

Implementacja wybranego protokołu przetargu elektronicznego

Protokoły Kryptograficzne

Dominik Januszewicz

Paweł Klimczuk

Prowadzący

mgr inż. Albert Sitek

semestr 15Z

Spis treści

1.	Opis tematu projektu	3
1.1.	Wybrany protokół przetargu elektronicznego	3
2.	Definicje wykorzystywanych narzędzi i pojęć	4
2.1.	Środowisko i narzędzia	4
2.2.	Pojęcia użyte w realizacji projektu	4
3.	Koncepcja rozwiązania problemu.....	5
3.1.	Ogólny zarys protokołu	5
3.2.	Szczegóły podprotokołów	6
3.2.1.	Podprotokół nr 1 – rejestracja i uwierzytelnianie użytkowników	6
3.2.2.	Podprotokół nr 2 – utworzenie przetargu	6
3.2.3.	Podprotokół nr 3 – ogłoszenie przetargu	6
3.2.4.	Podprotokół nr 4 – rozstrzygnięcie przetargu	7
3.3.	Koncepcja architektoniczna aplikacji.....	8
3.3.1.	GAP	8
4.	Zastosowania praktyczne implementowanego rozwiązania.....	9
5.	Instrukcja użytkowania programu	10
5.1.	KeyGen	10
5.2.	GAP	10
5.3.	eAukcja	10
5.3.1.	Tworzenie konta	10
5.3.2.	Uruchomienie aplikacji eAukcja	11
5.3.2.1.	Przygotowanie	11
5.3.2.2.	Logowanie	11
5.3.3.	Okno główne programu.....	11
5.3.4.	Wzięcie udziału w istniejącym przetargu	13
5.3.5.	Stworzenie przetargu	14
5.3.5.1.	Rozstrzygnięcie przetargu	15
6.	Raport z testów aplikacji	16
7.	Podsumowanie	17
	Bibliografia.....	17

1. Opis tematu projektu

Przetarg elektroniczny jest usprawnieniem realizacji przetargu w jego klasycznej formie. Użycie współczesnej techniki do realizacji tego sposobu wyboru najkorzystniejszej oferty pozwala na oszczędność czasu, środków, jest wygodne w użyciu oraz zapewnia nienaruszalną zgodność z przyjętymi zasadami przetargu.

Klasyczna forma przetargu zapewnia optymalizację kosztów związanych z realizacją zadania. Jest to szczególnie ważne w przypadku państwowych podmiotów, gdzie zarządzane środki powinny być wydawane z rozważą i w najkorzystniejszym dla ogółu obywateli sposób. Poza ceną (która zazwyczaj jest głównym i jedynym kryterium – 91% wszystkich ogłoszeń w 2010 roku), stosuje się także kryteria doświadczenia, długości gwarancji na przedmiot realizowanego zamówienia, kosztów eksploatacji czy czasu realizacji. Zamawiający zobowiązany jest podać wagę każdego kryterium bądź uszeregować je w kolejności według ważności.

Tematem projektu jest budowa narzędzia, które przy wykorzystaniu odpowiednich protokołów kryptograficznych, będzie mogło efektywnie wspomóc realizację wyżej opisanego procesu. W kolejnych rozdziałach zostaną przedstawione definicje związane z tym zagadnieniem oraz koncepcja rozwiązania problemu. Na samym końcu zamieszczono bibliografię.

Protokół zapewnia kompleksowe bezpieczeństwo:

- **Bezsporność uczestników** - GAP jako zaufana trzecia strona sprawdza wymagane dokumenty do uczestnictwa i ogłaszania przetargów.
- **Integralność danych** – oferty i wyniki przetargu są podpisywane odpowiednimi kluczami prywatnymi.
- **Pewność ofert** - oferent nie może wyprzeczyć swojej oferty
- **Anonimowość ofert** – oferty kodowane są kluczem publicznym aukcji. Odpowiedni klucz prywatny dzielony jest pomiędzy GAP, firmę ogłaszającą przetarg i oferentów.
- **Anonimowość zwycięzcy** – po wyborze zwycięzcy tylko indywidualny numer oferenta podawany jest do publicznej wiadomości.
- **Publiczna weryfikacja** – wszystkie indywidualne numery uczestników przetargu są publikowane po wyborze zwycięzcy – każdy z nich może sprawdzić czy jego oferta była wzięta pod uwagę, czy nie.

1.1. Wybrany protokół przetargu elektronicznego

Przyjęto w realizacji protokół przedstawiony w pozycji drugiej w bibliografii - *Cryptographic protocol for electronic auctions*. Zaimplementowana wersja pozwala na przeprowadzenia aukcji typu 1st - *Price Sealed-Bid*, tj. uczestnicy składają swoje oferty, które pozostają tajne do momentu zakończenia aukcji. Wówczas oferty stają się jawne, a wygrywa osoba, która zaoferowała najkorzystniejsze warunki.

2. Definicje wykorzystywanych narzędzi i pojęć

2.1. Środowisko i narzędzia

W projekcie planowane jest wykorzystanie języka C++ wraz z bibliotekami Qt (graficzna) oraz Crypto++ i Botan (kryptograficzne).

