Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *інфокомунікацій*
(повна назва)
Кафедра *інформаційно-мережної інженерії*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Технічний аудит інтернет-магазину, як засіб збільшення його продуктивності
(тема)

Виконав:
студент *2* курсу, групи *ІМІм-20-2*
Дяченко Д.А.
(прізвище, ініціали)

Спеціальність *172. Телекомунікації та радіотехніка*
(код і повна назва спеціальності)
Тип програми освітньо-наукова
Освітня програма *Інформаційно-мережна інженерія*
(повна назва освітньої програми)

Керівник *ст.викл. Калюжний М.М.*
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *Безрук В.М.*
(підпис) (прізвище, ініціали)

2022 р.

Не містить відомостей, заборонених до відкритого публікування

Керівник _____ / *М.М. Калюжний*

Студент _____ / *Д.А. Дяченко*

Харківський національний університет радіоелектроніки

Факультет *інфокомунікацій*

Кафедра *інформаційно-мережної інженерії*

Рівень вищої освіти другий (магістерський)

Спеціальність *172. Телекомунікації та радіотехніка*
(код і повна назва)

Тип програми освітньо-наукова

Освітня програма *Інформаційно-мережна інженерія*
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Дяченко Денису Андрійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Технічний аудит інтернет-магазину, як засіб збільшення його продуктивності*

затверджена наказом університету від *14 березня* 2022 р. № *379-Ст*

2. Термін подання студентом роботи до екзаменаційної комісії *31 травня* 2022 р.

3. Вихідні дані до роботи *Виконати огляд найбільш поширених типів кібератак. Дослідити сутність та особливості здійснення АРТ. Обгрунтувати, що застосування механізмів маскування даних набуло застосування як один зі складників АРТ. Дослідити загальні підходи до побудови системи стегоаналізу трафіку у реальному часі. Довести доцільність та дослідити принципи функціонування алгоритмів*

4. Перелік питань, що потрібно опрацювати в роботі *Вступ.*

1. *Чинники, які визначають необхідність застосування заходів з технічного аудиту*
2. *Реалізація заходів з технічного аудиту веб-ресурса*
3. *Забезпечення функціональності веб-ресурсів за рахунок виявлення зловмисного змісту у складі контенту*
4. *Методи стегоаналізу для виявлення та протидії прихованих зловмисних впливів на веб-вузли*

Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____
Слайди у форматі Power Point (назва та мета роботи, технічний аудит веб-ресурсів, схема аудиту контенту у реальному часі, поширені методи стегоаналізу, модифікація алгоритму візуального аналізу, висновки)

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Вступ		Виконано
2	Розділ 1		Виконано
3	Розділ 2		Виконано
4	Розділ 3		Виконано
5	Розділ 4		Виконано
6	Висновки		Виконано

Дата видачі завдання _____ _____20__ р.

Студент _____
(підпис)

Керівник роботи _____ *ст.викл. Калюжний М.М.*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 102 с., 33 рис., 34 джерела, 2 додатки

ТЕХНІЧНИЙ АУДИТ, ВЕБ-СЕРВЕР, СТЕГОАНАЛІЗ, APT, LSB, МЕТОД XI КВАДРАТ, PHP, БАЗА ДАНИХ, СИГНАТУРНІ МЕТОДИ

Об'єкт дослідження – алгоритми технічного аудиту веб-вузлів та виявлення прихованих даних.

Мета роботи – дослідити особливості використання типових заходів моніторингу стану сайту та механізмів стегоаналізу, як додаткових механізмів підтримки його функціональності.

Розглядаються базові стадії виконання технічного аудиту веб-ресурсу. Рокривається сутність застосування методів стеганографії, як додаткових засобів підвищення продуктивності сайту. Пропонується схема обробки потоку надходячих даних у реальному часі. Удосконалюється підхід до виявлення стегоконтейнерів на основі візуальних методів.

THE ABSTRACT

Explanatory note: 102 p., 33 fig., 34 sources, 2 app.

TECHNICAL AUDIT, WEB SERVER, STEGANALYSIS, ART, LSB, XI SQUARE METHOD, PHP, DATABASE, SIGNATURE METHODS

The object of research - algorithms for technical audit of websites and detection of hidden data.

The purpose of the work is to investigate the peculiarities of the use of standard measures for monitoring the state of the site and the mechanisms of stegoanalysis, as additional mechanisms to support its functionality.

Basic stages of technical audit of a web resource is considered. The essence of the use of steganography methods as additional means of improving the productivity of the site is revealed. The scheme of processing of a stream of incoming data in real time is offered. The approach to the detection of stegocontainers based on visual methods is being improved.

ЗМІСТ

С.
ПЕРЕЛІК СКОРОЧЕНЬ…….. 9
ВСТУП……. 10
1 ЧИННИКИ, ЯКІ ВИЗНАЧАЮТЬ НЕОБХІДНІСТЬ ЗАСТОСУВАННЯ ЗАХОДІВ З ТЕХНІЧНОГО АУДИТУ …… 12
1.1 Загальні вимоги з продуктивності веб-ресурсу ……. 12
1.2 Ключові шляхи збільшення технічної продуктивності сайту …… 14
1.3 Обгрунтування необхідності періодичного проведення заходів х технічного аудиту веб-ресурсу ……..…. 16
1.4 Типові сценарії технічного аудиту сайту ……. 21
2. РЕАЛІЗАЦІЯ ЗАХОДІВ З ТЕХНІЧНОГО АУДИТУ ВЕБ-РЕСУРСА 24
2.1 Технічний аудит вузла на стадії впровадження проекту …….. 24
2.1.1 Тестування серверної платформи …… 27
2.1.2 Тестування процесорної системи ……. 28
2.1.3 Тестування дискових пристроїв ……..….. 29
2.1.4 Виконання тесту бази даних MySQL 31
2.1.5 Тестування PHP-оточення …….. 33
2.2 Виконання технічного аудиту активних проектів ……..….. 37
2.3 Обгрунтування необхідності технічного аудиту контенту веб-вузла 45
2.3.1 Сутність механізмів прихованої доставки зловмисного коду на базі стеганографічних підходів …… 46
3. ЗАБЕЗПЕЧЕННЯ ФУНКЦІОНАЛЬНОСТІ ВЕБ-РЕСУРСІВ ЗА РАХУНОК ВИЯВЛЕННЯ ЗЛОВМИСНОГО ЗМІСТУ У СКЛАДІ КОНТЕНТУ ……...…….. 49
3.1 Загальний сценарій вконання заходів з аналізу контенту, надходячого до веб-ресурсу…….. 49
3.1.1 Аналіз можливості застосування файлів різних типів для реалізації стеганографічних контейнерів …….. 54
3.1.2 Ключові підходи щодо реалізації заходів стегоаналізу на основі групування прийнятих файлів за показником ефективності їх використання у ролі контейнеру прихованих даних …….. 60
3.2 Диференційна схема реалізації процесу стегоаналізу інформаційних пакетів, що надходять до веб-вузла …… 65
4 МЕТОДИ СТЕГОАНАЛІЗУ ДЛЯ ВИЯВЛЕННЯ ТА ПРОТИДІЇ 71

ПРИХОВАНИХ ЗЛОВМИСНИХ ВПЛИВІВ НА ВЕБ-ВУЗЛИ ……
4.1 Ключові принципи створення алгоритмів виявлення прихованого контенту ……..…. 71
4.2 Статистичні методи аналізу контенту …… 78
4.2.1 Алгоритм на базі збору статистики щодо двійкових переходів молодших біт …….…… 78
4.2.2 Алгоритм оцінки частот появи k-елементних ланцюжків у межах LSB контейнера 80
4.2.3 Алгоритм аналізу характеру розподілу елементів графічних об’єктів у двовимірному просторі 81
4.3 Концепція удосконалення алгоритмів візуального аналізу…….. 83
4.3.1 Модифікація механізму дослідження LSB…… 83
4.3.2 Механізм аналізу контурних зон ймовірних контейнерів 88
ВИСНОВКИ…… 96
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ…….. 98
ДОДАТОК А СЛАЙДИ ПРЕЗЕНТАЦІЇ…… 103
ДОДАТОК Б ТЕЗИ ДОПОВІДЕЙ……. 115

ПЕРЕЛІК СКОРОЧЕНЬ

APT – (Advanced Persistent Threat) – розвинута стійка загроза;
ПЗ – програмне забезпечення;
CMS – (Content Management System) система управління контентом сайту;
LSB – (Least Significant Bit) найменш значимий біт – молодший біт двійкового опису;
SEO – (Search Engine Optimization) – заходи з пошукової оптимізації сайту/сторінки;
API – (Application programming interface***)*** **–** програмний інтерфейс додатку;
GIS – (Geographic information system***)*** **—** геоінформаційні сервіси.

ВСТУП

У сьогоднішніх реаліях притання забезпечення функціональності та підсумкової продуктивності веб-вузла напряму пов’язані з досягненням максимально можливого рівня його технічної оптимізованості як на рівні апаратного забезпечення, так і на програмному рівні.

Одним з головних чинників, що викликали існування даної залежності, є той факт, що, з одного боку, на сьогодні більшість веб-вузлів функціонують у екстенсивному режимі. Тобто, збільшення продуктивності довільного веб-вузла (не беручи до уваги його нішеву належність, масштаб та завдання, які на нього покладаються) зараз частіше за все зводиться до розширення обсягу апаратних ресурсів, виділених для забезпечення його функціонування.

З іншого боку, даний шлях підвищення продуктивності веб-проекту не може вважатися ефективним, так як при цьому:

- зі збільшенням обсягу апаратних ресурсів, що виділяються для забезпечення функціонування веб-вузла, у геометричній прогресії зростають фінансові витрати з хостингу чи на розширення ресурсів апаратного серверу, що є найбільш критичним для стартапів та проектів невеликого масштабу;
- у ході нешевої конкуренції може виникнути ситуація, коли ряд ресурсів досягли гранично можливого рівня апаратних ресурсів як у технологічній, так і у фінансовій площині. Відтак, подальше зростання продуктивності в існуючому базисі технологій та рівні статку власника проекту буде неможливим;
- просування у нішах, де ТОП-10 формується виключно проектами масштабу, який на 1-3 порядки перевищують інші конкуруючі ресурси, за рахунок нарощування апаратної потужності серверної платформи неможливе з причини принципово різних фінансових можливостей «великих гравців» у ніші та дрібних конкурентів.

Так, очевидно, що досягти суттєвого зростання технічної продуктивності веб-вузла виключно шляхом збільшення його апаратних можливостей зараз неможливо.

У зв’язку з цим, необхідно звернути увагу на оптимізацію ключових програмних засобів, що застосовуються для реалізації веб-проекту, а саме:

- веб-серверу;
- бази даних;
- php-оточення;

- CMS тощо.

На сьогодні рішення з аналізу та налагодження стану функціонування кожної з перелічених програмних складових веб-проекту є окремо реалізованим. Утім, поки що відсутній інструментарій комплексного дослідження вузла.

Разом з тим, одним з факторів падіння технічної продуктивності веб-вузла можуть бути різнорідні зловмисні впливи, що супроводжуються прихованим втручанням у штатний режим функціонування вузлів. Значна частина подібних впливів реалізуються на базі класичних підходів і може бути виявлена та усунена з використанням стандартизованих засобів безпеки. Проте певний відсоток впливів здійснюється з використанням механізмів маскування даних, як самостійно, так і у складі APT (розвинутих стійких загроз), тому поширеними засобами кіберзахисту на можуть бути виявленими.

Такими чином, на сьогодні питання комплексного технічного аудиту веб-вузлу, включаючи також механізми виявлення прихованих зловмисних впливів, є гостро актуальними.

1 ЧИННИКИ, ЯКІ ВИЗНАЧАЮТЬ НЕОБХІДНІСТЬ ЗАСТОСУВАННЯ ЗАХОДІВ З ТЕХНІЧНОГО АУДИТУ

1.1 Загальні вимоги з продуктивності веб-ресурсу

Зараз у глобальному масштабі спостерігається ситуація, коли, у середньому близько 90% бюджету, виділеного на створення веб-проекту, витрачається на заходи з пошукової та технічної оптимізації, а також на розробку та реалізації маркетингової стратегії розвитку сайту [1].

Разом з тим, ще приблизно років 8-10 тому дана пропорція була оберненою до існуючою, а саме – близько 90 % бюджету планувалося витратити на створення ресурсу і приблизно 10% - на інші заходи.

Проте, як зазначалося раніше, ріст рівня конкуренції у мережі, впровадження більш жорстких механізмів ранжування з боку пошукових систем та надмірна пропозиція у галузі веб-розробок кардинально змінили існуючу парадигму [2]. Найбільш критичною є ситуація для інтернет-магазинів. У зазначених умовах більшість коштів витрачається на забезпечення ***функціональної*** (ФП) ***продуктивності сайту*** (термін актуальний як для інтернет-магазину, так і проектів іншого типу). Даний термін сьогодні можна інтерпретувати, як можливість того чи іншого веб-ресурсу ефективно вирішувати поставлені перед ним завдання [3].

Виходячи з того, що такі завдання можуть бути різними, іншими словами ФП можна розглядати, як спроможність веб-ресурсу за певний часовий відрізок забезпечувати:

- надходження певного обсягу фінансових коштів (або певної кількості операцій купівлі/продажу);
- певний обсяг трафіку (звернення до сайту тієї чи іншої кількості відвідувачів);
- конверсію на заданому рівні.

При цьому, у загальному випадку конверсія може сприйматися як відсоток відвідувачів, які виконали ті чи інші цільові дії на сайті [3-5]. Це може бути:

- купівля товарів та послуг;
- замовлення E-mail розсилки;
- розміщення відвідувачем посилання на сайм у соціальних медіа;
- проходження онлайн-анкетування тощо.

У свою чергу, ***технічна продуктивність*** (ТП) являє собою спроможність інтернет-магазину забезпечити відповідність тим чи іншим показникам комплексного оцінювання [3, 4].

Найчастіше такими показниками можуть бути час реакції вузла на клієнтське звернення, який вимірюється:

- у абсолютних одиницях (секунди);
- у відносних одиницях (внутрішні метрики пошукових систем).

Як у першому, так і другому випадку комплексні показники є відображенням великої кількості локальних показників, серед яких:

- апаратні характеристики веб-серверу, або хостингу;
- застосована система керування змістом (CMS) та її відповідність масштабу та завданням сайту;
- оптимізованість CMS та зовнішніх модулів;
- раціональність застосованих шаблонів CMS

При цьому, характер змін веб-середовища сьогодні має динамічний характер у наслідок впливу таких чинників, як [4]:

- впровадження нових технологічних рішень у веб-сфері та удосконалення рішень вже існуючих;
- суттєва залежість ряду ніш від сезонних та географічних локальних та глобальних процесів;
- залежність попиту на ті чи інші товари та послуги від зовнішніх чинників;
- непрогнозована поява у межах ніші одного або кількох «великих гравців».

Відтак, говорити про той чи інший фіксований рівень W_{func} функціональної продуктивності не є коректним. Натомість для даного показника встановлюється допустимий діапазон значень, тобто, загальною вимогою до функціональної, або загальної продуктивності буде наступна:

$$W_{func} \in [W_{func}^{(min)}; W_{func}^{(max)}], \quad (1.1)$$

де $W_{func}^{(min)}$ та $W_{func}^{(max)}$ - нижня допустима межа продуктивності та верхня межа відповідно.

У загальному випадку, фіксованого значення для величини $W_{func}^{(max)}$ не встановлюється [5, 6].

При цьому, взаємозв'язок між величинами функціональної W_{func} та технічної W_{tech} продуктивності може бути описано наступною функціональною залежністю:

$$W_{func} = \varphi(W_{tech}; \lambda), \quad (1.2)$$

де λ - показник продуктивності реалізованих заходів з SEO та інтернет-маркетингу.

Отже, вираз (1.2) показує, що одним з підходів щодо забезпечення росту функціональної продуктивності будь-якого веб-ресурсу є забезпечення збільшення технічної продуктивності.

1.2 Ключові шляхи збільшення технічної продуктивності сайту

Умови для зростання технічної продуктивності W_{tech} сайту а відтак – функціональної продуктивності W_{func} відповідно до виразу (1.1) можуть бути досягнуті за такими головними напрямками, як [5-7]:

1. Нарощування апаратної потужності платформи, на базі якої функціонує сайт, зокрема
   - зміна серверної системи на більш потужну;
   - корегування апаратної конфігурації серверної платформи (тобто, часткова модернізація обладнання);
   - вибір нової конфігурації хостингу, яка відрізняється вищими показниками обчислювальної потужності порівняно з існуючими.

Даний напрямок, безумовно, потенційно здатен тією чи іншою мірою забезпечити зростання показника W_{tech}.

Водночас, такому підходу властивий ряд недолікав та обмежень, серед яких головними є:

- обмеженість можливості нарощування обчислювальної потужності ω апартної платформи у заданих умовах у наслідок того, що залежно від множини η параметрів ніші, продуктивності застосованих SEO та меркетингових заходів λ та їх особливостей λ', а також архітектурних особливостей δ сайту, що унеможливлює подальший ріст W_{tech} навіть за умов досягнення максимально можливих значень ω, що може бути показано виразом:

$$W_{tech} = \lim \mid \omega = f(\lambda, \lambda', \delta) \to \max ; \qquad (1.3)$$

- у ряді умов збільшення продуктивності апаратної платформи може бути економічно невиправданим;
- заходи з модернізації апаратного забезпечення не є ефективними в умовах нераціональної реалізації взаємодії окремих вузлів, а також апаратно-програмної взаємодії.

2. Оптимізація програмної складової та процесів апаратно-програмної взаємодії. До цього напрямку у загальному випадку входять такі заходи, як [5]:

- налаштування бази даних та php-середовища;
- зміна чи налаштування CMS відповідно до завдань, які вирішуються сайтом;
- зміна чи раціоналізація шаблонів або тем сторінок;
- корегування способу подання контенту;
- усунення недоліків у функціонуванні обробників даних на рівні серверу тощо.

При цьому, за умови, що оптимізація програмної складової веб-ресурсу попередньо не була виконана, заходи з корегування апаратної частини не можуть гарантовано мати необхідний рівень ефективності.

Таким чином, у першу чергу підлягає корегуванню та оптимізації сукупність програмних рішень у складі сайту, за результатами чого далі виконується комплекс дій, орієнтованих вже на апаратну складову.

Разом з тим, на практиці більшість оперцій тут виконуютья спільно.

При чьому, необхідність внесення змін у програмну конфігурації тієї чи іншої структурної частини сайту, а також необхідність змін на апаратному рівні визначається за результатами комплексного технічного аудиту.

Необхідність спільного поглибленого апаратного та програмного аудиту пояснюється існуючими особливостями функціонування веб-ресурсів.

1.3 Обгрунтування необхідності періодичного проведення заходів х технічного аудиту веб-ресурсу

Однією зі складових продуктивності λ застосованих SEO та меркетингових заходів є обсяг χ реалізованого на сайті інструментарію, доступного для використання [3, 5-7].

При цьому, розширення обсягу доступних користувачам інструментів створює умови для позитивних змін у процесі функціонування веб-вузла, а саме:

- скорочення відсотку відмов у наслідок покращення поведінкових факторів відвідувачів;
- реалізації стратегії «повернення клієнтів»;
- формування постійної аудиторії відвідувачів.

За таких умов забезпечується:

- збільшення показника конверсії;
- покращення репутаційних показників сайту на рівні пошукових систем та аудиторії відвідувачів, що також у перспективі сприяє зростанню рівня показника W_{func}.

З іншого боку, розширення функціоналі сайту веде до помітного зростання обчислювального навантаження на його апаратно-програмну реалізацію.

Розглянемо дану проблематику більш детально.

У першу чергу зазначимо, що будь-який веб-ресурс має у своєму складі деякий обсяг статичних (ті, які не зазнають змін у часі) та динамічних сторінок.

При цьому, статичними є веб-документи, структура та зміст яких залишаються сталими.

Веб-документи такого типу надходять до браузеру користувача у вигляді аналогічного до того, у якому вони знаходяться на веб-сервері. Зазвичай статичними є сторінка контактних даних, сторінка з юридичною інформацією компанії, сторінка «про нас» тощо, як далі демонструє рисунок 1.1.

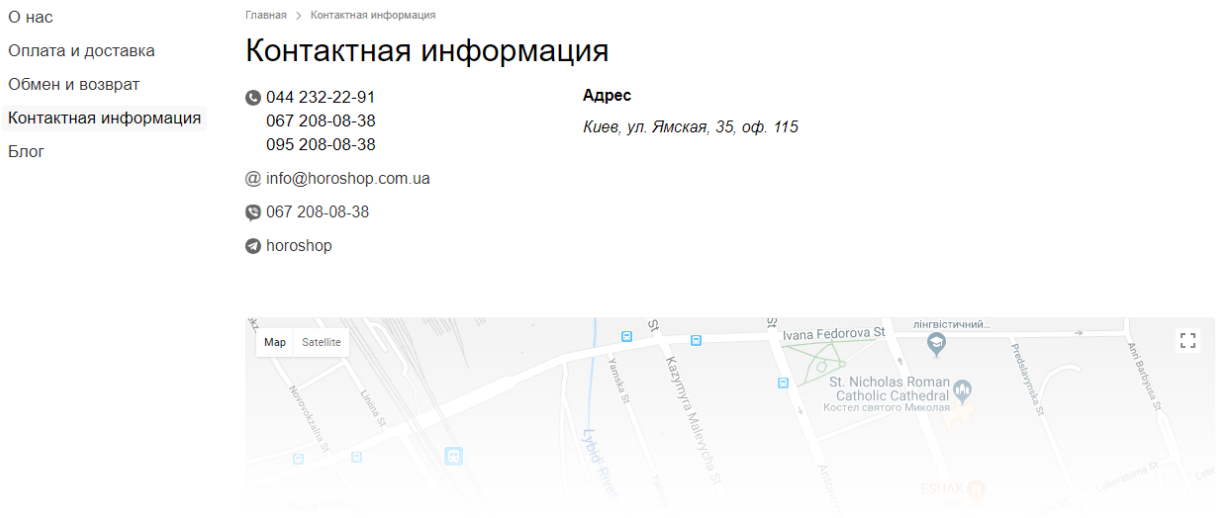


Рисунок 1.1 – Типовий приклад статичного веб-документу (сторінка контактних даних сайту)

На відміну від статичних, веб-документи динамічного типу формуються на рівні серверу за результатами опрацювання клієнтських запитів, які можуть мати ту чи іншу множину параметрів.

Типовим веб-документом, створеним подібним чином, є:

- сторінки пагінації;
- сторінка, що містить у собі результати фільтрації, застосованої відвідувачем сайту.

Так, на рисунку 1.2 представлено приклад веб-документу, який утворено за результатами застосування набору фільтрів характеристик товарів на сайті інтернет-магазину.

Тут слід зазначити, що хоча статичні та динамічні веб-документи можуть бути візуально подібними та мати майже однаковий розмір (кількість біт для опису), водночас, обчислювальне навантаження, яке створюється на веб-сервер у ході генерування динамічного документу, буде суттєво перевищувати даний показник для випадку статичної сторінки.

Така закономірність зумовлена тим, що для формування динамічного документу необхідно задіяти більший обсяг ресурсів для:

- опрацювання клієнтських запитів (обробку необхідно здійснювати у реальному часі);
- побудову коду сторінки з урахуванням необхідності залучення ряду зовнішніх модулів (шрифти, бібліотеки функцій, каскадні таблиці стилів, графічні об’єкти тощо);
- вивантаження сформованого документу у браузер клієнта.

Інакше кажучи, для обробки динамічного веб-документу необхідно зарезервувати суттєво більший обсяг обчислювальних ресурсів, ніж для випадку статичних сторінок. Звідси можна зробити одностайний висновок, що ускладнення механізмів та способів представлення даних клієнтам на рівні серверу веде до зростання обчислювального навантаження на веб-сервер.

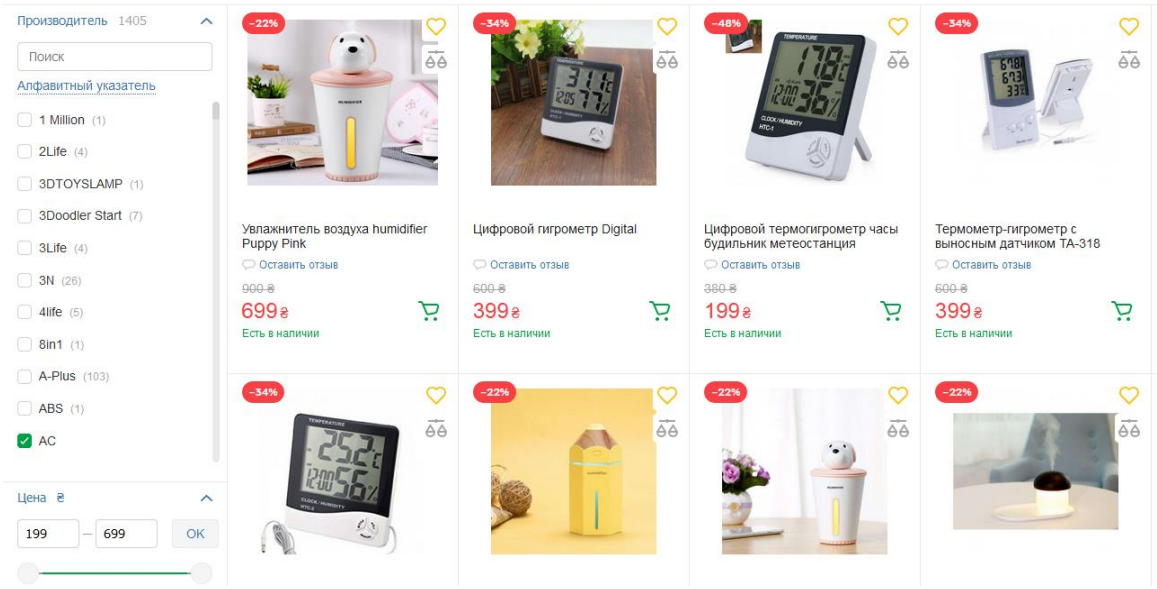


Рисунок 1.2 – Динамічна сторінка з сайту rozetka.com.ua, побудована застосуванням фільтрів

Разом з тим, однією з причин широкого застосування динамічних веб-документів сьогодні є те, що на базі динамічних стоірнок можливо суттєвим чином розширити інструментарій, який може бути використано відвідувачами. При цьому, найбільшою мірою розширенню обсягу інструментів, доступних відвідувачам, сприяють такі зовнішні модулі CMS, як [8]:

- віджети та плагіни;
- API.

