Politechnika Wrocławska

SPRAWOZDANIE Z ZAJĘĆ LABORATORYJNYCH
CYBERBEZPIECZEŃSTWO

Lab 07
Zastosowania kryptografii

Tomasz Mroczko, 266604

November 23, 2023

1 Zadania

Zadanie 1.1

Wygeneruj zestaw kluczy za pomocą OpenSSL (RSA) i OpenPGP (ECC i RSA).

Wygenerowano klucze RSA o długości 2048 bitów w obu przypadkach.

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openssl$ openssl genrsa -out private_key.pem 2048
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openssl$ openssl rsa -in private_key.pem -outform PEM
-pubout -out public_key.pem
writing RSA key
```

Figure 1: Generacja kluczy OpenSSL

```
sejsmo@LAPTOP-DJIP1POB:~$ gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
/home/sejsmo/.gnupg/pubring.kbx
--------------------------------
pub   rsa2048 2023-11-23 [SC]
      F057B8D8DA9FDC967B03B8B33B485066023EE65A
uid           [ultimate] Tomasz Mroczko (laboratorium cyberbezpieczeństwo) <tomekmr2001@gmai
l.com>
sub   rsa2048 2023-11-23 [E]
```

Figure 2: Generacja klucza RSA OpenPGP

```
pub   ed25519 2023-11-23 [SC]
      5A8B0A81C64F12DF31CAF6DB7DABC262213873A5
uid           [ultimate] Tomasz Mroczko (laboratorium cyberbezpieczenstwo) <tomekmr2001@gmai
l.com>
sub   cv25519 2023-11-23 [E]
```

Figure 3: Generacja klucza ECC OpenPGP

Zadanie 1.2

Eksport klucza publicznego z PGP (format ACSII).

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openpgp$ gpg --export --armour --output pgp_rsa_public_key "RSA-key"
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openpgp$ ls
pgp_rsa_public_key
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openpgp$ cat pgp_rsa_public_key
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBGVfFCABCACg5E+KSuYV65Rrq/+domGFFRrVefIbld3iZi1REvS6DhQEIKF9
keFaGGNsJGsyXiSuFkUtiFaTZN+loY5CcBGzn8QSrnYPswT9WtS04G9lJB1TSvy9
KrOzQ/MtBg0qpUh3WoGg7n7xKmnt7h8yMr4gUgE+4b8dtUcukmlZW3qWG9Vs30Qd
Bts7fI8mbBvaTcpKyGONesOCWODL12gQtdOewi9rLvzwpixfR3E3JwEcPTMcUNeS
Jmi9yRpXUTryMYRUqwb4762uSadl1Awg+L66rCIk8RhBohGwn2oscBYyMMSJpwt2
TRM0nhq5i6yhyl4lA6F2UEiWs37dwScGcHQ5ABEBAAG0KVJTQS1rZXkgKFJTQSBr
ZXkpIDx0b21la21yMjAwMUBnbWFpbC5jb20+iQFOBBMBCgA4FiEEJUIhN3tB7rW/
zw7jILwIwFXJE7sFAmVfFCACGwMFCwkIBwIGFQoJCAsCBBYCAwECHgECF4AACgkQ
ILwIwFXJE7tpvgf/Y4Xqdj2AwT26Oyvi2b5kGT86sHjL55ulIM1vs2hdNOWO4Hj3
XiE20oNhwkoVcgVvz2Oos5fNMV3wNw3lv9shN2ZGHL+IWaWxzat42RXni74HXU0e
cbrREmXqU3ZWxsAHOlUrsnFXt+IXgZOyecCOwqKLki52OmX0rg+ozPWhwbV8eEm7
f4xMgdC9ABPYoTOwmhy2CSjZRXGfp+ZhUVRwsxEADKFLBmrKDEfNfgwKXREIDajK
v2S1IY6G2lXia9k46mNtLZDra4aSGLgE5zpjO/zisvbv2s8eTSn7emKMO1l+1mqt
T8Z48Q2DMEOi/XhdyuAVPxdNQ9J5CI8jLCCf2bkBDQRlXxQgAQgAxuV9yed1uDpQ
LVmNqfBx/sCYLkPAJLqUoHj2Jo3TXjdM2XM7XqyRd/l0TozmgeV5ErtV/Nncdc3t
GhjVVB6DEm7W/UAJV8WRr4uiCzq8/5opDzfGErt7ffLWTT1dyGyxFcRErh39tMmQ
s9MCsGhVh956m33QUtOHI15K34DtL4psRApqZhunVV+7VzY5B8z878+2EiGP3m29
hQwECikyJJSCWY6jGEE8Rj5R+hqjFY9EVN2+qCVddyhdu17mdWJQ/Da06kiLoXSD
VPv2eS7eXxIYCKwxVGsWPCxGbwcNe5g0bux/T1zQnfM66gTc5/sNGz9nZ6uHIPTj
ymQEgq7lGQARAQABiQE2BBgBCgAgFiEEJUIhN3tB7rW/zw7jILwIwFXJE7sFAmVf
FCACGwwACgkQILwIwFXJE7t1IwgAkkohIRqIcXVQa8q2+8DkFgJNwikCbEeuQ/jV
00tSZrwI9y68hpOrR+zIum1T7HZKT2bGMyIAxfh+E7s2ArCU4A9lLscAaWNEW0YN
Z574yk9OegEgAsdo3GVVkKKaEuodLXLRh3XWEL3gUwPFL1VGGy6wWXSLZEAqzBw5
nPmmpT5pHH9REqOFB52zbQLvz+zHOnwiZUR9OWSDeMKJLaJ+8UG6li2bpyXFlvNv
BRif+NkWNw4UvdX8as7LYFNxd6ECeuxq+jTu7gnx3pAgzT1v011l+N2QMsliMiLv
dG+exoa368OSjj7ER9iMHxovCuA2/OKxJWET7IvJkWrsz+h1Tg==
=s/6/
-----END PGP PUBLIC KEY BLOCK-----
```

