i maksymalnej liczby jedynek w pierwszej połowie. Może się bowiem zdarzyć, że np. już w pierwszym kroku kule białe i czarne wymieszają się w równych proporcjach ($\widetilde{\gamma} = \frac{1}{2}$). W takim przypadku, pomimo tego, że wartość ϵ spadnie do 0 ($p = \frac{1}{2}$), to wektory w następnym kroku muszą mieć dokładnie $\frac{n}{2}$ jedynek w pierwszej połówce. Dalsza część tego podpunktu poświęcona jest rozważaniom o tym jak zmieniają się ograniczenia ilościowe w kolejnych krokach ujawniania przejść sieci mieszającej w procedurze RPC. Będziemy operować na uproszczonej klasyfikacji wektorów (C_k), czyli będziemy zakładać, że $A_1^{(i)} = H_1 = \{1, 2, \ldots, n\}$.

Oznaczenia:

- Q_i – zbiór wektorów w należących do B_n, które są możliwe na i-tym kroku procesu;
- dla wektorów binarnych $w = (w_1, w_2, \ldots, w_{2n})$ $(w \in B_n)$:
  - ▷ $\lambda(w) = \sum_{i=1}^{n} w_i$ – ilość jedynek na pierwszych n pozycjach,
  - ▷ $\lambda_0(w) = n - \sum_{i=1}^{n} w_i$ – ilość zer na pierwszych n pozycjach,

  zatem $\lambda(w) = k \Leftrightarrow w \in C_k$ oraz $\lambda_0(w) = k \Leftrightarrow w \in C_{n-k}$;
- dla $X \subseteq B_n$:
  - ▷ $\delta(X) = \max\{\min_{w\in X}\{\lambda(w)\}, \min_{w\in X}\{\lambda_0(w)\}\}$;
- $\delta_i = \delta(Q_i)$;
- $X \rightarrow_k Y$ oznacza, iż wektory zbioru Y można otrzymać z wektorów zbioru X poprzez permutację π taką, że $|\pi(H_1) \cap H_1| = k$, bardziej formalnie
$$\bigvee_{y\in Y} \bigexists_{\pi\in S_{2n}} \bigexists_{x\in X} |\pi[H_1] \cap H_1| = k \wedge \pi(x) = y.$$

Dalsza część rozważań skupi się na własnościach funkcji δ oraz relacji $\rightarrow_k$ w kontekście klas wektorów binarnych C_k. Własności te pozwolą udowodnić główne twierdzenia odnoszące się do tego, jaki charakter ma proces zmiany parametru δ_i oraz jak szybko δ osiąga wartość 0.

Uwaga A.2.1 *Aby wyliczyć δ można się skupić na zakresie ilościowym wyłącznie jedynek (lub wyłącznie zer)*

$$\begin{aligned}\delta(X) &= \max\{\min_{w\in X}\{\lambda(w)\}, n - \max_{w\in X}\{\lambda(w)\}\} \\ &= \max\{\min_{w\in X}\{\lambda_0(w)\}, n - \max_{w\in X}\{\lambda_0(w)\}\}.\end{aligned}$$

Niech X^c oznacza zbiór wektorów powstałych poprzez zamianę zer na jedynki i jedynek na zera w wektorach zbioru $X \subseteq B_n$. Zbiór ten będziemy nazywali lustrzanym w stosunku do X. Zauważmy, iż $\delta(X) = \delta(X^c)$. W szczególnym przypadku, ponieważ $C_k^c = C_{n-k}$, zamiast liczyć δ dla wektorów z C_k można to zrobić dla wektorów z C_{n-k}.

Lemat A.2.1 *Dla $z \leqslant \frac{n}{2}$ i $k \leqslant \frac{n}{2}$ zachodzi*

$$C_k \rightarrow_z C_{n-(k+z)} \cup \cdots \cup C_{n-|k-z|}.$$

Dowód Niech X oznacza maksymalny zbiór wektorów taki, że $C_k \rightarrow_z X$ dla $z \leqslant \frac{n}{2}$ i $k \leqslant \frac{n}{2}$.

Minimalne $\lambda(x)$ dla $x \in X$ jest równe minimalnej liczbie jedynek jakie można wziąć (biorąc z elementów) z pierwszej połowy pozycji (H_1) wektorów C_k zwiększonej o minimalną liczbę jedynek, które można wziąć z drugiej połowy pozycji. Pierwsza z tych wartości dla założonych k i z jest równa 0, ponieważ $z \leqslant n - k$. Możemy zatem skupić się na wyborze $n - z$ elementów z drugiej połówki (H_2). Minimalną liczbę jedynek pochodzących z drugiej połówki uzyskamy biorąc maksymalną liczbę zer równą k, tzn. minimalna liczba takich jedynek równa jest $n - z - k$.

Maksymalne $\lambda(x)$ dla $x \in X$ można wyliczyć następująco:

$$\max_{x \in X} \lambda(x) = \min\{z, k\} + \min\{n - z, n - k\}$$

$$= \begin{cases} k + n - z & z \geqslant k \\ z + n - k & z < k \end{cases}$$

$$= n - |z - k|.$$

Podsumowując, z H_1 można wziąć od 0 do $\min\{z, k\}$ jedynek, z H_2 od $n - (z + k)$ do $\min\{n - z, n - k\}$ jedynek, zatem $\lambda(x)$ może przyjąć dowolną wartość od $n - (z + k)$ do $n - |z - k|$, a w konsekwencji

$$C_k \rightarrow_z C_{n-(k+z)} \cup \cdots \cup C_{n-|z-k|}.$$

□

Lemat A.2.2 *Dla $z \leqslant \frac{n}{2}$ i $k > \frac{n}{2}$ zachodzi:*

$$C_k \rightarrow_z C_{|n-(k+z)|} \cup \cdots \cup C_{n+z-k}.$$

Dowód Niech X będzie maksymalnym zbiorem takim, że $C_k \rightarrow_z X$. Wyznaczenie minimalnej $\lambda(x)$ dla $x \in X$ wymaga rozpatrzenia dwóch przypadków.

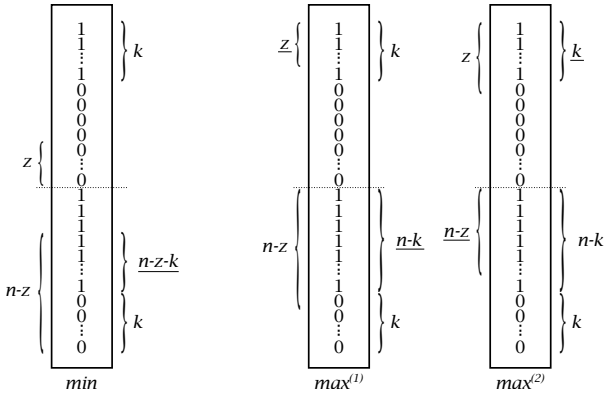
1. $z \geqslant n - k$. W tym przypadku z pierwszej połowy bierzemy wszystkie dostępne zera $(n - k)$, resztę będą stanowić jedynki, których będzie $z - (n - k)$. Z tego względu, iż $n - z \leqslant k$ to z drugiej połowy możemy wziąć same zera (k jest liczbą zer w drugiej połowie pozycji wektora). Zatem
$$\min_{x \in X} \lambda(x) = z - (n - k) = z + k - n.$$

