Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня бакалавра

студента *Довбиш Ольги Андріївни*

академічної групи *125-19-1*

спеціальності *125 Кібербезпека*

спеціалізації[1]

за освітньо-професійною програмою *Кібербезпека*

на тему *Дослідження алгоритмів криптографічного захисту інформації в електронних системах зберігання і передачі даних*

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	к.т.н., доц. Герасіна О. В.			
розділів:				
спеціальний	к.т.н., доц. Герасіна О. В.			
економічний	к.е.н., доц. Пілова Д.П.	90	відмінно	

Рецензент				

Нормоконтролер	ст. викл. Мєшков В. І.			

Дніпро
2023

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
_____ д.т.н., проф. Корнієнко В.І.

«_____»_____20___року

ЗАВДАННЯ
на кваліфікаційну роботу ступеня бакалавра

студенту *Довбиш Ольга Андріївна* **академічної групи** 125-19-1
(прізвище та ініціали) (шифр)

спеціальності *125 Кібербезпека*

спеціалізації _____

за освітньо-професійною програмою *Кібербезпека*

на тему *Дослідження алгоритмів криптографічного захисту інформації в електронних системах зберігання і передачі даних*

затверджену наказом ректора НТУ «Дніпровська політехніка» від 16.05.23 № 350-с

Розділ	Зміст	Термін виконання
Розділ 1	Аналіз загальних відомостей про криптографію, принципи роботи та приклади алгоритмів криптографічного захисту інформації AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS.	04.04.2023 – 30.04.2023
Розділ 2	Порівняння алгоритмів криптографічного захисту інформації AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS.	30.04.2023 – 20.05.2023
Розділ 3	Розрахунки капітальних витрат, економічного ефекту та терміну окупності капітальних інвестицій застосування запропонованих рішень	21.05.2023 – 09.06.2023

Завдання видано _____ Герасіна О.В.
(підпис керівника) (прізвище, ініціали)

Дата видачі завдання: _____

Дата подання до екзаменаційної комісії: _____

Прийнято до виконання _____ Довбиш О. А.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка 90 ст., 7 рис., 10 табл., 12 джерел, 4 додатка.

Об’єкт дослідження: процес криптографічного захисту інформації.

Предмет дослідження: алгоритми шифрування AES, DES, RSA, ECC; алгоритми цифрового підпису DSA, ECDSA; алгоритми аутентифікації HMAC, SSL/TLS.

Мета кваліфікаційної роботи: дослідження криптографічних алгоритмів шифрування, аутентифікації та цифрового підпису для їх використання в цифрових системах обробки і передачі інформації.

У першому розділі проаналізовано загальні відомості про криптографію, наведено принципи роботи та приклади алгоритмів криптографічного захисту інформації AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS.

У другому розділі представлене порівняння алгоритмів криптографічного захисту інформації AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS. За наслідками досліджень зроблено висновки щодо рішення поставленої задачі.

У економічному розділі виконані розрахунки капітальних витрат, економічного ефекту та терміну окупності капітальних інвестицій застосування запропонованих рішень.

Практична значимість роботи полягає в дослідженні алгоритмів шифрування і цифрового підпису для визначення найкращого з них для покращення якості безпеки в інформаційних системах підприємств.

КРИПТОГРАФІЯ, АЛГОРИТМИ ШИФРУВАННЯ, ЗАХИСТ ІНФОРМАЦІЇ, АУТЕНТИФІКАЦІЯ, ЦИФРОВИЙ ПІДПИС, СИСТЕМА ОБРОБКИ І ПЕРЕДАЧІ ІНФОРМАЦІЇ

ABSTRACT

Explanatory note of the 90th article, 7 figures, 10 tables, 12 sources, 4 appendices.

The object of research: the process of cryptographic protection of information.

Research subject: encryption algorithms AES, DES, RSA, ECC; DSA, ECDSA digital signature algorithms; authentication algorithms HMAC, SSL/TLS.

The purpose of the qualification work: research of cryptographic algorithms for encryption, authentication and digital signature for their use in digital systems of information processing and transmission.

In the first section, general information about cryptography is analyzed, principles of operation and examples of cryptographic information protection algorithms AES, DES, RSA, ECC are given; DSA, ECDSA; HMAC, SSL/TLS.

The second section presents a comparison of cryptographic information protection algorithms AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS. Based on the results of the research, conclusions were made regarding the solution to the problem.

In the economic section, calculations of capital costs, economic effect and payback period of capital investments of the proposed solutions are made.

The practical significance of the work lies in the study of encryption and digital signature algorithms to determine the best of them to improve the quality of security in information systems of enterprises.

CRYPTOGRAPHY, ENCRYPTION ALGORITHMS, INFORMATION PROTECTION, AUTHENTICATION, DIGITAL SIGNATURE, INFORMATION PROCESSING AND TRANSMISSION SYSTEM

СПИСОК УМОВНИХ СКОРОЧЕНЬ

AES – Advanced Encryption Standard – розширений стандарт шифрування;

DES – Data Encryption Standard – стандарт шифрування даних;

DSA – Digital Signature Algorithm – алгоритм цифрового підпису;

ECC – Elliptic Curve Cryptography – криптографія еліптичної кривої;

ECDSA – Elliptic Curve Digital Signature Algorithm – алгоритм цифрового підпису еліптичної кривої;

FTP – File Transfer Protocol – протокол передачі файлів;

HMAC – Hash-based Message Authentication Code – код автентифікації повідомлення на основі хешу;

HTTP – HyperText Transfer Protocol – протокол передачі гіпертексту;

IoT – Internet of things – інтернет речей;

MD5 – Message Digest 5 – дайджест повідомлень 5;

RSA – Rivest-Shamir-Adleman – рівест-Шамір-Адлеман;

SHA-256 – Secure Hash Algorithm 256-bit – Алгоритм безпечного хешування 256 біт;

SSL/TLS – Secure Sockets Layer / Transport Level Security – рівень захищених розеток / безпека транспортного рівня;

SEC – Standards for Efficient Cryptography – стандарти ефективної криптографії;

VPN – virtual private network – віртуальна приватна мережа;

ЗМІСТ

ВСТУП

У світі, де інформація є однією з найцінніших ресурсів, виникає необхідність в захисті цієї інформації від небажаних очей. Електронні системи зберігання і передачі даних стають основою для обміну інформацією у різних сферах життя, починаючи від особистих комунікацій до великомасштабних підприємств і державних організацій. Але як забезпечити конфіденційність, цілісність і доступність цих даних у світі, де кіберзагрози постійно зростають?

Один з основних інструментів, який використовується для захисту інформації, є криптографія - наука про застосування математичних алгоритмів для шифрування та розшифрування даних. Дослідження алгоритмів криптографічного захисту інформації в електронних системах зберігання і передачі даних стають ключовими для розробки надійних та ефективних методів захисту. Ці алгоритми криптографічного захисту інформації мають на меті забезпечити конфіденційність даних шляхом їх зашифрування Проте, у світі постійного розвитку технологій і злочинної діяльності, важливо постійно вдосконалювати алгоритми криптографічного захисту інформації. Дослідження в цій галузі дозволяють виявляти нові загрози, розробляти і вдосконалювати алгоритми шифрування та аутентифікації, а також забезпечувати стійкість цих алгоритмів до атак. У даній роботі буде розглянуто актуальні дослідження в галузі алгоритмів криптографічного захисту інформації в електронних системах зберігання і передачі даних. Буде досліджено різноманітні методи шифрування, включаючи симетричні та асиметричні алгоритми, а також протоколи аутентифікації та цифрові підписи. Крім того, зростаючий обсяг інформації, який зберігається та передається в електронних системах, ставить перед нами вимогу до високошвидкісних криптографічних рішень.

Данне дослідження проводиться з метою вдосконалення інформаційних систем, які використовуються компаніями, для передачі, зберігання та обробки конфіденційної інформації. Багато компаній недооцінюють значимість

проведення заходів, направлених на вдосконалення захисту цифрових даних, але потенційні збитки, які може зазнати компанція через відсутність проведення подібних досліджень, можуть бути критичними, та призвести компанію до банкротства. В данній роботі будуть розглянуті такі алгоритми шифрування як AES, DES, RSA та ECC. Також будуть розглянуті алгоритми цифрового підпису такі як DSA та ECDSA, а також алгоритми аутентифікації HMAC та SSL/TLS. Це необхідно для визначення кращого алгоритму для захисту цифрової системи. Отже, дане дослідження є необхідним заходом на вдосконалення інформаційних систем компаній, в яких зберігається, передається та оброблюється конфіденційна інформація, та є превентивною мірою від атаки зловмисників.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ АЛГОРИТМІВ КРИПТОГРАФІЧНОГО ЗАХИСТУ

1.1 Загальні відомості про криптографію

Криптографія – це наука про захист інформації шляхом її шифрування та дешифрування. У світі, де цифрові дані поширюються все більше і більше, криптографія стає надзвичайно важливою для забезпечення конфіденційності, цілісності та доступності інформації. Одним з основних завдань криптографії є шифрування даних. Шифрування перетворює звичайний текст (зрозумілий для людей) в криптограму (зашифрований текст), який стає незрозумілим без наявності ключа дешифрування. Існує багато різних методів шифрування, включаючи симетричні алгоритми, де ключ для шифрування і дешифрування однаковий, та асиметричні алгоритми, де використовуються пари ключів: публічний і приватний. Публічний ключ може бути доступним для всіх, тоді як приватний ключ залишається секретним і використовується лише власником. Асиметрична криптографія дозволяє забезпечити безпеку передачі даних через незахищені канали шляхом підписування цифрових підписів або використання протоколів обміну ключами. Крім шифрування, криптографія також займається іншими аспектами безпеки, такими як аутентифікація, цифровий підпис, контроль цілісності даних та безпечні протоколи передачі інформації. Застосування криптографії включають забезпечення безпеки електронної пошти, онлайн-транзакцій, мобільних комунікацій, хмарних обчислень та багатьох інших областей. Криптографія також має велике значення в урядових структурах, армії та розвідувальних організаціях для захисту державної та військової інформації. Проте, криптографія не є статичною наукою. Вона постійно еволюціонує, адаптуючись до нових технологій і загроз. Зловмисники постійно намагаються знайти вразливості у криптографічних алгоритмах, тому безперервний розвиток нових методів шифрування та захисту є надзвичайно важливим. Загальні відомості про криптографію дають нам можливість

розуміти основні принципи та методи, які використовуються для захисту нашої цифрової інформації. Знання криптографії допомагає нам зберегти конфіденційність наших даних і забезпечити безпеку в цифровому світі

1.1.1 Основні поняття та терміни: шифр, ключ, алгоритм, протокол, криптоаналіз.

У сучасному світі, де безпека і конфіденційність інформації мають вирішальне значення, розуміння основних понять і термінів в галузі криптографії є надзвичайно важливим. Розглянемо основні поняття і терміни, що становлять основу цієї науки. Розуміння цих основних понять і термінів є важливим кроком для розуміння криптографії і розробки ефективних методів захисту інформації. Детальне дослідження цих концепцій дозволить краще оцінити сучасні криптографічні системи та впроваджувати заходи для забезпечення безпеки даних. Наголос на вивченні основних понять і термінів криптографії стає особливо важливим у контексті сучасного цифрового світу, де обмін інформацією відбувається на різних рівнях та в різних сферах життя. Зростаючі загрози кібератак, спроби несанкціонованого доступу до конфіденційної інформації та широкий спектр криптоаналітичних методів підкреслюють необхідність міцного криптографічного захисту. В сучасному світі криптографія використовується не тільки у фінансовій, комерційній та військовій сферах, але й у буденному житті. Наприклад, шифрування з'єднання забезпечує безпеку електронної комунікації, шифрування даних на переносних пристроях забезпечує конфіденційність особистої інформації, а шифрування файлів та баз даних забезпечує їхню цілісність та недоступність для зловмисників. Розвиток криптографічних алгоритмів та протоколів є постійним процесом, оскільки криптографічні схеми, які були безпечними в минулому, можуть стати вразливими через нові технології та атаки. Тому дослідження і розвиток криптографії є постійним завданням для спеціалістів у цій галузі.

Наша мета полягає в тому, щоб розуміти принципи та механізми криптографії, щоб розробляти надійні алгоритми та протоколи, які забезпечують безпеку та захист інформації. Дослідження алгоритмів криптографічного захисту в електронних системах зберігання і передачі даних стає важливим фундаментом для забезпечення безпеки в цифровому світі

Шифр – це метод або алгоритм перетворення інформації у незрозумілу форму з метою забезпечення конфіденційності та захисту від несанкціонованого доступу. Шифрування дозволяє перетворити звичайний текст (відкритий текст) у шифрований текст (шифртекст) за допомогою певного алгоритму та ключа.

Ключ – це параметр, який використовується разом з алгоритмом шифрування для перетворення відкритого тексту у шифртекст і зворотно. Ключ визначає, як саме буде проводитись шифрування та розшифрування інформації.

Алгоритм – це точний набір інструкцій, які описують послідовність дій, необхідних для виконання певної операції. У контексті криптографії, алгоритми шифрування визначають, які конкретні перетворення будуть застосовуватись до даних для отримання шифртексту або розшифровування шифртексту у відкритий текст.

Протокол – це набір правил і процедур, які визначають, які дії повинні виконуватись між сторонами під час обміну даними. В контексті криптографії, криптографічні протоколи використовуються для забезпечення безпеки комунікації, аутентифікації сторін та обміну ключами.

Криптоаналіз – це наука про аналіз і дослідження криптографічних алгоритмів та протоколів з метою зламу або слабкого їхнього захисту. Криптоаналітики використовують різні методи і техніки, такі як атаки перебором, статистичний аналіз та аналіз вразливостей, для розкриття ключів або зламу шифру.

1.1.2 Роль криптографії в сучасному світі

Криптографія в сучасному світі відіграє надзвичайно важливу роль у забезпеченні безпеки і конфіденційності інформації. З огляду на постійний розвиток технологій та зростаючі загрози кібератак, захист даних є необхідним для збереження довіри і стабільності суспільства. Однією з основних ролей криптографії є забезпечення конфіденційності. Шифрування дозволяє перетворити звичайний текст у незрозумілу форму, що може бути прочитана лише за допомогою спеціального ключа. Це гарантує, що лише авторизовані особи зможуть отримати доступ до конфіденційної інформації, такої як фінансові дані, медична інформація, особисті повідомлення тощо. Захист персональних даних стає особливо важливим у світі, де інформація постійно передається через мережі Інтернет та зберігається в електронних системах.

Крім того, криптографія грає ключову роль у забезпеченні цілісності даних. Це означає, що інформація не може бути змінена чи пошкоджена під час передачі або зберігання без знання авторизованої сторони. Цілісність є критично важливою для переконання, що дані залишаються недоторканими і не підлягають втручанню, що може призвести до зловживань, псування інформації або порушення довіри.

Криптографія також відіграє важливу роль у забезпеченні автентичності даних. Механізми цифрового підпису і аутентифікації дозволяють перевірити, що повідомлення або документ були створені автентичною стороною і не були підроблені. Це забезпечує довіру до переданих даних, таких як електронний голос, фінансові транзакції або правові документи.

У сучасному світі, де електронна комунікація, електронна торгівля та обробка даних стають все більш поширеними, роль криптографії набуває ще більшого значення. Вона створює основу для безпеки та довіри в цифровому середовищі і допомагає забезпечити захист приватності, конфіденційності інформації.