Figure 4: Wyeksportowany klucz PGP RSA

Zadanie 1.3

Przenieś swój klucz publiczny (OpenSSL i OpenPGP) na inny komputer / konto użytkownika.

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openssl$ sudo cp public_key.pem /home/client2/lab7/openssl/
[sudo] password for sejsmo:
```

Figure 5: Przeniesienie klucza OpenSSL

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1/openpgp$ sudo cp pgp_rsa_public_key /home/client2/lab7/openpgp/
```

Figure 6: Przeniesienie klucza OpenPGP

Zadanie 1.4

W przypadku OpenPGP klucz publiczny należy zaimportować i podpisać lokalnym kluczem prywatnym. Sprawdź odcisk palca certyfikatu na obu kontach komputera /użytkownika.

```
client2@LAPTOP-DJIP1POB:~/lab7/openpgp$ gpg --import pgp_rsa_public_key
gpg: key 20BC08C055C913BB: public key "RSA-key (RSA key) <tomekmr2001@gmail.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

Figure 7: Import klucza na drugim koncie

```
client2@LAPTOP-DJIP1POB:~/lab7/openpgp$ gpg --sign-key 254221377B41EEB5BFCF0EE320BC08C055C913BB

pub  rsa2048/20BC08C055C913BB
     created: 2023-11-23  expires: never       usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/C746DAB283019EBF
     created: 2023-11-23  expires: never       usage: E
[ unknown] (1). RSA-key (RSA key) <tomekmr2001@gmail.com>


pub  rsa2048/20BC08C055C913BB
     created: 2023-11-23  expires: never       usage: SC
     trust: unknown       validity: unknown
 Primary key fingerprint: 2542 2137 7B41 EEB5 BFCF  0EE3 20BC 08C0 55C9 13BB

