ISSN 2519-2310

CS&CS Journal

KARAZIN UNIVERSITY
CLASSICS AHEAD OF TIME

2(20)2021

COMPUTER SCIENCE AND CYBERSECURITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені В.Н.КАРАЗІНА
ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ имени В.Н. КАРАЗИНА
V.N. KARAZIN KHARKIV NATIONAL UNIVERSITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА
КОМПЬЮТЕРНЫЕ НАУКИ И КИБЕРБЕЗОПАСНОСТЬ
COMPUTER SCIENCE AND CYBERSECURITY
(CS&CS)

Issue 2(20) 2021

Заснований 2015 року

Міжнародний електронний науково-теоретичний журнал
Международный электронный научно-теоретический журнал
International electronic scientific journal

The journal publishes research articles on theoretical, scientific and technical problems of effective facilities development for computer information communication systems and on information security problems based on advanced mathematical methods, information technologies and technical means.

The journal is published every six months.
Approved for placement on the Internet by Academic Council of the Karazin Kharkiv National University (December 28, 2021, protocol No.15)

The journal has Digital Object Identifier: **10.26565/2519-2310**.

Editorial office:
V.N. Karazin Kharkiv National University
Svobody Sq., 6, office 315a, Kharkiv, 61022, Ukraine *(North building of University, 3th floor)*
Phone: +38 (057) 705-10-83
E-mail: *cscsjournal@karazin.ua*
Web-page: *http://periodicals.karazin.ua/cscs (Open Journal System)*

Published articles have been internally and externally peer reviewed

TABLE OF CONTENTS

Issue 2(20) 2021

УДК 681.04 DOI: 10.26565/2519-2310-2021-2-01

ОЦІНКА ЕФЕКТИВНОСТІ СКАНЕРІВ БЕЗПЕКИ ВЕБ-ДОДАТКІВ

Олексій Прищепа, Дмитро Іваненко

Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна
pryshchepa2020kb51@student.karazin.ua, i8o.dima@gmail.com

Рецензент: В'ячеслав Калашников, д. ф.-м.н., проф., Технологічний університет Монтеррея, 64849 Монтеррей, Нуево-Леон, Мексика
kalash@itesm.mx

Надійшло: Жовтень 2021.

Анотація: *Рівень захищеності веб-додатків з кожним роком постійно зростає, але нові рейтинги найбільш поширених загроз безпеки, свідчать проте, що проблема забезпечення їх захищеності є дуже актуальною, та постійно змінною. Тому, вкрай актуально розуміти важливість використання автоматичних сканерів безпеки веб-додатків та вміти об'єктивно оцінювати їх реальну ефективність. У роботі розглянуто процес тестування веб-додатків на наявність вразливостей (та приклади їх виявлення), з використанням безкоштовних веб-сканерів (з відкритим кодом) методом «чорного ящику». Тобто, в даному випадку, сканери взаємодіють з додатками так само, як і типовий користувач через веб-інтерфейс, за допомогою протоколу HTTP. Головною метою проведених тестувань є порівняння декількох сканерів з відкритим кодом, та визначення їх ефективності. Підкреслено, що оцінити всі показники сканерів не є можливим, внаслідок існування багатьох факторів. - Тому, в межах даної роботи, усі судження та висновки були зроблені тільки на основі аналізу отриманих звітів кожного тестового сканеру. В статі приведені відомості, щодо окремих параметрів та кількості знайдених вразливостей. Отримані результати тестувань свідчать, про те, що практика використання тільки одного сканеру э мало ефективною, тому при тестуванні потрібно використовувати одразу декілька різних рішень. Це надасть змогу отримати більш об'єктивні результати, в частині детектування, як вже відомих загроз безпеки, так і знаходження нових вразливостей (з їх додаванням до звіту). Робота буде корисна для тих, хто цікавиться питаннями оцінки стану безпеки сучасних веб-додатків.*

Ключові слова: *вразливості веб-додатків; сканери вразливостей; ефективність сканерів; тестування безпеки веб-додатків, пентестинг.*

1 Вступ

Проблема небезпечного програмного забезпечення (ПЗ) є, найважливішою технічною проблемою нашого часу, яка з кожним роком не перестає бути актуальною. Різке зростання числа веб-додатків, за останні декілька років, що забезпечують роботу бізнесу, комунікацію через соціальні мережі, розваги та сервіси, пошуку інформації і т.д. Лише посилило вимоги до створення надійного методу до написання й захисту веб-додатків та даних у них.

Зрозуміло, що неможливо створити безпечний веб-додаток, не проводячи, при розробці та на стадії впровадження, тестування безпеки. Тестування є частиною більш широкого підходу до створення безпечної системи. Багато організацій, які займаються розробкою програмного забезпечення не включають, чи включають не повністю, тестування безпеки до свого стандартизованого процесу розробки програмного забезпечення.

Само по собі тестування безпеки не є особливо гарним показником того, наскільки захищений веб-додаток, тому що існує безліч способів, якими зловмисник може зламати додаток, і просто неможливо перевірити їх всі, але тестування безпеки може знайти прості у реалізації, та популярні вразливості. [1-2]

2 Основна частина

Одна з цілей тестування безпеки полягає в тому, щоб перевірити, що засоби управління безпекою працюють належним чином. Це задокументовано за допомогою вимог безпеки, які описують функціональність контролю безпеки. На високому рівні це означає підтвердження

конфіденційності, цілісності і доступності даних, а також послуг. Інша мета полягає в тому, щоб перевірити, що засоби управління безпекою реалізовані практично без вразливостей.

Тестування рівня безпеки, як в ручному режимі, так і автоматично, проходить у відповідності з наступними етапами [2]:

- Розвідка – програма чи людина намагається дізнатися якомога більше інформації про встановлену систему. Це включає в себе спроби визначити яке програмне забезпечення використовується та якої версії, які точки входу існують, пошук прихованого контенту, відомих вразливостей та інших ознак слабкості.
- Атака – на цьому етапі тестувальник намагається використовувати відомі або передбачувані вразливості, щоб довести їх існування.
- Звіт – цей документ є результатом роботи тестувальника або програми, до якого, як правило, включається перелік вразливостей, які можливо експлуатувати *(в деяких випадках включає потенційні вразливості)*, ступінь їх ступінь їх небезпеки, вірогідні наслідки і можливі варіанти вирішення цієї загрози та іншу додаткову інформацію.

Кінцевою метою тестування на безпечність – є пошук вразливостей, для подальшого їх усунення, а також для перевірки та підтвердження, що вразливості були виправлені.

Якщо говорити про порівняння сканерів вразливостей, то можна розглянути наступні завдання [3]:

1) порівняння функціональних можливостей сканерів;
2) прийняття чогось за еталон тестування та перевірка сканерів з метою визначення ефективності їх роботи з різними видами веб-додатків.

Перше завдання *(порівняння функціональних можливостей сканерів)* можливо вирішити прийнявши за основу порівняння контрольний список, наприклад створений на основі комбінації критеріїв представлених в проектах *«OWASP top-10»* або *«Web Application Security Scanner Evaluation Criteria»* та інших, можливостей ідеального сканера і оцінити інструменти на підтримку цих можливостей. В результаті отримаємо таблицю відповідності за даними критеріями. Порівняння може бути проведено, як теоретично, так і практично.

Друге завдання, поки ще, не має остаточного рішення.

Під ефективністю роботи сканерів вразливостей, будемо розуміти сукупність з двох компонентів: – повноти відповіді та частоти помилкових спрацьовувань. Повнота визначається, як відношення кількості виявлених вразливостей до загальної кількості вразливостей у додатку. Частота помилкових спрацьовувань визначається як відношення кількості неправильно знайдених вразливостей до загальної кількості повідомлених вразливостей. Теоретично, «ідеальний» інструмент повинен демонструвати повноту рівну одиниці, а частоту помилкових спрацьовувань як нуль [2].

Слід зазначити, що даний метод має два основних недоліки:

- зазначені показники залежать від тестової вибірки;
- визначення ефективності є марним з практичної точки зору.

Перший недолік показує, що показниками ефективності при порівнянні інструментів між собою можна маніпулювати, включаючи або виключаючи з тестової вибірки для визначення еталону ті чи інші веб-додатки або вразливості в них.

Другим недоліком є те, що сканери відрізняються один від одного, і коли, наприклад, один виявляє, в тестовій вибірці, всі вразливості до атак класу *XSS*, але не помічає жодної вразливості до атак інших класів, тоді як другий, навпаки, виявляє всі вразливості до атак класу *SQL*-ін'єкції, і «не бачить» жодної вразливості до атак інших класів. В такому разі ефективність, яка може бути задана у висновку - це кількісні характеристики тестової вибірки по класах вразливостей до атак *XSS* і *SQL*- ін'єкції, відповідно.

Також важливим фактором, при порівнянні ефективності веб-сканерів, є оцінка ефективності їх пошукових машин – «краулерів». Так як у автоматичних засобів сканування першим етапом аналізу є збір даних про веб-додаток та визначення точок входу, полів для вве-

дення даних, то неточні результати (на цьому етапі) істотно впливають на можливість знайти вразливості при тестуванні знайдених місць входу.

Отже переходячи з класифікації сканерів до вразливостей слід підкреслити, що сканери безпеки веб-додатків виявляють вразливості у таких основних категоріях як [2-3]:

- Вразливості на етапі кодування. Найкраще виявляються, серед вразливостей, пов'язані з некоректною обробкою вхідних і вихідних даних, у випадках коли може виникнути логічна помилка та вразливість.
- Вразливості на етапі впровадження та налаштування програми. До цих вразливостей відносяться некоректні налаштування оточення веб-додатку: веб-серверу, сервера додатків, *SSL/TLS*, фреймворків, сторонніх компонентів, увімкнутий режим *DEBUG* і т.д.
- Вразливості на етапі експлуатації серверу та ПЗ на ньому. До цих вразливостей, наприклад, відносяться: – використання застарілого ПЗ; занадто прості паролі; зберігання конфіденційних даних та архівних копій на веб-сервері в загальному доступі, та в не шифрованому вигляді; наявність у доступі службових модулів та компонентів і т.і.

Завдання пошуку вразливостей 2-го і 3-го типів автоматизується досить непогано При цьому, завдання пошуку вразливостей етапу розробки набагато успішніше вирішується при застосуванні сканерів саме в ручному режимі. Однак, слід зауважити, що в межах даної роботи, особливості аналізу веб-додатків с залученням ручного методу не розглядається, так як метою статті є аналіз автоматизованих сканерів безпеки.

В автоматичному режимі роботи сканеру, від тестувальника вимагається тільки правильно налаштувати параметри його запуску: додати URL, логін / пароль користувача, обрати тип сканування, глибину пошуку, кількість одночасних сканувань та інші можливості. Типовими представниками даного класу є наступні інструменти: *OWASP ZAP; Nikto; Wapiti* та ін.

3 Тестування

В мережі Інтернет, на запит *«Безкоштовні сканери веб-додатків»*, отримаємо велику кількість прикладів реалізації сканерів. Деякі з них здатні вирішувати вузькоспеціалізовані задачі, як, наприклад, програми перебору паролів чи пошуку відкритих файлів і директорій (*т.з.* «*краулери*»), та ін. Але, серед всього цього розмаїття, є «повноцінні» системи, які здатні виконувати сканування більшою кількістю методів. Розглянемо деякі з них нижче [4-6].

Так, у дослідному тестуванні використовувалися наступні відомі безкоштовні сканери:

- *OWASP ZAP* – це безкоштовний інструмент з відкритим кодом для тестування на проникнення, який підтримується під егідою *OWASP*. *ZAP* розроблений спеціально для тестування веб-додатків і є досить гнучким, та здатним до розширеня. В тестуванні використовувався *ZAP* сканер в версії 2.10.0.
- *Wapiti* – сканер вразливостей веб-додатків, який дозволяє перевіряти безпеку веб-сайтів або веб-програм. Виконує сканування методом «чорного ящика», тобто скануючи веб-сторінки та шукаючи сценарії і форми, де він може вводити дані. До експерименту залучався *Wapiti* в версії 3.0.5.
- *Nikto* - це сканер веб-додатків з відкритим кодом, який виконує комплексні тести веб-додатків для кількох елементів, включаючи понад 6700 потенційно небезпечних файлів/програм. До тестів залучалась версія *Nikto* сканеру 2.1.6.

Тестуванню піддавалися три веб-додатку [7-9]: - *DVWA, Juice Shop* та *WebGoat.*

DVWA - це веб-програма *PHP/MySQL*, яка є надзвичайно вразливою. Метою *DVWA* є відпрацювання однієї з найпоширеніших веб-вразливостей, з різними складними рівнями, та зрозумілим інтерфейсом. Версія додатку на момент експериментів - 1.10.

Juice-Shop – сучасний небезпечний веб-додаток, котрий «написаний» на JavaScript. Додаток містить величезну кількість завдань різного рівня складності. Його з успіхом можна використовувати для демонстрацій, та для тренуваннь. *Juice Shop* охоплює вразливості

OWASP Top-10, разом з багатьма іншими уразливостями, які зустрічаються в реальних додатках. Версія додатку на момент експериментів – v.12.9.3.

WebGoat – це, навмисно, небезпечний додаток, який дозволяє розробникам перевіряти вразливості, які зазвичай зустрічаються в додатках на базі *Java*, котрі використовують загальні та/або популярні компоненти з відкритим кодом. Тестова версія - 8.2.2.

Завантаження кожного з дослідних веб-додатків, виконується у своєму контейнері:

docker run -d --name juice --net locnet --restart always -p 3000:3000 bkimminich/juice-shop &&
docker run -d --name dvwa --net locnet --restart always -p 8000:80 sagikazarmark/dvwa &&
docker run -d --name goat --net locnet --restart always -p 8080:8080 webgoat/goatandwolf && docker ps

На рис. 1 представлений скріншейвер *(Skr)*, якій демонструє, що веб-додаткі вже запущені і працюють.

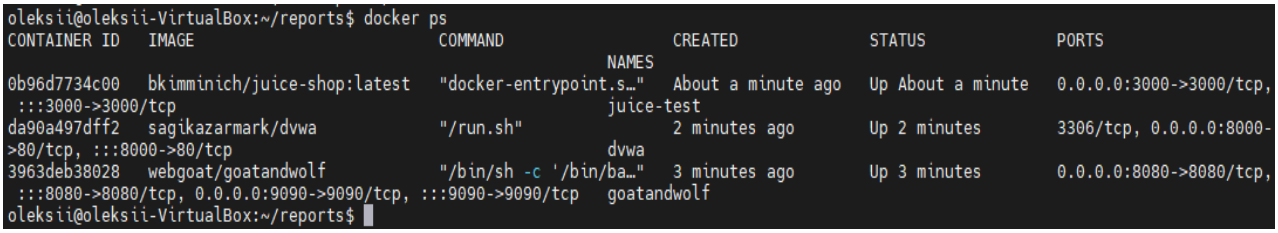


Рис. 1 – Scr працюючих веб-додатків

Визначимо їх адреси (рис. 2) командою «*docker inspect»,* для подальшого вказування їх у веб-сканерах.

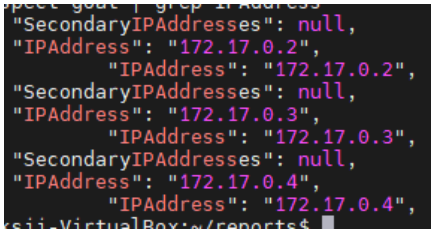


Рис. 2 – Визначення адрес працюючих веб-додатків

Враховуючи те, що обсяг статті доволі обмежений, виключимо розгляд сутності проведення перших двох етапів (*розвідування і атаки*), та зосередимось саме на розгляді найбільш важливої, для користувача частині – *Звіті*. Він є головною частиною, бо повинен зрозуміло та чітко надати відомості користувачу про знайдені у веб-додатку вразливості. При цьому, зовсім неважливо, як саме проводяться перші два етапу (розвідка і атаки), коли звіт буде не зрозумілим, та не містить потрібної суті. - Тому аналіз отриманих звітів є головним пунктом оцінки ефективності проведеного тестування.

Як вже було зазначено вище, пошук вразливостей проводився в 3-х тестових додатках: Juice-Shop, DVWA та WebGoat, які містять певні вразливості:

- *DVWA*: *XSS, SQLi, Insecure data, CSRF, Command injection, Misconfiguration.*
- *Juice-Shop: XSS, SQLi, Security Misconfiguration (SM), Sensitive Data Exposure (SDE), Vulnerable Components (VC), Cryptographic Issues (CI), Miscellaneous, Security through Obscurity (STO), Unvalidated Redirects (UR);*
- *WebGoat: XSS, SQLi, Sensitive Data Exposure, XXE, Vulnerable Components, Misconfiguration.*

Зазначимо пункти по яким порівнюються дані з відомостями звіту організації OWASP, «OWASP Top 10 2017 року» [1]. Варто зауважити, що є і нова версія, але для аналізу краще підходить стара, так як всі загрози, які були в ній, все ще залишаються актуальними.

OWASP Top 10, які можливо виявити:

- ін’єкції;
- міжсайтове виконання коду (*XSS*);

- розголошення конфіденційних даних (*Sensitive Data Exposure*);
- зовнішні XML сутності (*XXE*);
- невірне налаштування параметрів безпеки (*Security Misconfiguration*);
- використання компонентів з відомими вразливостями (*Vulnerable Components*).

Надамо стислий опис знайдених вразливостей для кожного веб-додатку.

OWASP ZAP:

- у *Juice-Shop* сканером OWASP ZAP було знайдено чотири невстановлених хедера: *Content Security Policy (CSP), X-Frame-Options, Deprecated Feature Policy, X-Content-Type-Options*; які призводять доступ *XML* файлі – а це зразу *XXE*. Інші загрози через неправильне налаштування серверу у кількості двох: *Bypassing 403, Cross-Domain Misconfiguration*. Виявлені загрози, які можуть призвести до розкриття конфіденційних даних у кількості п'ятьох: *Backup File Disclosure, Session ID in URL Rewrite, Source Code Disclosure – Java and PHP, Web Cache Deception*. Окрім розголошення даних *Source Code Disclosure – Java and PHP* призводить до розголошення версії продукту, призводячи пошук експлоїта набагато простіше. Також було знайдено одну *SQL* ін'єкцію.
- у *DVWA* було знайдено також невстановлені хедери у кількості три, з доступом до sitemap.xml. Виявлена *Application Error Disclosure* вразливість, яка при виникненні помилки, може призвести до розголошенню конфіденційної інформації. І відкритий доступ до директорій (*Directory Browsing, Directory Browsing - Apache 2*), що також може привести до розголошення даних.
- у *WebGoat* було знайдено один невстановлений хедер (*CSP*), вразливість високого рівня *Anti-CSRF* та невстановлений для нього токен.

Отже, у підсумку детекту для 3-х веб-додатків, маємо:

- *ін'єкцій – 1;*
- *міжсайтового виконання коду (XSS) -0;*
- *зовнішніх XML сутностей (XXE) – 2;*
- *загроз що сприяють розголошенні конфіденційних даних (SDE) – 7;*
- *параметрів безпеки з неправильним налаштуванням (SM) – 10;*
- *використання компонентів з вже відомими вразливостями (VC) – 2.*

По сканеру Nikto отримані наступні дані:

- *Juice-Shop* – сканер видав 167 потенційно небезпечні файли, але тільки один є реальним;
- *DVWA* – знайдено три хедери, та три відбитки ПЗ із застарілими версіями. А також, одну *CVE*, з отриманням усієї файлової структури;
- *WebGoat* – знайдено 1 невстановлений хедер, для сесійних *cookie*.

Отже, було знайдено:

- *ін'єкцій – 0;*
- *XSS – 0;*
- *XXE – 0;*
- *SDE – 2;*
- *SM – 4;*
- *VC – 0.*

Для сканера Wapiti отримані такі дані:

- *Juice-Shop* – детектовано велику кількість доступних посилань на файли *Backup*, які є помилковим викидом. Однак, при цьому є інша категорія - це потенційно небезпечні файли, які можуть призвести до розкриття конфіденційної інформації, та три невстановлених хедера і чотири відбитки версій використовуваних програм та фреймворків, що може призвести до можливості експлуатації експлойту.

- *DVWA* – знайдено п’ять потенційно небезпечних файлів та п’ять невстановлених хедера і три відбитки ПЗ, які розкривають версії використовуваних продуктів.
- *WebGoat* - знайдено чотири невстановлених хедера, два з яких відповідають за cookie, та п’ять відбитки версій використовуваних програм і фреймворків, що може призвести до експлуатації експлойту. Виявлена вразливість, типу *Internal Server Error*, яка може призвести до розголошенню конфіденційної інформації.

Таким чином, в цілому, було знайдено:

- ін’єкцій – *0;*
- *XSS – 0;*
- *XXE – 0;*
- *SDE – 5+;*
- *SM – 13;*
- *VC – 11.*

Так як жоден з тестових сканерів не знайшов вразливості типу *XSS*, то вилучимо її та побудуємо результуючу таблицю, що містить перелік відповідних загроз вразливостей.

Таблиця 1 – Зведений перелік загроз

Сканери	**Типи вразливостей**					
	SQLi	***XXE***	***SDE***	***SM***	***VC***	*Додаткові*
OWASP ZAP	1	2	7	11	2	1 - *CSRF*
Nikto	0	0	0	2	2	
Wapiti	0	0	5+	13	11	

Таким чином, як випливає з даних таблиці, за результатами сканування було знайдено велику кількість вразливостей у налаштуваннях сервера та визначено велику кількість потенційно небезпечних файлів.

4 Використані команди налаштувань та форми звітів

В загальному випадку, головною ідеєю використовуваних налаштувань веб-сканерів повинно бути отримання якомога більш інформативних даних їх звітів *(т.з. лог-файлів)*. При цьому, для більш детального огляду можливо скористатися наступним веб-ресурсом [10].

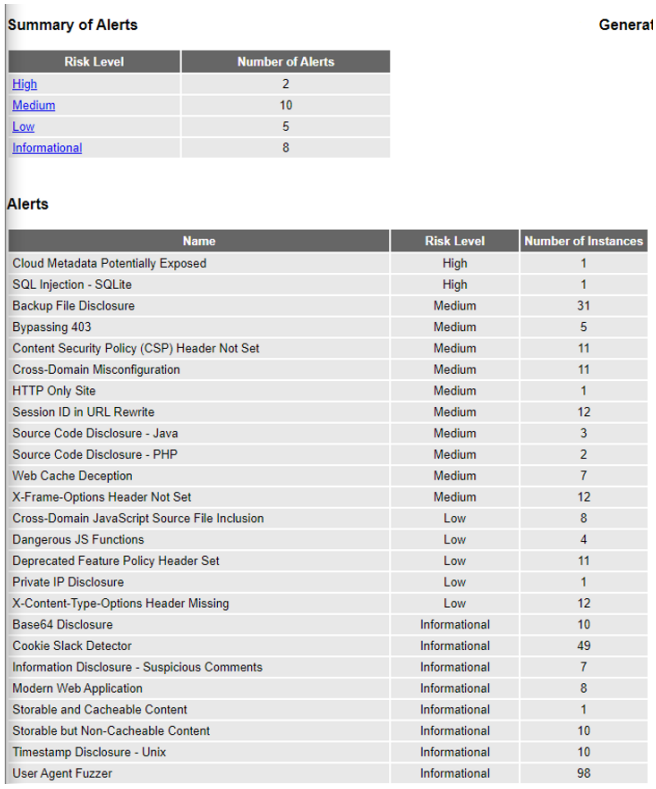
Для отримання сукупності відповідних звітів від сканеру *OWASP ZAP*, потрібно виконати наступні команди:

```
docker run --rm -d --name owasp_scanner_j -v $(pwd):/zap/wrk/:rw -t owasp/zap2docker-stable zap-full-scan.py -t http://172.17.0.2:3000/ -g gen.conf -m 10 -T 60 -I -j -r report_owasp_j.html &&
docker run --rm -d --name owasp_scanner_d -v $(pwd):/zap/wrk/:rw -t owasp/zap2docker-stable zap-full-scan.py -t http://172.17.0.3:80/vulnerabilities/ -g gen.conf -m 10 -T 60 -I -j -r report_owasp_d.html &&
docker run --rm -d --name owasp_scanner_g -v $(pwd):/zap/wrk/:rw -t owasp/zap2docker-stable zap-full-scan.py -t http://172.17.0.4:8080/WebGoat/ -g gen.conf -m 10 -T 60 -I -j -r report_owasp_g.html && docker ps
```

Нижче, на рис. 3, представлені вразливості, які були знайдені сканером ***ZAP*** (відповідно, для: - *Juice shop, DVWA, WebGoat*).

