Politechnika Wrocławska, Informatyka Stosowana

JEDNOKIERUNKOWE FUNKCJE SKRÓTU ALGORYTMY WYMIANY KLUCZA

Cyberbezpieczeństwo, Laboratorium nr.6 - raport

Autor: Aleksander Stepaniuk
Nr. Indeksu: 272644

Zad 1. Jednokierunkowe funkcje skrótu

Teksty których użyłem do analizy:

Tekst 1: (plik nieszkodliwy)

„”

[Ta wiadomosc zostala wyslana w formie jakiegos pliku]

wstep bedzie wiec taki sam dla tych plikow

Jeszcze gdy chodzilem do podstawowki, to byl tam taki Pawel, i ja jechalem na rowerze, i go spotkalem, i potem jeszcze pojechalem do biedronki na lody, i po drodze do domu wtedy jeszcze, juz do domu pojechalem

Z wyrazami szacunku,

Jan Kowalski

„”

Tekst 2: (plik niebezpieczny)

„”

[Ta wiadomosc zostala wyslana w formie jakiegos pliku]

wstep bedzie wiec taki sam dla tych plikow

Cesarz czesal wlosy cesarzowej cesarzowa czesala wlosy cesarza Dzdzystym rankiem gzegzolki i piegze zamiast wziac sie za dzdzownice nazarly sie na czczo miazszu rzezuchy i rzedem rzygaly do rozzarzonej brytfanny Idzie Sasza sucha szosa suszy sobie swoje szorty Gdzie jest kufel pyta brat Moze kufel w kufer wpadl Bracie zawsze ci tlumacze kufel wpadl do kufra raczej Wyjmij z kufra kufel bracie lepiej postaw go na blacie

Z wyrazami szacunku,

Jan Kowalski

„”

Zadanie 1.1-1.2;

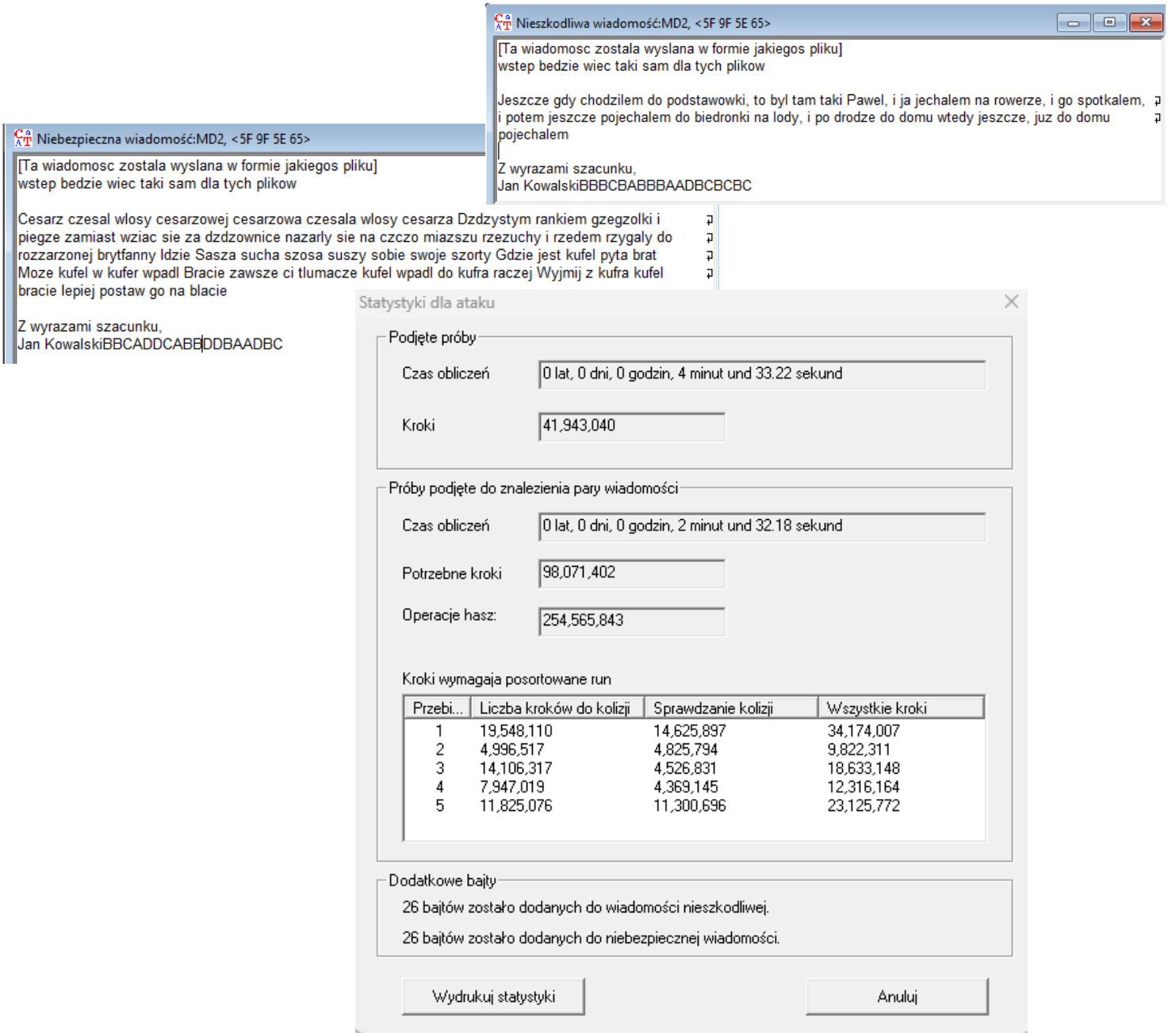
Przeprowadzono eksperymenty dla różnych wartości funkcji skrótu, liczby wspólnych bitów