2. $z < n-k$. Przy takim założeniu z pierwszej połowy (H_1) możemy wziąć same zera (liczba zer w H_1 jest większa od z), natomiast z drugiej możemy wziąć co najwyżej k zer, reszta spośród $n - z$ wybieranych elementów to jedynki. Otrzymujemy więc

$$\min_{x \in X} \lambda(x) = n - z - k.$$

Podsumowując wyniki dla obu przypadków możemy minimalne $\lambda(x)$ dla $x \in X$ wyrazić jako

$$\min_{x \in X} \lambda(x) = |n - z - k|.$$



Rysunek A.3: Dobór minimalnej i maksymalnej ilości jedynek

Maksymalna liczba jedynek jaką można uzyskać z C_k biorąc z z pierwszej połówki to z ($z < k$) oraz $n-k$ z drugiej (możemy wziąć wszystkie dostępne, gdyż $n - z > n - k$). Zatem

$$\max_{x \in X} \lambda(x) = z + n - k.$$

Podobnie jak w dowodzie poprzedniego lematu pokazaliśmy jakie mogą być maksymalne i minimalne wartości dla sumy jedynek wziętych z pierwszej i drugiej połówki. W obu połówkach możemy dobierać te ilości niezależnie, tym samym możemy otrzymać wektory $x \in X$ przyjmujące wszystkie pośrednie wartości $\lambda(x)$. □

Lemat A.2.3 *Dla $z \leqslant \frac{n}{2}$ i dowolnego dopuszczalnego k zachodzi*

$$C_k \rightarrow_z C_{|n-(k+z)|} \cup \cdots \cup C_{n-|z-k|}.$$

Dowód Wynika wprost z lematów A.2.1 i A.2.2. □

Lemat A.2.4 *Dla dowolnego dopuszczalnego z i k zachodzi*

$$C_k \to_z C_{|n-(k+z)|} \cup \cdots \cup C_{n-|z-k|}.$$

Dowód Dla $z > \frac{n}{2}$ możemy zastosować lemat A.2.3 dla C_{n-k} oraz $z_1 = n-z$ otrzymując

$$C_{n-k} \to_{z_1} C_{|n-(n-k+n-z)|} \cup \cdots \cup C_{n-|n-z-(n-k)|}$$

$$= C_{|n-(k+z)|} \cup \cdots \cup C_{n-|z-k|}.$$

Zauważmy dalej, że jeżeli zbiór X jest zbiorem spełniającym

$$C_{n-k} \to_{n-z} X,$$

to X jest również zbiorem spełniającym

$$C_k \to_z X.$$

□

Lemat A.2.5 *Niech X i Y oznaczają maksymalne podzbiory B_n takie, że $C_k \to_z X$ i $C_{n-k} \to_z Y$. Dla każdego dopuszczalnego z oraz k zachodzi*

$$\delta(X) = \delta(Y).$$

Dowód Z lematu A.2.4 otrzymujemy:

$$\min_{x \in X} \lambda(x) = |n - (k + z)|,$$

$$\max_{x \in X} \lambda(x) = n - |k - z|.$$

Można łatwo wykazać, że $|n - (k + z)|$ jest zawsze mniejsze od $n - |k - z|$. Zatem:

$$\delta(X) = \max\{|n - (k + z)|, n - (n - |k - z|)\} = \max\{|n - (k + z)|, |k - z|\}.$$

Ponownie korzystając z lematu A.2.4 możemy uzyskać ekstrema dla wartości $\lambda(y)$ dla $y \in Y$.

$$C_{n-k} \to_z C_{|n-(n-k+z)|} \cup \cdots \cup C_{n-|n-k-z|} = C_{|k-z|} \cup \cdots \cup C_{n-|n-k-z|}$$

$$\min_{y \in Y} \lambda(y) = |k - z|,$$

$$\max_{y\in Y}\lambda(y) = n - |n-k-z|.$$

Zatem z definicji funkcji δ

$$\begin{aligned}\delta(Y) &= \max\{|k-z|, n-(n-|n-k-z|)\}\\ &= \max\{|k-z|, |n-k-z|\} = \delta(X).\end{aligned}$$

□

Lemat A.2.6 *Dla każdego dopuszczalnego $z > 0$ oraz $k \leqslant \frac{n}{2} - 1$ istnieją $X, Y \subset B_n$ takie, że:*

$$C_k \to_z X, \quad C_{k+1} \to_z Y, \quad X \cap Y \neq \emptyset.$$

Dowód Niech $x \in X$ oraz $y \in Y$. Analizę możliwych zbiorów X i Y należy rozbić na 2 przypadki.

1. Jeśli $z + k < n$, wtedy z lematu A.2.4

   $$\min_{y\in Y}\lambda(y) = \min_{x\in X}\lambda(x) - 1 = n - (z+k+1),$$

   Zbiory X i Y mogą być rozłączne jedynie gdy

   $$\max_{y\in Y}\lambda(y) < \min_{x\in X}\lambda(x).$$

   a zatem

   $$\max_{y\in Y}\lambda(y) = \min_{y\in Y}\lambda(y).$$

   To jest możliwe tylko gdy z jest równe 0, co jest sprzeczne z założeniem.

2. Dla $z + k \geqslant n$, z lematu A.2.4

   $$\min_{y\in Y}\lambda(y) = \min_{x\in X}\lambda(x) + 1 = (z+k+1) - n.$$

   Zbiory X i Y mogą być rozłączne jeśli

   $$\max_{x\in X}\lambda(x) < \min_{y\in Y}\lambda(x),$$

   a zatem

   $$\max_{x\in X}\lambda(x) = \min_{x\in X}\lambda(x).$$

   To jednak wymaga, aby z było równe 0, co jest sprzeczne z założeniem.

□

Lemat A.2.7 *Niech q będzie pewną liczbą naturalną. Jeżeli $C_k \subset Q_i$ oraz $C_{k+q} \subset Q_i$ to*

$$\bigvee_{j<q} C_{k+j} \subset Q_i.$$

Dowód Wynika z lematów A.2.4, A.2.6 dla $z > 0$ oraz z lematu A.2.4 i prostej obserwacji, iż $X \to_0 X$. □

Lemat A.2.8 *Jeżeli* $\delta_i < \frac{n}{2}$*, to*

$$C_{\delta_i} \cup C_{\delta_i+1} \cup \cdots \cup C_{n-\delta_i} \subseteq Q_i.$$

Dowód Wynika z definicji δ oraz lematu A.2.7. □

Twierdzenie A.2.1 *Ciąg* (δ_i) *o elemencie początkowym* $\delta_0 = n$ *spełnia poniższą zależność rekurencyjną:*

$$\delta_{i+1} \leqslant \max\{0, \delta_i - \widetilde{\gamma}_{i+1} \cdot n\}.$$

Dowód Niech $z = Z_{i+1}$ oraz $\tilde{z} = \widetilde{\gamma}_{i+1} \cdot n = \min\{z, n - z\}$. Parametr δ_{i+1} w $i+1$ kroku zależy od z oraz δ_i. Badanie wartości δ_{i+1} podzielimy na przypadki w zależności od wartości δ_i i z.

1. $\delta_i > \frac{n}{2}$

   $$C_{\delta_i} \subseteq Q_i \vee C_{n-\delta_i} \subseteq Q_i$$

   Załóżmy, iż zachodzi $C_{\delta_i} \subseteq Q_i$ (przypadek $C_{n-\delta_i} \subseteq Q_i$ da nam to samo δ_{i+1} w następnym kroku, co zostało pokazane w lemacie A.2.5). Przy pomocy lematu A.2.4 możemy wyznaczyć dwie wartości graniczne:

   $$|n - (\delta_i + z)|, \quad n - |\delta_i - z|.$$

   Zauważmy, że dla wszystkich możliwych przypadków:

   $$n \geqslant \delta_i + z, \quad \delta_i \geqslant z,$$

   $$n < \delta_i + z, \quad \delta_i \geqslant z,$$

   $$n < \delta_i + z, \quad \delta_i < z,$$

   (przypadek $n \geqslant \delta_i + z, \quad \delta_i < z$ nie jest możliwy dla $\delta_i > \frac{n}{2}$)

   $$|n - (\delta_i + z)| \leqslant n - |\delta_i - z|.$$

   Możemy zatem oszacować z góry δ_{i+1} przez poniższą wartość

   $$\max\{|n - (\delta_i + z)|, n - (n - |\delta_i - z|)\} = \max\{|n - (\delta_i + z)|, |\delta_i - z|\}$$

   $$= \begin{cases} \max\{n - (\delta_i + z), \delta_i - z\} & n \geqslant \delta_i + z, \quad \delta_i \geqslant z \\ \max\{(\delta_i + z) - n, \delta_i - z\} & n < \delta_i + z, \quad \delta_i \geqslant z \\ \max\{(\delta_i + z) - n, z - \delta_i\} & n < \delta_i + z, \quad \delta_i < z \end{cases}$$

$$= \begin{cases} \max\{(n-\delta_i)-z, \delta_i - z\} & n \geqslant \delta_i + z, \quad \delta_i \geqslant z \\ \max\{\delta_i - (n-z), \delta_i - z\} & n < \delta_i + z, \quad \delta_i \geqslant z \\ \max\{z - (n-\delta_i), z - \delta_i\} & n < \delta_i + z, \quad \delta_i < z. \end{cases}$$