Наступний звіт від сканеру ***Nikto***, для котрого потрібно виконати наступні команди:

```
docker run --rm -d --name nikto_scanner_j -v $(pwd):/tmp olexii21/nikto_scanner -url http://172.17.0.2:3000/ -Plugins "@@ALL" -id "test@com.com:password" -Format html -output tmp/nikto_report_j.html &&
docker run --rm -d --name nikto_scanner_d -v $(pwd):/tmp olexii21/nikto_scanner -url http://172.17.0.3:80/vulnerabilities/ -Plugins "@@ALL" -id "admin:password" -Format html -output tmp/nikto_report_d.html &&
docker run --rm -d --name nikto_scanner_g -v $(pwd):/tmp olexii21/nikto_scanner -url http://172.17.0.4:8080/WebGoat/ -Plugins "@@ALL" -id "admin:password" -Format html -output tmp/nikto_report_g.html && docker ps
```



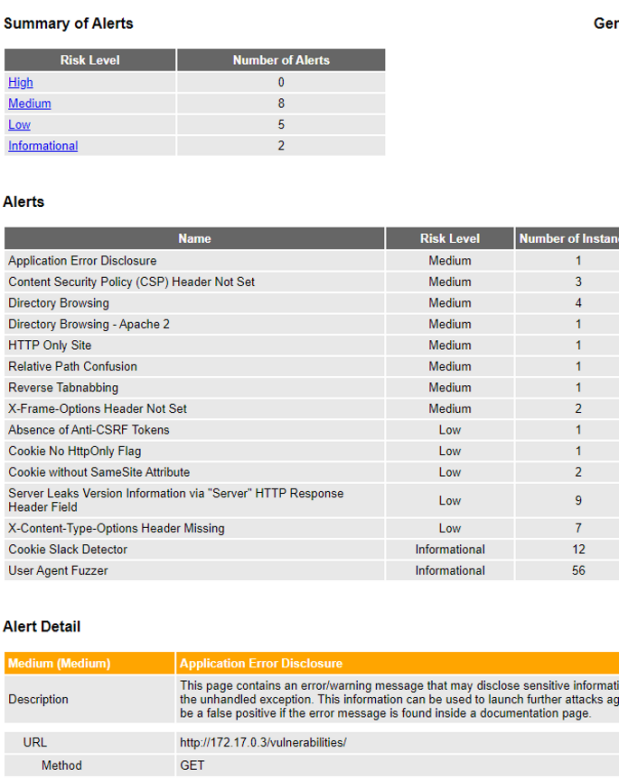
Summary of Alerts

Risk Level	Number of Alerts
High	2
Medium	10
Low	5
Informational	8

Alerts

Name	Risk Level	Number of Instances
Cloud Metadata Potentially Exposed	High	1
SQL Injection - SQLite	High	1
Backup File Disclosure	Medium	31
Bypassing 403	Medium	5
Content Security Policy (CSP) Header Not Set	Medium	11
Cross-Domain Misconfiguration	Medium	11
HTTP Only Site	Medium	1
Session ID in URL Rewrite	Medium	12
Source Code Disclosure - Java	Medium	3
Source Code Disclosure - PHP	Medium	2
Web Cache Deception	Medium	7
X-Frame-Options Header Not Set	Medium	12
Cross-Domain JavaScript Source File Inclusion	Low	8
Dangerous JS Functions	Low	4
Deprecated Feature Policy Header Set	Low	11
Private IP Disclosure	Low	1
X-Content-Type-Options Header Missing	Low	12
Base64 Disclosure	Informational	10
Cookie Slack Detector	Informational	49
Information Disclosure - Suspicious Comments	Informational	7
Modern Web Application	Informational	8
Storable and Cacheable Content	Informational	1
Storable but Non-Cacheable Content	Informational	10
Timestamp Disclosure - Unix	Informational	10
User Agent Fuzzer	Informational	98

а) Juice shop



General Summary of Alerts Ger

Risk Level	Number of Alerts
High	0
Medium	8
Low	5
Informational	2

Alerts

Name	Risk Level	Number of Instan
Application Error Disclosure	Medium	1
Content Security Policy (CSP) Header Not Set	Medium	3
Directory Browsing	Medium	4
Directory Browsing - Apache 2	Medium	1
HTTP Only Site	Medium	1
Relative Path Confusion	Medium	1
Reverse Tabnabbing	Medium	1
X-Frame-Options Header Not Set	Medium	2
Absence of Anti-CSRF Tokens	Low	1
Cookie No HttpOnly Flag	Low	1
Cookie without SameSite Attribute	Low	2
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	9
X-Content-Type-Options Header Missing	Low	7
Cookie Slack Detector	Informational	12
User Agent Fuzzer	Informational	56

Alert Detail

Medium (Medium)	Application Error Disclosure
Description	This page contains an error/warning message that may disclose sensitive informati the unhandled exception. This information can be used to launch further attacks ag be a false positive if the error message is found inside a documentation page.
URL	http://172.17.0.3/vulnerabilities/
Method	GET

б) DVWA

ZAP Scanning Report

Summary of Alerts **Generated on Fri, 1 Oct 2021 14:44:08**

Risk Level	Number of Alerts
High	1
Medium	2
Low	4
Informational	1

Alerts

Name	Risk Level	Number of Instances
Anti-CSRF Tokens Check	High	1
Content Security Policy (CSP) Header Not Set	Medium	1
HTTP Only Site	Medium	1
Absence of Anti-CSRF Tokens	Low	1
Cookie No HttpOnly Flag	Low	1
Cookie Slack Detector	Low	3
Cookie without SameSite Attribute	Low	1
User Agent Fuzzer	Informational	14

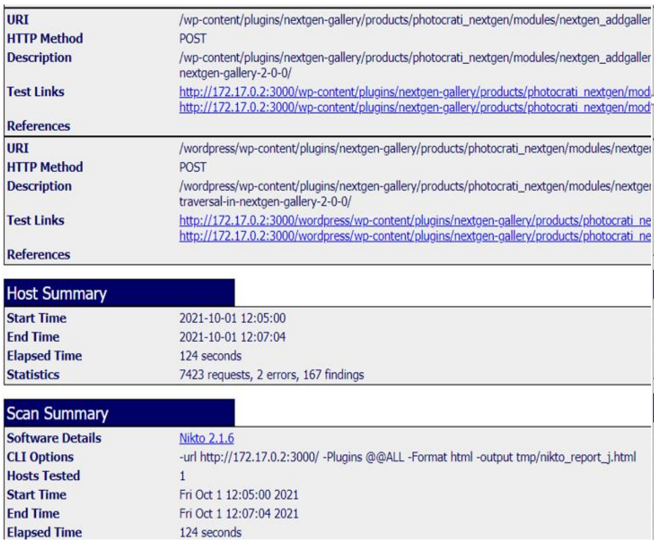
Alert Detail

High (Medium)	Anti-CSRF Tokens Check
Description	A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
URL	http://3.66.163.55:8080/WebGoat/login
Method	GET

в) WebGoat

Рис. 3 – Звіт для *ZAP*

В звітах сканеру ***Nikto*** (див. рис. 4) не має підсумкової таблиці, з усіма знайденими вразливостями, тому відображено тільки кінець з відповідною статистикою.

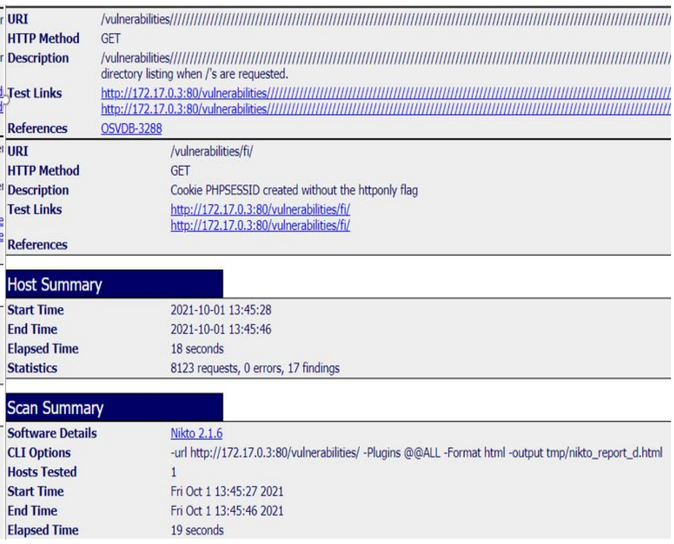


URI	/wp-content/plugins/nextgen-gallery/products/photocrati_nextgen/modules/nextgen_addgaller
HTTP Method	POST
Description	/wp-content/plugins/nextgen-gallery/products/photocrati_nextgen/modules/nextgen_addgaller nextgen-gallery-2-0-0/
Test Links	http://172.17.0.2:3000/wp-content/plugins/nextgen-gallery/products/photocrati_nextgen/mod http://172.17.0.2:3000/wp-content/plugins/nextgen-gallery/products/photocrati_nextgen/mod
References	
URI	/wordpress/wp-content/plugins/nextgen-gallery/products/photocrati_nextgen/modules/nextge
HTTP Method	POST
Description	/wordpress/wp-content/plugins/nextgen-gallery/products/photocrati_nextgen/modules/nextge traversal-in-nextgen-gallery-2-0-0/
Test Links	http://172.17.0.2:3000/wordpress/wp-content/plugins/nextgen-gallery/products/photocrati_ne http://172.17.0.2:3000/wordpress/wp-content/plugins/nextgen-gallery/products/photocrati_ne
References	

Host Summary	
Start Time	2021-10-01 12:05:00
End Time	2021-10-01 12:07:04
Elapsed Time	124 seconds
Statistics	7423 requests, 2 errors, 167 findings

Scan Summary	
Software Details	Nikto 2.1.6
CLI Options	-url http://172.17.0.2:3000/ -Plugins @@ALL -Format html -output tmp/nikto_report_j.html
Hosts Tested	1
Start Time	Fri Oct 1 12:05:00 2021
End Time	Fri Oct 1 12:07:04 2021
Elapsed Time	124 seconds

а) Juice shop

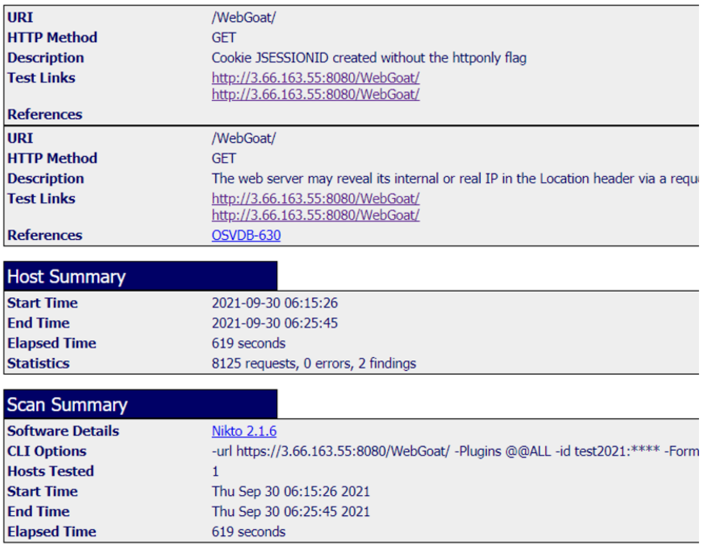


URI	/vulnerabilities//
HTTP Method	GET
Description	/vulnerabilities// directory listing when /'s are requested.
Test Links	http://172.17.0.3:80/vulnerabilities/// http://172.17.0.3:80/vulnerabilities///
References	OSVDB-3288
URI	/vulnerabilities/fi/
HTTP Method	GET
Description	Cookie PHPSESSID created without the httponly flag
Test Links	http://172.17.0.3:80/vulnerabilities/fi/ http://172.17.0.3:80/vulnerabilities/fi/
References	

Host Summary	
Start Time	2021-10-01 13:45:28
End Time	2021-10-01 13:45:46
Elapsed Time	18 seconds
Statistics	8123 requests, 0 errors, 17 findings

Scan Summary	
Software Details	Nikto 2.1.6
CLI Options	-url http://172.17.0.3:80/vulnerabilities/ -Plugins @@ALL -Format html -output tmp/nikto_report_d.html
Hosts Tested	1
Start Time	Fri Oct 1 13:45:27 2021
End Time	Fri Oct 1 13:45:46 2021
Elapsed Time	19 seconds

б) DVWA



URI	/WebGoat/
HTTP Method	GET
Description	Cookie JSESSIONID created without the httponly flag
Test Links	http://3.66.163.55:8080/WebGoat/ http://3.66.163.55:8080/WebGoat/
References	
URI	/WebGoat/
HTTP Method	GET
Description	The web server may reveal its internal or real IP in the Location header via a requ
Test Links	http://3.66.163.55:8080/WebGoat/ http://3.66.163.55:8080/WebGoat/
References	OSVDB-630

Host Summary	
Start Time	2021-09-30 06:15:26
End Time	2021-09-30 06:25:45
Elapsed Time	619 seconds
Statistics	8125 requests, 0 errors, 2 findings

Scan Summary	
Software Details	Nikto 2.1.6
CLI Options	-url https://3.66.163.55:8080/WebGoat/ -Plugins @@ALL -id test2021:**** -Form
Hosts Tested	1
Start Time	Thu Sep 30 06:15:26 2021
End Time	Thu Sep 30 06:25:45 2021
Elapsed Time	619 seconds

в) WebGoat

Рис. 4 – Звіт для *Nikto*

Наступний звіт від сканеру ***Wapiti***, для котрого потрібно виконати наступні команди:

docker run --rm -d --name wapiti_scanner_j -v $(pwd):/tmp olexii21/wapiti_scanner wapiti -u http://172.17.0.2:3000/ --scope folder -m "all" -l 2 -d 5 -f html -o tmp/wapiti_report_j &&

docker run --rm -d --name wapiti_scanner_d -v $(pwd):/tmp olexii21/wapiti_scanner wapiti -u http://172.17.0.3:80/vulnerabilities/ --scope folder -m "all" -a "admin%password" -l 2 -d 5 -f html -o tmp/wapiti_report_d &&

docker run --rm -d --name wapiti_scanner_g -v $(pwd):/tmp olexii21/wapiti_scanner wapiti -u http://172.17.0.4:8080/WebGoat/ --scope folder -m "all" -a "admin21%password" -l 2 -d 5 -f html -o tmp/wapiti_report_g && docker ps

Нижче, на рис. 5, представлені вразливості, які були знайдені сканером ***Wapiti*** (відповідно, для: - *Juice shop*, *DVWA* та *WebGoat*).

5 Висновки

1. Як свідчать результати тестів, ефективність розглянутих сканерів, доволі низька, але попри цього сформовані ними звіти, в більшій мірі допоможуть дізнатися, чи правильно налаштований сервер, та видно чи ні директорії і файли в них, які можуть бути потенційно

Wapiti vulnerability report
Target: http://172.17.0.2:3000/
Date of the scan: Fri, 01 Oct 2021 13:44:54 +0000. Scope of the scan: folder

Summary

Category	Number of vulnerabilities found
Backup file	2372
Weak credentials	0
CRLF Injection	0
Content Security Policy Configuration	1
Cross Site Request Forgery	0
Potentially dangerous file	141
Command execution	0
Path Traversal	0
Fingerprint web application framework	1
Fingerprint web server	0
Htaccess Bypass	0
HTTP Secure Headers	2
HttpOnly Flag cookie	0
Open Redirect	0
Secure Flag cookie	0
SQL Injection	0
Server Side Request Forgery	0
Subdomain takeover	0
Blind SQL Injection	0
Cross Site Scripting	0
XML External Entity	0
Internal Server Error	0
Resource consumption	0
Fingerprint web technology	3
HTTP Methods	0

а) Juice shop

Wapiti vulnerability report
Target: http://172.17.0.3:80/vulnerabilities/
Date of the scan: Fri, 01 Oct 2021 13:42:40 +0000. Scope of the scan: folder

Summary

Category	Number of vulnerabilities found
Backup file	0
Weak credentials	0
CRLF Injection	0
Content Security Policy Configuration	1
Cross Site Request Forgery	0
Potentially dangerous file	5
Command execution	0
Path Traversal	0
Fingerprint web application framework	0
Fingerprint web server	1
Htaccess Bypass	0
HTTP Secure Headers	4
HttpOnly Flag cookie	0
Open Redirect	0
Secure Flag cookie	0
SQL Injection	0
Server Side Request Forgery	0
Subdomain takeover	0
Blind SQL Injection	0
Cross Site Scripting	0
XML External Entity	0
Internal Server Error	0
Resource consumption	0
Fingerprint web technology	2
HTTP Methods	0

б) DVWA

Wapiti vulnerability report
Target: http://172.17.0.4:8080/WebGoat/
Date of the scan: Fri, 01 Oct 2021 13:43:47 +0000. Scope of the scan: folder

Summary

Category	Number of vulnerabilities found
Backup file	0
Weak credentials	0
CRLF Injection	0
Content Security Policy Configuration	1
Cross Site Request Forgery	0
Potentially dangerous file	0
Command execution	0
Path Traversal	0
Fingerprint web application framework	0
Fingerprint web server	0
Htaccess Bypass	0
HTTP Secure Headers	1
HttpOnly Flag cookie	1
Open Redirect	0
Secure Flag cookie	1
SQL Injection	0
Server Side Request Forgery	0
Subdomain takeover	0
Blind SQL Injection	0
Cross Site Scripting	0
XML External Entity	0
Internal Server Error	4
Resource consumption	0
Fingerprint web technology	5
HTTP Methods	0

в) WebGoat

Рис. 5 – Звіт для *Wapiti*

небезпечними, з боку конфіденційності. При цьому, вразливості типу *SQLi*, *XSS* та інші, де треба вводити різні послідовності для перевірки, залишаються занадто «важкими» для безкоштовних автоматизованих сканерів безпеки. - Т.ч., для пошуку таких загроз краще використовувати спеціалізовані сканери для кожного типу вразливостей.

2. Функціональне порівняння сканерів, в цілому, зробити нескладно. - Маючи список функціональних критеріїв, можна отримати оцінку відповідності цим критеріям для кожного інструменту зі списку, на вибір.

3. З усіх сканері, які були підвергнути дослідному тестуванню, саме *OWASP ZAP*, об'єктивно, є найпотужнішим. Насамперед, це зумовлено тим, що це є великий та відкритий проект, який має досить представницьке ком'юніті розробників, котрі постійно вдосконюють сканер, та є прихильниками вільного розповсюдження відповідного ПЗ.

4. Сканери безпеки веб-додатків не забезпечують гарантованого захисту та виявлення усіх можливих проблем безпеки. Причина очевидна, універсального рішення не існує, перш за все, в наслідок постійної та швидкої еволюції небезпечного програмного забезпечення. В наслідок цього, ПЗ для оцінки безпеки веб-додатків, слід розглядати не більш, як засіб здійснення оперативного початкового пошуку шаблонних проблем, в багаторівневому механізмі аналізу проблем безпеки сучасних веб-додатків.

Посилання

[1] OWASP Top 10 – 2017. (2017). Вилучено з https://owasp.org/www-project-top-ten/
[2] OWASP. Руководство по тестированию веб-безопасности, (2020).
[3] Петухов Андрей. Как выбрать сканер уязвимостей веб-приложений? (2015). Вилучено з https://www.securitylab.ru/blog/personal/andrepetukhov/143697.php
[4] ZAP - Full Scan. (2021). Вилучено з https://www.zaproxy.org/docs/docker/full-scan/
[5] The web-application vulnerability scanner. (2021). Вилучено з https://wapiti.sourceforge.io/
[6] Nikto: A Practical Website Vulnerability Scanner. (2021). Вилучено з https://securitytrails.com/blog/nikto-website-vulnerability-scanner
[7] Damn Vulnerable Web Application Docker container (2018). Вилучено з https://github.com/opsxcq/docker-vulnerable-dvwa
[8] OWASP Juice Shop. (2021). Вилучено з https://github.com/juice-shop/juice-shop
[9] WebGoat 8: A deliberately insecure Web Application. (2021). Вилучено з https://github.com/WebGoat/WebGoat
[10] Scan Reports. (2021). Вилучено з https://github.com/G4rr/reports

Reviewer: Vyacheslav Kalashnikov, Doctor of Sciences (Physics and Mathematics), Full Prof., Department of Systems and Industrial Engineering, Campus Monterrey, Monterrey, Mexico.
E-mail: kalash@itesm.mx

Received on October 2021.

Authors:
Dmytro Ivanenko, Ph.D., Senior Lecturer, Department of Security of Information Systems and Technologies, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine
E-mail: i8o.dima@gmail.com

Oleksii Pryshchepa, CSD Student, Department of Security of Information Systems and Technologies, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: pryshchepa2020kb51@student.karazin.ua

Evaluation of the efficiency of web-application safety scanners.

Abstract. The level of security of web applications is constantly growing every year, but new ratings of the most common security threats indicate that the problem of ensuring their security is very relevant and constantly changing. Therefore, it is essential to understand the importance of using automatic security scans of web applications and objectively assess their real effectiveness. The paper considers the process of testing web applications for vulnerabilities (and examples of their detection), using free web crawlers (with open-source) by the "black box" method. In this case, scanners interact with applications in the same way as a typical user through a web interface, through the HTTP protocol. The main purpose of the testing is to compare several open-source scanners and determine their effectiveness. It is underlined that it is impossible to evaluate all the indicators of scanners due to the existence of many factors. - Therefore, in the framework of this work, all judgments and conclusions were made only based on an analysis of the received reports of each test scanner. This article provides information about the individual parameters and the number of vulnerabilities found. The testing results indicate that the practice of using only one scanner is not effective, so you need to use

several different solutions at once when testing. This will allow you to get more objective results in terms of detecting both already known security threats and finding new vulnerabilities (with their addition to the final report). The work will be useful to those interested in assessing the security state of modern web applications.

Keywords*:* Web application vulnerabilities; vulnerability scanners; efficient scanners; web application security testing; pentesting.

Рецензент: Вячеслав Калашников, д.ф.-м.н., проф., Технологический университет Монтеррея, Монтеррей, Мексика.
E-mail: kalash@itesm.mx

Поступила: Октябрь 2021.

Авторы:
Дмитрий Иваненко, к.т.н., доцент кафедры Безопасности информационных систем и технологий, Харьковский национальный университет имени В. Н. Каразина, Харьков, Украина.
E-mail: i8o.dima@gmail.com
Алексей Прищепа, студент факультета компьютерных наук, Харьковский национальный университет имени В. Н. Каразина, Харьков, Украина.
E-mail: pryshchepa2020kb51@student.karazin.ua

Оценивание эффективности сканеров веб приложений.

Аннотация. Уровень защищенности веб-приложений с каждым годом постоянно растет, однако новые рейтинги наиболее распространенных угроз безопасности, свидетельствуют о том, что проблема обеспечения их защищенности является очень актуальной и постоянно изменяющейся. Поэтому крайне актуально понимать важность использования автоматических сканеров безопасности веб-приложений и уметь объективно оценивать их реальную эффективность. В работе рассмотрен процесс тестирования веб-приложений на наличие уязвимостей (и примеры их обнаружения), с использованием бесплатных веб-сканеров (с открытым кодом) методом «черного ящика». То есть, в данном случае, сканеры взаимодействуют с приложениями так же, как и типичный пользователь через веб-интерфейс, посредством протокола HTTP. Главной целью проведенных тестирований является сравнение нескольких сканеров с открытым кодом и определение их эффективности. Подчеркнуто, что оценить все показатели сканеров невозможно, вследствие существования многих факторов. - Поэтому, в рамках данной работы, все суждения и выводы были сделаны только на основе анализа полученных отчетов каждого тестового сканера. В статье приведены сведения относительно отдельных параметров и количества найденных уязвимостей. Полученные результаты тестирований свидетельствуют о том, что практика использования только одного сканера является мало эффективной, поэтому при тестировании нужно использовать сразу несколько различных решений. Это позволит получить более объективные результаты, в части детектирования, как уже известных угроз безопасности, так и нахождение новых уязвимостей (с их добавлением в конечный отчет). Работа будет полезна тем, кто интересуется вопросами оценки состояния безопасности современных веб-приложений.

Ключевые слова: уязвимости веб приложений; сканеры уязвимостей; эффективность сканеров; тестирование безопасности веб приложений; пентестинг.

УДК 004.4 DOI: 10.26565/2519-2310-2021-2-02

ПРОГРАМУВАННЯ МІКРОКОНТРОЛЕРІВ З ВИКОРИСТАННЯМ РІШЕНЬ SIMPLICITY STUDIO, НА МОВІ ASSEMBLER Ax51

Ольга Мелкозьорова, Ірина Гальцева

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
olha.melkozerova@karazin.ua, irina.galceva@karazin.ua

Рецензент: Володимир Хома, д.т.н., проф., Опольський політехнічний Університет, Ополе, Польща
xoma@wp.pl

Надійшла: Вересень 2021.

Анотація: *У статті наведено загальний огляд існуючих можливостей програмування на мові Assembler Ax51 мікроконтролера C8051F340 у середовищі розробки Simplicity Studio. Робота є керівництвом для практичного застосування мови Assembler. У керівництвах [1-2] дуже багато інформації, стосовно мови Assembler та опису мікроконтролерів C8051F340, але ця інформація відокремлена одна від одної. Більш того, замало пояснень, щодо практичного використання, разом зі середовищем розробки Simplicity Studio [3]. Матеріал містить пояснення, стосовно компонентів, на які поділяються команди, а саме директиви, інструкції та елементи контролю. Наведено стислий опис регістрів, директив, елементів контролю, інструкцій, роботи зі змінними. Є приклади використання, у вигляді коду, деяких логічних та арифметичних інструкцій, з поясненням результатів виконання, котрі можна побачити у скріншотах зі значеннями регістрів після виконання команд. Додано приклади роботи зі змінними, результати яких фіксуються у пам'яті мікроконтролера.*

Ключові слова*: мікроконтролер; директиви; інструкції; елементи контролю; регістри; Assembler.*

1 Вступ

Існуючи керівництва з програмування на мови *Assembler Ax51* [1-2], містять в собі досить багато відомостей, стосовно цієї мови і особливостей мікроконтролерів *mod. C8051F340,* проте всю цю інформацію досить важко сприймати, як єдине та гармонійно взаємопов'язане ціле. З метою нівелювання зазначеного змістового розриву, має сенс провести загальний огляд існуючих можливостей програмування на мові *Assembler Ax51* з використанням існуючих можливостей середовища розробки *Simplicity Studio.*

2 Основна частина

2.1 Середовище розробки

Для вивчення прийомів і можливостей програмування в статті використовувалося середовище розробки *Simplicity Studio ver. SV5.2.0.0 2021* [3] (*див. рис.1*).