Pytanie 1.3;

Wzrost czasu realizacji ataku wraz ze wzrostem wartości parametru opisującego długość ciągu bitów							
	8 bitów	16 bitów	32 bitów	48 bitów	64 bitów	128 bitów	160 bitów
MD2	0,1s	0,1s	16s	1h 22min	14,8 dni	10^94 lat	X
MD4	0,1s	0,1s	2s	3min 59s	5h 10min	10^94 lat	X
MD5	0,1s	0,1s	1s	4min 33s	5h 11min	10^94 lat	X
SHA	0,1s	0,1s	2s	1min 50s	8h 5min	10^94 lat	10^94 lat
SHA-1	0,1s	0,1s	2s	1min 3s	4h 40min	10^94 lat	10^94 lat
RIPEMD-160	0,1s	0,1s	3s	3min 4s	13h 46min	10^94 lat	10^94 lat

W miarę wzrostu liczby wspólnych bitów w plikach, czas realizacji ataku na funkcje hash rósł. Dla funkcji o silniejszych właściwościach kolizyjnych(bardziej odpornych na ataki kolizyjne), takich jak RIPEMD-160 lub SHA-256, czas ataku może drastycznie wzrastać, podczas gdy dla MD5 czy SHA-1, czas ataku rośnie wolniej. Z nieznanego mi powodu czas dla MD2 rósł najszybciej.

Im dłuższy jest ustalony wspólny ciąg bitów, tym trudniej (i dłużej) jest przeprowadzić skuteczny atak, ponieważ liczba możliwych kolizji maleje. Czas ataku rośnie wykładniczo razem ze wzrostem długości parametru długości bitów.



Pytanie 1.4;

Wybór funkcji skrótu ma wpływ na czas realizacji poszukiwania kolizji. Najdłuższy czas osiągnęły MD2 oraz RIPEMD-160. SHA osiągnęło krótszy czas, a MD4, MD5 oraz SHA-1 osiągnęły najkrótszy czas.

Pytanie 1.5;

Przewaga modyfikowania dwóch dokumentów jest możliwa dzięki tzw. „birthday paradox" i opiera się na tym, że potrzebujemy jakiejkolwiek pary równych haszy ze zbioru, a nie jakiegoś konkretnego:

- W przypadku ataku z modyfikacją jednego dokumentu, aby znaleźć kolizję musimy znaleźć konkretnie dopasowany hasz wygenerowany przez funkcję skrótu. Prawdopodobieństwo znalezienia takiej kolizji to $1/2^n$.

- W przypadku ataku z modyfikacją dwóch dokumentów wystarczy znaleźć dwa identyczne hasze, niezależnie jakie. Ważne jest jedynie to, aby dla obu dokumentów powstał ten sam skrót. Zwiększa to prawdopodobieństwo kolizji do $1/2^{n/2}$.

Prawdopodobieństwo rośnie więc znacząco z każdym kolejnym dokumentem.

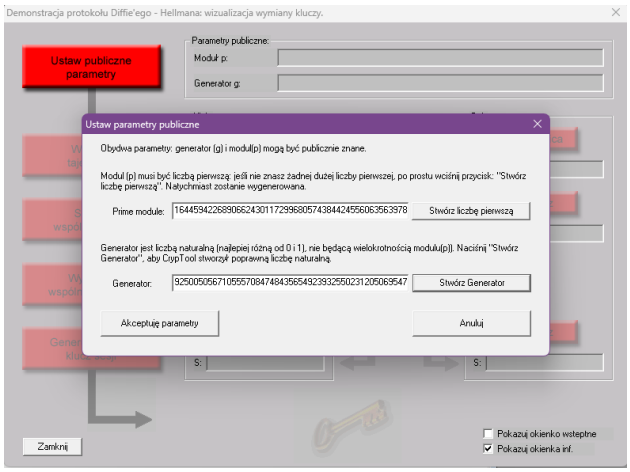
Pytanie 1.6;

Starsze funkcje skrótu takie jak MD5 czy SHA-1, są podatne na ataki kolizyjne, przez co nie zapewniają już wystarczającego bezpieczeństwa. Oznacza to, że nie można im w pełni ufać tym funkcjom dla zastosowań krytycznych, takie jak podpisy cyfrowe czy certyfikaty. Nowsze funkcje skrótu, takie jak SHA-256 czy RIPEMD-160 są bardziej odporne na kolizje, więc to one są zalecane do stosowania w miejscach wymagających wysokiego poziomu bezpieczeństwa.

Pytanie 1.7;

MD5 i SHA są podatne na ataki kolizyjne, z uwagi na małą długość generowanego skrótu (128/160bitów), co umożliwia znalezienie dwóch różnych fragmentów danych o tym samym hashu, co zagraża bezpieczeństwu podpisów cyfrowych. Ich złamanie stało się łatwiejsze przy obecnych mocach obliczeniowych. Dodatkowo odkryto przypadki praktycznego wykorzystania tych podatności, takie jak np. tworzenie fałszywych certyfikatów SSL z identycznymi hashami poprzez lukę w MD5, co poważnie zagrażało bezpieczeństwu w sieci. Ataki, umożliwiły także przechwytywanie i modyfikację danych wrażliwych w transakcjach finansowych oraz systemach autoryzacyjnych. Ze względu na te zagrożenia, wiele instytucji i standardów międzynarodowych zaleca stosowanie bardziej zaawansowanych funkcji haszujących, takich jak SHA-3 lub SHA-256.

Zad 2. Wymiana klucza kryptograficznego



Obydwa parametry: generator (g) i modul(p) mogą być publicznie znane.

