**Лабораторна робота № 5**

**СТВОРЕННЯ ПРОГРАМНОГО ЗАСОБУ ДЛЯ ЦИФРОВОГО ПІДПИСУ ІНФОРМАЦІЇ З ВИКОРИСТАННЯМ MICROSOFT**[**CRYPTOAPI**](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187736&displayformat=dictionary)

**Мета роботи:** ознайомитись з методами криптографічного забезпечення цифрового підпису, навчитись створювати програмні засоби для цифрового підпису з використанням криптографічних інтерфейсів.

**Теоретичні відомості.**

[Аутентифікація](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187756&displayformat=dictionary) захищає двох учасників, які обмінюються повідомленнями, від впливу деякої третьої сторони. Однак проста [аутентифікація](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187756&displayformat=dictionary) не захищає учасників один від одного, тоді як і між ними теж можуть виникати певні форми суперечностей.

У ситуації, коли обидві сторони не довіряють один одному, необхідно щось більше, ніж [аутентифікація](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187756&displayformat=dictionary) на основі спільного секрету. Можливим рішенням подібної проблеми є використання цифрового підпису. Цифровий підпис повинен володіти наступними властивостями:

1.     Повинна бути можливість перевірити автора, дату й час створення підпису.

2.     Повинна бути можливість встановити достовірність вмісту повідомлення на час створення підпису.

3.     Повинна бути можливість перевірки підпису третьою стороною для вирішення суперечок.

Таким чином, функція цифрового підпису включає, зокрема, і функцію аутентифікації.

На підставі цих властивостей можна сформулювати наступні вимоги до цифрового підпису:

1.     Підпис повинен бути бітовим зразком, який залежить від повідомлення, що підписується.

2.     Підпис повинен використовувати деяку унікальну інформацію відправника для запобігання підробки або відмови.

3.     Створювати цифровий підпис повинно бути відносно легко.

4.     Повинно бути обчислювально неможливо підробити цифровий підпис як створенням нового повідомлення для існуючого цифрового підпису, так і створенням фальшивого цифрового підпису для деякого повідомлення.

5.     Цифровий підпис повинен бути досить компактним і не займати багато пам'яті.

**Стандарт цифрового підпису**[**DSS**](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary)

Національний інститут стандартів і технології США (NIST) розробив федеральний стандарт цифрового підпису [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary). Для створення цифрового підпису використовується алгоритм [DSA](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187739&displayformat=dictionary) (Digital Signature Algorithm). В якості хеш-алгоритму стандарт передбачає використання SHA-1 (Secure Hash Algorithm). [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary) спочатку був запропонований в 1991 році й переглянутий в 1993 році у відповідь на публікації, що стосуються безпеки його схеми. У 1996 році в нього були внесені незначні зміни.

[DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary) використовує алгоритм, що розроблявся для використання тільки в якості цифрового підпису. На відміну від алгоритму [RSA](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187747&displayformat=dictionary), його не можна використовувати для [шифрування](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187834&displayformat=dictionary) чи обміну ключами.

Рис. 9. Створення й перевірка підпису згідно стандарту [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary).

Стандарт [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary) використовує при створенні підпису хеш-функцію. Значення хешу повідомлення є входом функції підпису разом з випадковим числом *k*, створеним для цього конкретного підпису (рис. 9). Функція підпису також залежить від приватного ключа відправника *KRA* і деякої множини параметрів, відомих усім учасникам. Можна вважати, що ця множина утворює глобальний відкритий ключ *KUG*. Результатом є підпис, що складається з двох компонентів, позначених як *s* та *r*.

Для перевірки підпису одержувач також обчислює [хеш-код](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187825&displayformat=dictionary) отриманого повідомлення. Цей [хеш-код](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187825&displayformat=dictionary) разом з підписом є входом функції верифікації. Функція верифікації залежить від глобального відкритого ключа *KUG* і відкритого ключа відправника *KUA*. Виходом функції верифікації є значення, що повинне дорівнювати компоненті *r* підпису, якщо підпис коректний. Функція підпису має такий вигляд, що тільки відправник, якому відомий приватний ключ, може створити коректний підпис.