     RSA-key (RSA key) <tomekmr2001@gmail.com>

Are you sure that you want to sign this key with your
key "sejmso (akjklj) <client2@gmail.com>" (4A8F652F9D5061D3)

Really sign? (y/N) y
```

Figure 8: Podpisanie zaimportowanego klucza publicznego za pomocą klucza Klienta 2

Zadanie 1.5

Utwórz plik tekstowy i podpisz go za pomocą obu metod.

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ nvim plik.txt
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ openssl dgst -sha256 -sign openssl/private_key.pem -out  encrypted_hash.sha256 plik.txt
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ ls
encrypted_hash.sha256  openpgp  openssl  plik.txt
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$
```

Figure 9: Podpis pliku za pomocą OpenSSL (klucza prywatnego z pliku .pem)

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ gpg --detach-sign -u 254221377B41EEB5BFCF0EE320BC08C055C913BB  --output plik.txt.
sig plik.txt
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ ls
encrypted_hash.sha256  openpgp  openssl  plik.txt  plik.txt.sig
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$
```

Figure 10: Podpisanie pliku za pomocą OpenPGP (prywatnego klucza RSA)

Zadanie 1.6

Przenieś plik i podpisy na inny komputer / konto użytkownika i zweryfikuj podpisy.

```
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ sudo cp plik.txt /home/client2/lab7/
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ sudo cp encrypted_hash.sha256 /home/client2/lab7/
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$ sudo cp plik.txt.sig /home/client2/lab7/
sejsmo@LAPTOP-DJIP1POB:~/lab7/client1$
```

Figure 11: Przeniesienie pliku textowego i podpisów na konto drugiego klienta