Дані засоби, поряд з розширенням функціоналу взаємодії з контентом сайту, дозволяють також реалізувати додаткові інструменти, серед яких:

- бізнес-телефонія;
- інструменти розрахунку вартості замовлення, як демонструє рисунок 1.3;
- засоби інтеграції платіжних систем та служб доставки товарів;

- елементи GIS систем (рис. 1.4) тощо.

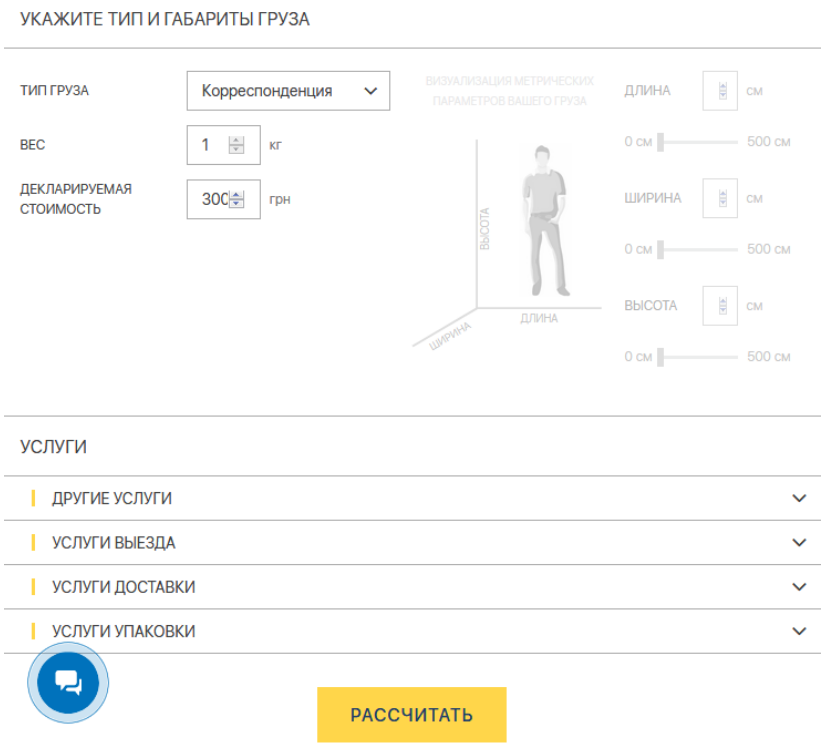


Рисунок 1.3 – Онлайн-інструментарій обчислення вартості доставки товару

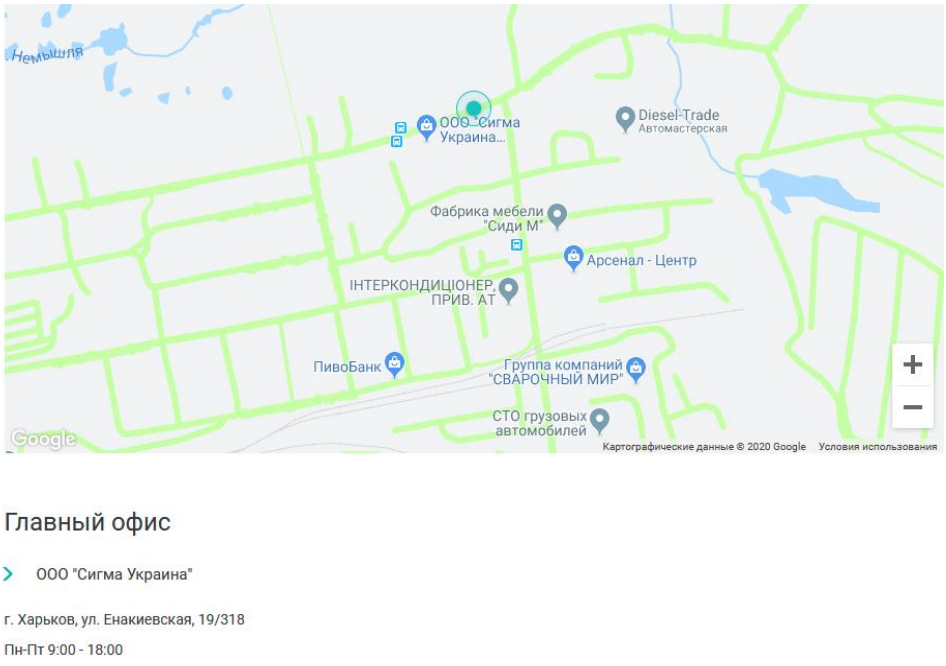


Рисунок 1.4 – Приклад інтеграції API GoogleMap на сторінки сайту

Перелічений вище інструментарій, як і аналогічні засоби, що розширюють функціональні можливості користувача з взаємодії з сайтом, значною мірою спрощують процес виконання цільових операцій на його сторінках. Це, як вже було зазначено, забезпечує умови для покращення як пошукового рейтингу сайту, так і більш ефективного вирішення тих завдань, які на нього покладаються. Зокрема, на випадок веб-ресурсів комерційного характеру у підсумку досягається ріст сукупного прибутку, як наслідок зростання трафіку на сайт, та покращення рівня конвертованості відвідувачів. Водночас, інформаційні ресурси при цьому можуть отримати значне розширення постійної користувацької аудиторії.

Таким чином, з одного боку, збільшення інструментарію, доступного на сайті, що забезпечується зокрема, як інтеграцією додаткових модулів на рівні CMS, так і залученням зовнішньог ПЗ для реалізації обробників.

Разом з тим, з іншого боку збільшення функціональних можливостей для відвідувача у загальному випадку веде до додаткового навантаження, що створює умови для скорочення як показника W_{tech}, так, у підсумку, і W_{func}.

Отже, маємо протиріччя, що ілюструється схемою на рис.1.5.

Рисунок 1.5 – Ситуація невизначеності, викликана протиріччями між функціональністю та продуктивність сайту

Найбільш ефективним шляхом усунення даного протиріччя, є досягнення балансу між обсягом інструментарію, утіленим у межах веб-ресурсу, та підсумковим рівнем його продуктивності. Для цього, у першу чергу, необхідно мати як об'єктивну оцінку W_{tech} технічної продуктивності сайту у цілому, так і стосовно окремих його апаратних пристроїв та програмних рішень. Такі оцінки може біти одержано за результатами виконання комплесного технічного аудиту.

Разом з тим, у процесі функціонування веб-ресурсу будь-якої спрямованості з тих чи інших причин періодично з'являється необхідність реконфігурування:

- його програмної та апаратної складових;
- контентного наповнення сайту.

Виходячи з цього, слід зазначити, що за таких умов заходи з технічного аудиту та наступною технічною оптимізацією за його результатами повинні мати періодичний характер.

1.4 Типові сценарії технічного аудиту сайту

Заходи з технічного аудиту веб-ресурсу у загальному випадку реалізовуються за двома сценаріями, а саме [5, 7, 8]:

- сценарій, що передує етапу тестового розміщення веб-ресурсу (виділений сервер, хостинг);
- сценарій, у ході якого виконується аудит вже функціонуючого проекту.

При цьому, у ході ***першого сценарію*** виконується грутнове тестування ймовірного серверного рішення на апаратному рівні. У ході цього, підлягає вирішенню ряд питань стосовно вибору апаратного забезпечення та обгрунтування попередньо зробленого вибору, а саме:

1. Вибір способу розміщення веб-ресурсу у мережі (фізичний сервер чи хостинг), а також конкретних параметрів у рамках обраного рішення.
2. Пошук та застосування найбільш прийнятної CMS. У даному випадку попередній вибір на користь тієї чи іншої системи керування контентом виконується з урахуванням:

- типу сайту (цільова сторінка, інтернет-магазин, інтернет-представництво тощо);
- нішевої належності сайту;

- існуючої специфіки контенту;

- базового функціоналу, який надає CMS.

У ході цього першочергово ураховується продуктивність попередньо обраної системи керування контентом для обраного апаратного оточення. Тобто, CMS обирається з точки зору можливості забезпечення виконання вимог з продуктивності проекту, як показано виразом (1.1).

У свою чергу, якщо технічний аудиту виконується за ***другим сценарієм***, тобто, відносно функціонуючого сайту, у загальному випадку тестуванню підлягають:

- спосіб розміщення та подання контенту на сторінках;
- раціональність функціонування програмної складової як у цілому, так і окремими модулями (окремо дослджується як клієнтський, так і серверний рівень);
- ступінь раціональності реалізації взаємодії програмної та апаратної складових.

На випадок вже існуючого проекту результатами виконання технічного аудиту є визначення рівня відповідності одержаних технічних показників аналогічним показникам для сайтів-конкурентів, що належать ТОП-10 у рамках поточної ніші.

При цьому у випадку, якщо виконана за підсумком технічного аудиту оптимізація веб-ресурсу не створила умов для покращення ключових показників (зокрема, це може бути час завантаження сторінки у браузер) до оптимального (або хоча б прийнятного) рівня, надалі реалізуються заходи з більш детального аудиту. До таких заходів додається дослідження апаратного забезпечення на системи керування контентом, як це виконується у рамках першого сценарію. Після цього, керуючись одержаними у підсумку аудиту даними, може виконуватися зміна або модифікація CMS, так і апаратної платформи, чи їхніх окремих складників.

Тобто, за таких умов тестування та ймовірна доробка чи переналаштування виконується відносно кожної зі складових сайту, як показує рисунок 1.6.

Проекти, які вже є активними, рекомендується тестувати з деякою періодичністю для того, щоб створити умови для вчасного виявлення проблемних аспектів їх функціонування і тим самим забезпечити продуктивності на прийнятному рівні.

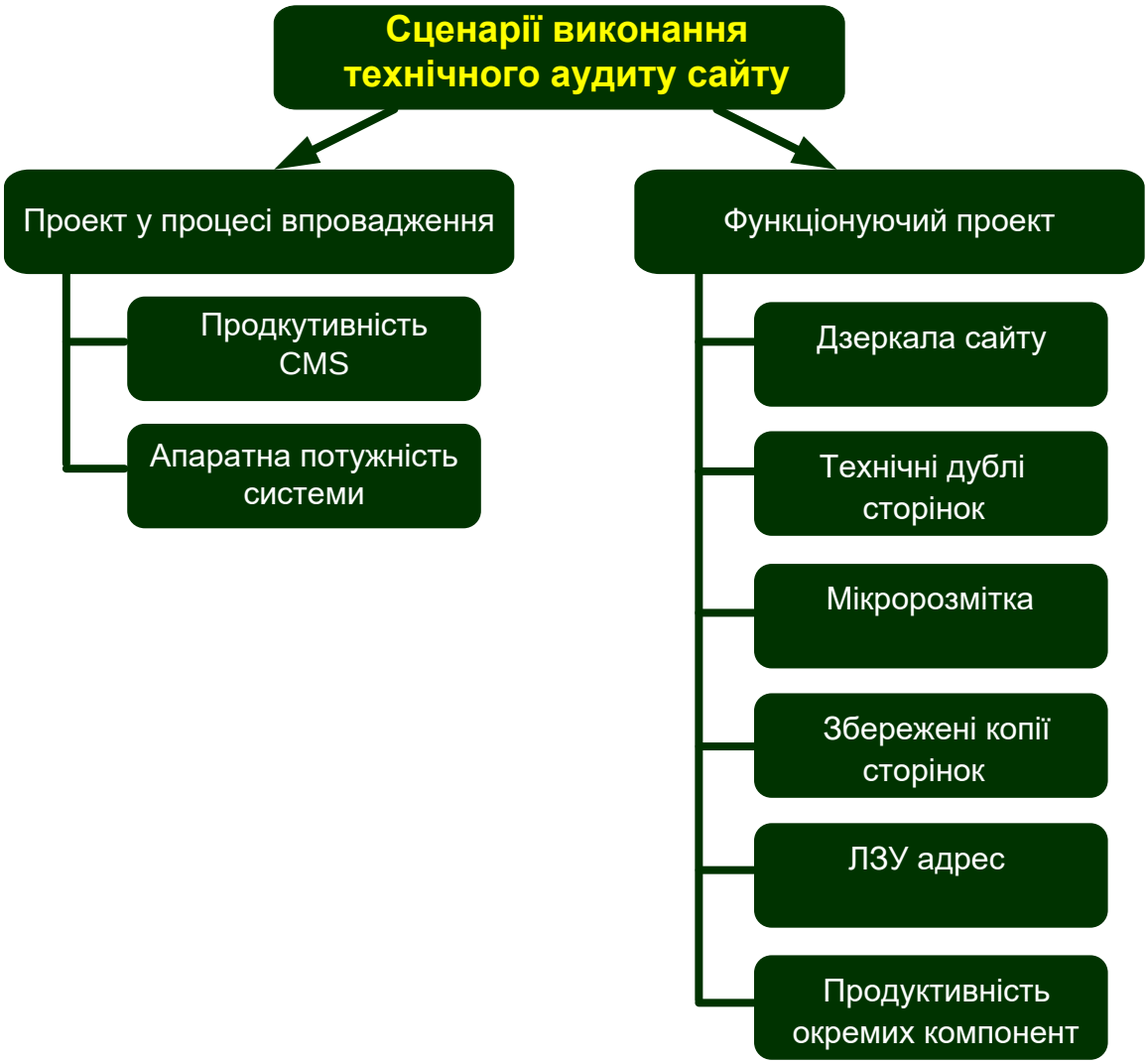


Рисунок 1.6 – Типовий склад кожного з базових сценаріїв реалізації технічного аудиту веб-вузла

Далі розглянемо більш детально кожен з зазначених сценаріїв виконання технічного аудиту

2. РЕАЛІЗАЦІЯ ЗАХОДІВ З ТЕХНІЧНОГО АУДИТУ ВЕБ-РЕСУРСА

2.1 Технічний аудит вузла на стадії впровадження проекту

Раніше зазначалося про те, що на даній стадії життєвого циклу веб-проекту виконується дослідження серверної системи, або хостингу, на базі яких передбачається розміщувати сайт, а також CMS [6-8].

При цьому, до параметрів серверної системи, що безпосередньо впливають на показник підсумкової технічної продуктивності, у загальному випадку належать:

- показник обчислювальної спроможності процесору, або підсистеми процесорів (на випадок мультипроцесорної реалізації платформи);
- параметри RAM, а саме – доступний фізичний обсяг та її швидкодія;
- показники, що відображають продуктивність дискових накопичувачів; тут важливими є швидкісні параметри, ступінь надійності та фактичний обсяг.

Разом з тим, у ході дослідження рівня продуктивність програмної складової веб-серверу зазвичай виконується оцінка таких ключових параметрів, як [6, 7, 9]:

- швидкодія інструментів та середовища розробки і підтримки програмних модулів на рівні серверу. Це, у першу чергу, є актуальним для php;
- підсумкова швидкодія бази даних. Дослідження виконується за аналогічними сценаріями незалежно від типу застосованої БД (MySQL, PostgreSQL тощо);
- показники фактичної ресурсоємності програмної платформи на боці серверу.

У свою чергу, у рамках оцінки продуктивності CMS, першим кроком є виконання аналізу відповідності попередньо застосованої системи управління змістом щодо ймовірного переліку завдань, які сайт має вирішувати.

Для цього, у загальному випадку, у першу чергу необхідно взяти до уваги ряд ключових характеристик таких систем (таблиця 2.1).

Таблиця 2.1 – Ключові характеристики найбільш відомих систем керування змістом сайту, до беруться до уваги у ході первинної оцінки CMS у рамках виконання технічного аудиту

CMS	Галузь застосування			Безчність системи	Рівень швидкодії системи	Ресурсоємність	Недоліки
	Інтернет-магазини	Блоги	Соціальні мережі				
Joomla	+	+	+	Середня	Середня	Середня	Специфічна архітектура системи
Drupal	+	+	+	Найвища	Висока	Висока	Надмірна складність системи
WordPress	+	+	-	Висока	Середня	Висока	Обмеженість каталогу
OpenCart	+	-	-	Висока	Висока	Середня	Вузька спеціалізація
Bitrix	+	+	+	Найвища	Середня	Середня	Специфічна архітектура системи

Також разом з переліченими характеристиками CMS важливими є особливості кожної з платформ, що кардинальним чином впливають на підсумкову продуктивність реалізованих на їх базі проектів. Це, у першу чергу, такі особливості, як [10, 11]:

- CMS Drupal, попри можливість забезпечення найвищих показників продуктивності та безпеки характеризується високим рівнем вимог щодо параметрів апаратної складової серверу та/або хостингу. При цьому, платформа відзначається високим рівнем складності;
- у рамках CMS Joomla на сьогодні існує значна кількість розширень для створення інтернет-магазинів. Водночас, більшість таких рішень рішення є недостатньо масштабованими, а також гнучкими з точки зору можливості конфігурування. Наприклад, JOOMShopping може бути використано для лише для невеликих проектів, тоді як RedSHOP – навпаки орієнтований на побудову та підтримку масштабних проектів та не є

прийнятним для магазинів середнього та невеликого розмірів. Така закономірність є справедливою і на випадок сайтів інших типів;

- платформа WordPress забезпечує високий рівень технічної продуктивності проектів, коли мова йде стосовно сайтів, масштаб яких умовно може бути охарактеризовано, як «нижче середнього» (тобто, у його складі знаходиться не більш, ніж 300-400 сторінок); слід зазначити також, що стандартне розширення WordPress – WooCommerce, на базі якого здійснюється розробка та підтримка інтернет-магазинів, є ефективним для каталогу, розмір якого не перевищує кількох десятків карток товару; у свою чергу, збільшення каталогу спричинює суттєве зменшення швидкодії сайту;
- CMS OpenCart, що являє собою потужну платформу для створення онлайн-магазинів будь-яких розмірів та конфігурацій, є обмеженою в плані можливостей реалізації та супроводження проектів іної спрямованості на її базі;
- по мірі розширення масштабу проектів, реалізованих з використанням Bitrix, виникає необхідність нарощування апаратної потужності серверу/хостингу.

Таким чином, заходи з технічного аудиту веб-ресурсу для випадку проекту, що знаходиться на стадії впровадження, чи первинного аналізу апаратного базису, передбачають застосування інструментів тестування наступних компонент:

- процесор, або процесори, задіяні у системі (CPU);
- модулі оперативної пам’яті (RAM);
- пристрої зберігання даних (дискова підсистема);
- база даних;
- середовище php.

При цьому, дослідження ймовірного рівня продуктивності системи управління контентом, тобто, прийнятності поточної CMS для реалізації тих чи інших проектів, має аналітичний характер.

У підсумку дослідження CMS та наявної апаратної складової сайту, надалі робиться висновок щодо того, наскільки ефективно побудовано серверну систему у цілому.

Окрім цього, формується прогноз відносно ефективності функціонування клієнтського та серверного ПЗ.

2.1.1 Тестування серверної платформи

Одним з інструментів тестування серверної платформи на рівні окремих компонент є крос-платформенна утиліта Sysbench [12].

Даний засіб може використовуватися як у ході підготовки до розміщення сайту у мережі, так і на стадії тестування веб-ресурсу у складі того чи іншого апаратного оточення.

Окрім того, що інструментарій Sysbench може застосовуватися на базі багатьох ОС, йому властиві також:

- модульна реалізація;
- простота налаштування та використання;
- можливість виконувати тестування багатьма паралельними потоками;

За дапомогою Sysbench може бути виконано оцінку більшої частини показників апаратної платформи, що у нашому випадку є актуальними.

Слід зазначити, що Sysbench є засобом, який спершу було розроблено для використання у межах Unix-платформ, зокрема - Ubuntu/Debian.

Таким чином, у найпростішому випадку інсталяція засобу потребує виконання відповідного запиту до офіційного репозиторію зазначених вище ОС. Для цього слугує команда:

```
apt-get install sysbench
```

Далі виконується процедура компіляції, за результатами якої засіб стає готовим до використання:

```
$ ./autogen.sh
$ ./configure
$ make
```

У свою чергу, синтаксиси команд Sysbench відзначається простотою та є уніфікованим, тобто, команди мають формат, аналогічний дял випадку виконання тестів будь-яких типів.

Команди у Sysbench мають наступний формат:

```
test=[ідентифікатор тесту] [опції] [режим]
```

При цьому, встановлення необхідного режиму тестування реалізується записом необхідного параметру у полі «ідентифікатор тесту», що приймає наступні значення:

- prepare, що є ідентифікатором підготовчої стадії до безпосереднього тестування;
- run, є активатором тесту.

За замовчуванням будь-який тест виконується у одному потоці.

Разом з тим, коли виникає потреба паралельних обчислень, необхідно вказати ключ:

```
--num-threads=N,
```

де N – необхідний обсяг потоків паралельних обчислень.

2.1.2 Тестування процесорної системи

Зазвичай у ході технічного аудиту у рамках дослідження апаратної складової у цілому першочергово виконується тестування CPU, або процесорної системи.

Даний підхід щодо черговості виконання тестів зумовлюється тим, що в умовах пікових навантажень обчислювальна потужність процесору являє собою базовий параметр на випадок веб-ресурсів певних типів, зокрема – інтернет-магазинів.

Відтак, якщо тестування процесору показує, що досліджуваний тип CPU в існуючих умовах не є прийнятним, це означає, що виконання інших тестів недоцільне, та з великою ймовірністю система потребує модифікації процесорної системи, або апаратної платформи у цілому.

За стандартною схемою тестування процесору за допомогою Sysbench, необхідно ініціювати процес розрахунку 20000 простих чисел, що виконується командою:

```
$ sysbench --test=cpu -- cpu-max-prime=20000 run
```

У підсумку, система виводить на консоль отримані результати тестування процесору у наступному вигляді:

```
Test execution summary:
 total time:                              17.3915s
 total number of events:                  10000
 total time taken by event execution:     17.3875
 per-request statistics:
   min:                                   1.66ms
   avg:                                   1.74ms
   max:                                   4.00ms
   approx.  95 percentile:                2.03ms
```

При цьому, найважливішим параметром серед наведених є показник *total time*.

У свою чергу, вибір апаратної платформи, що має найбільшу обчислювальну потужність, зазвичай виконується за рахунок тестування ряду серверних систем, після якого отримані значення показника *total time* порівнюються.

2.1.3 Тестування дискових пристроїв

Процес тестування дискових пристроїв з використанням Sysbench передбачає послідовне виконання ряду дій, а саме:

- свтворення сукупності тестових файлів;
- безпосереднє тестування з фіксацією одержаних показників;
- видалення множини тестових файлів.

У ході першого етапу тестування дискових пристроїв генерування множини тестових файлів виконується наступною командою:

```
$ sysbench --test=fileio -- file-total-size=[size,
Gb] prepare,
```

тут поле [size, Gb] слугує для запису підсумкового розміру множини сформованих файлів, поданого у гігабайтах, наприклад:

```
$ sysbench --test=fileio -- file-total-size=[70G]
prepare
```

Розглянутий приклад демонструє команду створення сукупності тестових файлів, що має підсумковий розмір 70 Гб.

Важливим тут є те, що для забезпечення високого рівня достовірності результатів тестування підсумковий розмір множини згенерованих файлів має суттєво перевищувати, наявний об’єм RAM. За таких умов вплив кешу ОС на результати тестування буде нівельовано.

Далі, після того, як множину тестових файлів згенеровано, реалізується безпосередньо саме тестування.

Зазначимо, що у даному випадку, як і тоді, коли тестуванню підлягала процесорна система, стандартизований режим тесту передбачає його виконання в одному потоці, за замовчуванням перевірка виконується в одному потоці.

Разом з тим тоді, коли попередньо було створено множину тестових файлів загальним обсягом 70 Гб, ініціація тесту виконуватиметься за допомогою команди:

```
$ sysbench --test=fileio -- file-total-size=70G --
file-test-mode=rndrw --init-rng=on -- max-time=300
--max-requests=0 run
```

у розглянутому прикладі команди застосовано режим випадкового читання (file-test-mode=rndrw), при цьому, якщо не зазначено іншого, до для виконання тесту встановлюється 300-секундний ліміт (max-time=300). Далі після того, як час виконання тесту буде вичерпано, утиліта демонструє одержані результати у наступному вигляді:

```
    Тест Fileio:
    Operations performed:  249517 Read, 166344 Write,
532224 Other = 948085
    Read 3.8073Gb  Written 2.5382Gb  Total transferred
6.3455Gb  (21.659Mb/sec)
     1386.18 Requests/sec executed

    Test execution summary:
        total time:                          300.0045s
        total number of events:              415861
        total time taken by event execution: 178.9646
        per-request statistics:
```

```
             min:
0.00ms
             avg:
0.43ms
             max:
205.67ms
             approx.                 95        percentile:
0.16ms

     Threads fairness:
         events (avg/stddev):           415861.0000/0.00
         execution time (avg/stddev):   178.9646/0.00
```

Тут найважливішим показником, що характеризує продуктивність дискових накопичувачів, є значення середньої швидкості передавання даних, яка у нашому випадку складає 21,659 Мб/сек.

На наступному кроці виконання тесту попередньо сформовані тестові файли видаляються. Для виконання зазначеної операції слугує команда:

```
$ sysbench --test=fileio cleanup
```

2.1.4 Виконання тесту бази даних MySQL

Sysbench дозоляє оцінити швидкодію бази даних з точки зору оптимальності OLTP. У свою чергу, OLTP, або Online Transaction Processing (система транзакцій) являє собою середовище опрацювання транзакцій у реальному масштабі часу. У сутності, OLTP є методом побудови баз даних, який передбачає, що система обробляє транзакції невеликих розмірів, проте за одиницю часу необхідно обробити великий масив таких транзакцій. Першочерговою тут є необхідність мінімізації часу відгуку системи.

Під час тесту баз даних MySQL утиліта Sysbench фіксує швидкісні показники часу опрацювання MySQL-транзакцій. При цьому, кожна з транзакцій у загальному потоці включає у себе запити на запит та зчитування.

Перевірка БД у даному випадку потребує попереднього стоврення тестової бази даних, та задання параметрів тестування, як показано наступним прикладом:

$ sysbench --test=oltp -- oltp-table-size=1000000 --mysql-db=test --mysql-user=root --mysql-password=pass prepare

Дана команда зазначає розмір тестової бази даних, яку необхідно згенерувати. Також тут встановлюються:

- час виконання тесту;
- уточнення типу бази даних;
- параметри доступу до бази даних за замовчуванням (user=root --mysql-password=pass prepare), оскільки мова тут іде відносно БД, яка одразу створюється.