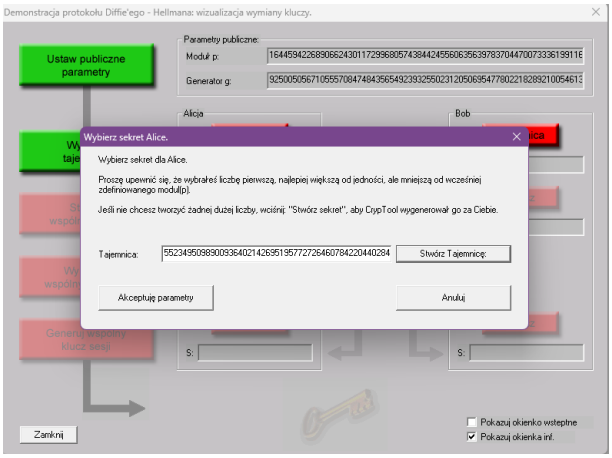
Modul (p) musi być liczbą pierwszą: jeśli nie znasz żadnej dużej liczby pierwszej, po prostu wciśnij przycisk: "Stwórz liczbę pierwszą". Natychmiast zostanie wygenerowana.

Prime module: 16445942268906624301172996805743844245560635639783

Generator jest liczbą naturalną (najlepiej różną od 0 i 1), nie będącą wielokrotnością modulu(p)). Naciśnij "Stwórz Generator", aby CrypTool stworzył poprawną liczbę naturalną.

Generator: 925005056710555708474843565492393255023120506954

☐ Pokazuj okienko wsteptne
☑ Pokazuj okienka inf.



Moduł p: 1644594226890662430117299680574384424556063563978370447007333619911

Generator g: 9250050567105557084748435654923932550231205069547780221828921005461

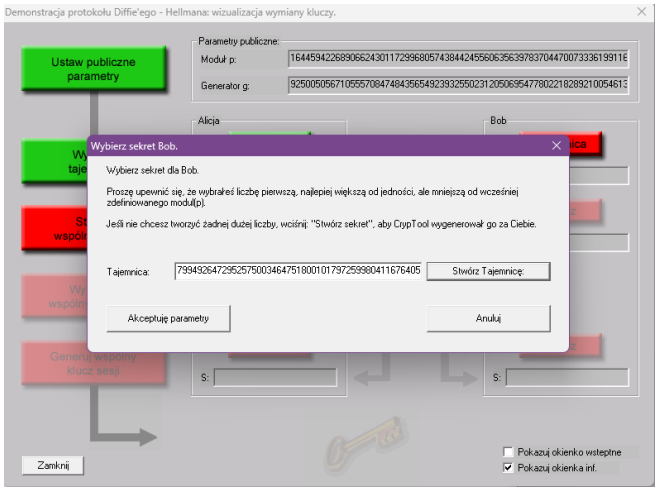
Wybierz sekret dla Alice.

Proszę upewnić się, że wybrałeś liczbę pierwszą, najlepiej większą od jedności, ale mniejszą od wcześniej zdefiniowanego modul(p).

Jeśli nie chcesz tworzyć żadnej dużej liczby, wciśnij: "Stwórz sekret", aby CrypTool wygenerował go za Ciebie.

Tajemnica: 5523495098900936402142695195772726460784220440284

☐ Pokazuj okienko wsteptne
☑ Pokazuj okienka inf.



Moduł p: 1644594226890662430117299680574384424556063563978370447007333619911

Generator g: 9250050567105557084748435654923932550231205069547780221828921005461

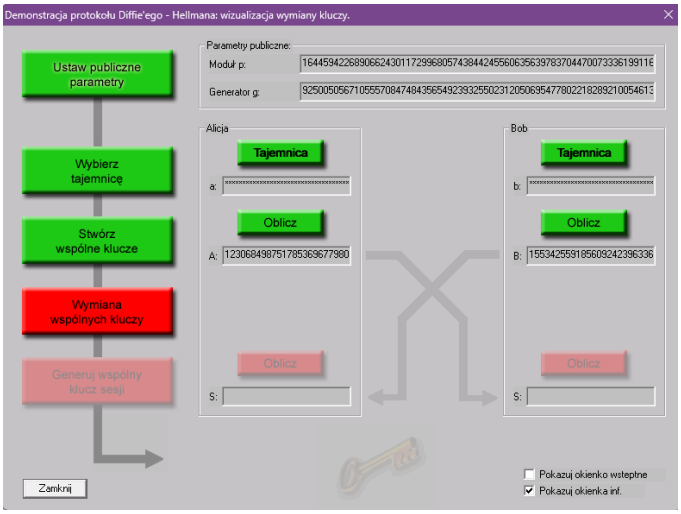
Wybierz sekret dla Bob.

Proszę upewnić się, że wybrałeś liczbę pierwszą, najlepiej większą od jedności, ale mniejszą od wcześniej zdefiniowanego modul(p).

Jeśli nie chcesz tworzyć żadnej dużej liczby, wciśnij: "Stwórz sekret", aby CrypTool wygenerował go za Ciebie.

Tajemnica: 7994926472952575003464751800101797259980411676405

☐ Pokazuj okienko wsteptne
☑ Pokazuj okienka inf.



