МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра кібербезпеки

ПАСТУХ Тарас Ігорович

Алгоритми захисту інформації в системах відеоспостереження / Information Security Algorithms in Video Surveillance Systems

спеціальність: 125 – Кібербезпека

освітньо-професійна програма – Кібербезпека

Кваліфікаційна робота

Виконав студент групи
КБзм -21
Т.І. Пастух

Науковий керівник
д.т.н., професор В.В.Яцків

Кваліфікаційну роботу
допущено до захисту:

«____» _____ ___2023 р.

Завідувач кафедри
_____ **В.В.Яцків**

ТЕРНОПІЛЬ - 2023

Факультет комп'ютерних інформаційних технологій
Кафедра кібербезпеки
Освітній ступінь «магістр»
спеціальність: 125 – Кібербезпека
освітньо-професійна програма – Кібербезпека

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ В.В.Яцків
«____» _____ 2022 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

ПАСТУХ Тарас Ігорович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи:
Алгоритми захисту інформації в системах відеоспостереження / Information Security Algorithms in Video Surveillance Systems
керівник роботи д.т.н., професор В.В. Яцків
затверджені наказом по університету від 1 грудня 2022 року №

2. Строк подання студентом закінченої кваліфікаційної роботи 1 грудня 2023 р.

3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити:
– провести аналіз підходів до захисту даних в системах відеоспостереження;
– дослідити технології побудови систем відеоспостереження;
– проаналізувати основні криптографічні примітиви для шифрування зображень;
– розробити структуру системи відеоспостереження на основі Інтернету речей;
– розробити загальний алгоритм шифрування зображення;
– розробити та дослідити шифрування зображень на Raspberry Pi.
5. Перелік графічного матеріалу у роботі:
– структура системи відеоспостереження;
– порівняння алгоритмів шифрування AES-ECB та AES-CBC;

– схема роботи коду автентифікації повідомлення на основі хешу;
– протокол автентифікації керування сеансом;
– перевірка чутливості ключів алгоритму AES-CBC.

6. Консультанти розділів кваліфікаційної роботи

	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 8 грудня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строки виконання етапів кваліфікаційної роботи	Примітка
1	Аналіз захисту даних в системах відеоспостереження	12.2022 р. – 03.2023 р.	
2	Теоретичні основи шифрування зображень	03.2023р. – 05.2023 р.	
3	Розробка та дослідження системи відеоспостереження	05.2023 р. – 11.2023 р.	

Студент _____ Пастух Т.І.
(підпис)

Керівник роботи _____ д.т.н., професор В.В. Яцків
(підпис)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Алгоритми захисту інформації в системах відеоспостереження» на здобуття освітнього ступеня «Магістр» зі спеціальності 125 «Кібербезпека» освітньо-професійної програми «Кібербезпека» написана обсягом 81 сторінок і містить 17 ілюстрації, 2 таблиці, 1 додаток та 37 джерел за переліком посилань.

Метою кваліфікаційної роботи є підвищення ефективності алгоритмів захисту інформації в системах відеоспостереження.

Методи досліджень. Для розв'язання поставлених задач у даній кваліфікаційній роботі використано: методи криптографічного захисту інформації, методи шифрування зображень, методи оцінки захищеності зображень. Результати дослідження: Удосконалено алгоритм захисту конфіденційності та цілісності зображення за рахунок використання алгоритму шифрування AES-CBC, який забезпечує шифрування.

Реалізовано та досліджено шифрування зображень на базі одноплатного комп'ютера Raspberry Pi. Проведено оцінку пропускної здатності при передачі зображень в системах.

Результати роботи можуть бути застосовані при розробці захищеної системи відеоспостереження.

Ключові слова: ШИФРУВАННЯ, СИСТЕМИ ВІДЕОСПОСТЕРЕЖЕННЯ, АЛГОРИТМ, КОНФІДЕНЦІЙНІСТЬ, АВТЕНТИФІКАЦІЯ.

ABSTRACT

Qualification work on "Information Security Algorithms in Video Surveillance Systems" in the specialty 125 "Cybersecurity" educational and professional program "Cybersecurity" is written in 81 pages and contains 17 illustrations, 2 tables, 1 appendice and 37 source according to the list of links.

The subject of research is algorithms for protecting confidentiality, integrity and availability of video data.

Research methods. To solve the tasks in this qualification work, the following methods of cryptographic information protection, image encryption methods, image security assessment methods were used.

Scientific novelty of the obtained results. The algorithm for protecting the confidentiality and integrity of the image has been improved, which, due to the use of the AES-CBC encryption algorithm, provides a different cipher at the output with the same input data.

Practical significance of the obtained results. Image encryption based on a Raspberry Pi single-board computer has been implemented and investigated. An assessment of the throughput during image transmission in the systems was carried out.

Keywords: ENCRYPTION, VIDEO SURVEILLANCE SYSTEMS, ALGORITHMS, CONFIDENTIALITY, AUTHENTICATIONS.

ЗМІСТ

ВСТУП

Актуальність роботи. За останнє десятиліття системи відеоспостереження та безпеки пройшли довгий шлях завдяки все більшій повсюдності Інтернет-протоколу (IP) та Інтернету речей (IoT). Надсилаючи та отримуючи дані безпосередньо через Інтернет і пропонуючи такі розширені функції, як датчики руху, хмарне сховище, відеоаналітика та автоматичні сповіщення, ці системи забезпечують надійний захист промислових і виробничих об’єктів, державних установ тощо [1, 2].

Але, незважаючи на свої переваги, системи IP-відео також мають значні ризики для безпеки, якщо вони використовують громадську інфраструктуру, оскільки вони дають кіберзлочинцям легкі шляхи атак. Вони також мають різноманітні топології та технології, які роблять їх більш складними та збільшують їхню «поверхню атаки». Зрештою, вони роблять користувачів уразливими до DDoS MitM, порушень конфіденційності, встановлення шкідливого програмного забезпечення та витоку даних [3].

Враховуючи кількість і масштаб інформації, яку державні органи збирають про громадян, зловмисники завжди прагнуть викрасти або викрити записи. Оскільки системи IP-відеоспостереження не є на 100% безпечними, отже, щоб мінімізувати свої ризики та захистити свої приміщення, дані та користувачів, необхідно розробка нових підходів та алгоритмів захисту даних в системах відеоспостереження.

Мета і завдання дослідження. Метою роботи є підвищення ефективності алгоритмів захисту інформації в системах відеоспостереження.

Досягнення визначеної мети передбачає вирішення таких завдань:

– провести аналіз підходів до захисту даних в системах відеоспостереження;

– дослідити технології побудови систем відеоспостереження;

– проаналізувати основні криптографічні примітиви для шифрування зображень;

– розробити структуру системи відеоспостереження на основі IoT;

– розробити загальний алгоритм шифрування зображення;

– розробити та дослідити шифрування зображень на Raspberry Pi.

Об’єкт дослідження – процеси шифрування та передавання зображень в системах відеоспостереженнях.

Предмет дослідження – алгоритми захисту конфіденційності, цілісності та доступності відеоданих.

Методи досліджень. Для розв’язання поставлених задач у даній кваліфікаційній роботі використано: методи криптографічного захисту інформації, методи шифрування зображень, методи оцінки захищеності зображень.

Наукова новизна одержаних результатів. Удосконалено алгоритм захисту конфіденційності та цілісності зображення, який за рахунок використання алгоритму шифрування AES-CBC забезпечує на виході різний шифр при однакових вхідних даних.

Практичне значення отриманих результатів. Реалізовано та досліджено шифрування зображень на базі одноплатного комп’ютера Raspberry Pi. Проведено оцінку пропускної здатності при передачі зображень в системах.

Публікації та апробація КР.

1. Пастух Т.І., Голод Ю.В., Мачуляк М.В. Алгоритм захисту інформації в системах відеоспостереження. Матеріали науково-практична конференція молодих вчених, аспірантів та студентів «Кібербезпека та комп’ютерно-інтегровані технології» (КБКІТ - 2023), Тернопіль, 2023. – С. 175-176.

2. Пастух Т.І., Дзівак О.А., Понедєльніков Г.М. Автентифікації та перевірка цілісності зображень на основі хешу. Матеріали науково-практичного симпозіуму «Захист інформації», Тернопіль, 2023. – С. 135-137.

1 АНАЛІЗ ЗАХИСТУ ДАНИХ В СИСТЕМАХ ВІДЕОСПОСТЕРЕЖЕННЯ

1.1 Аналіз підходів до захисту даних в системах відеоспостереження

Ключ до безпечної розробки систем відеоспостереження з передачею потокового відео – це інтегрована платформа, яка забезпечує конфіденційність і наскрізне шифрування. Цю широку тему можна розглядати з різних точок зору. Деякі фахівці вважають, що платформа, захищена паролем, необхідна для будь-якого захищеного відео, оскільки саме так вони можуть найкраще запобігти доступу небажаних глядачів до вмісту. Інші використовують більш комплексний підхід, маючи справу з конкретними програмами потокового відео, оптимізуючи свою інфраструктуру за допомогою шифрування AES та інших заходів безпеки.

Для забезпечення безпечного потокового передавання та приватного обміну відео використовують ряд підходів [1, 4].

1. Доставка через HTTPS. Використання HTTPS запобігає нападам з боку «людини посередині». Ці атаки не є рідкістю у світі розробки програм для потокового онлайн-відео, особливо коли користувачі підключаються з громадських місць, таких як кафе, бібліотеки та університети. Інформація, що надсилається через ці загальнодоступні мережі, вразлива до крадіжки хакерами.

Підключення користувача до веб-сайту приховано за допомогою шифрування HLS доставки HTTPS із використанням цифрових сертифікатів і ключів шифрування, що унеможливлює зловмисникам здійснити такий злом.

Під час перегляду відео на сервері весь зв’язок між сервером і глядачем шифрується за допомогою протоколу HTTPS. Використання HTTPS для передачі – чудовий спосіб захистити відео під час перегляду в Інтернеті. Це додає додатковий рівень безпеки до всього, що ви намагаєтеся захистити.

2. Відео, захищене паролем. Програми потокового відео, захищені паролем, хоч і здаються простими, але є чудовим підходом до запобігання будь-кому доступу до вашого вмісту.

Як і слід було очікувати, захищене паролем відео повністю функціональне. Спочатку виберіть кілька відео, які потрібно захистити паролем. Після цього для перегляду потрібен пароль. Вони не можуть дивитися відео, якщо не знають пароля. Попередній перегляд відео, внутрішнє корпоративне використання, відгуки клієнтів тощо можуть отримати переваги від захисту паролем, простого, але ефективного підходу до безпеки.

Треба мати увазі, що це не остання лінія захисту. Паролі можуть бути розголошені на публічних форумах, особливо якщо ваш бренд великий. Ось чому багато мовників змінюють свої паролі, щоб зберегти зони обмеженого доступу в секреті. У поєднанні з іншими заходами безпеки потокове відео, захищене паролем, може бути ефективним підходом для захисту вашого вмісту.

3. Захищені центри відеоданих і CDN. Мережа доставки вмісту (CDN) часто працює в центрі обробки даних, де зберігаються ваші завантажені файли програми потокового відео під час використання онлайнової відеоплатформи.

CDN – це комп’ютерна система, яка використовує складне програмне забезпечення для балансування навантаження, щоб сприяти швидкому розповсюдженню різних типів носіїв у всьому світі. CDN для розробки програм потокового відео гарантує, що ваше відео відтворюватиметься без помітних затримок.

Використання CDN із платформою розробки додатків для потокового онлайн-відео підвищує безпеку та захищає від загроз. Сюди входять розподілені напади на відмову в обслуговуванні, які мають на меті перевантажити цільовий веб-сайт трафіком. Мережа доставки контенту (CDN) значно знижує ефективність цієї кіберзагрози.

Крім того, CDN захищають вас від апаратних збоїв. Вони мають вбудовану надлишковість, яка забезпечує безпеку вашого вмісту та швидке завантаження вашої сторінки.

4. Шифрування AES. Зловмисникам буде важче переглядати ваші відео, якщо ви використовуєте AES для їх захисту. Зрештою, хакери, які перехоплюють безпечні відеопотоки, захищені шифруванням AES, не зможуть переглядати вміст.

Простий у встановленні та непомітний для споживачів, цей підхід є ідеальним. Більшість додатків для потокового відео дозволяють активувати шифрування AES без необхідності навчання.

Будь-яка служба відеохостингу корпоративного рівня значно виграє від додаткового захисту, який забезпечує шифрування AES. Комплексне рішення безпеки вмісту можна створити, поєднавши цю технологію з іншими функціями безпечного потокового передавання.

5. Шифрування даних для безпечної передачі. Усі відеоканали, а також така інформація, як імена користувачів і паролі, повинні бути зашифровані, щоб захистити дані, що передаються, особливо якщо вони переходять через Інтернет. Доступно багато варіантів шифрування, але найпоширенішими є SSL/TLS для інформації про користувача та IPsec або MACsec для даних. Належне шифрування допомагає запобігти прослуховуванню та маніпулюванню пакетами, які можуть статися під час атаки MitM.

Походження даних (підтвердження джерела даних) і використання цифрових водяних знаків для забезпечення цілісності відеовмісту також можуть пом’якшити фальсифікацію даних. Інший підхід полягає в проактивному виявленні та запобіганні присутності перехоплювачів за допомогою такої функції, як Active Fiber Monitoring від Allied Telesis.

6. Використання надійних паролів та багаторівневий доступ. Надійні паролі повинні бути важливим елементом системи безпеки. Довжина, складність і регулярні зміни мають вирішальне значення для надійності

пароля. Це особливо важливо, якщо пристрій використовує переадресацію портів для доступу.

Для додаткового захисту облікових записів адміністраторів відмінним вибором є багатофакторна автентифікація. Пароль надійніший, оскільки користувач повинен надати додаткову унікальну інформацію, як-от SMS-код, і він отримує сповіщення про кожну спробу доступу.

Якщо кілька користувачів отримують доступ до відеоканалів, система повинна забезпечити різні рівні доступу, захищеного паролем. Деякі авторизовані користувачі можуть мати доступ до пристрою, тому вони можуть переглядати зображення лише з цих пристроїв, а інші можуть мати доступ на рівні оператора. Деякі можуть мати доступ до адміністратора або керувати всіма налаштуваннями, як-от створення нового облікового запису, зміна напрямку камери, додавання нових камер до мережі тощо.

7. Використовуйте системи виявлення та запобігання вторгненням. Як частина надійної стратегії кіберзахисту, антивірусне програмне забезпечення має бути встановлено на терміналах користувачів і цифрових відеореєстраторах (DVR) для виявлення та запобігання поширенню заражень шкідливим програмним забезпеченням. У нерозподілених топологіях фізично відкритого каналу (POC), де мережеві хости, такі як камери та відеореєстратори, мають загальнодоступні IP-адреси, система виявлення вторгнень у мережу (NIDS) може виявляти шкідливі або аномальні шаблони трафіку, які можуть вказувати на присутність хакера. Брандмауери VPN, такі як брандмауер наступного покоління UTM від Allied Telesis, можуть бути простим способом впровадження NIDS для блокування загроз і шифрування критичного мережевого трафіку.

8. Оновлення програмного забезпечення. Кожна система IP-відеоспостереження потребує час від часу оновлення програмного забезпечення для підтримки її безпеки. Оновлення мікропрограми можуть випускатися регулярно або час від часу як частина випуску виправлення пристрою для певної вразливості. Важливо зареєструвати пристрій на веб-

сайті виробника, щоб отримувати нагадування про всі ці оновлення, які слід завантажити та виконати негайно.

Процес оновлення мікропрограми може заважати роботі мережі, оскільки перезавантаження пристрою припиняє відеопотік, а оновлення багатьох пристроїв може зайняти багато часу та бути ризикованим у великих мережах. Тому важливо враховувати функції, які допомагають мінімізувати збої та автоматизувати процес оновлення.

Компанія Allied Telesis створила технологію Continuous PoE, яка підтримує живлення підключеного пристрою, наприклад камери, під час перезавантаження комутатора. Це мінімізує тривалість відключення та відновлює потік відео, не чекаючи перезапуску камери.

Autonomous Management Framework (AMF) – це рішення для автоматизації від Allied Telesis, яке спрощує встановлення та керування великомасштабними мережами. Серед його багатьох можливостей є автоматичне оновлення мікропрограми, яке може розгортати оновлення з мінімальними збоями та без ручного втручання.

9. Навчання користувачів методам безпеки. Як і з будь-якою іншою мережею чи пристроєм, люди є найслабшою ланкою в профілі безпеки IP-системи відеоспостереження. Тому вкрай важливо розробити та задокументувати рекомендації та політику з кібербезпеки, а також забезпечити навчання з кібербезпеки для всіх користувачів, які отримають доступ до системи. Користувачів слід поінформувати про потенційні вектори атак і про те, що їм потрібно робити, щоб захиститися від потенційних запитів зловмисників під фальшивими приводами. Вони також повинні знати про ризики доступу до системи, скажімо, через мобільний додаток, у незашифрованій публічній системі Wi-Fi.

