dr Ryszard Kossowski

Laboratorium 1 - wprowadzenie do kryptologii

1. Wstep

Celem laboratorium jest prezentacja podstawowych uslug ochrony informacji w przesyle w sieci. Sa to: poufnosc realizowana przy pomocy szyfrowania kluczem symetrycznym oraz uwierzytelnienie nadawcy realizowane przy pomocy podpisu cyfrowego. Realizacja uslugi poufnosci wymaga tez bezpiecznego transportu klucza – tutaj wykorzystujaca szyfr asymetryczny RSA. Usluga poufnosci wykorzystuje tutaj prosty kryptosystem klucza frazowego (key phrase cypher). Nie jest on praktycznie stosowany (lamie sie go bardzo prosto), ale tutaj, dla prezentacji idei tzw. systemu hybrydowego, jest wystarczajacy. W zastosowaniach profesjonalnych bedzie uzywany jeden ze znanych szyfrów – potrójny DES, IDEA lub najnowszy – Rijndael. W praktyce obecnej do transportu klucza symetrycznego jest wykorzystywany kryptosystem klucza publicznego RSA, czy tez system ElGamala. Klucz symetryczny tez moze byc ustalany np. w oparciu o system Diffi-Helmana. Do podpisu cyfrowego wykorzystuje sie tutaj system RSA, równiez stosowany w praktyce, ale coraz wieksza popularnoscia cieszy sie amerykanski standard podpisu DSS (Digital Signature Standard, oznaczany tez DSA – Digital Signature Algorithm) majacy te zalete, ze jest przeznaczony wylacznie do podpisywania a nigdy do szyfrowania. To, z zalozenia, zapewnia tzw. separacje kluczy czyli wymaganie rozdzielenia kluczy do szyfrowania od kluczy do podpisywania. W realizacji uwierzytelnienia jest tez zapewniana integralnosc informacji, tutaj symulowana przy pomocy CRC. W praktyce ta metoda jest niedopuszczalna ze wzgledu na bardzo latwa mozliwosc podrobienia kodu CRC, nawet 32-bitowego. W dzisiejszych systemach uzywa sie MD5, jeden ze skrótów informacji (hash code) lub amerykanski standard SHS-1 (Secure Hash Standard, oznaczany tez SHA - Secure Hash Algorithm), obowiazkowy w przypadku stosowania DSS do podpisu. Jesli dodatkowo wymaga sie uwierzytelnienia zródla informacji, stosowany jest algorytm MAC (Message Authentication Code) wymagajacy uzycia klucza.

Calosc jest podzielona na czesci. W pierwszej (rozdzial 2) jest krótki opis prostych programików pozwalajacych na zapoznanie sie z wybranymi metodami obliczen stosowanych w kryptosystemie klucza publicznego RSA. W czesci drugiej (rozdzial 3) jest zwiezly opis kryptosystemu klucza frazowego. Czesc trzecia (rozdzial 4) prezentuje sposób szyfrowania i odszyfrowywania wiadomosci z wykorzystaniem systemu hybrydowego – systemu, który stosuje szyfr symetryczny do szyfrowania wiadomosci, a szyfr klucza publicznego do transportu klucza. Ostatnia czesc, czwarta (rozdzial 5), pozwala na zapoznanie sie z idea realizacji podpisu cyfrowego zawierajacego usluge integralnosci informacji.

Prosze pamietac, ze prezentowany tutaj szyfr frazowy oraz algorytm CRC sa w dzisiejszej praktyce ochrony informacji niedopuszczalne ze wzgledu na ich slabosc – tutaj tylko ilustruja idee realizacji celu (szyfrowanie czy integralnosc). Równiez stosowane tutaj dlugosci kluczy sa absolutnie niewystarczajace. Prezentowane metody maja za zadanie zapoznanie sie z ich idea.

2. Szyfr asymetryczny – RSA

Operacje podstawowe

2.1 Program exp.exe

Program exp.exe demonstruje algorytm liczenia potegi x liczby a w arytmetyce modulo n.

Po uruchomieniu programu nalezy podawac odpowiednie wartosci **x**, **a** oraz **n**. Po podaniu ostatniej wartosci program wyswietli schemat blokowy liczenia

$$y = a^x \,(\mathrm{mod}\, n)$$

oraz przedstawi przyklad liczbowy dla podanych wartosci.

Prosze przestudiowac schemat blokowy i przyklad liczbowy.

2.2 Program odwr.exe

Program **odwr.exe** demonstruje algorytm liczenia odwrotnosci liczby **w** w arytmetyce **modulo p**.

