МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

# Методичні вказівки

до виконання лабораторних робіт з дисципліні «Технології захисту інформації»

для студентів спеціальності

122 – Комп’ютені науки

# Лабораторна робота №1 Симетричне шифрування.

**Мета:** Ознайомитись з бібліотекою OpenSSL. Навчитися шифрувати повідомлення за допомогою бібліотеки OpenSSL.

I. Теоретичні відомості:

OpenSSL – це система захисту та сертифікації даних, назва SSL перекладається як система безпечних сокетів (secure socket layer). OpenSSL використовується практично всіма мережевими серверами для захисту інформації, що передається. Існує програмне API SSL (SSLEAY), що дозволяє створювати безпечні сокети з шифрацією даних, що передаються у власних розробках.

Т.к. OpenSSL підтримує дуже багато різних стандартів сертифікації, шифрування, хешування, використання даної команди досить складно. Всередині OpenSSL існують окремі компоненти, що відповідають за ту чи іншу дію, для отримання списку доступних компонентів можна викликати openssl із параметрами list-standart-commands. Можна також отримати список доступних алгоритмів хешування (list-message-digest- commands) та алгоритмів шифрування (list-cipher-commands). Існує також кілька допоміжних утиліт ssl: — openssl speed [список\_алгоритмов\_хешування\_або шифрування]: тестування швидкості різних алгоритмів, якщо запускати без параметрів, тестуються всі алгоритми; алгоритми всередині списку поділяються пробілом, наприклад: openssl speed md5 rsa idea blowfish des 3des sha1. Наприкінці роботи виводиться загальна швидкість роботи різних алгоритмів (1000-х байт на секунду), для обробки різної довжини блоків.

# Тепер розглянемо компоненти openssl, які виконують шифрування даних.

Шифрування файлу.

$ openssl enc -aes-256-cbc -salt -in **file.txt** -out **file.txt.enc**

|  |  |
| --- | --- |
| **Опції** | **Опис** |
| openssl | Утиліта командного рядка OpenSSL. |
| enc | Шифрування за допомогою симетричних алгоритмів шифрування. |
| -aes-256-cbc | Алгоритм шифрування, що використовується. |
| -salt | Збільшує стійкість шифрування. |
| -in | Вказує вхідний файл. |
| -out | Вказує вихідний файл. |

256bit AES – це криптографічний алгоритм, який використовує уряд Сполучених Штатів для шифрування інформації на найтаємнішому рівні.

Опція **-salt** (сіль) повинна використовуватися **ЗАВЖДИ** , якщо секретний ключ

формується з урахуванням пароля.

Без використання опції **-salt** існує можливість проведення високоефективної атаки за словником, що призведе до розкриття зашифрованих

даних. Причина цього полягає в тому, що без додавання 'солі', той самий пароль завжди генерує один і той же секретний ключ.

Коли використовується сіль, перші 8 байтів резервуються під неї. Вона генерується випадково при шифруванні файлу і зчитується з зашифрованого файлу під час дешифрації.

формується з урахуванням пароля.

Без використання опції **-salt** існує можливість проведення високоефективної атаки за словником, що призведе до розкриття зашифрованих

даних. Причина цього полягає в тому, що без додавання 'солі', той самий пароль завжди генерує один і той же секретний ключ.

Коли використовується сіль, перші 8 байтів резервуються під неї. Вона генерується випадково при шифруванні файлу і зчитується з зашифрованого файлу під час дешифрації.

Розшифровка файлу

$ openssl enc -aes-256-cbc **-d** -in **file.txt.enc** -out **file.txt**

|  |  |
| --- | --- |
| **Опції** | **Опис** |
| -d | Розшифровує дані. |
| -in | Вказує зашифрований файл. |
| -out | Вказує, куди зберегти розшифрований файл. |

Завантажте портативну версію бібліотеки OpenSSL. Завантажений архів розархівуйте в корінь робочого диска (наприклад, D:\). В результаті в корені диска має з'явитися папка OpenSSL. Командний процесор бібліотеки OpenSSL знаходиться в піддиректорії bin.