Важливо бути в курсі останніх стандартів кібербезпеки та найкращих практик на організаційному рівні та гарантувати їх дотримання на всіх рівнях. Ми співпрацюємо з NUARI, щоб запропонувати найновішу освіту із захисту від кіберзагроз, адаптовану до конкретних вимог вашої організації.

Завдяки низькій вартості володіння, легкому розгортанню та численним розширеним функціям багато виробничих організацій і державних установ переходять від аналогових систем відеоспостереження до IP-систем відеоспостереження для додаткової безпеки та спокою. Тим не менш, оскільки ландшафт кіберзагроз стає все більш складним, а кібер-зловмисники все більш невблаганними, організації повинні знати про ризики таких систем, особливо якщо вони знаходяться в тій же мережі, що й критично важливі для бізнесу дані та програми. Ігнорування їх може бути небезпечним і потенційно руйнівним, тому закриття циклу безпеки мережі має бути найвищим пріоритетом.

1.2 Шифрування в системах відеоспостереження

Фахівці з безпеки зазвичай використовують різні методи шифрування для захисту від різних загроз безпеці. Зокрема за допомогою шифрування можуть бути досягнуті конфіденційність, цілісність і автентифікація. Загалом шифрування можна розділити на дві основні категорії: симетричне та асиметричне. Кожна категорія містить різні алгоритми і може використовуватися для вирішення різних проблем [5, 6].

Як правило, симетричне шифрування використовується, коли продуктивність є проблемою, тому шифрування та дешифрування повинні проводитися швидко. Однак через те, що між сторонами, що спілкуються, потрібен спільний ключ, обмін ключами є загальною проблемою [7].

Найпопулярнішим алгоритмом симетричного шифрування є Advanced Encryption Standard (AES). Асиметричне шифрування, зазвичай використовується там, де не потрібне швидке виконання шифрування та дешифрування. Крім того, асиметричне шифрування не вимагає від об’єктів, що обмінюються повідомленнями, мати спільний секрет. Кожному потрібна пара ключів: один зберігається в секреті, а інший можна безпечно

оприлюднити. Пара ключів математично пов'язана так, що шифрування може виконуватись одним, а розшифровувати іншим. Для забезпечення конфіденційності повідомлення шифрується відкритим ключем одержувача, щоб ніхто, крім одержувача, не міг розшифрувати повідомлення. Щоб досягти автентифікації, відправнику потрібно зашифрувати повідомлення за допомогою власного закритого ключа, щоб одержувач розшифрував його за допомогою відкритого ключа відправника. Найпопулярнішим асиметричним алгоритмом шифрування є RSA. Безпека алгоритму RSA випливає з того факту, що його зламати обчислювально складно. В основному, складність зламу RSA залежить від добре встановленої проблеми факторизації. Починаючи з двох дуже великих простих чисел p і q, обчислюють результат їх множення $n = p * q$. Вважається, що важко знайти p і q знаючи n. Алгоритм RSA добре відомий у літературі. Він використовується в багатьох програмах, включаючи цифрові сертифікати.

Незважаючи на те, що довжина ключа в 1024 біти може бути достатньою, однак рекомендується використовувати набагато сильніший ключ довжиною 2048-бітний [8].

Методи симетричного та асиметричного шифрування є ортогональними. Наприклад, асиметричне шифрування можна використовувати для вирішення проблеми обміну ключами, від якої страждають методи симетричного шифрування. Після цього симетричне шифрування можна використовувати в довготривалому шифруванні, де шифрування та дешифрування виконуються швидко. AES схвалено для шифрування надсекретних документів для уряду США. Крім того, він використовується багатьма програмами, включаючи шифрування WPA2 WiFi. В AES можна використовувати ключі різної довжини: 128, 192 або 256 біт. Достатнім вважається ключ довжиною 128 біт.

Часто сторони, які спілкуються, повинні бути впевнені, що повідомлення не було підроблено під час передачі. Хеш-значення (або дайджест повідомлення) обчислюється та додається до повідомлення як

ознака його оригінальності. Окрім забезпечення цілісності повідомлення, одержувач також може бути зацікавлений у впевненості в автентичності повідомлення. Для досягнення цієї мети існує шифрування коду автентифікації повідомлень на основі хешування (HMAC). Одним із найпопулярніших варіантів тут є HMAC-MD5, який є ключем MD5, де спільний ключ бере участь у процесі хешування. HMAC-MD5 генерує 128-бітні хеші. Іншим варіантом є SHA-1, який генерує 160-бітні хеші [9–11].

На даний момент також існує ряд методів скремблування зображення які можна використовувати для скремблювання відеокадрів; однак їх потрібно зробити легшими, щоб вони вписувалися в середовище з обмеженими ресурсами, як-от камери Edge. Крім того, вони повинні досягти хорошого та оптимізованого балансу між конфіденційністю, чіткістю, оборотністю, безпекою та надійністю. Шифрування є найбезпечнішим представником класу редагування схем захисту конфіденційності відео. Шифрування захищає особисту інформацію та конфіденційні дані від несанкціонованого доступу та підвищує безпеку зв’язку між двома сторонами, які обмінюються інформацією. Однак, враховуючи природу відео в реальному часі, традиційні криптографічні механізми з симетричним ключем, такі як AES, не забезпечують захист від атак на цілісність та підміну зображення [12].

Механізми хаотичного шифрування пропонують кращі рішення, оскільки вони мають підвищену продуктивність, високий ступінь чутливості до незначних змін початкових умов, великий ступінь випадковості, величезний простір ключів, аперіодичність і високий рівень безпеки. Нещодавно запропоновані механізми шифрування хаотичного зображення, такі які^ є безпечними, але їх повільніше використовувати в Edge мережі. Крім того, схема захисту конфіденційності не захищає та не анонімізує конфіденційні атрибути, крім вікон, під час передачі. Застосований метод маскування призначений лише для розмиття певної області кадру та не може використовуватися для денатурації повного кадру з двох причин. По-перше,

він чудово працює з обчисленнями для маскування невеликих областей на кадрі, але його повільніше використовувати для повнокадрових програм. Використання обмеженої кількості параметрів керування з дуже великими числовими значеннями вплинуло на швидкість обчислень. По-друге, простір ключів менший і становить лише 128 біт [13].

1.3 Протокол передачі потокового відео

Протокол реального часу (RTP) широко використовується для потокового передавання медіа даних. RTP визначено робочою групою з розробки Інтернету (IETF) у RFC 3550 [14]. Потоками RTP можна маніпулювати та перехоплювати їх за допомогою різних атак, які можуть включати дані потоку. RTP може бути захищений IPSec. Однак обмеження, пов’язані з IPSec, вимагають більш універсального та масштабованого рішення. Це рішення також має зменшити динамічний розподіл сеансів, потребу в інфраструктурі відкритих ключів (PKI) і проблему проходження трансляції мережевих адрес (NAT). Таким чином бус розроблений безпечний протокол реального часу (SRTP) [6]. Для використання SRTP потрібен механізм обміну криптографічними ключами перед передачею відео. Таким чином, такі протоколи, як SDescriptions і Multimedia Internet KEYing (MIKEY), які використовуються для керування ключами, необхідні для механізмів керування ключами та для збереження безпеки мультимедійного сеансу. SRTP визначено в IETF RFC 3711 і опубліковано в 2004 році. Він забезпечує конфіденційність, цілісність і автентифікацію для потоків відео. SRTP було створено для забезпечення захисту відео додатків у режимі реального часу, таких як голосові та відеопрограми. Він підтримує як бездротові, так і дротові мережі з обмеженою пропускною здатністю, і доведено, що він є зручним методом захистом для неоднорідного дротового та бездротового мережевого середовища.

SRTP може досягти мінімального розширення пакетів і високої пропускної здатності. Він використовується для забезпечення безпеки для потоків RTP і RTCP. Ця структура забезпечує пакети RTP і RTCP механізмами автентифікації та шифрування. Для автентифікації повідомлень до зашифрованого потокового шифру додається функція на основі хешу-ключа (SHA-1) [15].

SRTP підтримує низьку вартість обчислень і низьку пропускну здатність для криптографічного перетворення. Його також можна адаптувати в програмах, які мають невеликий розмір через невеликий код реалізації. SRTP не залежить від фізичного, мережевого та транспортного рівнів, він вразливий до втрати пакетів і зміни порядку. Такі властивості спрощують реалізацію цього протоколу у вбудованих системах з обмеженими ресурсами процесора та пам’яті.

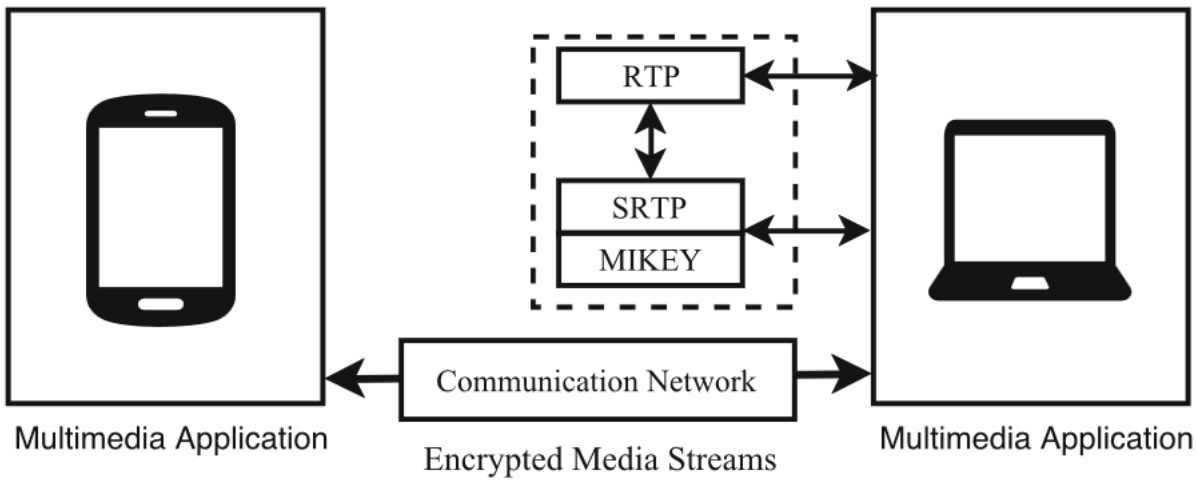


Рисунок 1.1 – Кодування в SRTP

Перш ніж надсилати пакети RTP через мережу, їх потрібно спочатку перетворити на пакети SRTP, а потім ці пакети потрібно розшифрувати в пункті призначення. Цей процес пояснюється на рисунку 1.1. Вхідні дані (наприклад, відео) надаються до програми з певних пристроїв, таких як камери. Потім цей сигнал кодується за допомогою спеціальної системи кодування, такої як Motion-JPEG (MJPEG) або H.264. Отримавши це, програма потім створює корисне навантаження даних RTP.

Корисне навантаження RTP потім шифрується за допомогою Advanced Encryption Standard (AES) у режимі лічильника з довжиною ключа 128 біт. Оскільки AES у режимі лічильника не потребує розширення для пакетів під час шифрування, це стандарт шифрування, який використовується за умовчанням для протоколу SRTP. AES розділяє дані на блоки, потім ці блоки даних обробляються непослідовним способом, отже, блоки даних можуть оброблятися паралельно. Якщо зашифровані пакети даних отримані не в порядку, кожен блок обробляється окремо, тому це не вплине на процес дешифрування. Програма SRTP перетворює пакети RTP на SRTP перед надсиланням через мережу, а пакети SRTP розшифровуються для перетворення на пакети RTP.

SRTP використовує додатковий тег під назвою MKI. Цей тег використовується для ідентифікації головного ключа, який використовується для отримання ключів сеансу, які використовуються для шифрування та/або автентифікації поточних пакетів. Типовий MKI має довжину 4 байти.

1.4 Конфіденційність у системах відеоспостереження

В даний час уряди, компанії та приватні особи широко розгорнули камери відеоспостереження в багатьох частинах міст, головною метою яких є забезпечення фізичної безпеки. Наприклад, підраховано, що пересічний лондонець потрапляє на камери відеоспостереження 300 разів на день під час пересування містом [5]. У результаті камери відеоспостереження збирають велику кількість інформації про людей без їх відома та згоди. Таким чином, найкращим рішенням для захисту конфіденційності людей на тлі збільшення кількості стаціонарних камер і безпілотників є вимога до компаній, що займаються камерою, розробити свої камери з урахуванням конфіденційності [16–18].

Виходячи з ретельних досліджень і спостережень, порушення конфіденційності в практиці систем відеоспостереження можна пояснити двома факторами, а саме перехопленням відео під час транзиту, зловживанням камер відеоспостереження та відео операторами, їх невибірковим характером, і проблема балансу між конфіденційністю та зручністю використання. Більшість існуючих систем спостереження розгортаються на основі архітектури хмарних обчислень. Як наслідок, відео можуть бути перехоплені під час передачі через мережу, використовуючи деякі з її вразливостей, що може спричинити розлив приватних даних у кіберпростір. По-друге, особи, відповідальні за камери відеоспостереження, могли незаконно збирати приватні дані. Додаткові зловживання камерами відеоспостереження, визначені Американським союзом громадянських свобод (ACLU), включають кримінальне зловживання, інституційне зловживання, зловживання в особистих цілях, дискримінаційне націлювання. Проте жоден із цих документів не пропонує універсально життєздатного рішення, окрім визначення слабких місць конфіденційності системи спостереження [19].

Атрибути конфіденційності особи. Делікатні речі, для яких люди вимагають захисту, відомі як особисті атрибути конфіденційності або загалом конфіденційні атрибути. Це інформація на відеокадрах і зображеннях, захоплених і переданих системами відеоспостереження, які можуть розкрити значну кількість особистої інформації та діяльності про особу, модель поведінки та щоденні дії осіб [8]. Такі атрибути, як риси обличчя, виявляють значні та потужні деталі, життєво важливі для ідентифікації людей. Крім того, передача кадрів, що містять деякі дуже особисті дії або ситуації, як є, може спричинити багато соціального збентеження у разі перехоплення. Люди також нервують, думаючи, що їх знімуть камери під час відвідування таких місць, як медичні центри. Крім того, люди не хочуть, щоб за ними спостерігали через такі отвори, як вікна, перебуваючи вдома. Ці роботи намагаються визначити та ідентифікувати

конфіденційні об’єкти, але вони не мають механізмів збереження конфіденційності. Наприклад, система рекомендацій щодо конфіденційності на основі машинного навчання для обміну зображеннями в соціальних мережах була запропонована в [8]. Однак він не дає інформації про те, як конфіденційність зображень користувачів соціальних мереж захищена під час надсилання на хмарний сервер для рекомендації.

У даній роботі основна увага буде приділена захисту конфіденційної інформації від перехоплення та розповсюдження в ширший простір під час передачі зображень, знятих камерами спостереження, щоб запобігти візуалізації та забезпечити анонімність. На сьогоднішній день не існує надійно розроблених схем протидії неправильному використанню камер відеоспостереження, окрім того, що це давня проблема конфіденційності.

1.5 Технології побудови систем відеоспостереження

Архітектура служби відеоспостереження відіграє важливу роль у процесі створення системи, яка забезпечує конфіденційність. Як наслідок, розробляють та досліджують системи з архітектурою на основі краю (Edge), туману (Fog) та хмари (Cloud). У хмарному середовищі практично не застосовуються заходи щодо збереження конфіденційності, тому що основною роботою периферійних камер є створення відео та пересилання його через глобальну мережу Інтернет до віддалених аналітичних центрів і центрів зберігання [20, 21]. Усі відеокадри, незалежно від того, цікаві вони працівникам служби безпеки чи ні, пересилаються на хмарні сервери, що створює непотрібне навантаження на мережу. Порівняно з іншими налаштуваннями, хмарна архітектура створює найбільший вплив на мережу. Крім того, існує високий попит на сховище, оскільки відеопотоки з багатьох периферійних камер надсилаються до хмарного центру. Архітектура туману часто відноситься до приватних мереж кампусу (CAN); і, як наслідок, лише

внутрішні користувачі можуть мати до нього доступ. Архітектури Fog мають відносно кращу конфіденційність, ніж налаштування хмари; однак це обмежено. У випадку Fog трафік ізольовано до корпоративної мережі, а відстань передачі скорочується; отже, вплив на пропускну здатність, затримку відповіді та вимоги до пам'яті менші.