1.2 Огляд існуючих алгоритмів криптографічного захисту інформації та їх класифікація

Одна з основних категорій алгоритмів криптографічного захисту - симетричні алгоритми. У цих алгоритмах використовується один ключ для як шифрування, так і розшифрування повідомлення. Симетричні алгоритми, такі як AES та DES (Data Encryption Standard), відомі своєю ефективністю та швидкодією. Вони широко використовуються для шифрування великих обсягів даних, таких як файли або дискові носії.

Іншою важливою категорією алгоритмів є асиметричні алгоритми, відомі також як криптографія з відкритим ключем. Вони використовують пару ключів - публічний та приватний ключі. Публічний ключ використовується для шифрування повідомлення, а приватний ключ - для розшифрування. RSA (Rivest-Shamir-Adleman) та ECC (Elliptic Curve Cryptography) є прикладами асиметричних алгоритмів, які широко застосовуються у сферах, де важлива безпека комунікації та цифровий підпис.

Крім того, існують алгоритми хешування, які використовуються для створення фіксованого розміру хеш-значення з вхідного повідомлення. Хеш-функції, такі як MD5 (Message Digest 5) та SHA-256 (Secure Hash Algorithm 256-bit), застосовуються для перевірки цілісності даних та створення цифрових підписів.

Класифікація алгоритмів криптографічного захисту також може базуватись на їхніх основних цілях, наприклад, шифрування, аутентифікація, цифровий підпис, ключовий обмін тощо.

Огляд і класифікація існуючих алгоритмів криптографічного захисту інформації допомагають розуміти їхню сутність, сильні та слабкі сторони та вибрати найбільш підходящі методи для конкретних потреб в безпеці даних. Знання про різні криптографічні алгоритми є невід'ємною частиною будь-якого

фахівця зі зберігання та передачі даних, адже воно допомагає розробляти ефективні та надійні системи захисту інформації.

1.2.1Дослідження симетричного алгоритму шифрування AES

Алгоритм шифрування AES (Advanced Encryption Standard) був впроваджений у 2001 році і став наступником попереднього стандарту шифрування DES (Data Encryption Standard). У 1997 році Національний інститут стандартів і технологій оголосив конкурс на створення нового стандарту шифрування для захисту державної інформації США. Було відібрано 15 кандидатів, серед яких були і різні варіанти AES. З 15 кандидатів, Національний інститут стандартів і технологій скоротив список до п'яти фіналістів. Ці алгоритми були піддані ретельному аналізу та оцінці їхньої ефективності та безпеки. Після ретельного вивчення і оцінки кожного з фіналістів, Національний інститут стандартів і технологій обрав алгоритм Rijndael, запропонований двома бельгійськими криптографами Жаном-Жаком Квізвелем і Венді Де Клерком, як стандартний шифр AES. AES використовує блочний шифр з фіксованою довжиною блоку 128 біт і ключем, який може бути 128, 192 або 256 бітовим. Він використовує заміни, перестановки, додавання і перемішування бітів для забезпечення безпеки і криптографічного стійкості. AES широко використовується як стандартний протокол шифрування для захисту конфіденційної інформації у різних сферах, включаючи комунікацію по мережі, зберігання даних і шифрування дисків. AES став міжнародним стандартом, прийнятим Міжнародною організацією зі стандартизації і Міжнародним електротехнічним комітетом під назвою ISO/IEC 18033-3. AES вважається одним з найбільш безпечних алгоритмів шифрування, оскільки його безпеку було ретельно вивчено і пройшло багато криптографічних атак і перевірок. AES став широко поширеним і отримав значну популярність в

криптографічних застосуваннях, включаючи протоколи TLS/SSL для захисту веб-трафіку, протоколи VPN для безпечного з'єднання та шифрування файлів і даних. AES залишається одним з основних шифрів у сучасних криптографічних системах і продовжує забезпечувати надійний рівень захисту інформації.

Основні принципи роботи алгоритму AES:

AES-алгоритм це симетричний блоковий шифр, що може зашифровувати та розшифровувати дані. Шифрування конвертує дані (впорядкований текст) до нечитаємої форми, що називається зашифрованим текстом, а процес розшифрування конвертує дані у зворотньому порядку. Криптографічний ключ має довжину 128, 192 або 256 біт, що дозволяє зберігати дані розбитими на зашифровані блоки по 128 біт. AES операційно працює з блоками даних розміром 128 біт. Шифрування та дешифрування виконуються послідовно на кожному блоку даних. Основні кроки шифрування AES включають підстановки байтів, зсуви рядків, змішування колонок та додавання ключа.

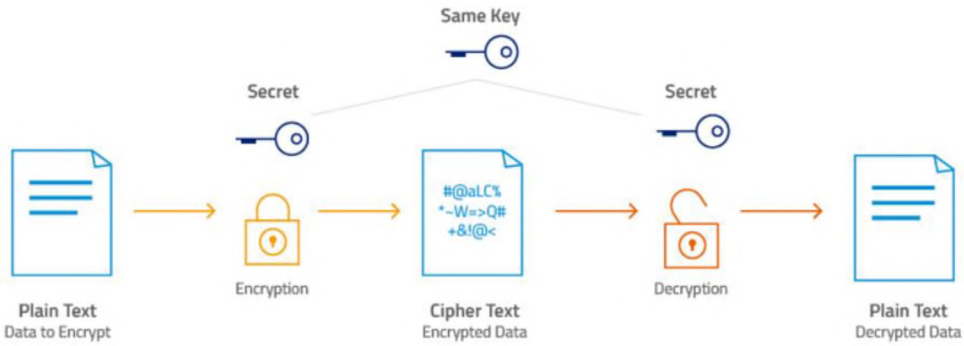

Рисунок 1.1 Схема принципу роботи симетричних блокових шифрів.

1. Розмір блоку: AES працює з блоками даних фіксованого розміру 128 біт (16 байт). Це означає, що кожні 128 біт даних шифруються окремо.

2. Ключі: AES використовує ключ для шифрування і розшифрування даних. Ключ може мати довжину 128, 192 або 256 бітів. Розмір ключа визначає рівень безпеки шифрування.

3. Підстановка байтів (SubBytes): Кожен байт замінюється на відповідний байт з S-блоку (таблиці заміни), що забезпечує необхідну нелінійність у шифруванні. Цей крок допомагає захистити шифр від криптоаналізу.

4. Зсув рядків (ShiftRows): Кожен рядок матриці зсувається циклічно вліво на деяку кількість байтів. Перший рядок залишається без зсуву, другий зсувається на 1 байт, третій - на 2 байти, а четвертий - на 3 байти. Цей крок допомагає забезпечити дифузію (розподіл інформації) у шифрі.

5. Змішування колонок (MixColumns): Кожна колонка матриці змішується за допомогою лінійних операцій над байтами. Цей крок допомагає забезпечити дифузію та запобігає лінійним атакам.

6. Додавання ключа (AddRoundKey): Кожен байт матриці об'єднується з відповідним байтом з раундового ключа. Раундовий ключ отримується з основного ключа шифрування шляхом його розширення та згортання. Ці кроки повторюються кілька раундів. Кожен раунд включає всі чотири кроки, за винятком останнього раунду, де виконується без змішування колонок.

Ці принципи використовуються разом для забезпечення високої безпеки і надійності шифрування в AES. Цей шифр широко використовується у багатьох застосуваннях, включаючи захист інформації в мережах зв'язку та зберігання даних.

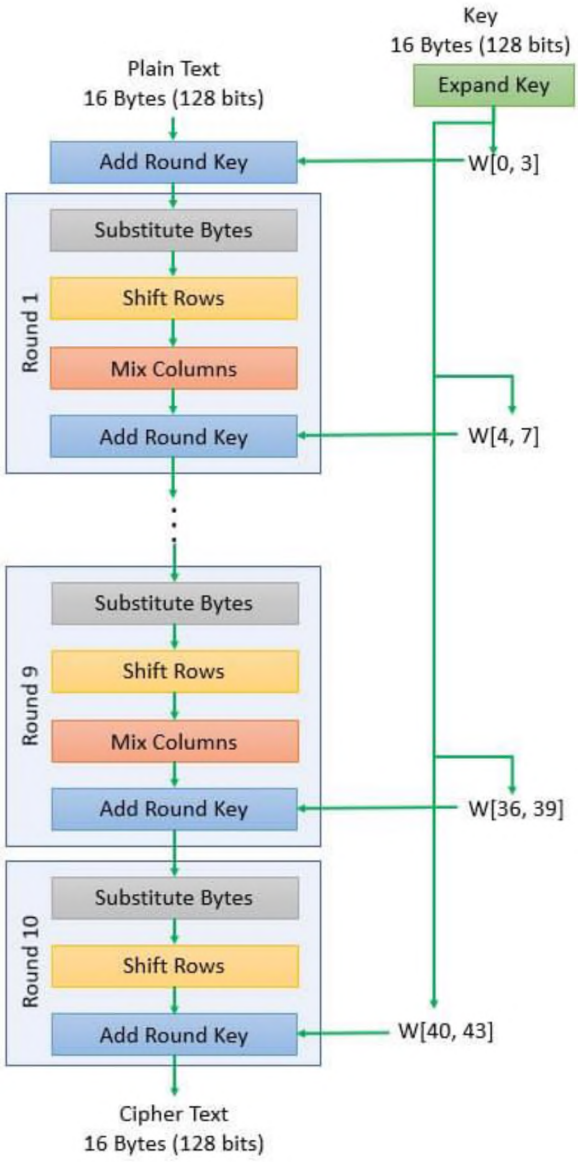

Рисунок 1.2 Приклад роботи алгоритму AES

Розшифрування:

Для розшифрування блоку даних за допомогою шифру AES слід виконати наступні кроки:

Вхідні дані:

- Зашифрований блок даних, який потрібно розшифрувати.

- Ключ, який був використаний для шифрування.

1) Розгортання ключів:

Використовуйте алгоритм Key Expansion для розгортання ключа шифрування на підключі, які будуть використовуватися для кожного раунду розшифрування.

2) Останній раунд:

- Застосуйте підстановку байтів (SubBytes) до зашифрованого блоку даних, використовуючи обернену таблицю S-Box.

- Застосуйте операцію зсуву рядків (ShiftRows) до зашифрованого блоку даних, використовуючи обернену версію операції.

- Додайте підключ до блоку даних, використовуючи операцію XOR.

3) Раунди після останнього:

- Застосуйте інверсію операції MixColumns до зашифрованого блоку даних.

- Застосуйте інверсію операції SubBytes до зашифрованого блоку даних, використовуючи обернену таблицю S-Box.

- Застосуйте інверсію операції ShiftRows до зашифрованого блоку даних, використовуючи обернену версію операції.

- Додайте підключ до блоку даних, використовуючи операцію XOR.

4) Останній розгорнутий раунд:

- Застосуйте підстановку байтів (SubBytes) до зашифрованого блоку даних, використовуючи обернену таблицю S-Box.

- Застосуйте операцію зсуву рядків (ShiftRows) до зашифрованого блоку даних, використовуючи обернену версію операції.

- Додайте підключ до блоку даних, використовуючи операцію XOR.

5) Вихідні дані:

- Отриманий розшифрований блок даних.

Ці кроки повторюються для кожного блоку даних, які потрібно розшифрувати. Зверніть увагу, що розшифрування вимагає знання правильного ключа, який був використаний для шифрування.

Приклад шифрування:

Зашифрування блоку даних "54 65 78 74" за допомогою шифру AES-128 і ключа "2B 7E 15 16" включає наступні кроки:

1. Початковий блок даних:

Блок даних "54 65 78 74" представлений у вигляді матриці 4x4:

54 65 78 74

2. Розширення ключа:

Використовуючи початковий ключ "2B 7E 15 16", генеруємо розширений ключ, який складається з 11 підключів.

3. Раунд 1:

Застосовуємо операцію XOR між початковим блоком даних і першим підключем:

54 65 78 74

XOR

2B 7E 15 16

7F 1B 6D 62

Проводимо заміну байтів (SubBytes) за допомогою S-блоку:

7F 1B 6D 62

SubBytes

---

16 AE 15 F1

Виконуємо зсув рядків (ShiftRows):

16 AE 15 F1

ShiftRows

---

16 AE 15 F1

Застосовуємо змішування стовпців (MixColumns):

16 AE 15 F1

MixColumns

8D 2A C8 58

Застосовуємо операцію XOR між отриманим результатом і другим підключем:

8D 2A C8 58

XOR

A0 88 23 2A

2D A2 EB 72

Результат раунду 1 є початковим блоком для наступного раунду.

4. Раунди 2-10:

Повторюємо кроки раунду 1 для кожного наступного раунду, використовуючи відповідні підключі.

5. Останній раунд:

Виконуємо кроки раунду 1, за винятком змішування стовпців (MixColumns) в останньому раунді.

Замість цього, просто застосовуємо операцію XOR між отриманим результатом і останнім підключем.

6. Результат шифрування:

Останній отриманий блок даних є зашифрованим блоком даних.

У цьому прикладі, зашифрований блок даних має значення:

29 C3 50 5F

Отже, блок даних "54 65 78 74" зашифровано у блок даних "29 C3 50 5F" за допомогою шифру AES-128 та ключа "2B 7E 15 16".

Це лише один блок шифрованих даних, але AES може застосовуватися до будь-якого розміру даних, розбиваючи їх на блоки і шифруючи їх незалежно один від одного. Дешифрування виконується аналогічно, протилежними операціями, включаючи інверсію кожного кроку. Це загальний опис роботи шифру AES. У реальності процес шифрування може бути більш складним, з урахуванням побітових операцій та ключових розкладань. Проте, основні принципи, описані вище, залишаються незмінними.

Приклад розшифрування:

Розшифрування блоку даних "29 C3 50 5F" за допомогою шифру AES-128 і ключа "2B 7E 15 16" включає наступні кроки:

1. Початковий зашифрований блок даних:

Блок даних "29 C3 50 5F" представлений у вигляді матриці 4x4:

29 C3 50 5F

2. Розширення ключа:

Використовуючи початковий ключ "2B 7E 15 16", генеруємо розширений ключ, який складається з 11 підключів.

3. Останній раунд:

Застосовуємо операцію XOR між початковим блоком даних і останнім підключем:

29 C3 50 5F

XOR

8D F2 D1 C6

A4 31 81 99

Виконуємо зворотню операцію заміни байтів (InvSubBytes):

A4 31 81 99

InvSubBytes

---

54 65 78 74

Виконуємо зворотні зсуви рядків (InvShiftRows):

54 65 78 74

InvShiftRows

---

54 65 78 74

Виконуємо операцію XOR між отриманим результатом і попереднім підключем:

54 65 78 74

XOR

2B 7E 15 16

7F 1B 6D 62

1. Розгортання раундів 2-10:

Повторюємо зворотні кроки для кожного наступного раунду, використовуючи відповідні зворотні підключі.

2. Розгортання першого раунду:

Виконуємо зворотні кроки першого раунду, використовуючи перший зворотний підключ.

3. Отриманий блок даних після останнього раунду є розшифрованим блоком даних.

У цьому прикладі, розшифрований блок даних має значення:

54 65 78 74

Отже, блок даних "29 C3 50 5F" розшифровано у блок даних "54 65 78 74" за допомогою шифру AES-128 та ключа "2B 7E 15 16".

