ISSN 2519-2310

CS&CS Journal

KARAZIN UNIVERSITY
CLASSICS AHEAD OF TIME

1(21)2022

COMPUTER SCIENCE AND CYBERSECURITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА

ISSN 2519-2310

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені В.Н.КАРАЗІНА
V.N. KARAZIN KHARKIV NATIONAL UNIVERSITY

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

Issue 1(21) 2022

Заснований 2015 року

Міжнародний електронний науково-теоретичний журнал
International electronic scientific journal

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

The journal is published every six months.
Approved for placement on the Internet by Academic Council of the Karazin Kharkiv National University (September 26, 2022, Protocol No.15).

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

Editorial office:
V.N. Karazin Kharkiv National University
Svobody Sq., 6, office 315a, Kharkiv, 61022, Ukraine *(North building of University, 3th floor)*

E-mail: *cscsjournal@karazin.ua*
Web-page: *http://periodicals.karazin.ua/cscs (Open Journal System)*

Published articles have been internally and externally peer reviewed

В журналі публікуються наукові статті з теоретичних і науково-технічних проблем, що пов'язані зі створенням ефективних засобів комп'ютерних інформаційно-комунікаційних систем та питань захисту інформації, на основі передових математичних методів, інформаційних технологій і технічних засобів.

Журнал виходить кожні півроку.
Схвалено до розміщення в мережі Інтернет Вченою радою Харківського національного університету імені В.Н. Каразіна (26.09.2022 р., Протокол № 15).

ISSN (Онлайн): **10.26565/2519-2310**.

Редакція:
Харківський національний університет імені В.Н. Каразіна
пл. Свободи, 6, офіс 315а, Харків, 61022, Україна *(Північний корпус університету, 3 поверх)*

Електронна пошта: *cscsjournal@karazin.ua*
Веб-сторінка: *http://periodicals.karazin.ua/cscs (Open Journal System)*

Опубліковані статті пройшли внутрішнє та зовнішнє рецензування.

ЗМІСТ
TABLE OF CONTENTS

УДК 681.04 DOI: 10.26565/2519-2310-2022-1-01

ВИКОРИСТАННЯ НЕЙРОННОЇ МЕРЕЖІ ЗАМІСТЬ БАЗИ ЗНАНЬ У ЕКСПЕРТНІЙ СИСТЕМІ ДЕТЕКТОРУ ЗЛОВМИСНОГО ТРАФІКУ ДО ВЕБ-РЕСУРСІВ

Поліна Рогоза, Віталій Єсін

Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна
polina.rohoza@gmail.com, v.i.yesin@karazin.ua

Надійшла: червень 2022. Прийнята: липень 2022.

Анотація: *Сучасний світ інформаційних технологій надає нам широкий спектр веб-застосунків. Звісно, існує постійна необхідність у надійному захисті веб-ресурсів та конфіденційної інформації, яка на них зберігається. Зі збільшенням числа кібератак зростають також критичні наслідки від них не тільки для приватних осіб, але і для підприємств. В роботі розглянуто елементи експертної системи та здійснено оцінювання їх ефективності. Основна мета застосування експертної системи – підвищення захищеності веб-ресурсів від кібератак (типу SQLi, XSS, SSI, BufferOverflow тощо) шляхом забезпечення швидкої обізнаності фахівців інформаційної безпеки про наявність атаки. Нейронна мережа здатна детектувати та класифікувати зловмисний трафік веб-серверів. До переваг застосування нейронної мережі відносяться: ефективна побудова нелінійних залежностей, адаптація до змін та оцінювання атак "нульового дня", відмовостійкість, відносна простота реалізації, швидкість обчислення після навчання. Результатом роботи є розроблений елемент експертної системи – навчена та верифікована модель нейронної мережі, яка гарантує 98% успішності детектування кібератак на веб-ресурси, а також менше 5% виникнення помилок першого та другого роду у відповідності до використаного набору даних.*

Ключові слова: *експертна система; нейронна мережа; захист веб-додатків; кібератака.*

1. Вступ

Підвищення захищеності веб-ресурсів є однією з найважливіших сфер інформаційної безпеки (ІБ). Збільшення кількості веб-ресурсів неминуче веде за собою до збільшення числа кібератак, погіршуються наслідки від цих атак них не тільки для приватних осіб, але і для підприємств. Проблема постійного збільшення чисельності кібератак потребує своєчасного інформування фахівців ІБ про поточний стан кіберзагроз. На сучасному етапі технологічного розвитку є кілька підходів для вирішення зазначеної проблеми, зокрема за рахунок впровадження автоматичних інтелектуальних систем. Однак, нині засоби захисту, які засновані на штучних нейронних мережах, досі надають широкий потенціал для різних наукових випробувань, зокрема для захисту веб-ресурсів, і не є повноцінно дослідженими. Враховуючи те, що застосування технології нейронних мереж (НМ) надає багато практичних можливостей і переваг, безумовно, дана тематика є вкрай актуальною.

В межах вирішення зазначеної проблематики в даній роботі розглядається можливість використання експертної системи (ЕС), що побудована на основі технологій НМ. Дана ЕС дозволяє детектувати зловмисний трафік веб-серверів. Створення автономної ЕС дозволяє знизити навантаження на фахівців ІБ, підвищити швидкість аналізу трафіку та виключити «людський фактор». В цілому це дозволить зменшити втрати підприємств (економічні, інформаційні, репутаційні) та підвищити рівень захищеності веб-застосунків від кібератак.

2. Основні поняття та пов'язані роботи

Експертна система – це комп’ютерна система штучного інтелекту, призначена для вирішення складних проблем (*прогнозування, контролювання, управління тощо*) і надання можливості приймати такі рішення, які здатна прийняти і людина-експерт. ЕС виконує це, беручи знання зі своєї бази знань, використовуючи правила міркувань і висновків відповідно

до запитів користувачів. Продуктивність ЕС базується на знаннях експерта, які зберігаються в базі знань.

Експертна система в основному складається з трьох компонентів:

- інтерфейс користувача (*англ. User Interface*);
- машина логічного виведення (*англ. Interface Engine*);
- база знань (*англ. Knowledge Base*).

Нейронна мережа (модель) – це метод штучного інтелекту, у якому комп'ютерні системи обробляють, аналізують, класифікують дані подібно спрощеній нервовій системі мозку людини. НМ працює завдяки великої кількості взаємопов'язаних абстрактних нейронів. Вони є блоками обробки даних, розташованими пошарово у відповідності до конкретної архітектури. У класичному варіанті архітектури завжди присутні вхідний шар даних, прихований шар математичних перетворень даних та вихідний шар.