Moduł p: 1644594226890662430117299680574384424556063563978370447007333619911

Generator g: 9250050567105557084748435654923932550231205069547780221828921005461

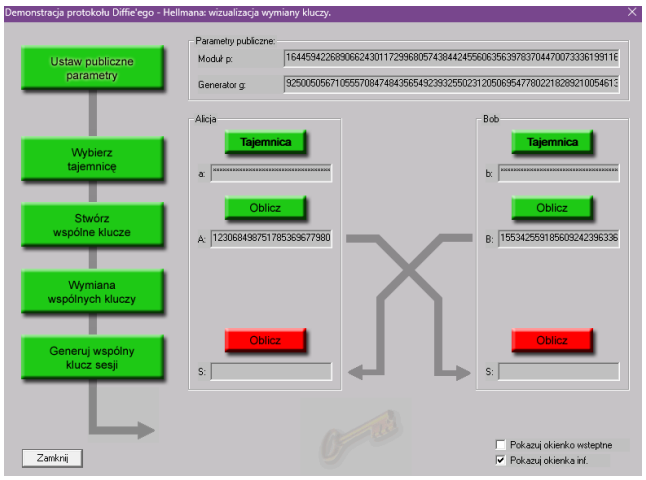
a: ************************************

A: 123068498751785369677980

b: ************************************

B: 155342559185609242396336

☐ Pokazuj okienko wsteptne
☑ Pokazuj okienka inf.



Moduł p: 1644594226890662430117299680574384424556063563978370447007333619911

Generator g: 9250050567105557084748435654923932550231205069547780221828921005461

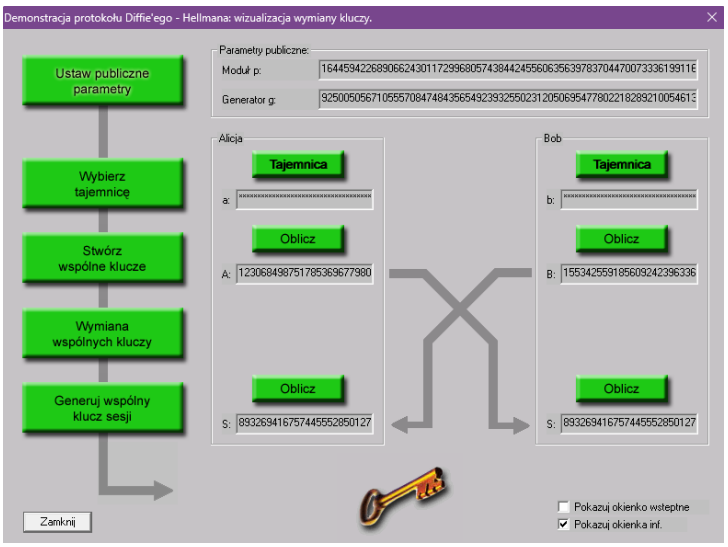
a: ************************************

A: 123068498751785369677980

b: ************************************

B: 155342559185609242396336

☐ Pokazuj okienko wsteptne
☑ Pokazuj okienka inf.



Moduł p: 1644594226890662430117299680574384424556063563978370447007333619911

Generator g: 9250050567105557084748435654923932550231205069547780221828921005461

a: ************************************

A: 123068498751785369677980

S: 893269416757445552850127

b: ************************************

B: 155342559185609242396336

S: 893269416757445552850127

☐ Pokazuj okienko wsteptne
☑ Pokazuj okienka inf.

a)

```
File  Actions  Edit  View  Help
┌──(stud㉿kali-vm)-[~]
└─$ openssl genpkey -genparam -algorithm DH -out dh_com_pub.pem
............+................................................................
.....+.......................................................................
.............................................................................
...............................................+................+............
.............................................................................
.............................................................................
.+....................+......................................+...............
..............+.......+......................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.+.................+.........................................................
......++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++
*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*+
+*++*++*++*++*++*++*++*++*++*++*++*++*++*++*

┌──(stud㉿kali-vm)-[~]
└─$ █
```

b)

```
┌──(stud㉿kali-vm)-[~]
└─$ cat dh_com_pub.pem
-----BEGIN DH PARAMETERS-----
MIIBDAKCAQEA0SUyuZN/YC1s4XuaqLPNZRQ4NWdlvv5sYFDqAiQE8rPGWLwfezB6
nCHBF62DV957syGjOu1Jlz7M9ogjQF4XV2+LG/0UUyc8rl+mNoQACNPmg9tAjWoz
BLnKr1EWxNRdGX9sVlSpFWYAcizHrKI30oECwxM+INfEUv3u8tU0zrTjkRl+14NY
IECzXVwcksR7IZLYRPTxtqHCUF7IdB8MlEsY7eS9d/c7cVlVzO6igR32fqMNUa7B
FYTKfwBsi3KhSJdnWaaLah1r1RvBLPfDHomswlc4vqlebtFhSoGNf2vFpa9cDFgk
PGXXDAqWCV8a5iIz/pTVYAQottras4U5BwIBAgICAOE=
-----END DH PARAMETERS-----
```

```
┌──(stud㉿kali-vm)-[~]
└─$ openssl pkeyparam -in dh_com_pub.pem -text
-----BEGIN DH PARAMETERS-----
MIIBDAKCAQEA0SUyuZN/YC1s4XuaqLPNZRQ4NWdlvv5sYFDqAiQE8rPGWLwfezB6
nCHBF62DV957syGjOu1Jlz7M9ogjQF4XV2+LG/0UUyc8rl+mNoQACNPmg9tAjWoz
BLnKr1EWxNRdGX9sVlSpFWYAcizHrKI30oECwxM+INfEUv3u8tU0zrTjkRl+14NY
IECzXVwcksR7IZLYRPTxtqHCUF7IdB8MlEsY7eS9d/c7cVlVzO6igR32fqMNUa7B
FYTKfwBsi3KhSJdnWaaLah1r1RvBLPfDHomswlc4vqlebtFhSoGNf2vFpa9cDFgk
PGXXDAqWCV8a5iIz/pTVYAQottras4U5BwIBAgICAOE=
-----END DH PARAMETERS-----
DH Parameters: (2048 bit)
P:
    00:d1:25:32:b9:93:7f:60:2d:6c:e1:7b:9a:a8:b3:
    cd:65:14:38:35:67:65:be:fe:6c:60:50:ea:02:24:
    04:f2:b3:c6:58:bc:1f:7b:30:7a:9c:21:c1:17:ad:
    83:57:de:7b:b3:21:a3:3a:ed:49:97:3e:cc:f6:88:
    23:40:5e:17:57:6f:8b:1b:fd:14:53:27:3c:ae:5f:
    a6:36:84:00:08:d3:e6:83:db:40:8d:6a:33:04:b9:
    ca:af:51:16:c4:d4:5d:19:7f:6c:56:54:a9:15:66:
    00:72:2c:c7:ac:a2:37:d2:81:02:c3:13:3e:20:d7:
    c4:52:fd:ee:f2:d5:34:ce:b4:e3:91:19:7e:d7:83:
    58:20:40:b3:5d:5c:1c:92:c4:7b:21:92:d8:44:f4:
    f1:b6:a1:c2:50:5e:c8:74:1f:0c:94:4b:18:ed:e4:
    bd:77:f7:3b:71:59:55:cc:ee:a2:81:1d:f6:7e:a3:
    0d:51:ae:c1:15:84:ca:7f:00:6c:8b:72:a1:48:97:
    67:59:a6:8b:6a:1d:6b:d5:1b:c1:2c:f7:c3:1e:89:
    ac:c2:57:38:be:a9:5e:6e:d1:61:4a:81:8d:7f:6b:
```