Po uruchomieniu programu nalezy podawac odpowiednie wartosci **w** oraz **p**. Po podaniu ostatniej wartosci program wyswietli schemat blokowy liczenia

$$w^{-1} \textit{ takie,ze : } w * w^{-1} = 1 \,(\mathrm{mod}\, p)$$

oraz przedstawi przyklad liczbowy dla podanych wartosci.

Prosze przestudiowac schemat blokowy i przyklad liczbowy.

2.3 Program mulmod.exe

Program **mulmod.exe** wymaga przy wywolaniu podania trzech liczb. Pierwsza i druga sa czynnikami które nalezy pomnozyc a trzecia jest wartoscia **n** wzgledem której wyznacza sie **modulo n**. Wywolanie:

mulmod 37 49 131

oznacza zadanie wyznaczenia wartosci: 37*49 (mod 131).

2.4 Program nwd.exe

Program **nwd.exe** wymaga przy wywolaniu podania dwóch liczb których najwiekszy wspólny dzielnik jest wyliczany i podawany jako wynik. Przyklad wywolania:

nwd 325 45

Testy na pierwszosc

2.5 Program fermat.exe

Program **fermat.exe** przedstawia algorytm sprawdzania, czy podana liczba **n** jest liczba pseudopierwsza. Test ten jest testem probabilistycznym i wymaga podania ilosci prób **k**. Wywolanie

fermat

2.6 Program sol_stra.exe

program **sol_stra.exe** sluzy do probabilistycznego sprawdzania pierwszosci liczb metoda Solovay'a-Strassena. Opis tej metody mozna znalezc w MOV, Chap4, str. 137. Wywolanie:

sol_stra

Zbiór liczb pierwszych

Zbiór **primes.txt** zawiera tablice liczb pierwszych mniejszych od 150,000.

Algorytmy logarytmowania dyskretnego

2.7 Program log.exe

Program **log.exe** demonstruje algorytm liczenia logarytmu liczby **y** przy podstawie **a** w arytmetyce **modulo n**.

Po uruchomieniu programu nalezy podawac odpowiednie wartosci **y**, **a** oraz **n**. Po podaniu ostatniej wartosci program wyswietli schemat blokowy liczenia

$$x = \log_a y (\bmod n)$$

oraz przedstawi przyklad liczbowy dla podanych wartosci.

Prosze przestudiowac schemat blokowy i przyklad liczbowy.

2.8 Program logn.exe

Program **logn.exe** tabelaryzuje wartosci x

$$x = \log_a y (\bmod n)$$

dla y = 0,1, ... ,n-1 zaznaczajac strzalka te pozycje, dla których zachodzi równosc.

Po uruchomieniu programu nalezy podac odpowiednie wartosci dla **a**, **y** oraz **n**. Program po wyswietleniu tabelki czeka na nacisniecie dowolnego klawisza co powoduje wyswietlenie nastepnej linii tabeli. Klawisz PgDn powoduje wyswietlenie nastepnej strony tabelki.

2.9 Program shanks.exe

Program **shanks.exe** wyznacza logarytm dyskretny metoda Shanksa zwana tez metoda baby-step giant-step. Opis tej metody mozna znalezc w MOV*, Chap.3, str 104. Wywolanie:

shanks

Algorytmy faktoryzacji

2.10 Program pollard.exe

Program **pollard.exe** faktoryzuje liczby calkowite korzystajac z metody Pollarda rho. Opis tej metody mozna znalezc w MOV Chap3, str 91. Wywolanie:

pollard

2.11 Program pollard1.exe

Program **pollard1.exe** faktoryzuje liczby calkowite korzystajac z metody Pollarda p-1. Opis tej metody mozna znalezc w MOV, Chap3, str 93. W tej metodzie uzywa sie pojecia *gladkiego ograniczenia B*. Liczba calkowita n jest B-gladka lub gladka wzgledem ograniczenia B jesli wszystkie jej czynniki pierwsze sa ? B. Wywolanie:

Pollard1

2.12 Program qsf.exe

Program **qsf.exe** faktoryzuje liczby calkowite korzystajac z metody Quadratic sieve. Opis tej metody mozna znalezc w MOV, Chap3, str 95. W tej metodzie uzywa sie pojecia *bazy czynników – factor base*. Jest to wektor zaczynajacy sie od liczby –1 i zawierajacy liczby pierwsze spelniajace warunek by symbol Jacobiego liczby faktoryzowanej wzgledem tej liczby pierwszej byl równy 1. Wywolanie:

qsf

3. Szyfr symetryczny. Kryptosystem key-phrase

Kryptosystem key-phrase jest szyfrem monoalfabetycznym. Kluczem szyfrowania jest wybrana fraza (mozliwie dluga i z jak najmniej powtarzajacymi sie literami) oraz wybrana litera. Tablice do kodowania buduje sie nastepujaco:

?? W wierszu piszemy alfabet odpowiadajacy alfabetowi tekstu otwartego.
?? Pod nim podpisujemy, poczawszy od wybranej litery, wybrana fraze pomijajac powtarzajace sie litery.
?? Pozostale litery uzupelniamy literami brakujacymi.
?? Szyfrujemy zastepujac litery z pierwszego wiersza literami z drugiego wiersza.

Przyklad:
Fraza: **zyzio**, litera **f**.

a	b	c	d	e	**f**	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
T	U	V	W	X	**Z**	**Y**	**I**	**O**	A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q	R	S

* Alfred J. Menezes, Paul C. Van Oorschot, Scott A. Vanstone, “Hadbook of Applied Cryptography, CRC Presss

4. Usluga poufnosci. Kryptosystem hybrydowy

Przygotowanie – generacja kluczy

1) Wybrac dwie liczby **p** i **q** i korzystajac z testu Fermata **fermat.exe** lub testu Solovay'a-Strassena **sol_stra.exe** sprawdzic czy wybrane liczby sa pseudopierwsze. Jesli test wykaze, ze nie sa, wybrac nastepna az do otrzymania dwóch liczb pseudopierwszych.

2) Korzystajac z tablicy liczb pierwszych **primes.txt** sprawdzic czy uzyskane liczby pseudopierwsze sa liczbami pierwszymi.
Przyklad: p=17, q=23.

3) Korzystajac z programu **mulmod** przemnozyc otrzymane liczby pierwsze **p*q=n**. Mozna to zrobic w ten sposób, ze dobierze sie trzeci parametr tak by wynik nie byl wyznaczany (modulo trzeci parametr).
Przyklad: n=17*23 = 391.

4) Przy pomocy któregos programu do faktoryzacji (**pollard.exe, pollard1.exe, qsf.exe**) wyznaczyc czynniki wybranych liczb. Nalezy sobie zdac sprawe, ze odpornosc tego systemu kryptograficznego polega **wylacznie** na trudnosci faktoryzacji **duzych** liczb.

5) Wyznaczyc iloczyn **? (n)=(p-1)*(q-1)** korzystajac z programu **mulmod**.
Przyklad: ? (n)=16*22 = 352.

6) Losowo wybrac liczbe **e** wzglednie pierwsza wzgledem iloczynu **? (n)** która bedzie kluczem publicznym. Sprawdzenie mozna wykonac poslugujac sie programem **nwd.exe**.
Przyklad: e=9.

7) Korzystajac z programu **odwr** dla liczby **e** i powyzszego iloczynu ? (n) (liczby Eulera) wyznaczyc liczbe **d** która bedzie kluczem prywatnym.
Przyklad: d=313.

Szyfrowanie

8) Przyjawszy:

a b c d e f g h i j k l m
97 98 99 100 101 102 103 104 105 106 107 108 109
n o p q r s t u v w x y z
110 111 112 113 114 115 116 117 118 119 120 121 122
A B C D E F G H I J K L M
65 66 67 68 69 70 71 72 73 74 75 76 77
N O P Q R S T U V W X Y Z
78 79 80 81 82 83 84 85 86 87 88 89 90
0 1 2 3 4 5 6 7 8 9
48 49 50 51 52 53 54 55 56 57

zakodowac wybrana fraze(kod ASCII). Fraza ta wraz z wybrana litera bedzie kluczem sesyjnym dla programu szyfrowania **phrase.exe**.
Przyklad frazy: "zyzio" = 122, 121, 122, 105, 111, wybrana litera: “f” = 102.

9) Przygotowac wiadomosc, która bedzie bezpiecznie wyslana do kolezanki/kolegi. Mozna ja napisac korzystajac z opcji **F**ile-**N**ew programu **phrase.exe**.

10) Korzystajac z programu **phrase.exe** zaszyfrowac wiadomosc **kluczem sesyjnym**. Program **phrase** szyfruje plik szyfrem monoalfabetycznym.

11) Pobrac od kolezanki/kolegi jego klucz publiczny czyli jego **e** oraz **n**.