Застосування НМ здійснюється поетапно:

1. Постановка завдання – формування мети застосування НМ.
2. Навчання НМ – послідовний процес зміни синаптичних ваг у початковій моделі, що відображають силу збудження зв'язків між нейронами. Початкова модель НМ, як правило, виконує велику кількість контрольованих навчальних завдань, будуючи стратегію прийняття рішень з того набору даних, де правильна відповідь надається заздалегідь. За кожне таке завдання початкова модель налаштовує показники ваг зв'язків між нейронами, підвищуючи точність своїх прогнозів. Для налаштування модель порівнює початкові результати з наданою правильною відповіддю або цільовим показником та фіксує певні кореляції. Цей процес НМ повторює багато разів, прагнучи покращити прогнозуючу здатність при налагодженні моделі. У результаті відбувається еволюція початкової моделі у навчену НМ.
3. Експлуатація НМ – пред'явлення нейронній моделі деяких нових невідомих ситуацій, які або розпізнаються або ні.

Нейронні мережі мають наступні переваги:

- Універсальність. Велика частина НМ здатна аналізувати навіть недосконалі вхідні дані – неповні, недостатні, занадто комплексні, які включають велику кількість параметрів. На всю вихідну генерацію не впливає пошкодження одного чи кількох нейронів, що робить нейронні мережі водночас відмовостійкими.
- Відносна простота. Побудова НМ за допомогою сучасних програмних засобів не є складною та не займає багато часу. Окрім цього, розробнику не обов'язково повноцінно розуміти внутрішнє функціонування штучних нейронів.
- Вихідні дані або результати НМ не обмежуються кількістю вхідних даних.

Однак при використанні НМ необхідно враховувати і певні недоліки, властиві їм:

- Проблема вибору архітектури мережі. На відміну від необов'язковості знання про те, як функціонує НМ, розробнику необхідно успішно підібрати архітектуру шарів (*яких існує велика кількість*) для певного завдання. До того ж, стандартних архітектурних рішень може не існувати.
- Водночас з властивістю відносної простоти, існує також проблема інтерпретації результатів роботи, оскільки складні внутрішні механізми НМ залишаються «*чорним ящиком*». У ситуації, коли необхідно пояснити, на чому ґрунтуються передбачення моделі, часто це зробити неможливо.

Сучасні нейронні мережі продовжують відмінно впорюватись з проблемами класифікації, прогнозування, кодування і декодування інформації, а також мінімізують суб'єктивну та

упереджену складову оцінки, що задовольняє поточній задачі детектування (*прогнозування та класифікації*) кіберзагроз у *HTTP*-трафіку. Окрім цього, підходи машинного навчання до програм кібербезпеки пропонують розумну можливість ідентифікувати нові модифікації зловмисного програмного забезпечення та атак «нульового дня», що особливо важливо у стрімкому темпі розвитку нових технології.

Розглянемо більш детально, існуючи релевантні нейронні моделі (НМ) і методи для обраної предметної області (*кібербезпека*). Наприклад, в роботі [1] група авторів (*Корченко та ін.*) запропонувала підхід до виявлення кібератак та вразливостей інформаційно-телекомунікаційних систем, згідно з яким розпізнавання атак за допомогою НМ зводиться до оцінок величин параметрів безпеки ресурсів інформаційної системи. Так, якщо визначена за допомогою НМ оцінка перевищує певне граничне значення, то вважається, що кібератака виявлена. У протилежному випадку вважається, що рівень безпеки знаходиться в допустимих межах. В межах цієї роботи дослідниками були представлені, описані і проаналізовані (*в порівнянні один з одним*), наступні нейромережеві методи та моделі [1]:

1) Методи простої та семантичної класифікації мережевих атак;
2) Нейромережевий підхід виявлення SQL-ін’єкцій;
3) Бінарний нейромережевий метод;
4) Метод використання НМ гібридної структури типу *Counter Propagation*;
5) Адаптивна система виявлення мережевих атак;
6) Метод побудови сукупного класифікатора трафіку тощо.

У дослідженні [2] автори пропонують обчислювальну систему на основі інтелектуальних гібридних моделей, яка за допомогою нечітких правил дозволяє будувати експертні системи в домені класифікації кібернетичних вторгнень, зосереджуючись на атаці *SQL* Injection.

Автори роботи [3] зазначають, що останнім часом для виявлення кіберзагроз, таких як мережева атака, проникнення шкідливого програмного забезпечення або фішинговий вебсайт, використовувалося кілька моделей глибокого навчання. При цьому останні зазвичай страждають від того, що не можуть бути пояснені експертами з безпеки. Експертам з безпеки необхідно не тільки виявляти вхідні загрози, але й знати вбудовані функції, які спричиняють цей конкретний інцидент безпеки. Тому *Mahdavifar* та інші [3] пропонують свою експертну систему глибокої вбудованої нейронної мережі (*deep embedded neural network expert system, DeNNeS*), яка отримує уточнені правила з архітектури навченої глибокої нейронної мережі (*deep neural network, DNN*) для заміни бази знань експертної системи. Пізніше база знань використовується для класифікації невидимого інциденту безпеки та інформування кінцевого користувача про відповідне правило, яке зробило цей висновок.

Все це свідчить про зацікавленість наукової спільноти у розвитку нейронних мереж у домені ІБ та проведення додаткових досліджень для вирішення наявних проблем.

3. Аналіз рейтингів OWASP та CWE

Визначимо релевантні кібератаки на веб-застосунки згідно рейтингів *OWASP* та *CWE Top 10*. Рейтинг *OWASP Top 10* 2017-2021 років, представлений на рис. 1, з якого можна побачити основні тренди зміни загроз ІБ за минулі 4 роки [4].

Актуальний рейтинг найбільш небезпечних загроз *CWE* наведений у таблиці 1 [5].

Відзначимо, що нейронна мережа отримує на вхід, та зможе розпізнавати шкідливий трафік, тобто вхідні дані (запити) до веб-серверів. У даному випадку аналізу піддаються усі атаки типу ін’єкцій, які використовують вхідні данні у формі модифікованого *HTTP* або *URL*-запиту, які посилають до веб-серверу.

Отже, після розгляду рейтингів *OWASP Top 10* та *CWE Top 10*, визначимо набір актуальних та додаткових атак, які експертна система має бути здатна розпізнавати [6]:

- *SQL Injection;*
- *Cross Site Scripting (XSS);*
- *XML External Entities Injection (XEE);*
- *Server-Side Includes Injection (SSI);*
- *Buffer Overflow;*
- *Carriage Return Line Feed Injection (CRLFi);*
- *XPath;*
- *Lightweight Directory Access Protocol Injection (LDAPi);*
- *Format String*.

