WFTIMS, Informatyka, studia dzienne, inż, I st. semestr V
Podstawy kryptografii r. ak. 2017/2018
Prowadzący: prof. dr hab. inż. Volodymyr Yemyets poniedziałek, 12:00

Szymon Adach
Łukasz Gołębiewski
Piotr Stańczyk

Kryptosystem ElGamal

1. Cel

Celem zadana jest implementacja kryptosystemu ElGamal poprzez umożliwienia wygenerowania poprawnych kluczy oraz przeprowadzenie za ich pomocą asymetrycznego szyfrowania wiadomości. Klucz publiczny używany jest zaszyfrowania wiadomości do postaci szyfrogramu, klucz prywatny używany jest do odszyfrowania szyfrogramu.

2. Działanie

Kryptosystem Elgamal jest zdefiniowany na grupie cyklicznej $\boldsymbol{G}$. Bezpieczeństwo kryptosystemu wynika z trudności obliczenia logarytmu dyskretnego zdefiniowanego w takiej grupie.

Generacja kluczy:

Należy najpierw zdefiniować cykliczną grupę G rzędu $\boldsymbol{p}$ z generatorem $\boldsymbol{g}$.
Jako p zostanie wybrana duża oraz bezpieczna liczba pierwsza. Mianem dużej liczby pierwszej można określić np. liczbę 512 bitową. Bezpieczna liczba pierwsza (*safe prime*) to liczba pierwsza spełniająca równanie $\boldsymbol{p = 2q + 1}$ gdzie $\boldsymbol{q}$ to liczba również liczba pierwsza.
Liczba pierwsza gwarantuje nam że będzie tylko jedno rozwiązanie u logarytmu dyskretnego (przy założeniu że u jest mniejsze od rzędu p, w przeciwnym przypadku rozwiązaniem będzie każda liczba $u + pn$, co wynika z Małego Twierdzenia Fermata). Wielkość liczby pierwszej zwiększa ilość możliwych reszt modulo co zwiększa trudność obliczenie logarytmu dyskretnego. Bezpieczna liczba pierwsza gwarantuje że $p - 1$ w rozkładzie na czynniki ma conajmnej jedną duża liczbę pierwszą czyli q. Pozwala to zabezpieczyć się przed poznaniem klucza prywatnego za pomocą redukcji Pohliga-Hellmana.
Generator g jest obliczany poprzez znalezienie dowolnego pierwiastka pierwotnego liczby p. Generator g grupy G to liczba należąca do tej grupy dla której każdy element tej grupy może zostać uzyskany poprzez $g^n \ (mod)\ p$. Aby uniemożliwić poznanie części bitów klucza prywatnego za pomocą symbolu Legendre'a oraz osiągnąć poprawność semantyczną (założenie Diffiego-Hellmana) przekształcamy generator na generator podgrupy rzędu q poprzez podniesienie pierwiastka pierwotnego do kwadratu modulo p.

Następnie wybierana jest losowo liczba $\boldsymbol{x}$ z przedziału $\{1, \ldots, q-1\}$. Obliczana jest wartość $\boldsymbol{h} \equiv g^x \pmod p$. Zbiór $\{p, g, h\}$ będzie stanowił klucz publiczny, natomiast zbiór $\{p, g, x\}$ to klucz prywatny.
Wartość h może być podana publicznie właśnie ze względu na wcześniej omawiany problem obliczania wartości logarytmu dyskretnego.

Szyfrowanie:

Szyfrowanie wiadomości $\boldsymbol{m}$ jest wykonywane za pomocą klucza publicznego.
- Wiadomość m jest kodowana jako element m' grupy G.
- Wybierane jest losowe $\boldsymbol{y}$ z przedziały $\{1, \ldots, p-1\}$ oraz obliczona wartość $\boldsymbol{c} \equiv g^y \pmod p$.
- Obliczana jest wartość $\boldsymbol{s} \equiv h^y \pmod p \equiv g^{xy} \pmod p$.
- Obliczana jest wartość $\boldsymbol{d} \equiv m's \pmod p$.
- Wysyłany jest szyfrogram jako para (c, d).

Deszyfrowanie:

Odszyfrowanie szyfrogramu czyli pary (c, d) wykonywane jest za pomocą klucza prywatnego.
- Obliczana jest wartość $s \equiv c^x \pmod p$.
- Obliczana jest wartość $m' \equiv ds^{-1} \pmod p$ a następnie m' jest kodowane z powrotem do m.