Архітектура відеоспостереження, заснована на периферії, відрізняється від архітектури на основі хмар і туману тим, що відео не передається на вищі рівні. Скоріше вся відеоаналітика виконується інтелектуально на краю (Edge) мережі, і лише сигнали попередження надсилаються електронною поштою або пересилаються тексти служби коротких повідомлень (SMS) за допомогою модемів Глобальної системи мобільного зв'язку (GSM) авторизованим сторонам. У крайовому налаштуванні конфіденційність може бути повністю забезпечена, оскільки жодне відео не передається та не переглядається персоналом. Крім того, це не впливає на пропускну здатність і обмежені вимоги до пам'яті. Однак крайові підходи страждають від серйозного зниження продуктивності. Наразі методи класифікації та виявлення легких об'єктів на основі DNN не можуть обробляти більше 4 кадрів/с на периферійному пристрої, чиї обчислювальні можливості еквівалентні останньому випуску серії Raspberry PI. Крім того, частина відеозаписів потрібна як доказ проти злочинців у судах, але це налаштування не зберігає такі відео для подальшого використання [22].

1.5.1 Технологія Інтернет-речей

IoT – це технологія, яка спрямована на забезпечення інфраструктури для програм, які можуть координувати взаємодію людей, речей і систем для певної мети [23, 24].

Ці програми не обов'язково мають загальноприйнятий стандарт, але архітектурна модель системи IoT зазвичай складається з трьох основних рівнів: рівня сприйняття, який використовує датчики та мікроконтролери для сприйняття фізичного середовища; комунікаційний або мережевий рівень,

який обробляє та транспортує дані; і прикладний рівень, який використовує дані для надання користувачам специфічних для програми послуг [1]. Комунікаційний/мережевий рівень використовує різні технології для упаковки та передачі даних через обмеження обробки та пропускної здатності багатьох пристроїв IoT. Протокол передачі гіпертексту (HTTP), який зазвичай використовується для зв'язку між пристроями в Інтернеті, не підходить для малопотужних пристроїв IoT через його повністю орієнтовану на підключення архітектуру, великий розмір заголовка та затримку [6, 7]. Встановлені протоколи зв'язку, які більше підходять для цих обмежених потужності та пропускної спроможності вимог IoT, включають протокол обмежених додатків (CoAP), телеметричний транспорт із чергою повідомлень (MQTT) і протокол розширеного обміну повідомленнями та присутності (XMPP). У [8] оцінили та порівняли продуктивність цих протоколів у реальному тестовому стенді IoT. Для визначення різниці затримок розглядалися такі показники, як час створення пакета та швидкість доставки пакета. Дослідження показало, що XMPP мав найгіршу продуктивність через використання формату Extensible Markup Language (XML), який збільшив затримку. MQTT і CoAP мали однакову загальну продуктивність щодо створення пакетів і часу передачі, але MQTT виявився більш оптимізованим і стандартизованим. На додаток до цих протоколів бездротові технології, такі як Low Range (LoRa), Low Range Wide Area Network (LoRaWAN) і Low-Power Wide Area Network (LPWAN), можна використовувати для забезпечення дальнього зв'язку та зв'язку з низьким енергоспоживанням для IoT. приладів [9]. Ці бездротові протоколи розроблені для забезпечення малопотужних глобальних мереж, що робить їх ідеальними для тих випадків, коли пристроям потрібно передавати невеликі обсяги даних на великі відстані, наприклад, у сільському господарстві, розумних містах і промислових додатках [10]. В [11] провели більш поглиблене порівняння між CoAP і MQTT, щоб визначити, який є найбільш придатним для використання з обмеженими пристроями, зокрема датчиками.

Порівняння враховувало затримку зв’язку та мережевий трафік. Обидва протоколи виявилися хорошим вибором для пристроїв з обмеженими ресурсами, з однаковою продуктивністю та часом відгуку. Проте вибір найбільш підходящого протоколу залежав від загальних вимог системи. Було визнано, що CoAP є оптимальним вибором для взаємодії з бізнес-системами завдяки малим середнім розмірам пакетів і мінімальному використанню акумулятора/даних. З іншого боку, було встановлено, що MQTT є кращим рішенням для таких систем, як домашня автоматизація та сенсорні мережі, де неоднорідність пристроїв більш виражена. MQTT було простіше використовувати для нових пристроїв і мав найефективніший потік даних завдяки моделі публікації/підписки та використанню якості обслуговування (QoS).

1.5.2 Граничні обчислення

Граничні обчислення – це мережева архітектура, яка передбачає обробку сенсорних (наприклад, візуальних даних) даних ближче до джерела, а не в хмарі [11]. Це забезпечує швидку обробку та ефективну обробку операцій із інтенсивним використанням даних у реальних сценаріях, таких як IoT. Хоча периферійні обчислення можуть запропонувати переваги для систем IoT, існують також обмеження з точки зору безпеки. Дослідники виявили, що необхідний подальший розвиток у таких сферах, як автентифікація та контроль доступу, і що захищені від втручання архітектури можуть бути одним із рішень для вирішення цих проблем безпеки. Однак захист великомасштабних і критичних за часом систем IoT також може бути складним через вартість таких методів, як шифрування з точки зору затримки, споживання енергії та пропускної здатності мережі [12]. Крім того, неоднорідність пристроїв, які спілкуються через ці мережі без добре налагодженого протоколу, також може створювати проблеми [13]. Дослідники в цій галузі працюють над подоланням цих обмежень і підвищенням безпеки та ефективності зв’язку між пристроями IoT.

1.5.3 Технологія FPGA

FPGA – це спеціалізовані апаратні пристрої, які набули популярності в периферійних обчислювальних задачах завдяки своїй здатності вирішувати проблеми за допомогою реконфігурованих апаратних схем. Ці схеми можна описати за допомогою мов опису обладнання (HDL), таких як Verilog і дуже високошвидкісної мови опису обладнання інтегрованої схеми (VHDL), і складаються з різних логічних блоків, таких як таблиці пошуку, тригери та мультиплексори. FPGA пропонують кілька переваг щодо безпеки, паралельних обчислень і гнучкості для оновлення конструкції апаратного забезпечення після розгортання. Вони також були вдосконалені завдяки використанню системи на кристалі (SoC), яка інтегрує програмовану логіку з процесорами реального часу. Прикладом цього є AMD-Xilinx+ MPSoC1, який включає центральний процесор зі скороченим набором інструкцій (ARM), програмовану логіку та блоки для обробки графіки та відео. Хоча FPGA та SoC мають схожість з мікроконтролерами, FPGA пропонують переваги у фізичній та кібербезпеці завдяки зашифрованим бітовим потокам і механізмам завантаження ключів, а також можуть діяти як корінь довіри (RoT), зберігаючи приватні ключі безпеки та важливі алгоритми. ПЛІС також показують більшу ефективність в алгоритмах обробки зображень і перекодування відео завдяки своїм можливостям паралельних обчислень [25].

Незважаючи на те, що FPGA пропонують значні переваги для IoT, вони вважаються складними через низькі знання апаратного забезпечення, наприклад VHDL і Verilog. Щоб вирішити цю проблему, постачальники FPGA пропагують використання високорівневих потоків проектування та інструментів, які дозволяють створювати проекти рівня передачі реєстру (RTL) з використанням мов високого рівня, таких як C, C++, System C і Open Computer Language (OpenCL). Однак залишається відкритим питання про те, наскільки добре ці проекти високого рівня порівнюються з написаними вручну проектами RTL з точки зору оптимізації. Хоча високорівневий синтез

(HLS) може бути не настільки оптимізованим, як написані вручну дизайни RTL для складних проектів, використання директив, таких як розгортання циклу, злиття циклу та конвеєрування, може значно покращити використання ресурсів, зменшити затримку, збільшити спільне використання ресурсів і оптимізація логіки для алгоритмів обробки відео. Ці висновки свідчать про те, що технологія FPGA може бути більш доступною для розробників без серйозних апаратних знань низького рівня, зберігаючи хорошу продуктивність.

Комунікаційні протоколи, такі як MQTT і CoAP, показали свою ефективність у середовищах з обмеженими ресурсами, але вибір між ними в кінцевому рахунку залежить від конкретних вимог системи. Граничні обчислення мають потенціал для підвищення ефективності та безпеки мереж IoT, але також мають свої обмеження, які потребують подальшого розвитку. Технологія FPGA пропонує розширену безпеку та можливості паралельної обробки для IoT, але може бути складною для впровадження. Інструменти синтезу високого рівня, такі як AMD-Xilinx-Vivado, показали, що покращують продуктивність проектів FPGA для програм обробки зображень у реальному часі, але не завжди можуть бути настільки оптимізованими, як проекти, написані вручну. Ці результати підкреслюють важливість ретельної оцінки різних технологій і підходів, доступних для конкретної системи IoT, щоб забезпечити оптимальну продуктивність і безпеку для систем відеоспостереження.

1.5.4 Атаки на системи відеоспостереження

Деякі загрози безпеці даних можуть поставити під загрозу систему відеоспостереження. Як наслідок, під час процесу потокового відео має бути гарантовано кілька заходів безпеки. Основні загрози [20]:

1. Конфіденційність. Нам потрібно, щоб відеопотоки могли використовуватися лише частинами системи. Ми не повинні дозволяти будь-

якій третій стороні переглядати те, що контролює система, або надсилати неавтентичні відеопотоки.

2. Цілісність. Якщо це викликає занепокоєння, то додаткові метадані (інформацію про дані) слід надіслати разом із самими даними. Можна використовувати контрольні суми, CRC-коди, коди автентифікації повідомлень. Водяні знаки (знаки, що забезпечують справжність і цілісність фотографій) або використовувати цифровий підпис.

3. Доступність. Якщо контрольоване місце може бути захоплене зловмисними камерами або шкідливі пристрої можуть перешкоджати камерам надсилати свій трафік, тоді периметр в цій зоні слід ретельно контролювати, щоб уникнути зловмисних дій.

2 ТЕОРЕТИЧНІ ОСНОВИ ШИФРУВАННЯ ЗОБРАЖЕНЬ

2.1 Криптографічні хеш-функції

Криптографічні хеш-функції є важливим інструментом у криптографії для таких програм, як цифровий відбиток повідомлень, автентифікація повідомлень і отримання ключів. Протягом останніх років було запропоновано кілька швидких хеш-функцій програмного забезпечення; більшість із них базуються на принципах дизайну MD4 Рона Ріввеста. Однією з таких пропозицій був RIPEMD, який був розроблений в рамках проекту ЄС RIPE (Race Integrity Primitives Evaluation). Є три вагомі причини розглянути таку заміну [27]:

1. 128-бітний результат хешування більше не забезпечує достатнього захисту. Атака пошуку зіткнення грубою силою на 128-бітному хеш-результаті вимагає 264 або приблизно 2,1019 оцінок функції. У 1994 році Пол ван Оорсхот і Майк Вінер показали, що цю роботу грубої сили можна виконати менш ніж за місяць із інвестиціями в 10 мільйонів доларів США. Очікується, що ці витрати будуть зменшуватися вдвічі кожні 18 місяців.

2. У першій половині 1995 року Ганс Доббертін виявив зіткнення для версії RIPEMD, обмеженої двома раундами з трьох. Використовуючи подібні методи, Ханс створив восени 1995 року зіткнення для (всіх 3 раундів) MD4. Атака на MD4 вимагає лише кількох секунд на комп'ютері та залишає певну свободу щодо вибору повідомлення, чітко виключаючи MD4 як хеш-функцію, стійку до зіткнень. Незабаром після цього, навесні 1996 року, Ганс також виявив зіткнення для функції стиснення MD5. Хоча ця атака ще не поширена на колізії для самої MD5, ця атака ставить під серйозні сумніви міцність MD5 як хеш-функції, стійкої до зіткнень. RSA Data Security, для якої Рон Рівест розробив MD4 і MD5, рекомендує більше не використовувати MD4 і не використовувати MD5 для майбутніх програм, які вимагають, щоб хеш-функція була стійкою до зіткнень.

3. Під час основної сесії Crypto 2004 було оголошено, що Сяоюнь Ван, Денгго Фенг, Сюецзя Лай і Хунбо Ю знайшли конфлікти для MD4, MD5, RIPEMD і 128-бітної версії HAVAL.

2.1.1 Хеш функція RIPEMD-160

Розмір хеш-результату та змінної ланцюжка для RIPEMD-160 збільшено до 160 біт (п'ять 32-бітних слів), кількість раундів збільшено з трьох до п'яти, а два рядки стали більш різними (не лише змінюються константи, а також булеві функції та порядок слів повідомлення).

Алгоритм обчислення хеш функція RIPEMD-160 складається з наступних кроків [28]:

1. Операції в один крок. $A := (A + f(B, C, D) + X + K)^{<<s} + E$ і

$C := C^{<<10}$. Тут <<s позначає циклічний зсув (обертання) на s позицій

2. Порядок слів повідомлення. Перестановка ρ здійснюється згідно таблиці:

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ρ(i)	7	4	13	1	10	6	15	3	12	0	9	5	2	14	11	8

Далі визначаємо перестановку π, встановивши π(i) = 9i + 5 (mod 16). Порядок слів у повідомленні наведено в наступній таблиці:

Line	Round 1	Round 2	Round 3	Round 4	Round 5
left	id	ρ	ρ^2	ρ^3	ρ^4
right	π	$\rho\pi$	$\rho^2\pi$	$\rho^3\pi$	$\rho^4\pi$

3. Булеві функції. Обчислюються наступні булеві функції

$$f_1(x, y, z) = x \oplus y \oplus z,$$
$$f_2(x, y, z) = (x \wedge y) \vee (\neg x \wedge z),$$
$$f_3(x, y, z) = (x \vee \neg y) \oplus z,$$
$$f_4(x, y, z) = (x \wedge z) \vee (y \wedge \neg z),$$
$$f_5(x, y, z) = x \oplus (y \vee \neg z).$$

Ці булеві функції застосовуються таким чином:

Line	Round 1	Round 2	Round 3	Round 4	Round 5
left	f_1	f_2	f_3	f_4	f_5
right	f_5	f_4	f_3	f_2	f_1

4. Зсув. Для обох рядків виконаємо такі зміни:

Round	X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
1	11	14	15	12	5	8	7	9	11	13	14	15	6	7	9	8
2	12	13	11	15	6	9	9	7	12	15	11	13	7	8	7	7
3	13	15	14	11	7	7	6	8	13	14	13	12	5	5	6	9
4	14	11	12	14	8	6	5	5	15	12	15	14	9	9	8	6
5	15	12	13	13	9	5	8	6	14	11	12	11	8	6	5	5

5. Константи. Беремо цілі частини наступних чисел:

Line	Round 1	Round 2	Round 3	Round 4	Round 5
left	0	$2^{30} \cdot \sqrt{2}$	$2^{30} \cdot \sqrt{3}$	$2^{30} \cdot \sqrt{5}$	$2^{30} \cdot \sqrt{7}$
right	$2^{30} \cdot \sqrt[3]{2}$	$2^{30} \cdot \sqrt[3]{3}$	$2^{30} \cdot \sqrt[3]{5}$	$2^{30} \cdot \sqrt[3]{7}$	0

RIPEMD-160 – це вдосконалена версія RIPEMD із 160-бітним результатом хешування, і очікується, що вона буде безпечною протягом наступних десяти років або більше. Філософія дизайну полягає в тому, щоб якомога більше спиратися на досвід, отриманий під час оцінювання MD4, MD5 і RIPEMD. Як і його попередники, RIPEMD-160 налаштований на 32-розрядні процесори, які, як ми вважаємо, залишатимуться важливими в наступному десятилітті.

RIPEMD-128 – це плагін, який замінює RIPEMD (або MD4 і MD5, якщо на те пішло) із 128-бітним результатом. З огляду на результат Пола ван Оорсхота та Майка Вінера, згаданий раніше, 128-бітні результати хешування не забезпечують достатнього захисту протягом наступних десяти років, і програми, які використовують 128-бітові хеш-функції, повинні розглянути можливість оновлення до 160-бітної хеш-функції.

RIPEMD-256 і RIPEMD-320 є додатковими розширеннями, відповідно, RIPEMD-128 і RIPEMD-160 і призначені для застосувань хеш-функцій, які

вимагають довший результат хешування без потреби у більш високому рівні безпеки.

Приклад обчислення хеш функції RIPEMD-160 з використанням openssl:

```
echo -n 'Taras Pastukh' | openssl ripemd160
    3338608f680d8c1049f8be5677d83857a98f3b40
```

2.2 Код автентифікації повідомлення на основі хешу

HMAC (код автентифікації повідомлення на основі хешу) підтримує використання ключа для хешування даних. Цей ключ зберігається в секреті між Бобом і Алісою, і його можна використовувати для автентифікації як даних, так і того, що відправник усе ще знає секрет [25].