Zauważymy, że $\delta_i > n - \delta_i$ oraz, że dla $\delta_i < z$ zachodzi $z > \frac{n}{2}$, czyli $z > n - z$, stąd $\tilde{z} = n - z$. Z drugiej strony, jeżeli $n \geqslant \delta_i + z$ i $\delta_i \geqslant z$, to $z \leqslant \frac{n}{2}$ i $\tilde{z} = z$. Stąd

$$\begin{aligned} \delta_{i+1} &\leqslant \begin{cases} \delta_i - z & n \geqslant \delta_i + z, \quad \delta_i \geqslant z \\ \max\{\delta_i - (n-z), \delta_i - z\} & n < \delta_i + z, \quad \delta_i \geqslant z \\ z - (n - \delta_i) & n < \delta_i + z, \quad \delta_i < z \end{cases} \\ &= \begin{cases} \delta_i - \tilde{z} & n \geqslant \delta_i + z, \quad \delta_i \geqslant z \\ \max\{\delta_i - (n-z), \delta_i - z\} & n < \delta_i + z, \quad \delta_i \geqslant z \\ \delta_i - \tilde{z} & n < \delta_i + z, \quad \delta_i < z \end{cases} \\ &= \begin{cases} \max\{\delta_i - (n-z), \delta_i - z\} & n < \delta_i + z, \quad \delta_i \geqslant z \\ \delta_i - \tilde{z} & n \geqslant \delta_i + z \vee \delta_i < z \end{cases} \\ &= \begin{cases} \delta_i - z = \delta_i - \tilde{z} & n < \delta_i + z, \quad \delta_i \geqslant z, \quad z \leqslant \frac{n}{2} \\ \delta_i - (n-z) = \delta_i - \tilde{z} & n < \delta_i + z, \quad \delta_i \geqslant z, \quad z > \frac{n}{2} \\ \delta_i - \tilde{z} & n \geqslant \delta_i + z \vee \delta_i < z \end{cases} \\ &= \delta_i - \tilde{z}. \end{aligned}$$

2. Dla $\tilde{z} < \delta_i \leqslant \frac{n}{2}$, czyli $z < \delta_i$ lub $z > n - \delta_i$, z lematu A.2.8 otrzymujemy:

$$C_{\delta_i} \cup C_{n-\delta_i} \subseteq Q_i.$$

Powyższe zbiory możemy przekształcić w kolejnym kroku na następujące zbiory.

(a) Dla $z < \delta_i$ ($\tilde{z} = z$) z lematu A.2.1:

$$C_{\delta_i} \to_z C_{n-|\delta_i - z|} = C_{n-(\delta_i - \tilde{z})},$$

$$C_{n-\delta_i} \to_z C_{n-((n-\delta_i)+z)} = C_{\delta_i - \tilde{z}}.$$

(b) Dla $z > n - \delta_i$ ($\tilde{z} = n - z$) z lematu A.2.4:

$$C_{\delta_i} \to_z C_{|n-(\delta_i + z)|} = C_{\delta_i - \tilde{z}},$$

$$C_{n-\delta_i} \to_z C_{n-|(n-\delta_i)-z|} = C_{n-(\delta_i - \tilde{z})}.$$

Podsumowując oba powyższe przypadki otrzymujemy:

$$C_{\delta_i} \cup C_{n-\delta_i} \rightarrow_z C_{n-(\delta_i-\tilde{z})} \cup C_{\delta_i-\tilde{z}}.$$

Stąd

$$C_{\delta_i-\tilde{z}} \cup C_{n-\delta_i+\tilde{z}} \subseteq Q_{i+1},$$

co oznacza

$$\delta_{i+1} \leqslant \delta_i - \tilde{z}.$$

3. Dla $\delta_i \leqslant \frac{n}{2}$ oraz $\tilde{z} \geqslant \delta_i$ ($\delta_i \leqslant z \leqslant n - \delta_i$) z lematu A.2.8 wynika, iż:

$$C_z \cup C_{n-z} \subseteq Q_i,$$

co oznacza, że następnym kroku możemy uzyskać C_n i C_0:

$$C_z \rightarrow_z C_n,$$

$$C_{n-z} \rightarrow_z C_0.$$

Stąd

$$C_0 \cup C_n \subseteq Q_{i+1},$$

czyli w $i+1$ kroku otrzymujemy

$$\delta_{i+1} = 0.$$

Podsumowując trzy rozważane przypadki otrzymujemy

$$\delta_{i+1} \leqslant \begin{cases} \delta_i - \tilde{z} & \frac{n}{2} < \delta_i \\ \delta_i - \tilde{z} & \tilde{z} \leqslant \delta_i \leqslant \frac{n}{2} \\ 0 & \tilde{z} < \delta_i \leqslant \frac{n}{2} \end{cases} = \min\{0, \delta_i - \widetilde{\gamma} \cdot n\}.$$

□

Twierdzenie A.2.2 *W stałej liczbie kroków δ_i osiąga wartość 0 z dużym prawdopodobieństwem.*

Dowód Skorzystamy oszacowania zawartego w fakcie 2.4.1:

$$\Pr\{|X - EX| > tn\} \leqslant 2e^{-2t^2n}.$$

Dla $tn = n^{\frac{2}{3}}$ otrzymujemy:

$$\Pr\left\{\left|X - \frac{n}{2}\right| > n^{\frac{2}{3}}\right\} \leqslant 2 \cdot e^{-2\sqrt[3]{n}}.$$

Dla procesu (δ_i)

$$\Pr\left\{\widetilde{\gamma}_i \cdot n < \frac{n}{2} - n^{\frac{2}{3}}\right\} = \Pr\left\{\left|Z_i - \frac{n}{2}\right| > n^{\frac{2}{3}}\right\}.$$

Zauważmy, że dla $n \geqslant 64$

$$n^{\frac{2}{3}} \leqslant \frac{1}{4} \cdot n, \quad \frac{n}{2} - n^{\frac{2}{3}} \geqslant \frac{1}{4} \cdot n,$$

zatem

$$\Pr\left\{\widetilde{\gamma}_i \cdot n < \frac{n}{4}\right\} = \Pr\left\{\widetilde{\gamma}_i < \frac{1}{4}\right\} \leqslant 2 \cdot e^{-2\sqrt[3]{n}}.$$

$$\Pr\{\delta_4 = 0\} = \Pr\{\widetilde{\gamma}_1+\widetilde{\gamma}_2+\widetilde{\gamma}_3+\widetilde{\gamma}_4 \geqslant 1\} \geqslant \Pr\left\{\widetilde{\gamma}_1 \geqslant \frac{1}{4}, \widetilde{\gamma}_2 \geqslant \frac{1}{4}, \widetilde{\gamma}_3 \geqslant \frac{1}{4}, \widetilde{\gamma}_4 \geqslant \frac{1}{4}\right\}$$

$$= (1 - e^{-2\sqrt[3]{n}})^4$$

Dla $n \geqslant 216$

$$n^{\frac{2}{3}} < \frac{1}{6} \cdot n, \quad \frac{n}{2} - n^{\frac{2}{3}} > \frac{1}{3} \cdot n,$$

prawdopodobieństwo „wyzerowania" się δ_i w 3 kroku wynosi

$$\Pr\{\delta_3 = 0\} = \Pr\{\widetilde{\gamma}_1 + \widetilde{\gamma}_2 + \widetilde{\gamma}_3 \geqslant 1\} > \Pr\left\{\widetilde{\gamma}_1 > \frac{1}{3}, \widetilde{\gamma}_2 > \frac{1}{3}, \widetilde{\gamma}_3 > \frac{1}{3}\right\}$$

$$= (1 - e^{-2\sqrt[3]{n}})^3$$

□

A.3 Ewolucja rozkładów brzegowych – proces „stapiania"