3. Implementacja

Implementacja algorytmu została wykonana w języku Java. Do obsługi typów większych niż 64 bity wykorzystywana jest klasa przechowująca liczbę w postaci tablicy cyfr. W najprostszej implementacji działania na takich liczbach są wykonywane są na poszczególnych cyfrach składowych — algorytmy przypominają liczenie ręczne na kartce. W naszej implementacji wykorzystywane są algorytmy optymalizujące np. algorytm Karatsuby dla operacji mnożenia czy algorytm Burnikela-Zieglera dla przeprowadzenia operacji dzielenia.

Przy kodowaniu czyli zamianie wiadomości na element grupy G, tekst dzielony jest na kilka wiadomości tak, aby wartość każdej wiadomości po mapowaniu na element G była mniejsza od rzędu grupy. Przy szyfrowaniu dla każdej wydzielonej części tekstu na nowo losowana jest wartość y oraz obliczana wartość c co dodatkowo pozwala poprawić bezpieczeństwo algorytmu. Dzięki temu że tekst przy kodowaniu zamieniany jest na tablicę bajtów można szyfrować dowolne znaki np. znaki polskie czy chińskie.

Przy losowaniu liczby pierwszej o podanej przez użytkownika ilości bitów wykorzystywany jest test pierwszości Solovaya-Strassena wykorzystujący symbol Jacobiego. Użytkownik podaje również liczbę iteracji dla których przeprowadzany jest test tj. ile razy dla kandydata na liczbę pierwszą losowana jest liczba względnie do niego pierwsza i dla takiej pary przeprowadzany jest kolejny test.

Program działa w dwóch trybach. Pierwszy to tryb szyfrowania: następuje generacja kluczy oraz odczytanie wiadomości z pliku, następnie klucze oraz szyfrogram są zapisywane do określonych

plików. Drugi tryb to tryb deszyfrowania: program odczytuje klucze oraz szyfrogram z podanych plików, następnie zapisuje odszyfrowaną wiadomość do pliku.

Argumenty uruchomienia programu dla trybu szyfrowania:

java -jar elgamal.jar encr {*ilość bitów liczby pierwszej do wygenerowania*} {*ilość iteracji w teście Solovaya-Strassena*} {*liczba reprezentująca w jakim systemie liczbowym zapisać szyfrogram*} {*ścieżka do jakiej zapisać klucz publiczny*} {ścieżka *do jakiej zapisać klucz prywatny*} {ścieżka *do jakiej zapisać szyfrogram*}

Argumenty uruchomienia programu dla trybu deszyfrowania:

java -jar elgamal.jar decr {ścieżka *do szyfrogramu}* {ścieżka *do klucza prywatnego*} {*liczba reprezentująca w jakim systemie liczbowym zapisano szyfrogram*} {ścieżka *do jakiej zapisać odszyfrowaną wiadomość*}

4. Dyskusja

Kryptosystem Elgamal jest jednym z najpopularniejszych obok RSA systemów szyfrowania asymetrycznego. Szyfrowanie asymetryczne umożliwia nam zaszyfrowanie wiadomości do adresata gdzie tylko adresat będzie mógł ją odczytać. Samo szyfrowanie oraz deszyfrowanie przebiega szybko nawet dla dużych liczb.

Wadą systemu jest jego deformowalność czyli możliwość zmiany szyfrogramu bez znajomości tekstu jawnego przez co może być podatny na atak z wybranym szyfrogramem. Dla przykładu dla nieznanej wiadomości *m* oraz jej szyfrogramu (*c*, *d*) można łatwo skonstruować poprawny szyfrogram postaci (*c*, *2d*) dla wiadomości *2m.*
Kolejną wadą może być czas generowania kluczy w szczególności dla bardzo dużych liczb np. czas znalezienia losowej 2048 bitowej bezpiecznej liczby pierwszej. Rozwiązaniem tego problemu może być jednak losowanie wartości z tablicy stałych.

5. Bibliografia

Taher ElGamal: *A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms* http://caislab.kaist.ac.kr/lecture/2010/spring/cs548/basic/B02.pdf

A. J. Menezes, P. C. van Oorschot, S. A. Vanstone: *Handbook of Applied Cryptography*

D. Boneh: *The decision Diffie-Hellman problem* http://crypto.stanford.edu/~dabo/abstracts/DDH.html