HMAC можна використовувати для перевірки цілісності та автентифікації повідомлення. Він передбачає хешування повідомлення за допомогою секретного ключа, і, таким чином, відрізняється від стандартного хешування, яке є виключно односторонньою функцією. Як і будь-який MAC, його можна використовувати зі стандартною хеш-функцією, такою як MD5 або SHA-1, що призводить до таких методів, як HMAC-MD5 або HMAC-SHA-1. Також, як і з будь-якою функцією хешування, міцність залежить від якості функції хешування та отриманої кількості бітів хеш-коду. Крім того, кількість бітів у секретному ключі є чинником міцності хешу. На рисунку 2.1 показано операцію, під час якої повідомлення, яке потрібно надіслати, перетворюється за допомогою секретного ключа та функції хешування на код HMAC. Потім цей код надсилається разом із повідомленням. При отриманні отримувач перераховує код HMAC з того самого секретного ключа, що й повідомлення, і звіряє його з отриманою версією. Якщо вони збігаються, перевіряються як відправник, так і повідомлення.

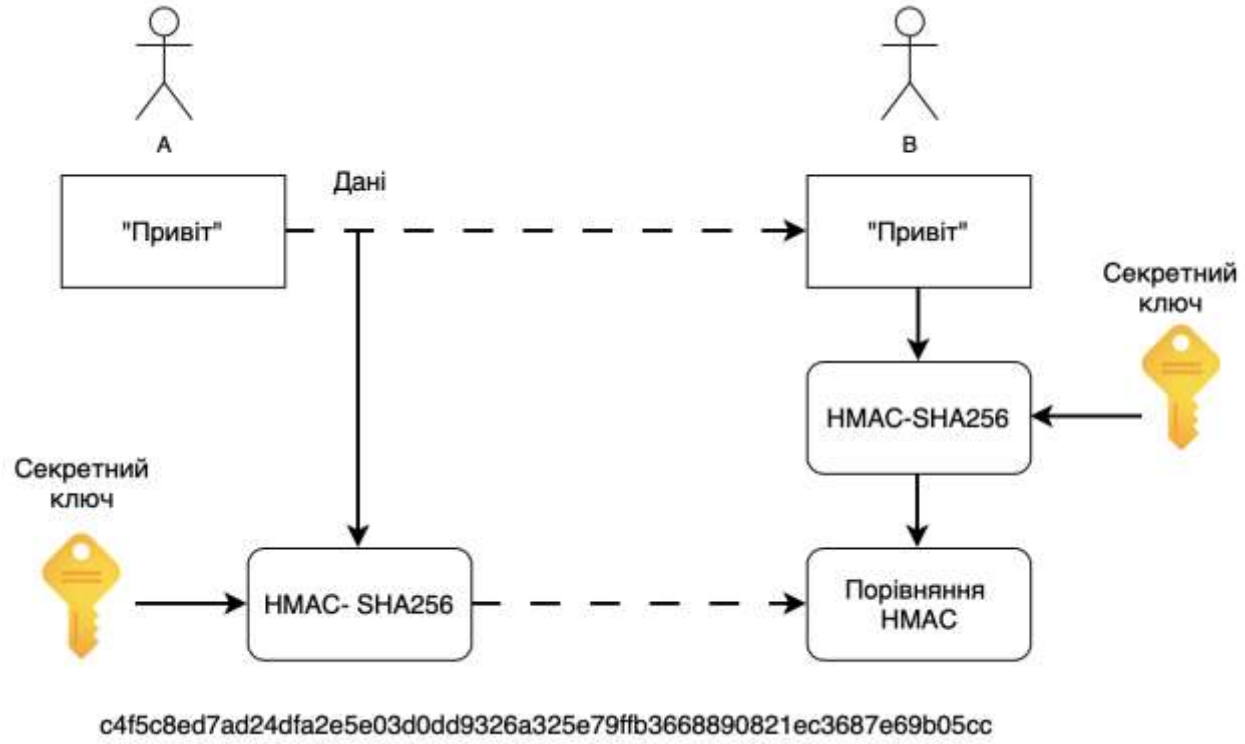


Рисунок 2.1 – Схема роботи коду автентифікації повідомлення на основі хешу

Повідомлення: Information Security Algorithms in Video Surveillance Systems
Ключ:
000102030405060708090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f
HMAC-Sha256:
c4f5c8ed7ad24dfa2e5e03d0dd9326a325e79ffb3668890821ec3687e69b05cc
HMAC-sha512:
cddc0686ac5888485a698d684fbcf2e6600b90efb253bc4ef6b1d3a644b52ad3c9d8b807c626bbaf7fb85b1c5e6e19ef5b519ccfe762d6661bcc5f6ddc0e6a41

Алгоритм HMAC. Спочатку вибирається алгоритм хешування. Залежно від алгоритму, дані будуть хешуватися в блоках певного розміру B і буде створено хеш розміру L. Тут важливо зазначити, що HMAC використовує алгоритми хешування, які є блоковими шифрами. Блокові шифри шифрують дані блоками. Наприклад, якщо у нас є потік із 160 біт, то блочний шифр може зашифрувати (або хешувати) дані у блоці з 8 біт. Навпаки, потоковий шифр шифрує дані побітово.

Внутрішній і зовнішній ключі. Тепер потрібно вивести з криптографічного ключа два ключі – внутрішній ключ і зовнішній ключ. Внутрішній ключ генерується шляхом додавання нулів до кінця ключа, щоб зробити його розміром B, а потім виконується операція XOR ключа за допомогою ipad, що становить 352 біти нулів до ключа, щоб зробити ключ розміром 512 бітів. Потім цей ключ обробляється XOR з байтом 0x36, що повторюється B разів.

Наприклад, якщо розмір блоку становить 64 байти (512 біт), а розмір ключа – 20 байт (160 біт), тоді додаємо 0x36, що повторюється 64 рази. Отриманий ключ є внутрішнім ключем.

Зовнішній ключ отримується шляхом додавання нулів до оригінального ключа, щоб зробити його розміром B, а потім виконується операція XOR його з opad, який 0x5C повторюється B разів. Тепер одержувач даних може використовувати дані, які він отримує, і ключ, який він має, і виконати наведений вище алгоритм і перевірити, чи відповідає отриманий хеш коду, який він отримав. Якщо вони збігаються, то вони доводять дві речі.

1. Вхідні дані та ключ однакові, оскільки одержувач отримав той самий вихід.

2. Оскільки ключ той самий і оскільки очікується, що ключ буде таємним між відправником і одержувачем, це підтверджує відправника даних.

Отже, алгоритм HMAC працює таким чином [28]:

K – Криптографічний ключ (за необхідності доповнюється нулями)

H(K XOR opad, H(K XOR ipad, дані)).

HMAC працює шляхом хешування конкатенації (||) ключа k1 і вхідного повідомлення, а потім хешування конкатенації ключа k2 з виходом першої операції. Ключі k1 і k2 є детермінованими похідними від секретного ключа k. Процес обчислення HMAC показано на рисунку 2.2.

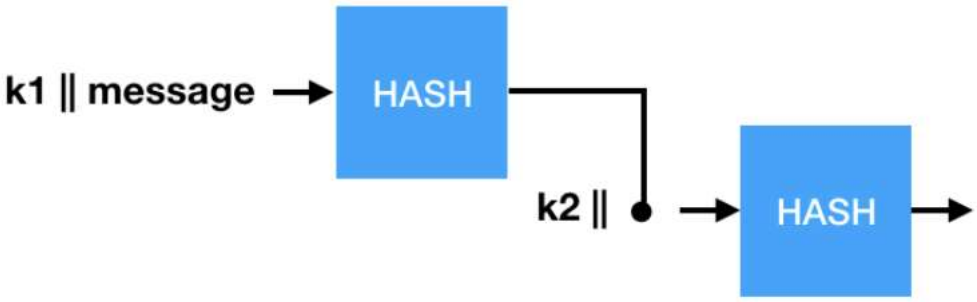


Рисунок 2.2 – Обчислення HMAC

Алгоритм обчислення HMAC виконує такі кроки:

1. Спочатку створюється два ключі з основного ключа: k1 = k ipad і k2 = k opad, де ipad (внутрішнє доповнення) і opad (зовнішнє доповнення) є константами та є символом для операції XOR.

2. Потім він об'єднує ключ k1 із повідомленням і хешує його.

3. Результат об'єднується з ключем k2 і хешується ще раз.

4. Це створює остаточний тег автентифікації.

Через це розмір тегу автентифікації залежить від використовуваної хеш-функції. HMAC-SHA256 використовує SHA-256 і створює тег автентифікації 256 біт. HMAC-SHA512 створює тег автентифікації 512 біт; і так далі.

Хоча вихід HMAC можна скоротити, щоб зменшити його розмір, тег автентифікації має бути щонайменше 128 біт. Це не завжди дотримується, і деякі програми опустяться до 64 бітів через явну обробку дуже обмеженої кількості запитів.

HMAC був побудований таким чином, щоб полегшити докази. У кількох документах доведено, що HMAC захищений від підробок, якщо хеш-функція під ним має деякі хороші властивості, які повинні мати всі криптографічно безпечні хеш-функції.

З цієї причини HMAC можна використовувати в поєднанні з великою кількістю хеш-функцій. Сьогодні HMAC в основному використовується з SHA-2.

2.3 Алгоритм цифрового підпису

З безперервним розвитком інформаційних технологій і обчислювальної потужності комп'ютера криптографія також розвивається. Оскільки мережеве середовище, особливо середовище Інтернету речей, є вразливим до атак, а цифрові зображення містять надлишкову інформацію, яка тісно пов'язана з особистою конфіденційністю. Багато алгоритмів працюють виключно з шифруванням цифрових зображень або шифруванням тексту відповідно. Проте, ряд алгоритмів, тим не менш, поєднують ці дві задачі [29].

До основних криптографічних примітивів, які широко використовуються для захисту конфіденційності, цілісності та доступності зображень, зокрема в системах відеоспостереження, належать: цифровий підпис, хеш функції та алгоритми симетричного та асиметричного шифрування. Розглянемо ці алгоритми детальніше.

Алгоритм цифрового підпису (Digital Signature Algorithm, DSA) – це криптосистема з відкритим ключем і федеральний стандарт обробки інформації для цифрових підписів. Він заснований на модульному зведенні в ступінь і проблемі дискретного логарифмування. DSA – це техніка підпису, подібна до схем підпису Шнорра та Ель-Гамала.

DSA був запропонований Національним інститутом стандартів і технологій (NIST) для включення до стандарту цифрового підпису (DSS) у 1991 році. Пізніше він був прийнятий як FIPS 186 у 1994 році. Фундаментальна специфікація зазнала чотирьох переглядів, з FIPS 186–4, випущений у липні 2013 року, будучи найновішим. Хоча DSA спочатку був запатентований, термін дії патенту закінчився, і NIST зробив алгоритм безоплатним у всьому світі. Відповідно до проекту FIPS 186–5 DSA більше не буде схвалено для створення цифрового підпису. Однак його можна використовувати для перевірки підписів, зроблених до дати набрання чинності стандартом.

Криптографія еліптичних кривих (ECC) спирається на алгебраїчну структуру еліптичних кривих над кінцевими полями. Використання еліптичних кривих у криптографії було запропоновано Нілом Кобліцем і Віктором С. Міллером незалежно один від одного в 1985 році; Алгоритми ECC стали широко використовуватися в 2004 році. Перевага алгоритму ECC перед RSA полягає в тому, що ключ може бути меншим, що призводить до покращеної швидкості та безпеки. Недолік полягає в тому, що не всі служби та програми сумісні з сертифікатами TLS/SSL на основі ECC (рисунок 2.3).

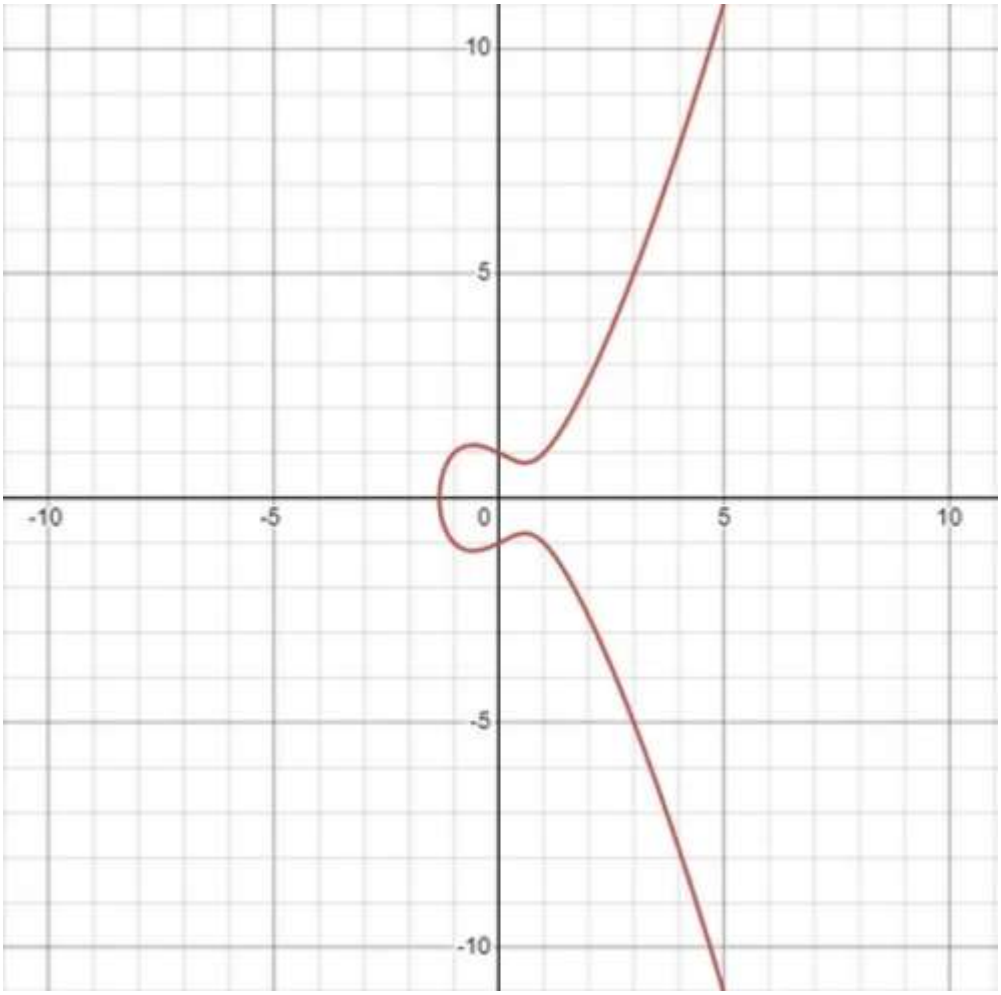


Рисунок 2.3 – Еліптична крива

Над полем K еліптична крива має точки, які можна визначити за допомогою координат (x, y). Якщо характеристика поля не дорівнює ні 2, ні 3, криву можна описати як плоску алгебраїчну криву за допомогою рівняння:

$$y^2 = x^3 + ax + b,$$

для конкретних коефіцієнтів a і b у K. Ця крива має бути несингулярною, тобто вона не повинна мати вершин або самоперетинів.

ECDSA (алгоритм цифрового підпису еліптичної кривої) описано в SEC-1. ECDSA виконує три основні функції: генерація пари ключів; підписання та перевірка (рисунок 2.4) [30].

Генерація пари ключів. Пара ключів ECDSA складається з:

Приватний ключ (ціле число): privKey.

Відкритий ключ (точка EC): pubKey = privKey * G.

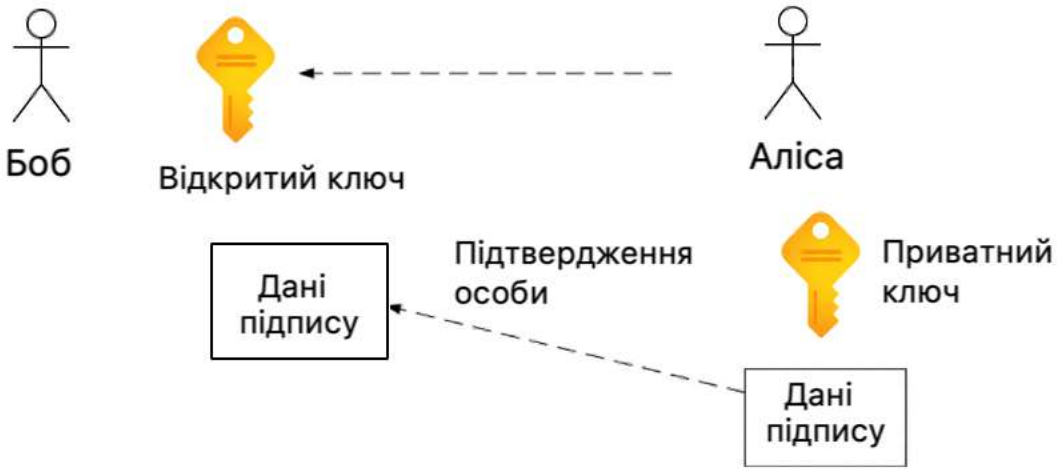


Рисунок 2.4 – Загальна схема цифрового підпису [30]

Приклад обчислення та перевірки цифрового підпису:

Повідомлення: Taras Pastukh

Хеш: sha256

Крива: secp256k1

Приватний ключ:

-----BEGIN PRIVATE KEY-----

MIGEAgEAMBAGByqGSM49AgEGBSuBBAAKBG0wawIBAQQg9uOh U8Qk3hqt313LiOD2

4uNbI0hMnfVVtvZBZpk1dJuhRANCAASdm0wZNjEdhVRAMfFVRS555
pk16LfaYw1H

cre1c8kKKR+mF29XQBHWXEP3BUuQqG4OeTajia4BFIIjUoRNviyM

-----END PRIVATE KEY-----

Публічний ключ:

-----BEGIN PUBLIC KEY-----

MFYwEAYHKoZIzj0CAQYFK4EEAAoDQgAEnZtMGTYxHYVUQDHx
VUUueeaZNei32mMN

R3K3tXPJCikfphdvV0AR1lxD9wVLkKhuDnk2o4muARSCI1KETb4sjA=
=

-----END PUBLIC KEY-----

Цифровий підпис:

304402207ceb48a56f7b53f4fc639ce480876ddc0af22c2f7078c06ad26db350
106f58db022067f778d018c7d9824ce0fa3d6ee41b512098021357b1a183d6d54a78
4fcfad57

Перевірка. Цифровий підпис правильний ?: так.