c1)

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ openssl genpkey -paramfile dh_com_pub.pem -out dh_client1_key.pem

┌──(stud㉿kali-vm)-[~/exchange]
└─$ ls
dh_client1_key.pem  dh_com_pub.pem
```

```
student@student-ubuntu:~/exchange$ openssl genpkey -paramfile dh_com_pub.pem -ou
t dh_client2_key.pem
student@student-ubuntu:~/exchange$ ls
dh_client1_key.pem  dh_client2_key.pem  dh_com_pub.pem
student@student-ubuntu:~/exchange$
```

c2)

```
student@student-ubuntu:~/exchange$ openssl pkey -in dh_client2_key.pem -text -no
out
DH Private-Key: (2048 bit)
private-key:
    58:f7:30:cf:33:34:f3:69:2e:1a:22:da:fb:1f:1f:
    bb:db:65:81:71:a9:93:c2:6e:dc:aa:40:d6:d4:9f:
    59:f4:59:f8:c0:60:9a:56:32:ff:f2:87:c3:fb:73:
    7d:d6:8f:64:53:4f:00:f9:0b:f2:51:59:3a:d2:23:
    01:66:73:d0:6a:61:cd:ee:f9:90:6f:68:93:2b:39:
    f5:e9:42:19:98:06:bf:53:0a:2c:69:36:42:1d:cd:
    1c:a5:aa:b1:1b:f4:b6:7b:69:d4:26:e6:b3:68:bd:
    26:65:1d:57:f7:86:ac:5a:6f:cc:cb:03:50:bd:0e:
    9b:a6:73:92:da:00:c1:52:d4:72:dc:8d:03:00:6b:
    24:99:80:22:02:c2:3c:8b:c2:ca:4d:7c:dd:0b:ef:
    9e:01:c6:e2:6a:c8:21:c4:e4:51:3c:31:ac:8a:d0:
    2a:3e:d1:02:7c:35:f4:1b:25:76:3d:c6:e3:54:67:
    f9:91:3d:7d:40:52:be:a9:ed:a2:47:f2:90:86:ae:
    92:dc:d2:82:2f:e6:88:c6:4f:ba:c9:a1:3d:74:6a:
    f5:3c:8f:6a:46:be:af:4b:e4:26:fb:aa:70:97:66:
    92:4a:1b:63:df:54:3d:a2:f5:14:e4:ce:4d:6a:8b:
    6e:e9:4c:98:2a:68:e8:bd:b9:4d:f9:2a:48:bb:63:
    83
public-key:
    23:08:c4:48:e2:d1:bf:5a:0d:e9:3a:a6:e1:97:29:
    d6:98:98:e6:45:70:0f:89:01:2c:e5:be:07:fe:8b:
    1f:e0:af:f6:1b:86:47:f4:57:b5:68:09:dc:67:df:
    cf:8a:a3:f5:8b:94:9e:c6:92:09:ed:80:03:92:e2:
    ad:5b:ab:01:b6:dc:3c:74:d7:db:00:6a:b8:23:cf:
    d2:ea:64:23:c2:c1:dc:05:d9:0f:c5:41:b4:bb:88:
    e9:20:90:08:06:3e:e7:28:13:c0:15:16:b4:69:4d:
    e7:4f:e4:79:f4:07:63:bb:71:d7:e1:cc:4a:91:55:
    57:b1:2c:64:73:d1:a4:20:38:52:67:87:e7:29:f4:
    59:17:bb:15:7b:90:56:51:fa:20:08:4d:51:4b:79:
    0d:8c:1d:29:ac:7b:ed:79:31:81:b4:ee:f2:69:61:
    f4:67:9c:a5:b9:d6:2c:f5:4d:cf:49:53:c0:79:18:
    58:44:a6:8e:83:5b:e9:3d:67:f2:34:4c:ef:e7:6c:
    5a:8d:2d:51:43:4b:4c:e1:e2:a2:54:3f:aa:45:b8:
    bf:7c:9c:6b:1f:eb:90:15:4c:58:00:eb:28:da:e5:
    d8:c9:31:c5:be:6c:e7:29:89:4f:42:b9:ea:c1:2b:
    b8:ca:63:c8:72:0c:8d:c0:8b:44:f8:b4:9a:ea:19:
    9d
P:
    00:98:75:4e:5d:94:a9:f4:ad:da:0f:8c:c0:bb:1f:
    c4:91:e3:21:b5:18:99:1d:87:00:c0:b3:e0:ca:32:
    ac:8d:f4:bf:8c:31:9e:e7:da:a1:39:ac:06:01:0b:
    a3:7d:02:63:80:0a:05:6a:df:87:0b:3e:1b:4a:cf:
    0a:12:d7:16:b5:71:6f:ba:bc:ce:d6:8e:b2:13:e0:
    1c:59:3d:2e:6c:79:ef:3e:b8:42:ef:b6:dd:37:db:
    1e:1e:21:e7:66:22:82:4e:af:c8:51:12:42:f0:0a:
    f3:93:88:83:d7:d6:2d:a3:45:0f:8e:93:e6:e7:a3:
    07:eb:b2:2c:b2:e6:32:b2:c4:fc:52:b0:7c:49:7b:
    ce:94:9b:0c:29:2a:ac:de:67:20:04:e6:9c:48:62:
    44:b7:a2:bc:52:1c:2f:54:38:d9:12:59:0c:ec:30:
    bd:2e:ce:43:1c:26:87:5d:b8:ae:72:01:ba:23:bf:
    c7:dd:06:fe:d2:dd:e5:ff:ab:e2:b0:a8:26:db:72:
    29:56:40:40:83:f0:18:22:cc:13:25:8c:66:fc:5e:
    9c:d1:49:5a:58:43:5e:93:fe:5c:86:73:2b:61:5e:
    8f:08:e0:8c:f9:ca:a8:bf:51:61:0f:15:c3:b8:00:
    8c:9d:77:04:9a:b7:9f:95:84:c5:88:76:ae:c8:a3:
```