1.2.2 Дослідження симетричного алгоритму шифрування DES

DES - це симетричний блочний шифр, розроблений для шифрування та дешифрування даних. DES був широко використовуваним стандартом протягом багатьох років, але сьогодні рекомендується використовувати більш сильні алгоритми, такі як AES. Шифр DES (Data Encryption Standard) є одним з найвідоміших історичних криптографічних алгоритмів. Розробка DES почалася в 1970-х роках в рамках проекту, який був ініційований Національним інститутом стандартів і технологій у США. Алгоритм був розроблений командою, на чолі зі знаменитим криптографом Шоном Богом. Початково DES називався "Lucifer" (Люцифер) - це було внутрішнє ім'я, яке давалося алгоритму під час розробки. Проте, після завершення розробки, алгоритм отримав назву "Data Encryption Standard" (стандарт шифрування даних). DES був прийнятий в 1977 році як національний стандарт шифрування у США і став використовуватись для захисту конфіденційної інформації у різних галузях, включаючи фінансову, комерційну та урядову сфери. DES використовує блочний шифр з довжиною блоку 64 біти та ключем завдовжки 56 біт. Він складається з ітеративного процесу шифрування, який включає операції перестановки, заміни та зсуву. Протягом часу DES став об'єктом критики через обмежену довжину ключа і ефективність атак. В 1997 році Національний інститут стандартів і технологій оголосив конкурс на розробку нового стандарту шифрування, що призвело до розвитку AES (Advanced Encryption Standard), який став наступником DES. DES використовувався протягом багатьох років для шифрування даних, особливо у фінансових системах та телекомунікаціях. Проте, він поступово заміщується більш сучасними алгоритмами, такими як AES. DES залишається важливим алгоритмом у світі криптографії, пам'яткою історії розвитку шифрування та основою для подальших досліджень і розробок в цій галузі.

Основні принципи роботи алгоритму DES:

1) Розширення ключа: Основний ключ DES має довжину 64 біта, але для забезпечення безпеки він розширюється до 56 байтів (64 біти включаючи контрольні біти парності). Розширення ключа включає перестановки та дублювання деяких бітів, що дає 16 раундових підключів по 48 бітів кожний.

2) Ітеративне шифрування: DES використовує ітеративний процес, який повторюється 16 раундів. Кожен раунд включає змішування даних з ключем шифрування та певні нелінійні операції.

3) Функція Фейстеля: DES використовує структуру Фейстеля, де блок даних розбивається на дві половини, і одна половина функціонує як вхід, а друга - як вихід. Функція Фейстеля включає розширення, перестановки, підстановки та XOR-операції з раундовими підключами.

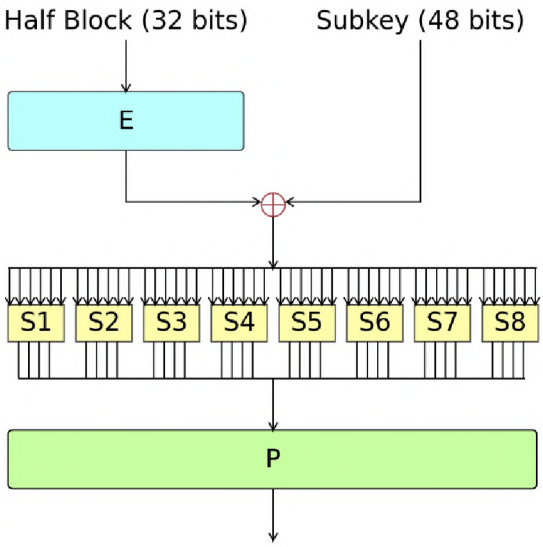

Рисунок 1.3 – Графічне представлення роботи алгоритма DES

DES використовує таблиці підстановок (S-box), де 6-бітні блоки замінюються на 4-бітні блоки залежно від вмісту таблиці. Також виконуються перестановки бітів згідно з фіксованими таблицями.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	S_1
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	S_2
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	S_3
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	S_4
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	S_5
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	

Рисунок 1.4 – Приклад таблиці S-box.

Процес шифрування DES включає наступні кроки:

1. Початкова перестановка (Initial Permutation): Початковий блок даних проходить початкову перестановку для отримання нового блоку.

2. Розбиття на ліву та праву половини: Початковий блок даних розбивається на ліву та праву половини.

3. Раунди шифрування: Ліва половина блоку стає вхідним для функції Фейстеля. Кожен раунд включає змішування правої половини з функцією Фейстеля, а потім обмін правої та лівої половинами. Раундовий підключ використовується для кожного раунду.

4. Фінальний раунд: Останній раунд виконується без обміну половинами після функції Фейстеля.

5. Фінальна перестановка (Final Permutation): Останній блок даних проходить фінальну перестановку, щоб отримати зашифрований блок.

6. Процес дешифрування DES включає обернений порядок раундів та використання обернених раундових підключів.

Розшифрування:

Процес розшифрування DES (Data Encryption Standard) включає наступні кроки:

1. Початкова обробка ключа:

- Вибір початкового ключа, який складається з 64 біт.

2. Розгортання ключів:

- Застосування алгоритму розгортання ключів DES для отримання 16 підключів, які використовуються в кожному раунді розшифрування.

3. Розшифрування:

- Вхідні дані, які потрібно розшифрувати, поділяються на блоки даних розміром 64 біти.

- Кожен блок даних проходить через 16 раундів розшифрування, включаючи такі операції.

- Початковий перестановка (Initial Permutation): Застосування перестановки бітів до блоку даних.

- Раунди Фейстеля (Feistel Rounds): Виконання ітераційних раундів, які включають операції зсуву, заміни, перестановки та операцію XOR з використанням підключів.

- Зворотна перестановка (Final Permutation): Застосування оберненої перестановки бітів до результуючого блоку даних після останнього раунду.

4. Об'єднання результуючих блоків даних:

- Об'єднання результуючих блоків даних для отримання оригінального розшифрованого повідомлення.

Слід зауважити, що DES був розроблений ще в 1970-х роках і вважається застарілим з точки зору сучасної криптографії, оскільки його ключова довжина надто коротка для поточних стандартів безпеки. Рекомендується використовувати більш сучасні алгоритми шифрування, такі як AES (Advanced Encryption Standard).

1.2.3 Дослідження асиметричного алгоритму шифрування RSA

Шифр RSA (Rivest-Shamir-Adleman) є одним з найвідоміших і популярних криптографічних алгоритмів. RSA був винайдений в 1977 році Роном Рівестом, Аді Шаміром та Леонардом Адлеманом, як відкритий криптографічний алгоритм. Вони опублікували свою роботу у науковій статті під назвою "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems" (Метод отримання цифрових підписів та криптосистем з відкритими ключами). Основна ідея RSA полягає у використанні математичних властивостей складних обчислень над великими простими числами. RSA використовує два ключі - публічний ключ для шифрування і приватний ключ для розшифрування. Прикметно, що ідея використання публічного та приватного ключів в криптографії була незалежною відкрита ще до Рівеста, Шаміра та Адлемана. Подібні ідеї були розглянуті в 1973 році Бріаном Вітхамом та в 1974 році Кліффордом Коксом та Мартіном Геллманом. RSA відомий своєю криптографічною стійкістю, основаною на складності факторизації великих цілих чисел. Використання великих чисел забезпечує безпеку шифрування та цифрових підписів за умови вибору відповідних ключів. RSA широко використовується у сучасних криптографічних системах, включаючи захист

комунікації по мережі, цифрові підписи, електронну комерцію та багато інших додатків. RSA став стандартом, прийнятим у багатьох країнах і організаціях. Він також входить до складу різних криптографічних протоколів, таких як TLS (Transport Layer Security) та SSH (Secure Shell). Шифр RSA зберігає свою вагомість і популярність в світі криптографії і продовжує бути використовуваним для забезпечення безпеки та конфіденційності важливої інформації.

Основні кроки шифрування RSA:

RSA (Rivest-Shamir-Adleman) - це асиметричний криптографічний алгоритм, який використовується для шифрування та підпису даних. RSA базується на математичній задачі факторизації чисел і використовує два ключі: публічний ключ для шифрування та приватний ключ для розшифрування.

Рисунок 1.5 – Схема роботи алгоритма RSA

1. Генерація ключів:

- Вибір двох простих чисел p і q.

- Обчислення модуля n, де

$$n = p * q. \quad (1.1)$$

- Обчислення значення функції Ейлера (phi) для n, де

$$phi(n) = (p-1) * (q-1). \quad (1.2)$$

- Вибір публічного експонента e, такого що $1 < e < phi(n)$ і є взаємно простим з phi(n).

- Обчислення приватного експонента d, такого що

$$(d * e) \bmod phi(n) = 1. \quad (1.3)$$

2. Шифрування:

- Припустимо, що ми маємо повідомлення M, яке ми хочемо зашифрувати.

- Використовуючи публічний ключ (n, e), обчислюємо шифротекст C за допомогою формули:

$$C = M^e \bmod n. \quad (1.4)$$

Розшифрування:

1. Припустимо, що ми отримали шифротекст C, який ми хочемо розшифрувати.

2. Використовуючи приватний ключ (n, d), обчислюємо повідомлення M за допомогою формули:

$$M = C^d \bmod n. \quad (1.5)$$

Отже, шифрування RSA включає публічний ключ (n, e), де n - модуль, а e - публічний експонент, для шифрування повідомлення, тоді як розшифрування використовує приватний ключ (n, d), де n - модуль, а d - приватний експонент, для отримання початкового повідомлення.

Приклад шифрування:

1. Генерація ключів:

- Виберемо два простих числа: p = 17 та q = 11.

- Обчислимо модуль n: n = p * q = 17 * 11 = 187.

- Обчислимо значення функції Ейлера phi(n): phi(n) = (p-1) * (q-1) = 16 * 10 = 160.

- Виберемо публічний експонент e, який є взаємно простим з phi(n) і меншим за нього. Припустимо, що e = 7.

- Обчислимо приватний експонент d, такий що (d * e) mod phi(n) = 1. В даному прикладі, d = 23.

2. Шифрування:

- Припустимо, що ми маємо повідомлення M = 88, яке ми хочемо зашифрувати.

- Використовуючи публічний ключ (n, e), обчислимо шифротекст C за допомогою формули: C = M^e mod n = 88^7 mod 187 = 11.

Розшифрування:

- Припустимо, що ми отримали шифротекст C = 11, який ми хочемо розшифрувати.

- Використовуючи приватний ключ (n, d), обчислимо повідомлення M за допомогою формули: M = C^d mod n = 11^23 mod 187 = 88.

Отже, зашифроване повідомлення 88 засобами RSA буде 11, а розшифроване повідомлення 11 знову стає 88.

1.2.4 Дослідження асиметричного алгоритму шифрування ECC

Еліптична криптографія (ECC) - це криптографічний метод, який використовує математичні властивості еліптичних кривих для створення криптографічних ключів і зашифрування даних. Історія алгоритму шифрування ECC починається з досліджень проведених у 1980-х роках. У 1985 році Ніл Кобліц і Віктор Міллер незалежно один від одного відкрили, що еліптичні криві можуть бути використані для побудови криптографічних систем. Вони показали, що використання арифметики на еліптичних кривих забезпечує високу стійкість до криптоаналізу при невеликій довжині ключа порівняно з іншими криптографічними алгоритмами. В середині 1990-х років стандартизація еліптичної криптографії отримала значний поштовх. У 1999 році Національний інститут стандартів і технологій США оголосив конкурс на вибір нового криптографічного стандарту, який включав алгоритми на основі еліптичних кривих. У результаті цього конкурсу у 2000 році був обраний стандарт ECDSA (Elliptic Curve Digital Signature Algorithm), який використовується для цифрового підписування. З часом ECC отримала значну популярність і застосування у різних сферах. Вона була використана в багатьох протоколах безпеки, таких як TLS/SSL для захисту передачі даних по мережі Інтернет, а також в смарт-картах, мобільних пристроях та інших вбудованих системах. З часом з'явилися нові методи криптоаналізу ECC, і розробники алгоритму стали працювати над покращенням стійкості і швидкості алгоритму. Були розроблені нові криві з покращеними властивостями і методи розрахунку ключів. Згодом були створені стандарти для ECC, такі як ANSI X9.62 і SEC (Standards for Efficient Cryptography). Ці стандарти встановлюють параметри кривих, методи шифрування та формати ключів для ефективного використання ECC в різних системах. За останні роки ECC стала широко використовуваним алгоритмом шифрування, особливо в сфері мобільних комунікацій та Інтернету

речей, де обмежені ресурси становлять проблему. Вона надає високий рівень стійкості при використанні коротших ключів порівняно з іншими алгоритмами.

Основні кроки шифрування ECC:

Алгоритм шифрування ECC (еліптична криптографія з використанням еліптичних кривих) базується на математичних властивостях еліптичних кривих над полем скінченних чисел. Дозволяє забезпечити безпеку шифрування та підпису даних. Давайте розглянемо основні кроки алгоритму ECC шифрування.

1. Генерація ключів:

Вибір еліптичної кривої: Спочатку вибирається певна еліптична крива над полем скінченних чисел. Ця крива визначається рівнянням вигляду

$$y^2 = x^3 + ax + b \tag{1.6}$$

де a і b - параметри кривої.

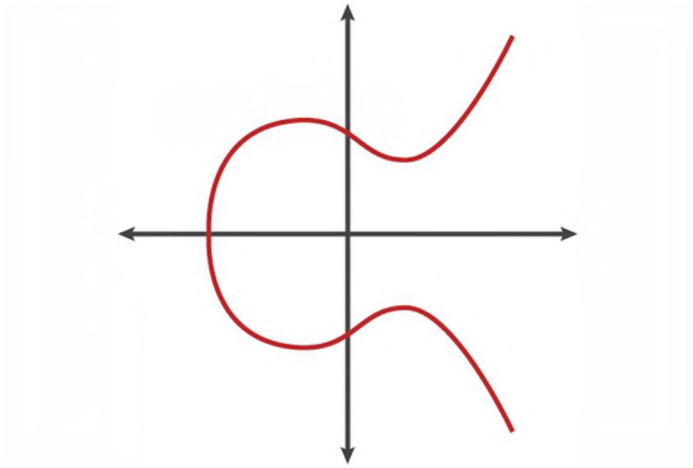

Рисунок 1.6 – Графік функції y^2 = x^3 + ax + b

Вибір точки-генератора: Обирається точка P на еліптичній кривій як точка-генератор.

Вибір приватного ключа: Генерується випадкове число d, яке служить приватним ключем.

Обчислення публічного ключа: Публічний ключ обчислюється як

$$Q = dP \tag{1.7}$$

де Q - публічний ключ, P - точка-генератор, а d - приватний ключ.

2. Шифрування:

- Вибір випадкового числа k: Випадково генерується число k.

- Обчислення точки шифрування: Точка-шифротекст обчислюється як

$$C = kP \tag{1.8}$$

де C - шифротекст, P - точка-генератор, а k - випадкове число.

- Перетворення повідомлення: Повідомлення, яке потрібно зашифрувати, представляється у вигляді числа або байтового рядка.

- Обчислення точки-публічного ключа одноразового використання: Обчислюється точка

$$Q = kP. \tag{1.9}$$

- Обчислення шифрованого тексту: Шифрований текст обчислюється як точка-публічний ключ одноразового використання, додана до результату XOR між повідомленням і координатою x точки-шифротексту.

3. Розшифрування:

- Обчислення точки-приватного ключа одноразового використання: Обчислюється точка

$$S = dQ \tag{1.10}$$

де Q - публічний ключ одноразового використання, а d - приватний ключ.