Після виконання перелічених вище дій може виконуватися саме тестування. Для уього необхідно виконати відповідну команду, яка у нашому випадку буде наступною:

$ sysbench --test=oltp -- oltp-table-size=1000000 --mysql-db=test --mysql-user=root --mysql-password=pass -- max-time=60 --oltp-read-only=off --max-requests=0 --num-threads=8 run,

у якій параметр --oltp-read-only зазначає тип запитів, що використовуватимуться під час тестування. Разом з тим, за умови використання параметру --oltp-read-only=off, тестування буде включати у себе лише запити на зчитування. За таких умов з'являється можливість тестування БД, наприклад, у режимі, slave-бази.

Результати, одержані у підсумку виконання тесту, надаються у наступному форматі:

```
OLTP test statistics:
    queries performed:
        read:                            564158
        write:                           0
        other:                           80594
        total:                           644752
    transactions:                        40297  (671.57 per sec.)
    deadlocks:                           0      (0.00 per sec.)
    read/write requests:                 564158 (9402.01 per sec.)
    other operations:                    80594  (1343.14 per sec.)
```

Test execution summary:
total time: 60.0040s
total number of events: 40297
total time taken by event execution: 479.8413
per-request statistics:
min: 1.14ms
avg: 11.91ms
max: 70.93ms
approx. 95 percentile: 15.54ms

При цьому, показником підсумкової продуктивність БД MySQL може слугувати об'єм транзакцій за секунду (transactions per sec). Означений параметр може бути застосовано як у якості метрики, що у цілому характеризує швидкодію бази даних, так і під час вибору тієї чи іншої конфігурації серверу шляхом їх порівняння за даним показником.

2.1.5 Тестування PHP-оточення

У процесі оцінки ефективності веб-ресурсу на рівні серверу, тестування php є одним з ключових елементів даного дослідження.

Важливість виконання даного тесту пояснюється існуючим протиріччям, сутність якого полягає у наступному:

- з одного боку, на базі технічного аудиту CPU, RAM, дискових накопичувачів а також БД у загальному випадку з високою вірогідністю може бути сформовано прогноз щодо раціональності функціонування web-додатків у рамках досліджуваної системи;

- з іншого боку, нерідко виникають умови, коли згідно результатів тестування з використанням Sysbench система система визнається продуктивною, разом з тим, php-додатокам відповідають неприпустимо низькі показники продуктивності.

Зазвичай загальний рівень продуктивності php залежить від:

- версії php, що використовується у системі;
- присутність акселератору;
- того, наскільки коректно було зкомпільовано php;
- кількість інстальованих та активованих розширень php.

Разом з тим, Sysbench не дозволяє виконати тестування php, так як не підтримує необхідні функції.

Відтак, оцінка процесу функціонування php вимагає застосування інших інструментів, серед яких – спеціалізована утиліти Af php bench, яка є орієнтована на виконання саме такого класу завдань [13].

Інстурментарій Af php bench наближено можна розглядати як бібліотеку скриптів, що реалізують функціонал детальної оцінки параметрів роботи php-оточення. Недоліком даного засобу є неможливість тестування php версій, які є нижчими, ніж 5.4.

Af php bench є відкритим рішенням. Завантаження засобу здійснюється з серверів github.com.

Відповідоно, для його завантаження виконуються відповідні команди, як показано далі:

```
$ wget github.com/florinsky/af-php-
bench/raw/master/build/phpbm.phar
$ php phpbm.phar
```

Тест php-оточення включає у себе 10 кроків досліджень. Тут передбачається що, протягом кожного з кроків досліджується один з параметрів, що тією чи іншою мірою характеризує роботу php.

При цьому, усі наявні тести логічно поєднуються у такі категорії, як:

1. General, або тести загальної групи, до якої відносяться:

- цикли;
- дослідження операцій на базі рандомізації;
- створення та видалення об'єктів.

2. Strings, або група тестів дій з рядками, що містить у собі:

- два типи тестів рядків;
- implode/explode;
- тестування процесу обчислення хешів.

3. Arrays – група тестів, що стосується роботи з масивами, а саме:

- тест роботи масивами файлів;
- тести сортування у двох режимах.

За підсумком тестування отримується детальний результат по кожному з 10 тестів, групованих за категоріями

Після виконання комплексного тестування отримуємо розгорнутий результат по кожній з трьох категорій, наприклад:

```
[GENERAL]
1/10 Cycles (if, while, do)  ..................  6.72s
2/10 Generate Random Numbers  .................  3.21s
3/10 Objects  ..................................  4.82s
Time: .. 14.76

[STRINGS]
4/10 Simple Strings Functions  ................  13.09s
5/10 Explode/Implode  .........................  15.90s
6/10 Long Strings  ............................  30.37s
7/10 String Hash  .............................  23.57s
Time: .. 82.93

[ARRAYS]
8/10 Fill arrays  .............................  22.32s
9/10 Array Sort (Integer Keys and Values)  ....  17.17s
10/10 Array Sort (String Keys and Values)  .....  14.29s
Time: .. 53.79

TOTAL TIME: . 151.47
```

У даному випадку ключовим та найбільш важливим параметром, що у цілому дозволяє судити про раціональність функціонування php, є загальний час виконання тесту (total time). Разом з тим, формат представлення результатів, як видно з наведеного прикладу, дозволяє окремо оцінювати вклад кожного з досліджених процесів у рамках зазначених 3 категорій тестів, у підсумкове значення total time.

У свою чергу, те, що Af php bench надає змогу оцінювання окремих показників, є важливою перевагою утиліти. Це пояснюється тією обставиною, що у ряді випадків загальний низький рівень продуктивності php може біти наслідком неефективного функціонування лише одного механізму. Так, наприклад, значною мірою впливати на підсумкову продуктивність у бік її зниження здатні:

- функції, орієнтовані на обробку довгих рядків (*Long Strings*);
- збої у роботі генератора випадкових чисел.

Таким чином, як Af php bench, так і Sysbench забезпечують можливість детального дослідження продуктивності апаратної складової веб-ресурсу.

Проте слід зазначити, що, дані засоби дозволяють виконувати дослідження лише у деякий поточний момент часу t.

Разом з тим, на деякий інший момент часу, наприклад, (t+1), та за інших умов, наприклад, коли на сервер попередньо завантажено CMS та додано контент ненаповнення сторінок (у сутності, розміщено сайт), апаратна складова зазнаватиме навантаження ζ. Відтак, у даних умовах виміряний рівень технічної продуктивності $W(0)_{tech}$ апаратної частини за відмутності навантаження буде суттєво різнитися з рівнем технічної продуктивності, виміряної з навантаженням ζ, що еквівалентно виразу:

$$W(0)_{tech} \neq W(\zeta)_{tech} \quad (2.1)$$

Водночас, показник продуктивності апаратної складової серверу залежить від процесів, які виконується паралельно на його базі.

Найбільшою мірою це проявляється тоді, коли один фізичний сервер застосовується для розміщення тієї чи іншої кількості веб-ресурсів. При цьому значення сумарного навантаження, якого зазнає апаратна складова, залежить у загальному випадку від:

- кількості сайтів, розміщених на сервері;
- об’єму обчислювальних потужностей (апаратних ресурсів) які кожному з розміщених сайтів є гарантованими;
- множини технологічних рішень, на базі яких побудовано кожен з розміщених веб-ресурсів;
- діючого трафіку кожного сайту у межах апаратного серверу.

Отже, тестування апаратної складової ще до розміщення сайту у мережі дає змогу лише оцінити її ймовірний потенціал стосовно можливості забезпечувати підтримку веб-ресурсів певних типів та масштабу.

Відповідно, розглянутий підхід до тестування апаратної платформи серверу дозволяє обрати найбільш оптимальне серверне рішення за рахунок порівняння сукупностей технічних показників деякого переліку систем.

Досліджений сценарій реалізації заходів з технічного аудиту апаратної складової веб-серверу доцільний для попередньої оцінки системи.

Водночас, уся множина застосовуваних при цьому інструментів є недостатньою для випадку технічного аудиту серверної системи, у межах якої розміщено той чи інший обсяг веб-ресурсів.

За таких умов доцільно розглядати будь-який веб-ресурс у вигляді поєднання змістовного наповнення а також апаратних та програмних модулів.

2.2 Виконання технічного аудиту активних проектів

Уся множина засобів тестування функціонуючих веб-ресурсів умовно поділяється на групи загальних та спеціалізованих [8].

При цьому, загальна група містить у собі такий перелік засобів, які дають змогу виконати оцінку як технічної продуктивності сайту, так і окремо сторінок у його складі.

Таке оцінювання може виконуватися за одним або декількома параметрами.

Тут слід зазначити, що засоби тестування апаратних складових серверів являють собою утиліти, у той час, як інструментарій виконання технічного аудиту функціонуючих веб-проектів більшою чистиною являє собою сервіси з онлайн-доступом.

Такий формат реалізації аналітичних інструментів дозволяє виконувати оцінку продуктивності сайту на технічному та функціональному рівнів без необхідності використання додаткових засобів. Це, у свою чергу, забезпечує скорочення часу, необхідного для аналізу.

При цьому, перелік заходів дослідження активних проектів дозволяє виконати як окремий аналіз усіх показників, наведених рис. 1.6., так і системи у цілому.

На цей випадок оцінюється підсумкова ефективність поєднання таких складових сайту, як серверної платформи, backend та frontent компонент і його змістовного наповнення.

У загальному випадку до складу заходів технічного аудиту функціонуючого веб-ресурсу відносяться дослідження:

- загальної продуктивності системи, та продуктивності її окремих складових, як раніше було показано рис.1.6;
- присутносіт дзеркал сайту;
- наявності та наближеного обсягу технічних дублів сторінок сайту;
- факту присутності та використаного типу мікророзмітки;
- збережених копій сторінок;
- ЛЗУ адрес.

2.2.1 Оцінка загальної продуктивності системи

Засіб Google Page Speed Insights

Застосування онлайн-сервісу Google Page Speed Insights [14, 15] дозволяє здійснити експрес-оцінювання загальної раціональності реалізації веб-ресурсу.

При цьому передбачається, тестування виконується для сторінок сайту, а не для ресурсу у цілому.

У підсумку такого тестування фіксується відноста швидкість завантаження поточної сторінки у браузер клієнта. Така оцінка виконується для базової та мобільної версій сторінок (рис.2.1).

Верхня межа шкали відносних швидкостей Google Page Speed Insights обмежена показником 100. У свою чергу, доступний діапазон значень поділено на такі ділянки, як:

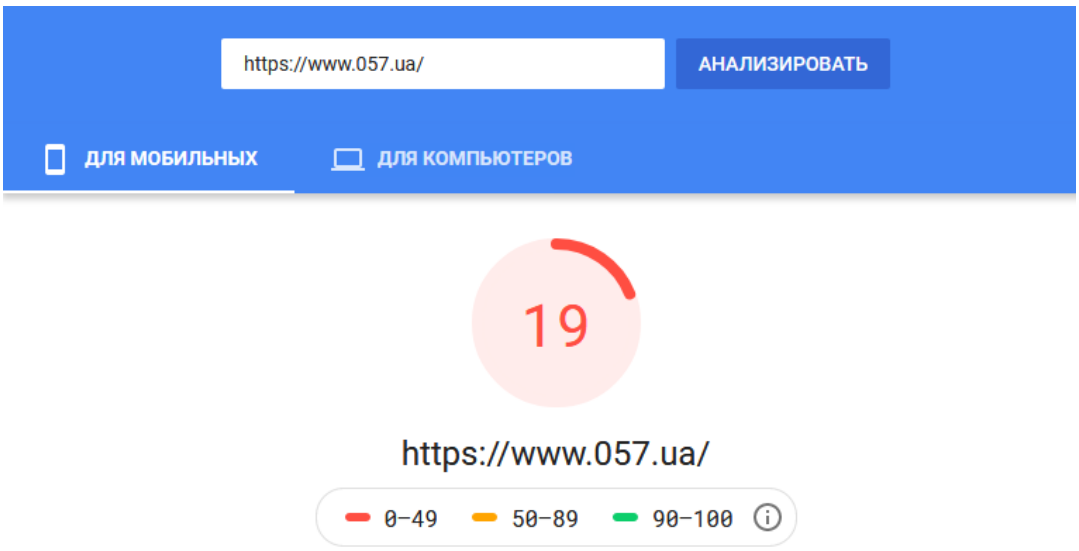
- зелена ділянка (100-90 пунктів швидкості);
- жовта зона (89-50 пунктів швидкості);
- червона зона (49-0 пунктів швидкості).

У цьому випадку на базі оцінки рівня відносної швидкості завантаження сторінки може бути зроблено висновок про те, наскільки є ефективними реалізація та налаштування (а відтак - продуктивність) програмної та апаратної складових сайту, приймаючи до уваги backend та frontend компоненти.

Таке дослідження щвидкісних кондицій окремих сторінок на базі Google Page Speed Insights у процесі технічного аудиту надає ряд переваг, оскільки при цьому:

- комплексне питання оптимізації веб-ресурсу локалізується до рівня окремих сторінок, що формують змістовну частину сайту;
- держані швидкісні показники сторінки безпосередньо впливають на процес їх ранжування (деякою мірою це стосується сайту у цілому) у межах пошукової системи Google; зокрема, сторінка, що належить «зеленій ділянці», є оптимізованою, тобто додаткові дії з позиції технічної доробки чи заходів SEO для неї не потрібні, у свою чергу, належність сторінок «жовтій ділянці» свідчить про те, що мають місце певні проблеми, що потребують відповідних дій;
- інструментарій Google Page Speed Insights супроводжує результати тестування швидкісних параметрів сторінки загальними рекомендаціями з оптимізації її ймовірної оптимізації.

а)



б)

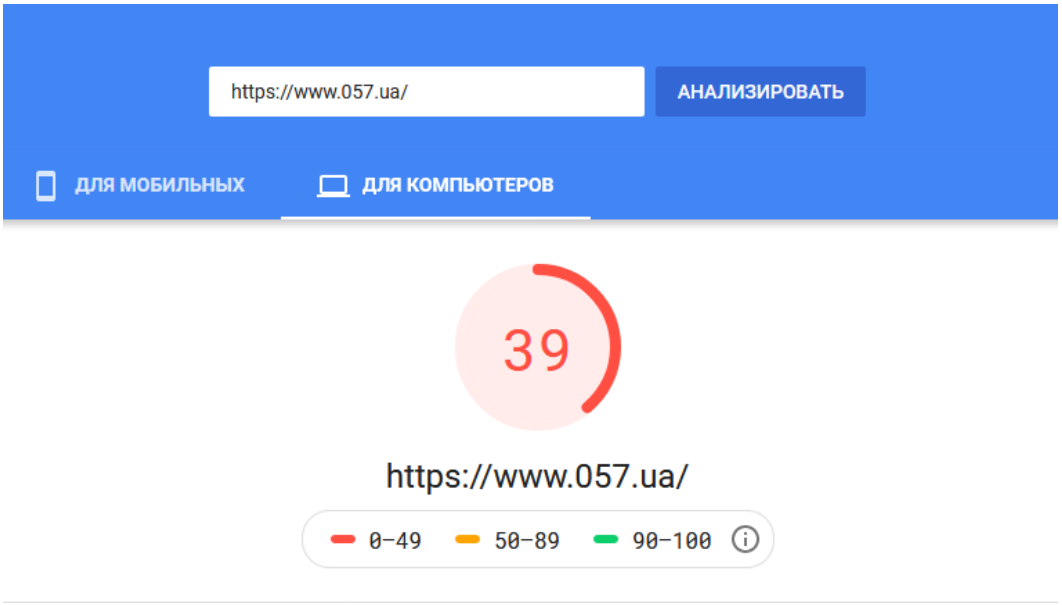


Рисунок 2.1 – Оцінка швидкісних показників головної сторінки сайту у мобільній а) та базових б) версіях

Як показує наведений рис.2.1 приклад, сторінці сайту в мобільному та базовому форматі відповідає «червонв зона» діапазону швидкості.

У загальному випадку це може свідчити про те, що потребує оптимізації повний перелік наведених складових сайту, або, щонайменше, частина з них, а саме:

- система керування контентом (CMS);
- використовуваний у поточний час шаблон CMS;
- кодовий опис сторінки та її контент не наповнення;
- апаратна складова веб-ресурсу.

Інструментарій Dotcom Monitor

Як і Google Page Speed Insights, засіб Dotcom Monitor являє собой онлайн-інструментарій, використовуючи який забезпечується більш ґрунтовне, ніж на випадок попередньо розглянутого засобу, дослідження веб-ресурсу, який необхідно проаналізувати з позиції загальної продуктивності [16], а саме:

1. Оцінити продуктивність функціонування веб-додатків. Для забезпечення даної можливості у рамках сервісу передбачено підтимку тестування програмних рішень з використанням технологій Silverlight, Ajax, Adobe Flash, а також JavaScript;

2. Виміряти швидкість, з якою окремі сторінки сайту завантажуватимуться для випадку різних браузерах (тест включає у себе Chrome, Internet Explorer, Safari, Mozilla Firefox, та на сьогодні понад 50 різних мобільних браузерів;

3. Дослідити функціонування веб-сервісів, що включає у себе:

- оцінку часу відгуку веб-серверу для випадку використання методів GET і POST;
- опит серверу на базі команди ping, у ході чого виконується звернення з понад 20 локацій;
- оцінка швидкодії різних компонент системи у випадку використання telnet тощо.

4. Виконати комплексну перевірку функціонування ряду поширених сервісів та протоколів, зокрема таких, як SSH, Streaming Media, Email POP3/IMAP4, FTP, UDP, SMTP.

5. веб-ресурсом можна вважати такий сайт, який має високій рівень довіри з боку як пошукових систем, так і з боку користувачів.

Рисунок 2.2 демонструє базовий інтерфейс онлайн-засобу Dotcom Monitor.

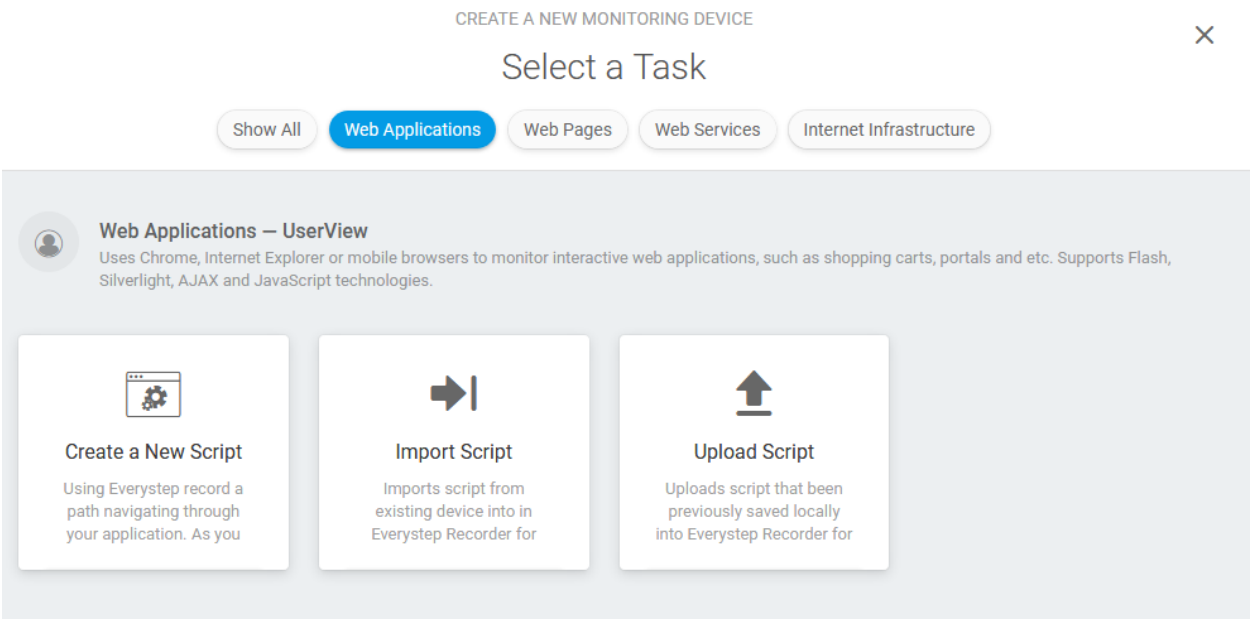


Рисунок 2.2 – Базовий інтерфейс засобу Dotcom Monitor

На той випадок, коли необхідно виконати оцінку поточного комплексного показнику функціонування веб-ресурсу, може бути застосовано засіб Page Speed Insights. Разом з тим, за потреби виконання значно детальнішого дослідження, доцільно використовувати обидва розглянуті засоби спільно.

При цьому, зібрані за участю одного, чи деякого переліку аналітичних інструментів різнорідні параметри веб-ресурсу показують ступінь раціональності його побудови та/або налагодження.

Це, у свою чергу, є справедливим також і до ряду компонент веб-ресурсу, серед яких:

- апаратна та програмна реалізація сервету та/або параметри хостингу;
- відповідність CMS покладеним на неї завданням;
- API, плагіни, віджети та інші типи розширень;
- backend та frontend складові;
- контентне наповнення сайту.

На сьогодні окрім наведеного інструментарію комплексного дослідження активних проектів, широко застосовуються подібні засоби у складі онлайн-сервісів, таких, як 2IP, Serpstat або LiveInternet. Проте набагато більш інформативними у плані можливості виявлення локальних проблем на рівні окремих сторінок сайту є т.з. тести-водоспади. Можливість виконувати таке тестування надає, зокрема, сервіс GTMetrix.

GTMetrix

У рамках GTMetrix вимірюються абсолютні швидкісні показники завантаження сторінок. За замовчування час представлений у секундах [17].

Тестування кожного веб-документу при цьому є поглибленим та передбачає збір ряду параметрів, де першочерговими є:

1. Підсумкова продуктивність (Performance), яка є комплексним значенням та формується на базі таких показників, як (рис.2.3):

- абсолютна швидкості завантажень контентної частини та коду сторінки у браузер (Performance Metrics);

- швидкість налагодження взаємодії між сервером та браузером у ході завантаження тестованої сторінки (Browser Timing).

2. Дані тесту-водоспаду (Waterfall), який являє собою візуалізацію черговості та часу завантаження складових елементів веб-документу до клієнтського браузеру (рис. 2.4).

3. Структурні характеристики веб-документу (рис.2.5), на базі дослідження яких виявляються недоліки розміщення складових сторінки та надаються загальні рекомендації щодо їх вирішення.

Окрім цього також здійснюється оцінювання підсумкової ефективності веб-документу (рис.2.6) на базі усіх зазначених показників.

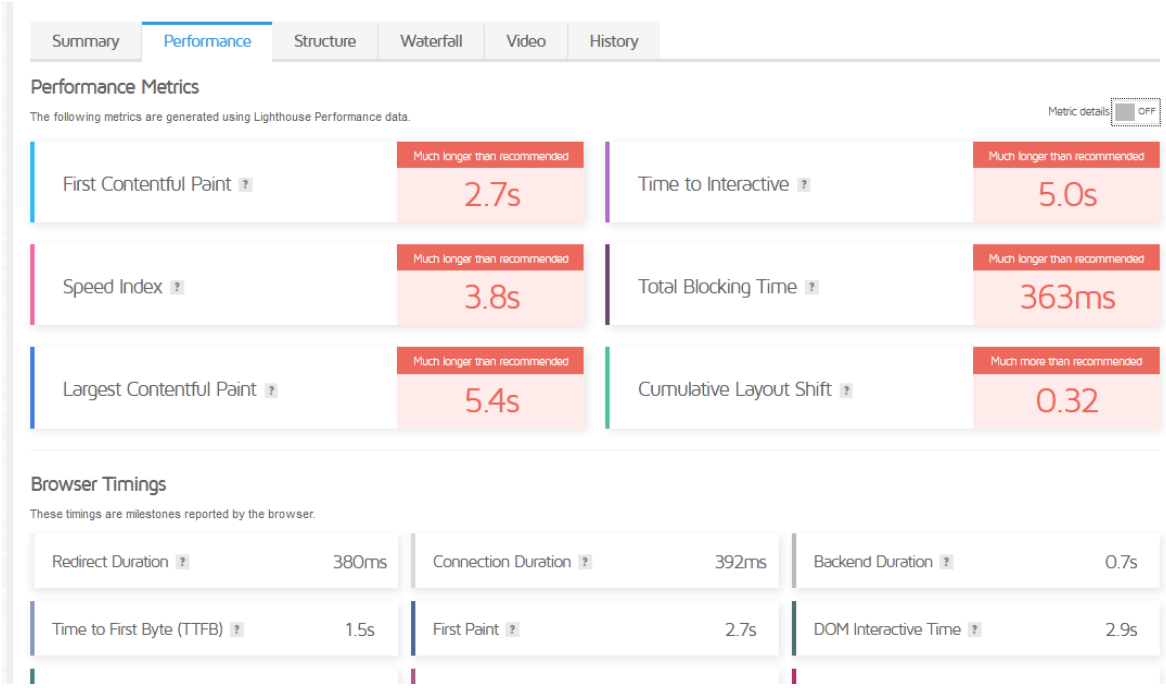


Рисунок 2.3 - Параметр Performance для головної сторінки nure.ua

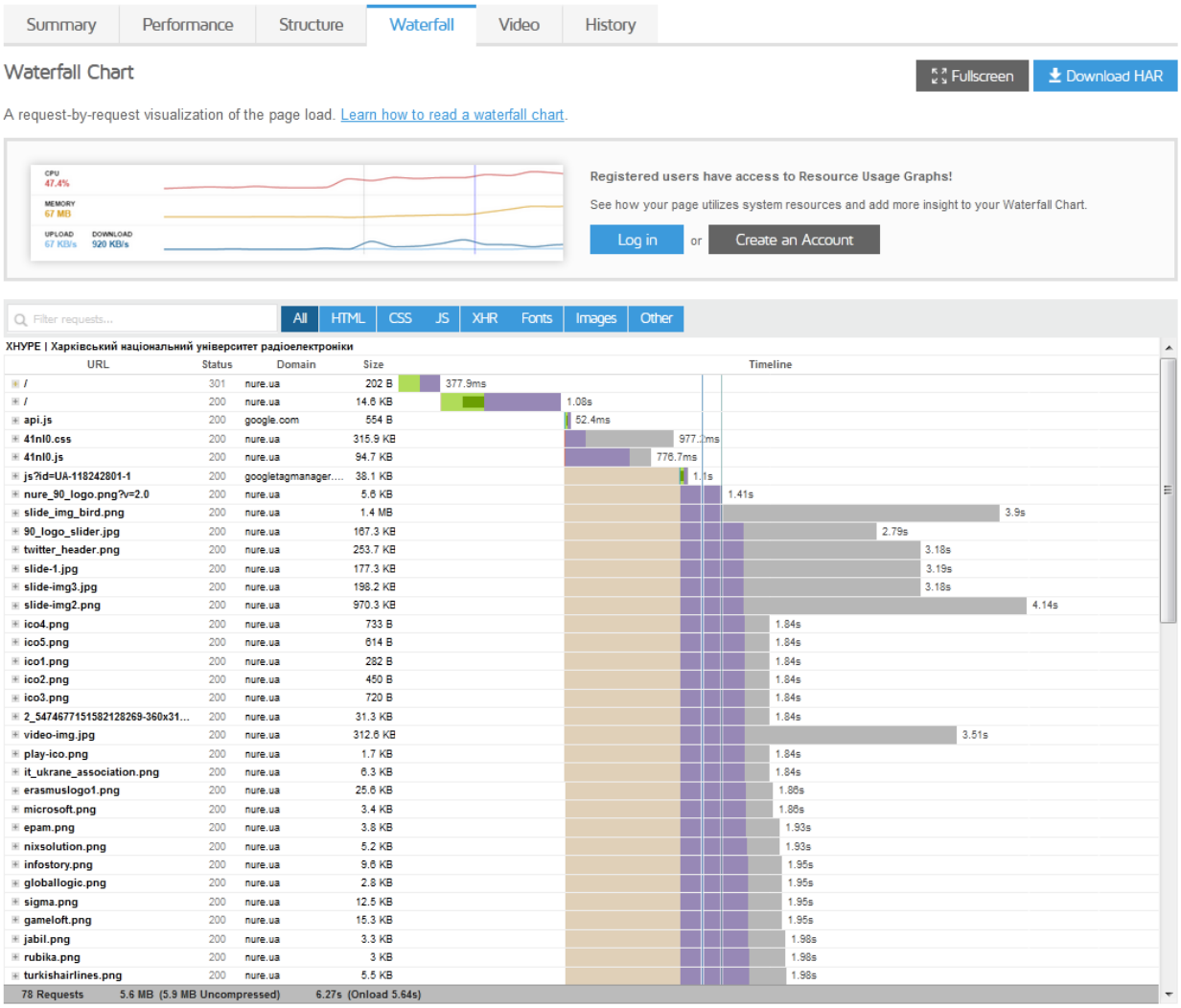


Рисунок 2.4 – Результат виконання тесту-водоспаду, отриманий для головної сторінки сайту nure.ua

На базі тесту-водоспаду GTMetrix забезпечується можливість гнучкого дослідження контенту веб-документу, зокрема, поелементно, чи категорійно, тим самим дозволяючи аналізувати кожну структурну одиницю у часовому просторі.