Рис.1 – Використана версія від студії *Simplicity Studio*

2.2 Опис регістрів мови Assembler

*Assembler Ax*51 визначає та використовує імена регістрів. Ці імена використовуються для доступу до регістрів процесору. У таблиці 1 наведено назви деяких регістрів, які, подальше, використовуються у прикладах, та стислий опис до них. При роботі у студії *Simlicity Studio* регістри, їх стислий опис та зміст наведено у вигляді таблиці (*див. рис. 3.1, де: a –*

шістнадцятирічна система числення; б – десятинний формат). Їх можна спостерігати після запуску відповідної програми.

Таблиця 1 – Опис регістрів та їх призначення

Регістр	*Опис*
A	Акумулятор. *Використовується з багатьма операціями, такими як, помноження, поділ, перенесення «із» та «до» зовнішньої пам'яті, логічні операції.*
DPTR	Регістр **Data Pointer**. *16-бітний показник, який використовує для адресації даних у пам'яті коду.*
PC	16-бітний лічильник. *Містить адресу інструкції, яка має бути виконана.*
C	Індикатор статусу операцій, які генерують біт переносу. *Також використовується операціями, яким потрібен додатковий біт.*
AB	Пара регістрів «**A**» та «**B**». *Використовується у інструкціях MUL та DIV.*
R0-R7	Вісім 8-бітних регістрів у поточному активному банку регістрів. *Всього доступні 4 банку регістрів.*
AR0-AR7	Абсолютні адреси даних від **R0** до **R7** у поточному банку регістрів. *Абсолютні значення адрес змінюються в залежності від банку регістрів, який буде обрано у поточний момент часу.*

2.3 Компоненти мови Assembler

В цілому, програма на *Assembler* складається з однієї або декількох ***компонент*** (*statements*). Ці компоненти мають:

- *директиви* (*directives*);
- *елементи управління* (*controls*);
- *інструкції* (*instructions or mnemonics*).

Директиви показують, як упорядкувати інструкції мови *Assembler*. Також директиви забезпечують шлях, щоб визначити константи програми та простір для змінних.

У мові *Assembler Ax51* є декілька директив, які дозволяють:

- визначити символьні значення;
- визначити ініціалізацію зберігання;
- контролювати розташування коду.

Директиви не можна переплутати з інструкціями, так як вони не виробляють виконуваний код, за винятком директив **DB**, **DW**, **DD**. Ці три директиви дозволяють:

- змінювати стан мови;
- визначати символи користувача;
- додавати інформацію до об'єктних файлів.

В якості приклада, в таблиці 2 наведено опис деяких характерних ***директив*** *Assembler*.

Таблиця 2 – Опис директив

Директива	Формат	Опис
BIT	symbol BIT bit_address	*Визначає бітову адресу в бітовому просторі даних*
BSEG	BSEG [absolute address]	*Визначає абсолютний сегмент з бітовим адресним простором*
CODE	symbol CODE code_address	*Призначає символьне ім'я спеціальної адреси у кодовому просторі*
DATA	symbol DATA data_address	*Призначає символьне ім'я до спеціальної адреси даних*
DB	[label:] DB expression [,expression]…	*Генерує список байтових значень*

Елементи управління керують операціями.

Умовні елементи управління (наприклад, **IF**, **ELSE**, **ENDIF** та **ELSEIF**) забезпечують набір операторів, які можна використовувати, щоб *увімкнути* або *вимкнути* частину програми з виконання. Наприклад, після виконання коду програми, котрий наведено нижче, маємо наступний результат: R0 105, R1 100.

```
VAR1 SET 105;
  MOV R0, #VAR1; load variable into the R0
  IF VAR1<200
    MOV R1, #100; load 100 into the R0
  ENDIF
```

Інструкції визначають програмний код, Assembler переводить інструкції у машинний код та зберігає їх у об'єктні файли. Інструкція має наступний вигляд:

[label:] mnemonic [operand] [, operand] [, operand] [; comment] ,

де, мітка (label) – місце (*адреса*) інструкції в програмі.

Операнди мови *Assembler* можуть бути із різними типами адресації (див Табл. 3).

Таблиця 3 – Операнди *Assembler*

Операнд	Опис
Immediate data	*Символи або константи, які використовуються як чисельне значення.*
Direct bit address	*Символи або константи, які є посиланням на бітовий адрес.*
Program addresses	*Символи або константи, які є посиланням на кодовий адрес.*
Indirect addresses	*Непряме посилання до пам'яті, опціонально зі зміщенням.*
Special assembler symbol	*Назва регістрів.*

Приклад програми з використанням інструкцій на мові програмування *Assembler*:

```
  mov R6, #01;
  mov R5, #01;
Loop0:  djnz  R5, Loop1;
  mov R0, #01;
Loop1:  djnz  R6, Loop0;
  mov R6, #01;
Loop2:  mov   R4, #90;
```

Після цього регістри, які задіяні у програмі, мають містить наступне значення:
R0 1, R4 90, R5 0, R6 1.

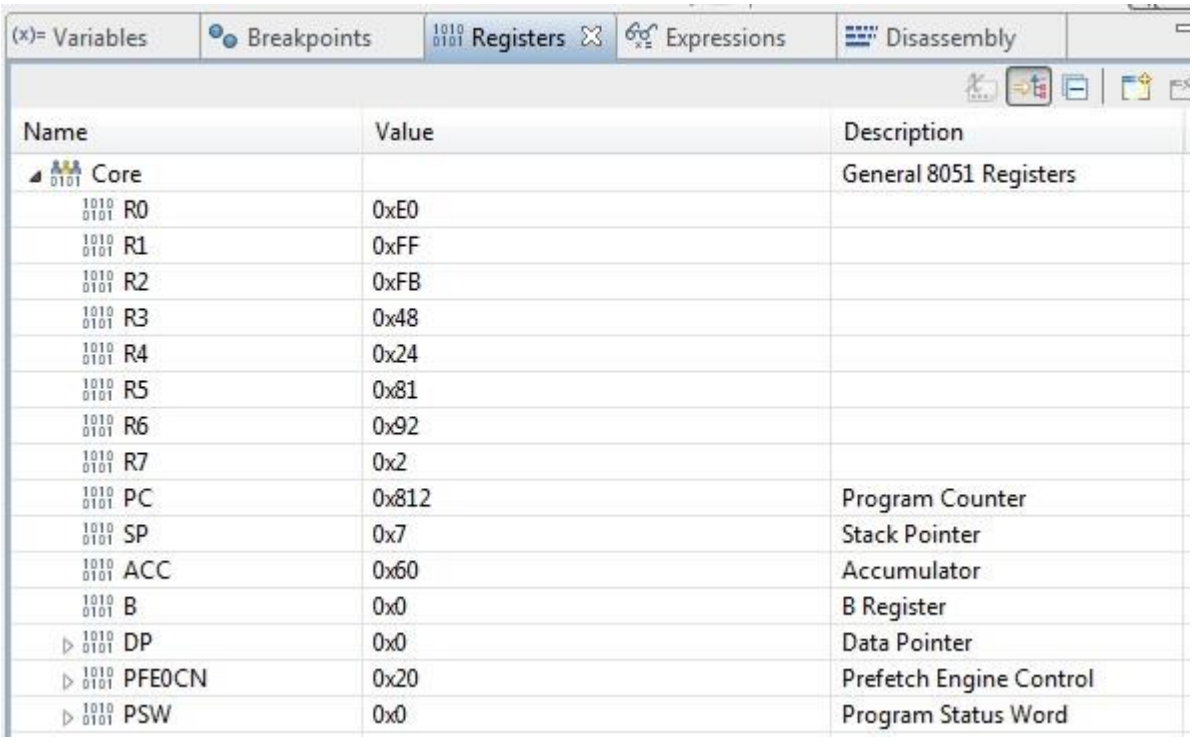
Операнди із прямою адресацією – це чисельні вирази, які кодуються, як частини інструкції. Такі значення використовуються у інструкціях, щоб змінити зміст регістрів або розташування пам'яті. Перед довільним виразом повинен бути символ «#», який використовується, як операнд з прямою адресацією. Приклад використання прямої адресації наведено нижче, а поруч з кодом наведено відповідні коментарі.

У значенні регістра **A** (*ACC accumulator*) можна побачити значення 60 (*див. рис 2а, де дані наведено в шістнадцятирічній системі числення*) або 96 (*рис. 2б, де дані наведено в десятинній системі числення*). 224= 11100000b, 128=10000000b.
$11100000 \oplus 10000000 = 01100000$ (*тут* $\oplus$ - *операція XOR*). $01100000=60h=96_{10}$.

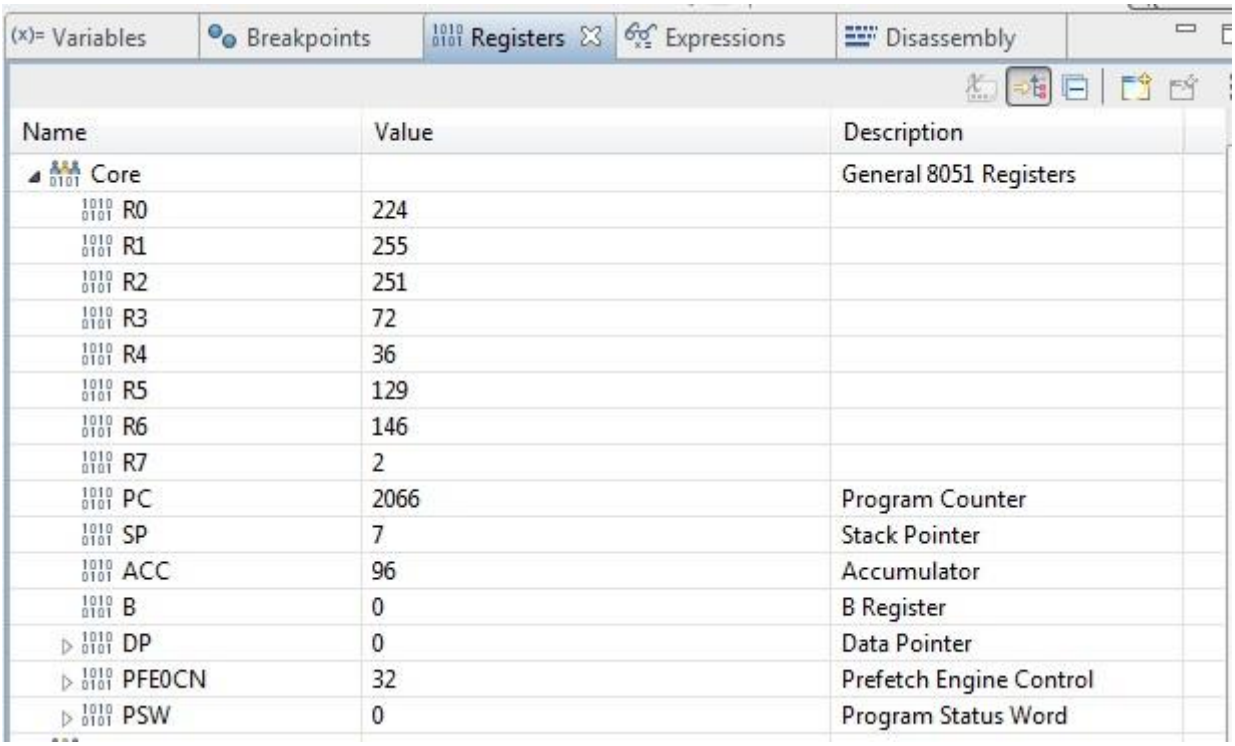
Остання команда (*нижній рядок, «load E0 into R0»*) завантажує значення 0E0H на регістр **R0** (див. рис. 2а) або 224 (на рис.2б).

```
  MOV A,#224; load E0 into the accumulator
```

```
XRL A, #128;  XOR the accumulator with 128
MOV R0, #0E0H;  load E0 into R0
```



а) – значення регістрів у шістнадцятирічному форматі



б) – значення регістрів у десятинному форматі

Рис. 2 – Регістри у середовищі розробки *Simlicity Studio*

Ще один приклад, це застосування інструкції «**MUL**» – множення.

Для цієї операції можна використовувати лише регістри **A** (тобто акумулятор) та **B**. З іншими регістрами така інструкція працювати не буде!

```
mov A, #9;
mov B, #2;
mul AB
```

Після виконання відповідні регістри мають значення:

```
ACC     18      Accumulator   ,
B       0       B Register    .
```

Якщо перемножити числа 204 та 2, то виникає переповнення регістра **A**, тому його значення буде 408-2^8=152 (408_{10}=110011000_2, 10011000_2 - 152_{10}). Натомість у регістрі **PSW** значення **OV** буде дорівнювати 1.

```
mov A, #204;
mov B, #2;
mul AB;
```

Значення регістрів мають відповідні значення:

ACC 152 *Accumulator*
OV 1 - SET (*An overflow occurred*) *Overflow Flag*

Аналогічно є інструкція поділу **DIV**:

```
mov A, #18;
mov B, #2;
div AB;
```

Після виконання, відповідні регістри мають наступне значення:

ACC 9 Accumulator ,
B 0 B Register .

Символьний опис ***директив*** дозволяє створювати символи, які можуть використовуватися у відповідних регістрах, числах та адресах.

Символи визначаються директивами, та не можуть перевизначитися. Виняток складає лише директива **SET**. Так, наприклад, у наступному коді:

```
LIMIT EQU 1200
VALUE EQU LIMIT -200+'A'
ACCU EQU A
COUNT EQU R5
COUNTER SET R1
TEMP SET COUNTER
TEMP SET VALUE*VALUE
```

- змінна «**TEMP**» перевизначається, як «**COUNTER**» або «**VALUE*VALUE».** - З іншими директивами такого робити не можна!

2.4 Приклад роботи зі змінними

Робота зі списками змінних слід виконувати тільки за допомогою директиви типу **DB**. Спостереження цих змінних в області пам'яті мікроконтролера – важлива задача, та перший етап у подальшій роботі з цими змінними. Нижче наведено приклад, який дозволяє формувати списки змінних та розташовувати їх у певній області пам'яті. Результат виконання цих процедур, представлено нижче, на рис. 3.

```
; Змінні знаходяться у пам'яті із адресою від 40H до 4BH
    ll          DATA    40H   ;
    lh          DATA    41H   ;
    U12l        DATA    42H   ;
    U12h        DATA    43H   ;
    U15l        DATA    44H   ;
```

```
        U15h        DATA        45H   ;
        U33l        DATA        46H   ;
        U33h        DATA        47H   ;
        U5l         DATA        48H   ;
        U5h         DATA        49H   ;
        Tc          DATA        4AH   ;
        X1          DATA        4BH   ;       допоміжна константа
; значення констант
Table:              DB    90    ;
                    DB    50    ;
                    DB    25    ;
                    DB    25    ;
                    DB    25    ;
                    DB    25    ;
                    DB    25    ;
                    DB    25    ;
                    DB    125   ;
                    DB    25    ;
                    DB    65    ;

mov DPTR, #Table
mov         X1,#0
mov    R1, #40H
mov         R0,#40h
M1:
        mov    A, #0
        movc   A, @A+DPTR
        mov    @R0,A
        inc DPTR
        inc    X1
        inc    R0
cjne    R0,#4BH,M1
```

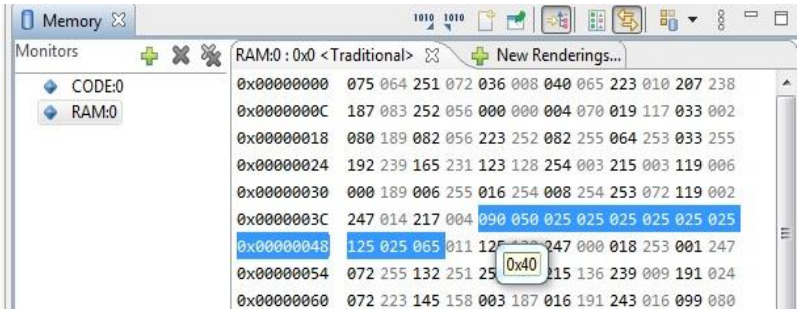


Рис. 3 – Моніторинг пам’яті зі змінними
(ACC 65 Accumulator)

3 Висновки

Робота містить стислий опис використання мови *Assembler ax51* для цілей програмування мікроконтролера C8051F340 у середовищі розробки Simplicity Studio. Звернено увагу на той факт, що всі три джерела [1-3] інформації (*керівництво по Assembler Ax51 та описи мікроконтролера та середовища Simplicity Studio*), змістовно відокремлені один від одного, в наслідок чого, важко орієнтуватися та оцінювати результат роботи команд.

В роботі розглянуто декілька прикладів, які демонструють результати і пояснюють їх роботу з точки зору тестування. Також, запропоновані пояснення, стосовно компонентів, на які поділяються команди, а саме директиви, інструкції та елементи контролю.

Наведено стислий опис регістрів, директив, елементів контролю, інструкцій та роботи зі змінними. Надано приклади використання деяких логічних та арифметичних інструкцій (у вигляді коду), з поясненням результатів виконання, які можна побачити у скріншотах зі значеннями регістрів після виконання цих команд. Додано приклади роботи зі змінними, результати яких фіксуються у пам’яті мікроконтролера.

Посилання

[1] Macro assembler and utilities for 8051 and variants/ [Електронний ресурс] – Режим доступу: https://web.engr.uky.edu/~jel/course/587/datasheets/A51.pdf - 24.08.2021.
[2] Опис мікроконтролера C8051F340/ [Електронний ресурс] – Режим доступу: https://www.silabs.com/documents/public/data-sheets/C8051F34x.pdf - 24.08.2021.
[3] Simplicity studio software/ [Електронний ресурс] – Режим доступу: https://www.silabs.com/developers/simplicity-studio - 24.08.2021.

Рецензент: Владимир Хома, д.т.н., проф., Опольский Политехнический Университет, Ополе, Польша.
E-mail: xoma@wp.pl

Поступила: Сентябрь 2021.

Авторы:
Ольга Мелкозерова, к.т.н., доцент кафедры Безопасности информационных систем и технологий, ХНУ имени В.Н. Каразина, Харьков, Украина.
E-mail: olha.melkozerova@karazin.ua
Ирина Гальцева, ст. преподаватель кафедры Безопасности информационных систем и технологий, ХНУ имени В.Н. Каразина, Харьков, Украина.
E-mail: irina.galceva@karazin.ua

Программирование микроконтроллеров с использованием решений Simplicity Studio, на языке Assembler Ax51.

Аннотация. В статье приведен обзор существующих возможностей программирования на языке Assembler Ax51 микроконтроллера C8051F340 в среде разработке Simplicity Studio. Статья является руководством для практического применения языка Assembler. В руководствах [1-2] довольно много информации, относительно языка Assembler и описания микроконтроллеров C8051F340, однако эта информация отделена друг от друга. Более того, мало объяснений по практическому использованию, вместе со средой разработки Simplicity Studio [3]. Материал содержит объяснение относительно компонентов, на которые делятся команды, а именно директивы, инструкции и элементы контроля. Представлено краткое описание регистров, директив, элементов контроля, инструкций, работы с переменными. Есть примеры использования, в виде кода, некоторых логических и арифметических инструкций с объяснением результатов выполнения, которые можно увидеть в скриншотах со значениями регистров после выполнения команд. Добавлены примеры работы с переменными, результаты которых фиксируются в памяти микроконтроллера.

Ключевые слова: микроконтроллер; директивы; инструкции; элементы контроля; регистры; Assembler.

Reviewer: Volodymyr Khoma, Dr. of Sciences (Eng.), Full Prof., The Opole University of Technology, Opole, Poland.
E-mail: xoma@wp.pl

Received: September 2021.

Authors:
Olha Melkozerova, Ph.D., Associate Professor Department of Security of Information Systems and Technologies, V.N. Karazin Kharkiv National University, Ukraine.
E-mail: olha.melkozerova@karazin.ua
Irina Galceva, Senior Lecturer, Department of Security of Information Systems and Technologies, V.N. Karazin Kharkiv National University, Ukraine.
E-mail: irina.galceva@karazin.ua

Programming of microcontrollers using Simplicity Studio solutions of language Assembler Ax51.

Abstract. The article provides an overview of the existing programming capabilities in the Assembler Ax51 language of the C8051F340 microcontroller in the Simplicity Studio development environment. The article is a guide for the practical application of the Assembler language. Guides [1-2] have a lot of information about the Assembler language and the description of the C8051F340 microcontrollers, but this information is separated from each other. Moreover, there was little explanation for its practical use with the Simplicity Studio development environment [3]. The paper provides explanations of the components into which the teams are divided, namely directives, instructions and controls. A brief description of registers, directives, control elements, instructions, work with variables is given. There are examples of using some logical and arithmetic instructions in the form of code with an explanation of the results of execution, which can be seen in screenshots with the values of the registers after the execution of commands. Added examples of working with variables, the results of which are stored in the memory of the microcontroller.

Key words: Microcontroller; Directives; Instructions; Controls; Registers; Assembler.

УДК 004.6:004.7 DOI: 10.26565/2519-2310-2021-2-03

ДОСЛІДЖЕННЯ ПРОЦЕСІВ ПОШИРЕННЯ ІНФОРМАЦІЇ В ДЕЦЕНТРАЛІЗОВАНИХ МЕРЕЖАХ

Дмитро Орлов, Ірина Гальцева

Харківський національний університет імені В.Н. Каразіна, Харків, 61022, Україна
d.eagle@ukr.net, irina.galceva@karazin.ua

Рецензент: Микола Карпінський, д.т.н., проф., університет Бельсько-Бяла, Бельсько-Бяла, Польща.
mkarpinski@ath.bielsko.pl

Надійшла: Жовтень 2021.

Анотація: *У статті наведені результати аналізу процесів розповсюдження вузлів, які створюють децентралізовану мережу, зберігають і генерують інформацію цієї мережі, та взаємодіють між собою впливаючи на загальний стан системи. Проведено моделююче дослідження ефективності дослідного алгоритму системи моніторингу і аналізу поведінки вузлів відповідної системи. Надано порівняння отриманих результатів з вже існуючими рішеннями. Для дослідження топології застосовано тестову версію програми, яка дозволяє проводити збір інформації про функціонуючи вузли шляхом безпосередньої взаємодії. Дослідження проводились на прикладі однорангових децентралізованих систем мережі Bitcoin. За результатами роботи було створено програмний продукт, який аналізує мережу Bitcoin, будує її топологію, відслідковує зміни, які здійснилися у мережі, та забезпечує візуалізацію результатів у режимі реального часу. Під час вивчення наявних реалізацій процесів розповсюдження в децентралізованих системах, було досліджено питання стосовно відкриття пірів та їх управління. Розглянуто проблему відкриття топології в біткоїн мережі та основні методи виявлення топології. Запропоновано альтернативний метод вирішення цього питання.*

Ключові слова: *блокчейн; біткоїн; децентралізація; топологія біткоїн мережі; криптовалюта*.

1 Вступ

Інтернет-комерція в більшості випадків спирається на фінансові установи, які виступають в ролі довірених посередників для проведення електронних платежів. Така схема добре працює для більшості транзакцій, але в її основі лежить довіра, що тягне за собою певні проблеми. Необхідність посередництва фінансових інститутів перешкоджає здійсненню незворотних транзакцій. Ціна цих послуг збільшує вартість транзакцій і встановлює мінімальну їх ціну, роблячи непрактичним проведення нечастих і невеликих транзакцій. Крім того, відсутність незворотних транзакцій збільшує і вартість сервісів, чиї послуги є невід'ємними. Оскільки платіж можна анулювати, продавець змушений бути насторожі, вимагаючи від покупця більше інформації, ніж йому необхідно. І певний відсоток шахрайства приймається просто, як неминучість. Ці націнки і невизначеності з банківськими платежами можуть бути подолані в разі використання готівки, однак механізму для проведення прямих електронних транзакцій поки не існує. При цьому, криптовалютні платіжні системи, що засновані на криптографії, а не на довірі, дозволяють будь-яким двом учасникам здійснити переказ коштів безпосередньо, без участі посередника. Крім того, обчислювальна «дорожнеча» процедури скасування транзакцій, потенційно, може відгородити продавців від шахрайства, а легко здійснювані механізми, відповідним чином, захистили б покупців.

2 Огляд відомих реалізацій процедур розповсюдження у децентралізованих мережах та розгляд запропонованого методу

У мережі біткоїн, піри (*тобто, учасники мережі*) ініціюють одну з можливих реалізацій *Bitcoin P2P* протоколу: понад 75% пірів мають версії Bitcoin Core *(клієнт Satoshi)*, а менше 10 % пірів мають інші реалізи (*наприклад, такі як BitcoinJ, Libbitcoin або btcd).* Всі вони можуть реалізовувати кілька функцій, таких як маршрутизація, використання гаманця ,бази даних блокчейн, майнинг, та запуск різних протоколів [1].

Гаманець - це засіб для здійснення обробки і циркуляції біткоїнів (*надсилання, отримання та зберігання*). Усі піри повинні реалізувати функцію маршрутизації для енергійної роботи в мережі. Інші функції, також, можна комбінувати для визначення різних цілей для пірів у мережі Bitcoin. Переважно розрізняється дві ролі: повний вузол та легкий гаманець. Повний вузол може бути або повним блокчейн-вузлом, коли він включає маршрутизацію та повну функцію бази даних Blockchain, або «соло майнером», коли поєднується маршрутизація, повна база даних (БД) Blockchain та майнинг функції. При цьому пір, який виконує цю роль, споживає велику кількість ресурсів (*CPU-GPU, потужність, дисковий простір, тощо*). Роль «легкого» гаманця (SPV) інтегрує функції маршрутизації та гаманця. На відміну від повної ролі вузла, він розроблений для пірів із обмеженими ресурсами.