Podobnie jak w przypadku procesu zanikania ograniczeń ilościowych proces przemieszania (lub stapiania) się obu grup wiadomości możemy wyrazić przy pomocy zmiennych losowych γ_i, $\widetilde{\gamma}_i$ oraz wyznaczającej je zmiennej losowej Z_i. Słowo stapianie nie jest tu przypadkowe – nasz proces możemy sobie wyobrazić w następujący sposób.

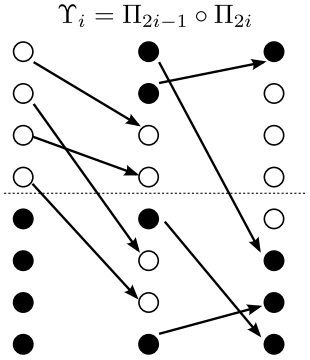
> W każdym kroku z urny zawierającej n kul ze stopu metalu m_1 oraz n kul ze stopu metalu m_2 wybieramy n kul, które stapiamy a następnie wytapiamy n nowych kul. Podobnie postępujemy z niewybraną grupą n kul.

W rezultacie na każdym etapie procesu otrzymujemy dwie grupy kul, składające się z 2 różnych stopów metali m_1 i m_2. Stopień przemieszania metali opisywał będzie parametr ε.

Zanim przejdziemy do opisu rozkładów brzegowych możemy sprawdzić zależność pomiędzy permutacjami bloków a wektorami binarnymi

$$\sum_{\Upsilon} \Pr\{Z_i = k|\Upsilon\} \cdot \Pr\{\Upsilon\} = \frac{\binom{n}{k}\binom{n}{k}k!\binom{n}{n-k}(n-k)!\,n!}{(2n)!} = \frac{\binom{n}{k}\binom{n}{n-k}}{\binom{2n}{n}}.$$

Zarysowany wcześniej proces „stapiania" będzie prowadził do dwóch typów rozkładów brzegowych w zależności od tego czy dana pozycja wektora należy do obrazu $\Upsilon_i(H_1)$, czy też do $\Upsilon_i(H_2)$. Wartości prawdopodobieństwa wystąpienia jedynki na pozycjach z tych grup będziemy oznaczać odpowiednio p_i oraz q_i. Zauważmy, iż $p_0 = 1$, $q_0 = 0$ oraz dla każdego i zachodzi $q_i = 1 - p_i$. Ta druga własność wynika z faktu, iż elementy grup mieszają się w odwrotnych proporcjach na pozycjach $\Upsilon_i(H_1)$ i $\Upsilon_i(H_2)$.



Rysunek A.4: Złożenie permutacji, a podział zbioru

Jeżeli przyjrzymy się mniej skrajnemu niż poprzednie przypadkowi wytworzenia się przekroju $H_1 \cap \Upsilon_i(H_1)$ w i-tym kroku (rysunek A.4), to zauważymy, iż proporcje uśredniania prawdopodobieństw dla pozycji bardziej białych i bardziej czarnych w $i+1$ kroku będą wynikały wprost z rozmiaru tego przekroju. We wspomnianym przypadku wynosi on 3, zatem $Z_i = 3, \gamma_i = \frac{3}{4}, \widetilde{\gamma}_i = \frac{1}{4}$, a w konsekwencji:

$$p_{i+1} = \frac{3}{4}p_i + \frac{1}{4}q_i, \quad q_{i+1} = \frac{1}{4}p_i + \frac{3}{4}q_i.$$

Rozważania te możemy uogólnić:

$$p_{i+1} = \gamma_{i+1}p_i + (1 - \gamma_{i+1})q_i,$$

$$q_{i+1} = \gamma_{i+1} q_i + (1 - \gamma_{i+1}) p_i.$$

Zależności te możemy również wyrazić przy pomocy wartości ε_i, będącej odchyleniem od $\frac{1}{2}$:

$$p_{i+1} = \frac{1}{2} + (2\gamma_{i+1} - 1)\varepsilon_i,$$

$$q_{i+1} = \frac{1}{2} - (2\gamma_{i+1} - 1)\varepsilon_i.$$

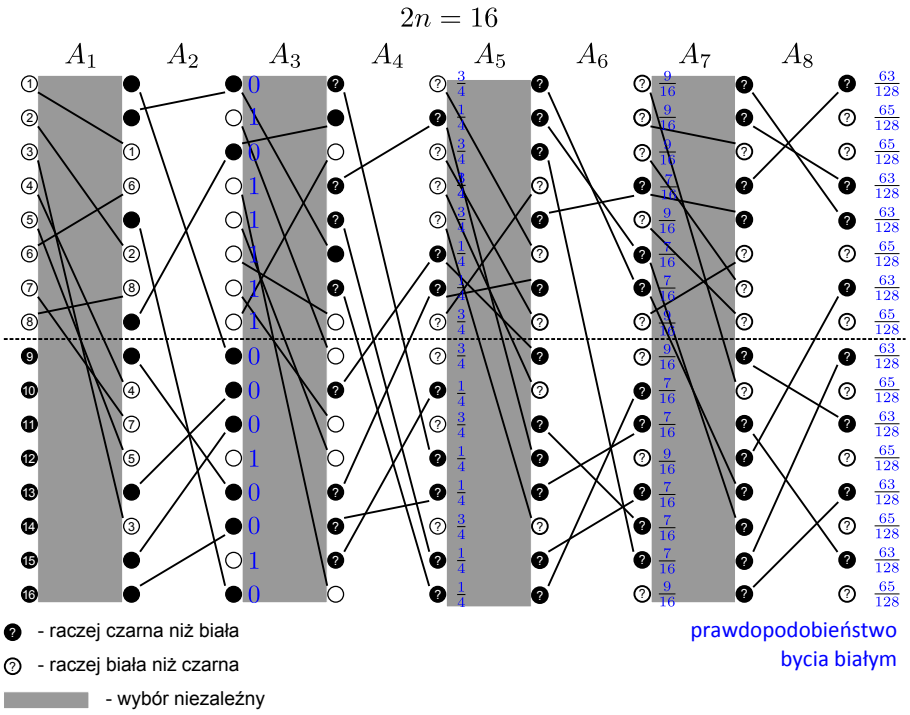
Zatem

$$\varepsilon_{i+1} = (2\gamma_{i+1} - 1)\varepsilon_i,$$

oraz w celu uniknięcia wartości ujemnych

$$\widetilde{\varepsilon}_{i+1} = (1 - 2\widetilde{\gamma}_{i+1})\widetilde{\varepsilon}_i = |2\gamma_{i+1} - 1|\widetilde{\varepsilon}_i.$$

Wartość $\widetilde{\varepsilon}_i$ będzie więc określała wiedzę obserwatora procesu ujawniania przejść w procedurze RPC po i krokach procesu ($\widetilde{\varepsilon}_0 = \varepsilon_0 = \frac{1}{2}$). Bardziej urozmaicony przykład ewolucji ε zawiera rysunek A.5.