- Обчислення координати x точки-шифротексту: Координата x точки-шифротексту витягується з шифрованого тексту.

- Відновлення повідомлення: Відновлення повідомлення здійснюється шляхом виконання операції XOR між шифрованим текстом і координатою x точки-шифротексту.

Це загальний опис алгоритму ECC шифрування. У ньому використовуються операції додавання точок на еліптичній кривій і множення точки на число. Для виконання цих операцій використовуються спеціальні формули, такі як формули для додавання двох точок і подвоєння точки на кривій.

Зауважте, що це загальний опис алгоритму, і існують різні варіації ECC, такі як ECDSA (еліптична цифровий підпис) і ECIES (еліптичний криптографічний механізм шифрування). Конкретні деталі і формули можуть варіюватися в залежності від використаного варіанту.

Приклад шифрування:

Припустимо, що ми працюємо з еліптичною кривою над простим полем GF(p) і використовуємо алгоритм ECC з ключами довжиною 256 біт.

1. Генерація ключів:

- Вибираємо параметри еліптичної кривої:

- Поле GF(p): p = 23.

- Коефіцієнти еліптичної кривої: наприклад, a = 1, b = 6.

- Початкова точка (генератор): наприклад, G(3, 10).

- Вибираємо приватний ключ: наприклад, d = 9.

- Обчислюємо публічний ключ: P = d * G

У нашому прикладі: P = 9 * G = (12, 7).

2. Шифрування:

Вибираємо випадкове число (ключ сеансу): наприклад, k = 4.

Обчислюємо шифрований текст:

Обчислюємо точку на кривій, що відповідає ключу сеансу: C = k * G.

У нашому прикладі: C = 4 * G = (17, 20).

Шифрований текст - координата x точки C: шифрований_текст = 17.

Отриманий шифрований текст можна передавати як відкритий текст, так як без знання приватного ключа дуже важко відновити ключ сеансу або вихідне повідомлення.

1.2.5 Дослідження алгоритмів цифрового підпису DSA

Алгоритм цифрового підпису DSA (Digital Signature Algorithm) був розроблений в 1991 році Дейвідом Кріппом та Джеффрі Шнорром. Він став одним із стандартів для цифрового підпису й отримав значне визнання у світі криптографії. Процес розробки DSA був ініційований Національним інститутом стандартів і технологій США з метою створення стандарту для цифрового підпису, який був би безпечним та ефективним. Основні вимоги до алгоритму полягали в його стійкості до атак на безпеку, швидкості обчислень та простоті використання. DSA базується на математичних проблемах, зокрема, обчисленні дискретного логарифма в скінченних полях. Ідея алгоритму полягає в тому, щоб створити підпис, використовуючи приватний ключ, який знає тільки власник, і перевірити цей підпис за допомогою публічного ключа. DSA отримав широке застосування в різних галузях, включаючи фінансові транзакції, електронну пошту, цифрові документи та інші області, де важлива цілісність та аутентичність інформації. Алгоритм також став стандартом урядових організацій, зокрема, у США. Однак, з часом з'явилися сумніви щодо безпеки

DSA через обмежену довжину ключів, що використовувалися. У 2000 році NIST рекомендував переходити до більш сильних алгоритмів, зокрема, до RSA та ECDSA. Незважаючи на це, DSA залишається важливим алгоритмом і використовується в різних системах та протоколах. Історія його розробки і впровадження позначила собою важливий етап у розвитку цифрових підписів та криптографії загалом.

Основні кроки шифрування DSA:

Алгоритм цифрового підпису DSA (Digital Signature Algorithm) є криптографічним протоколом, який використовується для створення та перевірки цифрових підписів. DSA був розроблений з метою забезпечення цілісності, аутентифікації та невідмовності в інформаційних системах.

1. Генерація ключів:

- Вибір великого простого числа q (розмірність, наприклад, 160 біт) та числа p (просте число розмірністю, наприклад, 1024-1600 біт), такого що q є дільником p-1.

- Вибір генератора g, який є елементом підгрупи порядку q в групі залишків по модулю p.

- Генерація приватного ключа x (випадковогочисла в діапазоні [1, q-1]).

- Обчислення публічного ключа y, де

$$y = g^x \bmod p \qquad (1.11)$$

2. Підписання повідомлення:

Обчислення хеш-функції повідомлення, наприклад, SHA-1, для отримання дайджесту.

Генерація випадкового числа k (в діапазоні [1, q-1]).

Обчислення r, де

$$r = (g^k \bmod p) \bmod q \quad (1.12)$$

Обчислення s, де

$$s = (k^{(-1)} * (\text{хеш-дайджест} + x * r)) \bmod q \quad (1.13)$$

Цифровий підпис - (r, s).

3. Перевірка підпису:

Отримання публічного ключа y, r та s.

Обчислення хеш-функції повідомлення для отримання дайджесту.

Обчислення w, де

$$w = s^{(-1)} \bmod q \quad (1.14)$$

Обчислення u1, де

$$u1 = (\text{хеш-дайджест} * w) \bmod q \quad (1.15)$$

Обчислення u2, де

$$u2 = (r * w) \bmod q \quad (1.16)$$

Обчислення v, де

$$v = ((g^{u1} * y^{u2}) \bmod p) \bmod q \quad (1.17)$$

Підпис вірний, якщо v = r.

Алгоритм DSA забезпечує невідмовність, оскільки приватний ключ x використовується тільки для підпису повідомлень, тоді як публічний ключ y використовується для перевірки підпису. Це дозволяє іншим особам перевірити автентичність повідомлення, не розкриваючи приватний ключ.

1.2.6 Дослідження алгоритмів цифрового підпису ECDSA.

Алгоритм цифрового підпису ECDSA (Elliptic Curve Digital Signature Algorithm) був вперше описаний у 1992 році Нілом Кобліцем (Neal Koblitz) та Віктором Міллером (Victor Miller). Вони незалежно один від одного розробили алгоритм цифрового підпису, базований на еліптичних кривих. Ідея використання еліптичних кривих для криптографічних цілей виникла вже в 1985 році, коли Ніл Кобліц висловив свої міркування про потенційну стійкість і ефективність такого підходу. Проте, перші конкретні алгоритми цифрового підпису на основі еліптичних кривих з'явилися лише в 1992 році. У 1999 році алгоритм ECDSA був внесений до стандарту американського Національного інституту стандартів і технологій (NIST) для використання в цифрових підписах. Відтоді ECDSA став одним з найпоширеніших алгоритмів цифрового підпису, особливо в контексті криптовалют, таких як Bitcoin та Ethereum, де використовуються еліптичні криві для забезпечення безпеки та автентифікації транзакцій. ECDSA відзначається своєю високою стійкістю до атак на безпеку, при цьому вимагаючи менше обчислювальних ресурсів порівняно з іншими алгоритмами цифрового підпису, такими як RSA. Це робить його привабливим вибором для широкого спектра застосувань, де важлива безпека і швидкодія, зокрема в мобільних пристроях та розподілених системах. За останні роки проводилися численні дослідження з метою вдосконалення ECDSA, включаючи вивчення нових кривих, аналіз безпеки та вдосконалення ефективності обчислень. Це сприяло покращенню безпеки та розширенню застосування ECDSA в сучасній криптографії.

Основні кроки шифрування ECDSA:

1. Вибір параметрів:

- Виберемо публічний параметр - еліптичну криву, над якою будемо працювати.

- Виберемо точку на цій кривій, відому як базова точка (base point), і публічний параметр n, який визначає порядок базової точки.

2. Генерація ключів:

- Користувач генерує випадковий приватний ключ (private key), який представляє собою випадкове число в певному діапазоні.

- За допомогою приватного ключа обчислюється відповідний публічний ключ (public key) за формулою: public key = private key * базова точка.

3. Підписування повідомлення:

- Користувач обчислює хеш (hash) повідомлення, яке потрібно підписати.

- Користувач генерує випадкове число k (в межах 1 до n-1), яке використовується для обчислення підпису.

- Користувач обчислює r і s за формулами:

r = (x координата точки R) mod n, де R = k * базова точка. (1.18)

s = (k^(-1) * (хеш повідомлення + private key * r)) mod n, де k^(-1) - обернене число до k за модулем n. (1.19)

4. Верифікація підпису:

- Отримувач повідомлення отримує публічний ключ від користувача, який підписав повідомлення, і отримує підписане повідомлення.

- Отримувач обчислює хеш повідомлення.

- Отримувач перевіряє підпис, обчислюючи точку R за формулою:

R = (s^(-1) * хеш повідомлення * базова точка + s^(-1) * r * публічний ключ). (1.20)

- Якщо координата x точки R співпадає з r, то підпис вважається коректним.

1.2.7 Дослідження алгоритмів аутентифікації HMAC

HMAC є одним з найбільш поширених алгоритмів для забезпечення цілісності та автентифікації повідомлень. HMAC був вперше описаний в 1996 році Дугласом Штернсом (Douglas Stinson) та Мао Ю Чжу (Mao Yu Chu) як покращення існуючого алгоритму аутентифікації на основі хеш-функцій. Він базується на застосуванні хеш-функції разом з ключем для створення коду аутентичності повідомлення. Основна ідея HMAC полягає в тому, щоб комбінувати ключ і повідомлення за допомогою хеш-функції, що дозволяє створити унікальний код аутентичності, який може бути перевірений при отриманні повідомлення. Це забезпечує інтегрітет повідомлення та підтверджує автентичність відправника. Вивчення стійкості алгоритму HMAC до різних атак, таких як атаки на колізії, передавання повідомлень та обчислювальні атаки. Дослідження спрямовані на виявлення можливих слабкість і вдосконалення безпеки алгоритму. Дослідження спрямовані на перевірку відповідності та рекомендації щодо використання в різних контекстах. Аналіз ефективності алгоритму HMAC, включаючи швидкодію обчислень та оптимізації для різних платформ. Дослідження спрямовані на вдосконалення швидкодії та зменшення обчислювальних витрат. Вивчення можливостей та застосувань алгоритму HMAC в різних сферах, включаючи мережеву безпеку, криптографічні протоколи, аутентифікацію користувачів та інші області.

Основні кроки шифрування HMAC:

Алгоритм цифрового підпису HMAC (Hash-based Message Authentication Code) працює на основі хеш-функцій і забезпечує цифровий підпис повідомлення:

1. Вибір хеш-функції:

- Обирається відповідна хеш-функція, така як SHA-1, SHA-256 або SHA-512.

2. Генерація ключа:

- Користувач генерує випадковий секретний ключ (secret key), який використовується для обчислення підпису.

3. Підписування повідомлення:

- Користувач обчислює підпис за формулою:

$$HMAC = H((key \text{ XOR } opad) \| H((key \text{ XOR } ipad) \| message)) \quad (1.21)$$

де H - хеш-функція, key - секретний ключ, opad - константа для зовнішнього падінгу (зазвичай 0x5C), ipad - константа для внутрішнього падінгу (зазвичай 0x36), message - повідомлення, яке потрібно підписати.

4. Верифікація підпису:

- Отримувач повідомлення отримує підписане повідомлення і секретний ключ.

- Отримувач обчислює очікуваний підпис за тим же алгоритмом HMAC.

- Отримувач порівнює отриманий підпис з очікуваним підписом. Якщо вони співпадають, підпис вважається коректним.

Це основні кроки алгоритму HMAC. Його безпека забезпечується за рахунок операцій XOR та хеш-функцій, які забезпечують інтегритет та автентифікацію повідомлення.

1.2.8 Дослідження алгоритмів аутентифікації SSL/TLS

SSL (Secure Sockets Layer) і TLS (Transport Layer Security) - це протоколи шару транспортного рівня, які забезпечують безпеку комунікації через інтернет. Вони використовуються для захисту передачі даних між клієнтом і сервером, забезпечуючи конфіденційність, цілісність та автентичність. SSL був створений компанією Netscape Communications у 1990-х роках як протокол для безпечної передачі даних по мережі. Згодом, Інтернетова інженерна робоча група (IETF) переробила SSL і створила TLS. Термін SSL все ще використовується для описування раніших версій протоколу, але в загальному розумінні SSL і TLS часто використовуються взаємозамінно.

Алгоритм аутентифікації SSL/TLS (Secure Sockets Layer/Transport Layer Security) є важливою складовою протоколів шару транспорту, які забезпечують безпеку комунікації в мережі Інтернет. Історія розвитку SSL/TLS складається з кількох ключових етапів:

SSL: Протокол SSL був розроблений компанією Netscape Communications Corporation у 1994 році з метою захисту передачі конфіденційної інформації через мережу. Перша версія SSL 1.0 була ніколи не була випущена публічно, і SSL 2.0 став першою широко використовуваною версією. Протокол SSL використовував шифрування з симетричними ключами для захисту даних.

TLS: У 1999 році Інженерна група Інтернету прийняла протокол SSL 3.0 в якості основи для розвитку TLS. TLS 1.0 був офіційно представлений у 1999 році і включав кілька виправлень і покращень порівняно з SSL 3.0. TLS використовується для забезпечення безпеки веб-переглядачів, електронної пошти, іншого клієнт-серверного зв'язку та багатьох інших додатків.

Послідовні версії TLS: Протокол TLS продовжував розвиватись з випуском нових версій для поліпшення безпеки та розширення функціональності. Найважливіші версії TLS включають TLS 1.1 (2006), TLS 1.2 (2008) та TLS 1.3

(2018). TLS 1.3 є найновішою версією протоколу та пропонує значні поліпшення в ефективності та безпеці.

Усі версії SSL/TLS використовують алгоритми аутентифікації, такі як RSA, DSA, ECDSA, HMAC та інші, для забезпечення цілісності, конфіденційності та автентичності комунікації. Протокол SSL/TLS застосовується в багатьох сферах, включаючи безпеку веб-сайтів, електронну комерцію, банківські та фінансові операції та інші області, де необхідна захищена передача даних.

Основні відомості про SSL/TLS:

1. Конфіденційність даних: Протоколи SSL/TLS використовують криптографічні алгоритми для шифрування переданих даних, що робить їх незрозумілими для сторонніх спостерігачів.

2. Цілісність даних: SSL/TLS включає механізми контролю цілісності, що дозволяють виявляти будь-які зміни в переданих даних під час передачі.

3. Автентичність: Протоколи SSL/TLS дозволяють перевіряти автентичність сервера, що допомагає уникнути атак заміни сервера і забезпечити співпрацю з вірним сервером.

4. Взаємодія з різними протоколами: SSL/TLS може бути використаний у поєднанні з різними протоколами, такими як HTTP, FTP та інші, для забезпечення безпеки комунікації в різних сферах.

Протоколи SSL/TLS використовують асиметричну криптографію (з публічними та приватними ключами) для обміну ключами і встановлення безпечного каналу зв'язку, а також симетричну криптографію для шифрування даних під час передачі. Вони широко використовуються для захисту онлайн-транзакцій, електронної комерції, веб-сайтів і будь-яких інших ситуацій, де забезпечення безпеки передачі даних є критичним.