При цьому, уся множина складових сторінки досліджується також під час аналізу структури сформованого документу на предмет оцінки ступеню його впливу на підсумковий час опрацювання клієнтського запиту з наступним завантаження документу у браузер.

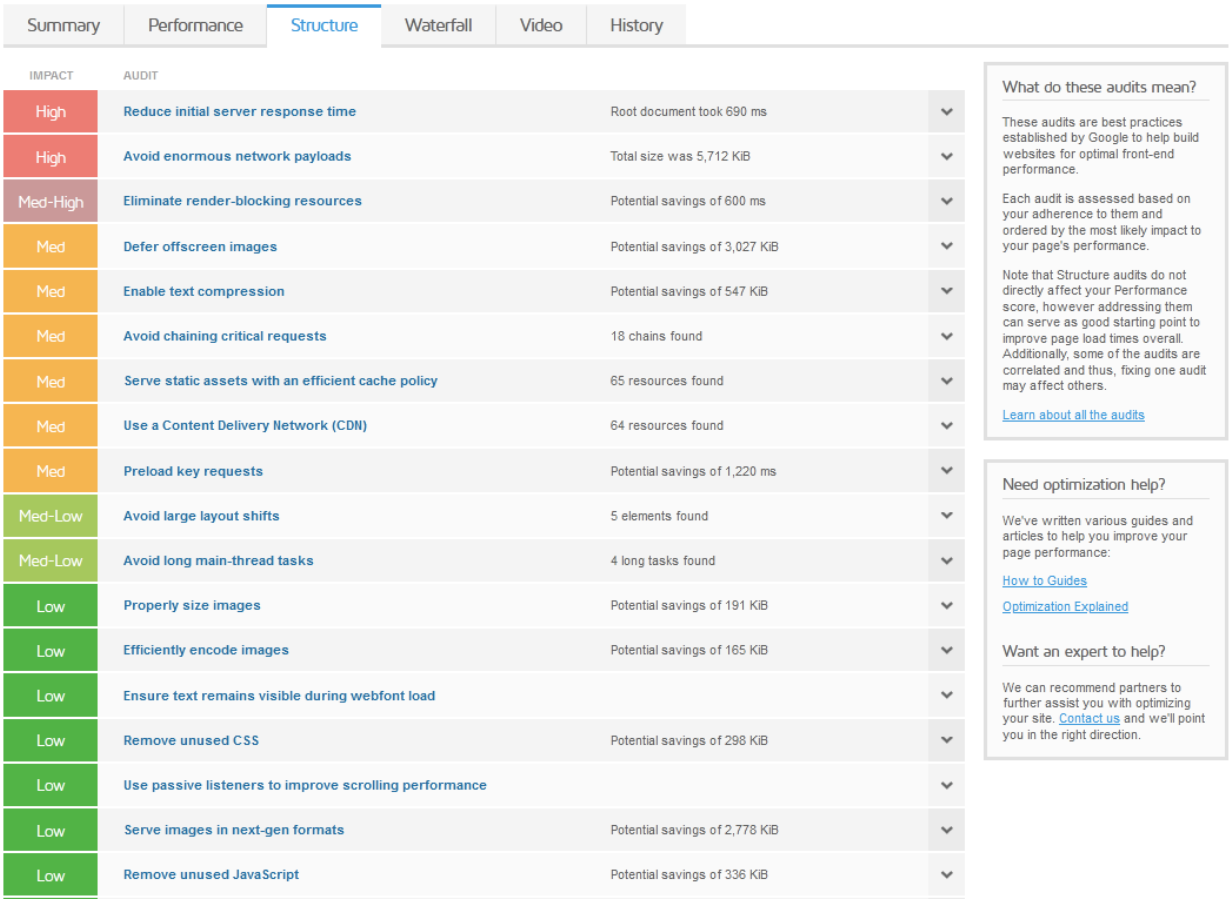


Рисунок 2.5 – Фрагмент дослідження структури сторінки головної сторінки сайту у ході тесту структури

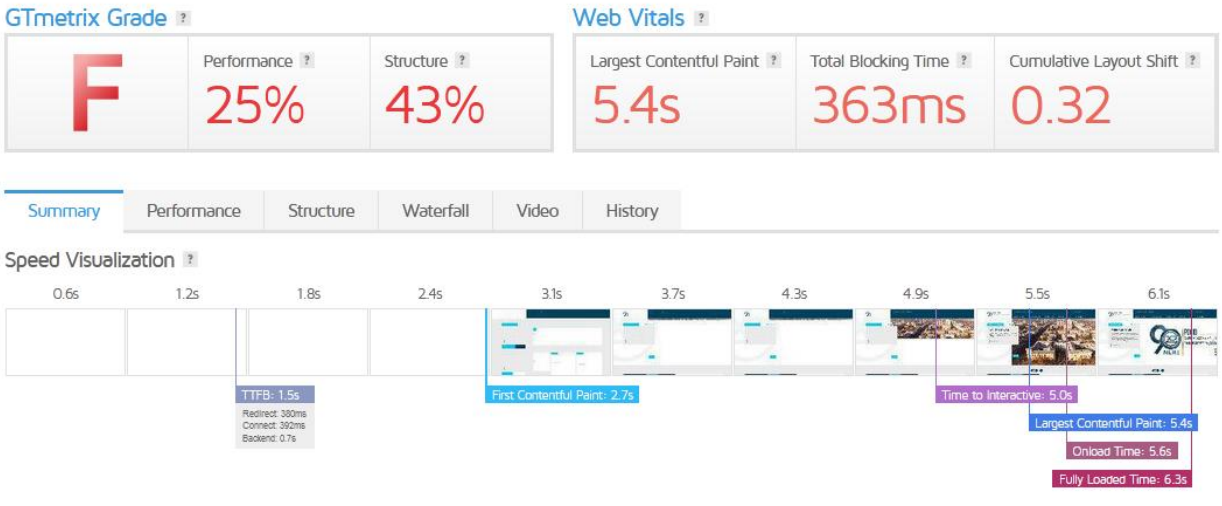


Рисунок 2.6 – Підсумкова узагальнена оцінка раціональності побудови головної сторінки сайту nure.ua

2.3 Обгрунтування необхідності технічного аудиту контенту веб-вузла

За підсумками дослідження сутності та складу заходів з технічного аудиту веб-вузла на різних етапах його життєвого циклу, можемо зробити ряд попередніх висновків:

1. Завданням технічного аудиту є виявлення недоліків апаратної та/або програмної конфігурації веб-ресурсу з подальшою розробкою низки заходів з їх усунення;

2. Недоліки у виборі апаратних або програмних складових сайту та/або їхнє нераціональне налаштування веде до зниження технічної продуктивності W_{tech} ресурсу, що, у підсумку, спричинює падіння показника W_{func}.

Таке падіння величини W_{tech} технічної продуктивності може проявлятися як на низькому (надмірне навантаження на систему CPU, резервування невиправдано високого відсотку RAM), так і на високому рівні (падіння швидкості опрацювання SQL-запитів та функціонування php-обробників, зростання часу відгуку плагінів тощо).

Ймовірною причиною зниження показника W_{tech}, та, у підсумку, фінальної величини W_{func}, можуть бути наслідки зловмисного впливу на систему. При цьому, ту чи іншу частину обчислювальних ресурсів може отримати у розпорядження зловмисний агент. У зазначеній ситуації веб-ресурс зазнає впливу ризиків 2 рівнів, а саме:

- ***технологічні ризики***, що пов'язані з суттєвим зменшенням значень величин W_{tech} та W_{func}, що зумовлюються відбором деякого відсотку ресурсів системи зловмисним агентом [18, 19];

- ***репутаційні ризики,*** зумовлені тим, що інфікований веб-вузол надалі може використовуватися для подальшого розповсюдження зловмисного коду; відтак ресурс може бути суттєво знижено у рейтингу, або повністю блоковано пошуковою системою, що також спричинить суттєве падіння W_{func}.

Разом з тим, на сьогодні існує велика кількість серверного ПЗ та онлайн-сервісів, що дозволяють виявляти спрямовані, або хаотичні кібератаки. Утім, їхня ефективність є низькою на випадок, коли мова йде про ***розвинуті стійкі загрози*** [20, 21], у першу чергу, такі, де однією зі

складових є механізми інкапсуляції (вбудовування) даних у різнотипні інформаційні носії з використанням стеганографічних методів.

2.3.1 Сутність механізмів прихованої доставки зловмисного коду на базі стеганографічних підходів

На сьогодні найвищу небезпеку в інформаційному середовищі несуть багаторівневі довгострокові кібератаки, або ATP (advanced percistent threat – розвинуті стійкі загрози).

Класичні методи кіберзахисту на сьогодні не здатні ефективно протистояти зловмисним впливам даного типу, так як ATP готуються з урахуванням специфіки наявних апаратно-програмних засобів об'єктів атаки, та задіяних засобів захисту. При цьому, зазвичай використовуються унікальні зловмисні модулі та засоби їх доставки до цільових об'єктів. Таким об'єктом, у свою чергу, нерідко є веб-вузол, що дозволяє зловмисникам отримати:

- доступ до клієнтських баз та корпоративного листування;

- доступ до корпоративної мережі підприємства шляхом інфікування ПК адміністраторів сайту;

- доступ до веб-аналітики ресурсу, direct-програм тощо.

Зазвичай APT має у своєму складі такі ключові етапи, як (рис.2.7) показано далі [20-23]:

- збір інформації щодо об'єкту атаки у пасивному режимі;

- перша стадія інфікування (спрямування адміністратору ресурсу на фішингові сайти, надсилання інфікованих документів);

- завантаження на атакований вузол функціональних зловмисних модулів (drive-by, експлуатація недолік захисту на рівні клієнтського браузеру та його розширень);

- проникнення до системи (тут зловмисник виконує підвищення привілеїв свого поточного аканту, здійснює обхід стандартних модулів захисту, виконує збір детальної інформації про систему та інтегрує головні зловмисні компоненти);

- повний (частковий) контроль атакованої системи (завантаження backdoors, кейлогерів тощо);

- обмін даними з віддаленими вузлами керування зловмисника (заходи з обходуд firewalls, використання для реалізації своїх потреб мережевих API, клієнтів соцмереж та месенджерів);

- виконання наміченого завдання (знищення та/або крадіжка інформації, здійснення нелегальних фінансових операцій, розгортання ботнетів тощо).

Рисунок 2.7 – Типові тапи реалізації APT

Методи стеганографії (маскування даних) дозволяють зловмиснику протидіяти засобам кіберзахисту як у ході перша стадія інфікування, так і під час завантажння компонент зловмисниого ПЗ до атакованої системи.

Після цього на базі методів стеганографії формується прихований каналу, що слугує для обміну даними з віддаленими центрами керування APT. Один з виявлених сьогодні сценаріїв завантаження зловмисного ПЗ з використанням маскування даних передбачає виконання таких технологічних кроків, як (рис. 2.8):

1. Доставка файлу-конвертору на атакований вузол. Такий файл не розпізнається системами кіберзахисту, так як не містить типових небезпечних ознак. Доставка реалізується з використанням фішингу, E-mail тощо.

2. Завантаження конвертором файлів, як містять маскований зловмисний контент у неявному вигляді. Дані файли не викликають підозр ззовні (зазвичай це файли невисокої роздільної спроможності - gif, jpeg або bmp) та не розпізнаються засобами кібербезпеки (з причин відсутності типових ознак зловмисного контенту).

3. Обробка файлів, як містять маскований зловмисний контент, конверторами. Така обробка здійснюється за певним алгоритмом, після чого з цих файлів виокремлюється масковане зловмисне навантаження.

4. Побудова функціональних зловмисних модулів з використанням вилученої маскованої інформації.

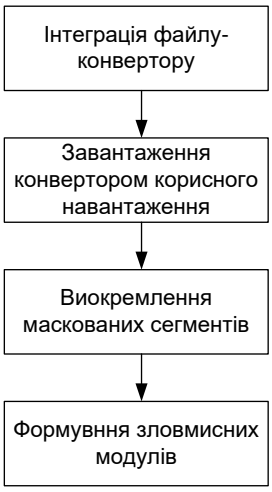


Рисунок 2.8 – Один з можливих сценарій застосування стеганографійних методів у складі APT

Подібним чином на сьогодні реалізовано перелік поширених засобів кібершпигунства, серед яких:

- six little monkeys;
- NetTraveler;
- Zberp;
- Enfal;
- Shamoon;
- KinS;
- ZeusVM;
- Fibbit та ін.

Отже, на цей випадок комплекс заходів з технічного аудиту має бути розширено включенням механізмів стегоаналізу для виявлення ймовірного прихованого зловмисного контенту.

3. ЗАБЕЗПЕЧЕННЯ ФУНКЦІОНАЛЬНОСТІ ВЕБ-РЕСУРСІВ ЗА РАХУНОК ВИЯВЛЕННЯ ЗЛОВМИСНОГО ЗМІСТУ У СКЛАДІ КОНТЕНТУ

3.1 Загальний сценарій вконання заходів з аналізу контенту, надходячого до веб-ресурсу

Сутність аналізу контенту, що може прийматися веб-ресурсом, зводиться до виконання операцій виявлення зловмисного вмісту, та включає у себе наступні кроки [19]:

1. Застосування методів кіберзахисту на базі:

- сигнатурних методів;
- евристичних методів.

2. Затосування методів глибокого аналізу контенту.

Такі методи спрямовані на виявлення прихованих потенційно зловмисних конструкцій, що не можуть бути розпізнаними на передуючому кроці у наслідок того, що фрагменти потенційно небезпечного коду може бути подано у неявному вигляді, зокрема:

- зловмисний контент може бути попередньо шифровано;
- трансляція небезпечного змісту може здійснюватися за декілька кроків з використанням агентів-збірників (на боці атґкованого вузла), як це має місце у випадку APT;
- зловмисник розміщує небезпечний контент з урахуванням особливостей даних, що використовуються у якості контейнеру.

За таких умов інструментарієм глибокого дослідження контенту можуть виступати методи стегоаналізу.

При цьому, їхнє застосування зводиться до виконання скану певної частини, або усього масиву даних, що надходять до веб-вузла.

У результаті цього може бути встановлено факт наявності, або відсутності прихованого змісту.

У випадку виявлення такого контенту, відповідно до засад стагоаналізу, робиться висновок про те, що отримані дані є стегоконтейнером.

Загальна схема аналізу прийнятого контенту може бути подана рисунком 3.1.

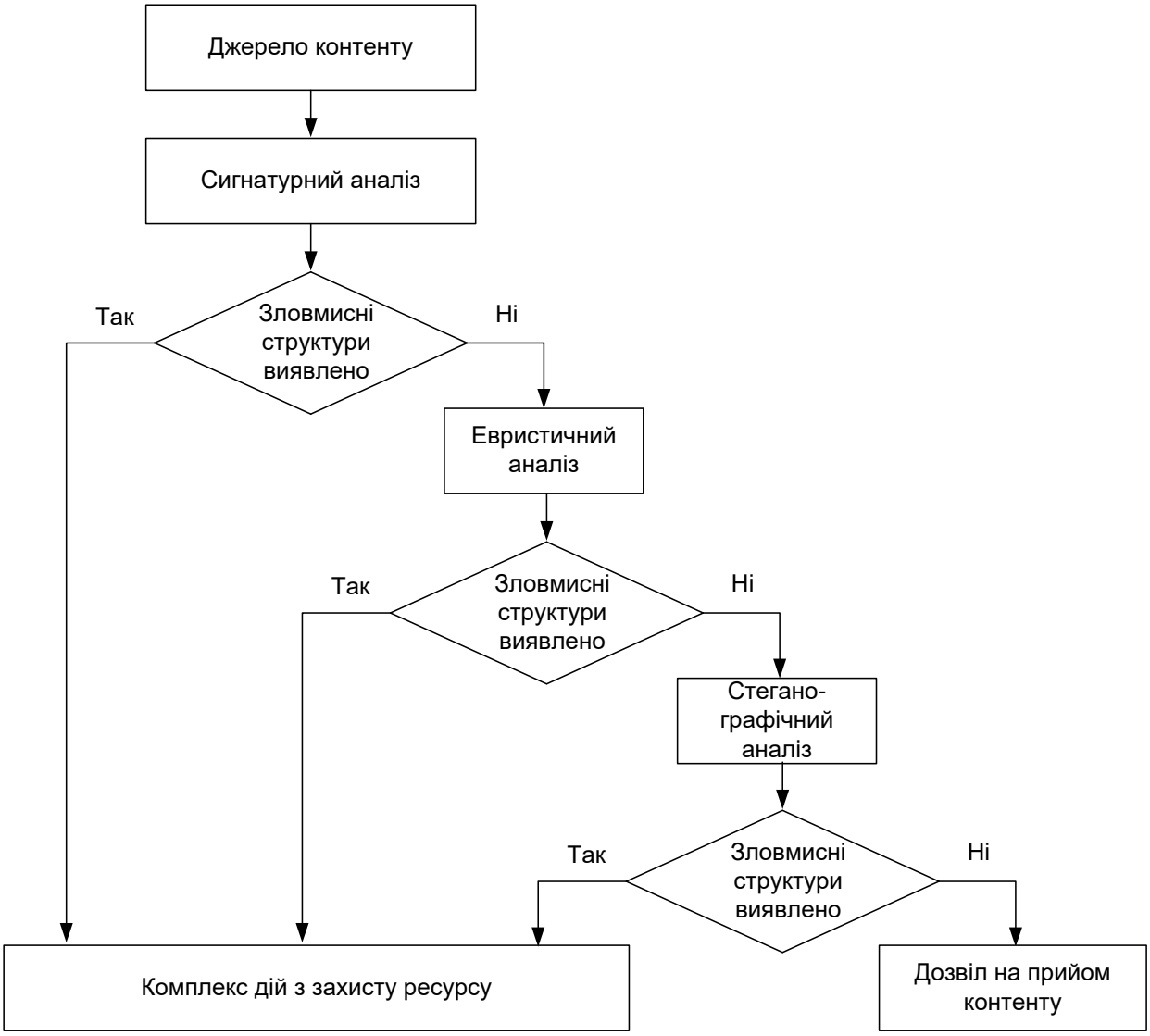


Рисунок 3.1 - Загальна схема аналізу прийнятого контенту з використанням інструментів кіберзахисту та стегоаналізу

Отже, з аналізу рисунку 3.1 можемо бачити, що методи глибокого аналізу (на основі алгоритмів стегоаналізу) застосовуються як або допоміжні інструменти глибокого аналізу контенту для виявлення потенційно небезпечних структур, так і у якості самостійних засобів.

У свою чергу, стегоаналіз контенту виконується відносно ***потенційно небезпечних класів даних.***

Такими зазвичай вважаються дані, які зловмисник може застосувати для побудови стеганографічних контейнерів [24].

При цьому, дані потенційно небезпечних класів можуть надходити до вузла ***потенційно небезпечними каналами***.

На сьогодні найчастіше зловмисник застосовує текстові, графічні та аудіо контейнери. Безвідносно до того, на базі чого зловмисником утворено стегоконтейнер, процес виконання стегоаналізу отримуваного контенту пропонується виконується відповідно до сценарію, зображеного рис. 3.2.

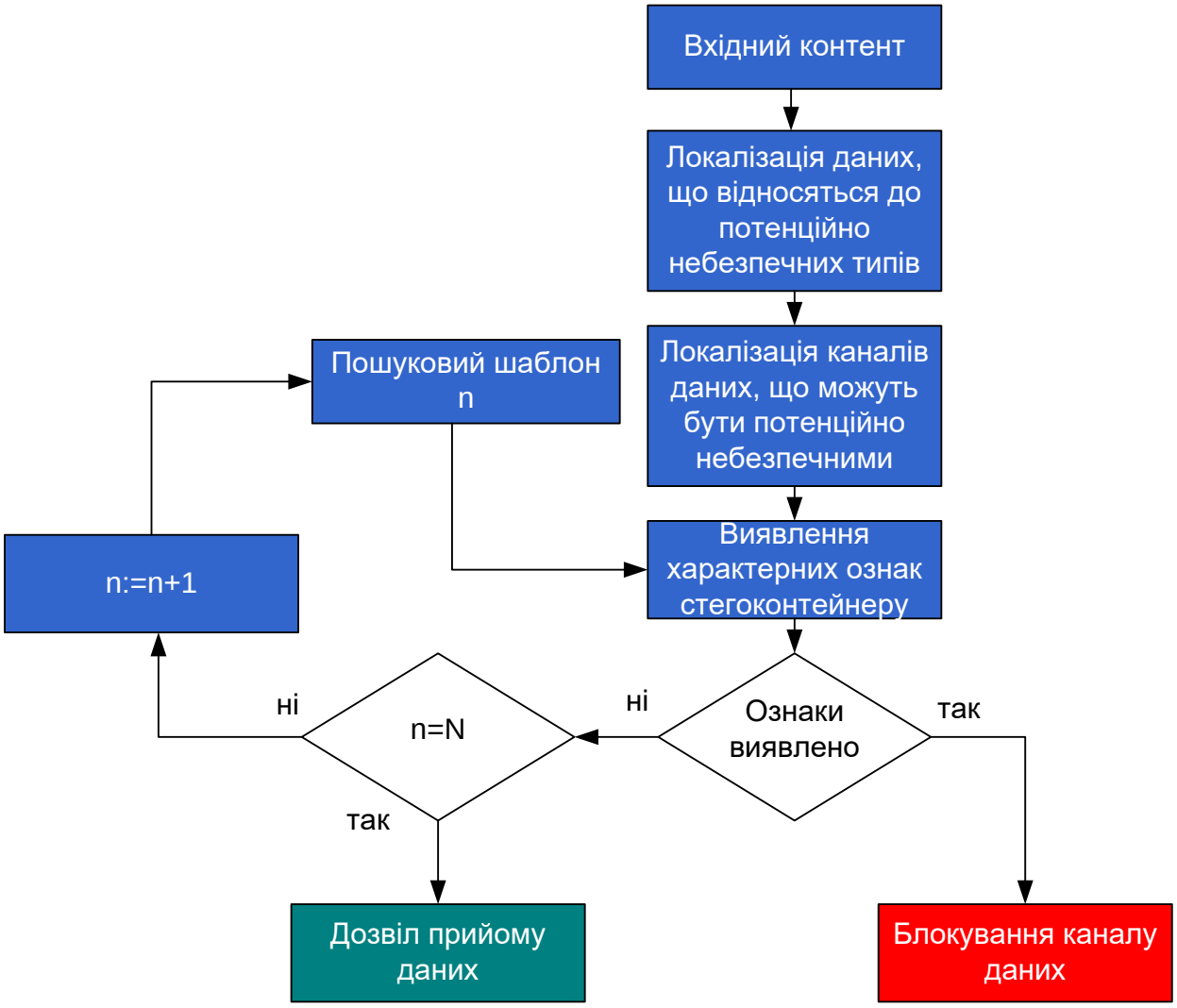


Рисунок 3.2 – Загальний сценарій виконання процедури стеганографічного аналізу контенту

Далі розглянемо загальний принцип функціонування алгоритму стегоаналізу без прив'язки до тих чи інших типів контейнерів, що зображений на рис.3.1.

У наведеному загальному сценарію стегоаналізу потік даних, які формують вхідний контент, інтерпретується як сукупність Ψ_{pack} окремих інформаційних пакетів.

Такі пакети, при цьому, формуються даними, які відносяться до різних класів [24].