ІІ. Завдання

1. Дізнатися інформацію про бібліотеку OpenSSL, доступні алгоритми симетричного шифрування.
2. Створити резюме та засобами біліотеки OpenSSL виконати його шифрування та дешифрування за допомогою одного з алгоритмів симетричного шифрування.
3. Вихідне повідомлення, ключ, шифроване повідомлення включити до звіту лабораторної роботи.
4. Крім того, необхідно визначити список доступних алгоритмів шифрування та ЦП, а також визначити швидкість їхньої роботи на даній ЕОМ. Отримані відомості також включати до звіту.

IV. Звіт

Звіт повинен надаватися в електронному (DOC, RTF) вигляді та повинен містити титульний лист, завдання, результати роботи.

# Лабораторна робота №2 Хешування

**Мета:** Ознайомитись з бібліотекою OpenSSL. Навчитися обчислювати хеші повідомлень за допомогою бібліотеки OpenSSL

I. Теоретичні відомості:

Т.к. OpenSSL підтримує дуже багато різних стандартів сертифікації, шифрування, хешування, використання даної команди досить складно. Всередині OpenSSL існують окремі компоненти, що відповідають за ту чи іншу дію, для отримання списку доступних компонентів можна викликати openssl із параметрами list-standart-commands. Можна також отримати список доступних алгоритмів хешування (list-message-digest- commands) та алгоритмів шифрування (list-cipher-commands). Існує також кілька допоміжних утиліт ssl: — openssl speed [список\_алгоритмов\_хешування\_або шифрування]: тестування швидкості різних алгоритмів, якщо запускати без параметрів, тестуються всі алгоритми; алгоритми всередині списку поділяються пробілом, наприклад: openssl speed md5 rsa idea blowfish des 3des sha1. Наприкінці роботи виводиться загальна швидкість роботи різних алгоритмів (1000-х байт на секунду), для обробки різної довжини блоків.

# Тепер розглянемо компоненти openssl, що виконують хешування даних.

Для обчислення хешів (їх ще називають відбитками або контрольними сумами) використовується команда openssl dgst-hashalg або коротка

форма openssl hashalg (перша команда може також виконувати маніпуляції з ЕЦП, але про це далі). Звичайне використання цієї команди таке: openssl hashalg [-c] file[s] Обчислюється хеш-повідомлення фіксованої довжини у вигляді одного рядка або, якщо вказано опцію -c, рядка, розділеного на пари

HEX-чисел двокрапкою. З алгоритмів хешування можуть застосовуватися такі: md2 (128 біт), md4 (128 біт), md5 (128 біт), mdc2 (128 біт), sha (160 біт), sha1 (160 біт), ripemd160 (160 біт). Наведемо кілька прикладів: \* обчислимо md5-хеш файлу: # openssl md5 -c file MD5(file)= ac:94 \* а тепер SHA1-хеш цього ж файлу: # openssl sha1 file SHA1(file)= 13f2b3abd8a7add2f3025d89593a0327a8eb83af Завантажте портативну версію бібліотеки OpenSSL. Завантажений архів розархівуйте в корінь робочого диска (наприклад, D:\). В результаті в корені диска має з'явитися папка OpenSSL. Командний процесор бібліотеки OpenSSL знаходиться в піддиректорії bin.

ІІ. Завдання

1. Дізнатися інформацію про доступні алгоритми хешування.
2. Створити резюме та засобами біліотеки OpenSSL отримати для нього хеші як мінімум 3 алгоритми.
3. Дайджести включити до звіту з лабораторної роботи.

IV. Звіт

Звіт повинен надаватися в електронному (DOC, RTF) вигляді та повинен містити титульний лист, завдання, результати роботи.

# Лабораторна робота №3 Шифрування із відкритим ключем.

**Мета:** познайомитися з реалізованими в бібліотеці OpenSSL асиметричними алгоритмами шифрування та цифрової підпси. Навчитися шифрувати/дешифрувати повідомлення за допомогою асиметричних алгоритмів шифрування (на прикладі RSA).

1. Теоретичні відомості:

Для шифрації та дешифрації RSA алгоритмом використовується програма rsautl. Дана утиліта має також можливість підписувати та перевіряти підпис повідомлень (проте працювати все одно доводиться з хешем повідомлення, тому що підписувати можна тільки невеликий обсяг даних, тому краще застосовувати openssl dgst).