2.1 Відкриття пірів

Коли процес тільки ініціюється, пір не має відомостей про «активних» сусідів, що мають «повну» роль вузла. Щоб визначити їх адреси (*загальнодоступний IP та номер порту*), необхідно використовувати кілька механізмів для проведення однорангового пошуку: взяття збережених у ньому адрес (*локальна БД адрес*), що виконуються по внутрішній команді консольного клієнту біткоїн (*-addnode та -connect*), надсилання DNS запиту (запитуючи інші піри), надсилання повідомлення *getaddr()* або чекаємо на повідомлення відповіді від вузлів *addr()*. Сформований список адрес зберігається і оновлюється в локальній БД адрес [2].

2.2 Стохастичне управління одноранговими адресами

Біткоїн пір підтримує БД локальних адрес, яка поєднує виявлені аналоги. Для кожного запису ця база включає: номер порту, IP-адресу однорангового партнера, часову позначку останнього з'єднання, тощо. Для того, щоб сприяти вибору «сусідів», усі адреси пірів, які зберігаються в БД, систематизуються та заносяться у відповідний «контейнер». Існує два типи контейнерів: - контейнер для випробуваних адрес та контейнери для нових адрес. При цьому, контейнери для випробуваних адрес містять адреси пірів, з якими вони мали, принаймні, хоча б одну «вдалу» вхідну/вихідну взаємодію (*тобто з'єднання*).

Контейнери для нових адрес складають адреси пірів, з якими не було конекту (*встановленого раніше*), або адреси, вилучені із спроб контейнеру. Існує 256 контейнерів для випробуваних, та, відповідно, 1024 контейнера для нових адрес. Кожен контейнер може зберігати max 64 адреси. Т.ч., загальна кількість записів в БД обмежено 81920 адресами.

Якщо вхідний або вихідний коннект з піром успішно встановлено, то пізніше, ця адреса буде зафіксована у контейнерах для випробуваних адрес. Процес вибору відповідного випробуваного контейнеру визначається функцією *GetTriedBucket1 (див. рис. 1)*.

```
сКлюч: секретний ключ з 256 біт для рандомізації вибору
контейнеру
Ап: IPv4-адреса піру та його номер порту
Група:  16  IPv4  префіксів пірів  IPv4 адрес .
1. хеш1 ← хеш(сКлюч, Ап) % 8
 2.хеш2 ← хеш(сКлюч, група, hash1)
3.контейнер ← хеш2 % 256
4.Повернути контейнер
```

Рис. 1 – Функція GetTriedBucket1

Кожна 16 група IPv4, випадковим чином, обирає 8 контейнерів із 256, а потім, теж випадковим чином визначає, одне з них на базі адресу піру (*IPv4 та номер порту*). Якщо адреса піру присутня у виділеному випробуваному контейнері, то його часовий маркер оновлюється. В іншому випадку адреса пірів вставляється в 1-ше доступне положення. Якщо вибраний контейнер «заповнений», то 4-ри випадкові записи визначаються випадковим чином, а найстарше, переходить з нього назад, до нових контейнерів. Щоб збільшити кількість випро-

буваних адрес у випробуваних контейнерах, вузол, після встановлення вихідних з'єднань, випадковим чином і періодично, підбирає піри з нових контейнерів. При цьому він намагається підключитися до них і додає IP-адреси з вдалим підключенням до контейнеру. Для нових адрес, вибирається новий контейнер, за рахунок функції *GetNewBucket1* (*див. рис. 2*).

```
сКлюч: секретний ключ з 256 біт для рандомізації вибору
контейнеру
Ап: IPv4-адреса піру та його номер порту
Група: 16 IPv4 префіксів пірів IPv4 адрес
Вих група : Префікс / 16 IPv4 вихідних  IPv4 адрес
рекламуючих
IPv4-адреси пірів .
1. хеш1 ← хеш(сКлюч, Група р, Вих група) % 64
 2.хеш2 ← Hash(сКлюч, Вих група , хеш1)
3.Контейнер ← хеш2 % 1024
 4.Повернути Контейнер
```

Рис. 2 - Функція *GetNewBucket1*

Кожна пара *(«група», «Вих група»)* випадковим чином визначає нові 64 контейнери та лише один з них обирається з використанням інформації про вхідну групу. Оскільки IPv4-адресу, яку потрібно інтегрувати, можна рекламувати одразу з декількох джерел, то вона може поєднувати до 8 різних відрізків.

При додаванні нової адреси до повного контейнеру, випадковим чином вилучається обрана адреса, з так званих, «не валідних» адрес. Адреса вважається не діючою, якщо:

а) її часовий маркер випереджається на більш ніж 10 хв. поточного часу;

б) відмічено 7 послідовних збоїв конекту за останній тиждень;

в) її часовий маркер старший більш ніж на тиждень.

2.3 Вибір випадкових «сусідів»

Біткоїн пір може відібрати до 8 (*значення за замовчуванням*) *max* вихідних з'єднань (*до інших пірів, що мають повну роль вузла*), а потім підтримувати їх. Пропускна здатність кожного вихідного конекту обмежена 160 кбіт / с, щоб запобігти перевантаження наявної пропускної здатності каналів, трафіком Bitcoin. Крім того, кожен пір може приймати *max* 117 (*значення за замовчуванням*) вхідних сусідів, або навпаки, повністю відмовитись від будь-яких вхідних взаємодій. Однак, ця відмова недоцільна, з точки зору для підтримки мережі Bitcoin. Визначення алгоритму нового вихідного з'єднання здійснюється щоразу, коли поточна кількість вихідних з'єднань менше 8. Це може статися, наприклад, коли пір перенавантажується, або коли один із його вихідних сусідів припиняє з'єднання. Як правило, пір може «закрити» вхідні з'єднання, але ніколи не повинен припиняти вихідні з'єднання, за винятком випадків, коли його вихідний сусід перебуває у забороненому реєстрі (*Stop листі*).

2.4 Труднощі відкриття топології мережі біткоїн

Існуюча структура топології мережі Bitcoin невідома, оскільки протокол Bitcoin не забезпечує виявлення топології. Ця функціональність свідомо не підтримується, щоб захистити мережу Bitcoin від проведення потенційних мережевих атак (*наприклад, Eclipse, Sybil та ін.*). Варто підкреслити, що існують рішення, які опосередковано відображають топологію Bitcoin, використовуючи для цього інформацію (*яка, від самого початку, не призначена для виявлення топології*) в протоколі Bitcoin [3].

2.5 Картографування топології на основі часових маркерів

Такий підхід був вперше використаний у *Coinscope*. В цьому разі БД локальних адрес, що підтримується кожним з пірів, містить відомості про інші піри, включаючи часову мітку. Ці часові позначки допомагають встановити ступінь новизни піру, а також, чи є він «сусі-

дом» вхідного / вихідного. Це пов'язано з тим, що ці часові маркери оновлюються щоразу, коли піри здійснюють взаємодіють, за умови, що відповідна часова мітка є старшою в заданої межі часового відрізку. Розміщуючи відповідні верхню та нижню границю на цій межі часових міток, для будь-якого піру, можна визначати його вихідних та вхідних сусідів.

Зазначений підхід «працював» до 0.10 версії клієнта ***Bitcoin Core***. Потім відбулися зміни, які оновлювали часові маркери лише після відключення даного піру. Таким чином стало неможливо виявляти вихідних та/або вхідних сусідів для пірів, які реалізують останні версії, використовуючи розглянутий вище підхід [3].

2.6 Топологія на основі концепції атаки Sybil

Даний підхід був запропонований відомим криптографом Люксембургського університету Алексом Бірюковим. Так, розроблений їм фреймворк спочатку підключається в мережі до всіх аналогів Bitcoin, а вже потім кожний стійкий коннект піддається з його сторони флуду (*flood*). Після цього даний фреймворк очікує від інших пірів, отримання всіх підроблених їм адрес назад, оскільки цільовий аналог передав їх вихідному / вхідному каналу сусідів.

Слід зазначити, що для даного підходу притаманно декілька особливостей:

а) при отриманні пересланого повідомлення addr() (*яке містить відповідні фальшиві адреси*), програмна структура фреймворку не може гарантувати, що відправник є першочерговим сусідом «атакованого» їм вузла [3];

б) програмний каркас отримує переслані повідомлення *«addr()»*, лише в разі, якщо він визначений пірами, як один з 1-2 випадково обраних «сусідів»;

в) для забезпечення більш коректної оцінки топології мережі Bitcoin, відповідні програмні каркаси повинні мати зв'язки з усіма пірами в межах цієї мережі.

2.7 Метод відображення топології, що пропонується

Отримавши зміни в протоколі Bitcoin, пропонується застосувати новий метод відображення оцінки топології Bitcoin мережі. Дослідний алгоритм візуалізації топології, спочатку, в режимі реального часу, взаємодіє з мережею Bitcoin. На цьому етапі забезпечується перевірка доступності мережі, та будується і оновлюється БД локальних адрес для кожного з вузлів, запитуючи її кілька разів протягом визначеного терміну часу. Після цього, алгоритм отримує в своє розпорядження всю необхідні відомості для того, щоб визначити сусідів для кожного з пірів, які емулюють реальну поведінку вузлів Bitcoin.

Як тільки дані про всіх сусідів для кожного з пірів отримані, дослідний алгоритм генерує відповідну мапу на основі відомостей, що були отримані від вузлів мережі біткоїн. Стуктурно-логічна схема роботи даного методу (*реалізовано на мові Python*) наведено на рис. 3.

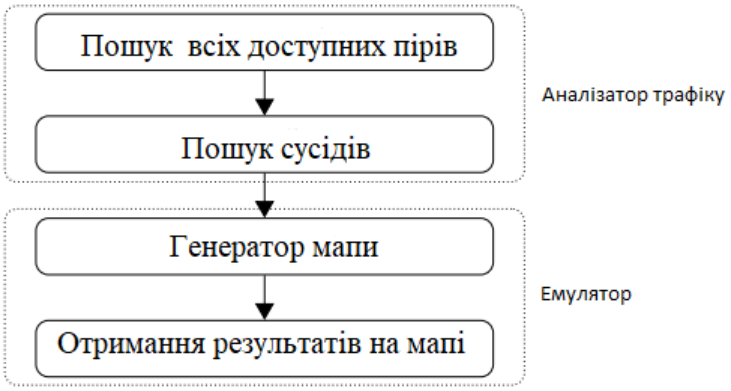


Рис. 3 – Спрощена схема методу

3 Отримані результати

3.1 Порівняння отриманих результатів мапою Bitnodes.io

Мапа *bitnodes.io* оперує даними про 13381 вузлів *(станом на 10.10.21)*, які є найбільш активними в мережі. Характерний вид цієї мапи наведено нижче, на рис. 4. Також, ця мапа оперує даними про країни, де зосереджена найбільша кількість активних вузлів (див. рис. 5).

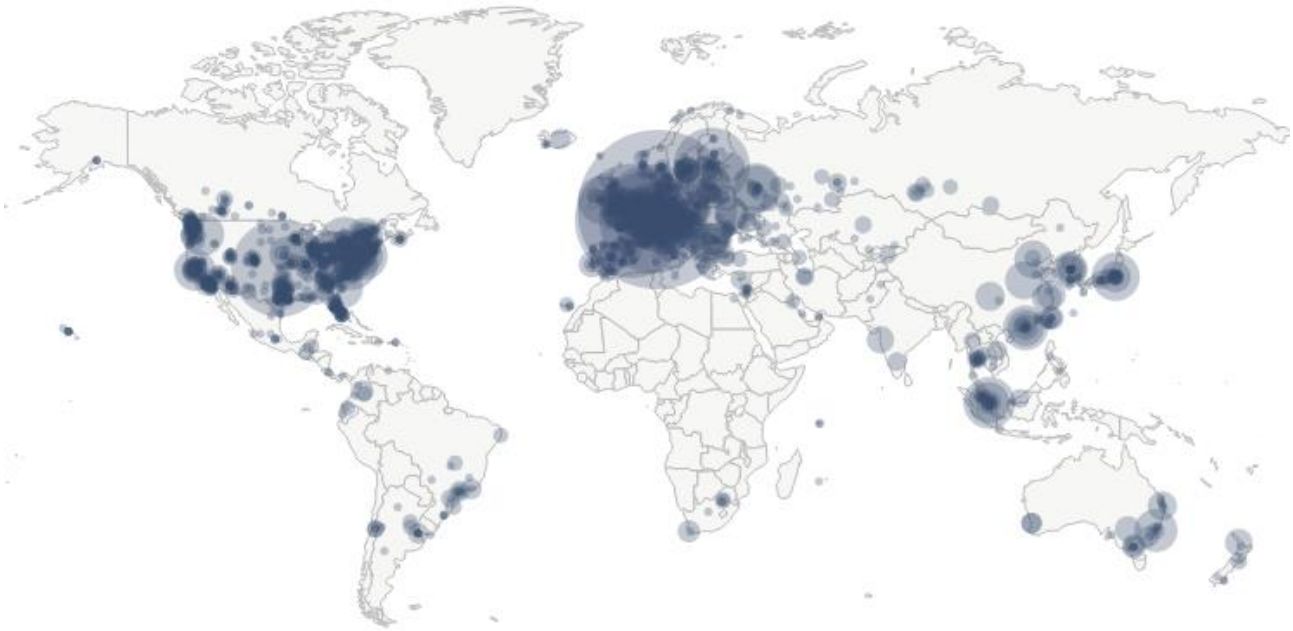


Рис. 4 – Мапа місцезнаходження біткоїн пірів *(за даними bitnodes.io)*

RANK	COUNTRY	NODES
1	n/a	5965 (44.58%)
2	United States	1859 (13.89%)
3	Germany	1807 (13.50%)
4	France	532 (3.98%)
5	Netherlands	377 (2.82%)
6	Canada	311 (2.32%)
7	United Kingdom	267 (2.00%)
8	Russian Federation	190 (1.42%)
9	Finland	187 (1.40%)
10	Switzerland	133 (0.99%)

Рис. 5 – Поточні дані про країни з найбільшою кількістю вузлів *(за даними bitnodes.io)*

Якщо порівняти наявні, на час проведення моделювання, статистичні дані, то можна побачити, що відомості, котрі були отримані запропонованим методом (*мапа на рис. 6 та, відповідна кількість вузлів на рис. 7*), у своєму відсотковому співвідношенні, практично співпадають з даними, які були отриманими з ресурсу *bitnodes.io*.

Різниця між двома мапами полягає в тім, що при використанні запропонованого методу його алгоритм оперує набагато більшою кількістю даних, а саме 58440 вузлів, в той час як *bitnodes.io* демонструє дані лише про 13381 пір. Різниця відчутна, що безсумнівно є великою перевагою даного методу дослідження топології користувачів біткоїн мережі. В середньому, за вісь цикл моделювання, кількість даних, що отримані з використанням методу, який пропонується, приблизно в 4,2–4,5 рази більша ніж даних, якими оперує *bitnodes.io*.

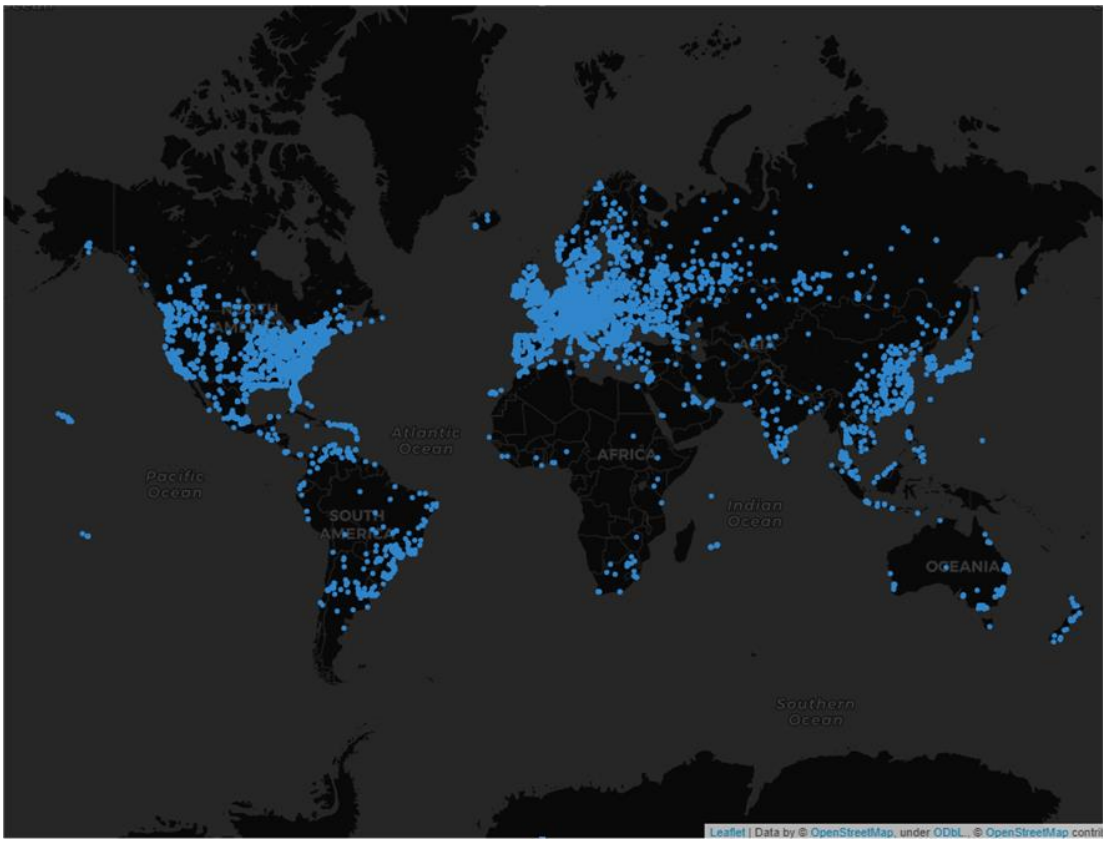


Рис. 6 - Мапа локації біткоїн пірів, яка отримана запропонованим методом

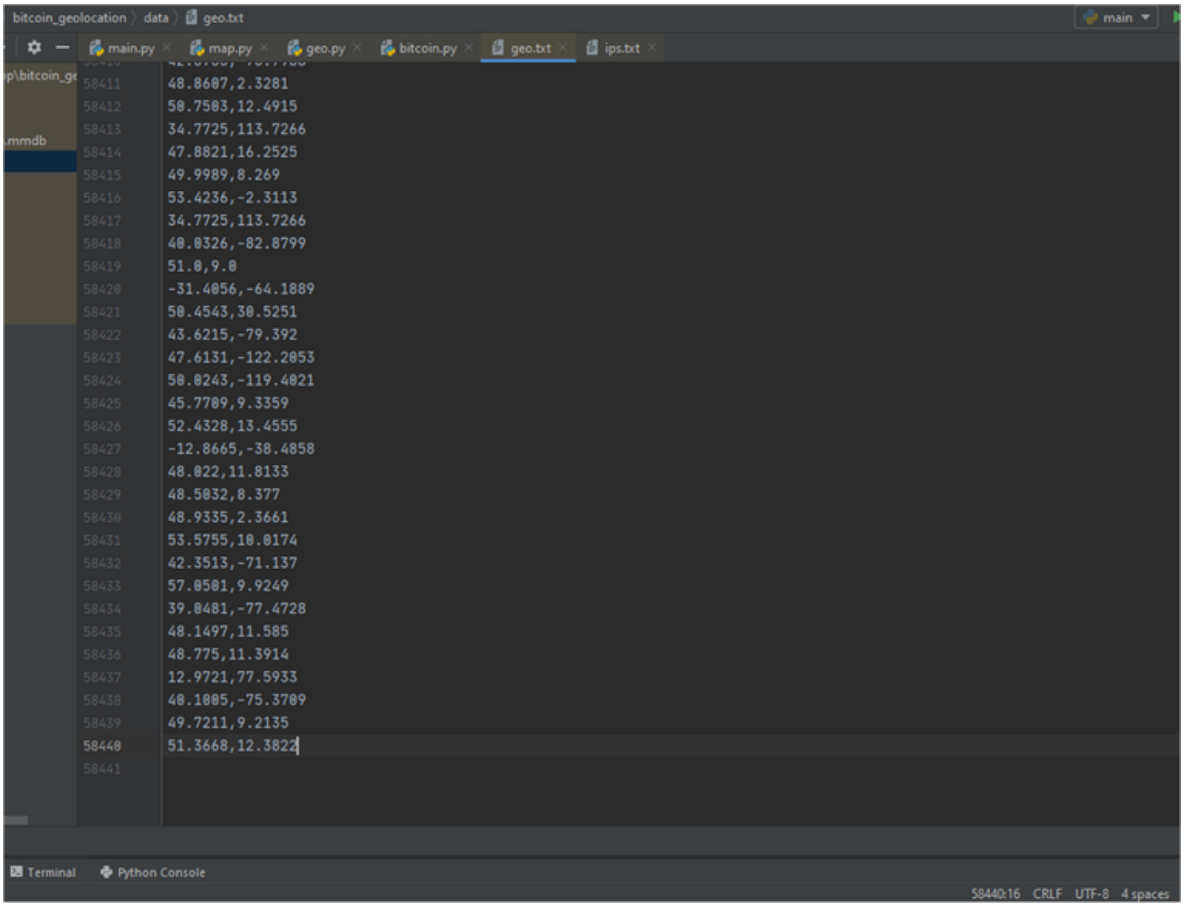


Рис. 7 – Кількість отриманих геолокацій
(*фрагмент інтерфейсу тестового програмного додатку*)

Зазначена перевага дозволяє отримати помітно більший обсяг геоданих, стосовно активних пірів, і як наслідок, синтезувати більш адекватну, для поточної ситуації, структуру мережі. Відповідно, детальний аналіз більш ємних та інформативних вихідних даних, надає змогу сформувати набагато більш об'єктивну картину користувачів мережі біткоін.

4 Висновки

1. За результатами проведеного циклу досліджень запропоновано новий метод аналізу мережі біткоїн, який будує її топологію, відслідковує всі поточні зміни, що здійснилися у мережі, та здійснює візуалізацію узагальнених відомостей в режимі реального часу.

2. Запропонований метод відрізняється від відомих тим, що формує відчутно більш адекватну *(за кількості вузлів, що аналізуються)* структуру мережі, чим створює відповідні умови для покращення можливостей автоматизованих систем моніторингу змін у децентралізованих мережах, та посилює функції відстеження аномальної активності в мережі.

3. Отримані результати можуть бути використані при реалізації систем моніторингу та захисту мережевих систем, які створені за децентралізованим принципом.

Посилання

[1] A. Antonopoulos, Mastering Bitcoin: Programming the Open Blockchain, 2nd Edition. Reading, MA: AO'Reilly Media, 2017.
[2] Satoshi client node discovery. DOI: https://en.bitcoin.it/wiki/Satoshi_Client_Node_Discovery
[3] BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology. Varun Deshpande, Hakim Badis, Laurent George. – 2018

Reviewer: Mikołaj Karpiński, Dr. of Sciences (Eng.), Full Prof., University of Bielsko-Biala, Bielsko-Biala, Poland.
E-mail: mkarpinski@ath.bielsko.pl

Received on October 2021.

Authors:
Dmytro Orlov, student, Faculty of Computer Science, V. N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: d.eagle@ukr.net

Irina Galceva, Senior Lecturer, Department of Security of Information Systems and Technologies, V.N. Karazin Kharkiv National University, Ukraine.
E-mail: irina.galceva@karazin.ua

Research of information dissemination processes in decentralized networks.

Abstract. The article presents the results of the analysis of the processes of node propagation, which create a decentralized network, store and generate information of this network, and interact with each other influencing the general state of the system. A modeling study of the effectiveness of the experimental algorithm of the system of monitoring and analysis of the behavior of the nodes of the corresponding system was carried out. A comparison of the obtained results with existing solutions is given. To study the topology, a test version of the program was used, which allows to collect information about functioning nodes by direct interaction. The research was conducted on the example of peer-to-peer decentralized Bitcoin network systems. As a result of the work, a software product was created that analyzes the Bitcoin network, builds its topology, and tracks the changes that have taken place dreamed in the network, and provides visualization of results in real time. During the study of the existing implementations of distribution processes in decentralized systems, the issues concerning the opening of peers and their management were investigated. The problem of topology discovery in bitcoin network and basic methods of topology detection are considered. An alternative method of solving this problem is proposed.

Keywords: Blockchain; Bitcoin; Decentralization; Bitcoin Network Topology; Cryptocurrency.

Рецензент: Николай Карпинский, д.т.н., проф., Университет Бельсько-Бяла, ул. Виллова 2, 43-309 Бельсько-Бяла, Польша.
E-mail: mkarpinski@ath.bielsko.pl

Поступила: Октябрь 2021.

Авторы:
Дмитрий Орлов, студент факультета компьютерных наук, Харьковский национальный университет имени В. Н. Каразина, Харьков, Украина.
E-mail: d.eagle@ukr.net

Ирина Гальцева, ст. преподаватель кафедры Безопасности информационных систем и технологий, ХНУ имени В.Н. Каразина, Харьков, Украина.
E-mail: irina.galceva@karazin.ua

Исследование процессов распространения информации в децентрализованных сетях.

Аннотация: В статье приведены результаты анализа процессов распространения узлов, которые создают децентрализованную сеть, сохраняют и генерируют информацию этой сети, взаимодействуют между собой, влияя на общее состояние системы. Проведено моделирующее исследование эффективности опытного алгоритма системы мониторинга и анализа поведения узлов соответствующей системы. Приведено сравнение полученных результатов с уже существующими решениями. Для исследования топологии применена тестовая версия программы, которая позволяет проводить сбор информации о функционирующих узлах путем непосредственного взаимодействия. Исследования проводились на примере одноранговых децентрализованных систем сети Bitcoin. По результатам работы создан программный продукт, который анализирует сеть Bitcoin, строит ее топологию, отслеживает изменения, которые произошли в сети, и обеспечивает визуализацию результатов в режиме реального времени. При изучении имеющихся реализаций процессов распространения в децентрализованных системах, были исследованы вопросы открытия пиров и их управления. Рассмотрена проблема открытия топологии в биткоин сети и основные методы выявления топологии. Предложен альтернативный метод решения этого вопроса.