2.2. Pojęcia użyte w realizacji projektu

B₁...B_N – uczestnik biorący udział w przetargu. Składa swoją ofertę.

F – organizator przetargu. Odpowiada za ustalenie warunków i wybór najkorzystniejszej oferty.

GAP – główny serwer przetargu. Zarządza przetargami, realizuje zadania strony trzeciej w procedurze przetargowej.

Integralność danych – zarówno zawartość wysłanych ofert jak i finalny rezultat przetargu nie mogą być modyfikowane.

Niezaprzeczalność ofert – biorący udział w przetargu nie może wyprzeczyć swojej oferty ani faktu jej złożenia.

Niezaprzeczalność uczestników – jedynie uprawnione osoby mogą zarówno ogłosić przetarg elektroniczny jak i wziąć w nim udział.

Pewność ofert – nikt nie może ustalić treści ofert przed końcem przetargu.

Prywatny klucz – klucz znany tylko dla jednej strony. Za jego pomocą strona może dekodować szyfrogramy wysyłane do niej zakodowane za pomocą jej klucza publicznego.

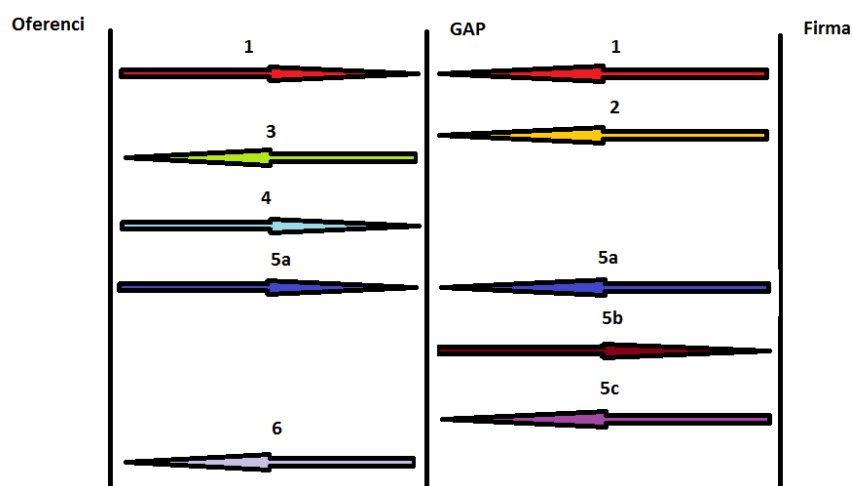
Publiczna weryfikacja – każdy może sprawdzić, czyja oferta wygrała przetarg. Uczestniczący w przetargu mogą ponadto sprawdzić, czy ich oferta została wzięta pod uwagę.

Publiczny klucz – klucz udostępniany wszystkim, służący do szyfrowania wiadomości. Za jego pomocą nie można odczytać szyfrogramu.

Podział sekretu – technika stosowana w sytuacji, w której dostęp do tajnych informacji mogą mieć tylko nieliczne osoby, a dodatkowo żadna z nich nie powinna tego dostępu uzyskać bez autoryzacji pozostałych.

3. Koncepcja rozwiązania problemu

3.1. Ogólny zarys protokołu



1. Rejestracja i uwierzytelnianie użytkowników (podprotokół nr 1).
2. Stworzenie przetargu (podprotokół nr 2).
3. Ogłoszenie przetargu (wiadomość jawna).
4. Składanie ofert przez uczestników (podprotokół nr 3).

Po punkcie czwartym kończy się czas na składanie ofert; po tym terminie GAP nie przyjmuje już więcej ofert.

5. Rozstrzygnięcie przetargu (podprotokół nr 4)
 - a. Wszyscy uczestnicy przesyłają część klucza, potrzebnego do odkodowania ofert (wcześniej GAP nie ma wglądu w ich treść).
 - b. GAP po odkodowaniu ofert przesyła je do organizatora przetargu.
 - c. Organizator przetargu rozpatruje oferty i przesyła informację o wyłonionym zwycięzcy do GAP.
6. GAP informuje uczestników o zwycięzcy (wiadomość jawna).

3.2. Szczegóły podprotokołów

3.2.1. Podprotokół nr 1 – rejestracja i uwierzytelnianie użytkowników

Użytkownik w celu zarejestrowania się w aplikacji powinien posiadać odpowiedni dokument D_c oraz prywatny klucz SK_{CCA} , pobrane z centrum autoryzacji. Za pomocą prywatnego klucza podpisuje dokument i koduje za pomocą klucza publicznego PK_{GAP} . Wynik wysyła do GAP.

GAP odkodowuje i weryfikuje dokument. W przypadku pozytywnej weryfikacji, generuje za pomocą generatora liczb losowych (KG) unikatowy numer rejestracyjny (UNR) dla danego użytkownika NR_c . UNR jest ważny tylko przez skończony czas, dlatego generowany jest również stempel czasowy T_{NRC} . GAP tworzy również prywatny klucz SK_c oraz publiczny klucz PK_c dla danego użytkownika, ważne do czasu kolejnej autoryzacji – będą one użyte w kolejnych podprotokołach. Ważność tych kluczy kończy się także wraz z czasem wynikającym ze stempla czasowego T_{NRC} . GAP podpisuje wygenerowane dane, koduje za pomocą klucza publicznego użytkownika i wysyła te dane do niego.