Rysunek A.5: Analiza RPC

Twierdzenie A.3.1 *Dla dowolnego $\rho > 0$ oraz $\varepsilon > 0$ oraz ustalonej liczby serwerów $2i$ (i par) istnieje n_0 takie, że dla każdej liczby wiadomości $n \geqslant n_0$ zachodzi*

$$\Pr\{\widetilde{\varepsilon}_i \geqslant \varepsilon\} < \rho.$$

.

Dowód

$$\Pr\{\widetilde{\varepsilon}_i \geqslant \varepsilon\} \leqslant \Pr\left\{\exists_j |2\gamma_j - 1| \geqslant \sqrt[i]{\varepsilon}\right\} \leqslant i \cdot \Pr\left\{\left|\gamma_j \cdot n - \frac{n}{2}\right| \geqslant \frac{n}{2} \cdot \sqrt[i]{\varepsilon}\right\}$$

$$= i \cdot \Pr\left\{\left|Z_j - \frac{n}{2}\right| \geqslant \frac{n}{2} \cdot \sqrt[i]{\varepsilon}\right\}$$

Skorzystamy z oszacowania ogonowego dla zmiennej X o rozkładzie hipergeometrycznym (fakt 2.4.1):

$$\Pr\{|X - EX| \geqslant tn\} \leqslant 2\exp(-2t^2 n).$$

Zatem

$$\Pr\{\widetilde{\varepsilon}_i \geqslant \varepsilon\} \leqslant i \cdot \Pr\left\{\left|Z_j - \frac{n}{2}\right| \geqslant \frac{n}{2} \cdot \sqrt[i]{\varepsilon}\right\}$$

$$\leqslant 2i \cdot \exp\left(-\frac{n(\sqrt[i]{\varepsilon})^2}{2}\right).$$

□

A.4 Modelowanie rozkładu wektorów binarnych przy pomocy niezależnych zmiennych losowych

W tej części pracy skupimy się na wprowadzonym w podrozdziale 2.4.3 rozkładzie łącznym zmiennych losowych o wartościach binarnych

$$(X_1^{(i)}, X_2^{(i)}, \ldots, X_{2n}^{(i)}).$$

Rozkład tych zmiennych reprezentuje wiedzę obserwatora uzyskaną z procedury częściowego sprawdzania. Przyjęcie wartości $X_j^{(i)} = 1$ oznacza wystąpienie na j-tej pozycji wyjściowej, po i-tym bloku serwerów mieszających, cebuli z grupy białej (wartość 1 oznacza białe, 0 czarne). W wyniku ujawnienia połowy przejść w procedurze częściowego sprawdzania obserwator zna pozycje wejściowe sieci należące do grupy białych i czarnych. Co więcej, jest

w stanie wskazać n pozycji wyjściowych po każdym i-tym bloku serwerów, na których prawdopodobieństwo wystąpienia cebuli białej wynosi p_i ($p_i \geqslant \frac{1}{2}$). Obserwator jest w stanie wyliczyć p_i oraz wie, że dla pozostałych pozycji to prawdopodobieństwo wynosi $1 - p_i = q_i$.

W dalszych rozważaniach skupimy się na modelowaniu rozkładu prawdopodobieństw w danym kroku mieszania, a zatem pominięta zostanie indeksacja symboli zmiennych poprzez numer kroku (bloku) i. Dotyczy to oznaczeń: $X_j^{(i)}$, p_i, q_i, $\tilde{p}_i$, $\tilde{q}_i$, Θ_i, $\widetilde{\varepsilon}_i$.

Przyjmijmy uproszczenie zakładające, że większe prawdopodobieństwo wystąpienia 1 w danym kroku jest na pozycjach $1, 2, \ldots, n$ (oczywiście przy takim uproszczeniu będziemy ujawniać dowolny n-elementowy podzbiór pozycji) i rozważmy rozkład łączny

$$(X_1, X_2, \ldots, X_n, Y_1, Y_2, \ldots, Y_n) = (X_1, X_2, \ldots, X_{2n})\,.$$

Zauważmy, że (przy naszym założeniu o pierwszych n pozycjach) pierwsze n zmiennych są symetrycznie zależnymi zmiennymi losowymi (wymienialnymi zmiennymi losowymi; ang. *exchangeable random variables*). Podobnie zmienne losowe odpowiadające ostatnim n pozycjom są symetrycznie zależne. Zmienne X_j mają rozkład Bernoulliego z parametrem p podczas gdy zmienne Y_j mają rozkład Bernoulliego z parametrem $q = 1 - p$. Dodatkowo wiemy, że

$$\sum_{j=1}^{n} X_j + \sum_{j=1}^{n} Y_j = n\,.$$

Jednym ze sposobów otrzymania rozkładu łącznego symetrycznie zależnych zmiennych losowych jest użycie w odpowiedniej przestrzeni warunkowej zmiennych losowych niezależnych. Fakt ten i przeprowadzane symulacje były powodem by modelować łączny rozkład

$$(X_1, X_2, \ldots, X_n, Y_1, Y_2, \ldots, Y_n)$$

za pomocą warunkowego rozkładu prawdopodobieństwa $2n$ niezależnych zmiennych losowych

$$(\widetilde{X}_1, \widetilde{X}_2, \ldots, \widetilde{X}_n, \widetilde{Y}_1, \widetilde{Y}_2, \ldots, \widetilde{Y}_n)$$

również o rozkładach Bernoulliego, spośród których n pierwszych ma parametr $\tilde{p}$, a n następnych $1 - \tilde{p}$. Dokładniej postawiliśmy hipotezę, że

$$\Pr\left\{X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n, Y_1 = y_1, Y_2 = y_2, \ldots, Y_n = y_n\right\}$$

$$= \Pr\left\{\widetilde{X}_1 = x_1, \ldots, \widetilde{X}_n = x_n, \widetilde{Y}_1 = y_1, \ldots, \widetilde{Y}_n = y_n \,\Big|\, \sum_{j=1}^{n} \widetilde{X}_j + \sum_{j=1}^{n} \widetilde{Y}_j = n\right\},$$

gdzie

$$x_1, x_2 \ldots, x_n, y_1, y_2 \ldots, y_n \in \{0, 1\}\,.$$

Oczywiście hipoteza ta nie jest prawdziwa w przypadku istnienia ograniczeń liczbowych. Stąd wynikała konieczność badania takich ograniczeń. Podstawowe wyniki pokazane zostały w A.2. Ograniczyliśmy się tam do najprostszego przypadku – zanikania ich w sensie uzyskania niezerowego prawdopodobieństwa dla dowolnego wektora binarnego z dokładnie n jedynkami.

Otrzymywane, przy założeniu prawdziwości naszej hipotezy, rozkłady z odpowiednio dobranym parametrem są bardzo bliskie w sensie metryki całkowitego wahania uzyskanym w symulacjach rozkładom. Należy zaznaczyć, iż wyniki te mogą być interesujące również w oderwaniu od analizy procedur częściowego sprawdzania.