Ключевые слова: блокчейн; биткоин; децентрализация; топология биткоин сети; криптовалюта.

УДК 621.327:621.391 DOI: 10.26565/2519-2310-2021-2-04

МОДЕЛЮВАННЯ ОСНОВНИХ ПРОЦЕДУР СТИСКУ ЗОБРАЖЕНЬ З ЧАСТКОВОЮ ВТРАТОЮ ІНФОРМАЦІЇ, НА ПРИКЛАДІ МЕТОДІВ КОДУВАННЯ З ПЕРЕТВОРЕННЯМ

Катерина Кузнецова, Єлизавета Кузнецова, Сергій Малахов

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
kate7smith12@gmail.com, kuznetsova2021kb11@student.karazin.ua, mailgate@meta.ua

Рецензент: Володимир Хома, д.т.н., проф., Опольський політехнічний Університет, Ополе, Польща
xoma@wp.pl

Надійшла: Листопад 2021.

Анотація: *Застосування групи методів кодування з перетворенням, як основи для синтезу універсальних алгоритмів стиску відеоданих, базується на концепції максимального використання особливостей зорової системи людини: - час експозиції; - відстань спостереження; - величина міжкадрової різниці; - розмірність і співвідношення основних об'єктів спостереження; - тип зображень та ін.. Кодування з перетворенням дозволяє переходити від просторового до спектрального представлення оброблюваних зображень, и забезпечує широкі можливості для їх подальшого стиску та/або стеганографічної обробки.*

В якості ключового компонента дослідного алгоритму використовувалося дискретне косинусне перетворення (ДКП). Воно є універсальним практично для всіх типів зображень. ДКП забезпечує хорошу спроможність до концентрації більшої частині значущої інформації, в меншій (у порівнянні з іншими перетвореннями) кількості коефіцієнтів. Ця властивість ДКП створює хороші стартові умови для проведення всіх наступних процедур стиснення зображень. Впливаючи на спектральне уявлення оброблюваних зображень, можна балансувати між якістю відтворення та ступенем стиску вихідного зображення, а впливаючи на розмір субблоків і параметри їх попередньої обробки, на загальний час виконання операцій.

При використанні методів кодування з перетворенням, основними параметрами, які впливають на ефективність процесу стиску, виступають: - використовуваний розмір субблоків, на які поділяється вихідне зображення; - спосіб відбору значимих коефіцієнтів; - використовуваний принцип квантування значимих коефіцієнтів. Розмір субблоків, є визначальним, так як від нього залежить чисельність подібних блоків та загальний час обчислення кожного елементу матриць ДКП.

Важливим етапом є процес округлення отриманих після ДКП значень. У дослідженні для цього використовується коефіцієнт загрублення. Процедура нормування елементів матриць реалізується шляхом ділення значень ДКП на задану константу загрублення, з подальшим округленням результату до цілих чисел.

Метою роботи є моделювання основних етапів процесу стиску статичних зображень,та дослідження впливу основних параметрів кодування на якісно-часові характеристики відтворюваних зображень (час обробки, обчислювальна складність, якість відтворення, характеристики спотворень тощо).

Ключові слова*: методи стиску зображень; кодування з перетворенням; ДКП; відбір коефіцієнтів; спотворення; квантування коефіцієнтів; візуальна помітність.*

1 Вступ

Методи кодування з перетворенням знайшли широке застосування в області обробки зображень і сигналів загалом, а також при реалізації різних методів компактного представлення (стиску) даних, фільтрації тощо. Протягом історії було розроблено багато різних методів і алгоритмів для вирішення конкретних прикладних задач [1-5]. В цілому, відомі методи можна класифікувати наступним чином: *а)* непрямі обчислення; *б)* пряма факторизація; *в)* рекурсивне обчислення. Так наприклад, основними механізмами при побудові дискретного косинус перетворення (ДКП), в (*а*) є швидке перетворення Фур'є (ШПФ) та перетворення Уолша-Адамара, але, на етапах обчислення часто залучаються і додаткові операції.

Алгоритми групи (*б*) отримують переваги швидкості над іншими реалізаціями, через використання розрідженої матричної факторизації. Алгоритми прямого розкладання на множники були адаптовані в т.ч. і до матриць ДКП, та вимагають меншої кількості множень або додавань.

Рекурсивний підхід в (*в*) призначений для створення ДКП вищого порядку з ДКП нижчого порядку. Спосіб реалізації даного алгоритму вимагає меншої кількості множень і додавань, і не потрібно виконувати інверсії або ділення коефіцієнтів. В якості компромісу в цьо-

му випадку потрібні операції зсуву та мультиплексування, а як відомо, при цифровій реалізації процесів обробки, операції зсуву набагато швидші за множення чи додавання.

В межах даної роботі розглядається класичне перетворення ДКП в рамках моделювання і дослідження основних процедур (*квантування, відбір і нормування значимих коефіцієнтів тощо*) дослідного алгоритму стиснення напівтонових зображень [3-5]. Дослідження проводилися з використанням математичного редактора MathCAD, який реалізовував потрібну модель процесів стиску та відновлення файлів статичних напівтонових зображень з різною ймовірністю перепаду яскравості між сусідніми елементами, відповідно до базових процедур дослідницького алгоритму. Був проведено аналіз способів реалізації зазначених процедур та висунення пропозицій щодо їх наступного застосування.

2 Основна частина

Як відомо [3-5], процес стиску зображень (*наприклад, що притаманний відомому алгоритму JPEG*), включає низку наступних етапів [4,6]:

I. *Перетворення зображення в оптимальний колірний простір.*

II. *Субдискретизація компонентів кольоровості (або яскравості) пікселів.*

III. *Сегментація вихідного зображення (поділ на фрагменти за правилом).*

Сегментація зображень застосовується з метою розподілу його на більш дрібні сегментів (фрагменти), що зменшує загальну обчислювальну складність алгоритму, спрощує буферизацію даних в пам'яті використовуваних пристроїв (*наприклад, різних мобільних гаджетів*), та прискорює їх довільну вибірку з диска, та буферної пам'яті. Результатом виконання етапу/процедури сегментації вихідних зображень, є отримання сукупності фрагментів (*субблоків заданої розмірності*), що «покривають» все зображення *(або безліч контурів виділених із зображення (для деяких випадків))*.

В залежності від обраного принципу формування субблоків (*пряме розбиття вихідного зображення [4,6] або проведення деяких спеціальних процедур попередньої обробки [7-9])* всі елементи зображень (*пікселі*), в кожному отриманому субблоці (*квадратної матриці або вектор-рядку*), можуть мати, як однакові характеристики всіх складових елементів, так і різні характеристики (*наприклад, фіксований діапазон різниці яскравостей між всіма (або ж тільки сусідніх) складовими елементів субблоків*). Таким чином, в залежності від використовуваних принципів (*або критеріїв*) формування субблоків, сусідні сегменти (*субблоки*), можуть значно відрізняються один від одного *[4, 7-8]*.

В рамках цієї роботи використовувалися субблокі однакової розмірності *(8×8, 16×16, 32×32 елементів)*, з однаковим принципом їх формування (*тобто без проведення будь-яких процедур попередньої обробки*).

IV. Застосування до кожного субблоку дискретного косинусного перетворення,та зменшення надмірності даних.

В певному сенсі, ДКП допомагає сепарирувати отримані фрагменти зображення на спектральні піддіапазони різного значення (*щодо візуальних характеристик певних деталей оброблюваних зображень*) [5]. Тобто використання ДКП забезпечує перехід від просторового представлення зображень в частотну область. При цьому, для обробки масивів цифрових зображень часто використовується двовимірна версія ДКП [3-4]. Рівняння для двовимірної версії ДКП (*для зображення розміром N × M*) визначається наступним рівнянням:

$$F(u,v)=\left(\frac{2}{N}\right)^{\frac{1}{2}}\cdot\left(\frac{2}{M}\right)^{\frac{1}{2}}\cdot\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}f(i,j)\cdot\xi(i)\cdot\xi(j)\cdot\cos\left[\frac{\pi\cdot u}{2\cdot N}(2\cdot i+1)\right]\cdot\cos\left[\frac{\pi\cdot v}{2\cdot N}(2\cdot j+1)\right],$$

де $\xi(i)=\begin{cases}\frac{1}{\sqrt{2}} & якщо \quad i=0\\ 1 & якщо \quad i>0\end{cases}$; $f(i,j)$ – інтенсивність пікселя в *i-му* рядку та *j-* стовпці;

– $F(u,v)$ – коефіцієнт ДКП у рядку *k1* та стовпці *k2* матриці ДКП.

Відповідне зворотне двовимірне перетворення ДКП є простим $F^{-1}(u,v)$.

В межах проведеного циклу досліджень, моделювалась спрощена схема кодування зображень з перетворенням, основний зміст і етапи якої, полягають в наступному [3-4, 10-11]:

- Маємо вхідне зображення розмірністю *N* на *M* елементів, де в залежності від типу зображення, та кінцевої мети його обробки, можуть використовуватися різні рівні/схеми кодування його елементів *(в умовах проведеного моделювання для кожного пікселя вихідного напівтонового зображення, застосовувалося 8 бітне кодування яскравості, тобто градація яскравості елементів змінювалась в межах від 0 до 255);*
- Сегментація вихідного зображення на субблоки потрібної розмірності *(в умовах проведеного моделювання використовувалися блоки розмірністю 8×8, 16×16 та 32×32 елемента);*
- Здійснення двовимірного ДКП для всіх субблоків з масиву оброблюваного кадру вихідного зображення (*N* × *M* елементів);
- Нормування отриманих значень спектральних коефіцієнтів (*для кожної з матриць*) та їх наступна селекція з використанням певних критеріїв, що враховують параметр «*P*» і запроваджений варіант налаштувань алгоритму (*будуть розглянуті нижче*);
- Для переважної більшості зображень більша частина енергії міститься в низькочастотних (НЧ) складових спектру[3-5, 11], які мають тенденцію групуватися у верхньому лівому куті отриманих матриць спектральних коефіцієнтів;
- Ближче к нижньому правому куту, кожної з отриманих матриць, групуються елементи, які характеризують дрібні деталі оброблюваних фрагментів зображень (*область високих частот - ВЧ*). В своїй переважній більшості вони набувають достатньо малих значень, тому ними можна знехтувати (*з врахуванням особливостей зорової системи людини*), з відносно прийнятними, для кожного конкретного випадку*, видимими спотвореннями вихідних елементів.

  * - *враховує режими обробки зображень, що обробляються (внутрішньокадровий або міжкадровий), час їх спостереження, структурну складність зображень, середнє значення ймовірності перепаду яскравості між сусіднім елементами кадру зображення; середній "розмах" значень яскравості у межах окремих фрагментів зображень тощо.*

- Зменшення надмірності даних зображень виконується за рахунок реалізації «порогового» способу відбору значущих елементів трансформант [3-5]. У загальному випадку «пороговий» спосіб селекції коефіцієнтів, втілює ідею адаптивної обробки елементів трансформант, що забезпечує більш кращу якість відновлюваних зображень. В межах проведеного моделювання було застосовано 3 варіанта *Налаштувань,* що втілюють різні параметри реалізації процедур селекції значущих елементів та їх квантування.

V. Квантування елементів, що зберігаються із застосуванням механізмів, які оптимізовані до параметрів візуального сприйняття людиною [3-4, 8].

Окрім процедури селекції (*відбору*) значущих коефіцієнтів, вибір параметрів *квантування* цих коефіцієнтів, в значній мірі визначає досягнення потрібного балансу, між ступенем стиску зображення та якістю його відновлення. Стандарт JPEG [4, 6, 12] дозволяє використовувати для цього матриці заокруглення (*визначаються показником фактору якості*), проте ISO розробила набір відповідних матриць квантування. В даній роботі еквівалентом *матриць квантування JPEG*, виступає ***коефіцієнт загрублення*** *– P (використані 3 дискретні значення)*, який втілює спрощену концепцію вибіркового зменшення обсягу цифрового опису вихідного зображення, за рахунок додаткового зменшення чисельності ненульових складових.

Крім того, при реалізації кожного з передбачених варіантів *Налаштувань*, забезпечувався різний порядок формування кінцевого масиву значущих коефіцієнтів. Це здійснювалось шляхом ділення всіх елементів, що пройшли пороговий відбір, на задану величину (*наприклад, розраховане для кожної з матриць, середнє значення амплітуди коефіцієнтів перетворення ($K_{ср}$), без врахування елементів з координатами (0;0), (0;1) та (1;0))*.

VI. Кодування результуючих коефіцієнтів перетворення.

Таким чином, у ході проведеного моделювання були досліджені основні процедури алгоритмів стиску зображень, що притаманні до групи методів кодування з перетворенням (*наприклад, стандарт JPEG*) [3, 10-15].

Основним завданням роботи є визначення природи взаємозв'язків між основними параметрами (*розмір блоків і величина коефіцієнта загрублення (параметр квантування))* процедур стиску зображень з перетворенням, та візуальною помітністю і структурою викривлень у відновлювальних зображеннях. З цієї причини деякі етапи стандартного алгоритму [4, 12] були «виключені» (*наприклад, перетворення зображення в оптимальний колірний простір і субдискретизація компонентів кольоровості тощо*), як несуттєві, з точки зору процесів, що впливають на обчислювальну складність і якість відновлюваних зображень.

Як вже було зазначено вище, дослідний алгоритм, для всіх передбачених розмірностей блоків (*8, 16 і 32*), моделює пороговий спосіб відбору значимих коефіцієнтів з обов'язковим збереженням 3-х кутових елементів (*(0;0),(0;1), (1;0)*), та наступним квантуванням елементів, що зберігаються, в 3-х різних варіантах *Налаштувань*:

- Налаштування I. Збереження значень матриць, які більші або дорівнюють середньому значенню амплітуди коефіцієнтів ($K_{ср}$), для кожної окремої матриці. Всі інші значення прирівнюються до нуля (*тобто не зберігаються*). Цей варіант налаштувань еквівалентне високій якості відновлюваних зображень;
- Налаштування II. Збереження значень матриць, які отримані шляхом поділу отриманих коефіцієнтів ДКП, на значення, що дорівнює *Kcp/4*, з наступним округленням результату до цілих. Цій варіант налаштувань є аналогом середньої якості відновлюваних зображень;
- Налаштування III. Збереження значень матриць, які отримані шляхом поділу вихідних коефіцієнтів, на розраховане середнє значення амплітуди коефіцієнтів цієї матриці ($K_{ср}$). Подібний варіант налаштувань є аналогом низької якості відновлюваних зображень.

Крім того для забезпечення більшої наочності отриманих результатів, використовується, *додатковий* 4-й варіант настроювань:

- Налаштування IV. Зберігається тільки кутовий елемент з координатами (0;0) (*тобто, значення середньої яскравості для кожного з блоків кадру*). Цій варіант налаштувань є демонстратором важливості збереження значимих коефіцієнтів для кожної з матриць зображення, та від ображає наслідки запровадження граничних настроювань алгоритму *(наприклад в разі, обробки сегментів зображень, що складають зони міжкадрової різниці в відео послідовності кадрів [4])*.

3 Загальні налаштування алгоритму та порядок реалізації етапів

I. Завантаження вихідних даних в форматі *.bmp24:

M := READBMP("10")

M =

	0	1	2	3	4	5	6
0	88	88	87	87	87	88	88
1	89	88	87	86	87	88	89
2	89	89	88	87	88	89	89
3	89	89	89	89	89	89	89
4	87	88	90	91	90	88	87
5	90	90	88	88	88	90	90
6	90	90	88	88	88	90	90
7	90	90	88	88	88	90	90
8	90	90	88	88	88	90	90
9	90	90	88	88	88	90	...

II. Графічне відображення використовуваного тестового зображення (*взято з мережі Інтернет*):

III. Процедури дослідницького алгоритму:

Загальні параметри:

- *змінна N_ задає розмір блоків (8, 16 і 32), на які поділяється зображення;*
- *змінна P_ величина порога загрублення (аналог матриці заокруглення JPEG).*

Перший етап. Сегментація зображення: розмір блоків N_ × N_ пікселів кожний, де N_ = 8; N_ = 16; N_ = 32.

```
r := for x ∈ 0..(m_ − 1)
       for y ∈ 0..(n_ − 1)
         for i ∈ 0..(N_ − 1)
           for j ∈ 0..(N_ − 1)
             R1[i,j] ← M[i+x·N_, j+y·8]
         r[x,y] ← R1
     r
```

Другий етап. Виконання прямого ДКП над кожним блоком тестового зображення. Алгоритм прямого (T1(V1)) ДКП наведено нижче:

$$T1(V1) := \left| \begin{array}{l} \text{for } i \in 0..(N\_ - 1) \\ \quad \text{for } j \in 0..(N\_ - 1) \\ \quad\quad T1_{i,j} \leftarrow \frac{1}{\sqrt{2N\_}} \cdot C_{-i} \cdot C_{-j} \cdot \frac{1}{P\_} \cdot \sum_{x=0}^{N\_-1} \sum_{y=0}^{N\_-1} \left[V1_{x,y} \cdot \cos\left[\frac{(2 \cdot x + 1) \cdot i \cdot \pi}{2 \cdot N\_} \right] \cdot \cos\left[\frac{(2 \cdot y + 1) \cdot j \cdot \pi}{2 \cdot N\_} \right] \right] \\ T1 \end{array} \right.$$

Функція T1(V1) реалізує пряме ДКП для масиву V1 із N_× N_ ел. В якості аргументу функції T1(V1), використовуються окремі блоки растрових даних у просторовій області.

Змінна C_ містить службовий масив даних із 8 елементів, що необхідні для коректного обчислення прямого та зворотного дискретного косинусного перетворення (ЗДКП).

$$C\_ := \left| \begin{array}{l} \text{for } i \in 0..(N\_ - 1) \\ \quad C_{-i} \leftarrow \frac{1}{\sqrt{2}} \text{ if } i = 0 \\ \quad C_{-i} \leftarrow 1 \text{ if } i > 0 \\ C\_ \end{array} \right. \qquad C\_ = \begin{pmatrix} 0.707 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Третій етап. Відбір значущих елементів матриць ДКП: - *налаштування I, II, III та IV*.

Важливо підкреслити, що в межах проведеного моделювання у кожній з матриць, що отримані після проведення ДКП, в обов'язковому порядку, зберігаються три кутові елементи з координатами: (0;0), (0;1) та (1;0). Для решти області трансформант, обчислюється середнє значення амплітуди складових коефіцієнтів кожного блоку (K_{cp}) за формулою S_(H1), змінна tr1 містить середні значення всіх блоків:

$$S\_(H1) := \frac{\left[\sum_{x=0}^{N\_-1}\sum_{y=0}^{N\_-1}\left(\left|H1_{x,y}\right|\right)\right]}{\left(N\_^{2}\right)}$$

$$tr1 := \left|\begin{array}{l}\text{for } i \in 0..(m\_-1)\\ \quad \text{for } j \in 0..(n\_-1)\\ \quad\quad tr1_{i,j} \leftarrow S\_\left(tr_{i,j}\right)\\ tr1\end{array}\right.$$

Налаштування № I:

$$tr := \left|\begin{array}{l}\text{for } i \in 0..(m\_-1)\\ \quad \text{for } j \in 0..(n\_-1)\\ \quad\quad \text{for } k \in 0..(N\_-1)\\ \quad\quad\quad \text{for } h \in 0..(N\_-1)\\ \quad\quad\quad\quad \text{continue if } h=0 \wedge k=0\\ \quad\quad\quad\quad \text{continue if } h=1 \wedge k=0\\ \quad\quad\quad\quad \text{continue if } h=0 \wedge k=1\\ \quad\quad\quad\quad \left(tr_{i,j}\right)_{k,h} \leftarrow 0 \text{ if } \left|\left(tr_{i,j}\right)_{k,h}\right| < tr1_{i,j}\\ tr\end{array}\right.$$

Налаштування № II:

$$tr := \left|\begin{array}{l}\text{for } i \in 0..(m\_-1)\\ \quad \text{for } j \in 0..(n\_-1)\\ \quad\quad \text{for } k \in 0..(N\_-1)\\ \quad\quad\quad \text{for } h \in 0..(N\_-1)\\ \quad\quad\quad\quad \text{continue if } h=0 \wedge k=0\\ \quad\quad\quad\quad \text{continue if } h=1 \wedge k=0\\ \quad\quad\quad\quad \text{continue if } h=0 \wedge k=1\\ \quad\quad\quad\quad \left(tr_{i,j}\right)_{k,h} \leftarrow \text{round}\left[\frac{\left(tr_{i,j}\right)_{k,h}}{\left(\frac{tr1_{i,j}}{4}\right)}\right]\\ tr\end{array}\right.$$

Налаштування № III:

$$tr := \left|\begin{array}{l}\text{for } i \in 0..(m\_-1)\\ \quad \text{for } j \in 0..(n\_-1)\\ \quad\quad \text{for } k \in 0..(N\_-1)\\ \quad\quad\quad \text{for } h \in 0..(N\_-1)\\ \quad\quad\quad\quad \text{continue if } h=0 \wedge k=0\\ \quad\quad\quad\quad \text{continue if } h=1 \wedge k=0\\ \quad\quad\quad\quad \text{continue if } h=0 \wedge k=1\\ \quad\quad\quad\quad \left(tr_{i,j}\right)_{k,h} \leftarrow \text{round}\left[\frac{\left(tr_{i,j}\right)_{k,h}}{tr1_{i,j}}\right]\\ tr\end{array}\right.$$

Налаштування № IV:

```
tr := | for i ∈ 0..(m_ − 1)
      |   for j ∈ 0..(n_ − 1)
      |     for k ∈ 0..(N_ − 1)
      |       for h ∈ 0..(N_ − 1)
      |         continue if | h = 0
      |                     | k = 0
      |         (tr_{i,j})_{k,h} ← (tr_{i,j})_{k,h}.0
      | tr
```

Четвертий етап. Виконання ЗДКП над кожним блоком.

Алгоритм зворотного (T1(V1)) ДКП:

$$V1(T1) := \left|\begin{array}{l} \text{for } i \in 0..(N\_ - 1) \\ \quad \text{for } j \in 0..(N\_ - 1) \\ \quad V1_{i,j} \leftarrow \text{round}\left[\sum_{x=0}^{N\_-1}\sum_{y=0}^{N\_-1}\left[T1_{x,y}\cdot P\_\cdot\frac{1}{\sqrt{2N\_}}\cdot C\_{x}\cdot C\_{y}\cdot\cos\left[\frac{(2\ i+1)\cdot x\cdot\pi}{2\cdot N\_}\right]\cdot\cos\left[\frac{(2\cdot j+1)\cdot y\cdot\pi}{2\cdot N\_}\right]\right]\right] \\ V1 \end{array}\right.$$

Функція V1(T1) реалізує зворотне дискретно-косинусне перетворення масиву T1 з N_×N_ чисел. В якості аргументу функції V1(T1) використовуються окремі блоки даних в частотній області (*матриці трансформант*).

Змінна C_ містить службовий масив даних із восьми елементів, аналогічній до того, що використовувався в алгоритмі ДКП.

П'ятий етап. Відтворення зображення: складання всіх блоків в єдине ціле:

```
M2 := | for x ∈ 0..(m_ − 1)
      |   for y ∈ 0..(n_ − 1)
      |     temp1 ← tr_{x,y}
      |     for i ∈ 0..(N_ − 1)
      |       for j ∈ 0..(N_ − 1)
      |         temp2_{x·N_+i, y·N_+j} ← temp1_{i,j}
      | temp2
```

4 Результати моделювання

Продемонструємо результати досліджень для блоків, розміром 8×8 елементів.

I. **Налаштування № I**.

1. Встановлені параметри кодування: N_ = 8 та P_ = 1:

$$tr_{1,2} = \begin{pmatrix} 711.125 & -0.841 & 3.413 & -5.578 & 2.875 & -0.886 & 0.074 & -0.282 \\ 0.616 & 5.434 & -6.538 & 4.59 & -4.455 & 2.543 & -1.212 & 0.239 \\ 1.51 & 1.225 & -2.481 & -0.558 & 1.995 & -1.537 & -1.399 & 2.392 \\ 2.966 & -0.888 & -6.967 & 8.714 & -6.088 & 2.714 & -2.112 & 2.026 \\ -1.125 & 4.58 & -3.605 & 1.454 & -0.375 & 0.308 & -0.536 & 0.709 \\ -4.027 & 3.247 & -1.709 & 0.153 & 0.614 & -0.696 & 1.055 & -0.626 \\ 0.051 & -0.138 & -0.899 & 2.071 & -2.044 & 0.823 & 1.231 & -1.412 \\ -0.646 & 0.678 & -0.02 & 0.194 & -0.522 & 0.317 & 0.794 & -0.951 \end{pmatrix} \quad tr_{1,2} = \begin{pmatrix} 711.125 & -0.841 & 3.413 & -5.578 & 2.875 & -0.886 & 0.074 & -0.282 \\ 0.616 & 5.434 & -6.538 & 4.59 & -4.455 & 2.543 & -1.212 & 0.239 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Рис. 1. Матриця коефіцієнтів після проведення ДКП та після порівняння з $K_{ср}$ ***
(*для блоку з координатами 1,2*)

*** - *які більші або дорівнюють середньому значенню амплітуди коефіцієнтів ($K_{ср}$).*

trl =

	0	1	2	3	4	5	6	7
0	11.828	12.365	15.713	19.366	25.895	29.273	25.272	13.061
1	12.395	12.023	13.029	23.756	23.976	32.297	27.635	12.848
2	12.543	12.618	12.781	29.036	19.727	22.367	29.209	13.484
3	12.16	12.151	12.334	25.471	24.09	23.878	33.792	16.127
4	12.452	11.928	12.051	19.399	17.378	22.299	28.449	25.809
5	12.663	12.467	12.44	18.833	23.617	32.222	40.814	44.306
6	12.167	12.285	12.329	17.898	19.119	33.818	41.911	32.093
7	12.427	12.169	12.518	16.826	21.397	27.928	19.988	28.244
8	12.012	11.734	11.895	12.075	19.483	19.642	24.185	26.918
9	12.406	11.675	11.937	12.022	23.939	19.235	18.89	22.768
10	11.812	11.481	11.68	11.855	22.122	12.77	15.706	27.481
11	11.245	11.091	11.03	11.185	14.939	14.641	12.876	22.616
12	10.239	10.551	10.789	10.861	18.685	17.912	13.284	32.166
13	10.512	10.619	10.636	10.677	24.224	14.486	15.601	18.764
14	10.658	11.184	10.9	10.764	17.033	21.093	23.237	17.315
15	11.283	10.894	10.929	11.016	15.093	16.127	10.726	...

