Dokumentacja końcowa projektu
Algorytm DSA

Jakub Turek

Podstawy teoretyczne

FIPS[1] jest zbiorem, w którym opisane są publiczne standardy bezpieczeństwa używane przez federalny rząd Stanów Zjednoczonych. Oficjalnym standardem podpisywania wiadomości cyfrowych zamieszczonym w FIPS jest DSS (ang. Digital Signature Standard). DSS opiera się o algorytm DSA (ang. Digital Signature Algorithm).

Standard DSS (wraz z algorytmem DSA) został opisany w dokumencie FIPS PUB 186[2]. Na potrzeby projektu zaimplementowany został oryginalny algorytm opublikowany w 1994 roku, który wykorzystuje funkcję skrótu SHA.

Algorytm

Generacja kluczy Należy wybrać parametry:

- Liczba pierwsza p w pierścieniu reszt modulo a, gdzie $2^{L-1} < p < 2^L$ oraz $512 \leqslant L \leqslant 1024$ i L jest wielokrotnością 64.
- Liczba pierwsza q będąca dzielnikiem liczby $p-1$ w pierścieniu reszt modulo a, gdzie $2^{159} < q < 2^{160}$.
- Liczba $g = h^{\frac{p-1}{q}} \pmod p$, gdzie h jest dowolną liczbą naturalną spełniającą warunek $1 < h < p-1$ taką, że $h^{\frac{p-1}{q}} \pmod p > 1$ (czyli g ma rząd $q \pmod p$).
- Losowo wygenerowana liczba x z przedziału $0 < x < q$.
- Liczba $y = g^x \pmod p$.
- Losowo wygenerowana liczba k z przedziału $0 < k < q$.

Liczby p, q oraz g są publiczne. Klucz prywatny użytkownika to x, natomiast klucz publiczny użytkownika to y. Parametr k musi być obliczany dla każdego nowego podpisu. Klucze są wielokrotnego użytku.

Podpisywanie wiadomości Podpisem wiadomości M jest para liczb (r, s) obliczanych według poniższego wzoru:

$$\begin{aligned} r &= (g^k \mod p) \mod q \\ s &= (k^{-1}(SHA(M) + xr)) \mod q \end{aligned}$$

gdzie k^{-1} jest odwrotnością liczby k w pierścieniu reszt modulo q (czyt. $k \cdot k^{-1} \equiv 1 \pmod q$).

Opcjonalnie można zweryfikować, czy $r \neq 0$ i $s \neq 0$. Jeżeli jeden z warunków nie jest spełniony, należy wygenerować podpis od nowa. Sytuacja taka nie powinna się jednak zdarzyć dla prawidłowo wygenerowanych kluczy.

[1] Skrót od **F**ederal **I**nformation **P**rocessing **S**tandard (ang. federalny standard przetwarzania informacji).
[2] `http://www.itl.nist.gov/fipspubs/fip186.htm`.

Weryfikacja podpisu Zakładamy, że otrzymaliśmy zestaw $(M', (r', s'))$ składający się z wiadomości oraz podpisu tej wiadomości. Aby zweryfikować podpis należy:

1. Dokonać sprawdzenia, że $0 < r' < q$, a ponadto $0 < s' < q$.
2. Obliczyć poniższe wartości:

$$\begin{aligned} w &= (s')^{-1} \pmod q \\ u_1 &= SHA(M') \cdot w \pmod q \\ u_2 &= r' \cdot w \pmod q \\ v &= (g^{u_1} \cdot y^{u_2} \mod p) \pmod q \end{aligned}$$

3. Sprawdzić, czy $v = r'$.
4. Podpis jest prawidłowy, jeżeli warunek z punktu 3. jest spełniony.

Testy

Scenariusz testów opisany jest testami jednostkowymi klasy `DSAKey`. Znajdują się one w module `DSAKeyTests`.