При цьому, алгоритми стегоаналізу безпосередньо взаємодіють з обмеженими у часі відрізками $\Psi(t)_{pack}$ потоку даних у діапазоні часу $t \in [-\infty;+\infty]$.

У підсумку, справедливим є твердження про те, що у будь-який момент часу t, який належить зазначеному вище діапазону, формальне представлення довільного відрізку $\Psi(t)_{pack}$ інформаційного потоку може бути здійснено у наступний спосіб:

$$\Psi(t)_{pack} = \{p(t)\} = \\ = \{p(t)_{td}\} \bigcup \{p(t)_{sd}\} \bigcup \{p(t)_{gd}\} \bigcup \{p(t)_{vd}\} \bigcup \{p(t)_{ef}\} \bigcup \{p(t)_{lf}\} \bigcup \{p(t)_{etc}\}, \quad (3.1)$$

де $\{p(t)_{td}\}$ - загальна сукупність інформаційних пакетів, які містять фрагменти (або файли повністю) файлів текстового характеру, наприклад, гіпертекст (htm) та супунті типи (css, js тощо), форматовані та неформатовані файли офісних пакетів MS Officet, Libre Office, Open office (txt, doc, rtf, odt) а також інші з даної категорії;

$\{p(t)_{sd}\}$ - сукупність інформаційних пакетів, що містять у собі аудіо інформацію (stream-аудіо та аудіо AoD у форматі MP3, звукові дані VoIP, аудіосупровід відео та ін);

$\{p(t)_{gd}\}$ - пакети даних графічного характеру, які відповідають растровим файлам (jpeg, heic, pic, png, webp, bmp, gif, pcx тощо) та файлам, що містять вектори (ai, cdr, psd, pdf та ін);

$\{p(t)_{vd}\}$ - сукупність пакетів, які відповідають даним відео інформаційного типу. Сюди, зокрема, відносяться пакети потокових відеоінформаційних сервісів, включаючи VoD, також пакети сервісів VR та AR, інтерактивне відео (відеоконференції та відеозв'язок, відеоспостереження реального часу); зазвичай це відповідає трансляції таких типів файлів, як mp4, webm, avi, mpg, flv та ін. Майже усі такі пакети у своїй основі мають кодування на засадах mpeg. Водночас, деяка частка файлів утворена з використанням альтернативних кодеків, таких, як fractal core, mjpeg або mjpeg-2000;

$\{p(t)_{ef}\}$ - сукупність файлів-додатків, іншими словами - виконуваних файлів (com, exe, bat);

$\{p(t)_{lf}\}$ - множина файлів-бібліотек, що найчастіше зараз представлена такими типами, як dll та lib. Дані бібліотеки застосовуються як локальні ресурси, спрямовані на забезпечення та підтримку функціональності виконуваних файлів;

$\{p(t)_{etc}\}$ - сукупність пакетів файлів усіх незазначених у даному переліку типів, з урахуванням як відомих широко, так і специфічних, зокрема, таких, які застосовуються у межах певних проектів.

Беручи до уваги вираз (3.1), можемо стверджувати, що зараз типова структура довільного відрізку $\Psi(t)_{pack}$ інформаційного потоку, який відповідає деякому моменту часу t є такою, як демонструє схема на рисунку 3.3 [25].

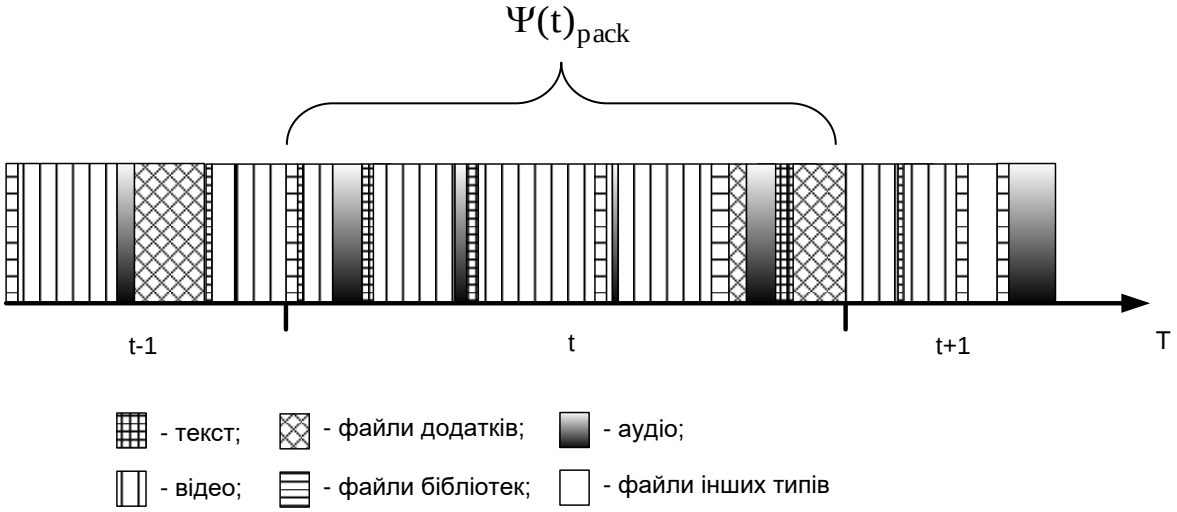


Рисунок 3.3 – Структурна схема довільного відрізку $\Psi(t)_{pack}$ потоку даних у деякий момент часу t

При цьому, у ході виконання першого кроцу алгоритму стегоаналізу здійснюється прийом та знаступною інтерпретацією даних, що містяться у межах відрізку $\Psi(t)_{pack}$ потоку даних.

Після інтерпретації отриманих даних, на наступному кроці функціонування алгоритму, серед наявної сукупності різнорідних інформаційних пакетів визначаються ті з них, на базі яких зловмисник має потенційну можливість побудови стеганографічних контейнерів.

Разом з тим, тут злід зазначити, що зазвичай стегоконтейнери може бути побудовано на базі лімітованого переліку типів файлів. Відтак, це стосуються також і пакетів, що містять такі файли.

Виходячи з такої залежності, розглянемо більш детально найбільш поширені типи файлів з веб-середовища для того, щоб визначити, які з них є потенційно прийнятним для побудови зловмисником стеганографічних контейнерів.

3.1.1 Аналіз можливості застосування файлів різних типів для реалізації стеганографічних контейнерів

Застосування файлів того чи іншого типу як стегоконтейнеру можливо тоді, коли виконується певний перелік вимог, а саме [24-26]:

1. Значна розповсюдженість файлів даного типу у Всесвітній мережі, intranet-сегментах та на веб-вузлах.
2. Суттєвий рівень надмірності γ опису даних, притаманний файлам даного типу. Це умова гарантує ефективність стегоконтейнеру з позиції потенційної захищеності від ймовірного викриття. Таку залежність може бути проілюстовано за допомогою системи нерівностей:

$$\begin{cases} \eta\uparrow \rightarrow P\uparrow; \\ \eta\uparrow \rightarrow D\uparrow, \end{cases} \tag{3.2}$$

де P - умовний ступінь захищеності вбудованих у контейнер даних; цей показник є одним з ключових для будь-якого алгоритму стеганографії незалежно від його архітектури та особливостей реалізації;

D - загальний обсяг інформації, яку може бути інкапсульовано у стеганографічний контейнер.

Разом з тим, показники P та D, пов’язані між собою залежністю, як показано далі:

$$P \sim \frac{1}{D}, \tag{3.3}$$

тобто, між означеними показниками існує обернена пропорційність. Відтак, в умовах росту обсягу D інформації, яка підлягає інкапсуляції, ступінь її захищеності буде суттєво знижуватися.

3. Функціональність файлу, що реалізує роль контейнеру стеганографічної системи, після інкапсулювання даних повиненна залишатися незмінною.

4. Трансляція файлу, що являє собою стегоконтейнер, до веб-вузлу, не повинна викликати підозри. Тобто, це має бути файл такого типу, який є властивим для даного середовища.

Порівняльний аналіз ряду файлів, що відносяться до різних типів , які було зазначено раніше, за критерієм розповсюдженості показано у таблиці 3.1

Таблиця 1 – Ступінь розповсюдженості фалів різних типів, які потенційно може бути використано у якості контейнерів

Тип файлу	**Розповсюдженість**
Текстові файли, гіпертекст	Висока. Файли присутні у складі усіх файлосховищ на всіх кінцевих вузлах
Файли додатків	Обмежена. Завантаження файлів даних типів з мережевих джерел обмежується або блокується рядом політик безпеки
Файли бібліотек	Обмежена. Пояснюється тим, що даний тип є супутній файлам додатків і окремо завантажується з мережі виключно у ході оновлення ПЗ або за необхідності розширення його функціоналу.
Файли зображень	Висока. Файли присутні у складі усіх файлосховищ на всіх кінцевих вузлах
Аудіо	Висока. Проте гарантовано файли можуть бути присутні у складі файлосховищ та на кінцевих вузлах за умови наявності ПЗ та апаратних пристроїв обробки та відтворення даних файлів
Відео	Висока. Проте гарантовано файли можуть бути присутні у складі файлосховищ та на кінцевих вузлах за умови наявності ПЗ та апаратних пристроїв обробки та відтворення даних файлів
Файли інших типів	Не гарантується. Разом з тим, деякі файли (наприклад, архіви) можуть бути достатньо розповсюдженими

Таким чином, судячи з даних, приведених у табл. 3.1, на сьогодні найвищій рівень розповсюдженості відповідає графічним та текстовим файлам.

Водночас, разом з файлами означених типів як у мережі, так і на рівні веб-вузлів суттєво розповсюджені відео файли та файли аудіо.

Утім, це гарантується виключно за тих умов, коли регламент роботи мережі та вузлів у її складі передбачає використання даних файлів.

Разом з тим, виходячи з критичності вимоги щодо суттєвого ступеню надлишковості γ опису файлів-контейнерів, це також є вирішальним чинником при виборі типу файлів для інкапсулювання приховуваного контенту.

У зв'язку з цим, виконаємо порівняльний аналіз файлів раніше згаданих типів типів на предмет їх відповідності означений вимозі щодо надмірності опису.

Ступені надмірності опису γ, характерні для різних типів, представлено у таблиці 3.2.

Таблиця 3.2 – Порівняльний аналіз фалів ряду типів за показником ступеню надлишковості опису

Тип файлу	**Ступінь надлишковості опису**
Текстові файли, гіпертекст	+
Файли додатків	-
Файли бібліотек	-
Файли зображень	+++
Аудіо	++
Відео	++++
Файли інших типів	+/-

Як показує аналіз табл. 3.2, максимальні ступені надлишковості опису притаманні файлам, що містять у собі відеодані, а також графічним та звуковим файлам.

Разом з тим, відносно високий рівень надлишковості представлення характерний для файлів текстового типу, хоча за даним показником він помітно поступається, вищеназваним файлам.

Виходячи з цього, можемо сказати про те, що за показником надмірності опису найбільш ефективними для застосування у якості контейнерів є файли відео, файли зображень та аудіофайли.

Далі порівняємо файли з точки зору можливості збереження функціональності та цілісності за результатами вбудовування приховуваного контенту, як показано табл. 3.3.

Таблиця 3.3 – Дані щодо забезпечення відповідності файлами різних типів вимоги щодо збереження функціональності та цілісності у наслідок виконання інкапсуляції приховуваного контенту

Тип файлу	**Гарантований рівень функціональності та цілісності**
Текстові файли, гіпертекст	Певною мірою зберігається
Файли додатків	-
Файли бібліотек	-
Файли зображень	+++
Аудіо	+++
Відео	+++
Файли інших типів	+/-

Отже, як можна бачити з табл 3.3, серед файлів розглянутих типів у наслідок виконання процедури інкапсуляції приховуваного контенту гарантується збереження функціональності та цілісності для відео, зрувоких та графічних файлів.

Разом з тим, файти додатків та бібліотек у наслідок інкапсулювання даних руйнуються, що не дозволяє їх використовувати у ролі контейнерів. Утім, вбудовувати дані у файли додатків (exe, com та bat-типів), та lib s dll-файли теоретично можливо. На цей випадок деякий обсяг даних може бути вбудовано, наприклад, у зерезервовані службові сегменти файлів означених типів.

Водночас, такий процес вимагає додаткового виконання таких операцій, як:

- аналіз структури службових полів файлу;

- виявлення сегментів службових даних, що є актуальними у поточній структурі;

- підрахунку поточноих контрольних сум та внесення їх величин у відповідні поля після того, як дані вбудовано.

Недоліки даного підходу наступні:

- забезпечується невисокий рівень захищеності даних (службові поля завжди знаходяться у фокусі уваги під час стегоаналізу, при цьому, вбудовані дані довільного зісту можуть розпізнаватися як потенційно зловмисні у ході евристичного аналізу на базі стандартних засобів кіберзахисту);

- підсумкова ємність стегосистеми може бути надмірно низькою;

- швидкість обробки файлів-додатків та бібліотек нижча, ніж на випадок використання контейнерів інших типів.

Таким чином, у наслідок аналізу даних, представлених табл. 3.1-3.3, можемо зазначити те, що за показниками збереження функціональності/цілісності, надлишковості опису та розповсюдженості найбільш прийнятними для використання у ролі стегоконтейнерів є файли відео та аудіо, а також графічні.

Отже, далі будемо розглядати файли, що належать лише зазначеним типам.

Для них виконаємо аналіз їх відповідності четвертій ключовій вимозі - Далі розглянемо файли виключно трьох вказаних типів з точки зору відповідності умові 4, тобто, потенційній відсутності підозр щодо факту трансляції того чи іншого файлу до веб- вузла.

Щоб мати змогу здійснити такий аналіз, слід ураховувати не лише загальну специфіку файлів певного типу, але також конкретні їх особливості на рівні певних форматів.

Відповідно у порівнянні слід задіяти хоча б найбільш поширені формати файлів у межах кожного з розглядуваних типів, як це показує таблиця 3.4 [27].

Таблиця 3.4 – Аналіз файлів за показником відсутності підозри щодо факту їх трансляції до веб-вузла

Тип даних	Формат файлу	Підозра	Примітка
Звук	wav	+++	Розмір файлу у 5-15 разів перевищує розмір mp3 файлу такого ж змісту та тривалості
	ogg	+/-	Даний тип не має досить широкої розповсюдженості у мережі
	Mp3	-	-
Графічні дані	Jpeg, jpg	-	-
	gif	-	Формат опису характеризується низьким рівнем надмірності, що знижує цінність даного типу як контейнеру
	bmp	+++	Розмір файлу у 10-20 разів перевищує розмір jpeg файлу такого ж змісту та візуальної якості
	png	+	Має досить вузький діапазон застосування, що не дозволяє розглядати цей тип як універсальний
Відео	mp4	-	-
	mpg	+++	Розмір файлу до 8-10 разів може перевищувати mp4
	mjpeg	+	Розмір файлу до 5-8 разів може перевищувати mp4
	webm	-	Має відносно обмежений діапазон застосовуваності, що представлений VoD на базі HTTP

Таким чином, як видно з таблиці 3.4, вимозі щодо відсутності підозр відносно факту трансляції файлу до веб-вузла найбільшою мірою

відповідають відеофайли mp4, аудіо, кодоване у базисі mp3 а також графічні файли jpeg.

3.1.2 Ключові підходи щодо реалізації заходів стегоаналізу на основі групування прийнятих файлів за показником ефективності їх використання у ролі контейнеру прихованих даних

У п. 3.1.1 було показано, що максимальний рівень маскування стегоканалу зловмисника досягається тоді, коли у ролі контейнерів прихованого контенту виступає певний перелік типів файлів, а також ряду конкретних форматів файлів у мехаж кожного з типів. При цьому на схемі, поданій рис 3.2, їх названо даними потенційно небезпечних типів [25-27].

Тут слід брати до уваги те, що:

1. Для усього обсягу D даних, що надходять до веб-вузла у часовому діапазоні $[-\infty; +\infty]$, справедливою є залежність [27]:

$$D = \{D_{ps}\} \bigcup \{D_{pd}\}, \qquad (3.4)$$
$$D_{ps}, D_{pd} = \overline{(0; D)},$$

де $\{D_{ps}\}$ та $\{D_{pd}\}$ - множини пакетів безпечних та потенційно небезпечних відповідно.

Інакше кажучи, з виразу (3.4) бачимо, що довільний відрізок $\Psi(t)_{pack}$ потоку даних рівноймовірно може мовністю складатися з потенційно небезпечного наповнення, або формуватися цілком з даних, що понетнціно не несуть небезпеки.

2. На випадок необхідності аналізу надходячого трафіку критичного та інтерактивного типів має виконуватися вимога:

$$t_{proc} \to 0, \qquad (3.5)$$

де t_{proc} - час обробки, у ході якого здійснюється глибокий аналіз змісту пакетів.

При цьому слід брати до уваги, що обчислювальний потенціал Θ апаратних платформ, на базі яких може реалізовуватися алгоритм

стегоаналізу, має певний ліміт, який залежить від ряду чинників, що еквівалентно виразу:

$$\Theta = \varphi(\overline{\Theta}; \{\Xi\}; \Omega), \qquad (3.6)$$

де $\overline{\Theta}$ - номінальний рівень обчислювальної потужності;

Ω - показник, що характеризує особливості оброблюваного трафіку;

$\{\Xi\} = (\xi_1 \& \xi_2 \& ... \& \xi_n)$ - множина з n моделей та шаблонів стегоаналізу.

Звідси виходить, що виконувати аналіз повного обсягу пакетів, що відносяться до відрізку даних $\Psi(t)_{pack}$, є недоцільним.

Виходячи з цього, ***пропонується*** попередньо локалізовувати файли потенційно небезпечних типів у межах довільного відрізку даних $\Psi(t)_{pack}$. Далі у випадку їх виявлення, джерело їх надходження (канал) попередньо визнаються потенційно зловмисним.

За результатами знаходження файлів потенційно небезпечних типів далі виконується виявлення ***типових особливостей контейнерів***. Такі особливості не є сталими, для них властиві суттєві варіації, що є наслідком впливу ряду факторів, серед яких:

- належність файлу-контейнеру до того чи іншого типу;
- особливості формату файлу, що використовується у ролі контейнеру;
- особливості використаного алгоритму інкапсуляції, що ймовірно було використано зловмисником.

У підсумку, після того, як попередньо виявлено понетнійно файли потенційно небезпечних типів, на наступному кроці обробки даних здійснюється знаходження типових ознак контейнеру для певних умов, які при цьому залежать лише від використаного методу інкапсуляції.

Відповідно, відносно файлів усіх інших типів, які не є потенційно небезпечними з причин неприйнятності використання у ролі контейнерів, аналіз не виконується.

Зазначимо, що на сьогодні найбільш широкого застосування набули кілька груп стеганографічних алгоритмів для тих чи інших умов виконання інкапсуляції (ураховуючи належність потенційного контейнеру до конкретного формату та типу). Тобто, за таких умов, має сенс виконувати

глибокий аналіз даних, що є потенційно небезпечними, на предмет характерних ознак використання алгоритмів стеганографії, що належать кожній з поширених груп. Така перевірка за замовчуванням може виконуватися послідовно, або у режимі паралельних обчислень. Оскільки формальні відмінності, окрім необхідності гнучкого управління обчислювальними ресурсами, між двома режимами функціонування процесу стегоаналізу відсутні, надалі будемо розглядами послідовну схему аналізу.

Доцільність саме такого сценарію стегоаналізу даних спричинена тим, що:

1. Ефективність уніфікованих алгоритмів стегоаналізу, тобто, тих, які не залежать від використаних зловмисником алгоритмів маскування, не може гарантуватися на високому рівні. У першу чергу, це стосується випадків, коли стегоконтейнер заповнено незначно (у середньому нижче, ніж на 50%);

2. На стогодні має місце широке розповсюдження значної кількості нестандартизованих реалізацій відомих стеганографічних алгоритмів. Відтак, ймовірність виявлення контейнеру за формальними ознаками, притаманними тому чи іншому методу маскування, у цих умовах також може бути низькою. Виходячи з цього, рішення про присутність контейнеру у межах аналізованого сегменту даних приймається за сукупністю непрямих ознак.

Якщо за результатами аналізу даних знайдено один або декілька контейнерів, сформованих на базі хоча б за одного стеганографічного алгоритму, надалі поточний відрізок $\Psi(t)_{pack}$ інформаційного потоку, а також сам файл, до якого він відноситься разом з файлами, що мають з ним логічний зв'язок, вилучаються. Далі означена сукупність файлів переміщується до т.з. «пісочниці» (захищеного файлосховища) для наступної посиленої перевірки.

Разом з тим, джерело, з якого було трансльовано підозрілі об’єкти, у поточний момент часу маркується як небезпечне. У свою чергу, прийом трафіку від цього джерела блокується повністю [27].

Водночас, хоча, сценарій виявлення типових характеристик стегоконтейнерів у захальних рисах за будь-яких умов залишається незмінним, проте передбачається, що у ході стегоаналізу може бути використано щонайменше 2 стратегії глибокого дослідження контенту на предмет виявлення стегоконтейнерів, а саме:

1. Повна заборона прийому файлів, що визнані підозрілими, та блокування джерел їх надходження без виконання попереднього аналізу.

У рамках даної стратегії передбачається послідовне виконання наступного переліку дій:

- файл, що має одну, або ряд ознак присутності підозрілого контенту, вилучається з загального інформаційного потоку для того, щоб відносно нього було виконано поглиблене дослідження; у ході такого дослідження підозрілі файли тимчасово розміщуються у безпечному файлосховищі;

- здійснюється блокування усієї множини пакетів даних, які транслює до веб-вузла джерело, з якого було прийнято потенційно небезпечний файл;

- за результатами поглибленого аналізу змісту підозрілого файлу приймається рішення щодо доцільності подальшого блокування потенційно небезпечного джерела; при цьому, якщо глибоке дослідження не виявлено присутності ознак стегоконтейнеру, для веб-вузла на деякий час Δt відновлюється можливість приймання пактів даних від цього джерела.

Одночасно з цим виконується у реальному часі продовжує вестись моніторинг як потенційно підозрілих файлів так і файлів, що формально не вважаються підозрілими.

2. Аналіз вмісту файлів, які формально не вважаються підозрілими.

У загальному випадку, частіше за все саме такі файли зловмисником може бути використано у ролі контейнерів, як свідчать дані, наведені таблицею 3.4. При цьому, виконується вилучення пакетів файлів саме таких форматів та типів для деякого довільного відрізку $\Psi(t)_{pack}$ потоку даних. Далі такі пакети перевіряються. Базова схема аналізу даних для цієї стратегії реалізації сценарію стегоаналізу включає у себе такі дії, як:

- прийом відрізку $\Psi(t)_{pack}$ потоку даних до вхідного буферу приймача;
- пошук та виокремлення з відрізку $\Psi(t)_{pack}$ множини пакетів файлів, відносно яких є справедливими вимоги 1-4 для стегоконтейнерів;
- блокування джерела пакетів у випадку виявлення ознак присутності стегоконтейнерів.

Тут використовується загальний підхід, властивий методам стегоаналізу, згідно якого достатньою підставою для блокування джерела даних (каналу) є факт виявлення контейнеру. Тим самим, створюється протидія зловмиснику щодо реалізації сценарію впливу на веб-вузол.

При цьому для того, щоб система зловмисника могла вважатися зламаною, не потрібно отримувати безпосередній доступ до інкапсулюованих даних та виконувати їх розшифрування, так як [28]:

1. Наявність джерела (каналу), що є потенційно небезпечним, факт існування якого при цьому для ймовірного об’єкта атаки (у нашому випадку – веб-вузла) є апріорі невідомим, є тією ознакою, що вказує на високу ймовірність зловмисного впливу. Відтак, якщо таке джерело виявляється, у першу чергу здійснюється блокування прийому пакетів, які йому належать. Після цього може виконуватися аналіз маскованих даних.

2. Виокремлення маскованих даних, виявлених у межах контейнеру, з наступним їх дешифруванням та інтерпретацією, являє собою завдання підвищеної складності, яке не може мати тривіальних рішень. Таким чином, для рішення завдань даного класу, може знадобитися значний резерв обчислювальною потужності та часу. Зрозуміло, що у рамках наявного сьогодні технологічного базису такі завдання, у сутності неможливо вирішити у реальному масштабі часу.

При цьому слід зазначити, що безпосередньо процедура стеганографічного аналізу змісту файлів потребує використання значного обсягу обчислювальних ресурсів як наслідок того, що:

- збільшення інтенсивності трафіку у мережі, а також зміна характеристик його структури, що спостерігається на постійній основі, суттєвим чином утруднює процеси зчитування фрагментів даних з загального інформаційного потоку; це саме є також справедливим для процесів компонування файлів зі зчитаних фрагментів;
- протягом суттєво лімітованого часу τ необхідно виконати аналіз значного обсягу D даних.

Таким чином, можемо говорити про те, що зараз існує суттєве протиріччя між, з одного боку – постійно зростаючими об’ємами інформації в умовах ускладення їъ структурних характеристик а з о іншого боку – існуючим лімітом продуктивності існуючих методів стеганографічного аналізу даних. При цьому також слід згадати про існуючу обмеженість обчислювальних ресурсів апаратного забезпечення (рис.3.4).