3.2.2. Podprotokół nr 2 – utworzenie przetargu

Użytkownik, który chce utworzyć przetarg (oznaczany dalej jako F), powinien posiadać numer rejestracyjny NR_F , jego stempel czasowy T_{NRF} , prywatny klucz SK_F oraz warunki przetargu WP_F . F generuje za pomocą generatora liczb losowych swój indywidualny numer N_F .

W pierwszym kroku F wysyła do GAP podpisane cyfrowo (SK_F) oraz zakodowane (PK_{GAP}) następujące informacje: swój numer rejestracyjny (NR_F), jego stempel czasowy (T_{NRF}), warunki przetargu (WP_F) oraz indywidualny numer N_F .

GAP weryfikuje numer rejestracyjny F (NR_F) oraz jego ważność czasową. W przypadku pozytywnej weryfikacji GAP generuje indywidualny numer przetargu (N_p) oraz parę kluczy dla tego przetargu (SK_p , PK_p). Prywatny klucz przetargu (SK_p) jest dzielony za pomocą schematu Shamira. Wykorzystano implementację w bibliotece *Crypto++*. Sekret jest dzielony na trzy części: dla F, dla GAP i dla uczestniczących w przetargu. Każda część jest niezbędna do reprodukcji klucza prywatnego (SK_p).

GAP wysyła cyfrowo podpisaną (SK_{GAP}) oraz zakodowaną (PK_F) część sekretu przeznaczoną dla F. Jednocześnie, GAP publicznie publikuje numer przetargu (N_p), jego warunki (WP_F) oraz jego publiczny klucz (PK_p).

3.2.3. Podprotokół nr 3 – ogłoszenie przetargu

Po założeniu i opublikowaniu informacji o przetargu, zainteresowane strony mogą składać swoje oferty. Uczestnik (B) chcący wziąć udział powinien posiadać numer rejestracyjny (NR_{O1}), prywatny klucz (SK_{O1}) oraz swoją ofertę (OF_{O1}). Użytkownik B generuje swój indywidualny numer (N_{O1}) i oznacza swoją ofertę stemplem czasowym (T_{O1}).

Następny krok składa się z wysłania do GAP cyfrowo podpisanych (SK_{O1}) oraz zakodowanych (PK_{GAP}) następujących informacji: N_p , N_o , NR_o , T_o . Oferta (OF_{O1}) jest także cyfrowo podpisana (SK_{O1}), ale zakodowana za pomocą klucza publicznego danego przetargu (PK_p) i jest później wysyłana do GAP.

Jeśli dane są poprawne, GAP odsyła użytkownikowi B potwierdzenie otrzymania oferty (W_{O1}). Potwierdzenie jest cyfrowo podpisane przez GAP (SK_{GAP}) oraz zakodowane (PK_{O1}).

3.2.4. Podprotokół nr 4 – rozstrzygnięcie przetargu

W ostatnim kroku, wykonywanym po upływie czasu na składanie ofert (jawnie podanym w warunkach przetargu), GAP ponownie dzieli fragment prywatnego klucza przetargu ($SK_{P(OF)}$) poprzez liczbę użytkowników biorących udział w przetargu za pomocą algorytmu podziału sekretu). Stworzone części są podpisywane cyfrowo (SK_{O1}), kodowane (PK_{O1}) i wysyłane do wszystkich uczestników.

Następnie F i wszyscy B wysyłają do GAP cyfrowo podpisane i zakodowane swoje części sekretu. W GAP zostają one złożone w główny sekret przetargu (SK_P). Wówczas można odkodować wszystkie oferty i wysłać je do F. Wszystkie oferty są wcześniej podpisane cyfrowo (SK_{GAP}) oraz zakodowane (PK_F).

Po otrzymaniu ofert F wybiera najlepszą ofertę i odsyła rezultat do GAP w celu poinformowania o zwycięzcy. Wysłana informacja zawiera: numer rejestracyjny zwycięzcy (NR_{O1}), swój numer rejestracyjny (NR_F), numery wszystkich użytkowników którzy byli brani pod uwagę (N_{ON}), swój indywidualny numer (N_F) oraz numer przetargu (N_P). Wymienione informacje są cyfrowo podpisane (SK_F) oraz zakodowane (PK_{GAP}).

Po otrzymaniu wyników GAP publicznie ogłasza zwycięzcę przetargu, jego ofertę oraz podaje do wiadomości wszystkich uczestników, których oferty były brane pod uwagę.

3.3. Koncepcja architektoniczna aplikacji

Napisaliśmy dwie aplikacje. Jedną bezobsługową – GAP oraz aplikację kliencką z graficznym GUI. Do realizacji funkcji stricte kryptograficznych zastosowaliśmy dwie biblioteki kryptograficzne – Botan do realizacji szyfrowania kluczem publicznym przy pomocy algorytmu RSA. Zastosowany klucz ma długość 2048bit. Drugą zastosowaną biblioteką kryptograficzną jest Crypto++. Użyto jej do realizacji algorytmu Shamira podziału sekretu. Oba programy zostały napisane w języku C++ z użyciem biblioteki Qt w wersji 5.5.