W naszych dalszych rozważaniach będziemy zakładać prawdziwość powyższej hipotezy. W szczególności prowadzi to do wzoru na prawdopodobieństwo uzyskania dokładnie k jedynek na pierwszych n pozycjach

$$\Pr\{C_k\} = \Pr\left\{\sum_{j=0}^{n} X_j = k\right\} = \Pr\left\{\sum_{j=0}^{n} \widetilde{X}_j = k \,\middle|\, \sum_{j=0}^{n} (\widetilde{X}_j + \widetilde{Y}_j) = n\right\}.$$

Stąd

$$\Pr\{C_k\} = \frac{\binom{n}{k}\binom{n}{n-k}\tilde{p}^k \cdot \tilde{p}^k \cdot \tilde{q}^{n-k} \cdot \tilde{q}^{n-k}}{\sum\limits_{j=0}^{n} \binom{n}{j}\binom{n}{n-j}\tilde{p}^j \cdot \tilde{p}^j \cdot \tilde{q}^{n-j} \cdot \tilde{q}^{n-j}} = \frac{\binom{n}{k}^2 \Theta^k}{\sum\limits_{j=0}^{n} \binom{n}{j}^2 \Theta^j} = \frac{\binom{n}{k}^2 \Theta^k}{C(n, \Theta)}\,,$$

gdzie

$$\Theta = \left(\frac{\tilde{p}}{\tilde{q}}\right)^2 = \left(\frac{\tilde{p}}{1-\tilde{p}}\right)^2, \quad C(n, \Theta) = \sum_{j=0}^{n} \binom{n}{j}^2 \Theta^j.$$

Zauważmy, że $\Theta \geqslant 1$ przy założeniu $\tilde{p} \geqslant \tilde{q}$. Prawdopodobieństwo konkretnego wektora o dokładnie k jedynkach na n pierwszych pozycjach (wektora należącego do C_k) wynosi zatem

$$\frac{\Theta^k}{C(n, \Theta)}.$$

Idąc dalej, można wyznaczyć zależność między parametrem p rozkładów brzegowych $X_1, \ldots, X_n$ (i parametrem $1 - p$ rozkładów brzegowych $Y_1, \ldots, Y_n$) a Θ (określonym powyżej za pomocą parametru $\tilde{p}$ związanym z niezależnymi zmiennymi losowymi $\widetilde{X}_1, \ldots, \widetilde{X}_n, \widetilde{Y}_1, \ldots, \widetilde{Y}_n$)

$$p = p(\Theta) = \Pr\{X_1 = 1\} = \sum_{k=1}^{n} \Pr\{X_1 = 1 \wedge X \in C_k\} = \frac{\sum\limits_{k=1}^{n} \binom{n-1}{k-1}\binom{n}{k}\Theta^k}{C(n, \Theta)}.$$

Z tego względu, iż pochodna funkcji $C(n,\Theta)$ wynosi

$$C'(n,\Theta) = n\sum_{j=1}^{n}\binom{n-1}{k-1}\binom{n}{k}\Theta^{j-1},$$

zależność p od Θ możemy wyrazić jako

$$p(\Theta) = \frac{n}{\Theta}\frac{C'(n,\Theta)}{C(n,\Theta)}.$$

Lemat A.4.1 *Rozkład prawdopodobieństw dla poszczególnych klas wektorów, oraz dla* $k \leqslant \left\lfloor \frac{n-1}{2} \right\rfloor$ *posiada następujące własności:*

W1. $\Pr\{C_{n-k}\} \leqslant \Pr\{C_k\}$,

W2. $\Pr\{C_k\} \leqslant \Pr\{C_{k+1}\} \quad (k < \left\lfloor \frac{n-1}{2} \right\rfloor)$,

W3. $\Pr\{C_k\} \leqslant \binom{n}{k}^2\binom{2n}{n}^{-1} \leqslant \Pr\{C_{n-k}\} \quad (\varepsilon < \frac{1}{n})$.

Dowód

W1. Ponieważ $\Theta \geqslant 1$

$$\frac{\Pr\{C_{n-k}\}}{\Pr\{C_k\}} = \Theta^{2k-n} = \left(\frac{1}{\Theta}\right)^{n-2k} \leqslant 1$$

W2.

$$\frac{\Pr\{C_k\}}{\Pr\{C_{k+1}\}} = \frac{\binom{n}{k}^2}{\binom{n}{k+1}^2}\frac{1}{\Theta} = \left(\frac{k+1}{n-k}\right)^2\frac{1}{\Theta}$$

Dla $k < \left\lfloor \frac{n-1}{2} \right\rfloor$ zachodzi $k \leqslant \frac{n}{2} - \frac{1}{2}$ oraz $k+1 \leqslant n-k$. Zatem

$$\left(\frac{k+1}{n-k}\right)^2 \leqslant 1 \leqslant \Theta,$$

a w konsekwencji

$$\frac{\Pr\{C_k\}}{\Pr\{C_{k+1}\}} \leqslant 1.$$

W3. Poniżej zdefiniowana jest funkcja f_k, która oddaje zależność pomiędzy Θ a prawdopodobieństwem wystąpienia pewnego wektora należącego do klasy C_k. Zauważmy, że $f_k(1)$ zwraca prawdopodobieństwa rozkładu jednostajnego, tzn wartość $\binom{2n}{n}^{-1}$. Zbadanie monotoniczności tej

funkcji oraz $\tilde{f}_k = f_{n-k}$ dla $\Theta \to_+ 1$ (założenie $\tilde{p} \geqslant \tilde{q}$) pozwoli ustalić, czy rzeczywiście wartości prawdopodobieństw rozkładu jednostajnego znajdują się pomiędzy prawdopodobieństwami wektorów klas C_k oraz C_{n-k}. Niech $u \in C_k$ oraz $v \in C_{n-k}$.

I. $\Pr\{u\} \leqslant \binom{2n}{n}^{-1}$

$$f_k(\Theta) = \frac{\Theta^k}{\sum\limits_{j=0}^{n} \binom{n}{j}^2 \Theta^j} = \frac{\Theta^k}{C(n,\Theta)}$$

$$f_k'(\Theta) = \frac{k\Theta^{k-1}C(n,\Theta) - \Theta^k C'(n,\Theta)}{[C(n,\Theta)]^2}$$

$$f_k'(\Theta) \leqslant 0$$

$$kC(n,\Theta) \leqslant \Theta C(n,\Theta)$$

$$k \leqslant \frac{\Theta C'(n,\Theta)}{C(n,\Theta)} = np$$

II. $\binom{2n}{n}^{-1} \leqslant \Pr\{v\}$

$$\tilde{f}_k(\Theta) = f_{n-k}(\Theta) = \frac{\Theta^{n-k}}{C(n,\Theta)}$$

$$\tilde{f}_k'(\Theta) \geqslant 0$$

$$\tilde{f}_k(\Theta) = \frac{(n-k)\Theta^{n-k-1}C(n,\Theta) - \Theta^{n-k}C'(n,k)}{[C(n,\Theta)]^2}$$

$$(n-k)C(n,\Theta) \geqslant \Theta C'(n,\Theta)$$

$$n - k \geqslant \frac{\Theta C'(n,\Theta)}{C(n,\Theta)} = np$$

$$k \leqslant n(1-p)$$

□

Lemat A.4.2 *Pomiędzy $\widetilde{\varepsilon}$, a prawdopodobieństwami poszczególnych klas wektorów zachodzi następująca zależność*

$$2\widetilde{\varepsilon} = \sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor} \frac{n-2k}{n} (\Pr\{C_{n-k}\} - \Pr\{C_k\}).$$

Dowód Zauważmy, że możemy założyć, iż najbardziej prawdopodobny jest wektor $(1,1,\ldots,1,0,0,\ldots,0)$, a najmniej prawdopodobny $(0,0,\ldots,0,1,1,\ldots,1)$.

$$2\widetilde{\varepsilon} = \Pr\{X_1=1\} - \Pr\{X_1=0\} = \sum_{k=1}^{n} \Pr\{X_1=1 \wedge C_k\} - \sum_{k=0}^{n-1} \Pr\{X_1=0 \wedge C_k\}$$

$$= \sum_{k=0}^{n} [\Pr\{X_1=1 \wedge C_k\} - \Pr\{X_1=0 \wedge C_k\}]$$

Przy wykorzystaniu prostej kombinatorycznej zależności

$$\binom{n-1}{k-1}\binom{n}{k} - \binom{n-1}{k}\binom{n}{k} = \binom{n}{k}\binom{n}{k}\frac{2k-n}{n}$$

możemy zapisać sumę różnic jako

$$\sum_{k=0}^{n} [\Pr\{X_1=1 \wedge C_k\} - \Pr\{X_1=0 \wedge C_k\}] = \sum_{k=0}^{n} \Pr\{C_k\}\frac{2k-n}{n}.$$