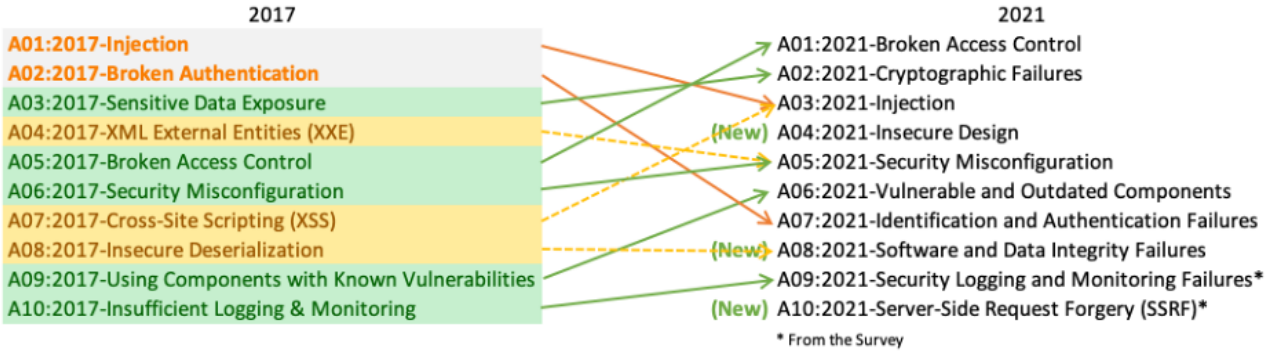


Рис. 1 – Рейтинг *OWASP Top 10 2021*

Таблиця 1 – Рейтинг загроз *CWE 2022*

№	Позначення	Назва	Оцінка
1	CWE-787	Запис даних поза меж (*Out-of-bounds*)	64.20
2	CWE-79	Неправильна нейтралізація вводу під час генерації веб-сторінок (Міжсайтовий скріптинг, *XSS*)	45.97
3	CWE-89	Неправильна нейтралізація спеціальних елементів при використанні команд SQL (Ін'єкція *SQL*)	22.11
4	CWE-20	Неправильна перевірка вводу	20.63
5	CWE-125	Читання даних поза меж (*Out-of-bounds*)	17.67
6	CWE-78	Неправильна нейтралізація спеціальних опцій при використанні команд ОС (Ін'єкція командного рядка ОС)	17.53
7	CWE-416	Використання динамічної пам'яті після її звільнення (*Use-After-Free*)	15.50
8	CWE-22	Неправильне обмеження шляху до каталогу з обмеженим доступом *(Path Traversal)*	14.08
9	CWE-352	Міжсайтова підробка запиту (*CSRF*)	11.53
10	CWE-434	Необмежене завантаження файлу небезпечного типу	9.56

4. Проектування та розробка нейронної мережі

При розробці НМ, перш за все, необхідно обрати придатну архітектуру моделі для задачі прогнозування і класифікації обраних класів кібератак. Існує багато відомих моделей класифікаторів, наприклад, метод k-найближчих сусідів (*K-NN)*, класифікатори *Naive Bayes, Random Forest, Decisions Tree*, НМ без нагляду (*вчителя*).

Зазначимо, що вхідний запит може або завдавати шкоди веб-ресурсу шляхом ін'єкції, або бути звичайним у рамках його трафіку. У цьому випадку класифікація містить результати «звичайний запит» (*normal*), або «зловмисний запит».

Проте, у даному випадку буде проводитись не бінарна, а багатокласова класифікація, адже, як показав аналіз рейтингів, ін'єкційні атаки можуть мати різні форми: деякі типи *SQL*-ін'єкцій, впровадження *XML, JSON* або *JS*-коду в запит, БД, *HTML*-код сторінки. Окрім цього, також виділимо клас *anomalous*, який відображає атаку «нульового дня». Для вирішення даної задачі класифікації пропонується застосувати згорткову нейронну мережу [6].

Згорткові нейронні мережі (ЗНМ, *англ. convolutional neural network, CNN,*) спочатку були розроблені для класифікації об'єктів зображень, а згодом вони успішно поширитись на обробку природної мови та текстових даних.

Дослідження наукової області розробок допомагають узагальнити процес побудови багатошарової нейронної мережі та визначити низку таких необхідних етапів [7].

1) Визначити вектор вхідних даних. Він повинен містити повністю всю достатню та необхідну для подальшого прогнозування інформацію.
2) У повній мірі визначити всі складові вихідного вектору, необхідні для повноцінної відповіді для поставленої задачі.
3) Вибрати оптимальну з точки зору завдання функцію активації нейронів.
4) Визначити архітектуру: тип та кількість шарів, і число нейронів у кожному з них.
5) Присвоїти початкові значення ваг і порогових рівнів. Для збереження прийнятної швидкості навчання, значення не повинні бути як великими, так і малими.
6) Провести навчання найкращим можливим чином, тобто вдало підібрати функцію втрат, за необхідності – оптимізатор, кількість епох навчання. Це налаштування дозволить уникнути повільного навчання і, у подальшому, перенавчання. У результаті НМ зможе вирішувати подібні невідомі завдання.
7) Перевірити успішність функціонування мережі шляхом подання на її вхід задачі у вигляді знайомого вхідного вектору. Далі оцінити наданий моделлю результат рішення у вигляді вихідного вектору на предмет справжності та дійсності.

Для програмної розробки нейронної мережі використовуються інструменти: *Jupyter Notebook* (*IDE з відкритим кодом для інтерактивної розробки та представлення наукових проектів*), мова програмування *Python 3.8*, бібліотеки *TensolFlow, Keras, sklearn, pandas*.

У цьому дослідженні навчання моделі проводиться на публічному наборі даних *CSIC 2012 Dataset*. Він був штучно сформований за допомогою фреймворку *Torpeda* [8] для веб-сервісу електронної комерції та навмисно містить суттєву кількість зловмисного трафіку для тестування засобів захисту.

CSIC 2012 Dataset містить усього 74 133 запитів, з яких: 49 311 є зловмисними, з різним типом ін'єкцій, 16 459 є аномальними (клас «*anomalous*» у програмній реалізації) та 8 363 є звичайними (клас «*normal*» у програмній реалізації). Кількість запитів для ін'єкційних класів наступна [6]:

43 013 – різновиди *SQL Injection*;
4818 – різновиди *XSS*;
451 – *SSI* (*Server-Side Includes Injection*);
412 – *Buffer Overflow*;
327 – CRLF (*Carriage Return Line Feed*);
175 – *XPath*;
74 – *LDAPi* (*Lightweight Directory Access Protocol Injection*);
41 – *Format String*.

На рис. 2 можна побачити програмну реалізацію поділу набору даних за класами.