Рис. 2 – Масив значень $K_{ср}$ для всіх блоків (*8×8 елементів*) тестового зображення

M2

Рис. 3 – Відновлене зображення для *N_ = 8* та *P_ = 1*

M2

Рис. 4 – Відновлене зображення для *N_ = 8* та *P_ = 5*

M2

Рис. 5 – Відновлене зображення для *N_ = 8* та *P_ = 10*

II. Налаштування № II.

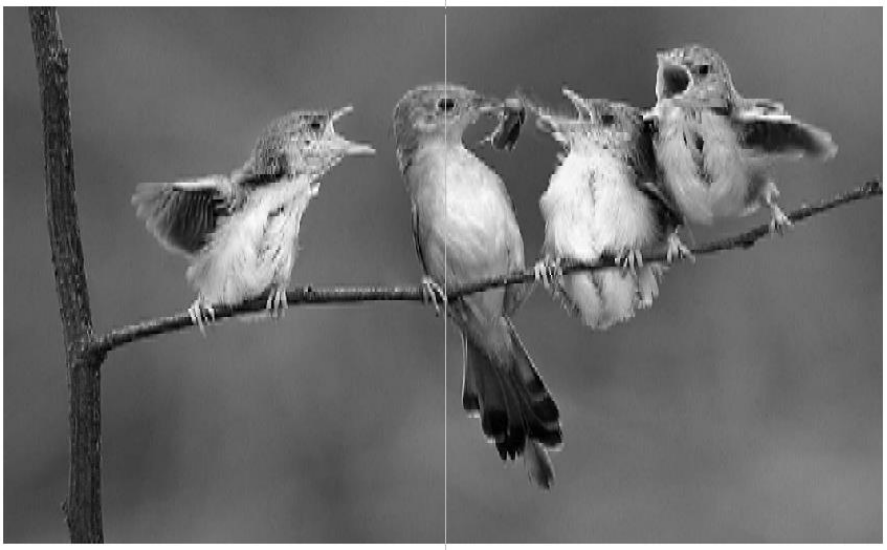


Рис. 6 – Відновлене зображення для *N_ = 8*; *P_ = 1*

Рис. 7 – Відновлене зображення для *N_ = 8*; *P_ = 5*

Рис. 8– Відновлене зображення для *N_ = 8*; *P_ = 10*

III. Налаштування № III.

Рис. 9 – Відновлене зображення для *N_ = 8; P_ = 1*

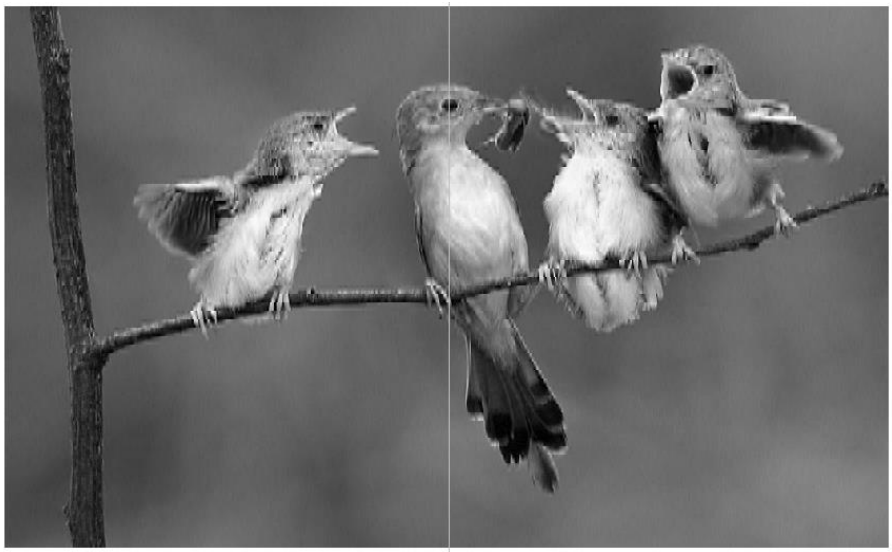


Рис. 10 – Відновлене зображення для *N_ = 8; P_ = 5*

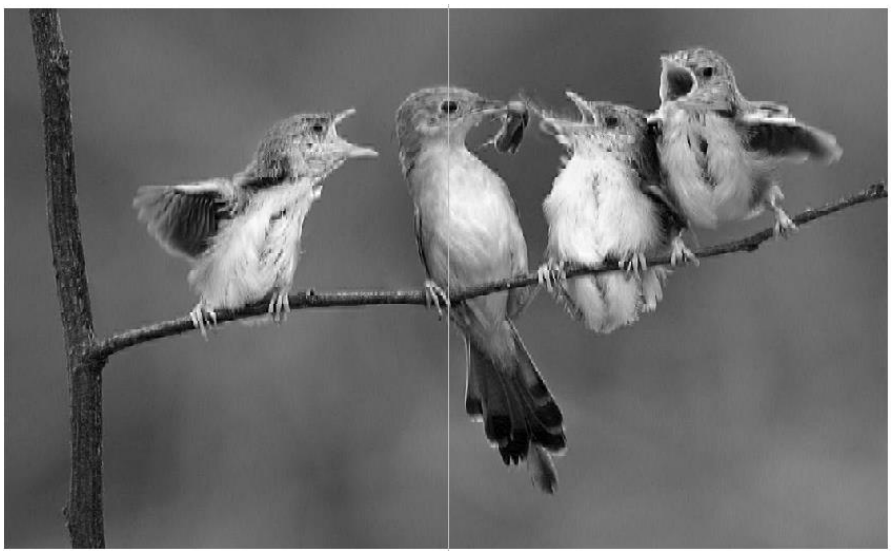


Рис. 11 – Відновлене зображення для *N_ = 8; P_ = 10*

IV. Налаштування № IV.

При застосуванні цього варіанта налаштувань, використання параметра «*P*» немає сенсу, так як для відновлення вихідних матриць субблоків (ЗДКП), використовується лише значення середньої яскравості блоку (0;0). Тобто, проводячи аналогію налаштувань алгоритму, результати використання котрих представлені на Рис. 3-11, у даному випадку, маємо абсолютно однаковий результат для всіх можливих значень параметру «*P*» (див. Рис. 12).

a)

b)

Рис. 12 – Вихідне *(a)* та відновлене *(b)* зображення
для IV–го варіанту налаштувань при *N_ = 8*

При подальшому збільшенні розмірності субблоків (*16 та 32*), загальна тенденція процесу, яка характерна для розмірності блоків 8×8, також зберігається. При цьому, візуальна помітність спотворень зображень, які відновлюються, при малих значеннях коефіцієнта загрублення (*P=1÷5*), знаходиться в прийнятних рамках, причому для всіх 3-х варіантів налаштувань. Це підтверджує хорошу "чутливість" (*адаптивність*) порогового способу відбору коефіцієнтів до локальної статистики оброблюваних блоків зображень.

В якості прикладу наведемо (Рис. 13-15) характерні результати роботи моделюючого алгоритму для блоків розмірністю 32×32 (*Налаштування № I, коеф. загрублення: P = 1*).

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	937.287	179.949	25.545	-78.389	8.68	8.089	21.672	-15.322	-11.726	6.239	7.399	-7.58	4.287	1.92	-9.846	0.89
1	0.449	-20.336	45.823	-14.656	-6.188	-22.335	32.565	-11.099	8.496	-13.326	-4.698	6.562	11.646	-10.175	-11.716	14.023
2	-28.382	31.59	-27.505	30.07	-13.29	6.655	-15.644	22.107	5.405	-15.232	-1.081	7.091	-0.633	0.595	-4.071	-4.867
3	-19.186	15.137	3.383	9.809	-21.253	17.405	-12.495	14.04	-16.507	9.861	6.619	-9.405	-5.299	25.38	-8.911	-4.304
4	-8.201	8.117	-10.347	8.337	-0.866	7.226	-22.456	21.259	-6.154	1.416	0.627	-4.772	6.332	-1.372	4.976	-6.263
5	-6.552	10.146	-17.436	16.782	-3.375	-9.959	7.551	-12.547	19.964	-20.704	12.862	-6.014	3.892	-5.124	4.238	8.932
6	3.052	-12.155	11.043	3.07	-2.334	-7.905	12.482	-12.832	14.22	-9.868	9.293	-15.468	16.312	-11.947	7.979	-2.155
7	-7.859	-5.317	19.785	-10.379	-3.661	-2.064	8.467	-7.096	-6.251	16.711	-19.592	13.554	-9.75	6.569	1.72	-10.481
8	5.098	-11.389	16.52	-4.799	-4.112	5.814	-11.183	21.998	-21.23	14.703	-14.04	19.176	-6.888	-9.801	10.286	2.962
9	-19.025	22.307	-19.238	19.804	-16.773	7.36	-4.088	2.69	-3.739	-0.62	7.624	-11.074	8.112	0.31	-0.584	-2.307
10	6.468	-7.8	7.463	-7.285	5.519	-4.063	-4.516	12.343	-15.277	13.43	-5.365	-3.038	11.835	-10.695	9.502	-6.352
11	9.093	-15.222	13.403	-2.945	-6.635	11.782	-8.048	2.48	5.854	-10.597	5.194	5.16	-14.772	10.214	-6.925	1.634
12	-7.614	8.86	-0.28	-1.295	-3.782	3.961	1.316	-6.267	3.681	0.666	-4.467	6.701	-3.506	-1.573	7.348	-6.554
13	-5.517	10.778	-13.411	11.175	-7.7	5.298	-3.401	0.161	-1.394	-3.98	3.509	4.495	-5.033	1.357	-2.676	8.218
14	10.069	-10.99	5.703	-10.118	11.843	-9.636	8.252	-7.762	7.387	-4.383	0.809	-5.122	14.113	-9	-12.57	16.045
15	10.083	-17.539	17.233	-17.778	14.764	-10.425	6.478	-4.221	-0.75	7.822	-7.612	3.084	-7.474	12.208	-3.363	-9.172

Рис. 13 – Фрагмент *(16×16)* матриці коефіцієнтів після проведення ДКП *(блок 1;2)*

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	937.287	179.949	25.545	-78.389	8.68	8.089	21.672	-15.322	-11.726	6.239	7.399	-7.58	4.287	1.92	-9.846	0.89
1	0.449	-20.336	45.823	-14.656	-6.188	-22.335	32.565	-11.099	8.496	-13.326	-4.698	6.562	11.646	-10.175	-11.716	14.023
2	-28.382	31.69	-27.505	30.07	-13.29	0	-15.644	22.107	0	-15.232	0	7.091	0	0	0	0
3	-19.186	15.137	0	9.809	-21.253	17.405	-12.495	14.04	-16.507	9.861	0	-9.405	0	25.38	-8.911	0
4	-8.201	8.117	-10.347	8.337	0	7.226	-22.456	21.259	0	0	0	0	0	0	0	0
5	0	10.146	-17.436	16.782	0	-9.959	7.551	-12.547	19.964	-20.704	12.862	0	0	0	0	8.932
6	0	-12.165	11.043	0	0	-7.905	12.482	-12.832	14.22	-9.868	9.293	-15.468	16.312	-11.947	7.979	0
7	-7.859	0	19.785	-10.379	0	0	8.467	-7.096	0	16.711	-19.592	13.554	-9.75	0	0	-10.481
8	0	-11.389	16.52	0	0	0	-11.183	21.998	-21.23	14.703	-14.04	19.176	-6.888	-9.801	10.286	0
9	-19.025	22.307	-19.238	19.804	-16.773	7.36	0	0	0	0	7.624	-11.074	8.112	0	0	0
10	0	-7.8	7.463	-7.285	0	0	0	12.343	-15.277	13.43	0	0	11.835	-10.695	9.502	0
11	9.093	-15.222	13.403	0	0	11.782	-8.048	0	0	-10.597	0	0	-14.772	10.214	-6.925	0
12	-7.614	8.86	0	0	0	0	0	0	0	0	0	0	0	0	7.348	0
13	0	10.778	-13.411	11.175	-7.7	0	0	0	0	0	0	0	0	0	0	8.218
14	10.069	-10.99	0	-10.118	11.843	-9.636	8.252	-7.762	7.387	0	0	0	14.113	-9	-12.57	16.045
15	10.083	-17.539	17.233	-17.778	14.764	-10.425	0	0	0	7.822	-7.612	0	-7.474	12.208	0	-9.172

Рис. 14 – Фрагмент *(16×16)* матриці коефіцієнтів блоку *(1;2)* після порівняння з K_{cp}

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4.587	5.954	7.092	8.058	7.38	6.563	4.814	1.932	1.752	1.72	1.725	1.776	1.811	1.821	1.795	1.741
1	2.632	4.344	6.758	8.911	10.083	9.832	8.694	6.22	1.818	1.722	1.811	1.838	1.888	1.906	1.868	1.854
2	1.544	3.441	4.42	5.252	6.239	5.915	5.519	4.71	1.807	1.718	1.771	1.771	1.809	1.797	1.802	1.819
3	1.424	2.956	4.3	4.89	5.799	5.935	5.395	4.861	2.45	1.714	1.767	1.774	1.795	1.807	1.776	1.804
4	1.423	2.39	4.227	4.696	5.911	6.119	5.399	5.053	3.257	1.582	1.676	1.705	1.762	1.767	1.765	1.761
5	1.494	1.709	3.535	4.245	4.63	5.719	5.304	4.953	4.434	1.627	1.585	1.599	2.433	2.928	3.32	3.892
6	1.578	1.636	3.497	4.591	5.123	6.507	6.334	5.77	5.363	2.629	1.656	1.68	1.854	2.288	2.799	3.352
7	1.632	1.644	2.99	4.461	5.598	6.543	6.894	6.589	5.695	4.288	1.824	1.659	1.663	1.671	1.7	1.876
8	1.643	1.641	1.712	3.385	4.494	5.585	6.371	6.236	5.806	4.578	2.68	1.672	1.666	1.662	1.679	1.683
9	1.644	1.639	1.661	2.344	4.083	4.838	6.67	8.43	8.482	8.868	8.784	7.476	7.913	7.53	7.507	7.34
10	1.577	1.62	1.638	1.65	3.947	6.425	7.301	7.738	8.027	7.206	6.833	6.387	4.728	3.476	2.689	1.692
11	1.531	1.536	1.585	1.628	2.363	3.275	3.733	4.281	4.552	4.474	4.166	3.447	1.479	1.477	1.507	1.527
12	1.586	1.571	1.583	1.564	2.208	3.466	3.998	4.273	4.888	4.572	4.282	3.808	1.347	1.332	1.351	1.379
13	1.424	1.441	1.461	1.43	1.675	3.167	3.468	3.876	4.732	4.332	4.118	3.733	1.256	1.233	1.284	1.316
14	1.371	1.355	1.385	1.638	3.288	3.994	4.096	4.238	3.626	3.335	3.201	2.747	1.412	1.461	1.508	1.519
15	1.506	1.491	1.536	1.619	3.401	3.975	4.124	4.43	4.26	4.38	4.392	3.873	1.77	1.752	1.825	1.871

Рис. 15 – Розраховані значення K_{cp} для всіх блоків кадру зображення (*фрагмент*)

3 Висновки

Застосування методів кодування з перетворенням, як основи для синтезу відповідних алгоритмів стиску відеоданих різного типу (*нерухомі та динамічні зображення; повнокольорові та напівтонові; із простою та складною структурою тощо*), базується на концепції максимального використання особливостей зорової системи людини, щодо характеристик сприйняття візуальній інформації для виявлення мінімальної кількості даних, які необхідні для цілей наступної однозначної ідентифікації, передачі та/або зберігання цієї інформації, при заданих умовах/критеріях її спостереження (*час експозиції, відстань спостереження, частота зміни сцен, розмірність і співвідношення основних об'єктів спостереження та ін.*).

В якості ключового компонента дослідного алгоритму використовувалося ДКП, яке практично з однаковою ефективністю можна застосовувати для всіх типів зображень [2-4]. Воно дозволяє переходити від просторового до спектрального представлення оброблюваних зображень, та забезпечує хорошу спроможність до концентрації більшої частині значущої інформації в меншій, у порівнянні з іншими перетвореннями, кількості відліків *(коеф. перетворення)*. Ця властивість ДКП створює хороші стартові умови для проведення всіх наступних процедур стиснення зображень та/або інкапсуляції стеганоконтенту [3-8, 11].

Впливаючи на спектральне уявлення оброблюваних зображень, можна балансувати між якістю відтворення та ступенем стиску (*або ступенем допустимих спотворень*) вихідного зображення, а впливаючи на розмір субблоків і параметри їх попередньої обробки [7-8], на загальний час обробки зображення.

При реалізації систем стиску відеоданих на основі методів кодування з перетворенням, основними параметрами, які безпосередньо впливають на ефективність реалізації процесу зменшення обсягів цифрового опису вхідних зображень виступають: - використовуваний розмір блоків, на які поділяється вихідне зображення; - спосіб відбору значимих коефіцієнтів перетворення; - використовуваний спосіб/принцип квантування значимих коефіцієнтів. Причому розмір субблоків, на які поділяється вихідне зображення, є визначальним, адже саме від нього, в значній мірі, залежить загальний час, що необхідний для обчислення кожного елемента матриці ДКП *(використовуються два вкладені цикли)* [12].

Відповідно до отриманих результатів моделювання, при обробці всіх типів зображень, підтверджено загальні тенденції спостережуваних процесів, що полягають в наступному: - із збільшенням розміру субблоків (*в нашому випадку квадратні матриці розмірністю 8×8;16×16 і 32×32 ел.*) уповільнюється процес обробки зображення; - із збільшенням розміру субблоків, плавно зростає і коефіцієнт стиску; - зменшується загальна кількість подібних блоків зображень [7-8]; - збільшується діапазон міжблокової різниці, середньої амплітуди коефіцієнтів трансформант ($K_{ср}$), одержуваних після проведення ДКП (*Рис. 2, 15*); - зростання розмірності субблоків, в певній мірі, компенсує процес загрублення порогових значень селекції коефіцієнтів перетворення (*Рис. 13-14*); - погіршуються вихідні умови для механізмів каскадної комбінаторики наступної стегановставки; - пороговий метод селекції значущих коефіцієнтів ДКП, забезпечує високі параметри адаптації до локальної статистики оброблюваних блоків зображень (*Рис. 15*); - пороговий спосіб відбору коефіцієнтів забезпечує досить хорошу спроможність до компенсації можливих наслідків від невдалих настроювань параметрів квантування, коефіцієнтів що зберігаються (*див. Рис. 3-5; 6-8; 9-11*); - подальша обробка масиву значень середніх амплітуд коефіцієнтів трансформант надає досить широкий спектр можливостей, як для оптимізації процедур стиску зображень, так і для реалізації певних процедур наступної стегановставки; - обробка трансформант при збереженні тільки кутових значень (*Рис.12*), може розглядатися, як варіант для обробки областей міжкадрової різниці динамічних зображень або для цілей попередньої візуалізації зображень (*тобто без їх повного розпакування*) в малоресурсних алгоритмах, що адаптовані до умов використання різних мобільних платформ.

Процес квантування, отриманих після проведення ДКП, значень є дуже важливим етапом у всьому ланцюжку процедур алгоритму стиснення зображень. В загальному сенсі, дана

процедура представляє собою процес зменшення кількості бітів, необхідних для зберігання коефіцієнтів матриці ДКП за рахунок втрати їх точності. В межах даної роботи для цього використовувався, відповідний коефіцієнт загрублення – *«P»*. Після проведення ДКП для всіх наявних блоків здійснюється послідовне нормування отриманих елементів матриць. Ця процедура реалізується шляхом ділення отриманих значень на задану константу *«P»*, з подальшим округленням результату до цілих. Як підтвердили результати моделювання пороговий метод селекції коефіцієнтів трансформант, при всіх варіантах налаштувань параметрів порогового відбору, забезпечує високі параметри візуалізації тестових зображень (*див. Рис. 3-5; 6-8; 9-11*) при зміні параметру *«P»* в діапазоні від 1 до 10.

Посилання

[1] N. Chelemal-D and K. R. Rao // Fast computational algorithms for the Discrete Cosine Transform // 9th Annual Asilomar Conf. on Circuit, Systems and Computers, Pacific Grove, CA, Nov. 1985.

[2] Красильников Н.Н. Статистическая теория передачи изображений. - М.: Связь, 1976. – 184 с.

[3] Прэтт У. Цифровая обработка изображений. т. 1,2. - М.: Мир, 1985. - 736 с.

[4] Зубарев Ю.Б., Дворкович В.П. Цифровая обработка телевизионных и компьютерных изображений. - М.: МЦНТИ, 1997. – 212 с.

[5] Королев А.В. Оценка информативности трансформант дискретного косинусного преобразования. Системи Обробки Інформації. 2003. № 3. С. 81-86.

[6] About JPEG // Електронний ресурс. Режим доступу – http://www.pcs-ip.eu/index.php/main/edu/5.

[7] Morozov, D., Shaforostov, M., Malakhov, S., & Serbin, V. (2018). Подвійна обфускація трансформант малоресурсного стеганоалгоритма. Комп'ютерні науки та кібербезпека, 9(1), 22-34. вилучено із https://periodicals.karazin.ua/cscs/article/view/12015

[8] Гончаров М. О. Дослідження процедур попередньої підготовки вихідних даних для стеганоалгоритма. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління*: матеріали 11-ї міжнар. наук.-техн. конф., 8–9 квіт. 2021 р. м. Баку–Харків–Київ–Жиліна. т. 2. с. 63. URL: http://repository.kpi. kharkov.ua/handle/KhPI-Press/52020 (дата звернення 1.05.2021).

[9] Рубан И. В., Клименко Л. А. Сжатие видеоданных длинами серий с загрублением цвета. *Информатика.* Киев: Наукова думка, 1998. № 5. С. 44-48.

[10] The Discrete Cosine Transform // Електронний ресурс. Режим доступу – https://users.cs.cf.ac.uk/Dave.Marshall/Multimedia/node231.html

[11] Кузнецов О. О., Полуяненко М. О., Кузнецова Т. Ю. Приховування даних у частотній області нерухомих зображень на основі кодування різниці абсолютних значень коефіцієнтів дискретного косинусного перетворення // Методичні рекомендації.

[12] Мастрюков Д. Краткое объяснение JPEG и его реализации. Алгоритмы сжатия информации. Часть 7. Сжатие графической информации.

[13] Kasperovich L.V. Multiplication free scaled 8 x 8 ДКП algorithm with 530 additions // Електронний ресурс. Режим доступу – https://www.spiedigitallibrary.org/conference-proceedings-of-spie/2419/0000/Multiplication-free-scaled-8-x-8-ДКП-algorithm-with-530/10.1117/12.206390.full?SSO=1.

[14] Douglas A. Kerr // Chrominance Subsampling in Digital Images // Електронний ресурс. Режим доступу – http://dougkerr.net/Pumpkin/articles/Subsampling.pdf.

[15] Hou H. A Fast Recursive Algorithm For Computing The Discrete Cosine Transform // Електронний ресурс. Режим доступу – https://www.researchgate.net/publication/3235321_A_Discrete_Fourier-Cosine_Transform_Chip.

Рецензент: Владимир Хома, д.т.н., проф., Опольский Политехнический Университет, Ополе, Польша.
E-mail: xoma@wp.pl

Поступила: Ноябрь 2021.

Авторы:

Екатерина Кузнецова, студентка кафедры безопасности информационных систем и технологий, ХНУ имени В. Н. Каразина, Харьков, Украина. E-mail: kate7smith12@gmail.com

Елизавета Кузнецова, студентка кафедры безопасности информационных систем и технологий, ХНУ имени В. Н. Каразина, Харьков, Украина. E-mail: kuznetsova2021kb11@student.karazin.ua

Сергей Малахов, к.т.н., с.н.с., доцент факультета компьютерных наук, ХНУ имени В. Н. Каразина, Харьков, Украина.
E-mail: mailgate@meta.ua

Моделирование основных процедур сжатия изображений с частичной потерей информации, на примере методов кодирования с преобразованием.

Аннотация. Применение группы методов кодирования с преобразованием, как основы для синтеза универсальных алгоритмов сжатия видеоданных, базируется на концепции максимального использования особенностей зрительной системы человека: - время экспозиции; - расстояние наблюдения; - величина межкадровой разницы; - размерность и соотношение

основных объектов наблюдения; - тип изображений и др. Кодирование с преобразованием позволяет переходить от пространственного к спектральному представлению обрабатываемых изображений и обеспечивает широкие возможности для их последующего сжатия и/или стеганографической обработки.