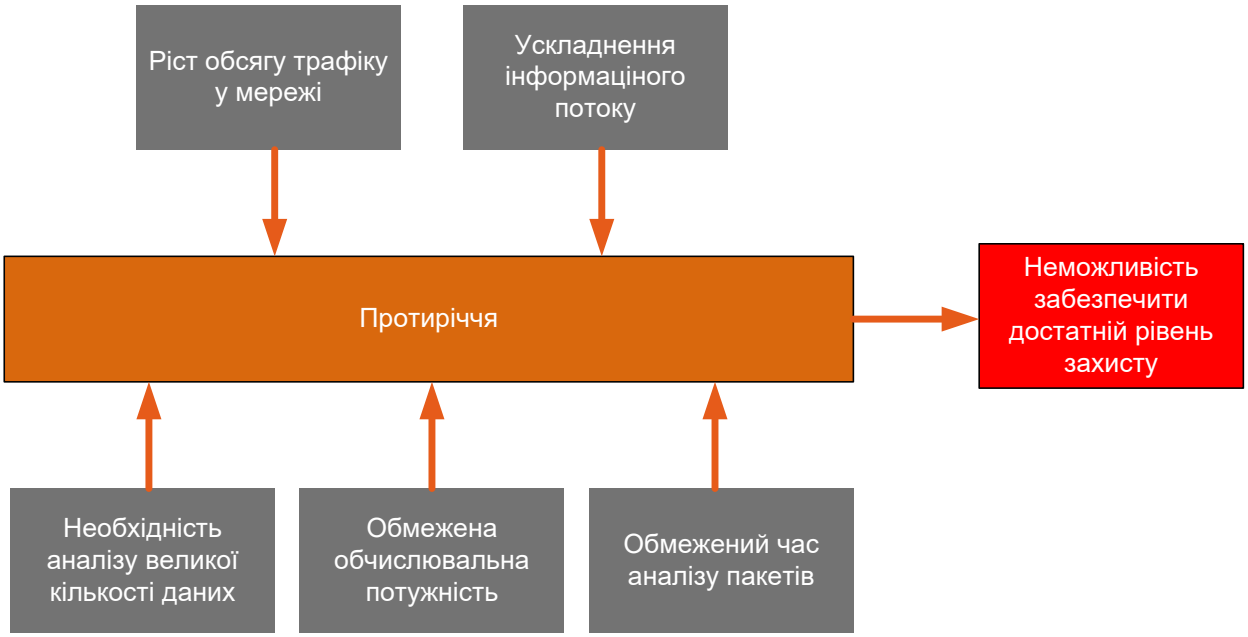


Рисунок 3.4 – Існуюче протиріччя, яке сьогодні має місце у ході захисту веб-вузлів від зловмисних впливів за рахунок виявлення маскованих джерел

Тобто, побудувати зараз ефективну систему стегоаналізу, орієнтовану на моніторинг повного обсягу інформаційних пакетів, що надходять до веб-вузла, беручи до уваги існуючі протиріччя та обмеження фактично неможливо.

Звідси виходить, що у зазначених умовах доцільною для використання є диференційна схема реалізації процесу стегоаналізу.

3.2 Диференційна схема реалізації процесу стегоаналізу інформаційних пакетів, що надходять до веб-вузла

Протиріччя, наведені рис. 3.3, на сьогодні не дозволяють ефективно утілювати системи стегоаналізу для моніторингу усього вхідного трафіку веб-вузла. Для подолання означених протиріч необхідно побудувати такі умови, коли:

- створюється можливість ефективного виявлення маскованих зловмисних каналів;
- аналіз підозрілих інформаційних пакетів проводиться з мінімальними часовими затримками згідно виразу (3.5), тобто, фактично виконується у реальному масштабі часу;

- обчислювальне навантаження, яке створюється у ході аналізу потенційно небезпечних даних, мінімізується.

Для забезпечення таких умов пропонується до використання диференційна схема реалізації процесу стегоаналізу.

Означена схема має ряд ключових відмінностей відносно класичних реалізацій систем стегоаналізу, серед яких наступні:

1. Глибоке дослідження виконується відносно окремих фрагментів даних, що можуть бути потенційно небезпечними; для цього використовується ряд механізмів вибірки.

2. Передбачається наявність двох кроків виконання аналізу даних, зокрема:

- дослідження фрагментів даних, які є потенційно небезпечними;
- дослідження фрагментів даних, які розглядаються, як підозрілі (у випадку, якщо такі фрагменти виявлено).

Одним з базових механізмів, що забезпечують реалізацію диференційної схеми стегоаналізу, є ***механізм формування вибірки даних*** у загальному інформаційному потоці**.**

Далі розглянемо, яким саме чином виконується формування вибірок.

Припустимо, що за деякий довільний проміжок часу $T = \bigcup_{j=1}^{J} t_j$ вхідний буферний пристрій приймача отримує деяку кількість J відрізку $\Psi(t)_{pack}$ загального потоку даних. При цьому, кожен з відрізків може рівно ймовірно містити у собі ту чи іншу кількість пакетів, що відповідають файлам практично будь-яких типів, як це показує рисунок 3.5.

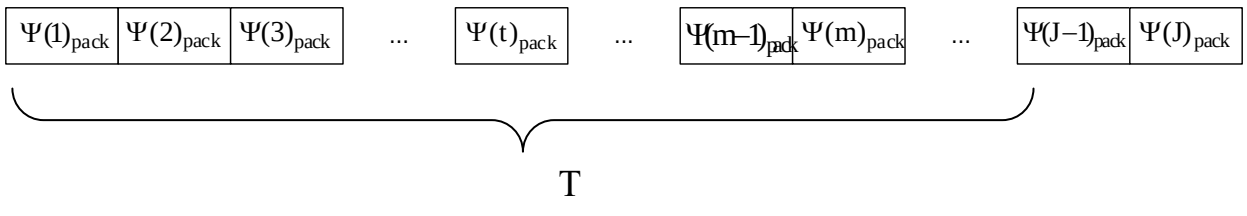


Рисунок 3.5 – Опис інформаційної послідовності, яка знаходиться у вхідному буферному пристрої приймача у вигляді множини J відрізків $\Psi(t)_{pack}$ потоку даних

Разом з тим, аналіз усіх J відрізків $\Psi(t)_{pack}$ потоку даних не виконується. Натомість у межах часового відрізку T досліджується лише певний обмежений обсяг Λ з усіх присутніх J відрізків $\Psi(t)_{pack}$.

При цьому, найпростіший у реалізації підхід до побудови множини Λ зводиться до збору ідентифікаторів t відрізків, які надалі буде досліджено, використовуючи для цього наведений нижче алгоритм, що включає у себе таку послідовність кроків:

1. Обчислення величини $\ell(\Lambda)$ як $\ell(\Lambda) = \text{rand}\{J\}$.

2. Формування сукупності $\{\Lambda\}$ ідентифікаторів t, керуючись при цьому виразом:

$$\{\Lambda\} = \bigcup_{i=1}^{\ell(\Lambda)} \text{rand}\,\Lambda_i \qquad (3.7)$$

3. Дослідження змісту відрізків даних, що утворюють множину $\{\Lambda\}$.

Безумовною перевагою такого способу формування сукупності $\{\Lambda\}$ досліджуваних відрізків $\Psi(t)_{pack}$ даних є його тривіальність та простота реалізації, а також низький рівень обчислювального навантаження. Разом з тим, розглянутому способу властивий ряд недоліків.

Перший недолік полягає у тому, що такий спосіб вибірки не забезпечує однакову ймовірність доступу до будь-якого відрізку $\Psi(t)_{pack}$ за період часу T. Це зумовлюється тим, що за підсумками обчислення розмірності $\ell(\Lambda)$, та знаходження безпосередньо самих індексів, існує висока ймовірність того, що:

- значення розмірності $\ell(\Theta)$ буде містити у собі незначну кількість елементів, що еквівалентно наступній нерівності:

$$\ell(\Theta) < \ell'(\Theta), \qquad (3.8)$$

де $\ell'(\Theta)$ - критично низьке значення розмірності множини $\{\Lambda\}$ ідентифікаторів, за якого ймовірність доступу до будь-якого відрізку $\Psi(t)_{pack}$ за період часу T не гарантується. Величина $\ell'(\Theta)$, у свою чергу, визначається виходячи з параметрів розмірності $\ell(\Psi(t)_{pack})$ відрізку $\Psi(t)_{pack}$ даних,

тривалості часового інтервалу T та показника Ω, який характеризує особливості оброблюваного трафіку, тобто:

$$\ell'(\Theta) = f(\ell(\Psi(t)_{pack}); T; \Omega); \quad (3.9)$$

- вибірка даних на базі простої рандомізації не дозволяє досягти рівноймовірного розподілу значень ідентифікаторів t у межах множини $\{\Lambda\}$. Як наслідок цього – можливість доступу та наступного аналізу змісту відрізків $\Psi(t)_{pack}$, які локалізуються у довільних частинах T фактично відсутня, як ілюструє рисунок 3.6.

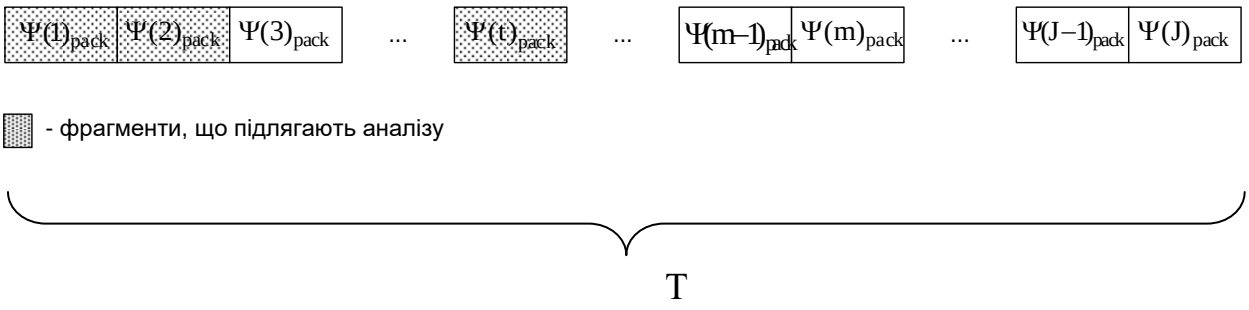


Рисунок 3.6 – Сукупність $\{\Lambda\}$ відрізків $\Psi(t)_{pack}$ у межах часового інтервалу T, яку сформовано на базі простої рандомізації

Як показує аналіз рисунку 3.5, досягнення необхідної рівноймовірності доступу до усіх відрізків $\Psi(t)_{pack}$ потоку даних, що підлягатимуть подальшому дослідженню на базі алгоритмів стегоаналізу, не гарантується .

Отже, можемо констатувати, що при такому способі реалізації механізму вибірки даних ефективність заходів стегоаналізу може бути досить низькою.

Водночас для того, щоб досягти можливості доступу до будь-яких відрізків з множини $\{\Lambda\}$ з приблизно однаковою ймовірністю, загальний обсяг J відрізків $\Psi(t)_{pack}$ даних на ділянці часу T, механізм здійснення вибірки даних ***пропонується*** реалізувати у наступний спосіб.

По-перше, ідентифікатори $t \in [1; J]$ відрізків $\Psi(t)_{pack}$ даних для формування множини $\{\Lambda\}$ пропонується визначати, керуючись системою виразів:

$$\begin{cases} t = \varepsilon + (t-1), \\ \varepsilon = \text{rand}(J); \end{cases} \tag{3.10}$$

для даної системи виразів передбачається, що початковий ідентифікатор (t-1) являє собою:

- фіксовану величину (це, зокрема, може бути 1, 2 та ін.);
- величину, яка обчислюється як випадкове значення у межах раніше встановленого діапазону J.

У свою чергу, параметр ε у виразі (3.10) являє собою ***розмірність кроку вибірки***. Значення ε у загальному випадку має відповідати умовам, як показує наступна нерівність:

$$\varepsilon_{min} \le \varepsilon \le \varepsilon_{max}, \tag{3.11}$$

де ε_{max} - найбільше порогове значення розмірності кроку вибірки, яке, у свою чергу, обчислюється згідно з наступним виразом:

$$\varepsilon_{max} = \text{ceil}(\frac{J}{\xi}), \tag{3.12}$$

де ξ - параметрична величина, що може приймати значення у діапазоні $\xi = \overline{3;5}$;

ε_{min} - найменше порогове значення розмірності кроку вибірки, що приймає цілочисельні значення у діапазоні величин $\varepsilon_{min} = \overline{1;2}$.

Таким чином за умови, що J=20, розмірність кроку вибірки ε належатиме діапазону значень від 7, до 1-2. При цьому зрозуміло, що відповідно до зростання параметричної величини ξ спостерігатиметься скорочення верхньої межі діапазону.

Далі, наступним кроком після визначення поточного значення параметру ε, інакше кажучи, отримання сукупності ідентифікаторів відрізків

$\Psi(t)_{pack}$ інформаційного потоку, локалізованого межами часового інтервалу T, здійснюється операція зчитуванням усієї множини $\{\Lambda\}$ відрізків, які увійшли до неї $\{\Theta\}$.

Водночас за умови, коли одержане таким чином значення параметру ε не задовольняє вимоги (3.11), далі її обчислення виконується повторно, керуючись при цьому виразом (3.12).

Так, рисунок 3.7 демонструє приклад множини $\{\Lambda\}$, для якої найбільше та найменше порогове значення розмірності кроку вибірки відповідно дорівнюють $\varepsilon_{min} = 1$ та $\varepsilon_{max} = 7$ за умови, що J=20.

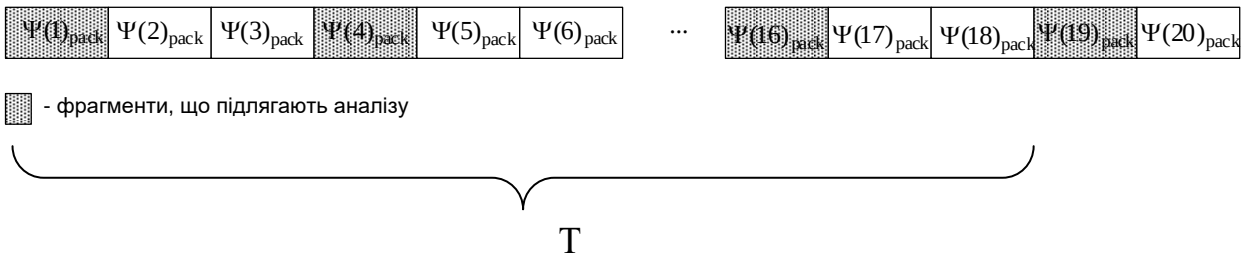


Рисунок 3.7 – Приклад сукупності $\{\Lambda\}$ ідентифікаторів відрізків $\Psi(t)_{pack}$ інформаційного потоку, утворена при ε =3 за умови, що J=20

Таким чином, у спосіб, наведений виразами (3.10)-(3.12), створюються умови для можливості рівноймовірного доступу практично до кожного з відрізків $\Psi(t)_{pack}$ потоку даних у рамках, обмежених часовим інтервалом T.

4. МЕТОДИ СТЕГОАНАЛІЗУ ДЛЯ ВИЯВЛЕННЯ ТА ПРОТИДІЇ ПРИХОВАНИХ ЗЛОВМИСНИХ ВПЛИВІВ НА ВЕБ-ВУЗЛИ

4.1 Ключові принципи створення алгоритмів виявлення прихованого контенту

Як показує попередньо виконаний аналіз файлів, що належать різним типам та представлені різними форматами, одними з найбільш ефективних у ролі стеганографічних контейнерів є файли, що містять у собі графічну інформацію.

Зазначимо, що даний висновок повністю збігається з даним офіційних досліджень [25-30].

Таким чином, під час подальшого дослідження принципів створення та функціонування алгоритмів стеганографічного аналізу у першу чергу необхідно взяти до уваги саме файли .

Хоча зараз, в умовах високого рівня кіберзагроз, використання методів стегоаналізу є значно затребуваним, водночас, більшість аналітичних алгоритмів, у сутності, перебувають на стадії доопрацювання та узгодження. Головними причинами для цього є:

- фактична відсутність уніфікованих стандартів, що описують архітектуру та базові засади побудови алгоритмів стегоаналізу;
- існування значного обсягу принципів та моделей реалізації алгоритмів.

Означені чинники, при цьому, відносяться безпосередньо як до математичної реалізації даних алгоритмів, так і до усіх можливих характеристик потенційних контейнерів, які потребують дослідження [28].

У той же час, серед усього обсягу існуючих сьогодні алгоритмів стеганографічного аналізу умовно можна виділити ряд груп, а саме [27, 29]:

1. Алгоритми стегоаналізу, спрямовані на протидію конкретним та початково відомим механізмам вбудовування інформації.
2. Уніфіковані алгоритми, тобто, такі, що теоретично здатні виявляти факт наявності контейнерів, утворених на базі будь-якого стегоалгоритму.

Для уніфікованих алгоритмів властивим є те, що у їх випадку комплекс виконуваних процедур зі стеганоаналізу може реалізовуватися не маючи попередніх даних відносно:

- типу алгоритму маскування використаного зловмисником;

- застосованого криптографічного алгоритму (за умови його використання);

- використаної технології стиснення даних;

- відомостей щодо стегоключа;

- даних про ймовірний або фактичний розмір прихованого повідомлення.

Водночас, щоб забезпечити результативність уніфікованих алгоритмів, попередньо необхідно виконати етап їх «навчання».

У ході даного технологічного етапу виконується аналіз великого обсягу (зазвичай 1-2 сотень) серій контейнерів та фрагментів даних, що не мають вбудовувань приховуваного змісту.

Слід також зазначити, що алгоритми стеганографічного аналізу обох груп створюються з урахуванням ймовірності того, що доступність для аналізу початкового незаповненого стегоконтейнеру, що слугує для інкапсулювання даних приховуваного повідомлення, не може бути гарантованою.

При цьому, до складу першої групи алгоритмів стегоаналізу входять механізми ***схемного*** та ***сигнатурного*** типів [27, 30].

У свою чергу, в основі роботи методів сигнатурного типу знаходяться механізми синтаксичного аналізу, на базі яких досліджуються послідовності символів, які надходять до буферу приймача, та у загальному випадку можуть можуть утворювати стегоконтейнер.

При цьому, якщо та чи інша послідовність символів, що надійшла до буферу, визнається такою, що належить тій чи іншій умовній мові опису стеганографічної системи, надалі вважається що файл, що містить у собі досліджений інформаційний фрагмент, являє собою стегоконтейнер.

Відповідно, канал, яким отримано цей сегмент даних, підлягає, блокуванню.

Алгоритми сигнатурного типу мають суттєву перевагу, яка полягає у одностайності отриманих результатів.

Водночас, істотним обмеженням застосування сигнатурних алгоритмів стегоаналізу є те, що відносно невеликому відсотку (на рівні 10% у загальному обсязі) програмного забезпечення, що використовується для маскування даних, властивим є залишати свої сигнатури у межах заповнених контейнерів.

У свою чергу, методи схемного типу [31] частіше за все застосовуються для того, щоб підтвердити або спростувати гіпотезу про існування чи, навпаки, відсутність вбудованої інформації на базі апріорно відомої системи маскування даних.

Найбільш широко сьогодні застосовуються такі методи схемного типу, як:

- метод Хі-квадрату;
- методи візуального дослідження;
- методи статистичного аналізу.

Так, ***метод Хі-квадрату*** найчастіше сьогодні застосовується для обгрунтування гіпотези щодо існування заповнених контейнерів, які побудовано на базі ряду спеціалізованих програмних засобів, таких, як Jpeg Hide & Seek, Jsteg, OutGuess та ін. [31, 32].

У даному випадку беруться до уваги існуючі особливості розподілу статистики усередині заповнених контейнерів, як одна з ключових ознак застосування раніше перелічених програмних засобів.

Серед найбільших переваг алгоритмів стегоаналізу схемного типу може бути зазначено такі:

- низька вірогідність помилкових рішень за результатами аналізу та інтерпретування отриманих даних;

- значний рівень вірогідності успішного вилучення маскованих даних за умови, що одночасно з розпізнаванням стегоконтейнеру вдається також виявити тип стегосистеми, контейнери якої при цьому не містять характерних сигнатур.

Разом з тим, базисом майже усіх існуючих сьогодні алгоритмів ***візуального дослідження*** є спроможність зорової системи людини аналізувати візуальні об’єкти з наступним розпізнаванням фактів існування значних відмінностей у межах зображень, що підлягають порівнюванню [33].

Зараз сімейство ***алгоритмів візуального аналізу*** надають найпростіші механізми дослідження файлів графічного типу, так як дана процедура, у сутності, найчастіше зводиться до візуальної оцінки отриманого графічного об’єкту .

Водночас, алгоритми візуального аналізу дозволяють задавати обмеження щодо обсягу інформації, яка інкапсулюється у контейнер.

Зокрема, якщо розглядати зображення у палітрі RGB, подане у форматі bmp-файлу, то на цей випадок непомітними для зору вважається рівень

викривлень візуальних даних, що у середньому не перевищує значення 3% [29].

Одночасно з цим, для випадку графічних контейнерів jpeg ймовірність успішного виявлення факту вбудованих даних є практично нульовою. Прчиною цьому є те, що несуттєві, але помітні спотворення, які можуть спостерігатися після виконання процедури інкапсуляції, частіше за все сприймаються як результати надмірного стиснення даних у ході кодування.

Водночас зазначимо, що головним недоліком алгоритмів візуального аналізу, окрім вже зазначеної обмеженості щодо візуального дослідження зображень jpeg, є неможливість оцінки суттєвих об’ємів графічної інформації у реальному масштабі часу [27].

Разом з тим, ключовий недолік алгоритмів стегоаналізу даної групи повністю відсутній для алгоритму ***візуального аналізу площин бінарних елементів***.

Гловний принцип означеного алгоритму, на відміну від класичного алгоритму візуального аналізу, полягає у тому, що:

- виконується співставлення прийнятого графічного об’єкту та виокремленими з його складу площинами бінарних елементів;

- у межах площин бінарних елементів ймовірного контейнеру, що аналізується, виконується пошук структурних аномалій.

Іншими словами, під час виконання процедури стегоаналізу з використанням даного алгоритму передбачається послідовне сканування понетційного контейнеру за на рівні площин бінарних елементів.

При цьому, ураховуючи те, що як яскравісні, так і хроматичні канали зображення jpeg потребують за замовчуванням 1 байт для опису, відтак у кодному з каналів необхідно опрацьовувати 8 площин бінарних елементів.

Таким чином, у межах кожного каналу перша площина бінарних елементів утворюється як зображення, що будується за рахунок об’єднання сукупності біт нульового, тобто, молодшого розряду.

У свою чергу, друга площина являє собою зображення, утворене поєднанням бінарних елементів першого розряду і т.д.

Одержані таким чином площин бінарних елементів далі порівнюються з композитним, тобто, початковим зображенням, яке досліджується. Розглянемо зазначений підхід більш детально.

При цьому за основу візьмемо випадок, коли має місце контейнер, утворений у базисі LSB (least significant bit), тобто, тут виконуэться вбудовування даних у наймолодший розряд.

Для цього випадку принцип інкапсулювання реалізується відповідно до наступного принципу [25]:

$$\begin{cases} \nu'(\alpha)^{(c)}_{(i,j)} := 0 \mid \alpha_i = 1 \,\&\, \nu(\alpha)^{(c)}_{(i,j)} = 0 \\ \nu'(\alpha)^{(c)}_{(i,j)} := 1 \mid \alpha_i = 1 \,\&\, \nu(\alpha)^{(c)}_{(i,j)} = 1 \end{cases}, \tag{4.1}$$

де $\alpha_{i,j}$ - молодший біт (за замовчуванням – біт нульового розряду), що належить i, j -й компоненті;

$\nu(\alpha)_{(i,j)}$ - біт повідомлення, що вбудовується у контейнер;

$\nu'(\alpha)_{(i,j)}$ - результуючий біт LSB, що утворюється у наслідок виконання процедури інкапсуляції.

Тобто, коли біт $\alpha_{i,j}$ молодшого розряду зображення-контейнера має такеж двійкове значення, як і біт $\nu(\alpha)_{(i,j)}$, що вбудовується, величина $\alpha_{i,j}$ не зазнає змін.

Якщо у ході виконуваного аналізу виявлено, що зміст хоча б однієї з площин бінарних елементів не корелюється зі змістом вихідного зображення, це може вказувати на те, що досліджуване зображення являє собою контейнер.

У свою чергу, на рисунку (4.1) бачимо приклад вихідного зображення та результат сканування площини бінарних елементів (нульовий розряд) для випадку, коли зображення не є контейнером LSB.

Водночас, рис.4.2 демонструє приклад площини бінарних елементів, що містить інкапсульовані біти приховуваного контенту.

а)

б)

Рисунок 4.1 – Приклад вихідного зображення, та площини бінарних елементів на рівні LSB, що йому відповідає

Рисунок 4.2 - Площина бінарних елементів з ознаками інкапсуляції

Ключовим питанням забезпечення функціональності методу аналізу площин бінарних елементів є урахування способу інкапсулювання біт приховуваного повідомлення.

Наприклад, у випадку, коли процедура вбудовування реалізується послідовно, тобто, інкапсуляція здійснюється у біти, які розміщені поряд, факт наявності контенеру може бути гарантовано виявлено за будь-яких обставин (рисунок 4.2).

Разом з тим, також візуально може бути виявлено факт присутності інкапсульованох даних приховуваного контенту за умови, коли має місце інкапсулювання зі значним рівнем заповнення.

Окрім цього, зазвичай статистичні характеристики приховуваного контенту, який передбачається вбудувати у зображення-контейнер, суттєво відрізняються від аналогічних характеристик для площини бінарних елементів контейнеру LSB.

Таким чином, візуальні відмінності між ділянками контейнеру, що мають заповнення, та незаповненими зонами можуть бути чітко помітні [27, 30, 31].

На відміну від візуальних алгоритмів, в основі іншої групи алгоритмів стегоаналізу - статистичних алгоритмів знаходиться концепція т.з. «природного» контейнера [31-33].

Зазначена група алгоритмів базується на дослідженні ймовірності присутності вбудовуваних даних, керуючись критерієм оцінки ступеню подібності зображення, що підлягає стегоаналізу, до «природного» контейнеру.

Основна перевага алгоритмів статистичного аналізу полягає у фактичній відсутності обмежень щодо області їх застосування з одночасним збереженням високого рівня ефективності.

Це, у свою чергу, дозволяє на їх базі:

- виконувати перевірку гіпотез щодо визнання зображень контейнерами, сформованими за участю апріорі невідомого стеганографічного методу;

- здійснювати розробку алгоритмів стегоаналізу на основі схемних підходів.

4.2 Статистичні методи аналізу контенту

4.2.1 Алгоритм на базі збору статистики щодо двійкових переходів молодших біт

В основі роботи зазначеного алгоритму є твердження про існування кореляційних закономірностей між:

- бінарними елементами LSB $\alpha(c-ch)_{x,y}$ розміщених поряд пікселів $p_{x,y}$;

- бінарними елементами $\alpha(ch)_{x,y}$, у складі компонент $\hat{p}_{x,y}$ будь-яких каналів (зроматичні, яскравісні);

- іншими бінарними елементами у межах природних контейнерів [31-33].

Так, у випадку аналізу зображень, поданих у форматі BMP, множиною елементів, яка досліджується, є сукупність сусідніх LSB-біт $\{\alpha(c-ch)_{x,y}\} = \bigcup_{a=x\pm1}^{x\pm\nu} \bigcup_{b=y\pm1}^{y\pm\lambda} \alpha(c-ch)_{a,b}$ у рамках кожного з каналів (R, G та B) пікселів.