Zauważmy, że suma ta jest równa

$$= \sum_{j=0}^{\lfloor \frac{n-1}{2} \rfloor} \frac{n-2j}{n} (\Pr\{C_{n-j}\} - \Pr\{C_j\}).$$

□

Twierdzenie A.4.1 *Dla* $\widetilde{\varepsilon} < \frac{1}{n}$ *zachodzi*

$$d_{TV}(X,\mu) \leqslant n\widetilde{\varepsilon}.$$

Dowód Niech $\mathcal{C} = \mathcal{L}(X)$, wtedy

$$d_{TV}(\mathcal{C},\mu) = \frac{1}{2}\sum_{k}\sum_{w\in C_k} |\mu(w) - \mathcal{C}(w)| = \frac{1}{2}\sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor}\sum_{w\in C_k} |\mu(w) - \mathcal{C}(w)| +$$

$$+\frac{1}{2}\sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor}\sum_{w\in C_{n-k}} |\mu(w) - \mathcal{C}(w)| + \frac{1}{2}R(n),$$

gdzie

$$R(n) = \begin{cases} \sum\limits_{w\in C_{\frac{n}{2}}} |\mu(w) - \mathcal{C}(w)| & 2 \mid n \\ 0 & 2 \nmid n \end{cases}$$

Z własności W3 zawartej w lemacie A.4.1 możemy pozbyć się wartości bezwzględnych

$$d_{TV}(\mathcal{C},\mu)=\frac{1}{2}\sum_{k=0}^{\lfloor\frac{n-1}{2}\rfloor}(\Pr\{C_{n-k}\}-\Pr\{C_k\})+\frac{1}{2}\left(\binom{n}{\frac{n}{2}}^2\binom{2n}{n}^{-1}-\Pr\{C_{\frac{n}{2}}\}\right)$$

Z lematu A.4.2

$$2\widetilde{\varepsilon}=\sum_{j=0}^{\lfloor\frac{n-1}{2}\rfloor}\frac{n-2j}{n}(\Pr\{C_{n-j}\}-\Pr\{C_j\})$$

$$n\widetilde{\varepsilon}=\frac{1}{2}\sum_{j=0}^{\lfloor\frac{n-1}{2}\rfloor}(n-2j)(\Pr\{C_{n-j}\}-\Pr\{C_j\})=\frac{1}{2}\sum_{j=0}^{\lfloor\frac{n-1}{2}\rfloor}(\Pr\{C_{n-j}\}-\Pr\{C_j\})+$$

$$+\frac{1}{2}\sum_{j=0}^{\lfloor\frac{n-1}{2}\rfloor-1}(n-2j-1)(\Pr\{C_{n-j}\}-\Pr\{C_j\})$$

$$\frac{1}{2}\sum_{j=0}^{\lfloor\frac{n-1}{2}\rfloor-1}(n-2j-1)(\Pr\{C_{n-j}\}-\Pr\{C_j\})\geqslant\frac{1}{2}\left(\binom{n}{\frac{n}{2}}^2\binom{2n}{n}^{-1}-\Pr\{C_{\frac{n}{2}}\}\right)$$

□

Bibliografia

[1] M. Abe. Mix-networks on permutation networks. In K.-Y. Lam, E. Okamoto i C. Xing, editors, *ASIACRYPT*, volume 1716 of *Lecture Notes in Computer Science*, pages 258–273. Springer, 1999.

[2] B. Adida i R. Rivest. Scratch and vote: Self-contained paper-based cryptographic voting. In R. Dingledine i T. Yu, editors, *ACM Workshop on Privacy in the Electronic Society*. ACM, 2006.

[3] D. Beaver, S. Micali i P. Rogaway. The round complexity of secure protocols. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 503–513, 1990.

[4] J. Benaloh. Secret sharing homomorphisms: keeping shares of a secret secret. In *Proceedings on Advances in cryptology, CRYPTO '86*, pages 251–260, London, UK, 1987. Springer-Verlag.

[5] R. Berman, A. Fiat i A. Ta-Shma. Provable unlinkability against traffic analysis. In *Financial Cryptography*, pages 266–280, 2004.

[6] R. Bubley i M. Dyer. Path coupling: A technique for proving rapid mixing in markov chains. In *FOCS*, pages 223–231. IEEE Computer Society, 1997.

[7] M. Burmester, V. Chrissikopoulos i P. Kotzanikolaou. Secure transactions with mobile agents in hostile environments. In *Proceedings of the 5th Australasian Conference on Information Security and Privacy*, pages 289–297, 2000.

[8] J. Castellà-Roca, J. Herrera-Joancomartí i A. Dorca-Josa. A secure e-exam management system. In *ARES*, pages 864–871, 2006.

[9] J. Castellà-Roca, J. Herrera-Joancomartí i J. Prieto-Blázquez. A secure electronic examination protocol using wireless networks. In *ITCC'04: Proceedings of the International Conference on Information Technology:*

Coding and Computing (ITCC'04) Volume 2, page 263, Washington, DC, USA, 2004. IEEE Computer Society.

[10] T. B. S. Center. Pdf-417. `http://www.mecsw.com/specs/pdf417.html`, 2003.

[11] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology Proceedings of Crypto 82*, pages 199–203, 1982.

[12] D. Chaum. Secret-ballot: True voter verifiable elections. *IEEE Security and Privacy*, 2(1):38–47, 2004.

[13] D. Chaum i T. Pedersen. Wallet databases with observers. In *CRYPTO '92: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 89–105, London, UK, 1993. Springer-Verlag.

[14] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, 1981.

[15] CIVTF. A report on the feasibility of internet voting. `http://www.ss.ca.gov/executive/ivote/final_report.htm`, 2000.

[16] J. Claessens, C. Diaz, B. Preneel i S. Seys. Towards measuring anonymity. In R. Dingledine i P. Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, 2002.

[17] M. R. Clarkson, S. Chong i A. C. Myers. Civitas: Toward a secure voting system. In *2008 IEEE Symposium on Security and Privacy*, pages 354–368, 2008.

[18] E. N. E. Committee. E-voting system - overview. `http://www.vvk.ee/elektr/docs/Yldkirjeldus-eng.pdf`, 2006.

[19] I. Damgård. Commitment schemes and zero-knowledge protocols. In *Lectures on Data Security*, pages 63–86, 1998.

[20] G. Danezis i A. Serjantov. Towards an information theoretic metric for anonymity. In *Proceedings of the 2nd international conference on Privacy enhancing technologies*, pages 41–53. Springer-Verlag, 2002.

[21] S. Delaune, S. Kremer i M. Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *Proceedings of the 19th IEEE workshop on Computer Security Foundations*, pages 28–42, Washington, DC, USA, 2006. IEEE Computer Society.

[22] Y. Desmedt i Y. Frankel. Threshold cryptosystems. In *Proceedings on Advances in cryptology*, CRYPTO '89, pages 307–315. Springer-Verlag New York, Inc., 1989.

[23] W. Feller. *Wstęp do rachunku prawdopodobieństwa*, volume II. Państwowe Wydawnictwo Naukowe, 1978.

[24] A. Fiat i A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

[25] A. Fujioka, T. Okamoto i K. Ohta. A practical secret voting scheme for large scale elections. In *ASIACRYPT '92: Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques*, pages 244–251, London, UK, 1993. Springer-Verlag.

[26] M. Gogolewski, L. Nitschke, M. Ren i T. Tyksiński. Przegląd systemów egzaminowania elektronicznego pod względem bezpieczeństwa. Technical Report 137/2010, Uniwersytet im. Adama Mickiewicza, 2010.