Dataset був об'єднаний у *Excel* таблицю *(тип файлу csv)*, кожен рядок якої підлягає виразу [6]:

- file (*назва вихідного файлу*), id (*номер запиту у певному класі*);
- label (відмітка класу);
- method (*методи HTTP-протоколу – GET або POST*);
- path (*частина URL, шлях до сторінки, яка відображається клієнту*);
- query (*набір параметрів, що передаються у запиті*);
- url (*повне посилання – path + query*).

Також *dataset* необхідно поділити на дві окремі частини у відношенні: данні для навчання моделі (70%) і данні для її тестування (30%).

```
df.to_csv('./all.csv')
```

```
df['label'].value_counts()
```

```
SQLi              43013
anomalous         16459
normal             8363
XSS                4818
SSI                 451
BufferOverflow      412
CRLFi               327
XPath               175
LDAPi                74
FormatString         41
Name: label, dtype: int64
```

Рис. 2 – Класифікація запитів набора даних

Наступним кроком буде проведення маніпуляцій з частиною набору даних для навчання, які носять назву обробки природної мови (*Natural Language Processing, NLP*). Мета попередньої обробки (*preprocessing*) полягає у переведенні текстових символів у формат, з яким у подальшому буде працювати НМ. Клас ***keras.preprocessing.text.Tokenizer*** є реалізацією методів векторизації текстових даних, тобто методів репрезентації тексту у числовому поданні. Таким чином, слова або фрази з кожного вхідного текстового значення відображаються у відповідний вектор дійсних чисел зі спільного для усіх запитів словника (*рис. 3*). Модель застосовує поточний вектор для пошуку передбачень на основі його подібності до інших векторів з відомим результатом.

```
# Character-level word segmentation, fitting on the training set
tokenizer = Tokenizer(filters='\t\n', char_level=True)
tokenizer.fit_on_texts(url_train)
# Build a dictionary and save
num_words = len(tokenizer.word_index)+1
vocab = tokenizer.word_index
print("The size of the dictionary is %d" % num_words)
print("dictionary: ")
print(vocab)
with open("./tokenizer/vocab.json", 'w') as f:
    json.dump(vocab, f, ensure_ascii=False)
```

```
The size of the dictionary is 72
dictionary:
{'%': 1, '2': 2, 'c': 3, '0': 4, 'i': 5, 'e': 6, 'r': 7, 'o': 8, '3': 9, 'a': 1
0, 'n': 11, '=': 12, 'l': 13, 'm': 14, '&': 15, 'd': 16, '1': 17, '7': 18, 't':
19, 's': 20, 'p': 21, '5': 22, '6': 23, 'u': 24, '9': 25, '8': 26, 'b': 27,
'/': 28, ',': 29, '4': 30, 'g': 31, '.': 32, 'f': 33, 'j': 34, 'w': 35, ' ': 3
6, '?': 37, 'v': 38, 'h': 39, 'z': 40, 'x': 41, '+': 42, 'y': 43, '-': 44, 'k':
45, '#': 46, 'q': 47, ';': 48, '<': 49, '>': 50, '"': 51, '_': 52, '@': 53,
'*': 54, ':': 55, '(': 56, ')': 57, "'": 58, '!': 59, '[': 60, ']': 61, '{': 6
2, '}': 63, '`': 64, '\r': 65, '$': 66, '~': 67, '|': 68, '\\': 69, '^': 70,
'\n': 71}
```

Рис. 3 – Словник для векторизації запитів

Архітектура кожної згорткової нейронної мережі обов'язково включає в себе наступні структурні шари у відповідній послідовності: згорткові шари (англ. *Convolutional layer*), пулінгові шари (англ. *Pooling layer*) і повнозв'язні шари (англ. *Fully-connected layer*). Дана програмна реалізація заснована на Sequential-моделі модулю *Keras*, яка містить у собі усі необ-

хідні шари з певною конфігурацією (*див. рис. 4*). Нижче наведений короткий опис сенсу введення кожного окремого шару моделі [6].

Шар *Embedded* можна представити у вигляді простого матричного множення, яке перетворює натуральні числа (індекси) на щільні вектори фіксованого розміру. *Conv1D* – це шар, який створює згорткове ядро за одним просторовим (або часовим) вимірюванням. Шар *Pooling* (Об'єднання) зменшує кількість параметрів, для подальшого навчання та, відповідно, обсяг обчислень мережі. Рівень об'єднання зберігає тільки найбільш суттєві ознаки, згенеровані попередньо шаром згортки.

```
# Build a network
model = Sequential()
model.add(layers.Embedding(num_words, 64, input_length=max_len))
model.add(layers.Conv1D(32, 7, activation='relu'))
model.add(layers.MaxPooling1D(5))
model.add(layers.Conv1D(32, 7, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dense(10, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

Рис. 4 – Словник для векторизації запитів

Щільний шар (*Dense*) є повністю пов'язаним із попереднім шаром. Нейрони щільного шару здійснюють матрично-векторне множення. *Dense* вводиться наприкінці, для зміни розмірності виходу (у даному випадку для 10 класів кінцева розмірність шару має бути 10) та реалізує операцію *output = activation (dot (input, kernel) + bias).*

Функція активації (*activation*) за певним правилом визначає, чи є внесок окремого нейрону в мережу значним і чи потрібно його зберегти надалі.

Як можна побачити на рис. 4, були застосовані дві найпопулярніші функцій активації.

Rectified Linear Unit (*ReLU*) – це нелінійна функція активації, яка повертає 0 у разі негативного аргументу, а при позитивному аргументі – його чисельне значення залишається незмінним на виході функції (*див. рис. 5*) [9].

Функція *Softmax* – це узагальнення логістичної функції для багатовимірного випадку. Вона застосовується для задач класифікації, коли кількість можливих класів більше двох.

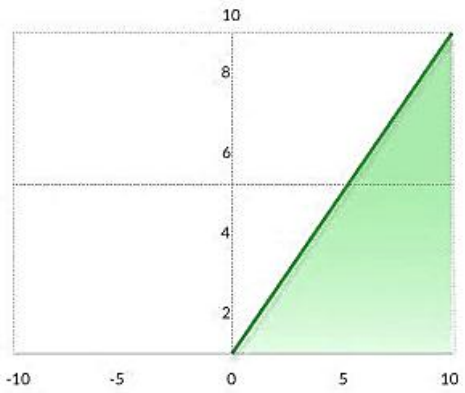


Рис. 5 – Графік функцій *ReLU*

Функція втрат – це міра того, наскільки добре модель прогнозування передбачає очікуваний результат (*або значення*). Бажано звести її до мінімуму. Для задач машинного навчання в переважній більшості випадків використовується крос-ентропія (*тобто, перехресна ентропія*), яка регулює ваги нейронів НМ під час тренування [10].

Оптимізатор – це алгоритм, який вирішує поширену проблему надто повільного навчання моделі. Він дозволяє зменшити час та скоригувати якість навчання моделі також шляхом певної незначної зміни ваг нейронів. Пропонується використовувати оптимізатор *Adam* – це один з найбільш ефективних алгоритмів оптимізації, який можна використовувати замість класичної процедури стохастичного градієнтного спуску для ітеративного поновлення ваг мережі на основі навчальних даних [11].

На рис. 6 представлений програмний звіт про спроектовану архітектуру моделі, кількість нейронів на кожному шарі та кількість вхідних параметрів. Навчання моделі відбувалося на ноутбуку Asus з процесором Intel *Core i5-6198DU CPU 2.3GHz* [6]. Процес складався з шістьох послідовних ітерацій (*тобто епох*), що емпірично визначено як оптимальна кількість, щоб уникнути як проблем недостатнього навчання, так і перенавчання. Кожна епоха тривала приблизно 5 сек. та наприкінці включала перевірку якості на контрольній множині. Завдяки цьому, у кожній наступній ітерації навчання відбувається на відкоригованих вагах і, як наслідок, зменшується величина функції втрат та підвищується точність передбачення. Після навчання НМ помилки детектування кібератак становлять 1,38% і точність – 99,56%.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 600, 64)           4608
_________________________________________________________________
conv1d (Conv1D)              (None, 594, 32)           14368
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 118, 32)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 112, 32)           7200
_________________________________________________________________
global_max_pooling1d (Global (None, 32)                0
_________________________________________________________________
dense (Dense)                (None, 10)                330
=================================================================
Total params: 26,506
Trainable params: 26,506
Non-trainable params: 0
_________________________________________________________________
```