Разом з тим, на випадок дослідження зображень jpeg, виконується аналіз сукупності $\{\alpha(ch)_{x,y}\} = \bigcup_{a=x\pm1}^{x\pm\nu} \bigcup_{b=y\pm1}^{y\pm\lambda} \alpha(ch)_{a,b}$ LSB-біт сусідніх компонент, величина яких не дорівнює 0 або 1.

При цьому, залежність між бінарними елементами у межах одного розряду потенційного контейнера підлягає марківському розподілу [26]. Водночас, характер таких залежностей залежить від номеру розряду.

У рамках даного алгоритму поняття «перехід» означає зміну значення і-го бінарного елемента щодо значення $i+1$-го елементу двійкового масиву x, де $i = 1,2,...,n-1,n$ - його довжина. У даному випадку, так яв масив є бінарним, аналізу підлягають такі класи переходів,як:

- $0 \rightarrow 0$;
- $0 \rightarrow 1$;
- $1 \rightarrow 0$;
- $1 \rightarrow 1$.

На наступному кроці, керуючись одержаними результатами, формується гістаграми переходів.

У цьому випадку, розряди діаграми ілюструють наявний обсяг переходів. Зокрема, існуючий обсяг переходів у LSB з 0 до 0 ілюструється першим стовпцем діаграми, аналогічним чином другий стовпець показує кількість переходів з 0 на 1. Відповідно до цього - третій стовпець відображає переходи з 1 до 0, а четвертий, у свою чергу - з 1 до 1, як показує рисунок 4.3.

В умовах, коли розглядається незаповнений контейнер (початкове зображення без інкапсулювання) та вже сформований стегоконтейнер, кількість бінарних переходів у множині LSB є значно відмінним. Наприклад, для заповненого контейнера стегосистеми характер розподілу LSB елементів зазвичай є випадковим. Звыдси виходить, що величина F, тобто, обсяг бінарних переходів у межах сукупності біт LSB для будь-якого з чотирьох станів є наближено рівною. Це, у свою чергу, є нехарактерним для зображень, що не є контейнерами [29].

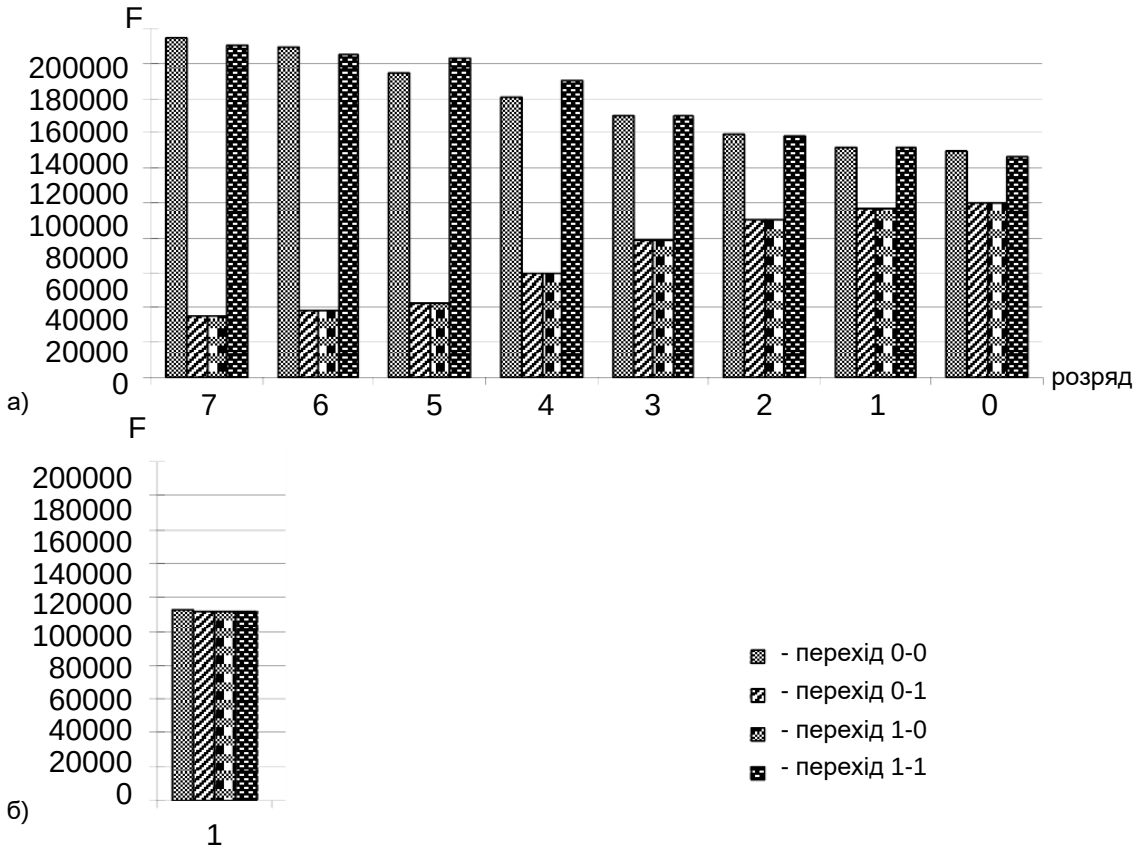


Рисунок 4.3 – Приклад гістаграми обсягу бінарних переходів для звичайного зображення а) та для заповненого контейнера б)

4.2.2 Алгоритм оцінки частот появи k-елементних ланцюжків у межах LSB контейнера

Означений алгоритм дозволяє оцінити наявний характер розподілу біт, а саме – його ступінь рівномірності, що має місце для послідовності, яка аналізується. При цьому ведеться збір статистики частоти появи окремих біт та їх серій, що формуються з k складових [29-31].

У ході цього, послідовності x, що підлягає аналізу, розглядається у бінарному форматі представлення.

У свою чергу, для даної послідовності фіксується частота появи:

- одиничних та нульових біт $(k = 1)$;

- двобітних ланцюжків $(00, 01, 10, 11 : k = 2)$;

- серій, утворених трьома елементами $(000, 001, 010, 011, 100, 101, 110, 111 : k = 3)$ і т. ін.

Далі, ураховуючі зібрані статистичні відомості, здійснюється формування гістаграми.

При цьому, коли виконується аналіз потенційного контейнеру, представленого зображенням jpeg, для створення гістаграми використовуються частоти появи бінарних ланцюжків у межах LSB компонент, утворених після процедур ДКП та квантування, які при цьому не дорівнюють 0 або 1.

Даний алгоритм базується на твердженні про те, що для зображень формату bmp та jpeg наближена рівність значень частот усіх компонент не є властивою, що демонструє рис. 4.4 а). Разом з тим, після вбудовування приховуваних біт значення частот буде змінено до приблизно однакових величин, як показано рисунком 4. 4 б).

Таким чином, у ситуації, коли справедливими є умови, подані рис. 4.4.б), у загальному випадку зображення, яке аналізується, попередньо може вважатися контейнером.

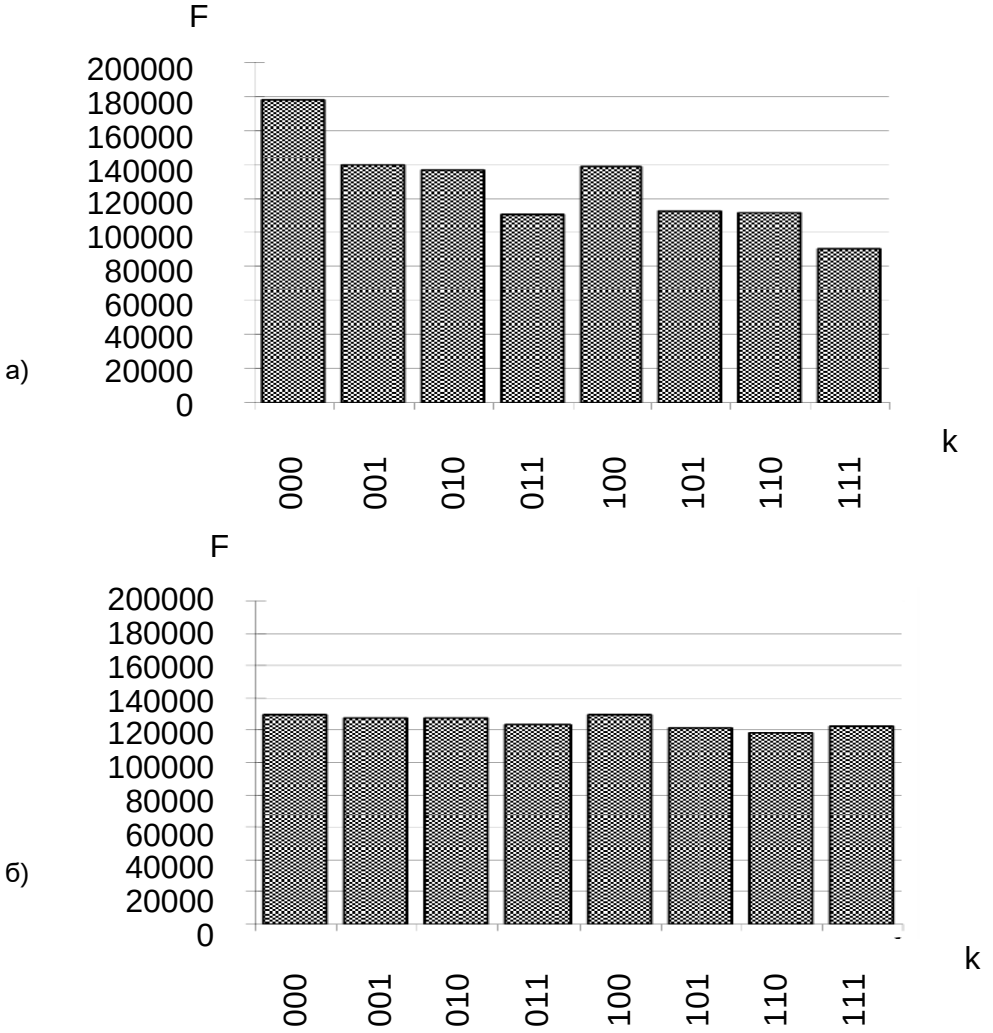


Рисунок 4.4 – Гістограма частот появи серії з тьох бінарних елементів для випадку LSB звичайного зображення а) та для контейнеру б)

Даному алгоритму властивий один недолік, що полягає у залежності між рівнем наповненості контейнеру, та ймовірністю його успішної ідентифікації. Так, алгоритм дозволяє одностайно розпізнати контейнер лише у випадку, коли його заповнено на рівні понад 60%.

4.2.3 Алгоритм аналізу характеру розподілу елементів графічних об’єктів у двовимірному просторі

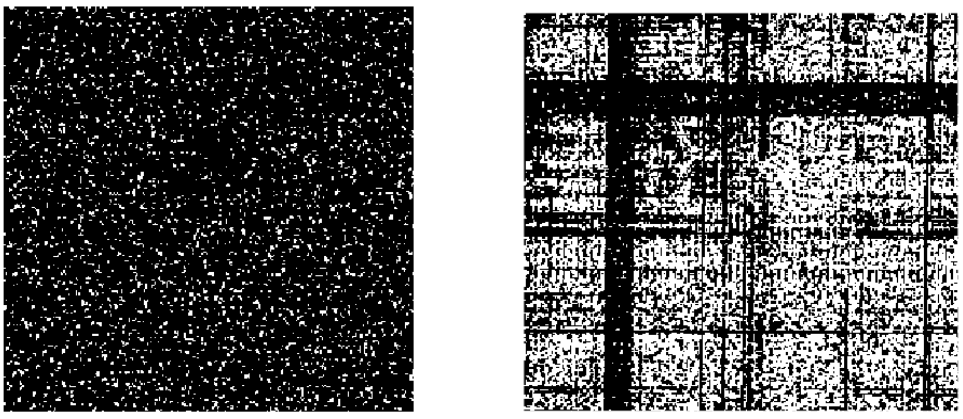
Даний алгоритм дозволяє виявити закономірності, що повязують між собою елементи потенційного графічного контейнера [30, 31].

У ході роботи алгоритму спочатку створюється віртуальний робочий простір, що являє собою двовимірний масив, що за розмірністю дорівнює $(2^R-1)\times(2^R-1)$, де R - кількість розрядів для опису елементу ймовірного

контейнеру, що підлягає аналізу. Далі у ході роботи алгоритму до робочого простору вносяться мітки, що мають координатами $(x_i; x_{i+1})$, де x_i - елементи аналізованої послідовності. Тут параметр x, що змінюється у деякому діапазоні $x, i = \{1,2,...,n-1,n\}$, є ідентифікатором довжини послідовності. Одержавши наявний розподіл довжин, що описує досліджуваний потенційний контейнер, далі виконується її оцінка та інтерпретація отриманих даних.

Зокрема, за умов рівномірного розподілу міток у межах віртуального робочого простору, можна зробити висновок, що аналізоване зображення не містить міжелементних зв'язків. Тобто, дане зображення з великою долею ймовірності є контейнером (рис.4.5 а).

Водночас, у межах робочого простору може спостерігаєтися структурованість елементів послідовності, що підлягає аналізу. Зазвичай це - ділянки суцільного заповнення/відсітності елементів, що, у свою чергу, формують геометричні фігури та/або примітивні малюнки. Наявність таких об'єктів у межах робочого простору дозволяє зробити висновок про те, що дане зображення не є контейнером, як показно рис. 4.5 б) [30].



а) б)

Рисунок 4.5 – Приклади розподілу елементів у межах віртуального робочого простору для випадку контейнеру а) та для звичайного зображення б)

4.3 Концепція удосконалення алгоритмів візуального аналізу

На сьогодні класичні алгоритми стегоаналізу, що належать до групи візуальних, не гарантують достатнього рівня продуктивності, який дозволяє виконувати дослідження потоку даних у реальному масштабі часу. З іншого боку, архітектура алгоритмів візуальної групи дозволяє вносити доробки та удосконалення в існуючі аналітичні механізми. Інакше кажучи, алгоритми даної групи є досить перспективними для подальшого використання за умови їх відповідної модифікації.

У нашому випадку ***пропонується*** виконати доробку класичного підходу до реалізації методів візуального аналізу. Для цього, у першу чергу, необхідно модифікувати ряд механізмів, а саме:

- механізм дослідження LSB-простору ймовірних контейнерів;
- механізм дослідження контурної інформації.

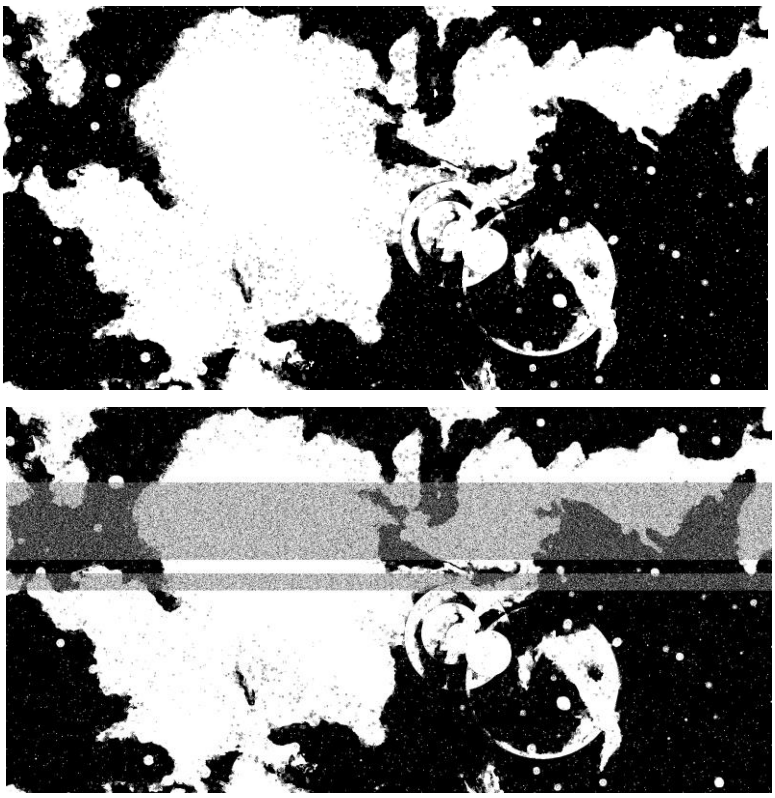
4.3.1 Модифікація механізму дослідження LSB

Запропонований механізм базується на існуючих засадах алгоритмів візуального аналізу.

Водночас, на відміну від випадку класичної реалізації алгоритму, тут передбачається застосування інструментів автоматичної обробки та інтерпретації інформації, отриманої за результатами обробки. Відтак, створюються умови для можливості опрацювання даних у базисі візуальних методів без участі експерта. Таким чином, забезпечується можливість застосування алгоритмів візуального аналізу у реальному часі.

У першу чергу, механізм може вважатися результативним для ідентифікації стегоконтейнерів, які заповнюються за фронтальним алгоритмом інкапсулювання, як показано рисунками 4.6 та 4.7.

При цьому, як видно з аналізу рисунку 4.6 б, коли інкапсуляція приховуваних даних у простір LSB стегоконтейнеру реалізується на базі фронтального алгоритму, у межах молодшої площини бінарних елементів візалізуються ділянки, які характеризуються невластивим для природного зображення способом розподілу біт [33].



а)

б)

Рисунок 4.6 – LSB зображення, що не містить вбудовувань а) та LSB зображення-контейнеру, заповненого за фронтальним принципом б)

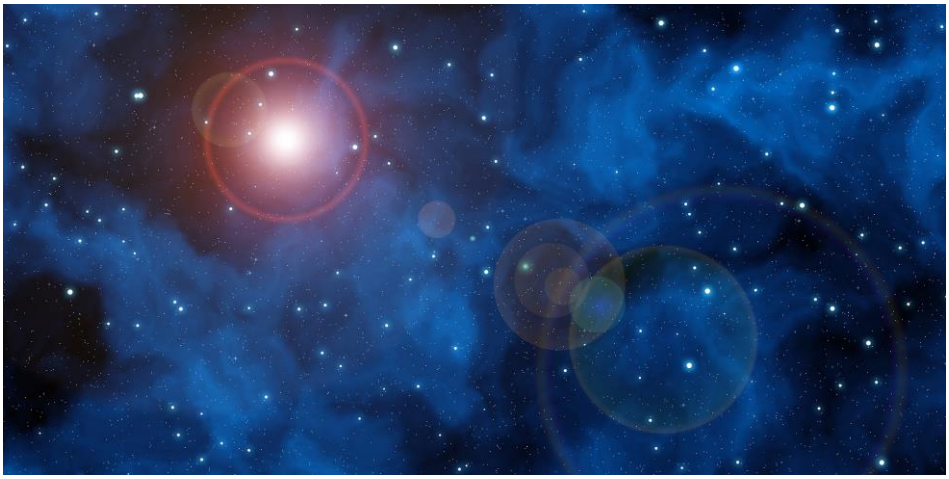


Рисунок 4.7 – Початкове зображення

При цьому, за класичною схемою візуального аналізу, у ході першого етапу дослідження потенційного стегоконтейнеру десятковий формат прдставлення компонент $\widehat{p}(ch)_{x,y}$ зображення змінюється на бінарного. У цьому випадку, якщо аналізується зображення jpeg, зміна формату його опису виконується відповідно до формули:

$$\hat{p}(ch)_{x,y} = \sum_{\theta=8}^{1} \alpha_\theta(ch)_{x,y} \times 2^{(\theta-1)}, \qquad (4.2)$$

де $\hat{p}(ch)_{x,y}$ - компонента зображення, що є елементом каналі ch, та міститься на позиції (x, y) у межах блоку $\hat{q}_{i,j}$ зображення, сформованого у результаті виконання ортогонального перетворення початкового блоку $q_{i,j}$ розміром 8x8 на базі ДКП. У даному випадку параметр ch є ідентифікатором каналу яскравості, або хроматичності, відтак, може дорівнювати Y, Cr чи Cb;

$\alpha_\theta(ch)_{x,y}$ - біт θ-го розряду каналу ch компоненти з координатами (x, y) у межах блоку $\hat{q}_{i,j}$ зображення.

Після зміни формату опису даних реалізується процедура сканування масиву двійкових елементів, що належать нульовому розряду, тобто, LSB $(\theta = 1)$, у підсумку виконання якої виконується побудова площини бінарних елементів згідно з наведеним нижче виразом:

$$\Xi(\theta) = \bigcup_{x=1}^{H} \bigcup_{y=1}^{W} \alpha_\theta(ch)_{x,y} \qquad (4.3)$$

У такий же спосіб здійснюється побудова площини бінарних елементів для якого завгодно розряду також і для контейнерів на основі зображень bmp. Проте, на відміну від зображень jpeg, де розглядаються компоненти $\hat{p}_{x,y}$, усы операції виконуються відносно пікселів $p_{x,y}$, кожен з яких являє собою сукупність чисельних значень у кожному з базових каналів (, а саме - R, G та B).

Далі після того, як множину $\Xi(\theta)$ побудовано, здійснюється оцінка середнього розміру серії бінарних елементів за рядками, а також за стовпцями. При цьому, дослідження рядків та стовпців виконується, як незалежні між собою операції.

У свою чергу, розрахунок значення середньої довжина $\bar{\ell}^{(r)}_{i,j}$ серії бінарних елементів за рядками здійснюється відповідно до виразу:

$$\bar{\ell}_{i,j}^{(r)} = \frac{\sum_{\varphi=1}^{8} \upsilon_{\varphi}}{8}, \tag{4.4}$$

де υ_{φ} - обсяг бінарних переходів, зафіксованих у межах одного рядку.

Аналогічним чином здійснюється обчислення середньої довжини $\bar{\ell}_{i,j}^{(c)}$ серії бінарних елементів за стовпцями:

$$\bar{\ell}_{i,j}^{(c)} = \frac{\sum_{\psi=1}^{8} \upsilon_{\psi}}{8}, \tag{4.5}$$

де υ_{ψ} - загальна кількість бінарних переходів, виявлених для одного стовпця.

Далі за підсумком розрахунку значень $\bar{\ell}_{i,j}^{(c)}$ і $\bar{\ell}_{i,j}^{(r)}$ при $\theta = 0$ для усіх блоків $\hat{q}_{i,j}$, що утворюють ймовірний контейнер, виконується порівняння між собою усіх одержаних показників.

Якщо у ході такого порівняння одному чи ряду блоків $\hat{q}_{i,j}$ відповідають показники $\bar{\ell}_{i,j}^{(c)}$ та $\bar{\ell}_{i,j}^{(r)}$, для яких є справедливим наступне співвідношення:

$$\bar{\ell}_{i,j}^{(c)} \approx \bar{\ell}_{i,j}^{(r)} \to 7, \tag{4.6}$$

тоді попередньо вважається, що даний блок може містити вбудовану інформацію на базі методу модифікації LSB з фронтальним алгоритмом розміщення біт.

Водночас, вираз (4.6) являє собою необхідну, проте недостатню ознаку присутності стегоконтейнеру.

У свою чергу для того, щоб обґрунтувати, або навпаки – спростувати гіпотезу про те, що досліджуваний графічний обєкт містить вбудований контент, тобто, є стегоконтейнером, подальшому дослідженню підлягає весь обсяг трансформованих блоків $\hat{q}_{i,j}$ у зображенні. За результатами цього утворюються наступні сукупності блоків:

- сукупність $\hat{q}_{i,j}^{(u)}$ блоків, що відповідають умові, поданій виразом (4.5). Тобто, блоків, що на поточному етапові дослідження попередньо визнано елементами стеганографічного контейнеру;

- масив $\hat{q}_{i,j}^{(s)}$ блоків, що не відповідають умові (4.6), тобто, ймовірно не є контейнерами.

На наступному кроці виконується оцінка фактичного розміщення блоків, які формують масив $\hat{q}_{i,j}^{(u)}$, у ході чого здійснюється обхід утвореного двовимірного масиву. У ході даної процедури зчитування даних починається з блоку $\hat{q}_{i,j}^{(u)}$, якому відповідає координата (1,1), іншими словами, поблокове сканування зображення починається фактично зі стартової точки координат.

Далі блоки $\hat{q}_{i,j}^{(u)}$ скануються у двох напрямках – за рядками та, за стовпцями, одночасно з чим фіксуються їхні координати (i,j) у зображенні.

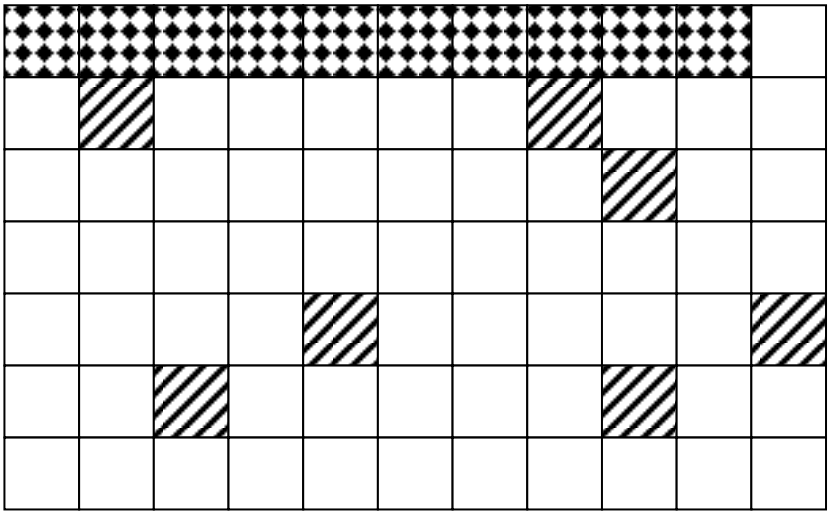
За умови, що у процесі виконання скану було виявлено хоча б один трансформований блок $\hat{q}_{i,j}$, який є сусіднім з поточним у рядку чи стовпцю, який відповідає умовам, поданим виразом (4.6), це дає підстави вважати, що досліджуваний графічний об’єкт являє собою стегоконтейнер [27].

У свою чергу, у ході виконання аналізу зображення може виникнути ситуація неповного виконання умови (4.6), що є еквівалентним наступному виразу:

$$\bar{\ell}_{i,j}^{(c)} \vee \bar{\ell}_{i,j}^{(r)} \to 7. \qquad (4.7)$$

Для такого випадку вважається, що ознака присутності контейнеру не є чіткою. Відтак, для подальшого аналізу зображення доцільно використати додаткові аналітичні механізми.