Z racji tego, że w szyfrowaniu RSA długość wiadomości nie może przekraczać długości klucza, przesyłane dane są dzielone na bloki odpowiedniej wielkości. Aplikacje napisane są zdarzeniowo – źródłem zdarzeń (w nomenklaturze Qt *signal*) są np. naciśnięcia przycisku w interfejsie klienta, timeout timera informującego o zakończeniu aukcji w serwerze, czy też przychodzące dane do bufora gniazda tcp (dotyczy obu aplikacji).

3.3.1. GAP

Aplikacja GAP odbiera odpowiednie polecenia od aplikacji klienckich – może to być logowanie, tworzenie nowej aukcji, czy dodawanie oferty do aukcji. GAP może obsługiwać dowolną liczbę użytkowników i aukcji jednocześnie. Użytkownik najpierw musi wykonać udane logowanie żeby móc wykonać inne operacje. Po zalogowaniu z użytkownikiem zostaje powiązana jest jedna instancja klasy Uzytkownik, która zawiera informacje związane z użytkownikiem oraz wskaźnik na gniazdo jego połączenia tcp. Użytkownicy z kolei są zgrupowani w kolekcji.

Podczas dodawania nowej aukcji tworzona jest nowa instancja klasy Aukcja, zawierająca informacje o aukcji. Ponadto ustawiany jest timer, który odlicza czas do zakończenia aukcji. Z aukcją powiązana jest kolekcja ofert, do której dokładane są obiekty podczas realizacji podprotokołu nr 3 – dodawanie oferty.

Po wywołaniu zdarzenia upłynięcia czasu aukcji (timeout odpowiedniego timera), GAP dokonuje podziału części klucza aukcji, przeznaczonego dla oferentów, na liczbę części równą liczbie uczestników, którzy złożyli oferty (przy czym minimalne uczestnictwo wynosi 2). Fragmenty przesyłane są do oferentów, następnie GAP czeka na ich powrót. W kolejnym kroku GAP żąda od firmy odesłania jej fragmentu klucza. Jeśli minimalne uczestnictwo jest spełnione, to odszyfrowuje oferty, jeśli nie – informuje wszystkich o nieważności przetargu. Jeśli aukcja jest ważna, oferty zostają odszyfrowane i przesłane do firmy. Po zwróceniu zwycięzcy przez firmę w sposób jawny informowani są uczestnicy o zwycięzcy. Natomiast sam obiekt zakończonej aukcji jest usuwany z serwera.

Aplikacja kliencka może pełnić rolę zarówno firmy, jak i oferenta. Użytkownik może przeglądać aukcje dostępne na serwerze i wziąć w wybranej udział lub założyć własną aukcję. W trakcie oczekiwania na rozstrzygnięcie przetargu klient nie może uczestniczyć w innych aukcjach. Początkowo wyświetlane jest okno logowania, gdzie po naciśnięciu przycisku *Zaloguj* wykonywany jest scenariusz podprotokołu 1 (logowanie).

Przy naciskaniu przycisku *Odśwież* pobierane są dostępne aukcje wraz parametrami z serwera protokołem jawnym i zapisywane w lokalnym wektorze. Po naciśnięciu na nazwę aukcji w celu podejrzenia jej szczegółów, dane są nie są pobierane z serwera, lecz są to dane pobrane wcześniej, przy pomocy przycisku *Odśwież*. Następnie po utworzeniu nowej aukcji lub dodaniu oferty, aplikacja przekierowuje zdarzenia pochodzące z połączenia tcp, do klasy która wykonuje scenariusz podprotokołu 4, w zależności, czy jest firmą, czy oferentem.

Po odebraniu informacji o zwycięzcy przetargu, lub odesłaniu zwycięskiej oferty, przyciski zostają odblokowane, a wewnętrznie aplikacja resetuje swój stan.

4. Zastosowania praktyczne implementowanego rozwiązania

Zaimplementowane rozwiązanie pozwala na przeprowadzenie prostego i bezpiecznego przetargu elektronicznego, który można rozstrzygnąć w krótkim okresie czasu (kilka-kilkadziesiąt minut). Zastosowane algorytmy i protokoły zapewniają bezpieczeństwo przesyłanych danych, niezaprzeczalność zarówno ofert jak i podmiotów je składających oraz anonimowość przy wyborze zwycięzcy. Rozwiązanie doskonale nadaje się do rozstrzygania przetargów na:

- zakup akcesoriów biurowych,
- zakup sprzętu elektronicznego,
- zakup podstawowego sprzętu medycznego dla placówek ochrony zdrowia,
- zakup wyposażenia placówek edukacyjnych, instytucji kultury i sztuki,
- obsługę linii autobusowych,
- obsługę cateringową,
- obsługę utrzymania porządku,

oraz inne, możliwe do szybkiej oceny i wyboru przedmioty lub usługi.