Рис. 6 – Звіт про архітектуру моделі

5. Верифікація нейронної моделі

Для тестування отриманих показників точності НМ використаємо другу частину попередньо розділеного набору даних. Підготовка до верифікації включала заходи: – усі запити були розміщені у файлі *url_test.txt*, а відомі відповідні до них мітки класів – у файлі *label_test.txt*; – на вхід нейронної моделі подавався *url_test.txt*, а результат детектування кібератак (моделлю) ставився у відповідність до *label_test.txt*, тобто НМ не мала доступу до дійсних міток класу на цей раз; – вручну проводилось порівняння фактичних показників із передбаченими. Результат верифікації показує, що нейронна мережа стовідсотково впоралась з передбаченням перших десяти запитів (*див. рис. 7*).

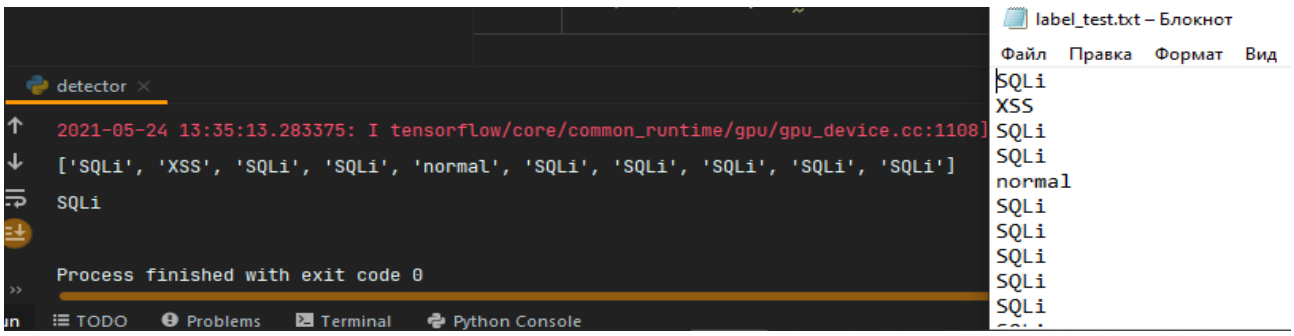


Рис. 7 – Результат виконання програми

В галузі машинного навчання, й зокрема для проблеми класифікації, матриця невідповідностей (англ. *confusion matrix*), або матриця помилок (англ. *error matrix*), - це метод підсумовування продуктивності алгоритму класифікації. Дана матриця відображає порівняне про-

гнозоване значення з фактичним значенням (рис. 8). Кожен рядок цієї таблиці є екземпляром фактичного класу змінної, тоді як кожен стовбець, є екземпляром прогнозованого класу.

З матриці помилок видно, що НМ є дійсно продуктивною та помилки першого та другого роду майже для усіх класів не виникають (*перетин стовпця та рядка дорівнює 0*).

На рис. 9 представлені метрики для кожного класу атак зокрема і загальна точність моделі. Як висновок, у контексті дослідження на наборі даних *Torpeda*, модель *CNN* є достатньо ефективною та детектує переважно всі ін'єкції у *HTTP*-запитах.

```
[[12903     0     0     0     0     0     0     1     0     0]
 [    3  4919    16     0     0     0     0     0     0     0]
 [    0    17  2492     0     0     0     0     0     0     0]
 [    6     0     0  1437     0     0     0     1     0     1]
 [    0     6     3     0   125     0     0     0     0     1]
 [    0     0     0     0     2   122     0     0     0     0]
 [    0     0     0     0     0     0    98     0     0     0]
 [   31     0     0     2     4     0     0    15     0     1]
 [    0     0     0     0     0     0     0     0    21     1]
 [    0     1     0     0     0     0     0     0     0    11]]
```

Рис. 8 – Матриця невідповідностей (*помилок*)

	precision	recall	f1-score	support
SQLi	1.00	1.00	1.00	12904
anomalous	1.00	1.00	1.00	4938
normal	0.99	0.99	0.99	2509
XSS	1.00	0.99	1.00	1445
SSI	0.95	0.93	0.94	135
BufferOverflow	1.00	0.98	0.99	124
CRLFi	1.00	1.00	1.00	98
XPath	0.88	0.28	0.43	53
LDAPi	1.00	0.95	0.98	22
FormatString	0.73	0.92	0.81	12
accuracy			1.00	22240
macro avg	0.96	0.90	0.91	22240
weighted avg	1.00	1.00	1.00	22240

Рис. 9 – Метрики якості НМ

5. Висновки

1. У запропонованій роботі розглянуті релевантні загрози ІБ у рейтингах *OWASP* та *CWE*, і на основі їх аналізу був сформований перелік атак типу ін'єкцій для детектування.

