

Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

КРИПТОГРАФІЯ
КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

Експериментальна оцінка ентропії на символ джерела відкритого
тексту

Виконали:

Студенти 3 курсу

Остапова О. А.

Литвин М. Р.

Київ – 2025

Мета роботи

Ознайомлення з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної крипtosистеми типу RSA; практичне ознайомлення з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.

Постановка задачі

Реалізувати крипtosистему RSA на мові Python, яка забезпечує безпечний обмін інформацією між двома абонентами. Система повинна виконувати наступні функції: генерацію великих 256-бітних простих чисел із застосуванням тесту пробних ділень та тесту Міллера-Рабіна; формування ключових пар для абонентів А та В; шифрування та розшифрування повідомень; створення та перевірку цифрового підпису із використанням хеш-функції SHA-256; а також реалізацію протоколу конфіденційної передачі ключів із автентифікацією для забезпечення достовірності та конфіденційності обміну.

Варіант 10

Хід роботи

Основні формули, які використовувалися при виконані роботи:

1. Генерація простих чисел:
 - шукаємо p , q такими що:
 $n = p \cdot q$
 p, q – прості
 - перевірка: тест Міллера-Рабіна
2. Функція Ейлера:
 $\phi(n) = (p - 1)(q - 1)$
3. Вибір відкритого експонента:
 $2 < e < \phi(n)$
 $\text{gcd}(e, \phi(n)) = 1$
4. Секретний експонент:
 $d \equiv e^{-1} \pmod{\phi(n)}$
 $e \cdot d \equiv 1 \pmod{\phi(n)}$
5. Відкритий ключ: (n, e)
Приватний ключ: (d, p, q)

Генерація двох ймовірних прости числа p і q за допомогою тесту Міллера-Рабіна:

Число 64008251336882740207846060329148097258456861435818959414505150783227163059847 не пройшло тест Міллера-Рабіна
Число 7530490029584527303831831179308028536302151807643041461685724534578510000833 не пройшло тест Міллера-Рабіна
Число 106717622309333937542354861749153299428850099512129516680489205773944582142297 не пройшло тест Міллера-Рабіна
Число 7557717607447184103126670417990673703079476102787561773651532317071230872733 не пройшло тест Міллера-Рабіна
Число 100523306829146255947228275838915904229800891614913359673557500609414932663089 є ймовірно прости (пройшло Міллера-Рабіна)
Число 11212556849712993524974382829045008943960097786682848902142160795320166549999 не пройшло тест Міллера-Рабіна
Число 67520388863873814305604770459775274490353681819309789274082633628090243603009 не пройшло тест Міллера-Рабіна
Число 93353745628309843510583617024147563207296909345817155591100318608611375338197 не пройшло тест Міллера-Рабіна
Число 59763449617338679511788724901360182215185035395417037354383294576307068917373 є ймовірно прости (пройшло Міллера-Рабіна)
Число 110419879863631927414791378553354848950869488625577170061855197259495209252501 не пройшло тест Міллера-Рабіна
Число 62142947553432873966091371032767514633643498580929973656531447899923964427287 не пройшло тест Міллера-Рабіна

Генерація ключових параметрів крипtosистеми RSA для абонентів А та В:

```
== Ключі абонента А ==
pA = 7396249592097984852327170529390095766479343850668377658290164069477742481959
qA = 11065247737564661840140263532260790661493520078667752415834657072779373816313
nA = 8184133414411975958577384174136302732643289939990380435180454321058694346687116747441202512278216957366251369649990495340519856371663780403024230382397167
eA = 65537
dA = 204612701214795841093260966619197278292201818311400252422954581458636658483704463287692031499065756497507418777882338050880188810114065037446574178937553

== Ключі абонента В ==
pB = 108851338398001752174093430842941016382183454786228874965282401048023844587227
qB = 8523223330656045571386172504091689189729164582297648805341923667388390723889
nB = 9277642670069847980065780300472382291393496574749875318241430849917108839559905057130694971537854913278076423632896761200112892344791750453574928933165803
eB = 65537
dB = 7175001008435237728636556582528071532981185901011988352072091813741836587336075666934157344804920049826284511057669751861031897054681092462503864142607489
```

Шифрування та дешифрування повідомлення:

```
== Тест шифрування ==
M = 336354809486859290194844111029360681775003220354227789377586672673741035147056477931940370072634779590162852177763305555668141895106557157478236560144985
C = 1534553308403930249354926149116033067277789446777096499645562967386073141043354668663197208884700123104965950686830957457655491406410985242108350657812793
Розшифровано: 336354809486859290194844111029360681775003220354227789377586672673741035147056477931940370072634779590162852177763305555668141895106557157478236560144985
```

Перевірка цифрового підпису:

```
== Тест цифрового підпису ==
Підпис валідний? True

== Протокол передачі ключа А → В ==
Отримано ключ k = 701432759051636959126881841587447335489836309377445065761032818289273090676636886762000477022331022781842635318028730815248805481558658370534409086884602, Підпис валідний? True
```

Висновок

Метою роботи було створення крипtosистеми RSA на Python для безпечноого обміну інформацією між двома абонентами. Було опрацьовано теоретичні основи, зокрема піднесення до степеня за модулем за схемою Горнера, поняття псевдопростих чисел та ймовірнісні тести Ферма, Соловея-Штрассена і Міллера-Рабіна. На їх основі реалізовано генерацію 256-бітних простих чисел і функції для створення ключових пар RSA, що забезпечують шифрування та розшифрування повідомень.

Крім того, реалізовано цифровий підпис із SHA-256 для автентифікації та перевірки цілісності повідомлень, а також протокол конфіденційної передачі ключів, який поєднує шифрування та перевірку підпису для підтвердження справжності відправника. Практична реалізація охопила всі ключові компоненти RSA, включно з перевіркою чисел на простоту, генерацією ключів, шифруванням і розшифруванням, підписом та протоколом безпечної передачі ключів, що продемонструвало ефективне застосування асиметричної криптографії.