НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. Ігоря

СІКОРСЬКОГО»

НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ №4

З дисципліни «Методи реалізації криптографічних механізмів»

«Дослідження особливостей реалізації існуючих програмних систем, які використовують криптографічні механізми захисту інформації.»

Виконав:

студент групи ФБ-41мн

Білик Д. М.

1 Мета роботи

Отримання практичних навичок побудови гібридних криптосистем.

2 Хід роботи

Надійність криптопровайдера визначається не лише криптостійкістю еліптичних кривих, а й комплексною безпекою системних механізмів зберігання та обробки даних.

Класифікація загроз та системних вразливостей:

1. **Прямий доступ до ключової інформації:**

   Основна загроза полягає у фізичному або логічному викраденні приватного ключа з файлової системи. Відсутність жорстких прав доступу (наприклад, chmod 600) до файлів формату .pem дозволяє стороннім процесам або користувачам з високими привілеями скомпрометувати систему.

2. **Детермінізм генератора випадкових чисел (RNG):**

   Унікальність параметра k є критичною для ECDSA. Якщо зловмисник маніпулює джерелами ентропії ОС або передбачає стан ГВЧ, це призводить до повторного використання k. У такому разі приватний ключ обчислюється за допомогою простої арифметики.

3. **Витоки через сторонні канали (Side-channel attacks):**

   Навіть високорівневі бібліотеки на кшталт PyCryptodome використовують скомпільовані C-модулі для обчислень. Це створює ризики аналізу часу виконання (timing attacks) або поведінки процесорного кешу, що дозволяє відновити секретні біти ключа під час операцій підпису.

4. **Ін'єкції та підміна коду (Runtime Interception):**

   Використання механізму LD_PRELOAD у Linux дозволяє зловмиснику перехоплювати виклики стандартних функцій. Це дає змогу підмінити легітимну бібліотеку шкідливою, яка буде дублювати ключі або підписи на сервер зловмисника ще до моменту їх легітимного використання.

5. **Інспекція адресного простору пам'яті:**

   Якщо ОС не блокує доступ до пам'яті процесів (наприклад, через інтерфейс /proc/[pid]/mem або системний виклик ptrace), атакуючий може «витягнути» приватний ключ d або проміжні параметри безпосередньо з RAM у момент виконання криптографічних перетворень.

3 Хід роботи

Було обрано атаку за допомогою механізму ptrace

Було реалізовано програму, яка генерує ключову пару ECDSA та зупиняється утримуючи ключ в пам’яті

```
from Crypto.PublicKey import ECC
from Crypto.Hash import SHA256
from Crypto.Signature import DSS
from base64 import b64encode, b64decode


def generate_keys(curve_name: str = "P-256"):
    private_key = ECC.generate(curve=curve_name)
    public_key = private_key.public_key()
    return private_key, public_key


        print("Status: FAILED — integrity violation detected.")
def hold_sigstop(private_key):


    pid = os.getpid()
    d_value = int(private_key.d)


    print(f"\nPID = {pid}")
    print(f"d = {hex(d_value)}")
    print(f"Stopped\n", flush=True)
    os.kill(pid, signal.SIGSTOP)


def main():
    message = b"Lab: ECDSA signing example (Linux, Python)."
```

```
    private_key, public_key = generate_keys(curve_name="P-256")


    hold_sigstop(private_key)
```

```
(.venv) friedreich@friedreich:~/Documents/Labs$ /home/friedreich/Documents/Labs/

 PID = 43016
 d = 0x2abb920479c6b82691d7548e6059c0bfc514155e7a07ec09c4dc81330ebb88af
 Stopped
```

При спробі під’єднання до процесу спрацьовує політика безпеки, що забороняє під’єднуватись до процесів

```
Could not attach to process.  If your uid matches the uid of the target
process, check the setting of /proc/sys/kernel/yama/ptrace_scope, or try
again as the root user.  For more details, see /etc/sysctl.d/10-ptrace.conf
ptrace: Operation not permitted.
```

Тому для демонстрації вразливості тимчасово вимикаю захист

```
friedreich@friedreich:~$ cat /proc/sys/kernel/yama/ptrace_scope
1
friedreich@friedreich:~$ sudo sysctl -w kernel.yama.ptrace_scope=0
[sudo] password for friedreich:
kernel.yama.ptrace_scope = 0
```

І маю можливість під’єнатись до процесу, перевірити адресний простір Python, провести пошук у купі або створити дамп пам’яті, що містить повний знімок оперативної пам’яті, включаючи потенційні секретні дані.

```
(gdb) generate-core-file  /home/friedreich/Documents/Labs/MPKM/Lab4/dump.core
warning: Memory read failed for corefile section, 4096 bytes at 0xfffffffff600000.
Saved corefile /home/friedreich/Documents/Labs/MPKM/Lab4/dump.core
```

Аналіз такого дампу дозволяє відновити внутрішній стан програми та продемонструвати ризики витоку криптографічних секретів.

4 Висновок

Реальна безпека асиметричних криптосистем визначається не тільки їх математичною моделлю, але й специфікою реалізації, зокрема механізмами керування пам’яттю на відповідній програмно-апаратній платформі.