2. В результаті проведення досліджень:

– продемонстровано, що застосована архітектура згорткової нейронної мережі є доцільною і оптимальною для розподілу HTTP-запитів за класами *anomalous, normal, SQLi, XSS, SSI, BufferOverflow, CRLFi, XPath, LDAPi, FormatString*;

– проведено успішне навчання моделі на наборі даних із 74 133 унікальних запитів;

– верифікація показала, що *CNN* гарантує 98% успішності детектування кібератак на веб-ресурси, а також менше 5% виникнення помилок першого та другого роду у відповідності до використаного набору даних *Torpeda*.

3. Спроектована модель НМ довела свою продуктивність та доцільність. Враховуючи такі переваги, як простота, швидкість побудови, незначне використання технічних ресурсів, вона може бути застосована при плануванні та реалізації відповідної експертної системи.

4. В умовах реального мережевого трафіку точність НМ буде нижче через можливу первинну непристосованість. Тому у подальшому для відстеження еволюції у перенавчанні та вдосконалення використовуваної моделі, доцільно реалізувати відповідну базу даних.

Список літератури

[1] Корченко, О. Г., Терейковський, І. А., Дзюбаненко, А. В. (2014). Сучасні нейромережеві методи та моделі оцінки параметрів безпеки ресурсів інформаційної системи. Вилучено із https://doi.org/10.18372/2410-7840.16.7539

[2] Batista, L. O., de Silva, G. A., Araujo, V. S., Araujo, V. J. S., Rezende, T. S., Guimarães, A. J., Souza, P. V. D. C. (2019). Fuzzy neural networks to create an expert system for detecting attacks by sql injection. Вилучено із https://doi.org/10.48550/arXiv.1901.02868

[3] Mahdavifar, S., Ghorbani, A. A. (2020). DeNNeS: deep embedded neural network expert system for detecting cyber attacks. Neural Computing and Applications. Вилучено із https://doi.org/10.1007/s00521-020-04830-w

[4] OWASP Top 10 Application Security Risks. (2021). Вилучено із https://owasp.org/Top10/

[5] Common Weakness Enumeration (CWE) Top 25 Most Dangerous Software Weaknesses. (2022). Вилучено із https://cwe.mitre.org/top25/archive/2022/2022_cwe_top25.html

[6] Рогоза П. В. Оцінка захисту web-ресурсів від кібератак. Пояснювальна записка до дипломної роботи бакалавра: напрям підготовки 125 – Кібербезпека / П. В. Рогоза; Харківський національний університет імені В. Н. Каразіна. – Харків: [Б. В.], 2021. – 67 с.

[7] Дунець, Р. Б., Рак, Ю. П., & Зачко, О. Б. (2008). Класифікація територій засобами нейронних мереж для управління проектами в забезпеченні екологічної безпеки. https://sci.ldubgd.edu.ua/jspui/handle/123456789/2505

[8] Torrano C., Perez A., Alvarez G. (2022). What is Torpeda. Вилучено із https://www.tic.itefi.csic.es/torpeda/default.html

[9] Соснин А. С., Суслова И. А. (2019). Функции активации нейросети: сигмоида, линейная, ступенчатая, RELU, TAHN. Екатеринбург: РГППУ.

[10] Гафаров Ф. М., Галимьянов А. Ф. (2018). Искусственные нейронные сети и их приложения. Уч. руководство. Казань: Издательство Казанского университета.

[11] Brownlee J. (2017). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning: Deep Learning Performance. Вилучено із https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Received: on June 2022. Accepted: on July 2022.

Authors:

Vitalii Yesin, Doctor of Engineering Sciences, Professor, Department of Security of Information Systems and Technologies, V.N. Karazin National University, Kharkiv, Ukraine.

ORCID ID https://orcid.org/0000-0003-1977-7269

E-mail: v.i.yesin@karazin.ua

Polina Rohoza, CSD Student (magistrate), Department of Security of Information Systems and Technologies, V.N. Karazin National University, Kharkiv, Ukraine.

E-mail: polina.rohoza@gmail.com

Using a neural network instead of the knowledge base in the expert system of web resources malicious traffic detector.

Abstract. The modern world of information technology provides us with a wide range of web applications. Indeed, there is a constant need for solid protection of web resources and their confidential information. As the number of cyber-attacks increases, so do their critical consequences for organizations and individuals. This work developed the elements of the expert system and evaluated their effectiveness. The main purpose of using an expert system is to increase the protection of web resources against cyberattacks (such as *SQLi, XSS, SSI, BufferOverflow*, etc.) by ensuring that information security specialists are quickly aware of the attack presence. The neural network is capable of detecting and classifying malicious web server traffic. The advantages of using a neural network include: effective construction of non-linear dependencies, adaptation to changes and evaluation of "*zero-day*" attacks, fault tolerance, relative simplicity of implementation, calculation speed after training. The result of the work is a developed element of the expert system – a trained and verified neural network model that guarantees 98% success in detecting cyberattacks on web resources, as well as errors types I and II in the neural model do not exceed 5%.

Keywords*:* Expert System; Web Application Protection; Cyber-Attack; Neural Network.

UDC 004.415.53 DOI: 10.26565/2519-2310-2022-1-02

A CONCISE OVERVIEW OF THE SPECIFIC FEATURES OF USING EXPLOYTS

Bogdanova Elizaveta, Pavlova Larysa, Pohorila Karina

V.N. Karazin National University, Kharkiv, 61022, Ukraine
xa12850323@student.karazin.ua, l.v.pavlova@karazin.ua, karina.pogorelka@gmail.com

Received: on June 2022. Accepted: on July 2022.

Abstract: *The issue of exploiting the software vulnerabilities is considered in the article. Particular attention has been paid to the two aspects of the practical usage of exploits, as an attack tool and as a means of testing protected information systems. It is stressed that integrating exploits into a single exploit-kit, increases the efficiency of searching for existing vulnerabilities of the modern information systems. The scheme of the exploit kit operation in the target information system is presented. Analysis of the known incidents related to the use of exploits, al-lows us to assert the existence of a relationship between the degree of popularity of a software product or device, and the probability of the exploits being created. The extreme importance of the timely release of security patches as an effective means of preventing the usage of identified software vulnerabilities is emphasized. Releasing security patches is a basic element of possible defensive reactions when dealing with such issues.*

Keywords*: exploits; software vulnerabilities; security patches; information security.*

1. Introduction

Over the past 20 years the exploits have remained one of the most serious problems in the field of information security [1]. Moreover, this problem equally concerns software developers and developers of security solutions, as well as employees of information security departments of companies and organizations. At the same time, among specialists there is ambiguity in determining the actual role and place of exploits, especially in deciding whether an exploit can independently harm an information system (ISs) and/or its information resources, or whether it should be considered as a tool (means) for penetrating and then launching another malicious code.