Наприклад, рисунок 4.8 демонструє графічну інтерпретацію результатів виконаного аналізу зображення, що містить ознаки контейнеру, та має у своєму складі окремі блоки і їх сукупності, які за необхідною ознакою присутності вбудованої інформації містять інкапсульовані дані. Разом з тим, за певних умов необхідна ознака присутності вбудовувань може виникнути як результат дії завад.



- сукупність блоків з ознаками інкапсуляції за фронтальним принципом;

- блоки, які мають необхідну ознаку ймовірного контейнеру

Рисунок 4.8 – Приклад результатів дослідження зображення на ознаки LSB-модифікацій за фронтальним принципом

Водночас, такий підхід до модифікації алгоритму стегоаналіз LSB-у не може вважатися ефективним у таких випадках:

- вбудовування інформації у LSB зображення реалізується на базі механізмів, що передбачають не фронтальний спосіб розміщення інкапсульованих даних;

- інкапсуляція даних виконується у межах ділянок зображення, які містять зони шумоподібного наповнення, що має «природний» характер, як це продемонстровано рис. 4.9;

- підлягають вбудовуванню LSB блоків, які містять контури.

У свою чергу, на випадок модифікації контурних зон пропонується застосовувати механізм аналізу контурних зон ймовірних контейнерів [34].

4.3.2 Механізм аналізу контурних зон ймовірних контейнерів

Зараз класичні LSB-методи стеганографії мають найнижчий рівень захищеності. Разом з тим, у випадку, коли дані за принципом LSB інкапсулюються у контурну ділянку, за умови низької наповненості

контейнеру ймовірність його виявлення є майже нульовою, що пояснюється наступним:

- місткість LSB-контейнеру у межах якого дані інкапсулюються у контурні зони, у середньому є не більшою, ніж 1%. У таких умовах застосування майже усіх відомих статистичних методів стегоаналізу є неефективним;

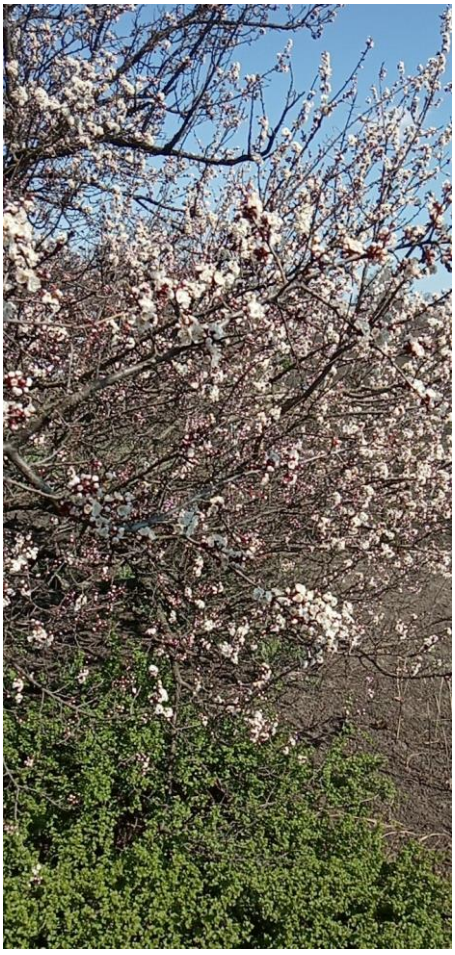
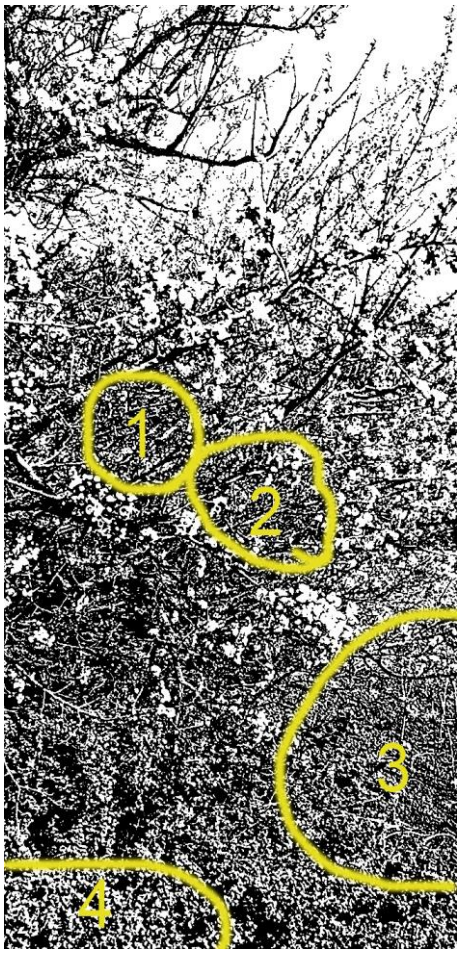
а)  б) 

Рисунок 4.9 – Початкове зображення а) LSB-площина зображення, що містить зони шумоподібного заповнення, які зловмисник може застосовувати для інкапсуляції даних

- на швидкість виконання аналізу LSB потенційного контейнеру на основі розрахунку середньої кількості бінарних переходів для кожного окремого блоку $\hat{q}_{i,j}$ можуть суттєво впливати особливості змісту зображення. Таким чином, той чи інший рівень швидкодії не може гарантуватися.

На цей випадок пропонується виконати модифікації механізмів аналізу LSB.

У рамках цього зображення підлягає процедурі контурного аналізу, за результатами якої у його структурі локалізується множина наявних контурів. Далі відносно ділянок, що містять контурні дані, виконується операція поглибленого аналізу.

Передбачається можливість використання для виявлення контурів наступних аналітичних інструментів, таких, як:

- загальні механізми пошуку контурів;
- дослідження статистичних характеристик блоків $q_{i,j}$ зображення до виконання процедури ортогонального пертворення, або трансформованих блоків $\hat{q}_{i,j}$.

Якщо оцінювати наведені інструменти з прозицій потенційної продуктивності, можна зазначити, що як складова частина заходів зі стегоаналізу кожен з них є ефективним для застосування.

Водночас, наведені інструменти є відмінними на рівні загальної архітектури.

Наприклад, механізми виявлення контурів дозволяють лише встановити факту присутності стеганографічного контейнеру.

Далі на базі цих відомостей виконується подальше блокування каналу надходження файлу, який було визнано контейнером.

При цьому, на відміну від першої групи механізмів, підхід на базі дослідження статистичних характеристик блоків не орієнтується на пошук ознак контейнерів за результатами дослідження LSB зображень.

Разом з тим, такий підхід є превентивним.

Інакше кажучи, на його базі будуються захисні заходи, що, у свою чергу, передбачають цілеспрямовану модифікацію LSB-простору зображень з забезпеченням мінімізації внесеної помилки.

Також для того, щоб забезпечити зменшення обчислювального навантаження, що виникає у ході виконання аналізу зображень (що першочергово необхідно для реалізації заходів стегоаналізу у реальному масштабі часу) передбачається проведення дослідження статистичних характеристик блоків $q_{i,j}$ до виконання процедури ортогонального перетворення.

Для таких умов застосовується двоетапний механізм пошуку блоків, які містять контури.

Спочатку здійснюється розрахунок величини нормованого внутрішнього динамічного діапазону $d_{int}(q_{i,j})$ блоку у перерахунку до рядку та до стовпцю, керуючись наступною системою виразів:

$$\begin{cases} d_{int}^{(x)}(q_{i,j}) = \prod_{y=1}^{8} \frac{(p_{x,y}^{(max)} - p_{x,y}^{(min)})}{8}; \\ d_{int}^{(y)}(q_{i,j}) = \prod_{y=1}^{8} \frac{(p_{x,y}^{(max)} - p_{x,y}^{(min)}),}{8} \end{cases} \quad (4.8)$$

де $d_{int}^{(x)}(q_{i,j})$ - нормований внутрішній динамічний діапазон сегменту у перерахунку на рядок;

$d_{int}^{(y)}(q_{i,j})$ - нормований внутрішній динамічний діапазон сегменту у перерахунку на стовпець;

$p_{x,y}^{(max)}$ та $p_{x,y}^{(min)}$ - найбільша та найменша величина пікселя у межах стовпця або рядку.

На наступному кроці серед усього масиву пікселів досліджуваного зображення знаходиться такий піксель, якому відповідає максимальний показник $d_{int}^{(x)}(q_{i,j})$ та/або $d_{int}^{(y)}(q_{i,j})$.

У підсумку далі обирається результуюче значення $d_{int}(q_{i,j}) = f(d_{int}^{(x)}(q_{i,j}); d_{int}^{(y)}(q_{i,j}))$, що відповідає максимальній величині $d_{int}^{(max)}(q_{i,j})$.

У свою чергу, за умови, що для довільного блоку $q_{i,j}$ досліджуваного виконується умова, наведена наступним виразом:

$$d_{int}(q_{i,j}) \in [d_{int}^{(max)}(q_{i,j}); \vartheta d_{int}^{(max)}(q_{i,j})], \quad (4.9)$$

надалі такий блок включається у множину C' блоків, що йновірно можуть містити контури.

У виразі (4.9) множник ϑ знаходиться у діапазоні значень $\vartheta = \overline{0,6; 0,9}0,9$.

При цьому, його конкретне значення задається виходячи з особливостей зображення, яке підлягає аналізу.

Після того, як множину C' сформовано на базі виразів (4.8) та (4.9), дані щодо включення у неї тих чи інших блоків $q_{i,j}$, отриманих зазначеним чином, підлягають уточненню.

У ході уточнення попередньо отриманих даних відносно з кожного з блоків, що входять до множини C' виконується операція ортогонального перетворення на основ ДКП.

На наступному кроці розраховується значення добутку $\Omega(\hat{p}_{x,y})$ значень компонент $\hat{p}_{x,y}$, що належать перетвореному блоку $\hat{q}_{i,j}$.

Для цього використовуються компоненти, починаючи з DC (що знаходиться на позиції (1,1)) та включно до компоненти $\hat{p}_{x,y}$, що має координату (4,1).

При цьому, компоненти скануються у напрямку зиг-заг скану, тобто, зчитуються перші 10 компонент, що формують НЧ-групу. Тобто:

$$\Omega(\hat{q}_{i,j}) = \prod_{\delta=1}^{10} \hat{p}_{x,y}^{(\delta)}, \tag{4.10}$$

де $\hat{p}_{x,y}^{(\delta)}$ - компонента, що відноситься до зони низькочастотних.

Наступною дією, аналогічно визначенню максимальної величини нормованого внутрішнього динамічного діапазону за виразом (4.9), керуючись попередньо заданим значенням $\Omega(\hat{q}_{i,j})^{(max)}$, у межах множини C' виявляються перетворені блоки $\hat{q}_{i,j}$, для яких справедливою є умова:

$$\hat{q}_{i,j} \in [\Omega(\hat{q}_{i,j})^{(max)};\ \mu\Omega(\hat{q}_{i,j})^{(max)}], \tag{4.11}$$

де множник $\mu \in (1; 0)$, при цьому його значення може варіюватися залежно від особливостей конкретногозображення

У свою чергу, блок $q_{i,j}$ включається до множини C контурних у тому випадку, якщо виконується наступна умова:

$$q_{i,j} \in C \mid (q_{i,j} \in C') \ \&\ (\hat{q}_{i,j} \in [\Omega(\hat{q}_{i,j})^{(max)};\ \mu\Omega(\hat{q}_{i,j})^{(max)}]) \tag{4.12}$$

Далі для блоків $q_{i,j}$, що належать множині C блоків контурного типу, бінарна LSB-площина зазнає модифікації. Для цього пропонується використовувати наступні підходи:

1. Модифікація LSB-площина шляхом накладення рівномірного шуму.

Для цього у площину молодших бінарних елементів вносяться зміни, характер яких описується виразом:

$$\alpha_0(ch)_{x,y} := \mathrm{rand}[1;0], \tag{4.13}$$

таким чином, у межах LSB кожен символ отримує випадкове значення.

2. Обнулення LSB-площини.

У рамках цього підходу змісту LSB зображення зазнає знім відповідно до виразу:

$$b_\theta(ch)_{x,y} := 0, \tag{4.14}$$

це, у свою чергу, також повністю руйнує ймовірний контейнер, тим самим досягається безпечність отриманих даних.

І у першому, і у другому випадках понетційний контейнер, реалізований на рівні LSB елементів, що є елементами контурів, руйнується незворотньо.

Разом з тим, як видно з рис. 4.10-4.12, обнулення та рандомізація змісту LSB-площини бінарних елементів не веде до виникнення візуально помітних викривлень у зображення.

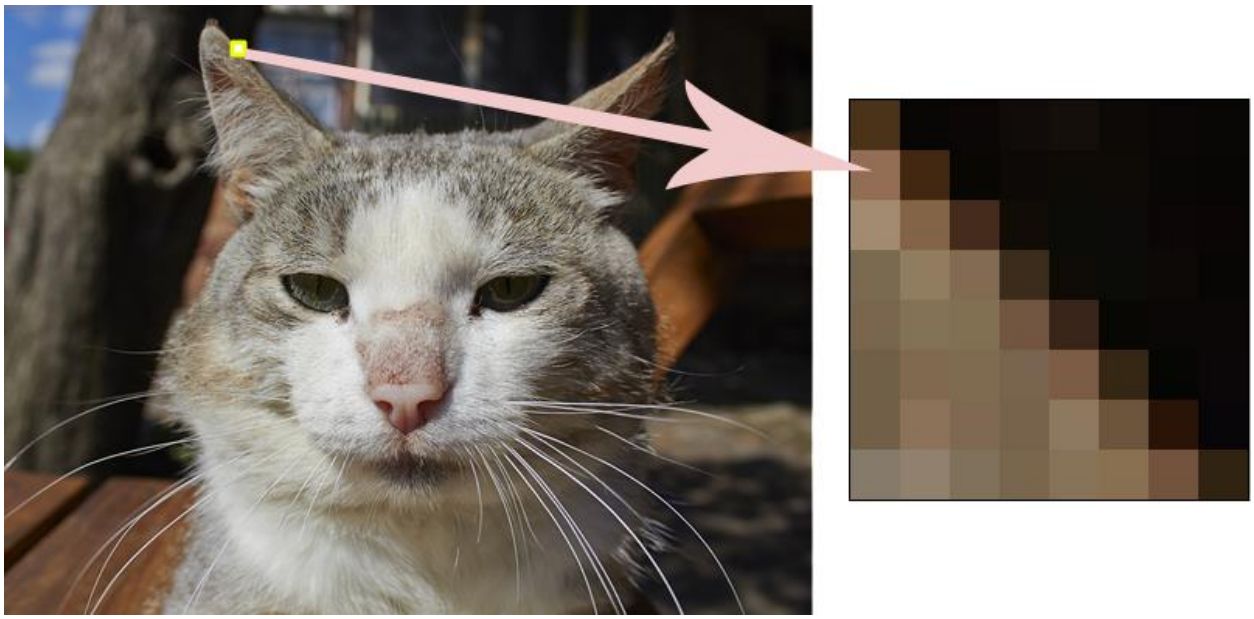


Рисунок 4.10 – Зовнішній вигляд графічного контейнеру та один з блоків у його складі, що містить контурні дані

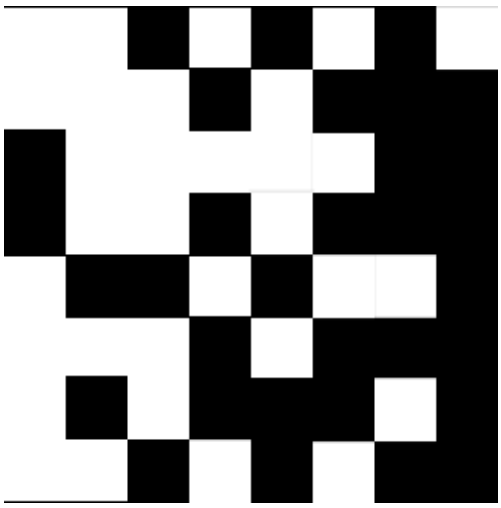
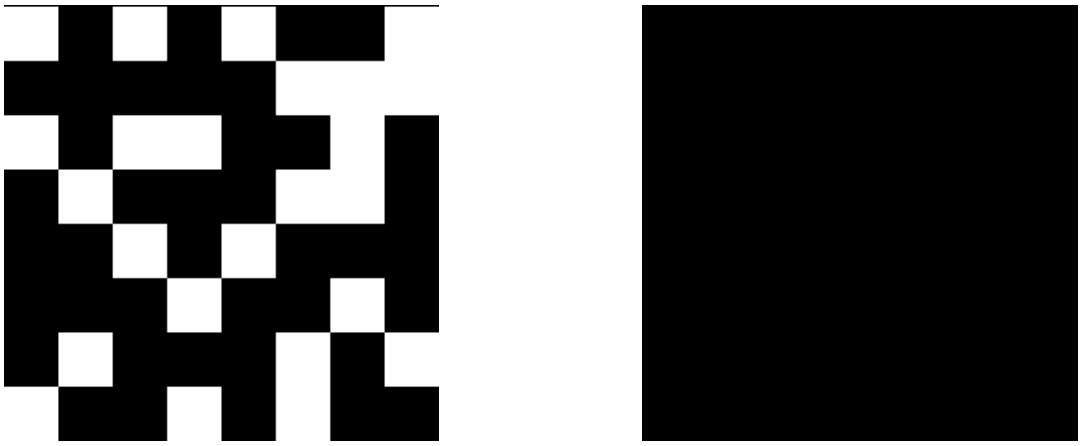


Рисунок 4.11 – LSB блоку зображення, яке містить контурну інформацію (належить множині C)



а) б)

в) г)

Рисунок 4.12 – Зміст LSB блоку за результатами виконання процедури рандомізації а) процедури обнулення обнулення б), початковий блок в) реконструйований блок, одержаний після превентивної модифікації LSB контейнеру г)

Розглянутий підхід дозволяє реалізувати додатковий інструментарій захисту в умовах, коли за попереднім аналізом зображення виявлено одну, або кілька часткових ознак присутності вбудованих даних, хоча за жодною з них наявність контейнеру достовірно не підтверджується. Передбачається можливість його спільно використання з іншими аналітичними алгоритмами.

ВИСНОВКИ

У відповідності з умовани технічного завдання, у ході виконання роботи було виконано дослідження комплексу заходів з технічного аудиту веб-ресурсу, як одного з інструментів забезпечення його продуктивності. Дослідження було проведено за 2 ключовими напрямками, а саме:

- дослідження загальних інструментів технічного аудита сайту на різних рівнях архітектури;

- дослідження методів виявлення та протидії зловмисним впливам на веб-вузол на базі алгоритмів стеганографії.

У рамках першого напрямку досліджень виконано:

- розгляд переліку технічних показників, які є ключовими для веб-вузла, незалежно від особливостей його реалізації, та роль кожного з таких показників;

- загальний аналіз переліку інструментів, що використовуються для виконання технічного аудиту.

Разом з тим, зазначено, що ключова роль технічного аудиту веб-вузла – виявлення проблемних аспектів його функціонування для подальшої розробки системи рекомендацій з їх усунення.

У свою чергу, другий напрямок досліджень присвячено алгоритмам виявлення та протидії прихованим зловмисним впливам на веб-вузол з використанням прихованої передачі зловмисного контенту, як складової частини APT. У рамках цього напрямку виконано наступне:

- розкрито сутності поняття APT та ролі стеганографічних алгоритмів у його складі;

- розроблено загальну схему заходів зі стегоаналізу, орієнтовану на обробку вхідного трафіку у реальному часі;

- досліджено найбільш поширені сьогодні алгоритми стегоаналізу, виявлено їхні недоліки та переваги;

- запропоновано концепцію побудови алгоритму стегоаналізу, орієнтованого на виявлення прихованих даних у рамках LSB графічних контейнерів.

Окрім цього, показано, що для випадку інкапсуляції приховуваних даних у контурні зони LSB зображень, їх виявлення є практично неможливим. На цей випадок запропоновано підхід, спрямований на превентивну модифікації змісту LSB ймовірного контейнеру, у наслідок чого

вбудовані дані руйнуються без можливості реконструювання. При цьому, візуальна якість зображень не змінюється.

Отже, усі завдання до кваліфікаційної роботи опрацьовано у повному обсязі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Digital in 2021: World’s internet users pass the 4 billion mark [Електронний ресурс] – Режим доступу: https://wearesocial.com/blog/2018/01/global-digital-report-2021.

2. Статистика Интернета 2021: сайты, блоги, домены, электронная коммерция — интересные цифры и факты со всего мира [Електронний ресурс] – Режим доступу: https://sdvv.ru/articles/elektronnaya-kommertsiya/statistika-interneta-2021-sayty-blogi-domeny-elektronnaya-kommertsiya-interesnye-tsifry-i-fakty-so-v/.

3. Евдокимов Н., Лебединский И. Раскрутка веб-сайта. Практическое руководство/ Н. Евдокимов, И. Лебединский. – М. : Вильямс, 2011. – 288 с.

4. Севостьянов И. Поисковая оптимизация. Практическое руководство по продвижению сайта в Интернете. – Спб. : Питер, 2012. – 240 с.

5. Ашманов И. Методы Оптимизация и продвижение в поисковых системах. – Спб. : Питер, 2016. – 512 с.

6. Халворсон К., Рэч М. Контентная стратегия управления сайтом / К. Халворсон, М. Рэч. – Спб. : Питер, 2013. – 224 с.

7. Способы повышения производительности и безопасности сайта [Електронний ресурс] – Режим доступу: https://www.internet-technologies.ru/articles/povysheniya-proizvoditelnosti-i-bezopasnosti-sayta.html

8. K. Douglas van Duyne, A. James Landay, I. Jason Hong. The Design of Sites : Patterns for Creating Winning Web Sites, 2ed Douglas K. van Duyne. / R. Bergevin. – Crawfordsville: Prentice Hall, 2006/ - 1024 p.

9. Хокинс С. Администрирование Web-сервера Apache и руководство по электронной коммерции / С. Хокинс – Вильямс, 2001. – 336 с.

10. Joomla – Content Management System to build websites&apps. [Електронний ресурс] – Режим доступу: https://www.joomla.org/about-joomla.html

11. Blog Tool, Publishing Platform, and CMS WordPress. [Електронний ресурс] – Режим доступу: https://wordpress.org/

12. Sysbench: Scriptable database and system performance benchmark [Електронний ресурс] – Режим доступу: http://github.com/akopytov.

13. PHP Benchmark tool [Електронний ресурс] – Режим доступу: https://github.com/florinsky/af-php-bench

14. Brian Clifton. Advanced Web Metrics with Google Analytics, 3rd ed / Clifton Brian. – Диалектика, 2017. – 608 с.

15. Pagespeed insights [Електронний ресурс] – Режим доступу: https://developers.google.com/speed/pagespeed/insights/

16. Dotcom-Monitor: Website Monitoring & Performance Testing [Електронний ресурс] – Режим доступу: https://www.dotcom-monitor.com/

17. GTmetrix | Website Performance Testing and Monitoring [Електронний ресурс] – Режим доступу: https://www.gtmetrix.com

18. Microsoft security report [Електронний ресурс] – Режим доступу: https://microsoft.com/securityinsights.

19. Кібератаки – определение, виды, профилактика [Електронний ресурс] – Режим доступу: https://techarks.ru/category/security/ Кібератаки – определение, виды, профилактика.html.

20. WTF is APT? Продвинутые атаки, хитрости и методы защиты [Електронний ресурс] – Режим доступу: https://xakep.ru/2018/07/20/wtf-is-apt/.

21. Can we test ATP defenses even if we can’t agree on how to define APTs? [Електронний ресурс] – Режим доступу: https://news.sophos.com/en-us/2015/10/23/can-we-test-apt-defenses-even-if-we-cant-agree-on-how-to-define-apts.

22. Bejtlich R. The Practice of Network Security Monitoring: Understanding Incident Detection and Response / Richard Bejtlich. – San Francisco: Search Press Inc, 2013. – 341 с.

23. Shostack A. Threat Modeling: Designing for Security / Adam Shostack., 2014. – 624 с.

24. Шелухин О.И., Канаев С.Д. Стеганография. Алгоритмы и программная реализация. Горячая линия – Телеком, научно-техническое издательство 2017, 592 с.

25. Грибунин В. Г., Оков И. Н., Туринцев И. В. Цифровая стеганография. М.: СОЛОН-Пресс, 2016, - 315 с.

26. Конахович Г. Ф., Пузыренко А. Ю. Компьютерная стеганография. Теория и практика. - К.: МК-Пресс, 2006. - 288 с

27. Гизунов Д.С. Методика автоматизированного обнаружения скрытой информации в компьютерных файлах / Д.С. Гизунов, О.А. Демченко, Е.И. Никутин // Известия ТРТУ. – 2006. – Т. 71, № 16. – С. 49-53.

28. Fridrich Y. Steganography in Digital Media: Principles, Algorithms and Applicaticks. Cambridge Press, 2010. 462 p.

29. Provos N. Detecting steganographic content on the internet / N. Provos, P. Honeyman. // Technical Report CITI 01-1a, University of Michigan, 2001.

30. Кустов В.Н., Параскевопуло А.Ю. Простые тайны стегоанализа / В.Н. Кустов, А.Ю. Параскевопуло // Защита информации, INSIDE. – 2005. – № 4. – С. 72-78.

31. Юренский П.В. МЕТОДЫ СТАТИСТИЧЕСКОГО И НЕЙРОСЕТЕВОГО СТЕГОАНАЛИЗА СКРЫТЫХ КАНАЛОВ // Инновации в науке: научный журнал. – № 1(89). – Новосибирск., Изд. АНС «СибАК», 2019. – С. 11-13.

32. Голуб В.А. Комплексный подход для выявления стеганографического скрытия в JPEG-файлах / В.А. Голуб, М.А. Дрюченко // Инфокоммуникационные технологии. – 2009. – Т. 7, № 1. – С. 44-50

33. Dumitrescu, S., W. Xiaolin and Z. Wang, 2003. Detection of LSB steganography via sample pair analysis. In: LNCS, Vol. 2578, Springer-Verlag, New York, pp: 355-372.

34. Быков С. Ф. Алгоритм сжатия JPEG с позиции компьютерной стеганографии Защита информации. Конфидент. - СПб.: 2000, № 3.