Приватний ключ – це випадкове ціле число від 0 до n-1. Відкритий ключ, pubKey, є точкою еліптичної кривої, отриманою шляхом множення закритого ключа з точкою генератора G. Точку EC відкритого ключа (x, y) можна додатково стиснути до однієї з її координат плюс біт парності.

Підписання. Алгоритм підпису ECDSA, як описано в RFC 6979, приймає повідомлення та закритий ключ як вхідні дані та створює підпис, що складається з двох цілих чисел, r і s.

Процес заснований на схемі підпису ElGamal і може бути побудований (з деякими спрощеннями) як:

r: координата x конкретної точки еліптичної кривої, отримана з випадкового числа та точки генератора G;

s: обчислюється з використанням хешу повідомлення, r, закритого ключа та випадкового ключа.

Перевірка. Процедура перевірки підпису отримує повідомлення та підпис (r, s) як вхідні дані та визначає дійсність підпису. Ось спрощений огляд перевірки:

Хеш: обчисліть хеш h повідомлення m, використовуючи той самий алгоритм хешування, який використовувався під час створення підпису, як правило, один із варіантів SHA.

Обернене число за модулем: обчисліть обернене s за модулем p як:

$$s^{-1}(mod\ p).$$

Випадкова точка: повторно обчисліть точку еліптичної кривої R, використовуючи відкритий ключ відправника.

Декомпресія: розпакуйте точку еліптичної кривої, щоб отримати її x-координату.

Порівняння: перевірте підпис, порівнявши конкретні отримані значення.

За допомогою шифрування з відкритим ключем ми створюємо пару ключів: відкритий та закритий ключ. Якщо Аліса надсилає дані Бобу, вона може додати свій цифровий підпис, який підтвердить, що вона є відправником, а також підтвердить, що дані не були змінені. Вона робить це, підписуючи дані своїм закритим ключем, а потім Боб може підтвердити підпис відкритим ключем Аліси [30].

Приклад обчислення цифрового підпису еліптичної кривої.

Повідомлення: Taras Pastukh

Приватний ключ Аліси:

key=185693916370532853213066052606190167377241167147806910216142107865551025046 77

Відкритий ключ Аліси:

key=(110337485666861667902730247049640181259642133065657628803460915214492340087475,

613876407189686778569977083867306060108486981528647607560707862409
71174611896);

k=
842645429295151126872551425703838317796927418104773474195581542533
46650999908;

r=706266393775660759908387814785092069783380530120976051515275
56728098932894322;

s=107431821602471952492501488313815921808704185857757477164067
0177701763312796348;

Результат
=706266393775660759908387814785092069783380530120976051515275672809
8932894322.

Підпис підтверджено!

=== Message ===

Msg=Taras Pastukh

Hash=b03908137e3e8fc8cbfe6eb4ca77a0910ac372ea6ab3f816bd729067146
bfa76

=== Private key ===

Private
key=e8c0253528717f72ecb475fd740a23235c6c683d209f4ad23e243cd0cfa6c54e

Curve=P-256k1

Bit size=256

Base point (G) = (55066263022277343669578718895168534326250603453777594175500187360389116729240,
32670510020758816978083085130507043184471273380659243275938904335757337482424)

Prime=115792089237316195423570985008687907853269984665640564039457584007908834671663,

Order=115792089237316195423570985008687907852837564279074904382605163141518161494337

=== Public key (X,Y) ===

X=84380129088822328747585547191496237175882446310576539435074989770355368004042

Y=42282126486060939822231249863467672746829986808875990951178992694443582458830

Hex:

X=ba8d75ea1ede9503c6d3b7751f3c1b170c72eb67915f17e36177671e10b409ca

Y=5d7ad5e4432d25dd74a4151dd6310c6856aa5ad973c3e058d93c3c1e3e248bce

=== Signature (R,S) ===

R=24889448698025559550427534887808130430855901146975891559947509846690694260022

S=22276930577598298856359014226020285819044661940845923915639679001560648278121

Hex:

R=3706edc1fab6f7ea8218ad3cafe75f0b72dfab4250102b7f9467b6c3f2096536

S=31404bbf2b2c302d9077d1ba44d0e96c4130d1f5f60df52eefe6a5fe33dc6069

Signature verifies

Підписи ECDSA схильні до атак, якщо їх не було реалізовано належним чином, тому необхідно визначити основні правила безпечної реалізації цифрового підпису ECDSA.

Перевага ECDSA полягає в тому, що не потрібно зберігати відкритий ключ, але можна перевірити підпис із хешованої версії закритого ключа. Таким чином блокчейну не потрібно було зберігати відкриті ключі тих, хто його використовував, і це був один із перших випадків, коли була створена справді децентралізована інформаційнуа інфраструктура.

Основні правила надійного використання цифрових підписів ECDSA [31].

1. Ніколи не розкривайте Nonce

Приклад: зламати ECDSA через витік nonce (SECP256k1). ECDSA з nonce.

За допомогою підпису ECDSA ми підписуємо повідомлення закритим ключем (priv) і підтверджуємо підпис відкритим ключем (pub). Потім випадкове значення (nonce) використовується для рандомізації підпису. Кожного разу, коли ми підписуємо, ми створюємо випадкове значення nonce, і воно створюватиме інший (але перевірений) підпис. Загалом підписувач має розкрити лише елементи підпису та їхній відкритий ключ, а не одноразове значення. Якщо підписувач помилково відкриває лише одне значення nonce, зловмисник може виявити закритий ключ. У цьому прикладі розкриємо значення nonce, визначимо приватний ключ і використаємо криву secp256k1 (яка використовується для біткойнів).

У ECDSA користувач створює випадковий закритий ключ (priv), а потім відкритий ключ (pub) із [31]:

$$pub = priv \times G$$

Далі, щоб створити підпис для повідомлення M, він створює випадкове число (k) і генерує підпис:

$$r = k \cdot G$$

$$s = k\text{^}\{-1\}(H(M) + r \cdot priv)$$

Тоді сигнатурою є (r, s), де r – координата x точки kG. $H(M)$ – хеш SHA-256 повідомлення (M), перетворений у ціле число. Якщо для будь-якого з підписів виявлено значення k, зловмисник може визначити закритий ключ за допомогою:

$$priv = r\text{^}\{-1\} \times ((k \cdot s) - H(M))$$

Це працює тому, що:

$$s \cdot k = H(M) + r \cdot priv,$$

і так

$$r \cdot priv = s \cdot k - H(M),$$

і для priv:

$$priv = r - 1(s \cdot k - H(M)).$$

2. Не використовуйте слабкі Nonces.

Розглянемо як можна розкрити приватний ключ ECDSA за допомогою слабких значень nonce.

За допомогою підпису ECDSA ми підписуємо повідомлення закритим ключем (priv) і підтверджуємо підпис відкритим ключем (pub). Потім випадкове значення (nonce) використовується для рандомізації підпису. Кожного разу, коли ми підписуємо, ми створюємо випадкове значення nonce, і воно створюватиме інший (але перевірений) підпис. Приватний ключ, однак, можна виявити, якщо користувач підпише два різних повідомлення тим самим nonce [31].

3. Не діліться секретами.

Розглянемо як можна розкрити закритий ключ з двох ключів і спільних нонсів (SECP256k1).

За допомогою підпису ECDSA ми підписуємо повідомлення закритим ключем (priv) і підтверджуємо підпис відкритим ключем (pub). Потім випадкове значення (nonce) використовується для рандомізації підпису. Кожного разу, коли ми підписуємо, ми створюємо випадкове значення nonce, і воно створюватиме інший (але перевірений) підпис. Проте закритий ключ можна виявити, якщо користувач підпише чотири повідомлення двома ключами та двома одноразовими nonce. У цьому випадку вона підпише повідомлення 1 першим закритим ключем (x1), підпише повідомлення 2 другим закритим ключем (x2), підпише повідомлення 3 першим закритим ключем (x1) і підпише повідомлення 4 другим закритим ключем (x2) Той самий nonce (k1) використовується для підписання повідомлень 1 і 2, а інший nonce (k2) використовується для підпису повідомлень 3 і 4 [29].

4. Остерігайтеся помилок. У атаці з помилками в ECDSA нам потрібні лише два підписи. Один виготовлений без дефекту (r,s), а інший має дефект (rf,sf). З них ми можемо створити закритий ключ.

При атаці з помилками в ECDSA потрібні лише два підписи. Один створений без помилки (r, s), а інший має помилку (rf, sf). З них можна створити закритий ключ [29].

2.4 Режими шифрування алгоритму AES

Режими шифрування необхідні для звичайної реалізації AES. Неправильна реалізація або застосування режимів може серйозно поставити під загрозу безпеку алгоритму AES. У AES доступно кілька режимів шифрування. Деякі з режимів шифрування AES, які часто використовуються, наведені нижче [32].

Режим ECB: режим електронної кодової книги.

Режим CBC: режим ланцюжка блоків шифру.

Режим CFB: режим зворотного зв'язку шифру.

Режим OFB: режим вихідного зворотного зв'язку.

Режим CTR: Режим лічильника.

Режим GCM: режим Галуа/лічильник.

Проведемо порівняльний аналіз найпоширеніших режимів ланцюжка блоків шифру (CBC) і режим Galois/Counter (GCM).

Режим CBC: режим ланцюжка блоків шифру. У режимі CBC кожне шифрування одного і того ж відкритого тексту має призвести до іншого зашифрованого тексту. Режим CBC робить це за допомогою вектора ініціалізації. Вектор має такий самий розмір, як і зашифрований блок.

Принцип роботи AES-CBC. Спочатку до блоку відкритого тексту з вектором застосовується операція XOR. Після цього процес шифрування буде виконано за допомогою ключа шифрування. Після попередньої операції

результат кожного блоку проходить через операцію XOR наступного блоку відкритого тексту (рисунок 2.5).

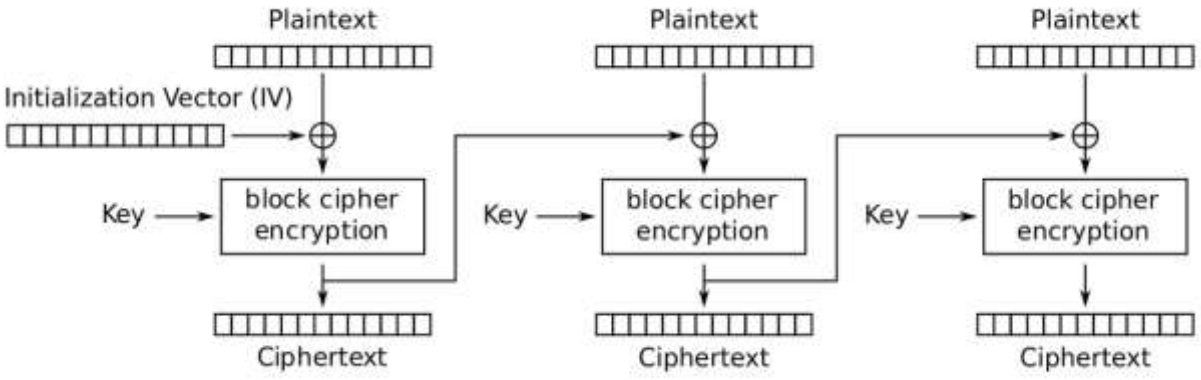


Рисунок 2.5 – Принцип шифрування AES-CBC

Проблеми в режимі CBC. Однією з головних проблем є помилка одного блоку відкритого тексту, яка вплине на всі наступні блоки. У той же час режим ланцюжка блоків шифру (CBC) вразливий до кількох типів атак.

Атака вибраного відкритого тексту (CPA) – атаки з набором вибраних відкритих текстів і отримання відповідного зашифрованого тексту.

Атака вибраного зашифрованого тексту (CCA) – Атаки з набором вибраних зашифрованих текстів для отримання відповідних відкритих текстів.

І AES-CBC, і AES-GCM здатні захистити цінні дані за допомогою якісної реалізації. але для запобігання складним атакам CBC, таким як атака вибраного відкритого тексту (CPA) і атака вибраного зашифрованого тексту (CCA), необхідно використовувати автентифіковане шифрування. Тому найкращим варіантом для цього є GCM. AES-GCM написаний паралельно, що означає, що пропускна здатність значно вища, ніж AES-CBC, завдяки зниженню накладних витрат на шифрування (рисунок 2.6).

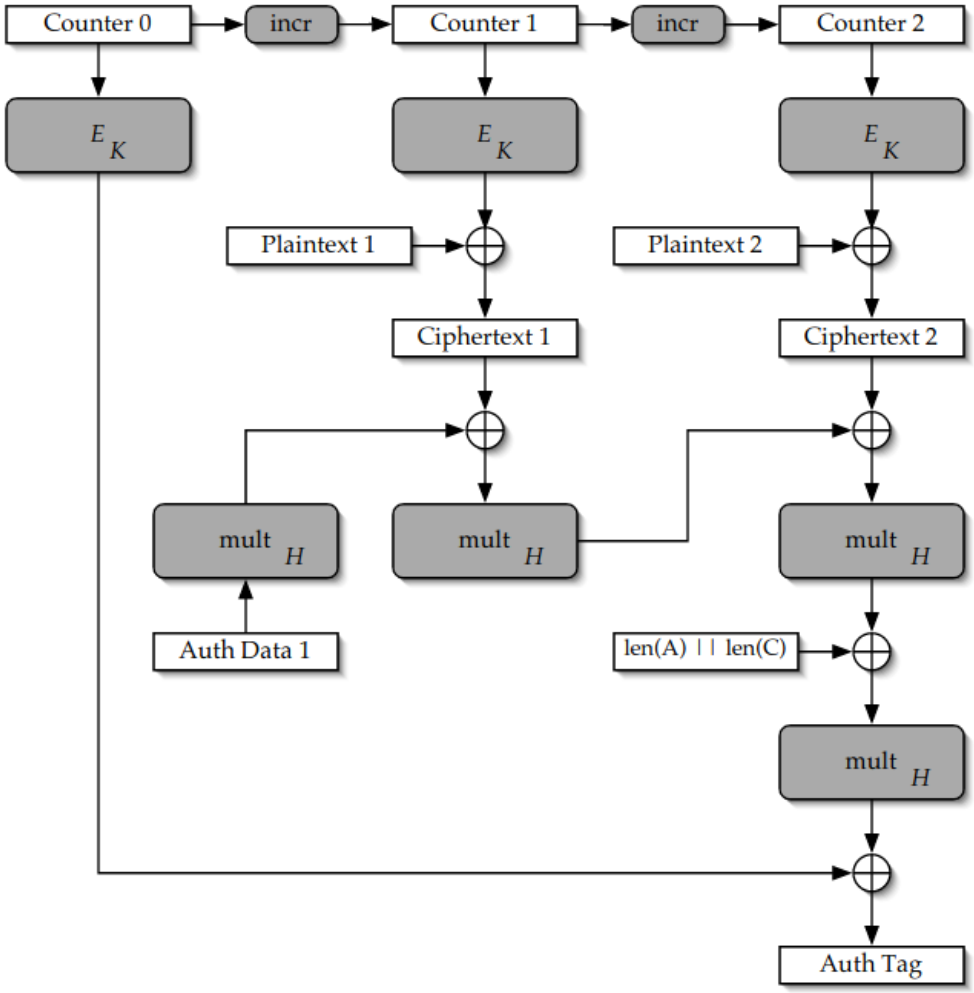


Рисунок 2.6 – Принцип шифрування AES- GCM

Режим GCM допускає конвеєрні та розпаралелені реалізації та матиме мінімальну затримку обчислень, щоб бути корисним на високих швидкостях передачі даних.