```
└─$ openssl pkey -in dh_client1_key.pem -text -noout
DH Private-Key: (2048 bit)
private-key:
    64:9e:30:38:cf:a3:93:dd:7a:b7:05:3a:37:5b:4e:
    75:59:45:1d:6b:4d:4a:45:7f:2b:8b:68:d0:8a:c4:
    e3:f5:e7:8c:10:69:31:65:96:30:46:d8:11:c8:33:
    38:4c:21:13:99:a1:92:17:08:f2:ee:b6:81:61:b2:
    dc:19:7b:f1:00:ff:2a:92:32:19:8b:0c:2a:44:f2:
    ca:cb:2b:10:31:ae:eb:92:6a:05:04:5b:0b:97:6e:
    e9:ef:fd:f0:e2:7e:e0:9e:c3:ed:88:ad:ae:de:f6:
    b1:ba:95:85:fa:15:06:95:71:3c:f7:a6:2a:c0:77:
    f7:ab:f6:8a:8e:b1:66:2c:1d:ab:f9:44:a3:c8:fd:
    3b:67:49:34:b7:f6:90:c9:66:a5:33:4d:0c:39:67:
    8d:c1:da:3c:dd:8d:a5:c0:5a:e8:05:d2:40:34:d3:
    96:fa:a9:ff:5d:53:f2:8c:90:e8:8d:05:c3:c4:dc:
    b3:91:de:7b:3b:1d:c4:a8:d2:1b:9f:10:6f:ce:a5:
    d1:9d:d2:aa:be:d7:fb:f5:a7:0d:fb:f7:5b:39:7c:
    11:ca:97:88:10:66:3d:53:68:18:e0:63:3b:56:e9:
    f9:31:c0:bd:7a:24:9a:50:14:4d:75:f5:68:e3:64:
    48:fb:13:cc:20:4f:b0:44:c0:1f:78:b8:56:d6:6e:
    82
public-key:
```

d)

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ openssl pkey -in dh_client1_key.pem -pubout -out dh_client1_pub_key.pem

┌──(stud㉿kali-vm)-[~/exchange]
└─$ ls
dh_client1_key.pem      dh_client2_key.pem
dh_client1_pub_key.pem  dh_com_pub.pem

┌──(stud㉿kali-vm)-[~/exchange]
└─$
```

```
student@student-ubuntu:~/exchange$ openssl pkey -in dh_client2_key.pem -pubout -
out dh_client2_pub_key.pem
student@student-ubuntu:~/exchange$ ls
dh_client1_key.pem      dh_client2_key.pem      dh_com_pub.pem
dh_client1_pub_key.pem  dh_client2_pub_key.pem
student@student-ubuntu:~/exchange$
```

e)

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ openssl pkeyutl -derive -inkey dh_client1_key.pem -peerkey dh_client2_pub
_key.pem -out secret_key1.bin

┌──(stud㉿kali-vm)-[~/exchange]
└─$ ls
dh_client1_key.pem      dh_client2_key.pem      dh_com_pub.pem
dh_client1_pub_key.pem  dh_client2_pub_key.pem  secret_key1.bin

┌──(stud㉿kali-vm)-[~/exchange]
└─$
```

```
student@student-ubuntu:~/exchange$ openssl pkeyutl -derive -inkey dh_client2_key
.pem -peerkey dh_client1_pub_key.pem -out secret_key2.bin
student@student-ubuntu:~/exchange$ ls
dh_client1_key.pem      dh_client2_pub_key.pem  secret_key2.bin
dh_client1_pub_key.pem  dh_com_pub.pem
dh_client2_key.pem      secret_key1.bin
student@student-ubuntu:~/exchange$
```

f)

cmp nie zwróciło żadnej informacji o różnicach, więc oba sekrety są dokładnie takie same.

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ cmp -l secret_key1.bin secret_key2.bin

┌──(stud㉿kali-vm)-[~/exchange]
└─$
```

g)