5. Instrukcja użytkowania programu

5.1. KeyGen

KeyGen jest bardzo prostym, dodatkowym programem do generowania kluczy. Jego uruchomienie powoduje stworzenie dwóch nowych kluczy RSA 2048bit, zapisanych do plików `Private_key_<obecna_data>` oraz `Public_key_<obecna_data>`. Nazwę klucza prywatnego należy zmienić na `skcca` i razem z kluczem publicznym GAPa przekazać użytkownikowi. Klucz prywatny należy nazwać `<nazwa_użytkownika>_PKcca` i skopiować do folderu z aplikacją.

Generowanie pary kluczy dla GAPa przebiega podobnie, z tym że klucze należy nazwać odpowiednio `gap` – jako publiczny klucz udostępniany klientom oraz `gapPriv` – jako prywatny klucz do umieszczenia w katalogu z aplikacją.

5.2. GAP

GAP odpowiada za bycie zaufaną trzecią stroną. Nie ma interfejsu okienkowego. Czuwa nad przebiegiem przetargu. Wystarczy jego uruchomienie. Aby dodać użytkownika, należy edytować plik `users.txt` i dodać nową linię w postaci `<nazwa_użytkownika>=<hasło_użytkownika>`.

5.3. eAukcja

5.3.1. Tworzenie konta

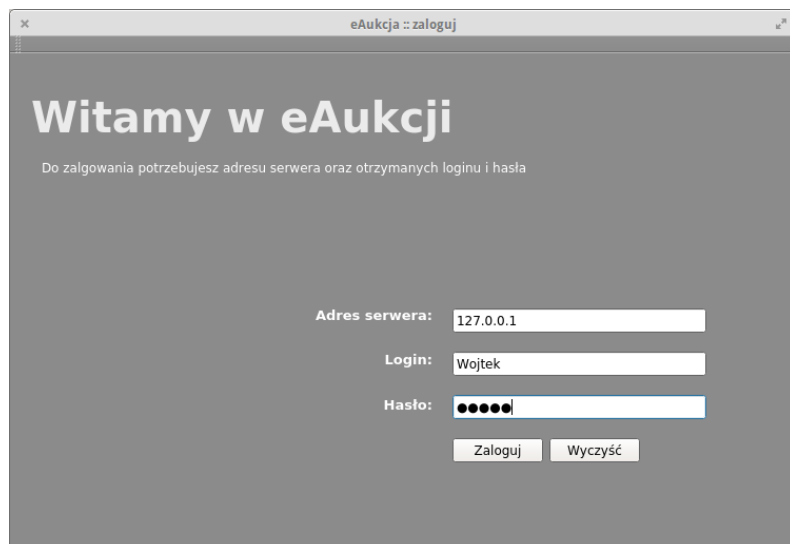
W celu stworzenia konta, które umożliwi dostęp do zasobów (przetargów) należy uzyskać od strony obsługującej GAPa login, hasło oraz klucze: publiczny i prywatny. Login i hasło zostaną użyte do identyfikacji, zaś klucze do uwierzytelnienia i bezpiecznego korzystania z aplikacji.

5.3.2. Uruchomienie aplikacji eAukcja

5.3.2.1. Przygotowanie

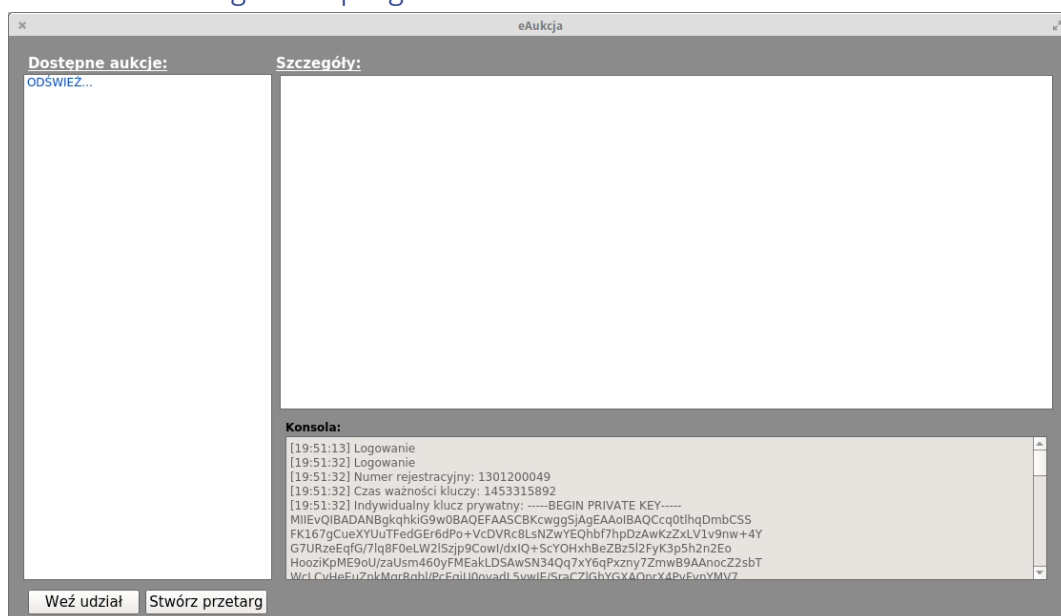
Uzyskane w podpunkcie 1 klucze (publiczny oraz prywatny) należy skopiować do katalogu z plikiem uruchomieniowym aplikacji. Następnie należy uruchomić aplikację eAukcja.