12) Korzystajac z programu **exp** przy pomocy klucza publicznego kolegi **e** i **n** zaszyfrowac klucz sesyjny.
Przyklad (dla klucza publicznego e=9 i n=391):
337, 223, 337, 371, 332, 204.

13) Przekazac szyfrogram kolezance/koledze wraz z zaszyfrowanym kluczem sesyjnym

. **Odszyfrowanie**

14) Po otrzymaniu szyfrogramu zaszyfrowanego Twoim kluczem publicznym, poslugujac sie programem **exp.exe** oraz swoim kluczem prywatnym **d** i swoja wartoscia **n**
?? odszyfrowac otrzymany klucz sesyjny,
?? odkodowac go (uwaga: ostatnia liczba odpowiada wybranej literze klucza),
?? a nastepnie odszyfrowac wiadomosc korzystajac z programu **phrase.exe**.

15) Poniewaz proces odszyfrowania odbywa sie przez obliczenie wartosci

$$m = c^d \,(\mathrm{mod}\, n)$$

dla danych wartosci **c**, **n** oraz klucza prywatnego **d,** stad wynika, ze jesli analityk zna pare: wiadomosc **m,** szyfrogram **c** oraz liczbe **n** to ze wzoru

$$d = \log_c m \,(\mathrm{mod}\, n)$$

moze wyznaczyc wartosc klucza prywatnego **d**.

Poslugujac sie programem **log.exe** wyznaczyc dla danej pary wiadomosc - szyfrogram wartosc klucza prywatnego.
Przyklad: dla m=122 c=337 n=391 otrzymujemy d=137.

Prosze tez spróbowac wyznaczyc te wartosc logarytmu poslugujac sie programem **shanks.exe**.

Poslugujac sie programem **logn.exe** wyznaczyc dla danej pary wiadomosc - szyfrogram wartosci klucza prywatnego. Moze byc ich kilka, ale tylko jedna wartosc spelnia wymaganie **e*d = 1 (mod ? (n))**.

Przyklad: dla m=122 c=337 n=391 otrzymujemy dwie wartosci d_1=137 oraz d_2=313.

5. Podpis cyfrowy. CRC.

5.1 CRC – Cyclic redundancy Check.
Prosze uruchomic program **crc_z.exe**. Demostruje on zasade dzialania kodu CRC.

5.2 Wyznaczanie CRC
Do wyznaczania 24-bitowej wartosci CRC dla danego pliku posluzymy sie programem **crc.exe**. W tym celu prosze najpierw utworzyc plik, np. o nazwie ala.txt zawierajacy tekst "ala ma kota" (uwaga: bez nowej linii na koncu!). Nastepnie prosze uruchomic program **crc.exe** poleceniem:

crc ala.txt

Pojawi sie wynik:

CRC pliku ala.txt: 48F969 (hex) 4782441 (dec)

Prosze spróbowac zmienic zawartosc pliku ala.txt np. przez dodanie nowej linii po slowie "kota" i ponownie wyznaczyc CRC dla tego pliku.

5.3 Podpisywanie wiadomosci z wykorzystaniem RSA

1) Dla poprzednio wyslanej wiadomosci do kolezanki/kolegi prosze wyznaczyc wartosc CRC pliku, w którym znajdowala sie ta wiadomosc.

2) Bedziemy teraz podpisywali te wiadomosc **naszym kluczem prywatnym**. Niech on bedzie równy**: d=313, n=391**.

3) Dziesiatkowa wartosc CRC podzielimy na grupy liczb mniejszych od naszego **n**, np. CRC pliku ala.txt podzielimy:

4782441 ? **4 – 78 – 24 – 41**.

4) Korzystajac z programu **exp.exe** przy pomocy naszego klucza prywatnego **d** i **n** zaszyfrowac otrzymany ciag.
Przyklad (dla klucza prywatnego d=313 i n=391):

242 – 330 – 231 - 95.

Otrzymany ciag jest naszym podpisem cyfrowym naszej wiadomosci.

5) Przekazac szyfrogram CRC kolezance/koledze.

5.4 Weryfikacja podpisanej wiadomosci

Po odebraniu od kolezanki/kolegi szyfrogramu CRC wiadomosci jaka przedtem odebralismy, korzystajac z klucza publicznego kolezanki/kolegi

?? odszyfrowujemy kod CRC,
?? nastepnie wyznaczamy przy pomocy programu **crc.exe** wartosc kodu CRC pliku z wiadomoscia
?? i te dwie wartosci porównujemy.
?? Jesli sa one równe to uznajemy wiadomosc za wiarygodna.