Необхідно створити відкритий та закритий ключі. Перша команда згенерує закритий ключ private.pem, друга створить відкритий ключ

public.pem

Для шифрації/дешифрації використовується наступний синтаксис: openssl rsautl -in file -out file.cr -keyin public.pem -pubin -encrypt (Шифрація "file" з використанням публічного ключа "public.pem") openssl rsautl -in file.cr - out file -keyin private.pem -decrypt (Дешифрація "file.cr" з використанням секретного ключа "public.pem")

ІІ. Завдання

* 1. Створити текстовий файл resume.txt, який містить Ваше резюме. (Скористатися будь-яким текстовим редактором).
	2. Створити секретний та відкритий ключі з використанням алгоритму RSA засобами бібліотеки OpenSSL.
	3. Зашифрувати критим ключем, а потім розшифрувати закритим ключем, файл resume.txt.

IV. Звіт

Звіт повинен надаватися в електронному (DOC, RTF) вигляді та повинен містити титульний лист, завдання, результати роботи, а саме:

1. згенерований секретний ключ;
2. згенерований октритий ключ;
3. протокол їхньої генерації за допомогою команд OpenSSL;
4. протокол шифрування/дешифрування файлу resume.txt за допомогою команд OpenSSL;

V. Контрольні питання:

1. У чому полягає схема асиметричного шифрування?
2. Які алгоритми асиметричного шифрування ви знаєте? Що впливає на їхню криптостійкість? Які недоліки у алгоритмів асиметричного шифрування?
3. Які команди OpenSSL виконують алгоритми асиметричного шифрування?

# Лабораторна робота №4

**Електронний цифровий підпис**

**Мета:** навчитися підписувати повідомлення за допомогою ЕЦП і верефікувати підписане повідомлення.

Теоретичні відомості:

Утиліта openssl dgst може використовуватися для підписування повідомлення секретним ключем та перевірки ЕЦП публічним ключем. Для цього використовується наступний синтаксис: openssl dgst -sign private\_key -out signature -hashalg file[s] Підписування file за допомогою секретного ключа "private\_key", використовуючи алгоритм хешування "hasalg" (зазвичай застосовуються sha1 або md5). openssl dgst -signature signature -verify public\_key file[s] Перевірка підпису в "file", використовуючи публічний ключ "public\_key" та ЕЦП "signature". Ця програма виводить "Verification OK" при правильному підписі або "Verification Failure" у будь-якому іншому

випадку. Врахуйте, що ЕЦП у разі зберігається окремо від файла, який нею підписаний (причому у якомусь кривому форматі).

ІІ. Завдання

1. Вихідний файл resume.txt підписати за допомогою секретного ключа, а потім верифікувати підпис за допомогою відкритого ключа.

IV. Звіт

Звіт повинен надаватися в електронному (DOC, RTF) вигляді та повинен містити титульний лист, завдання, результати роботи, а саме:

1. згенерований секретний ключ;
2. згенерований октритий ключ;
3. протокол їхньої генерації за допомогою команд OpenSSL;
4. протокол шифрування/дешифрування файлу resume.txt за допомогою команд OpenSSL;
5. протокол створення/верифікації ЕЦП файлу resume.txt за допомогою команд OpenSSL;

**Лабораторна робота №5**

**Електронно-цифровий підпис**

|  |  |
| --- | --- |
| ***Мета лабораторної роботи:*** | **1) Ознайомитись з основними принципами роботи центру****можливостями для роботи із сертифікатами. 3) Ознайомит з стресами бібліотеки OpenSSL .** |

**Зміст роботи.**

**I. Теоретичні відомості** :

Всі необхідні теоретичні відомості щодо роботи Центру Сертифікації та життєздатного циклу сертифіката представлені CA.pdf

Всі необхідні для виконання лабораторної роботи теоретичні відомості з основ використання бібліотеки OpenSSL представлені в OpenSSL3.doc

Завантажте портативну версію бібліотеки OpenSSL. Завантажений архів розархівуйте в корінь робочого диска (наприклад, D:\). В результаті в корені диска має з'явитися папка OpenSSL . Командний процесор

бібліотеки OpenSSL знаходиться в піддиректорії bin.

# ІІ. Завдання

1. Використовуючи команди бібліотеки OpenSLL, змоделювати роботу центру сертифікації
2. Показати на конкретному прикладі повний життєвий цикл сертифіката та його використання для цифрового підпису документів.

# IV. Звіт

Звіт повинен надаватися в електронному (DOC, RTF) вигляді та повинен містити титульний лист, завдання, результати роботи.