Taking into account the rapid informatization of all spheres of modern society and its critical dependence on the implementation of network technologies, providing information security for modern ISs and online services is of an absolute priority. This is due to the constant evolution of tools, techniques and technologies used by attackers in the course of searching for and exploiting various software vulnerabilities and security settings of ISs software and hardware. Exploits are one of the tools used to overcome the security measures of attacked ISs. They were most popular over the past decade, but are still considered by attackers as an effective «precision» tool for penetrating modern ISs *(Wanna Cryptor virus attack).* At the same time, exploits are becoming more sophisticated, taking into account the peculiarities of both the attacked resource and its security system.

2. Main part

Software is created by humans, and it is known that «*errare humanum est»*. As a result, even the rigorous testing of newly developed software does not eliminate the possibility of vulnerabilities in a created program code. Moreover, one cannot exclude the threat of deliberate introduction of certain vulnerabilities into created software products. The motivation for introducing vulnerabilities can vary widely from the insider's desire to cause reputational damage to the company to the pursuit of personal gain by trying to peddle those vulnerabilities on special online resources. In any case a user exploiting an existing vulnerability (*including a deliberate attacker*) is potentially able to control the target IP and/or gain unauthorized access to sensitive data.

Using an existing vulnerability of a system program, an application or an online service, the exploit allows for subsequent unauthorized actions [2]. At the same time, the direct destructive im-

pact on the compromised device and/or the target ISs and data is typically carried out by other malicious programs at the later stages of the attack. For example, having gained access to an ISs, an attacker can firstly modify their access rights, and then change the operation parameters of network devices and/or install their own software, which, in any case, should be considered as unauthorized interference with the operation of the system (*device*) [3].

Generally, exploits being a subspecies of malware can be not only a separate application, but a small fragment of program code or a set of commands as well [4]. In addition, for the purposes of penetration testing, exploits can be grouped into the set, a so-called exploit kit. Such kits are effective for preliminary monitoring of devices in the target ISs selected for attacking (*a gateway between the wireless and wired segments of the victim's corporate network, for example*), or monitoring of the entire ISs for various vulnerabilities in the used software. Such network reconnaissance being "successful" [5], the attacker uploads the main "body" of the malicious code (*keylogger*) and starts its phased deployment on a compromised resource.

Theoretically, even regular updating of existing software does not guarantee complete protection against exploits, because there is always a certain time interval between the discovery of another vulnerability and its elimination (*the release of the so-called software patch*). During this time gap potential intruders can use the so-called zero-day vulnerabilities (*exploits)* [3, 6], which exist due to the fact that software developers in their attempts to eliminate the vulnerabilities of the program code are always "*catching*" up with relatively new techniques and methods of attacking existing software and hardware solutions. Moreover, the strategy chosen by the developer of the compromised software to handle discovered vulnerabilities is extremely important. A responsible developer will not try to hide the discovered vulnerability, but promptly inform the users about the identified threat and form appropriate recommendations for its temporary containment, while expediting the measures for eliminating this problem.

It is important to emphasize that any elements of modern ISs such as application programs, modules of special software, operating system modules (OS) or hardware can be attacked by means of exploits [4]. Since the existence of different exploits implies the unauthorized actions on various compromised objects (*a single PC, ISs, local network device or online service*), it is logical to classify exploits according to the objects being attacked (*targets*) (Fig. 1).

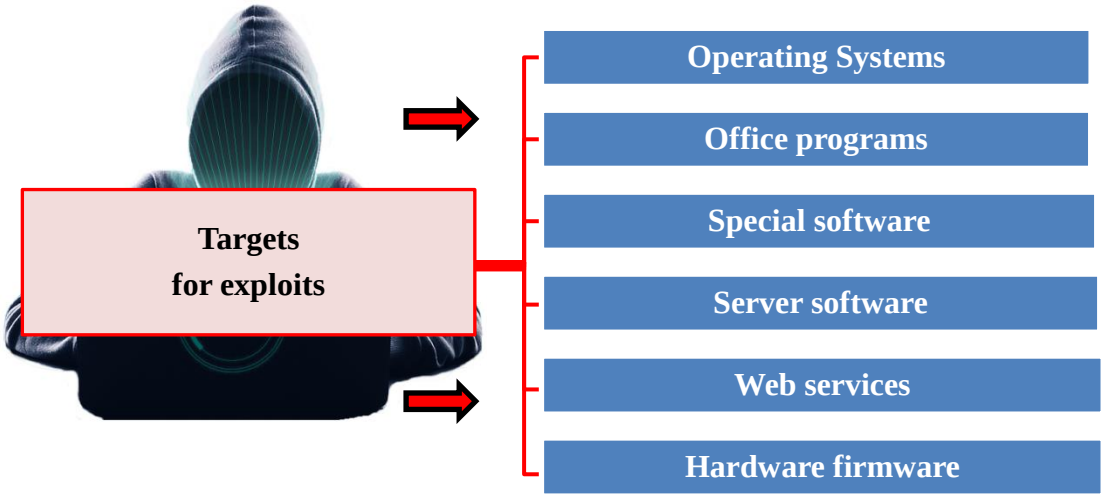


Fig. 1 - Classification of exploits by targets

Analysis of the known incidents related to the use of exploits (*Eternal Blue, for example*), allows us to assert the existence of a relationship between the degree of popularity (*number of uses*) of a software product or device, and the probability of the exploits being created. That is, the wider

the client base of the target being attacked and the higher the expectations for the monetization of a "successful" attack, the higher the probability of creating a corresponding exploit.

There are various ways for exploits to penetrate the target ISs [6-7]: - through an "infected" website, a link in an e-mail, through local access to the system (*see Fig 2*).