В режимі ECB (Electronic Code Book) у AES ви завжди отримуватимете той самий зашифрований текст для того самого ключа. Оскільки це блоковий шифр, доповнення змінить кінцеві символи, але поки у нас є 16 байтів, перший блок буде таким самим. Розглянемо режим потокового шифрування: GCM, і подивимося, що станеться, коли ми використовуємо той самий ключ і той самий IV.

У цьому випадку ми перетворюємо пароль на ключ шифрування, а потім просто встановлюємо вектор ініціалізації IV на всі нулі разом із тією

самою сіллю, яка використовується для генерації ключа. Якщо ми спробуємо « Taras 123» і пароль «password321», ми отримаємо:

Message: Taras 123

Cipher: **80fc096b6f98d14f54** 0d7212bffa94ff59f9b3697792d83595

Key: 2f7ffec39904ee5b61a73d881f6d2f36c27d2a60a42d828b52b6409dc13d1318

Nonce: 000000000000000000000000

Decrypted: Taras 123

Якщо використаємо той самий ключ шифрування та IV і зашифруємо «Taras 1234», ми отримаємо:

Message: Taras 1234

Cipher: **80fc096b6f98d14f54** 1f dc 29533530d4384f180dde4eb20538dc

Key: 2f7ffec39904ee5b61a73d881f6d2f36c27d2a60a42d828b52b6409dc13d1318

Nonce: 000000000000000000000000

Decrypted: Taras 1234

Якщо використаємо той самий ключ шифрування та IV і зашифруємо «Taras 12345», ми отримаємо:

Message: Taras 12345

Cipher: **80fc096b6f98d14f54** 1f 8c 7d34ee0ee13f2ca7e58b2dec68ea1068

Key: 2f7ffec39904ee5b61a73d881f6d2f36c27d2a60a42d828b52b6409dc13d1318

Nonce: 000000000000000000000000

Decrypted: Taras 12345

Відповідно, ми бачимо спільну частину «**80fc096b6f98d14f54**» для кожного шифру, і в основному це відповідає «Taras 123». Таким чином, шифрування «4»: «1f», а «5» — «8c».

Отже, отримуємо такий самий шифр для повідомлення, при однаковому ключі та тому самому векторі ініціалізації IV для кожного з режимів.

Як висновок, необхідно вибирати режим відсікання блоків Galois Counter Mode (GCM), щоб досягти безпеки даних у стані спокою.

Розглянемо приклад шифрування зображення. Для цього зашифруємо зображення версію ECB і версію CBC алгоритму AES.

Оригінальне зображення приведено на рисунку 2.7.

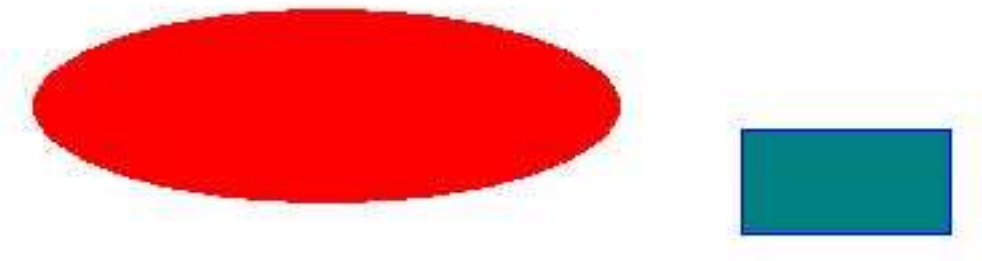


Рисунок 2.7 – Оригінальне зображення

Використовуючи бібліотеку openSSL зашифруємо файл зображення test.bmp за допомогою AES-ECB, ввівши команду:

openssl enc -aes-128-ecb -e -in test.bmp -out ECBpic.bmp -K 1011011 -iv 1011001

Зашифроване зображення версією алгоритму AES- ECB показано на рисунку 2.8.

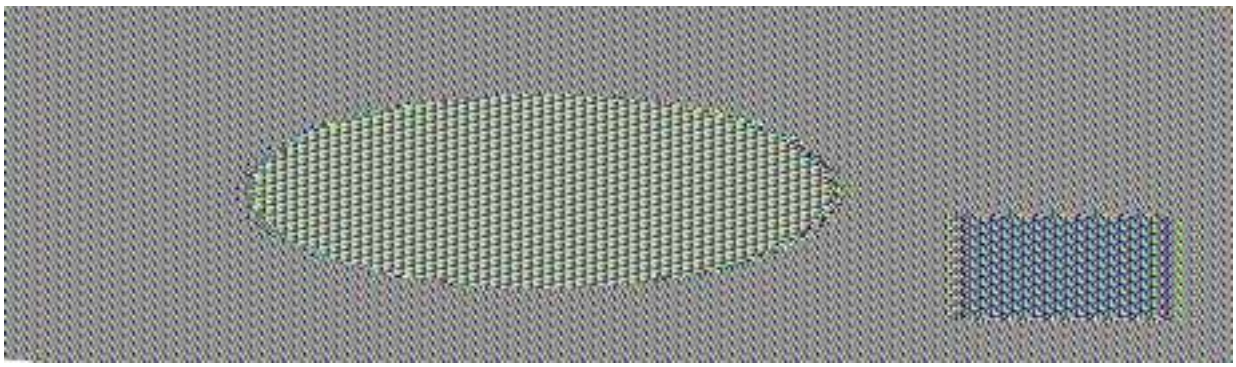


Рисунок 2.8 – Зашифроване зображення алгоритмом AES-ECB

Для шифрування файлу зображення test.bmp за допомогою версії AES-CBC вводимо команду:

openssl enc -aes-128-cbc -e -in pic_original.bmp -out CBCpic.bmp -K 1011011 -iv 1011001

Зашифроване зображення алгоритмом версії AES-CBC приведено на рисунку 2.9.

Рисунок 2.9 – Зашифроване зображення алгоритмом AES- CBC

Як видно з рисунку 2.9 алгоритм AES- CBC надійно шифрує зображення.

3 РОЗРОБКА ТА ДОСЛІДЖЕННЯ СИСТЕМИ ВІДЕОСПОСТЕРЕЖЕННЯ

3.1 Структура системи відеоспостереження на основі Інтернету речей

Великомасштабні системи спостереження стають невід'ємною частиною сучасних систем цивільного захисту. Такі системи широко розгорнуті в багатьох містах і громадах. В даний час бездротове відеоспостереження широко використовується для забезпечення моніторингу об'єктів у реальному часі. Використання бездротової системи відеоспостереження демонструє імпульс завдяки низьким накладним витратам і доступній вартості розгортання [23]. Бездротові великомасштабні системи відеоспостереження вимагають розгортання величезної кількості камер у територіально розподілених областях для досягнення своїх цілей. Ця система підключених камер зазвичай підключається через Інтернет до одного або кількох центрів агрегації відеоданих.

Характеристики бездротових систем відеоспостереження подібні до того, що сьогодні відомо як Інтернет речей (IoT), де речами в системі відеоспостереження є камери. Система відеоспостереження може використовувати наявну інфраструктуру Інтернету, яка пропонує низьку вартість і повсюдне підключення. Відеодані, зібрані набором розгорнутих камер, передаються в централізовану точку або набір точок, які завершують процес спостереження, виконуючи автоматичний аналіз відео в реальному часі.

Основна проблема такої системи пов’язана з обсягом відеоданих, створених камерами, що вимагає дуже високої пропускної здатності каналу на стороні бездротового зв’язку (Інтернету) і великої обчислювальної потужності та сховища на стороні обробки даних.

В розробленій структурі системи відеоспостереження запропоновано використати дві технології для вирішення вищезазначених задач, а саме хмару та мобільні периферійні обчислення (МПО) [24]. Система МПО

допоможе в управлінні збором даних і пропускною здатністю, щоб забезпечити ефективну передачу даних на сторону обробки системи, тоді як хмарна система запропонує практично необмежену кількість ресурсів для зберігання та аналізу відео в режимі реального часу. Зокрема, запропоновано структуру на основі Інтернету речей, яка забезпечує автоматизоване рішення для відеоспостереження. Система підтримує набір груп датчиків камери. Кожна група підключена до сервера МПО, який розташований разом із базовою станцією, до якої підключена ця група камер. Ця група камер може використовувати технологію зв'язку, яку підтримує базова станція. Набір цих груп підключено через Інтернет до центрального хмарного сервера. Камери знімають і передають відео на сервери МПО. Кожен сервер МПО зберігає відео локально, а потім пересилає його на центральний хмарний сервер. Сервер МПО може зберігати відео локально, доки він не гарантує отримання відео хмарою, а потім зможе видалити його (рисунок 3.1).

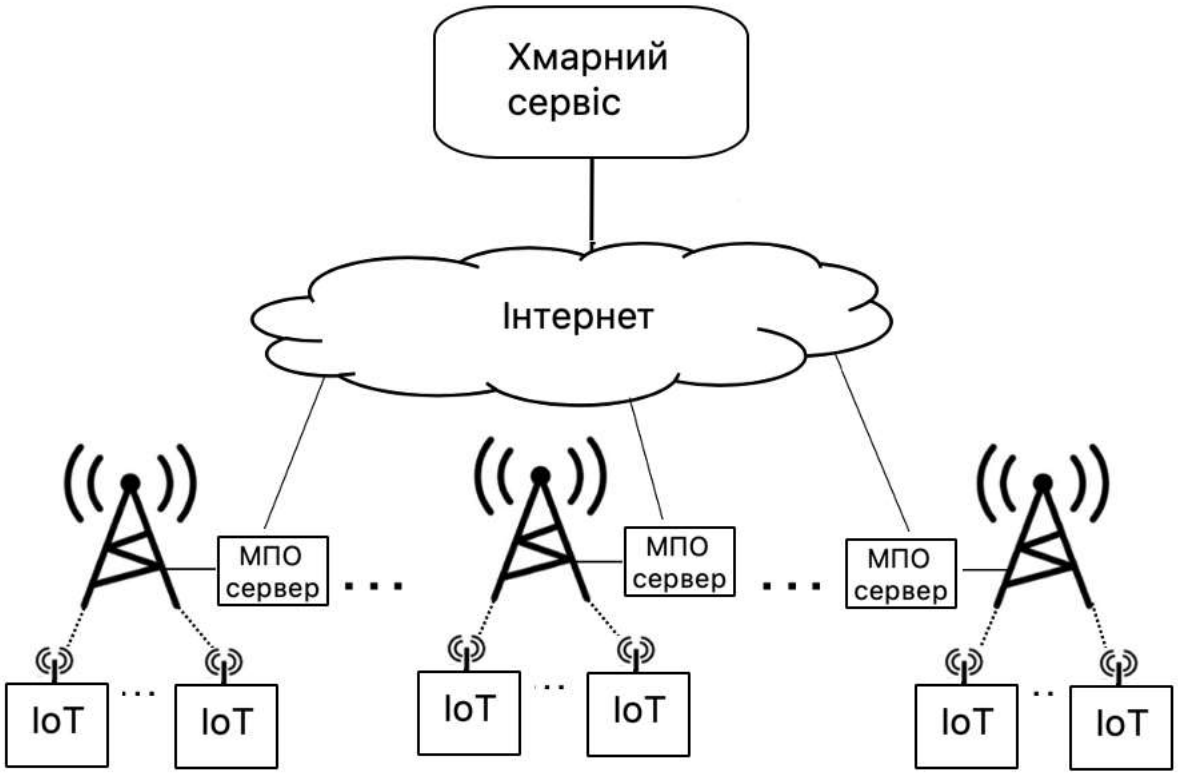


Рисунок 3.1 – Структура системи відеоспостереження

Структура забезпечує надійне зберігання відеоданих на центральному хмарному сервері з оптимальною якістю, яка повністю використовує загальну пропускну здатність центральної хмари. Управління смугою пропускання та розподіл виконується у дві частини. Усередині комірки та серед камер у групі, підключеної до однієї базової станції, керування смугою пропускання та розподіл виконується сервером МПО, розміщеним разом із цією базовою станцією. У глобальному масштабі серед серверів МПО хмарний сервер розподіляє доступну пропускну здатність між осередками.

Оскільки система спостереження, приведена на рисунку 3.1, використовує загальнодоступну мережу для підключення МПО до хмари, це може поставити під загрозу безпеку всієї системи. Це вимагає забезпечення надійної системи безпеки, яка забезпечує конфіденційність і цілісність відеопотоків, що надсилаються камерами в хмару через МПО. В запропонованій системі використано стандартні алгоритми шифрування та хешування. Зокрема, алгоритм симетричного шифрування AES для виконання наскрізного шифрування потокового відео та розроблений алгоритм для розподілу ключів AES на основі RSA [28].

Крім того для забезпечити цілісності відеопотоків використано аутентифікація повідомлень на основі хешування (HMAC-SHA256) [26]. Оскільки потокове передавання відео є дуже чутливим до затримки програмою, одна з головних цілей, якої необхідно дотримуватися під час розробки нової системи безпеки це мінімізувати затримку зв’язку.

3.2 Загальний алгоритм шифрування зображення

У процесі шифрування зображення перетворюється на послідовність байтів. В процесі шифрування виконують різні криптографічні алгоритми. Потім зашифровані байти передаються через відкриті канали в іншу систему, де вони модифікуються для отримання оригінальних зображень за

допомогою алгоритму розшифрування. Як для процесів шифрування, так і для розшифрування ми використовуємо алгоритми на основі заданого ключа.

Шифрування зображення і його дешифрування показано в рівняннях (3.1) і (3.2):

$$E(X) = Y; \quad (3.1)$$

$$D(Y) = X, \quad (3.2)$$

де E(X) – функція шифрування, що виконується на зображенні X, а її результатом є зображення Y;

D(Y) – функція розшифрування зображенні Y, а її результатом є вихідне зображення X.

Запропоноване рішення спрямоване на забезпечення автентифікації, конфіденційності та цілісності системи спостереження. Камери, МПО і хмарні пристрої складають захищену систему відеоспостереження.

Перед передачею даних пристрої повинні бути аутентифіковані один для одного. Такий процес автентифікації встановлює довіру, щоб можна було використовувати певні алгоритми шифрування для збереження конфіденційності. Для досягнення конфіденційності в запропонованій структурі та для вирішення проблем із продуктивністю використаємо алгоритм AES для шифрування відеопотоків, які надсилаються кожною камерою в хмару. Як відомо, AES потребує, щоб обидві сторони мали однаковий ключ. Для обміну цим ключем ми пропонуємо протокол, який використовує асиметричне шифрування. Протокол також використовується для автентифікації камер у системі. Запропонований алгоритм складається з наступних кроків (рисунок 3.2).

1. Камери та хмарний сервер повинні генерувати власні пари відкритих/приватних ключів.

Цей крок необхідний для автентифікації хмарного серверу на камерах. Камери можуть спілкуватися лише з хмарним сервером із відомим відкритим

ключем. Враховуючи це, усі камери повинні бути налаштовані за допомогою відкритого ключа хмарного сервісу.

Рисунок 3.2 – Протокол автентифікації та керування сеансом

2. Усі камери мають бути автентифіковані на хмарному сервісу за допомогою наступного чотиристороннього протоколу рукостискання автентифікації, і необхідно створити ключ сеансу:

2.1. Камера (DC) надсилає повідомлення (M1) до хмарного пристрою (CL) для автентифікації. M1 містить відкритий ключ DC і шифрується відкритим ключем CL: E(CL's K PUB, M1), де функція E() – функція асиметричного шифрування (наприклад, RSA).

2.2. CL розшифровує M1 і відповідає DC запитом (M2), який містить випадково згенероване число (Rnd). M2 шифрується двічі. Спочатку він шифрується закритим ключем CL, а потім відкритим ключем DC: E(DC's K PUB, E(CL's K PRIV, M2)). Це гарантує, що лише DC може читати Rnd і що він напевно надходить із CL. Що ще важливіше, цей процес забезпечує свіжість інформації від атак повторів шляхом періодичного генерування випадкових чисел.

2.3. DC розшифровує M2 спочатку за допомогою його закритого ключа, а потім за допомогою відкритого ключа CL (через двошарове

шифрування). Після цього DC збільшує Rnd на 1 для створення M3 і шифрує M3 за допомогою дворівневого шифрування; Спочатку він шифрує M3 за допомогою закритого ключа CM, а потім за допомогою відкритого ключа CL: $E(CL's\ K\ PUB, E(DC's\ K\ PRIV, M3))$.

2.4 CL розшифровує M3 і переконується, що це приріст Rnd, щоб DC пройшов виклик. Повідомлення про прийняття (M4) шифрується та надсилається до DC. M4 включає сповіщення про прийняття, ключ сеансу для подальшого зв'язку, який буде використовуватися в симетричному шифруванні, такому як AES, і час очікування сеансу. На цьому етапі сесія готова до обміну даними.