Podstawowym testem na poprawne działanie algorytmu jest podpisanie wiadomości kluczem prywatnym z wykorzystaniem algorytmu DSA, a następnie zweryfikowanie podpisu tej samej wiadomości przy użyciu klucza publicznego. Opisuje to scenariusz `test_positive_workflow()`. W scenariuszu tym:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.
2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości `Test message` uzyskany w punkcie 2.
4. Do poprawnego zakończenia testu wymagane jest, aby podpis był prawidłowy.

W dalszej kolejności należy upewnić się, że jakakolwiek modyfikacja wiadomości powoduje, że podpis przestaje być ważny. Sprawdzenie takie wykonywane jest w trzech osobnych scenariuszach:

- Zmodyfikowana wiadomość posiada zmienioną literę na jednej pozycji.
- Zmodyfikowana wiadomość ma zamienioną kolejność dwóch liter.
- Zmodyfikowana wiadomość została skrócona.

Scenariusze te mają za zadanie sprawdzić, że każda drobna modyfikacja wiadomości powoduje, iż podpis traci ważność.

Pierwszy ze scenariuszy negatywnych (zmieniona litera na jednej pozycji) przebiega następująco:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.
2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości uzyskany w punkcie 2., przy czym wiadomość M zostaje zastąpiona treścią `Best message`.
4. Do poprawnego zakończenia testu wymagane jest, aby podpis został odrzucony.

Kolejny ze scenariuszy negatywnych (zamieniona kolejność dwóch liter) przebiega następująco:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.
2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości uzyskany w punkcie 2., przy czym wiadomość M zostaje zastąpiona treścią `Tset message`.
4. Do poprawnego zakończenia testu wymagane jest, aby podpis został odrzucony.

Ostatni ze scenariuszy negatywnych (skrócona wiadomość) przebiega następująco:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.
2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości uzyskany w punkcie 2., przy czym wiadomość M zostaje zastąpiona treścią `message`.
4. Do poprawnego zakończenia testu wymagane jest, aby podpis został odrzucony.

Dodatkowo należy upewnić się, że zmiana podpisu wiadomości (a więc jednej lub obu liczb z pary (r, s)) również spowoduje, że podpis zostanie odrzucony. Testowane są trzy przypadki:

- Wybierana jest losowa wartość r'.
- Wybierana jest losowa wartość s'.
- Wybierana jest losowa wartość pary (r', s').

Aby „uprawdopodobnić", że losowa wartość klucza (lub jego części) zostanie uznana za poprawną w każdym teście generowane są zmienione liczby o długości bitowej zgodnej z oryginalnymi długościami bitowymi (r, s).

Pierwszy ze scenariuszy negatywnych dla zmienionego podpisu (r, s) wiadomości przebiega następująco:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.
2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Generowana jest losowa wartość r' o długości bitowej równej długości bitowej r.
4. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości `Test message` o wartości (r', s).
5. Test zakończy się poprawnie wtedy i tylko wtedy, gdy weryfikacja zakończy się tym samym wynikiem co porównanie $r' == r$.

Kolejny ze scenariuszy negatywnych dla zmienionego podpisu (r, s) wiadomości przebiega następująco:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.
2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Generowana jest losowa wartość s' o długości bitowej równej długości bitowej s.
4. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości `Test message` o wartości (r, s').
5. Test zakończy się poprawnie wtedy i tylko wtedy, gdy weryfikacja zakończy się tym samym wynikiem co porównanie $s' == s$.

Ostatni ze scenariuszy negatywnych dla zmienionego podpisu (r, s) wiadomości przebiega następująco:

1. Generowana jest losowa para kluczy prywatny oraz publiczny.

2. Przy pomocy klucza prywatnego podpisywana jest wiadomość `Test message`.
3. Generowana jest losowa wartość r' o długości bitowej równej długości bitowej r.
4. Generowana jest losowa wartość s' o długości bitowej równej długości bitowej s.
5. Przy pomocy klucza publicznego weryfikowany jest podpis wiadomości `Test message` o wartości (r', s').
6. Test zakończy się poprawnie wtedy i tylko wtedy, gdy weryfikacja zakończy się tym samym wynikiem co porównanie $r' == r \wedge s' == s$.