[27] M. Gogolewski, L. Nitschke, M. Ren i T. Tyksiński. Bezpieczne systemy zdalnego egzaminowania w e-learningu i gospodarce opartej na wiedzy. *Zeszyty Naukowe Uniwersytetu Szczecińskiego Nr 651 Ekonomiczne Problemy Usług nr 68*, pages 204–212, 2011.

[28] S. Goldwasser i M. Bellare. Lecture notes on cryptography, 2001.

[29] S. Goldwasser i S. Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, STOC '82, pages 365–377, New York, NY, USA, 1982. ACM.

[30] P. Golle i M. Jakobsson. Reusable anonymous return channels. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2003)*, 2003.

[31] M. Gomułkiewicz, M. Klonowski i M. Kutyłowski. Rapid mixing and security of chaum's visual electronic voting. In *In Proceedings of ESORICS 2003*, pages 132–145. Springer-Verlag, 2003.

[32] M. Gomułkiewicz, M. Klonowski i M. Kutyłowski. Provable unlinkability against traffic analysis already after o(log(n. In *Steps!, Information Security Conference (ISC)2004, LNCS 3225*, pages 354–366. Springer-Verlag, 2004.

[33] B. Hosp i S. Popoveniuc. An introduction to punchscan. In *IAVoSS Workshop On Trustworthy Elections (WOTE)*, 2006.

[34] A. Huszti i A. Pethö. A secure electronic exam system, 2008.

[35] A. Huszti i A. Pethö. A secure electronic exam system. Technical Report 77/3-4, University of Debrecen, 2010.

[36] M. Jakobsson. On quorum controlled asymmetric proxy re-encryption. In *Public Key Cryptography*, volume 1560 of *Lecture Notes in Computer Science*, pages 112–121. Springer, 1999.

[37] M. Jakobsson i A. Juels. Millimix: Mixing in small batches. Technical report, Center for Discrete Mathematics and Theoretical Computer Science (DIMACS), 1999. PODC'01.

[38] M. Jakobsson, A. Juels i R. L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX Security Symposium*, pages 339–353, 2002.

[39] S. Janson, T. Łuczak i A. Ruciński. *Random Graphs*. Wiley-Interscience, 2000.

[40] W. S. Juang i C. L. Lei. A secure and practical electronic voting scheme for real world environment. *TIECE: IEICE Transactions on Communications/Electronics/Information and Sytems*, pages 64–71, 1997.

[41] I. Jung i H. Yeom. Enhanced security for online exams using group cryptography. *IEEE TRANSACTIONS ON EDUCATION, VOL.52, NO.3*, 2009.

[42] S. Khazaei i D. Wikstrom. Randomized partial checking revisited. 7779:115–128, 2013.

[43] J. Kilian i K. Sako. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In L. Guillou i J.-J. Quisquater, editors, *Advances in Cryptology - EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 393–403. Springer, 1995.

[44] M. Klonowski, M. Kutyłowski, A. Lauks i F. Zagórski. A practical voting scheme with receipts. *Lecture Notes in Computer Science*, 3650:490–497, 2005.

[45] M. Kutyłowski i F. Zagórski. Verifiable internet voting solving secure platform problem. *Lecture Notes in Computer Science*, 4752:199–213, 2007.

[46] B. Lee i K. Kim. Receipt-free electronic voting scheme with a tamper-resistant randomizer. In *ICISC*, pages 389–406, 2002.

[47] A. Menezes, S. Vanstone i P. Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[48] A. Neff. A verifiable secret shuffle and its application to e-voting. In P. Samarati, editor, *ACM CCS '01*, pages 116–125. ACM Press, 2001.

[49] W. Niemiro. Markowowskie monte carlo v. elementarna teoria łańcuchów markowa. `http://mst.mimuw.edu.pl/lecture.php?lecture=sst&part=Ch14`, 2010.

[50] L. Nitschke. Protokoły interakcyjnego generowania kluczy. Master's thesis, Uniwersytet im. Adama Mickiewicza w Poznaniu, 2003.

[51] L. Nitschke. Remote voting using smart cards with display. In *Tatra Mountains Mathematical Publications*, pages 119–131, 2008.

[52] L. Nitschke. Secure internet voting based on paper ballots. In *Proceedings of ICISS 2008*, pages 102–115, 2008.

[53] L. Nitschke, M. Paprzycki i M. Ren. Mobile agent security. *NATO Security through Science Series, D: Information and Communication Security Volume 6*, 2006.

[54] T. Okamoto. Receipt-free electronic voting scheme for large scale elections. In *5th International Workshop on Security Protocols*, pages 25–35, London, UK, 1997. Springer-Verlag.

[55] R. Oppliger. How to address the secure platform problem for remote internet voting. In *Proceedings 5th Conf. on Security in Information Systems (SIS 2002)*, pages 153–173, 2002.

[56] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology, EUROCRYPT 1999*, pages 223–238. Springer-Verlag, 1999.

[57] C. Park, K. Itoh i K. Kurosawa. Efficient anonymous channel and all/nothing election scheme. In *EUROCRYPT '93: Workshop on the*

theory and application of cryptographic techniques on Advances in cryptology, pages 248–259, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[58] T. P. Pedersen. A threshold cryptosystem without a trusted party. In *EUROCRYPT*, pages 522–526, 1991.

[59] B. Pfitzmann i A. Pfitzmann. How to break the direct rsa-implementation of mixes. In *EUROCRYPT*, volume 434 of *Lecture Notes in Computer Science*, pages 373–381. Springer, 1989.

[60] M. Radwin i P. Klein. An untraceable, universally verifiable voting scheme, 1995.

[61] R. Rivest, A. Shamir i L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21:120–126, 1978.

[62] R. L. Rivest. Abstract the threeballot voting system. `http://theory.csail.mit.edu/~rivest/Rivest-TheThreeBallotVotingSystem.pdf`, 2006.

[63] P. Ryan i T. Peacock. Prêt à voter: a systems perspective. Technical report, Newcastle University, 2005.

[64] P. Ryan i S. Schneider. Prêt à voter with re-encryption mixes. In *ESORICS*, pages 313–326, 2006.

[65] T. Sander i C. Tschudin. Towards mobile cryptography. *International Computer Science Institute technical report 97-049*, 1997.

[66] T. Sander i C. Tschudin. Protecting mobile agents against malicious hosts. *Lecture Notes in Computer Science 1419*, 1998.

[67] B. Schneier i A. Shostack. Breaking up is hard to do: Modeling security threats for smart cards, 1999.

[68] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[69] SureVote. E-voting system - overview. `http://www.vote.caltech.edu/wote01/pdfs/surevote.pdf`, 2001.

[70] S. Tate i K. Xu. On garbled circuits and constant round secure function evaluation. *Computer Privacy and Security Lab, Department of Computer Science, University of North Texas, Technical Report 2003-02*, 2003.

[71] J. van de Graaf. Merging prêt à voter and punchscan. eprint.iacr.org/2007/269.pdf, 2007.

[72] Z. Xia, S. Schneider i J. Heather. Analysis, improvement and simplification of pret a voter with paillier encryption. In *USENIX/ACCURATE Electronic Voting Technology Workshop*, 2008.

[73] A. Yao. How to generate and exchange secrets. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, pages 162–167, Washington, DC, USA, 1986. IEEE Computer Society.

[74] A. Young i M. Yung. The dark side of black-box cryptography, or: Should we trust Capstone? *Lecture Notes in Computer Science*, 1109, 1996.

[75] A. Young i M. Yung. Kleptography: Using cryptography against cryptography. In *EUROCRYPT*, pages 62–74, 1997.

[76] A. Young i M. Yung. Yygen: A backdoor-resistant rsa key generator. http://cryptovirology.com/cryptovfiles/newbook/Chapter12.pdf, 2005.