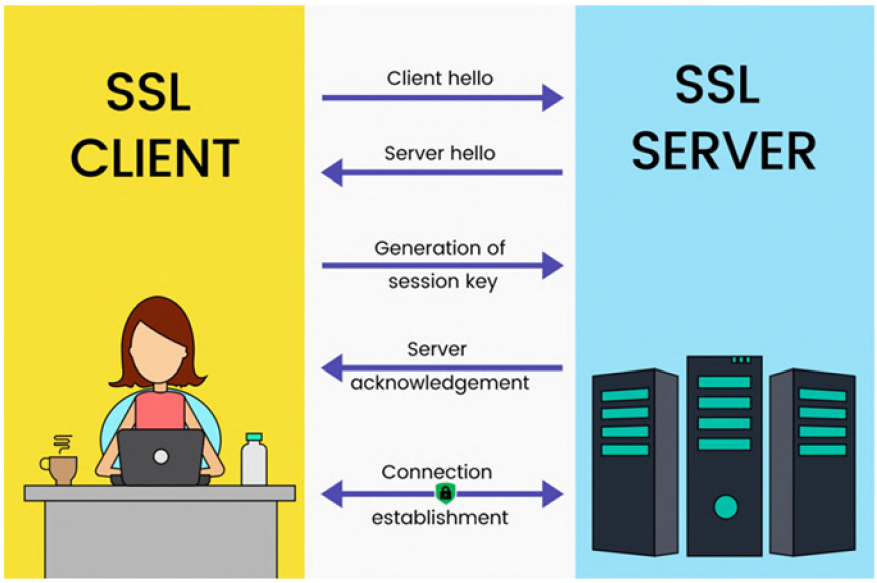

Рисунок 1.7 - Приклад роботи алгоритма

Алгоритми, які використовує SSL/TLS:

SSL/TLS (Secure Sockets Layer/Transport Layer Security) використовує різні алгоритми аутентифікації для забезпечення безпеки комунікації. Основними алгоритмами аутентифікації, які використовуються в SSL/TLS, є наступні:

RSA (Rivest-Shamir-Adleman): RSA є асиметричним алгоритмом аутентифікації, який використовує пару ключів - приватний ключ і публічний ключ. Публічний ключ використовується для шифрування даних та перевірки цифрових підписів, тоді як приватний ключ використовується для розшифрування даних та створення цифрових підписів.

DSA (Digital Signature Algorithm): DSA також є асиметричним алгоритмом аутентифікації, який використовується для створення цифрових підписів. Він базується на складності обчислення дискретного логарифму в групі цілих чисел modulo простого числа.

ECDSA (Elliptic Curve Digital Signature Algorithm): ECDSA є алгоритмом аутентифікації, який використовує еліптичні криві для створення цифрових

підписів. Він надає ту саму рівень безпеки, що й RSA або DSA, але з меншими обчислювальними витратами.

HMAC (Hash-based Message Authentication Code): Хеш-заснований код аутентифікації повідомлення (HMAC) використовується для перевірки цілісності та автентичності даних, переданих по SSL/TLS. Він використовує хеш-функцію (наприклад, SHA-256) та секретний ключ для створення коду аутентичності повідомлення.

Ці алгоритми аутентифікації використовуються разом з іншими криптографічними методами, такими як симетричне шифрування та хеш-функції, для забезпечення конфіденційності, цілісності та автентичності комунікації за допомогою SSL/TLS.

1.2 Висновки і постановка задачі

У цьому розділі було розглянуто різні алгоритми шифрування, такі як AES, DES, RSA, ECC, а також алгоритми цифрового підпису, такі як DSA і ECDSA. Кожен з цих алгоритмів має свої унікальні особливості і застосування у сфері криптографії.

AES і DES є симетричними блочними шифрами, де ключ використовується для шифрування і розшифрування повідомлення. RSA є асиметричним алгоритмом, де використовується пара ключів - приватний і публічний, для шифрування і розшифрування повідомлення. ECC також є асиметричним алгоритмом, але використовує еліптичні криві для криптографічних операцій.

Алгоритми цифрового підпису, такі як DSA і ECDSA, використовуються для забезпечення автентифікації та цілісності повідомлень. Вони використовують пару ключів - приватний і публічний, для створення та перевірки цифрових підписів.

Крім алгоритмів шифрування та цифрового підпису, ми також розглянули SSL/TLS (Secure Sockets Layer/Transport Layer Security). SSL/TLS - це протоколи, які забезпечують захищену комунікацію через мережу Інтернет.

SSL/TLS забезпечує шифрування даних та аутентифікацію сторінок, що спілкуються, що дозволяє забезпечити конфіденційність та безпеку під час передачі інформації між клієнтом і сервером. Цей протокол використовує різні алгоритми шифрування та цифрового підпису, такі як RSA, ECC, ECDSA, DSA, HMAC для забезпечення безпеки даних.

Кожен з цих алгоритмів має свої переваги і недоліки, і вибір певного алгоритму залежить від конкретних потреб і вимог безпеки. Наявність різноманітних алгоритмів дозволяє використовувати ефективні методи шифрування та підписування повідомлень для захисту конфіденційності, цілісності та автентичності наших даних у цифровому середовищі.

У наступному розділі буде проведений аналіз та порівняння даних алгоритмів шифрування, алгоритмів підпису а алгоритмів аутентифікації для визначення найкращого алгоритму для захисту даних в інформаційних системах, в яких зберігається, обробляється та преедається конфіденційна інформація компанії.

РОЗДІЛ 2. СПЕЦІАЛЬНА ЧАСТИНА

2.1 Порівняння ефективності та безпеки різних алгоритмів криптографічного захисту інформації.

Порівняння ефективності та безпеки різних алгоритмів криптографічного захисту інформації є важливою задачею для забезпечення безпеки в цифровому світі. Різні алгоритми мають свої переваги та обмеження, і їх вибір залежить від контексту використання та потреб користувача.

Один з головних критеріїв порівняння - це криптографічна стійкість безпека. Криптографічні алгоритми повинні бути стійкими до атак і математичних аналізів. Безпека алгоритму оцінюється за його опірність до таких атак, як brute-force атаки, атаки з використанням відкритого тексту, атаки на ключі та інші.

Також важливим критерієм є ефективність алгоритму. Це означає, що алгоритм повинен бути швидким і витрачати мінімальну кількість ресурсів, таких як обчислювальна потужність та пам'ять. Швидкість шифрування та розшифрування, розмір ключа, розмір блоку даних - це параметри, які враховуються при оцінці ефективності.

Додаткові фактори, які враховуються при порівнянні, включають розмір ключа, доступність реалізації алгоритму на різних платформах, витривалість до атак.

При порівнянні алгоритмів криптографічного захисту інформації не існує універсального "найкращого" алгоритму, оскільки вибір залежить від конкретних вимог і контексту використання.

Шкала оцінок критеріїв алгоритму:

1 – погано;

2 – добре;

3 – відмінно;

2.1.1 Порівняння алгоритму AES та DES

Порівняння алгоритмів AES (Advanced Encryption Standard) і DES (Data Encryption Standard) є важливим аспектом оцінки їх ефективності та безпеки в контексті криптографічного захисту інформації. Наведемо деякі ключові аспекти порівняння цих двох алгоритмів:

Основні відомості про шифр AES та DES:

Принцип роботи: Принцип дії алгоритму AES (Advanced Encryption Standard) полягає в шифруванні та розшифруванні блоків даних розміром 128 біт за допомогою ключа, який може мати довжину 128, 192 або 256 біт. Алгоритм складається з кількох раундів, кількість яких залежить від довжини ключа. Кожен раунд включає чотири операції: заміну байтів, зсув рядків, перемішування колонок і комбінування з раундовим ключем. Ці операції забезпечують зміну і перемішування даних для забезпечення безпеки і дифузії. Алгоритм AES відомий своєю ефективністю, безпекою та широким застосуванням у різних областях, включаючи захист даних, збереження і передачу інформації.

Принцип дії алгоритму DES (Data Encryption Standard) полягає в шифруванні та розшифруванні блоків даних розміром 64 біти з використанням 56-бітного ключа. Алгоритм DES використовує заміну, перестановку та комбінування даних у процесі шифрування. Він складається з 16 раундів, кожен з яких включає перестановки та заміни бітів, комбінування даних з підключами та виконання побітової операції XOR. Хоча DES був популярним шифром у минулому, він вважається застарілим з точки зору безпеки і був замінений більш сильними алгоритмами, такими як AES.

1. Стійкість:

- AES і DES використовують різні ключові розміри та криптографічні механізми:

- AES використовує ключі довжиною 128, 192 або 256 біт, що робить його більш стійким до атак з використанням brute-force.

- DES використовує ключ довжиною 56 біт, що робить його менш стійким до сучасних обчислювальних атак. DES має кілька відомих атак, таких як атака з використанням зламування з використанням криптоаналізу лінійного та диференціального, атака з використанням вибраних шифрувальних текстів та атака з використанням залежностей ключів.

2. Довжина ключа:

- AES використовує ключі фіксованого розміру, які залежать від варіанту (128, 192 або 256 біт).

- DES використовує ключ довжиною 56 біт, що вважається недостатньою для стійкого шифрування.

3. Швидкість:

- У порівнянні з DES, AES забезпечує кращу ефективність. AES використовує більш потужні шифрувальні блоки та оптимізовані алгоритми, що дозволяє забезпечити швидке шифрування та розшифрування даних.

- DES має меншу швидкість обробки даних через свою меншу ключову довжину та застаріліший алгоритм.

4. Використання:

- З огляду на свою старість, DES менше підтримується у сучасних системах та протоколах. DES розроблений ще у 1970-х роках і незадовольняє сучасним вимогам безпеки.

- AES, натомість, є широко використовуваним алгоритмом у багатьох сферах, включаючи застосування в протоколах безпеки, мережевих комунікаціях та зберіганні даних. AES використовує сучасні криптографічні методи, такі як замішання байтів, зсуви рядків, лінійне перетворення та ключовий розклад. Ці методи забезпечують високу стійкість алгоритму до

різних атак, включаючи криптоаналіз лінійного та диференціального. AES є ефективним алгоритмом шифрування, який може бути реалізований на різних пристроях та платформах. AES широко використовується для захисту конфіденційної інформації, включаючи дані, що передаються по мережі, зберігаються на пристроях або зберігаються у файловій системі. AES є одним з найбільш використовуваних шифрів у сучасних криптографічних системах і забезпечує надійний рівень безпеки для шифрування і розшифрування даних.

Таблиця 2.1 – Порівняльна таблиця для алгоритмів AES та DES

Алгоритми	Критерії оцінки				
	Стійкість	Розмір ключа	Швидкість	Використання	Сума
AES	3	2	2	3	10
DES	1	1	1	1	4

Висновок: Загалом, AES є більш безпечним, ефективним та сучасним алгоритмом порівняно з DES. У більшості випадків рекомендується використовувати AES для шифрування даних замість DES, з урахуванням безпеки, ефективності та підтримки алгоритму у конкретній системі чи протоколі.

Ключова довжина AES, застосування сучасних криптографічних методів та відсутність відомих практичних атак роблять його більш підходящим для сучасних потреб у криптографії. DES, з короткою ключовою довжиною та вразливостями до відомих атак, вважається застарілим і не рекомендується для нових застосувань. Отже, якщо мова йде про вибір між AES і DES для сучасної системи обробки і передачі даних, то AES є більш підходящим вибором з огляду на його вищий рівень безпеки і сучасні стандарти.

2.1.2 Порівняння алгоритму RSA та AES

Порівняння алгоритмів RSA (Rivest-Shamir-Adleman) і AES (Advanced Encryption Standard) включає оцінку їх використання в криптографічних застосуваннях, швидкості роботи та стійкості до атак. Розглянемо деякі аспекти порівняння обох алгоритмів.

Основні відомості про шифр RSA:

RSA використовує два ключі: публічний ключ для шифрування і підпису даних та приватний ключ для розшифрування і перевірки підпису. Алгоритм базується на математичній складності факторизації великих чисел, що становить основу безпеки RSA. Головний принцип RSA полягає у використанні публічного ключа для шифрування повідомлення, а приватного ключа - для розшифрування. Застосовується модульна арифметика для обчислень, зокрема, для піднесення до степеня і виконання операцій з числами за модулем. Генерація ключів RSA включає в себе вибір двох великих простих чисел, обчислення модуля та експоненти, а також обчислення приватного ключа на основі публічного. Застосовується для шифрування конфіденційних даних, передачі ключів, підпису документів та інших криптографічних операцій. RSA є одним із найпоширеніших алгоритмів шифрування і використовується в багатьох протоколах і системах, включаючи TLS/SSL для безпечної передачі даних через Інтернет. Відносно повільний у порівнянні з деякими іншими алгоритмами, особливо при обробці великих обсягів даних. Шифр RSA вимагає довгих ключів для забезпечення високого рівня безпеки, що може вплинути на продуктивність та обчислювальну складність.

1. Стійкість:

- AES вважається стійким до криптоаналітичних атак, таких як differential cryptanalysis, linear cryptanalysis та багато інших. Для сучасних комп'ютерів і

атак з використанням brute-force, AES з великою ключовою довжиною є практично некомпромісним.

- RSA заснований на складності факторизації великих простих чисел. На сьогоднішній день немає ефективних алгоритмів для швидкого факторизації великих чисел, тому RSA вважається стійким до факторизаційних атак. RSA також стійкий до багатьох криптоаналітичних атак, таких як chosen plaintext атаки, chosen ciphertext атаки, timing атаки тощо. Однак, його безпека може бути підірвана, якщо використовуються некоректні реалізації або вразливості в самому протоколі RSA.

2. Ключова довжина:

- RSA підтримує ключі від 1024 до 4096 біт. Забезпечує великий простір ключів, що робить алгоритм стійким до атак з використанням brute-force.

- AES підтримує ключі довжиною 128, 192 або 256 біт. Вважається дуже стійким алгоритмом, який не піддається атакам з використанням brute-force.

3. Швидкість:

- RSA має високий обчислювальний навантаження, особливо при роботі з довгими ключами. Шифрування та розшифрування повідомлень RSA можуть бути повільними порівняно з AES.

- AES відомий своєю високою швидкістю обробки даних. Завдяки оптимізації та використанню сучасних криптографічних методів, AES може працювати швидше порівняно з RSA.

4. Використання:

- RSA широко використовується для забезпечення конфіденційності цифрового підпису. Використовується для обміну ключами, шифрування повідомлень та аутентифікації.

- AES використовується для шифрування та розшифрування повідомлень. Застосовується у багатьох криптографічних протоколах та системах забезпечення.

Таблиця 2.2 – Порівняльна таблиця для алгоритмів AES та RSA

Алгоритми	Критерії оцінювання				
	Стійкість	Ключова довжина	Швидкість	Використання	Сума
AES	3	2	2	3	13
RSA	2	2	2	3	10

Висновок: Загалом, якщо ключі вибрані належним чином і використовуються належні режими шифрування, як AES, так і RSA є досить стійкими алгоритмами шифрування. Однак, вибір між ними залежить від конкретних вимог до шифрування, таких як швидкість, розмір ключа, підтримка протоколів тощо.

Загалом, RSA та AES мають різні сильні сторони. RSA добре підходить для забезпечення цифрового підпису та обміну ключами, тоді як AES є ефективним для шифрування повідомлень. Обидва алгоритми широко використовуються в промислових та комерційних застосуваннях, і вибір між ними залежить від конкретних потреб безпеки та швидкості в конкретному сценарії використання.

Отже, вибір між AES і RSA залежить від конкретного застосування. У більшості випадків AES використовується для шифрування фактичних даних, тоді як RSA використовується для захисту ключів та маленьких обсягів інформації, які потрібно передавати за допомогою публічних каналів. У реальних системах шифрування часто використовують комбінацію симетричного шифрування AES та асиметричного шифрування RSA для поєднання швидкості та безпеки.