В качестве ключевого компонента исследовательского алгоритма, использовалось дискретное косинусное преобразование (ДКП). Оно является универсальным практически для всех типов изображений. ДКП обеспечивает хорошую способность к концентрации большей части значимой информации, в меньшем (по сравнению с другими преобразованиями) количестве коэффициентов. Это свойство ДКП создает хорошие стартовые условия для всех следующих процедур сжатия изображений. Влияя на спектральное представление обрабатываемых изображений, можно балансировать между качеством воспроизведения и степенью сжатия исходного изображения, а влияя на размер субблоков и параметры их предварительной обработки, на общее время выполнения операций.

При использовании методов кодирования с преобразованием, основными параметрами, влияющими на эффективность процесса сжатия, являются: - используемый размер субблоков, на которые делится исходное изображение; - метод отбора значимых коэффициентов; - используемый принцип квантования важных коэффициентов. Размер субблоков является определяющим, т.к. от него зависит количество подобных блоков и общее время вычисления каждого элемента матриц ДКП. Важным этапом является процесс округления, полученных после ДКП значений. В исследовании для этого используется коэффициент загрубления. Процедура нормирования элементов матриц реализуется путем деления значений ДКП на заданную константу загрубления с последующим округлением результата до целых чисел.

Целью работы является моделирование основных этапов процесса сжатия статических изображений и исследование влияния основных параметров кодирования на качественно-временные характеристики воспроизводимых изображений (время обработки, вычислительная сложность, качество воспроизведения, характеристики искажений и т.п.).

Ключевые слова: методы сжатия изображений; кодирование с преобразованием; ДКП; отбор коэффициентов; искажения; квантование коэффициентов; визуальная заметность.

Reviewer: Volodymyr Khoma, Dr. of Sciences (Eng.), Full Prof., The Opole University of Technology, Opole, Poland.
E-mail: xoma@wp.pl

Received: November 2021.

Authors:
Kateryna Kuznetsova, student of the Department of Security of Information Systems and Technologies, Kharkiv National University named after V.N. Karazin, Kharkiv, Ukraine.
E-mail: kate7smith12@gmail.com

Yelyzaveta Kuznetsova, student of the Department of Security of Information Systems and Technologies, Kharkiv National University named after V.N. Karazina, Kharkiv, Ukraine.
E-mail: kuznetsova2021kb11@student.karazin.ua

Serhii Malakhov, Ph.D., Senior Researcher, Computer Science Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine. E-mail: mailgate@meta.ua

Modeling of basic image-lossy compression procedures using transformation coding methods, as an example.

Annotation. Using the set of transformation coding methods as the ground to synthesize of universal video compression algorithms is based on using the features of the human visual system: - exposure time; - observation distance; - the value of the interframe difference; - dimension and ratio of main observed objects; - type of images, etc. Transform coding allows transiting from spatial to spectral representation of the processed images and provides ample opportunities for their subsequent compression and/or steganographic processing.

The main component of the research algorithm is the discrete cosine transform (DCT). It is universal for almost all types of images. DCT provides a good ability to concentrate most of the relevant information in a smaller (compared to other transformations) number of coefficients. This DCT property creates good initial conditions for all the following image compression procedures. By influencing the spectral representation of the processed images, it is possible to balance between the reproduction quality and the degree of compression of the original image, and by influencing the size of the sub-blocks and the parameters of their preprocessing, it is possible to reduce the total time of operations proceed.

When using transform coding methods, the main parameters affecting the efficiency of the compression process are: - the used size of the sub-blocks into which the original image is divided; - the method of selecting significant coefficients; - the used principle of quantizing important coefficients. The size of the sub-blocks is decisive, since the number of such blocks and the total computation time for each element of the DCT matrices depend on it. An important step is the rounding process of the values obtained after DCT. The study uses a coarsening factor for this. The procedure for normalizing matrix elements is implemented by dividing the DCT values by a given coarsening constant, followed by rounding the result to the integer numbers.

The aim of the work is to simulate the main steps of the compression of static images and study the influence of the main coding parameters on the qualitative and temporal characteristics of the reproduced images (processing time, computational complexity, reproduction quality, distortion characteristics, etc.).

Keywords: Image compression methods; Transform encoding; DCT; Selection of coefficients; Distortion; Quantization of coefficients; Visual visibility.

УДК 621.327:621.391 DOI: 10.26565/2519-2310-2021-2-05

ДОСЛІДЖЕННЯ ВЛАСТИВОСТЕЙ ПРОТОТИПУ ГІБРИДНОГО СТЕГАНОАЛГОРИТМУ

Микита Гончаров, Юлія Лєсная, Сергій Малахов

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
worldxdark@gmail.com, xa12284109@student.karazin.ua, mailgate@meta.ua

Рецензент: В'ячеслав Калашников, д.ф.-м.н., проф., Технологічний університет Монтеррея, 64849 Монтеррей, Нуево-Леон, Мексика
kalash@itesm.mx

Надійшла: Листопад 2021.

Анотація: *Метою даного матеріалу є ознайомлення з основними етапами адаптивного малоресурсного алгоритму стеганографічної обробки зображень і результатами моделювання процедур попередньої обробки вихідних даних різних типів. Процедури імітаційного алгоритму дозволяють: - враховувати особливості оброблюваних даних (типи контейнера і контенту) і коригувати параметри роботи основних модулів стеганоалгоритму (модуль попередньої обробки вхідних даних та модуль спеціальних перетворення). Досліджено інші параметри обробки зображень, які безпосередньо впливають на обчислювальну складність 1-го етапу алгоритму (згладжування) та якість візуалізації контейнера та вмісту. Зазначено, що для всіх типів зображень, варіант попереднього згладжування вхідних блоків, за принципом «перебір всіх з усіма», надає кращі результати. В даному випадку, зі зменшенням розмірності матрищь згладжування, інтенсивність візуально помітних аномалій зменшується.*

Підкреслено, що при збільшені значення порога загрублення яскравості елементів (P_Z), кількість і помітність артефактів зростає. Зростання фіксується для всіх типів зображень і всіх варіантів їх попереднього згладжування. Стійке зростання викривлень відбувається при виборі значень P_Z більш 7. Для реалістичних зображень, критично припустимою слід вважати величину P_Z = 14. Звернено увагу на те, що в незалежності від встановлених значень P_Z та обраного варіанту згладжування, візуальна помітність викривлень, посилюється в наступній послідовності: «портрет – пейзаж – мнемосхема». Найбільш «чутливими» до варіантів попереднього згладжування, виявилися зображення типу «мнемосхема». Це можна пояснити чутливістю контурів до їх найменших змін та особливостями структури таких зображень.

Звернено увагу на те, що факт малої обчислювальної складності процедур попередньої обробки зображень, є принциповим з урахуванням концепції створення мобільних додатків. Зроблено висновок, що потенційний виграш від введення етапу попередньої обробки вихідних даних, дозволяє отримати 3 важливі ефекти: 1 - знизити обчислювальну складність алгоритму; 2 - використовувати різні принципи обробки даних контейнера та контенту; 3 - створити необхідні умови для асиметричного режиму обробки даних контейнера та контенту.

Ключові слова: *зображення; стеганографія; кодування з перетворенням; контейнер; контент; згладжування зображень; візуальна помітність викривлень; кодування серій; мультиплексування.*

1 Вступ

Добре відомо, що одним із ефективних напрямів забезпечення приховування фактів передачі і зберігання інформації, є застосування різних стеганографічних методів [1]. Цифрова стеганографія, як окремий науковий напрямок, вивчає можливості використання властивостей цифрового контенту різного типу для забезпечення більш ефективного вирішення завдань, які пов’язані с синтезом нових методів і способів прихованої передачі, зберігання та маркування цільової інформації (*надалі контенту*). Принциповим є те, що в незалежності від використаного напрямку стеганографії, необхідно забезпечувати мінімізацію демаскуючих аномалій використовуваних контейнерів та підтримувати заданий рівень стійкості контенту стовно спроб його неавторизованої екстракції, а в деяких випадках, і стійкості до спроб навмисного спотворення контейнерів.

2 Основна частина

При приховувані в цифрових об'єктах будь-якої іншої – корисної інформації, нажаль, виникають певні спотворення цих об'єктів – переносників даних [1]. При збалансованих налаштуваннях відповідного стеганоалгоритму, можливо забезпечувати рівень спотворень кон-

тейнерів, які знаходяться нижче порога чутливості зорової системи людини [2] або чутливості його слухової системи, що забезпечує фактичну відсутність візуально або аудіо помітних змін (аномалій) переносників інформації (*контейнерів*). Тим самим забезпечується необхідний баланс між збереженням характерних властивостей для використовуваного типу контейнерів (*у даному випадку, напівтонових зображень* [2]), та величиною допустимих спотворень, прийнятних для заданого типу прихованого контенту (*в даному випадку зображень з різною вірогідністю перепаду яскравості між сусідніми елементами*). В цьому сенсі зрозуміло, що кількість і інтенсивність проявів різних артефактів зображень контейнеру і контенту, знаходяться в прямий залежності від коректності обраних для них режимів обробки. При цьому, слід мати на увазі, що при обробці даних контейнерів і контенту можуть бути використані, як однотипні (*симетричні*) режими обробки, так і режими, які реалізують різні параметри обробки даних (*асиметричні*). Такими відмінностями можуть бути: – відмінності в розмірах блоків; – відмінності в параметрах попередньої обробки масивів даних контейнерів і контенту *(що є головним змістом даного матеріалу*); – відмінності в критеріях оцінки значущої інформації контейнерів і контенту; – відмінності реалізацій прискорення обчислювальних процедур та інше. Таким чином, на одних і тих же типах даних (*в даному випадку зображень*) можна отримати дуже різний ефект, з точки зору помітності артефактів, що надає зайвий привід задуматися про причини та можливі наслідки появи цих аномалій [1-3].

В основу дослідного алгоритму положено принцип внутрішньо-кадрового стиску зображень, який передбачає поетапне використання двох різних методів кодування: 1 - методу довжин серій [5-6]; 2 - кодування з перетворенням, а саме дискретного косинус перетворення(ДКП) [2-3, 6-7]. Використання властивостей ДКП дозволяє «вбудовувати» контент в матриці коефіцієнтів перетворення контейнерів, визначених алгоритмом попередньої обробки, як опорний блок (*тобто перший в серії із подібних блоків*). Процедура інкапсуляції стеганоконтенту реалізується за рахунок послідовного впровадження двох рівнів обробки: – внутрішньоблоковий (*в межах кожного окремого блоку*) та міжблоковий (*тобто в межах кадру*).

Перший рівень реалізується за результатами проведення відбору коефіцієнтів перетворення [4-7] для контейнера і контенту (*при цьому можливі досить серйозні відмінності в параметрах їх обробки*). На кожному з цих рівнів здійснюється перемішування значущих коефіцієнтів кожного з блоків або цілих блоків, у відповідності з маскою мультиплексу, яка має свої особливості для кожного з рівнів обробки контенту. Використання методів кодування з перетворенням забезпечує отримання матриць спектральних коефіцієнтів, в яких значна частина коефіцієнтів може бути виключена з їх подальшої обробки, а врахування властивостей зорової системи людини і умов спостереження, дозволяє апроксимувати коефіцієнти перетворення з прийнятними (*або заданими*) втратами якості відновлюваних зображень, або ступеню помітності візуальних викривлень зображення-контейнеру [2-3, 6].

Для авторизованої екстракції контенту потрібна інформація о параметрах внутрішньо та міжблокового перемішування коефіцієнтів та/або опорних блоків. У разі її отримання здійснюється відновлення вихідного стану матриць опорних блоків контенту та (тобто *довжин серій опорних блоків*), після чого проводиться зворотне ДКП. Відомості стосовно розмірів блоків, значення параметра маскування коефіцієнтів з координатами (0;0), способу та параметрів селекції значущих коефіцієнтів, та параметрів маски внутрішнього і міжблокового перемішування, складають ключову інформацію для легітимного відновлення контенту.

Для протидії процедурам неавторизованого вилучення контенту, в дослідному алгоритмі використано спрощений механізм міжблокового мультиплексування (*тобто, тільки для опорних блоків контенту*). Згідно з ним, опорні блоки контенту і контейнеру, мають однакову розмірність, та вбудовуються в контейнери не послідовно, а відповідно до варіанту маски шифрування, яка зависить від поточного співвідношення кількості опорних блоків контейнера і контенту, та є варіативним елементом складового ключа для вилучення прихованої інформації. В цьому контексті, варто зазначити, що в межах проведеного моделювання, окремого мультиплексування для елементів трансформант, що визначають середню яскравість

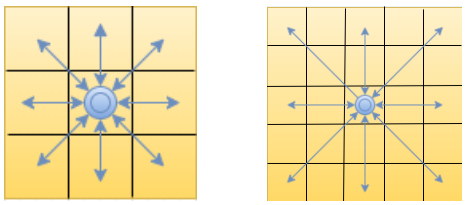
блоків, не проводилося. – Хоча, цей напрямок, формує ще одну, площину для додаткового ускладнення несанкціонованої екстракції даних контенту, так само, як і будь-які маніпуляції із середніми пороговими значеннями коефіцієнтів трансформант, котрі обчислюються для кожної з матриць оброблюваних зображень.

В цілому, всі процедури дослідного алгоритму умовно поділені на окремі етапи, кожен з яких, має своє функціональне значення. Основним змістом даного етапу є дослідження варіантів реалізації процедур попередньої обробки вихідних масивів даних для контейнерів та контенту. Головним завданням етапу, є зменшення кількості візуально помітних перепадів яскравості елементів вихідних зображень, за рахунок попереднього згладжування їх малоінформативних областей. Реалізація процедур перетворення структури вхідних даних дозволяє забезпечити необхідні умови для одночасного вирішення 3-х основних та 1-го додаткового завдань. До основних завдань слід віднести: - зменшення обчислювальної складності алгоритму (*за рахунок зменшення чисельності блоків, потребуючих проведення прямого та зворотнього перетворень*); - ускладнення процедур аналізу і неавторизованої екстракції прихованого контенту (*різні принципи обробки даних контейнера та контенту*); - створення необхідних умов для асиметричного режиму обробки даних контейнера і контенту. До додаткового ефекту слід віднести: - створення сприятливих умов (*в сенсі різниці кількості опорних блоків*) для досягнення бажаного дисбалансу в структурі зображень контейнера і контенту і, як наслідок, покращення стартової ситуації для подальшої реалізації стегановставки.

2.1 Варіанти обробки та отримані результати.

В межах проведеного моделювання були досліджені та скореговані три варіанти реалізації процедур згладжування, з різним порядком взаємного порівняння складових елементів блоків, та різним способом оновлення їх змісту при перевищенні заданих критеріїв подібності (*для контейнера і контенту*). Особливу увагу звернено на важливість локалізації присутності контурів (їх фрагментів) в блоках які обробляються. Для всіх досліджуваних варіантів проводилося розбиття вхідних масивів даних на квадратні блоки (маска згладжування) $m \times m$, де $m = 3, 5$ і 7 елементів [8] . Після цього в кожному блоці, кожного з варіантів, проводилась відповідна оцінка різниці значень яскравості для всіх складових елементів цього блоку. Для всіх варіантів обробки, значення порога загрублення яскравості складових елементів (P_Z), дискретно змінювалося в діапазоні 3, 5, 7, 16, 32 та 64 градацій яскравості. Також всі варіанти згладжування і зміни параметрів *«m»* та *«P_Z»* перевірялись для 3-х типів зображень, з різною ймовірність перепаду яскравості (p) між сусідніми елементами: - мнемосхема (0.03÷0.01), портрет (0.05÷0.03) та пейзаж (0.1÷0.05).

В 1-му варіанті згладжування, здійснювалась оцінка різниці значень яскравості центрального елемента з його периферійним оточенням (Рис. 1). У разі, якщо різниця менше заданого значення порога загрублення (P_Z), то елемент в цій позиції заміщався значенням яскравості центрального елементу [4, 9].



а) *б)*

Рис. 1 – Варіанти обробки елементів
а) для маски 3×3;
б) для маски 5×5.

На рис. 3 наведено результати використання першого варіанту згладжування для маски згладжування 5×5 елементів (Рис. 1) для тестового зображення типу портрет (Рис. 2). З аналізу Рис. 3 видно, що дана реалізація процедур згладжування вимагає певного доопрацювання, оскільки на зображенні візуально помітні певні дефекти (*уривчастість ліній і розмитість*).

Згідно другого варіанту обробки [4, 9], якщо периферійні значення перевищують яскравість центрального елемента більш ніж на величину P_Z, , то виконуємо заміну всіх значень даного блоку на середнє значення яскравості елементів для цього блоку. В результаті, отримуємо поліпшену версію 1-го варіанту (див. Рис. 4).

Рис. 2 – Вихідне зображення
(типу «Портрет»)

Рис. 3 – Перший варіант обробки
(маска 5×5; P_Z =3)

Рис. 4 – Другий варіант обробки
(маска 5×5; P_Z =3)

Рис. 5 – Третій варіант обробки
(маска 5×5; P_Z =3)

Третій варіант обробки, забезпечує перебір *«всіх з усіма»*, де головним завданням є перевірка на наявність елементів, різниця між якими перевищує значення P_Z. При наявності такої різниці даний блок залишається без змін (*так як можлива присутність контуру*), в іншому випадку здійснюється заміна всіх значень на середнє значення яскравості всіх елементів блоку. Таким чином, отримуємо згладжене зображення зі збереженими контурами (*див. Рис.5*). По результатам моделювання 3-го варіанту згладжування, встановлено, що основні контури відновлюваних зображень типів портрет та пейзаж, зберігаються до $P_Z = 14$.

Для порівняння дослідних варіантів способів згладжування, використано значення середньоквадратичної помилки (СКП), яке надає можливість оцінити ступінь спотворень вихідних зображень після їх обробки (*чим менше СКП тим менш помітні аномалії і тим краще отриманий результат*). На рис. 6-7 представлено вихідне тестове зображення типу «Пейзаж», та результат його обробки з використанням 1-го варіанту згладжування *(найгіршого, як показали наступні результати моделювання*). Як видно з рис. 7, навіть при використанні великої матриці згладжування та явно завищеного значення P_Z, візуальна помітність спотворень знаходиться на дуже прийнятному рівні (*за дрібними винятками*). Більш того, зображення не зазнало а ніяких критичних змін структури фону і контурів основних деталей [4].

Рис. 6 – Вихідне зображення
(типу «Пейзаж»)

Рис. 7 – Перший варіант обробки
(маска 5×5; P_Z =32)

На рис. 8 представлена спрощена структурно – логічна схема дослідного алгоритму.

Всі процедури розробленого алгоритму поділені на **окремі етапи**, кожен з яких, має **своє** функціональне значення.

Рис. 8 – Спрощена структурно-логічна схема алгоритму

Головне завдання 1-го етапу: Зменшення обчислювальної складності алгоритму шляхом зменшення кількості візуально малопомітних перепадів яскравості елементів вихідних зображень за рахунок згладжування їх ***малоінформативних*** областей.

Головне завдання 2-го етапу: формування ***серій подібних блоків*** в зображені для контенту і контейнеру, та отримання ***опорних блоків*** (*надалі - ОБ*) і відповідних їм параметрів довжин серій. Забезпечує ***основне*** зменшення обчислювальної складності всього алгоритму!

Головне завдання 3-го етапу: проведення ДКП для кожного із ОБ зображень контейнера і контенту та здійснення відбору і квантування значущих коефіцієнтів трансформант. Ініціюючі параметри цього етапу (*формат селекції і квантування коефіцієнтів перетворення*), формують окремі позиції в структурі загального ключа шифрування для легітимного вилучення контенту при його декодуванні.

Головне завдання 4-го етапу: Приховування (*шифрування*) отриманого масиву ОБ, шляхом багатомірного мультиплексування низки параметрів, котрі формують основні позиції в структурі загального ключа екстрактора. В цілому, до таких параметрів відносяться використані характеристики мультиплексу щодо: - числа ОБ; - параметрів довжин серій ОБ; - значень середньої яскравості ОБ; та признаку поточної симетрії обробки контейнера і контенту.

Варто зазначити, що в діючої тестової версії прототипу гібридного стеганоалгоритму, процедури 4-го етапу слід розглядати, не більш як «*Демонстратор можливостей*», який підтверджує працездатність запропонованих механізмів стеганографічного захисту контенту. Тобто в діючий тестової версії алгоритму, він втілює значно спрощені механізми обробки даних (*версія «лайт»*), однак при дотриманні всіх основних параметрів та етапності проведення згаданих вище процедур.

Параметри налаштувань та класифікація використаних режимів роботи прототипу дослідного гібридного стеганоалгоритму наведені в Табл.1-2.

Таблиця 1 – Основні параметри налаштувань дослідного алгоритму

Параметр	Зміст параметру/процедури
P_z	Значення порога загрублення яскравості сусідніх елементів блоків зображень *(для контейнеру і контенту)*
N	Розмірність блоків *(для контейнеру і контенту)*
P	Порядок селекції та зберігання коефіцієнтів перетворення в ОБ
q	Порядок взаємного перемішування ОБ *(демонстратор шифрування)*

Таблиця 2 – Основні режими роботи алгоритму

Режим роботи	**Поріг загрублення**	**Розмірність блоку**
Швидкий режим	$P_z = 9$	$N = 16$
Захищений режим	$P_z = 4$	$N = 4$
Збалансований режим	$P_z = 6$	$N = 8$

Результати роботи 1-го варіанту попередньої обробки вхідних даних (*див. рис.1*), по закінченню процедур 1-го етапу дослідного алгоритму представлені нижче, на рис. 9.

a) Вихідне тестове зображення типу «портрет»;

b) Відновлене зображення для маски 5×5 ел.; Pz = 64 (!);

c) Відновлене зображення для маски 5×5 ел.; ***Pz = 127(!!).***

Рис. 9 – Результати роботи 1-го варіанту згладжування на 1-му етапі алгоритму (*ймовірність перепаду яскравості між сусідніми елементами p = 0.034*)

Як видно з рис. 9, збільшення значень *порогу загрублення* сусідніх елементів зображення *(далі – викривлення)* збільшують ступінь викривлень відновлюваних зображень, причому, при збільшенні розміру *маски згладжування* цей процес стає більш помітним.

Значення СКП для різних типів зображень та варіантів попередньої обробки вхідних даних представлені, на рис. 10.

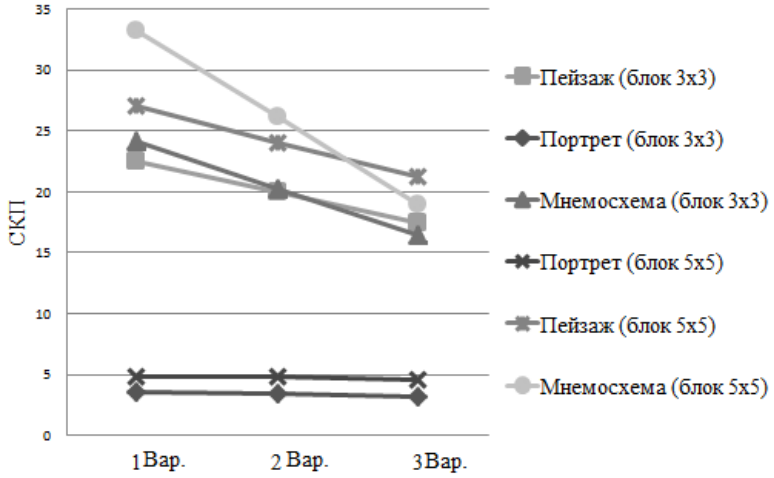


Рис. 10 – СКП для різних типів зображень та масок згладжування
(для випадку P_z = 3, тобто в межах зорової непомітності)

Значення СКП для різних типів зображень та варіантів предобробки вхідних даних, при різних значеннях *порогу загрублення* сусідніх елементів, представлені на рис. 11.