3. Після процесу автентифікації подальші ключі сеансу можуть бути безпечно згенеровані між DC і CL. Такими ключами можна керувати та безпечно надсилати їх на пристрої МПО. Після завершення автентифікації та процесу обміну ключами AES використовується для довгострокового шифрування.

4. Режим ланцюжка блоків шифру (CBC) використоано, щоб уникнути надсилання того самого шифру, коли попередні дані надсилаються знову. З CBC, навіть якщо те саме повідомлення буде надіслано знову, шифр буде іншим, оскільки кожен звичайний текст з'єднується з попереднім блоком шифру перед шифруванням. Шифрування може виконуватися для всього відеокадру, якщо використовується MJPEG, або для всього GOP, якщо використовується MPEG4 або H264, або шифрування може бути виконано для кожного пакета окремо. Щоб гарантувати цілісність відеопотоків, використовується хеш HMAC-SHA256 за допомогою узгодженого ключа сеансу.

Як і у випадку з шифруванням, HMAC-SHA256 можна обчислити для кожного пакета або кожного відеокадру. Хеш-значення додається до потоку перед шифруванням і відправленням у хмару.

3.3 Реалізація та дослідження шифрування зображень на Raspberry Pi

Raspberry Pi – це портативний чотирьохядерний комп’ютер із відкритим вихідним кодом. Він був розроблений Raspberry Pi Foundation і випускається в різних моделях, остання модель – Raspberry Pi 4.

Raspberry Pi може бути корисним у сферах обробки зображень через його портативність, паралелізм, низьку вартість і мінімальне енергоспоживання [33]. Розмір оперативної пам’яті Raspberry Pi model 4 залежить від потреб програми. Оскільки ми використовуємо модель Raspberry Pi 4 у наших програмах, ми зосередимося на вимогах останньої моделі Raspberry Pi у цьому розділі. Raspberry Pi 4, показаний на рисунку 3.3, має процесор 1,5 ГГц і доступний з LPDDR4-3200 SDRAM і декількома варіантами оперативної пам’яті 1 ГБ, 2 ГБ або 4 ГБ. Він має чотирьохядерний процесор Cortex-A72 (A.R.M. v8) і Broadcom BCM2711. Він також має два інтерфейси micro-HDMI, 40-контактний роз’єм GPIO і Gigabit Ethernet.

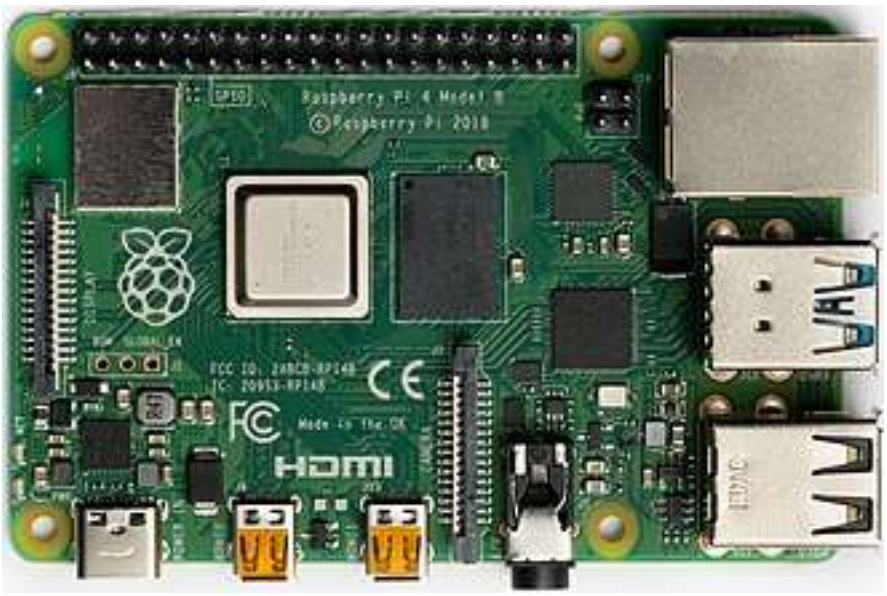


Рисунок 3.3– Raspberry Pi модель 4

Завдяки своїй портативності та можливості керування через Інтернет, платформу Raspberry Pi широко використовується для досліджень в області обробки зображень.

Програми реального часу використовують Raspberry Pi для мінімізації складності системи.

Усі етапи автентифікації та шифрування зображень виконуються на Raspberry Pi model 4.

На рисунку 3.4 показано механізм захисту переданих кольорових зображень за допомогою Raspberry Pi. Raspberry Pi шифрує зображення за допомогою AES, щоб забезпечити криптографічну автентифікацію та приховати будь-яку видиму інформацію в оригінальному зображенні.

а)

б)

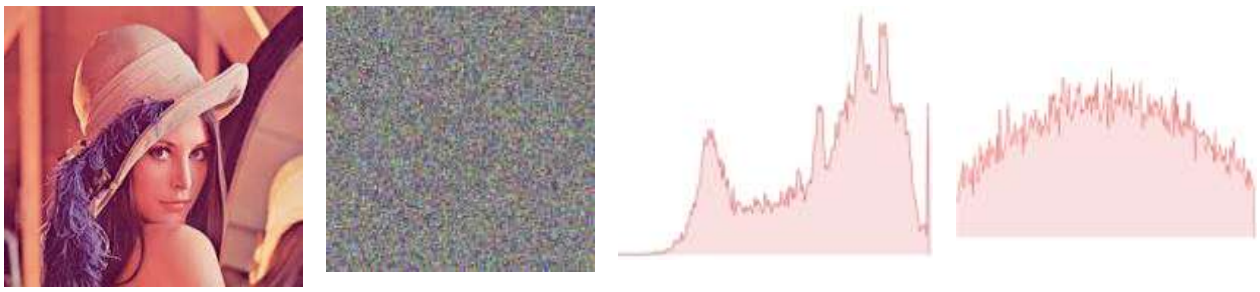
Рисунок 3.4 – Блок-схема шифрування (а) та дешифрування (б) зображення

Далі станція мобільних периферійних обчислень перевіряє цілісність отриманого зображення, щоб підтвердити його автентичність і те, що воно не було змінено. Криптографія є одним із найвідоміших методів захисту даних.

Це спосіб передачі та отримання зашифрованих даних, які може декодувати лише відправник або одержувач. Ключ, який використовується для дешифрування, відомий лише відправнику та одержувачу, тому лише призначений отримувач може інтерпретувати та декодувати дані. Цей підхід також широко використовується для захисту цифрових зображень від несанкціонованого доступу. Для захисту переданих зображень AES є практичним підходом, який можна використовувати.

Оригінальне зображення на Raspberry Pi шифрується за допомогою алгоритму AES-CBC для захисту переданого зображення. Шифрування використовує 32-байтний ключ (256-біт), який відомий лише між відправником і одержувачем для більшої безпеки. Лише отримувач із правильним ключем може розшифрувати зашифроване зображення. Інакше не має сенсу те, що зловмисники не можуть отримати з нього жодної інформації чи деталей. Ця техніка застосовується до кольорових зображень розміром 256 × 256 з різних наборів даних, щоб забезпечити продуктивність шифрування та дешифрування за допомогою AES-CBC на мові програмування C++, як показано на рисунку 3.5.

Станція моніторингу отримує ці зашифровані дані від Raspberry Pi і виконує кроки перевірки, щоб переконатися в цілісності отриманих зображень.



а) б) в) г)

Рисунок 3.5 – Кольорове зображення (а); зашифровані зображення (б); гістограма червоного каналу зображення (в); гістограма червоного каналу зашифрованого зображення (г)

3.3.1 Аналіз ключів AES

Безпечний і ефективний алгоритм шифрування повинен бути чутливим до відкритого тексту та ключа [34].

Високонадійні алгоритми шифрування зображень вимагають високої чутливості, щоб гарантувати, що зашифроване зображення не може бути правильно розшифровано, якщо ключі шифрування та дешифрування незначно відрізняються [30].

Результат аналізу показує силу алгоритму AES-CBC, де зображення не може бути розшифровано, якщо значення ключа змінено навіть на один біт, як показано на рисунку 3.6.

Рисунок 3.6 – Перевірка чутливості ключів алгоритму AES-CBC

3.3.2 Час виконання шифрування AES

Експерименти проводились на Raspberry Pi модель 4 з процесором 1,5 ГГц. Для запуску алгоритму шифрування AES-CBC з використанням crypto++, алгоритму потрібно 0,004 с для шифрування одного кольорового

каналу зображення «Lena» розміром 256 × 256 і 32-байтним ключем. А для шифрування всіх трьох каналів RGB зображення потрібно 0,011 с.

3.3.3 Аналіз інформаційної ентропії

Ентропія використовується для оцінки продуктивності алгоритму шифрування. Ентропія є важливою характеристикою, яка відображає випадковість джерела інформації та визначає непередбачуваність. Кожен канал RGB у кольоровому зображенні представлено у вигляді 8 біт із значеннями пікселів, що змінюються від 0 до 255.

Таким чином, ентропія має максимальне значення 8. Значення ентропії зашифрованого зображення має бути близьким до 8 для ефективного алгоритму шифрування зображення. Ентропія зображення шифру «Lena» наведена в таблиці 3.1. Ентропію зображення можна розрахувати як [35].

$$H\ (m) = -\sum_{i=0}^{2^n-1} P(x_i) \cdot log_2(x_i)$$

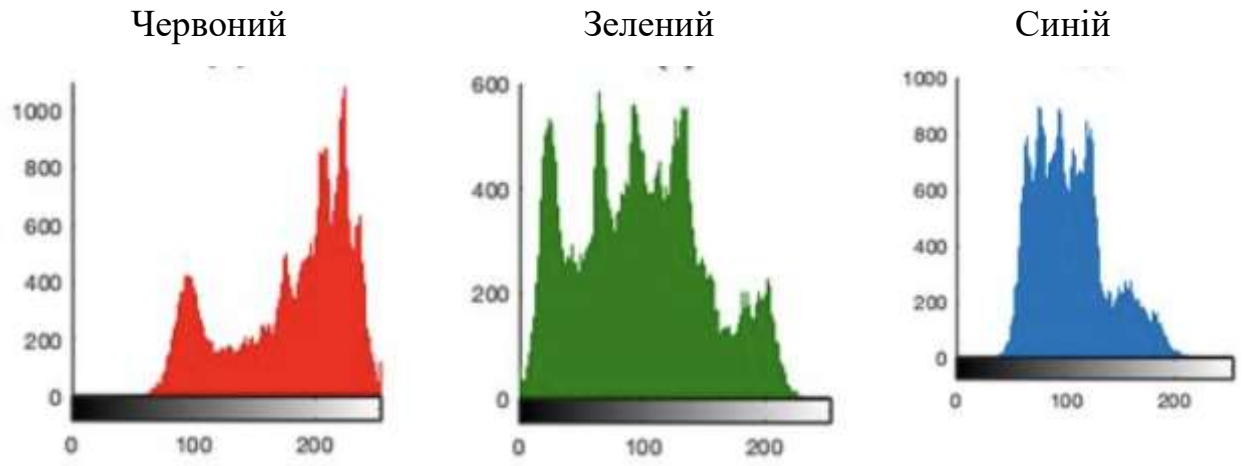
Таблиця 3.1 – Інформаційна ентропія як простого, так і зашифрованого зображення «Lena»

H(m)	R	G	B
Зашифроване зображення	7,8961	7,8964	7,8964

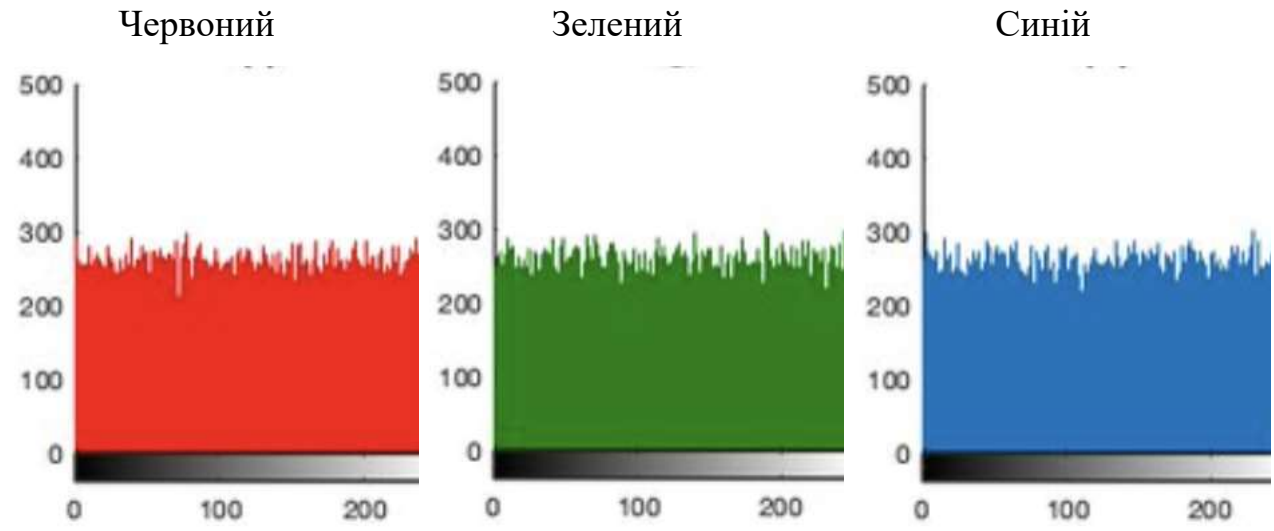
3.3.4 Аналіз гістограм

Гістограма зображення відображає, як розподілені пікселі на зображенні. Коли гістограма рівномірно розподілена, це означає, що статистичні атаки мають меншу ймовірність успіху. Аналіз гістограми використовується для ідентифікації розподілів значень пікселів відкритого та зашифрованого тексту. Гістограма зашифрованого зображення рівномірно розподілена. Воно суттєво відрізняється від простого зображення, як показано на рисунку 3.7.

Крім того, немає втрати якості зображення після етапу дешифрування. Наявність зашифрованого зображення ускладнює для зловмисників отримання оригінальних зображень або будь-якої інформації про них. Це означає, що AES-CBC достатньо потужний, щоб впоратися зі статистичними атаками. На рисунку 3.7 показана гістограма для каналів RGB для звичайного зображення Lena і зашифрованого зображення Lena відповідно.



а)



б)

Рисунок 3.7 – Гістограма простого зображення «Lena» (a); гістограма зашифрованого зображення «Lena» (b)

3.3.5 Кореляційний аналіз

Коефіцієнт кореляції описує, як співвідносяться сусідні пікселі зображення один з одним. Як і в таблиці 3.2, сусідні пікселі сильно корелюють у всіх трьох напрямках на простому зображенні «Lena» (≈ 1). Кореляція між сусідніми пікселями в трьох напрямках зашифрованого зображення повинна бути настільки малою, наскільки це можливо (≈ 0), щоб протистояти статистичним атакам.

Таблиця 3.2 – Коефіцієнт кореляції як звичайного, так і зашифрованого зображення «Lena»

	Просте зображення			Зашифроване зображення		
	R	G	B	R	G	B
Вертикаль	0,9563	0,9523	0,9521	-0,0036	0,0007	0,0009
Горизонталь	0,9663	0,9413	0,9302	-0,0031	0,0007	-0,0047
Діагональ	0,9463	0,9343	0,9114	-0,0022	-0,0007	0,0037

Коефіцієнт кореляції сусідніх пікселів обчислюється за формулою [35]:

$$r_{mn} = \frac{Cow(m,n)}{\sqrt{D(m)\cdot\sqrt{D(n)}}}$$

де:

m і n – значення сусідніх пікселів зображення;

$Cov(m,n)$ – коваріація;

$D(m)$ – це дисперсія.

Інформаційна безпека має важливе значення для захисту переданих медіа, таких як зображення, щоб забезпечити тріаду CIA. Запропонований підхід захисту даних реалізовано на пристрої Raspberry Pi, який можна використовувати в складних умовах, оскільки він вирішує проблему обмеженої мобільності стаціонарного комп'ютера. Raspberry Pi шифрує дані перед надсиланням за допомогою техніки шифрування AES-CBC із використанням 256-бітного симетричного ключа, відомого лише передавачу

Raspberry Pi і приймачу. Потім приймач може розшифрувати отримані дані, щоб підтвердити їх цілісність та конфіденційності зображень.

3.4 Оцінка пропускної здатності

Розподіл пропускної здатності періодично виконується хмарним сервісом. Нехай B – загальний бюджет пропускної здатності, наданий постачальником хмарних послуг, який буде використовуватися системою спостереження. Припустимо також, що M_i представляє і-й МПО, підключений до хмари, де $i = 1, 2, \ldots, N$ і N – загальна кількість МПО, підключених до хмари. Крім того, припустимо, що c_i – кількістю камер, підключених до МПО M_i.