2.1.3 Порівняння алгоритму RSA та DES

Порівняння алгоритму RSA (Rivest-Shamir-Adleman) і DES (Data Encryption Standard) включає оцінку їх ефективності та стійкості. Давайте розглянемо кожен алгоритм окремо:

1. Стійкість:

- RSA заснований на складності факторизації великих простих чисел. На сьогоднішній день немає ефективних алгоритмів для швидкого факторизації великих чисел, тому RSA вважається стійким до факторизаційних атак. Успіх RSA ґрунтується на великості числа, яке використовується для генерації ключів. Чим більше число, тим більша безпека, оскільки факторизація стає важчою задачею.

- Зараз DES вважається вразливим до brute-force атак через обмежену довжину ключа. За допомогою потужних обчислювальних ресурсів, DES може бути зламаний шляхом перебору всіх можливих ключів.

2. Довжина ключа:

- Як було зазначено, RSA підтримує ключі різної довжини, зазвичай від 1024 до 4096 біт. Більша довжина ключа забезпечує більшу стійкість до brute-force атак, але також збільшує обчислювальні навантаження при генерації ключів та операціях шифрування/розшифрування.

- DES використовує ключ довжиною 56 біт, що вважається недостатньою для стійкого шифрування.

3. Швидкість:

- DES є швидшим в порівнянні з RSA, оскільки використовує меншу довжину ключа та простіші арифметичні операції.

- RSA є повільнішим у порівнянні з DES, оскільки вимагає складних обчислень, таких як піднесення до степеня за модулем.

4. Використання:

- DES широко використовувався в минулому, але сьогодні вважається застарілим і небезпечним для багатьох застосувань.

- RSA використовується для шифрування, підпису та обміну ключами. Він є одним з найпоширеніших алгоритмів асиметричного шифрування.

Таблиця 2.3 – Порівняльна таблиця для алгоритмів RSA та DES

Алгоритм	Критерії оцінки				
	Стійкість	Довжина ключа	Швидкість	Використання	Сума
RSA	3	2	1	3	9
DES	1	1	2	1	5

Висновок: RSA та DES мають різні характеристики та застосування. RSA часто використовується для шифрування малого обсягу даних та цифрового підпису, оскільки він забезпечує високий рівень стійкості до факторизаційних атак. З іншого боку, DES, хоча вважається застарілим, все ще може мати застосування у спеціалізованих ситуаціях, де вимагається швидкість та ефективність. Для більшої стійкості і безпеки рекомендується використовувати RSA з довшими ключами, а DES не рекомендується для нових систем через свою обмежену довжину ключа та вразливість до brute-force атак. Отже, для сучасних цифрових систем обробки і передачі даних, використання RSA є більш підходящим вибором порівняно з DES. RSA забезпечує вищий рівень безпеки, особливо для захисту ключів і обміну даними по відкритим каналам. Застосування RSA дозволяє забезпечити конфіденційність і цілісність даних, а також аутентифікацію та надійну обміну ключами.

2.1.4 Порівняння алгоритму AES та ECC

AES (Advanced Encryption Standard) і ECC (Elliptic Curve Cryptography) є двома різними криптографічними алгоритмами, які використовуються для різних цілей.

AES є симетричним алгоритмом шифрування, що означає, що він використовує один і той самий ключ для шифрування та розшифрування даних. Він був прийнятий як стандартний замінник DES і забезпечує високий рівень безпеки. AES використовує блокову шифрування і працює з фіксованими розмірами блоків (128 біт, 192 біти або 256 біт). Він широко застосовується в сучасних криптографічних протоколах та системах.

ECC є асиметричним алгоритмом шифрування, що базується на математичних принципах еліптичних кривих. Цей алгоритм використовує дві взаємно пов'язані ключові пари: приватний ключ та публічний ключ. ECC володіє властивістю криптографічної стійкості на основі складності задачі обчислення дискретного логарифму на еліптичних кривих. Це дозволяє досягти еквівалентного рівня безпеки з меншими розмірами ключів порівняно з іншими асиметричними алгоритмами, такими як RSA.

Основні відмінності між AES і ECC полягають у їхніх принципах роботи та використовуваних ключах.

1. Стійкість:

Якщо правильно реалізовані, обидва алгоритми забезпечують високий рівень безпеки. Щодо витрат ресурсів.

- ECC зазвичай вимагає менше обчислювальної потужності та меншого обсягу ключа, що робить його вигіднішим у деяких сценаріях. ECC також відомий своєю криптографічною стійкістю. Оскільки він базується на складності задачі обчислення дискретного логарифму на еліптичних кривих, ECC вважається ефективним та безпечним алгоритмом. Проте, стійкість ECC

також залежить від використовуваних параметрів і правильної реалізації. При виборі кривих та параметрів ECC необхідно дотримуватись рекомендацій та стандартів безпеки, оскільки некоректний вибір може призвести до вразливостей.

- AES вважається дуже безпечним симетричним алгоритмом шифрування. Він пройшов широкий розгляд і випробування, включаючи криптоаналітичні атаки, і виявився стійким до багатьох атак, коли використовується правильно. Загальноприйнятою думкою є те, що AES забезпечує достатній рівень безпеки для більшості застосувань.

Однак, слід зауважити, що безпека AES залежить від правильного вибору і реалізації ключа, режиму роботи та застосування відповідних заходів безпеки. Недоліки можуть виникнути, якщо ключі зберігаються або передаються ненадійним способом, або якщо використовуються слабкі режими шифрування.

2. Розмір ключа:

- AES використовує ключі фіксованого розміру, які залежать від варіанту (128, 192 або 256 біт).

- ECC зазвичай забезпечує еквівалентний рівень безпеки з меншими розмірами ключів порівняно з іншими асиметричними алгоритмами. У ECC розмір ключа вимірюється у бітах або байтах, в залежності від реалізації. Розмір приватного ключа: Використовуються ключі з розміром від 128 до 521 біта (наприклад, ключі P-256, P-384, P-521). Розмір публічного ключа: Розмір публічного ключа зазвичай менше, ніж розмір приватного ключа, але точні розміри залежать від використовуваної кривої.

3. Швидкодія:

- AES вважається дуже швидким алгоритмом, особливо при використанні апаратного прискорення.

- ECC також може бути досить ефективним, особливо на пристроях з обмеженими ресурсами, завдяки своїй здатності досягати еквівалентного рівня безпеки з меншими розмірами ключів.

4. Використання:

- ECC широко використовується в бездротових комунікаціях та системах з обмеженими обчислювальними ресурсами, такими як мобільні пристрої.

- AES широко використовується для шифрування даних в різних додатках, таких як захищені з'єднання з Інтернетом, безпечні протоколи обміну даними та зберігання конфіденційної інформації.

Таблиця 2.4 – Порівняльна таблиця для алгоритмів AES та ECC

Алгоритм		Критерії оцінки			
	Стійкість	Розмір ключа	Швидкість	Використання	Сума
AES	3	2	3	3	11
ECC	3	3	3	3	12

Висновок: Загалом, обидва алгоритми, AES і ECC, є високостійкими та добре вивіреними з точки зору безпеки. AES забезпечує сильне симетричне шифрування, тоді як ECC забезпечує ефективне асиметричне шифрування з меншими розмірами ключів. Вибір між ними залежить від конкретних потреб вашої системи, обчислювальних ресурсів та безпекових вимог.

Вибір між ними залежить від конкретних потреб системи та контексту застосування. З точки зору обчислень, AES є досить ефективним і швидким алгоритмом шифрування, особливо при використанні апаратного прискорення. Виконання AES-операцій майже лінійно залежить від розміру блоку шифрування. Це означає, що AES може бути швидким і оптимальним в контексті обчислень.

У випадку ECC, обчислювальна складність залежить від розміру еліптичної кривої та використовуваних алгоритмів. Зазвичай ECC потребує менше обчислювальних ресурсів порівняно з іншими асиметричними алгоритмами, такими як RSA, що робить його привабливим для пристроїв з обмеженими ресурсами.

Вибір між AES і ECC залежить від конкретних потреб і вимог системи обробки і передачі даних. Якщо вам потрібно шифрування великих обсягів даних з ефективністю і надійністю, AES може бути кращим варіантом. У той же час, якщо важлива компактність ключів та безпека при обміні даними, ECC може бути більш підходящим вибором.

В практиці часто використовують комбінацію різних алгоритмів шифрування для досягнення потрібного рівня безпеки і ефективності. Наприклад, AES може використовуватись для шифрування фактичних даних, а ECC - для обміну ключами та аутентифікації.

2.1.5 Порівняння алгоритмів ECC та DES

Порівняння алгоритму DES (Data Encryption Standard) і ECC (Elliptic Curve Cryptography) включає оцінку їхньої захищеності та ефективності. Давайте розглянемо основні аспекти кожного з них:

Порівняння принципу роботи:

DES є симетричним алгоритмом шифрування, розробленим у 1970-х роках. У свій час DES був широко використовуваним стандартом шифрування.

ECC є асиметричним алгоритмом шифрування, який базується на математичних принципах еліптичних кривих.

1. Стійкість:

- ECC зазвичай надає більшу безпеку порівняно з DES. DES є застарілим алгоритмом і має більше вразливостей з точки зору криптоаналізу. Його 56-бітний ключ занадто короткий і може бути зламаний шляхом перебору. Крім того, були виявлені інші криптоаналітичні атаки, такі як атака з відомим текстом і атака з відомим шифротекстом. ECC базується на складних математичних властивостях еліптичних кривих і забезпечує високий рівень безпеки при використанні коротких ключів порівняно з іншими криптографічними алгоритмами. Це дозволяє забезпечувати сильну криптографічну міцність при меншій довжині ключа, що робить його ефективним для використання на ресурсом обмежених пристроях, таких як мобільні пристрої. Загалом, ECC надає більшу безпеку та ефективність порівняно з DES.

- DES був розроблений в 1970-х роках і, хоча його безпека була висока на той час, він став вразливим до атак з використанням потужних обчислювальних ресурсів. DES вважається застарілим алгоритмом з низьким рівнем безпеки. При виборі алгоритму шифрування слід враховувати поточні рекомендації безпеки та вимоги конкретного застосування.

2. Довжина ключа:

- DES використовує ключ довжиною 56 бітів, що вважається недостатньою для сучасних криптографічних стандартів.

- ECC забезпечує той самий рівень безпеки, що і інші криптографічні алгоритми, такі як RSA, при використанні набагато коротших ключів. Наприклад, еквівалентний рівень безпеки RSA-ключа 2048 біт може бути досягнутий за допомогою ECC-ключа лише 224 біти.

3. Швидкість:

- DES вимагає виконання багатьох ітерацій для шифрування або розшифрування повідомлення, що може впливати на продуктивність, особливо при обробці великих обсягів даних.

- ECC використовує короткі ключі та має меншу обчислювальну складність, що призводить до ефективного виконання.

4. Використання:

- DES не використовується в нових системах через його недостатню безпеку.

- ECC є більш сучасним і безпечним алгоритмом, який застосовується в багатьох протоколах обміну ключами та криптографічних системах.

Таблиця 2.5 – Порівняльна таблиця для алгоритмів DES та ECC

Алгоритм	Критерії оцінювання				
	Стійкість	Довжина ключа	Швидкість	Використання	Сума
ECC	3	3	3	3	12
DES	1	1	2	1	5

Висновок: В порівнянні з DES, ECC надає більшу безпеку і ефективність. ECC може забезпечити еквівалентний рівень безпеки з меншими розмірами ключів порівняно з іншими асиметричними алгоритмами. ECC вважається більш безпечним і ефективним алгоритмом шифрування порівняно з DES. ECC заснований на проблемі обчислення дискретного логарифму на еліптичних кривих, яка вважається складною для розгадування. Завдяки цьому ECC може бути добрим вибором для систем з обмеженими ресурсами, де важлива швидкодія та ефективність обчислень. Використання ECC може дозволити зменшити вимоги до обчислювальних ресурсів і пропускної здатності.

Отже, з урахуванням безпеки та сучасних вимог до цифрових систем обробки і передачі даних, ECC є більш підходящим вибором порівняно з DES. ECC надає вищий рівень безпеки при меншому розмірі ключів, що забезпечує ефективність і захищеність передачі даних. Використання ECC може допомогти забезпечити конфіденційність, цілісність та аутентифікацію даних у цифрових системах.

2.1.6 Порівняння алгоритму RSA та ECC

Порівняння алгоритму RSA (Rivest-Shamir-Adleman) і ECC (Elliptic Curve Cryptography) включає оцінку їхньої захищеності, ефективності та вимог до обчислень. Ось деякі основні аспекти порівняння обох алгоритмів:

Порівняння принципу роботи:

- RSA є асиметричним алгоритмом шифрування, який базується на проблемі факторизації великих цілих чисел. Захищеність RSA залежить від складності факторизації великих чисел на прості множники. За поточних умов, RSA вважається стійким, якщо використовуються достатньо великі ключі (зазвичай більше 2048 біт).

- ECC є асиметричним алгоритмом шифрування, який базується на математичних принципах еліптичних кривих. В порівнянні з RSA, ECC надає більшу безпеку та ефективність при використанні менших розмірів ключів. ECC використовує властивості еліптичних кривих, що дозволяють забезпечити еквівалентний рівень безпеки з меншими ключами порівняно з RSA. Наприклад, еквівалентний рівень безпеки RSA ключа 3072 біта може бути досягнутий з ECC ключем всього 256 біт. Це робить ECC особливо привабливим для протоколів обміну ключами та систем з обмеженими ресурсами.

1. Стійкість:

Загалом, якщо розглядати еквівалентний рівень безпеки, ECC може забезпечувати більшу криптографічну стійкість при менших розмірах ключа порівняно з RSA. Наприклад, ключі ECC з розміром 256 біт можуть мати еквівалентний рівень безпеки ключів RSA з розміром 3072 біти. Це робить ECC більш ефективним з точки зору використання ресурсів і забезпечує високу стійкість шифрування. Проте, важливо враховувати, що криптографічна стійкість залежить не тільки від використовуваного шифру, але й від правильного використання протоколів і реалізації криптографічних систем.

2. Розмір ключа:

- Зазвичай, RSA використовуються ключі розміром 2048, 3072 або 4096 біт для надійного шифрування. Розмір публічного ключа зазвичай дорівнює розміру приватного ключа.

- У ECC розмір ключа вимірюється у бітах або байтах, в залежності від реалізації. Розмір приватного ключа: Використовуються ключі з розміром від 128 до 521 біта (наприклад, ключі P-256, P-384, P-521). Розмір публічного ключа зазвичай менше, ніж розмір приватного ключа, але точні розміри залежать від використовуваної кривої.

3. Швидкість:

- ECC є більш ефективним алгоритмом, оскільки вимагає менше ресурсів для досягнення еквівалентного рівня безпеки порівняно з RSA.

- RSA вимагає більшого обчислювального зусилля, особливо при використанні великих ключів, тоді як ECC має менші обчислювальні вимоги.

4. Використання:

- RSA широко використовується і є стандартом для багатьох криптографічних протоколів.

- ECC також отримує все більше популярності і використовується в різних застосуваннях, зокрема в сфері Інтернет речей (IoT) та мобільних пристроях.