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ xxd secret_key1.bin
00000000: 6805 0d81 b4b0 06d8 f684 4114 1679 9cd4  h.........A..y..
00000010: 714f 24e9 7467 f122 2f7f 9798 ef41 dddf  qO$.tg."/....A..
00000020: 2b34 679b ba45 0e79 28ab ad81 46b5 b755  +4g..E.y(...F..U
00000030: 23d9 b39e a216 51dd d7fa 2a6b 4cd9 d647  #.....Q...*kL..G
00000040: a573 dd18 b448 4fce b638 0f90 a547 6ce9  .s...HO..8...Gl.
00000050: 60f8 49cf 25e9 9e57 7d78 8d65 e60a fc50  `.I.%..W}x.e...P
00000060: 45d4 0664 e386 b869 1eee 401d e3e4 18e7  E..d...i..@.....
00000070: 6e9b ee54 48fd 5c38 9b4f cdc8 07d7 ecfa  n..TH.\8.O......
00000080: 6600 545f 6f68 45fd 73b4 2654 d080 de6b  f.T_ohE.s.&T...k
00000090: 80be 84b4 e9cc 3a24 0e4e 2c2a b460 c341  ......:$.N,*.`.A
000000a0: 7633 6ffe 1a6d a488 2da6 ccf5 85d3 1fc1  v3o..m..-.......
000000b0: aeaf ec65 040e 3318 c8d2 0efe c0ee 7037   ...e..3.......p7
000000c0: 5fa8 a5a0 05e3 e466 1d60 9336 4987 abf6  _.......f.`.6I...
000000d0: 7a3d 7c1b 4274 25fe f366 2840 1199 3d3a  z=|.Bt%..f(@..=:
000000e0: b65e d045 d2b6 f560 4769 45db 968d a406  .^.E...`GiE.....
000000f0: 67bb 94ed d1f8 04c9 c4c0 e048 1a35 66fa  g..........H.5f.
```

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ xxd secret_key2.bin
00000000: 6805 0d81 b4b0 06d8 f684 4114 1679 9cd4  h.........A..y..
00000010: 714f 24e9 7467 f122 2f7f 9798 ef41 dddf  qO$.tg."/....A..
00000020: 2b34 679b ba45 0e79 28ab ad81 46b5 b755  +4g..E.y(...F..U
00000030: 23d9 b39e a216 51dd d7fa 2a6b 4cd9 d647  #.....Q...*kL..G
00000040: a573 dd18 b448 4fce b638 0f90 a547 6ce9  .s...HO..8...Gl.
00000050: 60f8 49cf 25e9 9e57 7d78 8d65 e60a fc50  `.I.%..W}x.e...P
00000060: 45d4 0664 e386 b869 1eee 401d e3e4 18e7  E..d...i..@.....
00000070: 6e9b ee54 48fd 5c38 9b4f cdc8 07d7 ecfa  n..TH.\8.O......
00000080: 6600 545f 6f68 45fd 73b4 2654 d080 de6b  f.T_ohE.s.&T...k
00000090: 80be 84b4 e9cc 3a24 0e4e 2c2a b460 c341  ......:$.N,*.`.A
000000a0: 7633 6ffe 1a6d a488 2da6 ccf5 85d3 1fc1  v3o..m..-.......
000000b0: aeaf ec65 040e 3318 c8d2 0efe c0ee 7037   ...e..3.......p7
000000c0: 5fa8 a5a0 05e3 e466 1d60 9336 4987 abf6  _.......f.`.6I...
000000d0: 7a3d 7c1b 4274 25fe f366 2840 1199 3d3a  z=|.Bt%..f(@..=:
000000e0: b65e d045 d2b6 f560 4769 45db 968d a406  .^.E...`GiE.....
000000f0: 67bb 94ed d1f8 04c9 c4c0 e048 1a35 66fa  g..........H.5f.
```

Pytanie 2.1;

Wymiana kluczy za pomocą algorytmu Diffiego-Hellmana jest bezpieczniejsza w porównaniu do RSA, ponieważ pozwala na ustanowienie tajnego klucza bez jego bezpośredniego przesyłania, co chroni przed przechwyceniem go przez osoby trzecie. DH umożliwia też PFS (perfect forward secrecy), co oznacza, że nawet jeśli klucz prywatny jednej ze stron zostanie złamany, wcześniejsze sesje pozostaną bezpieczne. Jednak algorytm Diffiego-Hellmana nie pozwala uwierzytelniać autorów wiadomości, więc nie nadaje się on do podpisów elektronicznych – w tych zastosowaniach stosuje się RSA, które obsługuje zarówno szyfrowanie, jak i uwierzytelnianie za pomocą klucza prywatnego.

Pytanie 2.2;

Główne ryzyko, które wiąże się z wymianą klucza Diffiego-Hellmana polega na możliwości ataku pośredniego typu „*man in the middle*” (MitM), podczas którego atakujący przejmuje kontrolę nad komunikacją i wprowadza w błąd strony, które biorą w niej udział, tworząc oddzielne sesje szyfrowane z każdą ze stron, które pozostają tego nieświadome. DH nie posiada uwierzytelniania, więc bez dodatkowych mechanizmów uwierzytelniających strony nie mogą być pewne, czy faktycznie komunikują się akurat ze sobą. Oczywiście, jeśli stosowane są niewystarczająco silne parametry (np. małe liczby pierwsze) klucz wymieniony przez DH może być podatny na złamanie.