Розглянемо деталі алгоритму, що використовується в стандарті [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary). Цей алгоритм заснований на труднощі обчислення дискретних логарифмів.

***Спільні компоненти групи користувачів (глобальний відкритий ключ)***

Існує три параметри, які є відкритими й можуть бути спільними для великої групи користувачів.

Вибирається 160-бітне просте число *q*, тобто 2159 < *q* < 2160.

Потім вибирається таке просте число *p* довжиною від 512 до 1024 бітів, так, щоб *q* було дільником (*p*-1).

Нарешті, вибирається число *g* виду , де *h* є цілим в проміжку від 1 до (*p*-1) з тим обмеженням, що *g* повинне бути більшим за 1.

Знаючи ці числа, кожний користувач вибирає приватний ключ і створює відкритий ключ.

**Приватний ключ відправника.**

Приватний ключ *x* повинен бути числом між 1 і (*p*-1) і повинен бути обраним випадково або псевдовипадково.

*x* – випадкове або псевдовипадкове ціле, 0<*x*<*q*.

**Відкритий ключ відправника.**

Відкритий ключ обчислюється із приватного ключа за формулою . Обчислити *y* за відомим *x* досить просто. Однак, маючи відкритий ключ *y*, обчислювально неможливо визначити *x*, що є дискретним логарифмом *y* за основою *g*.

**Випадкове число, унікальне для кожного підпису.**

*k* – випадкове або псевдовипадкове ціле число, 0<*k*<*q*, унікальне для кожного підпису.

**Підписування.**

Для створення підпису відправник обчислює дві величини, *r* і *s*, які є функцією компонентів спільного відкритого ключа (*p*,*q*,*g*), приватного ключа користувача (*x*), [хеш-код](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187825&displayformat=dictionary)у повідомлення *H*(*M*) і цілого числа *k*, унікального для кожного підпису.

**Перевірка підпису.**

Одержувач виконує перевірку підпису наступним чином. Він обчислює величину *v*, що є функцією компонентів спільного відкритого ключа, відкритого ключа відправника й [хеш-код](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187825&displayformat=dictionary)у отриманого повідомлення. Якщо значення цієї величини дорівнює значенню компонента *r* в підписі, то підпис вважається дійсним.

підпис коректний, якщо *v*=*r*.

Зверніть увагу на те, що перевірка здійснюється зі значенням *r*, що не залежить від повідомлення взагалі. Значення *r* є функцією *k* й трьох компонентів глобального відкритого ключа.

При труднощі обчислення дискретних логарифмів для противника виявляється нереальним з погляду обчислень знайти *k* за відомим *r* або знайти *x* за відомим *s*.

**Використання функцій**[**CryptoAPI**](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187736&displayformat=dictionary)**для реалізації цифрового підпису**

Базова функція отримання підпису хешу даних має наступний опис:

*BOOL CryptSignHash(HCRYPTHASH hHash, DWORD dwKeySpec, LPCTSTR s*[*Des*](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187737&displayformat=dictionary)*cription, WORD dwFlags, BYTE\* pbSignature, DWORD\* pdwSigLen);*

В якості першого параметру використовується значення дескриптора хеш-об'єкту, вже ініціалізованого даними (за допомогою, наприклад, функції CryptHashData). Параметр *dwKeySpec* визначає, яка саме пара ключів буде використана для формування підпису (*AT\_KEYEXCHANGE* (пара для обміну ключами) чи *AT\_SIGNATURE* (пара для формування цифрового підпису)). В багатьох (однак не в усіх) [криптопровайдер](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187781&displayformat=dictionary)ах пара ключів, призначена для обміну ключами, може також використовуватись і для формування цифрового підпису. Параметр *s*[*Des*](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187737&displayformat=dictionary)*cription* більше не використовується в цій функції, і його значення повинно завжди бути встановленим в *NULL*. Параметр *dwFlags* зазвичай також встановлюють в 0. Параметри *pbSignature*та*pdwSigLen*використовують для коректного визначення посилання на масив вихідних даних та його розмір.