Таблиця 2.6 – Порівняльна таблиця для алгоритмів RSA та ECC

Алгоритми	Критерії оцінювання				
	Стійкість	Розмір ключа	Швидкість	Використання	Сума
RSA	2	2	2	3	9
ECC	3	3	3	3	12

Висновок: ECC вважається кращим вибором порівняно з RSA з точки зору безпеки та ефективності. Однак, вибір між ними може залежати від конкретних потреб, обчислювальних обмежень та стандартів безпеки, що застосовуються в конкретній системі. Захищеність алгоритмів RSA і ECC залежить від використовуваних ключів. Обидва алгоритми можуть забезпечити високий рівень безпеки, якщо використовуються достатньо великі ключі. Однак, ECC зазвичай вимагає менше обчислювальних ресурсів і забезпечує еквівалентний рівень безпеки з меншими ключами порівняно з RSA. Це робить ECC більш привабливим для обмежених пристроїв з обчислювальними обмеженнями, таких як мобільні пристрої та Інтернет речей (IoT).

Важливо враховувати поточні стандарти безпеки та рекомендації при виборі алгоритму шифрування і ключів для конкретних застосувань. У цілому обидва алгоритм RSA та ECC добре підходять для інформаційних системи.

2.1.7 Порівняння алгоритмів цифрового підпису DSA та ECDSA

DSA (Digital Signature Algorithm) і ECDSA (Elliptic Curve Digital Signature Algorithm) - це два різних алгоритми цифрового підпису, які використовуються

для забезпечення цілісності, автентичності та невідмінності даних. Давайте порівняємо ці два алгоритми за декількома ключовими аспектами:

1. Стійкість:

- Якщо використовуються достатньо великі ключі і встановлені правильні параметри, як DSA, так і ECDSA забезпечують високий рівень стійкості.

- Однак, ECDSA вважається більш майбутньоорієнтованим алгоритмом, оскільки він може забезпечувати еквівалентний рівень безпеки з меншими ключами порівняно з DSA.

2. Розмір ключів:

- DSA працює зі значеннями параметрів p, q і g. Загальний розмір ключа DSA визначається значенням q, яке зазвичай становить 160 біт або 256 біт.

- ECDSA працює з ключами, пов'язаними з еліптичними кривими. Розмір ключа ECDSA визначається розміром еліптичної кривої, і він може бути значно меншим за DSA для еквівалентного рівня безпеки. Наприклад, ключ ECDSA розміром 256 біт може забезпечувати еквівалентний рівень безпеки, як DSA з ключем 3072 біта.

3. Ефективність обчислень:

- DSA використовує модульні операції, такі як піднесення до степеня і модульне множення. Обчислення в DSA можуть бути трохи повільнішими порівняно з ECDSA.

- ECDSA використовує операції над точками на еліптичних кривих, що зазвичай вимагає менше обчислювальних ресурсів. ECDSA зазвичай швидший за DSA, особливо при використанні більших ключів.

4. Використання:

- DSA традиційно використовується в багатьох криптографічних протоколах та стандартах, таких як SSH, TLS.

- ECDSA отримує все більшу популярність, особливо в обмежених пристроях з обчислювальними обмеженнями, таких як мобільні пристрої та IoT.

Таблиця 2.7 – Порівняльна таблиця для алгоритмів DSA та ECDSA

Алгоритм	Критерії оцінювання				
	Стійкість	Розмір ключа	Швидкодія	Використання	Сума
DSA	3	3	2	3	11
ECDSA	3	3	3	3	12

Висновок: В цілому, ECDSA вважається більш ефективним і майбутньоорієнтованим алгоритмом порівняно з DSA. Вибір між ними може залежати від конкретних вимог, обчислювальних обмежень та стандартів безпеки, що застосовуються в конкретній системі.

2.1.8 Порівняння алгоритмів аутентифікації HMAC та SSL/TLS

HMAC (Hash-based Message Authentication Code) і SSL/TLS (Secure Sockets Layer/Transport Layer Security) - це два різних алгоритми, які використовуються для забезпечення аутентифікації та цілісності даних в різних контекстах. Давайте порівняємо ці два алгоритми за декількома ключовими аспектами:

1. Функціональність:

- HMAC є алгоритмом аутентифікації повідомлень, який використовує хеш-функції для генерації аутентичного коду повідомлення. HMAC гарантує цілісність та автентичність повідомлення шляхом порівняння отриманого аутентичного коду зі збереженим значенням.

- SSL/TLS, з іншого боку, є протоколом захищеного з'єднання, який включає шифрування, аутентифікацію та цілісність даних. SSL/TLS забезпечує

безпечний канал зв'язку між клієнтом і сервером, використовуючи симетричне і асиметричне шифрування, аутентифікацію сертифікатами та обмін ключами.

2. Використання:

- HMAC зазвичай використовується для аутентифікації повідомлень в різних протоколах та системах. Він може бути застосований до будь-якого повідомлення, яке потребує перевірки цілісності.

- SSL/TLS використовується для захищеного з'єднання між клієнтом і сервером в мережевих протоколах, таких як HTTPS для безпечного передавання веб-сторінок. SSL/TLS забезпечує шифрування даних, аутентифікацію сторінок за допомогою сертифікатів та інші механізми безпеки.

3. Залежність від ключів:

- HMAC використовує секретний ключ, який використовується для генерації аутентичного коду повідомлення. Цей ключ повинен бути захищений і відомий лише аутентифікуючим сторонам.

- SSL/TLS використовує асиметричну криптографію з використанням пари ключів: приватного та публічного. Приватний ключ зберігається на сервері, а публічний ключ розповсюджується у сертифікатах, які використовуються для аутентифікації та обміну ключами.

4. Переваги та недоліки:

- HMAC є простим і ефективним алгоритмом для аутентифікації повідомлень, але він не надає шифрування та захисту від прослуховування.

- SSL/TLS забезпечує не тільки аутентифікацію повідомлень, але й шифрування даних для захисту від прослуховування та захист від підробки сертифікатів. Однак, встановлення SSL/TLS-з'єднання може бути більш складним та вимагати додаткових обчислювальних ресурсів.

Таблиця 2.8 – Порівняльна таблиця для алгоритмів HMAC та SSL/TLS

Алгоритми	Критерії оцінювання			
	Функціональність	Залежність від ключів	Використання	Сума
HMAC	3	3	3	9
SSL/TLS	3	3	3	9

Висновок: Обидва алгоритми, HMAC і SSL/TLS, мають свої унікальні застосування та переваги, і їх вибір залежить від конкретних вимог щодо безпеки та контексту застосування.

Отже, HMAC та SSL/TLS виконують різні функції в контексті безпеки цифрових систем обробки та передачі даних. HMAC забезпечує аутентифікацію повідомлення та перевірку цілісності даних, в той час як SSL/TLS забезпечує шифрування та захист від перехоплення даних під час їх передачі по мережі. Обидва механізми можуть бути використані разом для забезпечення комплексної безпеки в системах обробки та передачі даних.

2.2 Висновки

Таблиця 2.9 – Порівняльна таблиця для розглянутих алгоритмів шифрування

Алгоритми	Критерії оцінки				
	Стійкість	Довжина ключа	Швидкість	Використання	Сума
AES	9	6	7	9	31
DES	3	3	5	3	14
RSA	7	6	5	9	27
ECC	9	9	9	9	36

Висновок до порівняння алгоритмів шифрування AES, DES, ECC і RSA полягає в наступному:

AES (Advanced Encryption Standard) є сучасним симетричним алгоритмом шифрування, який надає високий рівень безпеки, швидкодії та ефективності. Він замінив застарілий DES як стандартний алгоритм шифрування для багатьох застосувань. Є гарним варіантом серед інших алгоритмів шифрування.

DES (Data Encryption Standard) є старим симетричним алгоритмом шифрування, розробленим у 1970-х роках. Хоча DES був довгий час використовуваний, він вважається застарілим і недостатньо безпечним для сучасних систем. Рекомендується використовувати AES замість DES. DES виявився менш швидким та менш надійним алгоритмом шифрування ніж інші, більш сучасні.

ECC (Elliptic Curve Cryptography) є асиметричним алгоритмом шифрування, який базується на математичних властивостях еліптичних кривих. Він надає високий рівень безпеки при використанні коротких ключів, що робить його ефективним для пристроїв з обмеженими ресурсами. Загалом, даний алгоритм шифрування виявився одним з найкращих та серед інших алгоритмів, які було розглянуто. Його криптографічна стійкість та швидкість на високому рівні.

RSA (Rivest-Shamir-Adleman) також є асиметричним алгоритмом шифрування, який використовується для шифрування та підпису даних. Він базується на обчислювальній складності факторизації великих простих чисел. RSA забезпечує високий рівень безпеки, але вимагає довгих ключів для забезпечення еквівалентного рівня безпеки, порівняно з ECC. В цілому даний алгоритм є непоганим, але тим не менш поступається компактністю ключів і швидкістю в порівнянні з більш потужними алгоритмам такими як ECC.

Загалом, AES є найбільш популярним і рекомендованим алгоритмом шифрування для багатьох застосувань. ECC є ефективним алгоритмом шифрування з короткими ключами та підходить для ресурсом обмежених

пристроїв. RSA залишається важливим для підпису та обміну ключами. DES вважається застарілим і не рекомендується для використання у нових системах.

Остаточний вибір алгоритму повинен бути зроблений з урахуванням специфіки системи, вимог до безпеки та ефективності, а також підтримки та актуальності алгоритму у галузі криптографічного захисту інформації.

Також, у розділі було розглянуто та порівняно алгоритми цифрового підпису DSA (Digital Signature Algorithm) та ECDSA (Elliptic Curve Digital Signature Algorithm). Нижче наведено деякі висновки, які можна зробити на основі порівняння цих алгоритмів:

ECDSA, як вказує сама назва, базується на еліптичних кривих, що дозволяє йому пропонувати вищу ефективність в порівнянні з DSA, який використовує більш традиційні математичні операції. У результаті ECDSA зазвичай вимагає менше обчислювальних ресурсів та менше обсягу даних для передачі при забезпеченні того ж рівня безпеки.

Обидва алгоритми вважаються криптографічно стійкими, але безпека їх базується на різних математичних основах. DSA використовує проблему дискретного логарифмування, тоді як ECDSA ґрунтується на проблемі дискретного логарифмування на еліптичних кривих. З огляду на швидкий розвиток криптоаналізу, важливо обирати достатньо довгі ключі та параметри, щоб запобігти можливим атакам.

Один з переваг ECDSA полягає в тому, що він може забезпечувати той самий рівень безпеки, що й DSA, використовуючи ключі меншого розміру. Наприклад, ECDSA з ключами довжиною 256 біт може забезпечити рівень безпеки, що вимагається для DSA з ключами довжиною 3072 біти. Це дозволяє зменшити вимоги до обсягу даних для передачі та обчислювальних ресурсів.
DSA є стандартом Федерального уряду США та використовується у багатьох застосунках, включаючи протоколи TLS/SSL для безпечного обміну даними. ECDSA також отримав широке визнання і використовується, наприклад, в стандарті ECIES для шифрування даних та в стандарті ECDSA для цифрового підпису.

Загалом, обидва алгоритми мають свої переваги та використовуються у різних контекстах. ECDSA набуває все більшої популярності завдяки своїй ефективності та можливості забезпечити високий рівень безпеки з меншими ключами. Однак вибір між DSA та ECDSA повинен залежати від конкретного контексту застосування, безпекових вимог та стандартів, які ви прагнете виконати.

У розділі було розглянуто та порівняно алгоритми аутентифікації HMAC (Hash-based Message Authentication Code) та SSL/TLS (Secure Sockets Layer/Transport Layer Security). Основні висновки, які можна зробити з цього порівняння, включають наступне:

HMAC та SSL/TLS є двома різними алгоритмами, які виконують різні функції в контексті безпеки. HMAC використовується для аутентифікації повідомлення шляхом обчислення коду перевірки цілісності за допомогою хеш-функції та спільного секретного ключа. SSL/TLS, з іншого боку, є протоколом забезпечення безпеки для захищеного обміну даними в мережі, що включає аутентифікацію сервера, шифрування та аутентифікацію клієнта.

HMAC використовується в різних контекстах для аутентифікації повідомлень, таких як передача даних по мережі або перевірка цілісності файлів. SSL/TLS, з іншого боку, використовується для захищеної комунікації між клієнтом та сервером через мережу, включаючи веб-сайти, електронну пошту та інші додатки.

Обидва алгоритми використовуються для забезпечення безпеки, але на різних рівнях. HMAC забезпечує аутентифікацію повідомлень та перевірку їх цілісності, що допомагає виявити будь-які зміни або модифікації даних. SSL/TLS, з іншого боку, надає комплексну захист від атак, включаючи аутентифікацію сторін, шифрування даних та забезпечення цілісності.

В порівнянні з HMAC, SSL/TLS є більш складним і більш широкою системою, оскільки включає в себе протоколи, криптографічні алгоритми та інші компоненти. HMAC, з іншого боку, є відносно простим алгоритмом аутентифікації повідомлень.

Загалом, HMAC та SSL/TLS є важливими компонентами безпеки, які можуть використовуватись у поєднанні або окремо, в залежності від конкретного контексту та вимог безпеки. HMAC часто використовується для аутентифікації повідомлень у різних додатках, тоді як SSL/TLS є ключовим протоколом для захищеної комунікації через мережу.

РОЗДІЛ 3. ЕКОНОМІЧНА ЧАСТИНА

Метою розрахунків є економічне обґрунтування доцільності проведення дослідження алгоритмів криптографічного шифрування. Необхідно визначити розмір капітальних та експлуатаційних витрат на дослідження, можливі збитки у випадку втрати конфіденційної інформації компанії.

3.1 Розрахунок капітальних витрат

Визначення трудомісткості розробки політики безпеки інформації.
Для того, щоб розрахувати витрати за аналіз даних алгоритмів шифрування, скористуємось даною формулою:

$$t = t_{тз} + t_{в} + t_{а} + t_{вз} + t_{озб} + t_{овр} + t_{д}, \text{ годин} \quad (3.1)$$

$t_{тз}$ – тривалість складання технічного завдання на дослідження алгоритмів шифрувань в цифрових системах складає 16 годин;

$t_{в}$ – тривалість розробки концепції захисту цифрового середовища складає 12 годин;

$t_{а}$ - тривалість дослідження стійкості алгоритмів шифрувань в цифрових системах 48 годин;

$t_{озб}$ – тривалість вибору рішень що до впровадження певного алгоритму шифрування складає 8 годин;

$t_{овр}$ – тривалість організації виконання відновлювальних робіт складає 5 годин;

$t_{д}$ – тривалість документального оформлення запровадженого алгоритму шифрування складає 7 годин;

$$t = 16 + 12 + 48 + 8 + 5 + 7 = 96 \text{ годин}$$

3.2 Розрахунок витрат на дослідження та впровадження алгоритму шифрування.

Включає в себе заробітну плату спеціаліста з інформаційної безпеки Ззп і вартості витрат машинного часу.

$$Крп = Ззп + Змч \quad (3.2)$$

$$Крп = 15\ 360 + 556{,}8 = 15\ 916{,}8 \text{ грн.};$$

$$З\ зп = t * З\ іб, \text{ грн.}; \quad (3.3)$$

$$Ззп = 96 * 160 = 15\ 360 \text{ грн.};$$

t – загальна тривалість розробки політики безпеки.

Зіб – середньогодинна заробітна плата спеціаліста з інформаційної безпеки. Грн/годину.