5.3.2.2. Logowanie



Po uruchomieniu aplikacji powita nas okno logowania. Do wypełnienia są trzy pola - adres serwera, login oraz hasło. W polu adres serwera należy podać adres IP serwera na którym został uruchomiony GAP. Pola login oraz hasło należy uzupełnić wcześniej uzyskanymi danymi. Jeśli któraś z podanych wartości będzie niepoprawna, aplikacja wyświetli komunikat informujący o błędzie. Jeśli wszystkie dane będą poprawne, aplikacja połączy się z GAPem, a użytkownik zostanie przeniesiony do głównego okna programu.

5.3.3. Okno główne programu

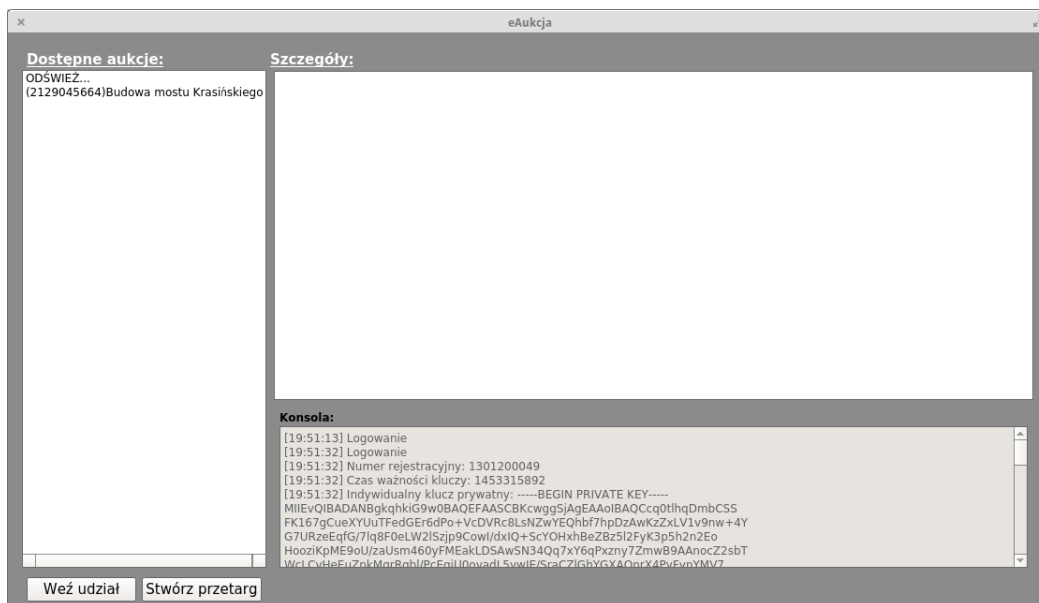


Poprawne zalogowanie skutkuje pobraniem z GAPa aktualnej listy aukcji. Główne okno programu składa się z trzech obszarów:

- **Dostępne aukcje** – pole pokazuje dostępne aukcje – pierwszy raz pobrane po zalogowaniu się; listę można odświeżyć dwukrotnie klikając pole *ODŚWIEŻ...*. Numer w nawiasie to numer aukcji, natomiast tekst za numerem to nazwa aukcji.
- **Szczegóły** – po dwukrotnym kliknięciu na wybraną aukcję w polu **Dostępne aukcje** jej detale (nazwa aukcji, opis aukcji, data zakończenia, kryteria oraz numer aukcji) pojawią się w tym polu tekstowym.
- Konsola – konsola pokazuje jakie informacje wymieniane są z serwerem.

Ponadto u dołu aplikacji wyświetlają się dwa przyciski. Z lewej strony *Weź udział* oraz *Stwórz przetarg*. Ich opis i efekty kliknięcia opisano w kolejnych podpunktach.

Po odświeżeniu listy aukcji aplikacja pyta GAPa o aktualny stan dostępnych aukcji. GAP zwraca bieżący stan.



5.3.4. Wzięcie udziału w istniejącym przetargu

The screenshot shows a web application window titled "Składanie oferty". Inside, the text "Składasz swoją ofertę w przetargu:" is followed by "Budowa mostu Krasinskiego" and "Data zakończenia: 20:01:2016 20:00:17". Below this is a table with two columns: "Cena" and "180 000 000 PLN". The table is mostly empty, with a large white rectangular area below the header row. At the bottom of the window are two buttons: "Złóż ofertę" and "Wróć".

Cena	180 000 000 PLN

Aby wziąć udział w przetargu, należy wybrać poprzez dwukrotne kliknięcie lewym przyciskiem myszy wybraną pozycję. W oknie **Szczegóły** zostaną wyświetlone wszystkie dostępne informacje o wybranym przetargu. Następnie należy nacisnąć przycisk *Weź udział*. Zostanie wyświetlone okno *Składanie oferty*. Informacja w górnej części okna zawiera nazwę aukcji i jej datę zakończenia. Niżej znajduje się okno z tabelą. Lewa kolumna tabeli to nazwa wymaganego przez organizatora kryterium. Prawa kolumna zawiera edytowalne komórki, w których należy wpisać własną ofertę. Nie jest możliwe złożenie oferty bez zaproponowania własnych wartości dla każdego kryterium – zostanie wówczas wyświetlony odpowiedni komunikat. Jeśli wszystkie pola są uzupełnione i chcemy wziąć udział w przetargu, należy nacisnąć przycisk *Złóż ofertę*. Aplikacja zwróci informację o statusie naszego żądania.