```
client2@LAPTOP-DJIP1POB:~/lab7$ openssl dgst -sha256 -verify ./openssl/public_key.pem -signature encrypted_hash.sha256 plik.txt
Verified OK
client2@LAPTOP-DJIP1POB:~/lab7$
```

Figure 12: Zweryfikowanie podpisu OpenSSL

Figure 13: Zwerfikowanie podpisu OpenPGP

Udało się zweryfikować poprawność obu podpisów

Zadanie 1.7

Po pomyślnym zweryfikowaniu podpisu należy zmodyfikować treść oryginalnego dokumentu i ponownie zweryfikować podpis cyfrowy

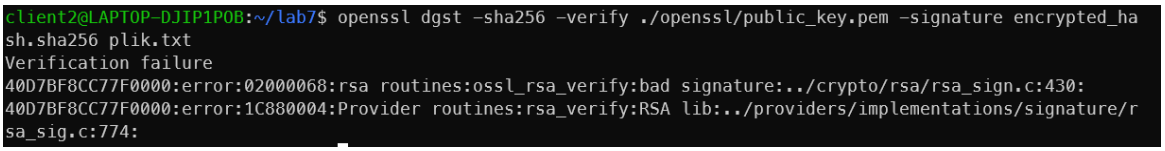

Figure 14: Porażka przy weryfikacji zmienionego pliku (OpenSSL)

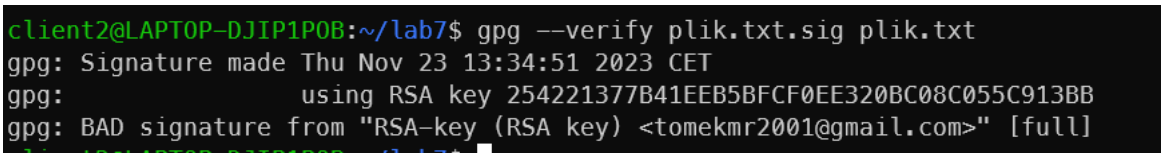

Figure 15: Porażka przy weryfikacji zmienionego pliku (OpenPGP)

Zadanie 1.9

Czy można wygenerować tylko jeden z pary kluczy dla algorytmów asymetrycznych (np. klucz prywatny)? Czy to miałoby sens?

W teorii można wygenerować tylko jeden klucz(np. prywatny). Narzędzie lini komend gpg daje możliwość wygenerowania jednego klucza (sign only). W praktyce trudno wyobrazić sobie korzyści płynące z pojedynczego klucza. Posiadanie tylko klucza prywatnego pozwala na podpisanie klucza publicznego należącego do innego użytkownika w celu potwierdzenia jego autentyczności. Operacja taka wymaga hasła więc wprowadza dodatkowy poziom bezpieczeństwa do systemu. Jeden klucz może być w teorii używany do podpisania, jednak użytkownicy którzy nie posiadają naszego klucza publicznego (a nie posiada go nikt jeśli nie istnieje), nie są w stanie zweryfikować że plik jest podpisany przez nas.

Zadanie 1.10

Jaką formę ma klucz GPG? Czym różni się klucz PEM od klucza OpenPGP?

Oba klucze wygenerowane zostały algorytmem RSA oraz długości 2048. Klucz OpenPGP wymagał wyeksportowania w celu odczytania go w formie ASCI. Pomimo tej samej długości klucza, wyeksportowany plik jest znacznie dłuższy niż klucz .pem wygenerowany *openssl*. Wynika

to z faktu, że klucze OpenPGP zawierają dodatkowe metadane. Dostarczają one informacji na temat właściciela klucza(nazwisko i email), daty utworzenia, daty wygaśnięcia klucza, identyfikatora klucza, algorytmu użytego do utworzenia klucza oraz odcisku palca (fingerprint) czyli ciągu znaków, który jest wynikiem funkcji skrótu klucza i pozwala potwierdzić jego oryginalność. Klucz taki, po wyeksportowaniu i udostępnieniu, daje adresatowi dużo informacji o swoim pochodzeniu co pozwala zwiększyć bezpieczeństwo.

Zadanie 1.11

Czy wymiana kluczy prywatnych jest uzasadniona?

Wymiana kluczy prywatnych wydaje się nie być uzasadniona.

Zadanie 1.12

Jaki rezultat zostałosiągniety w zadanie 1.7 i dlaczego ?

W zadaniu 1.7, po modyfikacji otrzymanego pliku, nie udało się zwerfyikować poprawności podpisów dotyczących tego pliku. Wynika to z tego, że proces podpisywania plików bazuje na treści podpisywanego dokumentu. Jeśli dokument zostanie zmodyfikowany to weryfikacja nie powiedzie się. Funkcje skrótu powinny posiadać efekt lawinowy, który mówi o tym, że nawet najmniejsza zmiana dokumentu powinna znacząco zmienić wartość końcową.

Zadanie 1.13

Co to jest odcisk palca?

Odcisk palca (fingerprint) to ciąg znaków generowany na podstawie klucza publicznego. Generowany jest za pomocą zaaplikowania funkcji skrótu na kluczu. Używany jest do weryfikacji czy klucz publiczny jest autentyczny. Właściciel może przekazać innym kanałem fingerprint, który adresat może porównać z odciskiem otrzymanego klucza.

Zadanie 1.15

Jaka jest różnica w sygnaturze (OpenPGP) dla algorytmu ECC i RSA?

Główna różnica jest taka że algorytm ECC daje dużo krótszy klucz (957 kontra 1757 znaków). Klucz ECC jest dużo krótszy nić RSA, ponieważ krytpografia krzywych eliptycznych pozwala na znacznie mniejszą długość klucza przy zachowaniu bezpieczeństwa.

Zadanie 1.16

Czy wiadomość może być podpisana kluczem publicznym? Jeśli tak, jakie mogą być konsekwencje.

Wiadomosc teorytycznie może być podpisana kluczem publicznym, jednak jednak jest to niebezpieczne, niezalecane i błędne. Konsekwencjami takiego działania jest możliwosc podpisania w naszym imieniu pliku przez każdego kto jest w posiadaniu klucza. Dodatkowo zweryfikowanie poprawności takiego podpisu jest niemożliwe.