96 - годин на дослідження

160 – грн заробітна плата грн / годину;

Вартість машинного часу для розробки політики безпеки інформації на ПК визначається за формулою:

$$Змч = t * Смч, \text{ грн} \quad (3.4)$$

$$Змч = 96 * 5{,}8 = 556{,}8 \text{ грн.}$$

Де t – трудомісткість дослідження алгоритмів шифрування, годин;

Смч - вартість 1 години машинного часу ПК, грн./година.

$$Смч = Р * tнал * Се + (Фзал * На)/Fр + (Клпз * Напз)/Fр, \text{ грн.} \quad (3.5)$$

См = 0,7 * 3 * 2,12 + (40 * 0,8) / 2800 + (5399 * 0,7) / 2800 = 4,452 + 0,0114 + 1,34975 = 5,8 грн/год

де Р - встановлена потужність ПК. кВт. 0,7 кВт;

tнал – кількість задіяних роб.станцій при дослідженні; tнал – 3;

Се – тариф на електричну енергію, грн/кВт година; Се – 2,12 грн/кВт год;
Фзал – залишкова вартість ПК на поточний рік, грн.; Фзал – 40 грн;
На – річна норма амортизації на ПК, частки одиниці; На – 0,8;
Напз – річна норма амортизації на ліцензійне програмне забезпечення, частки одиниці; Напз – 0,7;

Клпз – вартість ліцензійного програмного забезпечення, грн.; Клпз – 5399 грн

Fр – річний фонд робочого часу (за 40-годинного робочого тижня Fр = 2800).

3.3 Розрахунок потенційних збитків у випадку відсутності досліджень алгоритмів шифрування на прикладі компанії, що яка володіє персональними даними клієнтів

Причини збитків компанії:

- Втрата даних клієнтів.
- Втрата корпоративних даних компанії.

Витяг з Закону України «Про захист персональних даних».

«Відповідно до ч. 1 ст. 24 Закону, володільці, розпорядники персональних даних та треті особи зобов'язані забезпечити захист цих даних від випадкових втрати або знищення, від незаконної обробки, у т. ч. незаконного знищення чи доступу до них. Відповідно до ст. 22 Закону, контроль за дотриманням законодавства про захист персональних даних здійснюють уповноважений Верховної Ради України з прав людини у сфері захисту персональних даних (далі — Уповноважений) та/або суди.»

Витяг з Кодексу України про адміністративні правопорушення:

«Недодержання встановленого законодавством про захист персональних даних порядку захисту персональних даних, що призвело до незаконного доступу до них або порушення прав суб'єкта персональних даних, тягне за собою накладення штрафу на громадян від ста до п'ятисот неоподатковуваних мінімумів доходів громадян і на посадових осіб, громадян, суб'єктів підприємницької діяльності від трьохсот до однієї тисячі неоподатковуваних мінімумів доходів громадян.

Повторне протягом року вчинення порушення, передбаченого частиною четвертою цієї статті, за яке особу вже було піддано адміністративному стягненню, тягне за собою накладення штрафу від однієї тисячі до двох тисяч неоподатковуваних мінімумів доходів громадян.»

Отже, за дане правопорушення накладається штраф у розмірі від ста до п'ятисот неоподаткованих мінімумів доходів компанії.

Приблизна кількість випадків на рік: 23.

Неоподаткований мінімум становить – 17 грн.

Отже сума штрафу складає 17 * 400 = 6 800 грн.

Отже потенційні витрати за рік 6 800 * 23 = 156 400 грн.

3.4 Оцінка величини збитку

Приведемо приклад компанії, в якій не було проведено дослідження алгоритмів шифрування. Що призвело до атаки та систему зберігання і передачі даних.

Упущена вигода від простою атакованого сегмента корпоративної мережі становить:

$$U = Пп + Пв + V, \quad (3.5)$$

$$U = 994 + 4\,773 + 19000 = 24\,767 \text{ грн.}$$

Де Пп - оплачувані втрати робочого часу та простої співробітників атакованого вузла або сегмента корпоративної мережі, грн; Пп –грн.;

Пв - вартість відновлення працездатності сегмента корпоративної мережі, грн; Пв – грн.;

V - втрати від зниження обсягу продажів за час простою атакованого вузла або сегмента корпоративної мережі, грн. V – грн.

Втрати від зниження продуктивності співробітників атакованого сегмента корпоративної мережі являють собою втрати їхньої заробітної плати (оплата непродуктивної праці) за час простою внаслідок атаки:

$$Пп = (\Sigma Зс/F)*tп, \text{ грн.}, \quad (3.6)$$

$$Пп = (35\,000 /176)*5 = 994$$

де Зс – загальна кількість витрат на заробітну плату співробітників за місяць, Зс – 35 000 грн;

F – 176

tп – час простою внаслідок атак, tп – 5 год.

Витрати на повторне введення інформації Пви розраховуються виходячи з розміру заробітної плати співробітників атакованого вузла або сегмента корпоративної мережі Зс, які зайняті повторним введенням втраченої інформації, з урахуванням необхідного для цього часу tви:

$$Пви = (\Sigma Зс/F)*tви, \quad (3.7)$$

$$Пви = (35\,000 / 176) * 24 = 4\,773 \text{ грн.};$$

Витрати на відновлення вузла або сегмента корпоративної мережі Ппв визначаються часом відновлення після атаки tв і розміром середньогодинної заробітної плати обслуговуючого персоналу (адміністраторів):

$$Ппв = (\Sigma Зо/F)*tв \quad (3.8)$$

де Зо = заробітна плата системного адміністратора, 9000 грн на місяць;

F – місячний фонд робочого часу (при 40–а годинному робочому тижні становить 176 ч.);

tв = 24 годин повторного введення загубленої інформації унаслідок атаки;

$$Ппв = (9000 / 176) * 24 = 1\ 227 \text{ грн};$$

Пзч – вартість заміни устаткування або запасних частин складає 20000 грн.

Витрати на відновлення працездатності вузла або сегмента корпоративної мережі включають кілька складових:

$$Пв = Пви + Ппв + Пзч \quad (3.9)$$

де Пви – витрати на повторне уведення інформації, грн.;

Ппв – витрати на відновлення вузла або сегмента корпоративної мережі, грн;

Пзч – вартість заміни устаткування або запасних частин, грн.

Пв = 4 773 + 1 227 + 20000 = 26 000 грн.;

Втрати від зниження очікуваного обсягу прибутків за час простою атакованого вузла або сегмента корпоративної мережі визначаються виходячи із середньогодинного обсягу прибутку і сумарного часу простою сегмента корпоративної мережі:

$$V = (O / Fr) * (tп + tв + tви), \quad (3.10)$$

$$V= (400000 / 2200)*(5 + 24 + 24) = 182 * 53 = 9\ 636 \text{ грн.};$$

де О – обсяг продажів атакованого вузла або сегмента корпоративної мережі, О – 400000 грн у рік;

Fг – річний фонд часу роботи організації; Fr – 2200 год.

tп – час простою вузла унаслідок атаки; tп – 5 год.;

tви = час відновлення після атаки персоналом, що обслуговує корпоративну мережу; tви – 24 год.;

I – число атакованих вузлів або сегментів корпоративної мережі. 3 вузли

Таким чином, загальний збиток від атаки на вузол або сегмент корпоративної мережі організації складає:

$$B = \Sigma i \, \Sigma n * U \quad (3.11)$$

$$B = 24\,767 * 14 * 3 = 1\,040\,214 \text{ грн}$$

Отже, загальні збитки можуть скаладати:

$$B = B_1 * R_1 + B_2 * R_2 \quad (3.11)$$

$$B = 156\,400 * 0{,}063 + 1\,040\,214 * 0{,}038 = 9\,853 + 39\,528 = 49\,381 \text{грн}$$

3.5 Висновок

У даному розділі було проведено підрахунок капітальних витрат на дослідження алгоритмів криптографічного шифрування та потенційних збитків у разі відсутності проведення даного дослідження. Дослідження алгоритмів криптографічного шифрування дозволяє виявляти потенційні уразливості і покращувати систему захисту інформації. Це допомагає запобігти можливим атакам і злому шифрування, збільшує стійкість системи та забезпечує конфіденційність даних. У випадку відсутності проведення дослідження алгоритмів криптографічного шифрування можуть виникнути серйозні наслідки. Відсутність оновлення алгоритмів та незнання потенційних вразливостей можуть призвести витоку даних, фінансових втрат і порушення довіри користувачів. Таким чином, інвестиції в дослідження алгоритмів криптографічного шифрування є критично важливими для забезпечення безпеки інформації. Ці витрати виправдовуються необхідністю забезпечення стійкості системи та запобігання потенційним загрозам. Відсутність таких досліджень може мати серйозні наслідки, які загрожують інформаційній безпеці та довірі до системи шифрування. Згідно розрахунків, загальні збитки у разі

здійснення атаки на інформаційну систему компанії складають 49 381 грн, а загальні капітальні витрати становлять 15 916,8 грн.

ВИСНОВОК

У рамках даної кваліфікаційної роботи було проведено комплексне дослідження алгоритмів криптографічного шифрування в цифрових системах. Дане дослідження було проведено з метою подальших вдосконалення захисту інформаційних систем, в яких зберігається, передається та обробляється конфіденційна інформація компаній. Результати дослідження свідчать про важливість використання ефективних алгоритмів шифрування для забезпечення безпеки інформаційної системи. Перш за все, були визначені основні принципи криптографічного шифрування та його роль у сучасних цифрових системах. Для цього було проведено аналіз різних алгоритмів шифрування, зокрема симетричних та асиметричних, таких як AES, DES, RSA, ECC. Також було проведено аналіз таких алгоритмів цифрового підпису як DSA та ECDSA. Крім того було розглянуто алгоритми аутентифікації HMAC та SSL/TLS. У результаті дослідження було встановлено, що ефективність алгоритмів криптографічного шифрування залежить від таких факторів, як довжина ключа, рівень складності алгоритму та його стійкість до атак. На основі отриманих результатів можна зробити висновок, що дослідження алгоритмів криптографічного шифрування є важливою складовою в області інформаційної безпеки. Використання ефективних алгоритмів дозволяє забезпечити конфіденційність, цілісність та доступність інформації в інформаційних системах. Дані результати можуть бути використані для подальшого вдосконалення криптографічних алгоритмів, розробки нових методів шифрування та застосування їх у практичних сферах.

ПЕРЕЛІК ПОСИЛАНЬ

1) Alex Biryukov., Dmitry Khovratovich. Related-key Cryptanalysis of the Full AES-192 and AES-256. Luxembourg, 2009. page 2 - 4

2) Baivab Kumar Jena. What Is AES Encryption and How Does It Work? [Електронний ресурс] - https://www.simplilearn.com/tutorials/cryptography-tutorial/aes-encryption#:~:text=The%20AES%20Encryption%20algorithm%20(also,together%20to%20form%20the%20ciphertext (Дата звернення) .

3) Стандарт шифрування DES [Електронний ресурс] - http://studcon.org/standart-shyfruvannya-des (Дата звернення 04.04.2023).

4) RSA [Електронний ресурс] - https://uk.wikipedia.org/wiki/RSA (Дата звернення 10.04.2023).

5) Алгоритм Elliptic Curve Cryptography - ECC быстрее и надёжнее широко используемых аналогов [Електронний ресурс] - https://proverkassl.com/book_algoritm_ecc.html (Дата звернення 12.04.2023).

6) Elliptic Curve Cryptography Definition [Електронний ресурс] - https://avinetworks.com/glossary/elliptic-curve-cryptography/ (Дата звернення 15.04.2023).

7) Digital Signature Algorithm [Електронний ресурс] - https://en.wikipedia.org/wiki/Digital_Signature_Algorithm (Дата звернення 23.04.2023).

8) ECDSA: Elliptic Curve Signatures [Електронний ресурс] - https://cryptobook.nakov.com/digital-signatures/ecdsa-sign-verify-messages (Дата звернення 25.04.2023).

9) HMAC Algorithm in Cryptography [Електронний ресурс] - https://www.tutorialandexample.com/hmac-algorithm-in-cryptography (Дата звернення 27.04.2023).

10) HMAC [Електронний ресурс] - https://en.wikipedia.org/wiki/HMAC

11) Transport Layer Security [Електронний ресурс] - https://en.wikipedia.org/wiki/Transport_Layer_Security (Дата звернення 29.04.2023).

12) A complete overview of SSL/TLS and its cryptographic system [Електронний ресурс] - https://dev.to/techschoolguru/a-complete-overview-of-ssl-tls-and-its-cryptographic-system-36pd (Дата звернення 30.04.2023).

ДОДАТОК А. ВІДОМІСТЬ МАТЕРІАЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

№	Формат	Найменування	Кількість листів	Примітка
1	А4	Реферат	2	
2	А4	Список умовних скорочень	1	
3	А4	Зміст	2	
4	А4	Вступ	2	
5	А4	Стан питання. Постановка задачі	39	
6	А4	Спеціальна частина	25	
7	А4	Економічний розділ	7	
8	А4	Висновки	1	
9	А4	Перелік посилань	2	
10	А4	Додаток А	1	
11	А4	Додаток Б	1	
12	А4	Додаток В	1	
13	А4	Додаток Г	2	

ДОДАТОК Б. Перелік документів на оптичному носії

1 Презентація Довбиш_презентація.ppt

2 Диплом Довбиш_диплом.pdf

3 Диплом Довбиш_диплом.docx

ДОДАТОК В. ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

Відгук керівника економічного розділу

Економічний розділ виконаний відповідно до вимог, які ставляться до кваліфікаційних робіт, та заслуговує на оцінку 90 б. («відмінно»).

Керівник розділу _____ доц. Пілова Д.П.

(підпис) (ініціали, прізвище)

ДОДАТОК Г. Відгук керівника кваліфікаційної роботи

В І Д Г У К

на кваліфікаційну роботу студентки групи 125-19-1

Довбиш Ольга Андріївна

на тему: «Дослідження алгоритмів криптографічного захисту інформації в електронних системах зберігання і передачі даних»

Пояснювальна записка складається зі вступу, трьох розділів і висновків, викладених на 90 сторінках.

Метою кваліфікаційної роботи є дослідження криптографічних алгоритмів шифрування, аутентифікації та цифрового підпису для їх використання в цифрових системах обробки і передачі інформації.

Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності бакалавра спеціальності 125 «Кібербезпека». Для досягнення поставленої мети в кваліфікаційній роботі вирішуються наступні задачі: проаналізовано загальні відомості про криптографію, наведено принципи роботи та приклади алгоритмів криптографічного захисту інформації AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS. Проведено порівняння алгоритмів криптографічного захисту інформації AES, DES, RSA, ECC; DSA, ECDSA; HMAC, SSL/TLS. За наслідками досліджень зроблено висновки щодо рішення поставленої задачі.

Практична значимість роботи полягає в дослідженні алгоритмів шифрування і цифрового підпису для визначення найкращого з них для покращення якості безпеки в інформаційних системах підприємств.

Оформлення пояснювальної записки до кваліфікаційної роботи виконано з незначними відхиленнями від стандартів.

За час дипломування Довбиш О. А. проявила себе фахівцем, здатним самостійно вирішувати поставлені задачі та заслуговує присвоєння кваліфікації бакалавра за спеціальністю 125 Кібербезпека, освітньо-професійна програма «Кібербезпека» .

Рівень запозичень у кваліфікаційній роботі не перевищує вимог “Положення про систему виявлення та запобігання плагіату”.

Кваліфікаційна робота заслуговує оцінки «_____».

Керівник кваліфікаційної роботи Герасіна О. В.

Керівник спец. розділу Герасіна О. В.