Pytanie 2.3;

Wśród innych metod ustalania wspólnego klucza kryptograficznego można nadmienić: *„Elliptic Curve Diffie-Hellman”* (ECDH), który jest wariantem Diffie-Hellmana opartym na krzywych eliptycznych, zapewniającym większe bezpieczeństwo przy krótszych kluczach oraz *„Kluczowanie symetryczne z centralnym serwerem”* (np. Kerberos), gdzie centralny serwer uwierzytelniający generuje i rozsyła klucze sesji do obu stron komunikacji. Kolejną metodą jest *„Password-Authenticated Key Agreement”* (PAKE), która pozwala stronom uzgodnić wspólny klucz kryptograficzny przy użyciu częściowej znajomości wspólnego hasła, zapewniając bezpieczeństwo nawet w przypadku podsłuchu i bez potrzeby przesyłania hasła w formie jawnej. Wszystkie te metody oferują różne poziomy bezpieczeństwa, zależnie od wymaganego zastosowania.

Pytanie 2.4;

Element protokołu DH nie przesyłany pomiędzy klientami to klucze tajne każdego uczestnika komunikacji. Używane są one jedynie do operacji generowania klucza publicznego. Wspólny klucz wynikowy jest efektem połączenia klucza prywatnego jednej strony z kluczem publicznym strony przeciwnej i będzie dokładnie taki sam dla obu stron. Ten fakt sprawia, że klucz tajny pozostanie tajny. Dodatkowo do generowania kluczy wykorzystuje się bardzo duże liczby pierwsze, przez co proces faktoryzacji tych liczb (a więc odwrócenia klucza) jest obecnie bardzo czasochłonny i nie do wykonania w sensownym czasie. W związku z tym nie jest możliwe podsłuchanie komunikacji i odtworzenie na tej podstawie klucza, ponieważ nie posiadamy dość informacji.

Pytanie 2.5;

Tak udało się ustalić ten sam klucz dla obu klientów:

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ xxd secret_key1.bin
00000000: 6805 0d81 b4b0 06d8 f684 4114 1679 9cd4  h.........A..y..
00000010: 714f 24e9 7467 f122 2f7f 9798 ef41 dddf  qO$.tg."/....A..
00000020: 2b34 679b ba45 0e79 28ab ad81 46b5 b755  +4g..E.y(...F..U
00000030: 23d9 b39e a216 51dd d7fa 2a6b 4cd9 d647  #.....Q...*kL..G
00000040: a573 dd18 b448 4fce b638 0f90 a547 6ce9  .s...HO..8...Gl.
00000050: 60f8 49cf 25e9 9e57 7d78 8d65 e60a fc50  `.I.%..W}x.e...P
00000060: 45d4 0664 e386 b869 1eee 401d e3e4 18e7  E..d...i..@.....
00000070: 6e9b ee54 48fd 5c38 9b4f cdc8 07d7 ecfa  n..TH.\8.O......
00000080: 6600 545f 6f68 45fd 73b4 2654 d080 de6b  f.T_ohE.s.&T...k
00000090: 80be 84b4 e9cc 3a24 0e4e 2c2a b460 c341  ......:$.N,*.`.A
000000a0: 7633 6ffe 1a6d a488 2da6 ccf5 85d3 1fc1  v3o..m..-.......
000000b0: aeaf ec65 040e 3318 c8d2 0efe c0ee 7037  ...e..3.......p7
000000c0: 5fa8 a5a0 05e3 e466 1d60 9336 4987 abf6  _......f.`.6I...
000000d0: 7a3d 7c1b 4274 25fe f366 2840 1199 3d3a  z=|.Bt%..f(@..=:
000000e0: b65e d045 d2b6 f560 4769 45db 968d a406  .^.E...`GiE.....
000000f0: 67bb 94ed d1f8 04c9 c4c0 e048 1a35 66fa  g..........H.5f.
```

```
┌──(stud㉿kali-vm)-[~/exchange]
└─$ xxd secret_key2.bin
00000000: 6805 0d81 b4b0 06d8 f684 4114 1679 9cd4  h.........A..y..
00000010: 714f 24e9 7467 f122 2f7f 9798 ef41 dddf  qO$.tg."/....A..
00000020: 2b34 679b ba45 0e79 28ab ad81 46b5 b755  +4g..E.y(...F..U
00000030: 23d9 b39e a216 51dd d7fa 2a6b 4cd9 d647  #.....Q...*kL..G
00000040: a573 dd18 b448 4fce b638 0f90 a547 6ce9  .s...HO..8...Gl.
00000050: 60f8 49cf 25e9 9e57 7d78 8d65 e60a fc50  `.I.%..W}x.e...P
00000060: 45d4 0664 e386 b869 1eee 401d e3e4 18e7  E..d...i..@.....
00000070: 6e9b ee54 48fd 5c38 9b4f cdc8 07d7 ecfa  n..TH.\8.O......
00000080: 6600 545f 6f68 45fd 73b4 2654 d080 de6b  f.T_ohE.s.&T...k
00000090: 80be 84b4 e9cc 3a24 0e4e 2c2a b460 c341  ......:$.N,*.`.A
000000a0: 7633 6ffe 1a6d a488 2da6 ccf5 85d3 1fc1  v3o..m..-.......
000000b0: aeaf ec65 040e 3318 c8d2 0efe c0ee 7037  ...e..3.......p7
000000c0: 5fa8 a5a0 05e3 e466 1d60 9336 4987 abf6  _......f.`.6I...
000000d0: 7a3d 7c1b 4274 25fe f366 2840 1199 3d3a  z=|.Bt%..f(@..=:
000000e0: b65e d045 d2b6 f560 4769 45db 968d a406  .^.E...`GiE.....
000000f0: 67bb 94ed d1f8 04c9 c4c0 e048 1a35 66fa  g..........H.5f.
```