5.3.5. Stworzenie przetargu

Stwórz nowy przetarg

Podaj dane tworzonego przetargu:

Nazwa przetargu: Budowa mostu Krasieńskiego

Opis przetargu: Budowa przeprawy przez Wisłę z Targówka na Zoliborz. Kryterium 100% cena.

Data zakończenia: 20.01.2016 20:00

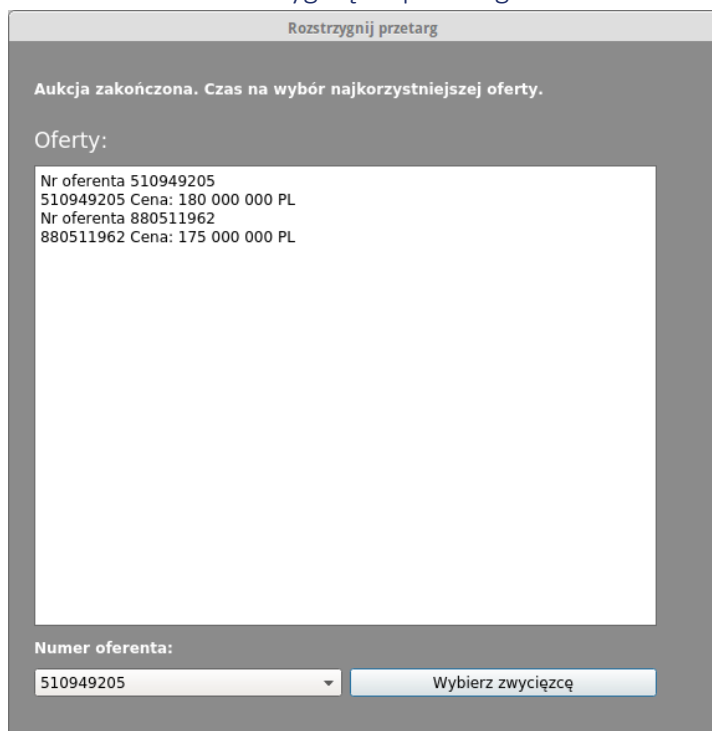
Kryteria przetargu: Cena

Dodaj nowe kryterium Usuń zaznaczone

Stwórz Wróć

Użytkownik może również założyć własny przetarg. Okno stworzenia nowego przetargu pojawi się po naciśnięciu przycisku *Stwórz przetarg* w oknie głównym. Zostanie wówczas wyświetlone okno jak na rysunku wyżej. Należy uzupełnić wszystkie pola, w tym dodać choć jedno kryterium. Po wybraniu przycisku *Stwórz* zostanie podjęta próba przekazania danych do GAPa. Informacja o powodzeniu bądź jego braku zostanie zwrócona użytkownikowi w formie okna dialogowego. Jeśli wystąpi niepowodzenie, użytkownik ma możliwość zmodyfikowania elementów składowych tworzonego przetargu (tj. dane z pól nie znikają, nie trzeba po raz kolejny ich uzupełniać).

5.3.5.1. Rozstrzygnięcie przetargu



Rozstrzygnij przetarg

Aukcja zakończona. Czas na wybór najkorzystniejszej oferty.

Oferty:

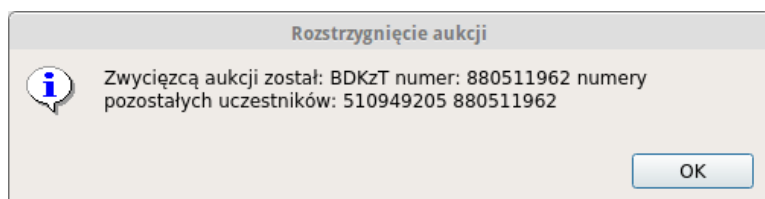
Nr oferenta 510949205
510949205 Cena: 180 000 000 PL
Nr oferenta 880511962
880511962 Cena: 175 000 000 PL

Numer oferenta:


510949205

Wybierz zwycięzcę

Po upływie czasu przewidzianego przez organizatora przetargu na składanie ofert nie jest możliwe złożenie oferty. Po tym czasie (z krótkim odstępem na weryfikację klucza) organizator przetargu powinien nacisnąć przycisk *Wybierz zwycięzcę*. Pojawi się wówczas okno dialogowe zawierające kolejne oferty (numer oferenta, numer oferty oraz odpowiedzi na zadane kryteria). Organizator wybiera numer zwycięskiego oferenta z rozwijanej listy na dole okna dialogowego i następnie przekazuje informację o zwycięzcy do serwera poprzez naciśnięcie przycisku *Wybierz zwycięzcę*. Jeśli operacja się powiedzie, wszyscy biorący udział w przetargu zostaną powiadomieni o rezultacie jego rozstrzygnięcia, jak na obrazku niżej:



Rozstrzygnięcie aukcji

 Zwycięzcą aukcji został: BDKzT numer: 880511962 numery pozostałych uczestników: 510949205 880511962

OK

6. Raport z testów aplikacji

W celu przetestowania aplikacji stworzono czterech użytkowników o nazwach Wojtek, BDKzT, Obserwator oraz UMWaw. Każdemu z użytkowników przypisano hasło, oraz parę kluczy. Przetestowanie aplikacji zakładało zaliczenie następujących etapów:

- Zalogowanie wszystkich użytkowników (wykonanie podprotokołu pierwszego).
- Stworzenie przetargu przez użytkownika UMWaw (podprotokół drugi).
- Wzięcie udziału w przetargu przez użytkowników Wojtek oraz BDKzT (podprotokół trzeci).
- Rozstrzygnięcie przetargu i poprawne poinformowanie wszystkich uczestników o wyniku (podprotokół czwarty).

Pierwszy krok aplikacji przebiegł bez zarzutu. Nie było możliwe podszycie się pod drugiego użytkownika ani również podanie błędnych danych. System zareagował tylko w przypadku podania poprawnych danych. Po zalogowaniu przydzielił każdemu zalogowanemu użytkownikowi jego własny dziesięciocyfrowy numer i czekał na dalsze działania.

W drugim kroku użytkownik UMWaw zażądał stworzenie nowej aukcji. Aplikacja użytkownika przeprowadziła wstępną walidację wprowadzonych danych i gdy stwierdziła ich poprawność, przekazała dane do stworzenia na przetargu po stronie GAPa. GAP przydzielił nowostworzonemu przetargowi numer oraz stempel czasowy.

Trzeci krok zakładał wzięcie udziału tylko dwóch użytkowników. Czwarty, Obserwator nie brał udziału w aukcji. Zabieg ten miał na celu sprawdzenie, czy GAP nie roześle wrażliwych informacji do wszystkich zalogowanych uczestników (powinien tylko do użytkowników biorących udział). Obydwaj przeznaczeni do złożenia ofert użytkownicy zrobili to, a GAP przydzielił każdemu numer rejestracyjny, stempel czasowy oferty oraz numer uczestnika.

Czwarty krok rozpoczyna się w momencie zakończenia czasu na składanie ofert. GAP dzieli wówczas pozostały fragment klucza i rozsyła go do wszystkich biorących udział w przetargu. Następnie czeka na odebranie go podpisanego od każdego uczestnika i dopiero wówczas odszyfrowuje oferty i przekazuje je do organizatora przetargu. Stąd wynika pewna, niewielka (rzędu kilkunastu-kilkudziesięciu sekund) zwłoka w pojawieniu się okna wyboru zwycięzcy. Użytkownik organizujący przetarg wybrał na podstawie anonimowego numeru oferenta zwycięską ofertę i odesłał wynik do GAPa. GAP powiadomił wszystkie strony biorące udział w przetargu o jego wyniku – natomiast użytkownik Obserwator nie dowiedział się o żadnych szczegółach.

7. Podsumowanie

Przetarg elektroniczny to efektywne i eleganckie narzędzie do usprawnienia realizacji procedury przetargowej. Z faktu zastosowania matematyki i dobrze skonstruowanych algorytmów oraz protokołów wynika duże bezpieczeństwo i poprawność przeprowadzanych akcji.

Sam protokół przeznaczony do zaimplementowania jest bezpieczny i sprawdzony. Został rozdzielony na cztery części w celu optymalizacji pracy. Część pierwsza odpowiada za rejestrację i uwierzytelnianie użytkowników, część druga – za stworzenie przetargu, trzecia – umożliwia jego ogłoszenie zaś czwarta – pomaga przetarg rozstrzygnąć.

Koncepcja architektoniczna zakłada rozbięcie programu na dziewięć komponentów. Komponent to abstrakcyjne pojęcie, które modeluje określoną funkcjonalność aplikacji. Kryje się pod tym jedna klasa lub ich zestaw, bądź też biblioteka, podprogram albo inny proces. Komponenty komunikują się ze sobą za pomocą interfejsów. Taki podział pozwala na łatwe modyfikacje poszczególnych elementów programu.

Funkcje F i B zostaną połączone w jednej aplikacji. Użytkownik będzie mógł wybrać, czy chce ogłosić przetarg czy też wziąć udział w jakimś. Za taką formą przemawia złożoność aplikacji – B potrzebuje 4 podprotokołów, F trzech, zatem byłoby bezsensowne rozbijanie tej całości na kilka różnych programów.

Bibliografia

1. **Karbowski Marcin.** *Podstawy Kryptografii*. Gliwice : Helion, 2015.
2. **Księżopolski Bogdan i Kotulski Zbigniew.** Cryptographic protocol for electronic auctions. *Annales UMCS, Informatica*,. 2004.
3. **Sadowy Jacek.** *Kryteria oceny ofert w postępowaniach o udzielenie zamówienia publicznego – przykłady i zastosowanie*. Warszawa : Wydział Wydawnictw i Poligrafii, 2011.
4. **Schneier Bruce.** *Kryptografia dla praktyków. Protokoły, algorytmy i programy źródłowe*. Warszawa : WN-T, 2002.