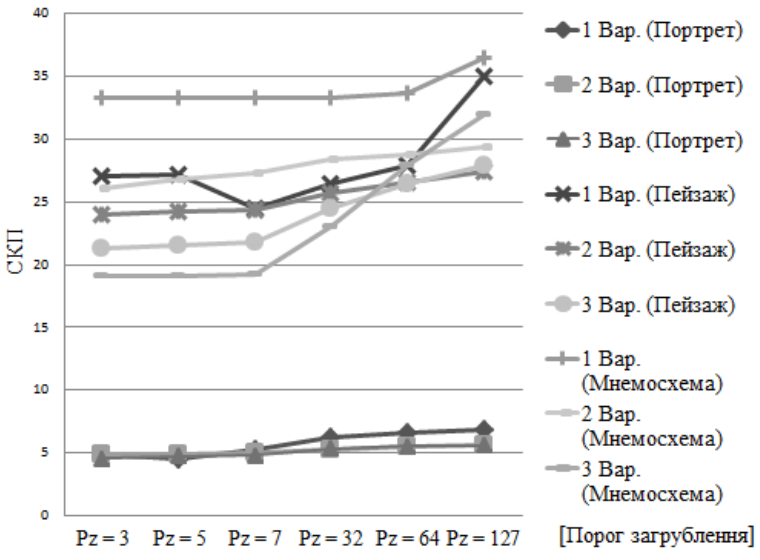


Рис. 11 – Залежність СКП від P_Z для різних типів зображень *(для блоків 5×5 ел.)*

Загальний аналіз Рис.10-11 дозволяє констатувати, наступне:

- при збільшені порогу викривлення і розміру маски згладжування, кількість та помітність артефактів зростає для *всіх* типів зображень;

- для зображень типу мнемосхема, стрімке зростання викривлень відбувається при виборі параметра P_z більш 7. Для зображень типу пейзаж і портрет, граничним значенням можна вважати P_z = 32 (*при використанні 256 рівнів яскравості !*);

- для зображень типу мнемосхема, вкрай бажано використовувати 3-й варіант обробки *(«всі з усіма»)*, який забезпечує гарну сенсорику обрисів контурів (*це дещо більш ресурсноємний режим див. рис. 12*), для інших типів зображень *основні контури* зображень утримуються до P_z = *14*;

- незважаючи на відносно високі значення СКО для зображень типу пейзаж, візуальна помітність спотворень залишається на низькому рівні завдяки впливу ефекту просторового маскування (див. рис.7);

Типові значення часу виконання процедур згладжування 1-го етапу, для 2-х розмірностей *масок згладжування* при різних *варіантах попередньої обробки* вхідних даних, та різних значеннях *порогу загрублення* сусідніх елементів зображень, представлені, на рис. 12. За результатами узагальнення залежностей на Рис.12 можна стверджувати наступне:

- час виконання всіх варіантів предобробки зображень залежить від розмірності блоків та значення порога загрублення *(P_z)*;

- при збільшенні розмірності маски згладжування загальний час обробки буде значно зменшено (*на рис. 12, вся група залежностей для блоків 5 × 5 розташована нижче відповідної групи залежностей для блоків 3×3 ел.);*

- застосування 3-го варіанту згладжування *(«всі з усіма»*), потребує більше часу для всіх розмірностей матриць згладжування, причому різниця досить помітна;

- в межах кожної фіксованої розмірності матриць згладжування, при збільшенні порогу загрублення *(P_z)* в межах значень малої візуальної помітності спотворень (*до 7 градацій яскравості, зелений пунктир на рис.12-13*), практично для всіх варіантів предобробки спостерігається невелике стійке зростання часу виконання цих операцій. Можна припустити, що зі збільшенням розмірності матриць згладжування, зростає і кількість логічних процедур, які реалізуються в рамках відповідних інструкцій предобробки (*див. рис. 1*);

- за критерієм *min* часу обробки гідною альтернативою для 1-го і 3-го варіантів предобробки, є 2-й варіант, який має перевагу при любих розмірностях масок згладжування.

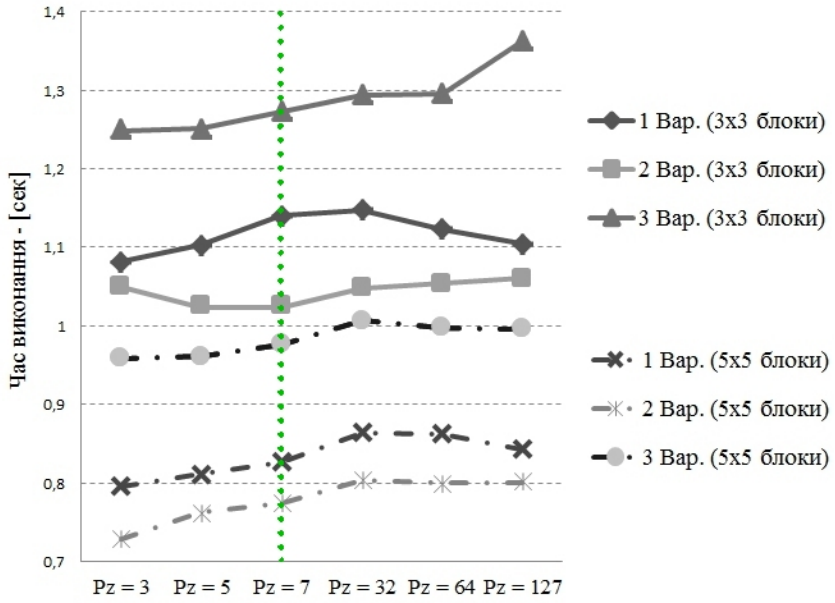


Рис. 12 – Час виконання різних варіантів предобробки зображень *(для блоків 3×3 та 5×5 ел.)*

Характерні залежності кількості ОБ від типу зображень та параметрів їх обробки (*різні значення P_z та розміри блоків*) представлені на рис. 13.

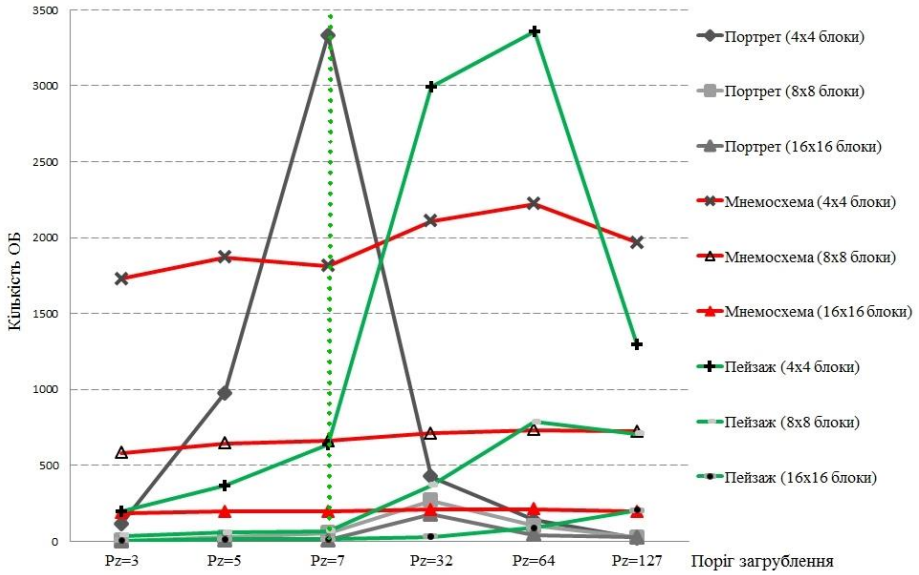


Рис. 13 – Кількість ОБ для різних налаштувань та типів зображень

Узагальнення отриманих залежностей дозволяє зробити певні висновки:

- кількість сформованих ОБ напряму залежить від типу використовуваного зображення та від заданого значення порогу загрублення *(P_z)*;

- найбільша кількість ОБ формується при мінімальній розмірності блоків (*4×4*) та використанні зображень типу «*Портрет*» і «*Пейзаж*», однак, їх «цінність» з точки зору наступної інкапсуляції контенту дуже сумнівна (*принаймні на даному етапі моделювання*);

- використання блоків малої розмірності (*4×4*) для обробки зображень типу «Мнемосхема» має ті ж самі наслідки, що і в попередньому випадку, однак при, практично вдвічі меншої їх чисельності;

- зображення типу мнемосхема, в порівнянні з іншими типами зображень, виявили значно меншу залежність від зміни параметру P_z. Ця обставина, очевидно пояснюється особливостями структури зображень цього типу, які мають виключно чіткі силуети контурів об'єктів, що відображаються, та великі області однорідної яскравості або колірної заливки (*ймовірність перепаду яскравості між сусідніми елементами зображень цього типу, знаходиться в межах* $0{,}01 \le p \le 0{,}03$);

- параметри довжин серій ОБ, так само, як і фактична кількість ОБ контейнеру, є одними із головних елементів в структурі загального ключа екстрактора стеганоалгоритму.

Приклад результатів роботи прототипу гібридного стеганоалгоритму в його *«лайт»* версії *(як «Демонстратор можливостей»)*, яка підтверджує реалізованість запропонованих механізмів попередньої обробки та інкапсуляції контенту, наведено нижче на рис. 14.

a) Вихідне тестове зображення – контент, типу «портрет»;

b) Зсув ОБ тільки на одну позицію (тобто серію);

Рис. 14 – Приклад невдалого підбору тільки одного параметру ключа вилучення даних (*розташування ОБ контенту*)

Рис. 14 (b), ілюструє спробу несанкціонованого відновлення ***тільки*** одного параметру кодування: - розташування ***опорних блоків*** контенту в умовах деактивації 4-ох інших параметрів обробки *«лайт»* версії алгоритму *(див Табл. 1)*.

7 Висновки

1. Розглянутий гібридний алгоритм забезпечує можливість дослідження взаємозв'язків окремих параметрів обробки на кожному з його етапів та властивостей відновлюваних зображень, в залежності від характеристик міжблокової статистики вихідних даних. Основні властивості алгоритму, дозволяють класифікувати його, як дослідний засіб моделювання процедур стеганографічних перетворень зображень, при реалізації режиму внутрішньокадрової обробки даних.

2. В поточній версії прототипу алгоритму, процедури 4-го етапу слід розглядати, як «*Демонстратор можливостей*», що підтверджує коректність запропонованих механізмів предобробки зображень і стеганографічного захисту контенту. Діючий реліз алгоритму моделює спрощені механізми обробки зображень (*версія «лайт»*) з дотриманням всіх основних параметрів та етапності проведення згаданих процедур.

3. Результати, що представлені в даній роботі є логічним продовженням низки досліджень [4, 8-10], що присвячені питанням стиску, фільтрації та стеганографічного захисту відеоданих різного типу. За умови внесення деяких доопрацювань (*насамперед у модулі введення даних налаштувань та модулі візуалізації проміжних результатів*), та концептуальній зміні параметрів моделі інтерфейсу користувача, діючий прототип гібридного стеганоалгоритму може бути використаний в освітніх цілях для цілей створення дослідних програмних імітаторів стеганографічного захисту відеоданих.

4. Збільшення кількості областей зображень з заздалегідь підготовленою, відповідним чином текстурою, створює необхідні стартові умови для ефективної реалізації етапу формування серій подібних блоків. Кількість таких серій залежить, як від структури зображень, так і встановлених параметрів їх розмірності та згладжування.

5. Використання для зображень контенту і контейнеру різних варіантів попередньої обробки та/або різних параметрів їх настроювань (різна розмірність матриць згладжування та різні значення P_Z), вносить потрібну асиметрію в структуру даних, які обробляються, з усіма очевидними корисними наслідками.

6. Виходячи з відомостей про поточні характеристики зображення-контенту, можливо адаптивно і оперативно змінювати властивості наявних контейнерів (*наприклад, що є в пам'яті мобільного пристрою*).

Посилання

[1] Грибунин В.Г. Цифровая стеганография / Грибунин В. Г., Оков И. Н., Туринцев И. В. – М.: Солон-Пресс, 2002. – 272 с.

[2] Зубарев Ю.Б., Дворкович В.П. Цифровая обработка телевизионных и компьютерных изображений. – М.: МЦНТИ, 1997. – 212 с.

[3] Ярославский Л.П. Введение в цифровую обработку изображений. – М.: Сов.Радио, 1979. – 312 с.

[4] Гончаров М. О., Малахов С. В. Моделювання процедур підготовки даних стеганоалгорітма з багаторівневим мультиплексуванням контенту. Комп'ютерне моделювання в наукоємних технологіях (КНМТ-2021): матеріали 7-ї міжнар. наук.-техн. конф., 21-23 квіт. 2021 р. Харків: ХНУ ім. В.Н. Каразіна. http://www-csd.univer.kharkov.ua/wp-content/uploads/2018/02/www-csd.univer.kharkov.ua-maket-pdf-konf.pdf

[5] Обработка изображений на ЭВМ / Е.А Бутаков, В.И. Островский, И.Л. Фадеев. – М.: Радио и связь, 1987. – 240 с.

[6] Прэтт У. Цифровая обработка изображений. т. 1,2. - М.: Мир, 1985. - 736 с.

[7] Быков С.Ф. Алгоритм сжатия JPEG с позиции компьютерной стеганографии / Защита информации. Конфидент. – СПб.: 2000, № 3. – С. 26.

[8] Morozov, D., Shaforostov, M., Malakhov, S., & Serbin, V. (2018). Подвійна обфускація трансформант малоресурсного стеганоалгоритма. *Комп'ютерні науки та кібербезпека*, *9*(1), 22-34. вилучено із https://periodicals.karazin.ua/cscs/article/view/12015

[9] Гончаров М. О. Дослідження процедур попередньої підготовки вихідних даних для стеганоалгоритма. *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління* : матеріали 11-ї міжнар. наук.-техн. конф., 8–9

квіт. 2021 р. м. Баку–Харків–Київ–Жиліна. т. 2. с. 63. URL: http://repository.kpi. kharkov.ua/handle/KhPI-Press/52020 (дата звернення 01.11.2021).

[10] Гончаров М.О. Моделювання та дослідження властивостей гібридного внутрікадрового стеганоалгоритму: Пояснювальна записка до дипломної роботи бакалавра: напрям підготовки 125 – Кібербезпека / М. О. Гончаров; Харківський національний університет імені В. Н. Каразіна. – Харків: [Б. В.], 2021. – 75 с.

Рецензент: Вячеслав Калашников, д.ф.-м.н., проф., Технологический университет Монтеррея, Монтеррей, Мексика.
E-mail: kalash@itesm.mx

Поступила: Ноябрь 2021.

Авторы:

Никита Гончаров, студент кафедры Безопасности информационных систем и технологий, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина.
E-mail: worldxdark@gmail.com

Юлия Лесная, студентка кафедры Безопасности информационных систем и технологий, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина.
E-mail: xa12284109@student.karazin.ua

Сергей Малахов, к.т.н., с.н.с., доцент факультета компьютерных наук, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина.
E-mail: mailgate@meta.ua

Исследование свойств прототипа гибридного стеганоалгоритма.

Аннотация. Целью данного материала является ознакомление с основными этапами адаптивного малоресурсного алгоритма стеганографической обработки изображений и результатами моделирования процедур предварительной обработки исходных данных разных типов. Процедуры имитационного метода позволяют: - учитывать особенности обрабатываемых данных (типы контейнера и контента) и корректировать параметры работы основных модулей стеганоалгоритма (модуль предварительной обработки входных данных и модуль специальных преобразований). Исследованы другие параметры обработки изображений, непосредственно влияющие на вычислительную сложность 1-го этапа алгоритма (*сглаживание*) и качество визуализации контейнера, и его содержимого. Отмечено, что для всех типов изображений, вариант предварительного сглаживания входных блоков, по принципу «перебор всех со всеми», дает лучшие результаты. В данном случае, с уменьшением размерности сглаживающих матриц, интенсивность визуально заметных аномалий уменьшается.

Подчеркнуто, что при увеличении значения порога загрубления яркости элементов (P_Z), количество и заметность артефактов возрастает. Рост фиксируется для всех типов изображений и всех вариантов предварительного сглаживания. Устойчивый рост искажений происходит при выборе значений P_Z более 7. Для реалистических изображений, критически допустимой следует считать величину $P_Z = 14$. Обращено внимание на то, что в независимости от установленных значений P_Z и выбранного варианта сглаживания, визуальная заметность искажений, усиливается в следующей последовательности: «*портрет – пейзаж – мнемосхема*» Наиболее «чувствительными» к вариантам предварительного сглаживания, оказались изображения типа «мнемосхема». Это можно объяснить чувствительностью контуров к их малейшим изменениям и особенностям структуры таких изображений.

Обращено внимание на то, что факт малой вычислительной сложности процедур предварительной обработки изображений, является принципиальным с учетом концепции создания мобильных приложений. Сделано вывод, что потенциальный выигрыш от ввода этапа предварительной обработки исходных данных, позволяет получить 3 важных эффекта: 1 – снизить вычислительную сложность алгоритма; 2 - использовать разные принципы обработки данных контейнера и контента; 3 – создать необходимые условия для асимметричного режима обработки данных контейнера и контента.

Ключевые слова: изображение; стеганография; кодирование с преобразованием; контейнер; контент; сглаживание изображений; визуальная заметность искажений; кодирование серий; мультиплексирование.

Reviewer: Vyacheslav Kalashnikov, Doctor of Sciences (Physics and Mathematics), Full Prof., Department of Systems and Industrial Engineering, Campus Monterrey, Monterrey, Mexico.
E-mail: kalash@itesm.mx

Received: November 2021.

Authors:

Mykyta Honcharov, student of the Department of Security of Information Systems and Technologies, Kharkiv National University named after V.N. Karazin, Kharkiv, Ukraine.
E-mail: worldxdark@gmail.com

Yuliia Liesnaia, student of the Department of Security of Information Systems and Technologies, Kharkiv National University named after V.N. Karazin, Kharkiv, Ukraine.
E-mail: xa12284109@student.karazin.ua

Serhii Malakhov, Ph.D., Senior Researcher, Computer Science Department, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: mailgate@meta.ua

Investigation of the properties of the prototype of a hybrid steganographic algorithm.

Abstract. The purpose of this material is to get acquainted with the main stages of the adaptive low-resource algorithm of steganographic image processing and the results of modeling the procedures of pre-processing of source data of different types. The simulative algorithm procedures allow: - take into account the features of processed data (*types of container and content*) and to adjust parameters of operation of main modules of a steganoalgorithm (*the module of preprocessing of input data, and the module of specia conversions*). Other image processing parameters that are directly investigated affect the computational complexity of the 1st stage of the algorithm (*smoothing*) and the quality container and content visualization. It is noted that for all types of images, the option of pre-smoothing the input blocks, on the principle of "busting everyone with everyone", provides better results. In this case, with reducing the dimensionality of anti-aliasing matrices, the intensity of visually noticeable anomalies decreases.

Underlined that when the threshold for adjusting the brightness of elements (P_Z) increases, the number and visibility of artifacts is increasing. Growth is fixed for all types images and all options for their pre-smoothing. Steady growth of distortions occurs when choosing values of P_Z, more than 7. For realistic images, the value $P_Z = 14$ should be considered critically valid.

Attention is drawn to the fact that regardless of the established values of P_Z and the selected smoothing option, visual visibility of distortions, intensifies in the following sequence: «portrait – landscape – mnemonic». The most "sensitive" to the options for pre-smoothing, were mnemonic chart image. This can be explained by the sensitivity of the contours to their changes and structure of such images.

Attention is drawn to the fact that the low computational complexity of the previous procedures image processing is fundamental given the concept of creating mobile applications. It is concluded that the potential winnings from the introduction of the preprocessing stage output data allows you to get 3 important effects: 1 - reduce computational complexity algorithm; 2 - use different principles of container and content data processing; 3 - create the necessary conditions for the asymmetric mode of data processing of the container and content.

Keywords: Image; Steganography; Transform encoding; Container; Content; Anti-aliasing of images; Visual noticeability of distortion; Series coding; Multiplexing.

АНДРЄЄВ Фелікс Михайлович

(26.12.1936 – 01.01.2022)

Редколегія журналу та співробітники факультету комп'ютерних наук Харківського національного університету імені В.Н. Каразіна з глибокою скорботою сповіщають про те, що передчасно пішов з життя професор кафедри електроніки та управляючих систем, доктор технічних наук Андрєєв Фелікс Михайлович.

Фелікс Михайлович народився у м. Горький (*Нижній Новгород*) у сім'ї військового і вчительки. В школу пішов у 1944 році, за спогадами самого Фелікса Михайловича, «у будівлі школи було розгорнуто шпиталь, тож тимчасово вчилися у холодному, малопристосованому для навчання двоповерховому будинку, а зошити з чистописання робили з обгорткового паперу». Сім'я постійно переїжджала за новими місцями служби батька, тож хлопчику прийшлося змінити декілька шкіл, але він завжди був відмінником навчання і спортсменом. Фелікс наполегливо займався легкою атлетикою, стрільбою (*мав 2-й розряд зі стрільби з гвинтівки на 300 м*), та баскетболом (*у складі шкільної команди стали віце-чемпіонами області серед юнаків*). Поряд з цим він достроково закінчив музичну школу, добре грав на акордеоні та писав вірші.

В 1954 році Фелікс Михайлович закінчив із срібною медаллю середню школу в м. Пенза. Фелікс, а пізніше і його молодший брат В'ячеслав, вже за сімейною традицією, обрали шлях військового. Успішно склавши вступні іспити (*47 балів з 50*), Фелікс був прийнятий в Артилерійську радіотехнічну академію Радянської Армії (*м. Харків*). Під час навчання спритний юнак був членом збірних команд академії з баскетболу і кульової стрільби, чемпіоном академії зі стрільби (*нормативи 1 розряду*), та віце-чемпіоном із багатоборства.

В 1959 році Ф.М. Андрєєв закінчив АРТА РА за спеціальністю «інженер з радіолокації», отримавши диплом з відзнакою. Військову службу молодий офіцер проходив з 1959 по 1962 роки інженером на Державному науково-дослідному випробувальному полігоні №10 (*м. Приозерськ, республіка Казахстан*). Там він займався випробовуваннями перших полігонних зразків вітчизняних станцій надгоризонтного виявлення (НГВ) балістичних та космічних об'єктів типу «Днестр» і «Днестр-М». До речі, він не покинув спорт і в той період входив у збірні команди в/ч з баскетболу і кульової стрільби, приймав участь у змаганнях на першість Середньоазіатського військового округу.

В 1962-1965 рр. Фелікс Михайлович навчався в ад'юнктурі АРТА РА. Після навчання й успішного захисту кандидатської дисертації (у 1966 р), працював викладачем і старшим викладачем академії з перервою на навчання в докторантурі. У 1971 році Фелікс Михайлович очолив науково-дослідницьку групу при кафедрі засобів раннього виявлення балістичних ракет та космічних об'єктів для проведення наукових досліджень, які були пов'язані із розробкою методів отримання й обробки інформації в РЛС НГВ при спостереженні складних балістичних цілей (СБЦ). Пізніше Фелікс Михайлович став засновником наукової школи з організації та проведення напівнатурних випробувань РЛС НГВ СБЦ та багатоелементних космічних об'єктів.

Дисертацію на здобуття наукового ступеню доктора технічних наук Ф.М. Андрєєв захистив у 1989 році. В роботі розглядалась проблематика радіолокаційного виявлення складних і багатоелементних об'єктів, питання побудови трактів обробки сигналів, що притаманні подібним об'єктам спостереження, та особливості вторинної обробки відповідної інформації.

У 1991р. Феліксу Михайловичу було присвоєно вчене звання «професор».

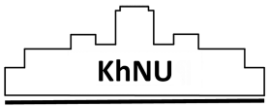
З 1993 по 2004 роки він був професором Харківського Військового Університету (ХВУ), з 2004 по 2005 – професор Харківського університету Повітряних сил ім. Івана Кожедуба. В 1996 році Фелікса Михайловича було звільнено з лав Збройних сил. В цей період роботи основним напрямом його наукових інтересів були питання отримання й використання інформації про вищі похідні дальності в інформаційно-вимірювальних системах. За період служби в академії Фелікса Михайловича було нагороджено орденами «За службу Батьківщині в рядах ЗС СРСР» III ступеню (1978) та Пошани (1990). З 19 жовтня 2005 р. Фелікс Михайлович отримує посаду професора кафедри теоретичної та прикладної системотехніки, а з 2008 р. – професора кафедри електроніки та управляючих систем факультету комп'ютерних наук Харківського національного університету імені В.Н. Каразіна.

Андрєєв Фелікс Михайлович - заслужений винахідник УРСР, мав більш ніж 60 авторських свідоцтв на винаходи СРСР, 3 патенти України, 12 патентів України на корисну модель. Лауреат премії МО СРСР та Держкомітету народної освіти СРСР III ступеню (1988), академік Міжнародної АН Прикладної радіоелектроніки (1993). Він був автором більш ніж 250 наукових та науково-методичних праць, серед яких 4 підручника. Фелікс Михайлович підготував 2 докторів та 19 кандидатів технічних наук. Мав 10 медалей, Почесну грамоту Харківської обласної адміністрації (2001) та Грамоту Департаменту з гуманітарних питань Харківської міської ради.

Варто підкреслити, що окрім суто професійних науково-теоретичних і прикладних питань, Фелікс Михайлович захоплювався і суспільно-історичними розслідуваннями, за результатами чого з'явилися такі цікаві видання як: «Тайное становится явным»: Страницы истории подготовки офицерских кадров для РЛС дальнего обнаружения ПРН и ПРО (2014); «Каразінці – родоначальники імпульсної радіолокаційної техніки»: науково-популярний нарис до 210-річчя Харківського національного університету імені В.Н. Каразіна (2015); «Выдающиеся личности Военной инженерной радиотехнической академии имени Л.А. Говорова»: Страницы истории (2016); «Героїчні особистості Харківського університету імені В.Н. Каразіна (до 212-річчя Харківського національного університету імені В.Н. Каразіна)»: біографічний довідник (2017); «Харьковская научная школа средств защиты импульсных РЛС от маскирующих пассивных помех: монография» (2019). Монографія Фелікса Михайловича «Военной инженерной академии имени Л. А. Говорова ─ 80 лет» побачила світ наприкінці 2021 року.

Таким він був. Цілеспрямований, завзятий, активний. Мав грандіозні плани щодо створення історії кафедри, що стала його останньою домівкою. Не дозволяв собі зневіри й песимізму навіть тоді, коли доля приносила неприємні «сюрпризи». І з лікарняного ліжка він міг підтримати «бойовий дух» колег та однодумців. І при всьому цьому – дуже відкрита, добра, щира людина з м'якою посмішкою без всякого пафосу, та завжди з пречудовим почуттям гумору. Спілкуючись із ним, часто здавалося, що то була Душа факультету. Тепер – нема, порожнеча… Спочивайте з миром, Феліксе Михайловичу. А ми про Вас – **пам'ятаймо.**

ISSN 2519-2310 DOI: 10.26565/2519-2310-2021-2



Наукове видання

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА

Випуск 2(20) 2021

Міжнародний електронний науково-теоретичний журнал

Англійською і українською мовами

Комп'ютерне верстання – Федоренко В.В., Єсіна М.В.

61022, Харків, майдан Свободи, 6
Харківський національний університет імені В.Н. Каразіна

V. N. Karazin Kharkiv National University Publishing

CS&CS

2021