Приклад використання цієї функції наведено нижче:

*// Цифровий підпис хеш значення*

*count = 0;*

*if(!CryptSignHash(hHash, AT\_SIGNATURE, NULL, 0, NULL, &count))*

*{*

*Error("CryptSignHash");*

*return;*

*}*

*char\* sign\_hash = static\_cast<char\*>(malloc(count + 1));*

*ZeroMemory(sign\_hash, count + 1);*

*if(!CryptSignHash(hHash, AT\_SIGNATURE, NULL, 0, (BYTE\*)sign\_hash, &count))*

*{*

*Error("CryptSignHash");*

*return;*

*}*

*std::cout << "Signature created" << std::endl;*

*// Вивід на екран значення цифрового підпису*

*std::cout << "Signature value: " << sign\_hash << std::endl;*

Для перевірки цифрового підпису хеш-значення використовується базова функція, що має наступний опис:

*BOOL CryptVerifySignature(HCRYPTHASH hHash, BYTE\* pbSignature, DWORD dwSigLen, HCRYPTKEY hPubKey, LPCTSTR s*[*Des*](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187737&displayformat=dictionary)*cription, DWORD dwFlags);*

В якості першого параметру в функцію передається дескриптор хеш-об'єкту, попередньо ініціалізованого даними засобами функції CryptHashData. Другий і третій параметри відповідають за передачу значення підпису, що перевіряється. Параметр *hPubKey* використовується для зазначення дескриптора публічного ключа відправника підпису. Параметр *s*[*Des*](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187737&displayformat=dictionary)*cription* на даний час більше не використовується і його значення повинно бути встановленим в *NULL*. Параметр *dwFlags* також зазвичай не несе корисної інформації і встановлюється в 0.

Приклад перевірки підпису наведено нижче:

*// Отримання публічного ключа (для перевірки цифрового підпису)*

*HCRYPTKEY hPublicKey;*

*if(!CryptGetUserKey(hProv, AT\_SIGNATURE, &hPublicKey))*

*{*

*Error("CryptGetUserKey");*

*return;*

*}*

*std::cout << "Public key is received" << std::endl;*

*// перевірка цифрового підпису*

*BOOL result = CryptVerifySignature(hHash, (BYTE\*)sign\_hash, count, hPublicKey, NULL, 0);*

*std::cout << "Check is completed" << std::endl;*

*// Вивід на екран результату перевірки цифрового підпису*

*std::cout << "Check result: " << ((result)? "Verified!" : "NOT verified!") << std::endl;*

Розмір підпису дорівнює розміру ключа [RSA](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187747&displayformat=dictionary) або [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary), відповідно.

**Завдання до виконання роботи**

З використання функцій [CryptoAPI](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187736&displayformat=dictionary) створити прикладну програму для створення і перевірки цифрового підпису за стандартом [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary). Програмна реалізація повинна виводити значення підпису як для рядка, заданого в полі вводу, так і для файлу. Результат роботи програми повинен відображатись на екрані з можливістю наступного запису в файл. Крім того програма повинна мати можливість перевірити цифровий підпис будь-якого файлу за наявним файлом підпису, записаним у шістнадцятковому форматі. У звіті навести протокол роботи програми та зробити висновки.

**Контрольні запитання.**

1.     Для чого призначений цифровий підпис?

2.     Назвіть вимоги до цифрового підпису.

3.     Який алгоритм хешування використовується в стандарті [DSS](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187738&displayformat=dictionary)?

4.     Чи можна використати алгоритм [DSA](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187739&displayformat=dictionary) для обміну ключами?

5.     На якій математичній проблемі засновано алгоритм [DSA](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187739&displayformat=dictionary)?

6.     Що таке глобальний відкритий ключ алгоритму [DSA](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187739&displayformat=dictionary)?

7.     Яка функція [CryptoAPI](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=8932&eid=187736&displayformat=dictionary) використовується для створення цифрового підпису?