Один із спосіб обчислення пропускної здатності для і-го МПО, B_i, полягає в тому, щоб розділити загальну пропускну здатність, яку забезпечує хмара (B), на кількість підключених до хмари МПО (N):

$$B_i = \frac{B}{N}. \tag{3.1}$$

Цей розподіл забезпечує справедливість між областями, які контролюються різними осередками. Цей розподіл називають простим глобальним розподілом (ГР). Як видно із формули (3.1), єдиною інформацією, необхідною для цього розподілу, є N, що є кількістю підключених МПО. Цей розрахунок може бути виконано хмарним сервером, після чого повідомлення надсилається широкомовно на всі сервери МПО із загальною пропускною здатністю, яку вони повинні використовувати локально.

Іншим варіантом розподілу пропускної здатності є розподіл загальної пропускної здатності між МПО, враховуючи кількість камер, підключених до

кожного МПО: Розподіл пропускної здатності, із врахуванням кількості камер, обчислюється за формулою:

$$B_i = \frac{B}{\sum_i^N c_i} \times c_i. \quad (3.2)$$

Розподіл обчислений за формулою (3.2) називають зваженим глобальним розподілом (ЗГР). Зважений глобальний розподіл обчислює частку $\frac{B}{\sum_i^N c_i}$ пропускної здатності кожної камери в системі. Значення ЗГР може бути обчислено хмарним сервером після отримання кількості підключених камер c_i від кожного M_i. Потім значення транслюється всім користувачам МПО. Потім кожен МПО може обчислити свою частку від загальної пропускної здатності, помноживши отримане значення терміну на кількість камер, підключених до нього.

3.3.1 Локальний розподіл пропускної здатності

Коли кожен МПО визначає свою смугу пропускання B_i, він розподіляє B_i між підключеними камерами в комірці МПО. Як правило, це розділення смуги пропускання МПО на кількість камер, підключених до МПО. Однак тут є ряд проблем, на які слід звернути увагу.

1. Фізична бітова швидкість комірки. Це максимальна швидкість передачі даних (пропускна здатність), яку підтримує стільникова мережа. Цей бітрейт залежить від технології зв'язку, що використовується в комірці.

2. Ефективна пропускна здатність у комірці. Це пропускна здатність, яку можна використовувати для доставки відеопотоків спостереження на сервер МПО. У мережі частка фізичної смуги пропускання споживається накладними витратами протоколу зв’язку. Крім того, інша частина споживається помилками зв'язку та перехресним трафіком у мережі. Очевидно, що ефективна пропускна здатність набагато менша, ніж фізична пропускна здатність будь-якої мережі. Ефективна смуга пропускання завжди

повинна бути більшою, ніж частка смуги пропускання МПО, щоб МПО міг використовувати всю свою частку пропускної здатності.

3. Кількість підключених камер. Це число може змінюватися з часом, оскільки деякі камери можуть вийти з ладу або розрядитися акумулятор.

4. Тип стиснення відео, який використовується камерою. Більшість камер спостереження підтримують три алгоритми стиснення: MJPEG, MPEG4 і H.264.2. MJPEG використовує коефіцієнт якості як вхідний сигнал і зазвичай створює кращу якість відео з вищою швидкістю, ніж два інших алгоритми стиснення. Що вищий коефіцієнт якості, то вища якість відео та бітрейт. MPEG4 і H.264 приймають бітрейт як вхідні дані та створюють відеопотік із цим бітрейтом. H.264 забезпечує більший рівень стиснення, ніж MPEG4.

5. Як примусово розподілити смугу пропускання на камерах. В [34] запропоновано оптимальне рішення, яке можна використовувати для розподілу доступної смуги пропускання в комірці на основі Wi-Fi між набором камер, щоб мінімізувати спотворення отриманого відео на центральній станції. Центральну станцію можна замінити сервером МПО.

3.3.2 Показники продуктивності

При оцінці ефективності використано такі показники.

1. Пікове відношення сигнал/шум (PSNR): це один із найбільш використовуваних показників для оцінки QoS передачі відео на рівні програми. PSNR описується Міжнародним союзом електрозв'язку як [35]:

$$PSNR(n)_{db} = 20\ Log\left[\frac{V_{peak}}{\sqrt{MSE(n)}}\right]$$

де $Vpeak = 2^k - 1$, k означає кількість бітів, які використовуються для представлення пікселя. Середня квадратична помилка (MSE) – це оцінка дисперсії помилки, а значення MSE обчислюється за формулою:

$$MSE(n) = \frac{\sum_{i=1}^{N_{col}} \sum_{j=1}^{N_{row}} [Y_S(n,i,j) - Y_D(n,i,j)]^2}{N_{col} \cdot N_{row}}$$

де N_{row} і N_{col} – кількість рядків і стовпців у зображенні, i і j – поточна позиція стовпця та рядка, n – номер поточного кадру, Y_S – світлова складова вихідного зображення, Y_D – світлова складова зображення призначення, як визначено в [35].

Загальна швидкість отриманих даних: це загальна кількість даних, отриманих на хмарному сервері протягом певного періоду часу від усіх камер, усереднена за час. Він розраховується як загальний розмір пакетів, отриманих від усіх камер, поділений на час, протягом якого збираються дані. У наших експериментах загальна швидкість отриманих даних розраховується за кожну секунду. Згодом ці значення усереднюються для обчислення загального середнього.

Затримка пакетів. Під час розрахунку затримки пакетів враховувалися два основні фактори [36, 37].

1. Час обробки, необхідний для алгоритмів AES.

2. Мережева затримка.

Час, необхідний для передачі пакету по всьому шляху в мережі. Затримка мережі спричинена:

а) затримкою обробки, тобто часом, необхідним для аналізу, обробки та прийняття рішення про те, куди буде надіслано пакет;

б) затримкою буфера, тобто часом, протягом якого певний пакет залишається в черзі, поки він вилучено з черги для передачі;

c) затримку передачі, яка є загальним часом, протягом якого всі біти пакету надсилаються до потрібного середовища передачі;

d) затримку розповсюдження, яка є часом, протягом якого біт розповсюджується через мережі, доки не досягне кінця своєї фізичної траєкторії.

Під час обчислення середньої затримки в кожному експерименті ми використовуємо різницю в часі між часом, коли пакет отримано на прикладному рівні хмарного сервера, і часом, коли пакет було надіслано з прикладного рівня камери. Наприкінці кожної секунди обчислюється середня затримка всіх пакетів, отриманих протягом цієї секунди, а потім обчислюється середнє значення цих середніх значень. Крім того, розраховується час шифрування для кадру та обчислюється середнє значення часу шифрування для всіх відеокадрів, а потім воно ділиться на середню кількість отриманих пакетів з кожного кадру, в результаті середній час шифрування на пакет додається до розрахованої затримки пакета.

Загальне навантаження на мережу. Це загальна швидкість трафіку (даних), створена всіма джерелами відео в мережі. Швидкість надсилання кожного джерела відео обчислюється на прикладному рівні як загальний розмір пакетів даних, надісланих до точки доступу, поділений на час, протягом якого дані збираються. У експериментах швидкість надсилання обчислюється щосекунди, а потім зберігається. Потім середня швидкість надсилання обчислюється для кожного джерела відео як середнє значення збережених значень швидкості надсилання. Зрештою, загальне навантаження на мережу було розраховано як суму середніх швидкостей надсилання всіх джерел відео, які беруть участь в експерименті.

ВИСНОВКИ

В кваліфікаційній роботі розв’язано актуальну задачу підвищення ефективності алгоритмів захисту інформації в системах відеоспостереження

При цьому отримано наступні результати.

1. Проведено аналіз підходів до захисту даних в системах відеоспостереження. Серед який виділено: доставка контенту через HTTPS, програми потокового відео, захищені паролем, захищені центри відеоданих, алгоритми симетричного шифрування, шифрування даних при передачі.

2. Досліджено технології побудови систем відеоспостереження, зокрема: Інтернет-речей, граничні обчислення та технологія FPGA. Проектування систем відеоспостереження на основі вказаних технологій дозволить значно підвищити їх ефективність.

3. Проаналізовано основні криптографічні примітиви для шифрування зображень. Вказано їх переваги та можливість застосування при проектуванні захищених систем відеоспостереження.

4. Розроблено структуру системи відеоспостереження на основі технології мобільних периферійних обчислень та хмарного сервісу, яка . забезпечує надійне зберігання відеоданих на центральному хмарному сервері з оптимальною якістю. Для забезпечити цілісності відеопотоків використано аутентифікація повідомлень на основі хешування HMAC-SHA256.

5. Розробити алгоритм шифрування зображення на основі AES з режимом ланцюжка блоків, що забезпечує створення нового шифру при надсиланні тих самих зображень.

6. Розроблено та досліджено шифрування зображень на базі Raspberry Pi. При оцінці ефективності використано такі показники: пікове відношення сигнал/шум (PSNR), затримка пакетів та загальне навантаження на мережу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Zhang, X.; Seo, S.-H.; Wang, C. A Lightweight encryption method for privacy protection in surveillance videos. IEEE Access 2018, 6, 18074–18087.

2. Zhang, K.; Ni, J.; Yang, K.; Liang, X.; Ren, J.; Shen, X.S. Security and privacy in smart city applications: Challenges and solutions. IEEE Commun. Mag. 2017, 55, 122–129.

3. Kim, J.; Lee, D.; Park, N. CCTV-RFID enabled multifactor authentication model for secure differential level video access control. Multimed. Tools Appl. 2020, 79, 23461–23481.

4. Aamir Nizam Ansari, Mohamed Sedkyl, Neelam Sharm. „An Internet of Things Approach for Motion Detection using Raspberry Pi“ in 2015 International Conference on Intelligent Computing and Internet of Things, pp. 131-134

5. Ways to Protect IP Video Surveillance Systems. Електронний ресурс]. - Режим доступу: https://www.alliedtelesis.com/ua/en/blog/5-ways-protect-ip-video-surveillance-systems

6. The Fun and User-Friendly Guide to the Secure Real-time Transport Protocol [Infographic]. Електронний ресурс]. - Режим доступу: https://medium.com/callstatsio/the-fun-and-user-friendly-guide-to-the-secure-real-time-transport-protocol-infographic-b5ddf7da3f3e

7. Nazare, A.C., Jr.; Schwartz, W.R. A scalable and flexible framework for smart video surveillance. *Comput. Vis. Image Underst.* **2016**, *144*, 258–275.

8. Sultana, T.; Wahid, K.A. IoT-guard: Event-driven fog-based video surveillance system for real-time security management. *IEEE Access* **2019**, *7*, 134881–134894.

9. Guo, J.; Zheng, P.; Huang, J. An efficient motion detection and tracking scheme for encrypted surveillance videos. *ACM Trans. Multimed. Comput. Commun. Appl.* 2017, *13*, 1–23.

10. Obermaier, J.; Hutle, M. Analyzing the security and privacy of cloud-based video surveillance systems. In *Proceedings of the 2nd ACM International*

Workshop on IoT Privacy, Trust, and Security; Association for Computing Machinery: New York, NY, USA, 2016; pp. 22–28.

11. K. M. Hosny, A. Magdi, A. Salah, O. El-Komy and N. A. Lashin, "Internet of things applications using Raspberry-Pi: A survey," *International Journal of Electrical & Computer Engineering*, vol. 13, pp. 2088– 8708, 2023.

12. V. A. Daisy, C. V. Joe and S. S. S. Sugi, "An image-based authentication technique using visual cryptography scheme," in *2017 Int. Conf. on Inventive Systems and Control (ICISC)*, Piscataway, IEEE, pp. 1–6, 2017. https://doi.org/10.1109/ICISC.2017.8068666

13. M. A. Razzaq, R. A. Sheikh, A. Baig and A. Ahmad, "Digital image security: Fusion of encryption, steganography, and watermarking," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 8, no. 5, pp. 224–228, 2017.

14. P. Chinnasamy, S. Padmavathi, R. Swathy and S. Rakesh, "Efficient data Security using hybrid cryp- tography on cloud computing," in *Inventive Communication and Computational Technologies*, Singapore: Springer, pp. 537–547, 2021.

15. M. Begum and M. S. Uddin, "Digital image watermarking techniques: A review," *Information*, vol. 11, no. 2, pp. 110, 2020.

16. S. Gupta, U. Raikar, B. M. P. Patil and R. Molavade, "Image processing based intelligent traffic control system by using Raspberry Pi," *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, vol. 6, pp. 66–70, 2018.

17. K. S. Shilpashree, H. Lokesha and H. Shivkumar, "Implementation of image processing on Raspberry Pi," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 5, pp. 199–202, 2015.

18. A. S. Alanazi, N. Munir, M. Khan, M. Asif and I. Hussain, "Cryptanalysis of novel image encryption scheme based on multiple chaotic substitution boxes," *IEEE Access*, vol. 9, pp. 93795–93802, 2021.

19. M. Khan, F. Masood and A. Alghafis, “Secure image encryption scheme based on fractals key with Fibonacci series and discrete dynamical system,” *Neural Computing and Applications*, vol. 32, no. 15, pp. 11837–11857, 2019.

20. R. S. Devi, A. N. Aravind, J. C. Vishal, D. Amritha, K. Thenmozhi *et al.,* “Image encryption through RNA approach assisted with neural key sequences,” *Multimedia Tools and Applications*, vol. 79, pp. 12093–12124, 2020.

21. Aazam M, Huh EN (2014) Fog computing and smart gateway based communication for cloud of things. In: 2014 International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, pp 464–470

22. Brengel, M., & Rossow, C. (2018, September). Identifying key leakage of bitcoin users. In International Symposium on Research in Attacks, Intrusions, and Defenses (pp. 623–643).

23. Sadeeq, M. M., Abdulkareem, N. M., Zeebaree, S. R., Ahmed, D. M., Sami, A. S., & Zebari, R. R. (2021). IoT and Cloud computing issues, challenges and opportunities: A review. Qubahan Academic Journal, 1(2), 1-7.

24. Balaji, S., Nathani, K., & Santhakumar, R. (2019). IoT technology, applications and challenges: a contemporary survey. Wireless personal communications, 108, 363-388.

25. Yazdeen, A. A., Zeebaree, S. R., Sadeeq, M. M., Kak, S. F., Ahmed, O. M., & Zebari, R. R. FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review. *Qubahan Academic Journal*, *1*(2), 2021, pp. 8-16.

26. Buchanan, William J (2023). HMAC (Key Hash Message Authentication Code) in Golang. Asecuritysite.com. https://asecuritysite.com/mac/go_hmac

27. Buchanan, William J (2023). *RIPEMD (RACE Integrity Primitives Evaluation Message Digest)*. Asecuritysite.com. https://asecuritysite.com/hash/ripe

28. The hash function RIPEMD-160. Електронний ресурс]. - Режим доступу: https://homes.esat.kuleuven.be/~bosselae/ripemd160.html

29. Genç, Y., & Afacan, E. Design and implementation of an efficient elliptic curve digital signature algorithm (ECDSA). In 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2021, pp. 1-6.

30. Buchanan, William J (2023). Elliptic Curve Digital Signatures with Node.js (ECDSA). Електронний ресурс]. - Режим доступу: https://asecuritysite.com/node/node_signec

31. Poddebniak, D., Somorovsky, J., Schinzel, S., Lochter, M., & Rösler, P. Attacking deterministic signature schemes using fault attacks. In 2018 IEEE European Symposium on Security and Privacy (EuroS&P), 2018, pp. 338–352.

32. Sullivan, G. A., Sippe, J., Heninger, N., & Wustrow, E. Open to a fault: On the passive compromise of {TLS} keys via transient errors. In 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 233–250.

33. Huu-Quoc Nguyen, Ton Thi Kim Loan, Bui Dinh Mao and Eui-Nam Huh, „Low Cost Real-Time System Monitoring Using Raspberry Pi“ in ICUFN 2015, pp. 857-859

34. Alsmirat M. A, Jararweh Y, ObaidatI, Gupta B.B. Internet of surveillance: a cloud supportedl argescale wireless surveillance system. J Supercomput, 2016, pp.1–20.

35. Ke CH, Shieh CK, Hwang WS, Ziviani A et al. An evaluation framework for more realistic simulations of mpeg video transmission. J Inf Sci Eng, 2008, 24(2), pp.425–440

36.Пастух Т.І., Голод Ю.В., Мачуляк М.В. Алгоритм захисту інформації в системах відеоспостереження. Матеріали науково-практична конференція молодих вчених, аспірантів та студентів «Кібербезпека та комп’ютерно-інтегровані технології» (КБКІТ - 2023), Тернопіль, 2023. – С. 175-176.

37. Пастух Т.І., Дзівак О.А., Понедєльніков Г.М. Автентифікації та перевірка цілісності зображень на основі хешу. Матеріали науково-практичного симпозіуму «Захист інформації», Тернопіль, 2023. – С. 135-137.

ДОДАТОК А

Копії публікацій