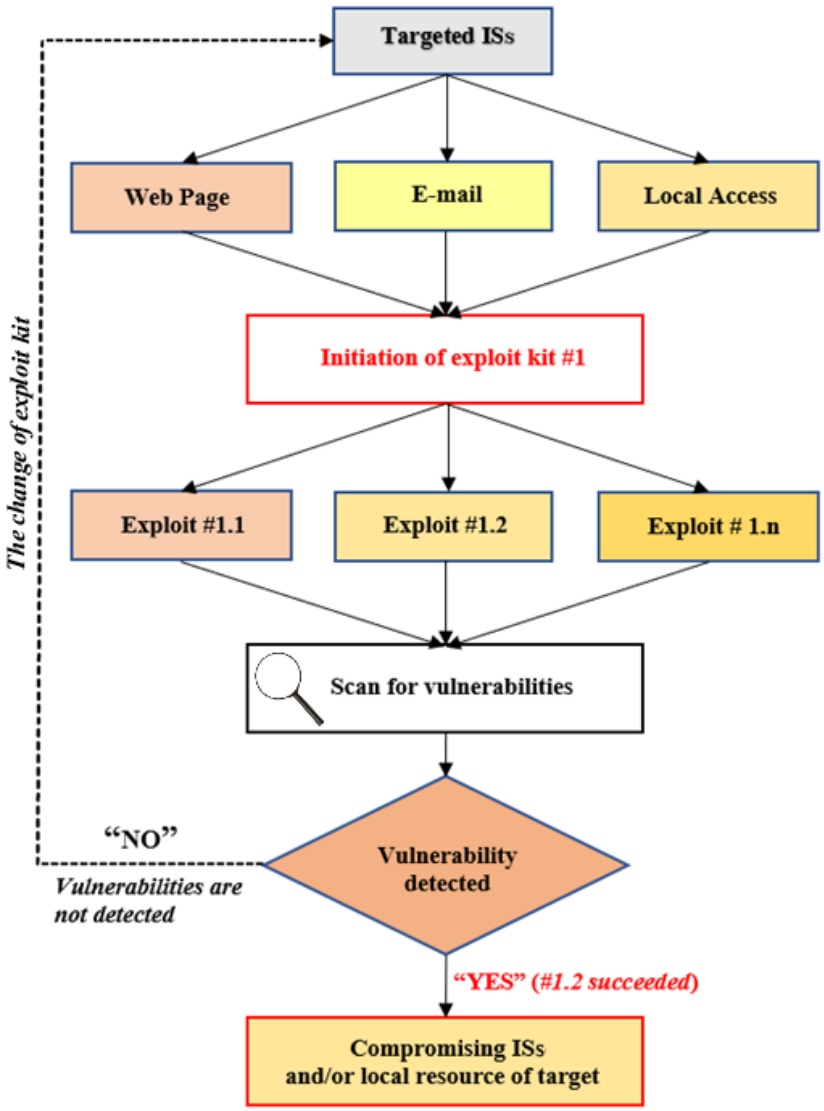


Fig. 2 - The operating principle of exploit kit in the target information system

However, in order for the exploit to start scanning the target ISs or local device for vulnerabilities, the user of the system needs to run the corresponding malicious code (*regardless of the way it has got into the attacked system, see Fig. 2*). Therefore, a local ISs node *(server or computer)*, network hardware (*switch or gateway*), a software utility and/or a process within the attacked IS (*including «cloud» ones*) can be a local/final resource to be attacked. However, it should be noted, that the exploit kit can also be launched from outside. This option is possible within the framework of network reconnaissance of the external perimeter of the attacked system [5]. In this case, the «success» of the exploit kit activation is determined by the presence of gross errors in the functional parameters of the elements of the corporate information security system (*for example, the input (first) firewall or proxy server*) [8].

Obviously, after unauthorized penetration of the ISs, the «*attacker*» acts according to his goals and motives. In other words, usage of exploits can be harmful, i.e., to carry out an attack on the information resources of the victim, as well as beneficial. For example, information security

specialists test their corporate ISs specifically to identify vulnerabilities (the so-called *Pen test - Penetration Testing*), thereby trying to identify existing security problems in advance and determine possible vectors of cyber-attacks on their information resources [9].

Based on the results of external penetration tests and internal activations of exploit kits, information security specialists get an idea of the presence of vulnerabilities in the systems they maintain and creates appropriate exploits for inform developers about vulnerabilities in their product *(including the results of an external information security audit)*. This gives them the opportunity to provide a security patch in advance, before this exploit is released to the public domain. However, there are the certain exceptions, for example, the practice of using exploits as a cyber weapon by special services providing the information security for their states.

3. Conclusions

1. In the most of information security incidents, the use of exploits provides the necessary conditions for the subsequent unauthorized actions in the infrastructure of the attacked target.

2. The wider the client base of the target being attacked and the higher the expectations for the monetization of a *«successful»* attack, the higher the probability of creating a corresponding exploit.

3. The most dangerous are *«zero-day»* exploits, which exclude the time limit of the security services and developers of compromised software from reacting in time. That is why these exploits are the main means of implementing covert attacks and are in steady demand among a certain category of the Internet community.

4. The measures that can potentially mitigate the consequences of attacks which use zero-day vulnerabilities are the usage of layer *NGFW*, *Honeypot* and *IDS* proactive protection tools integrated within a single *XDR*-platform and subjected to regular updates and internal penetration tests (*vulnerability scanners*).

5. Integrating exploits into a single exploit kit increases the efficiency of searching for existing vulnerabilities, which makes it possible to identify several points (*directions*) of penetration into the target infrastructure at once.

6. The practical use of exploits develops along two main vectors, as a means of providing attacks on a compromised resource (*i.e., having a confirmed vulnerability*), and as a means of testing protected infrastructure objects for vulnerabilities (*pentesting*).

7. The main reasons for the emergence of software vulnerabilities should be considered as follows: - insufficient personnel qualification; - failures in following the proper stages of developing and testing of software; - excessive use of IT-outsourcing opportunities; - presence of a corporate insider; - deliberate introduction of hidden functions software and/or logical triggers, as part of the interaction between software manufacturers with representatives of national special services in charge of cybersecurity.

References

[1] Синцов А. (2015). Куда катится безопасность? Хакер, (192), 58-59. Извлечено из: http://surl.li/certn

[2] Мелкозьорова, О., Лєсная, Ю., & Малахов, С. (2022). Особливості забезпечення захисту від НСД в сучасних інформаційних системах. *InterConf,* (97). Retrieved from https://ojs.ukrlogos.in.ua/index.php/interconf/article/view/18428

[3] Касперский Е.В. (2012). Эксплойты, зеродеи, их опасность и её профилактика. Retrieved from: https://eugene.kaspersky.ru/2012/05/25/exploits-and-zerodays-protection/

[4] (2017). Эксплойты, (Exploits). Извлечено из https://www.antimalware.ru/threats/exploits

[5] Рузудженк, С., Погоріла, К., Кохановська, Т., & Малахов, С. (2020). Особливості захисту корпоративних ресурсів за допомогою технології Honeypot. Комп'ютерні науки та кібербезпека, (4), 22-29. Retrieved from: https://doi.org/10.26565/2519-2310-2019-4-03

[6] Daniel Simpson. (2022). Наборы эксплойтов и эксплойтов. Retrieved from: http://surl.li/certr

[7] Закрожевский В. (2010). Лазутчики киберкриминала. Retrieved from: https://securelist.ru/lazutchiki-kiberkriminala/1424/

[8] Джон Маллери, & Джейсон Занн (2007). *Безопасная сеть вашей компании.* (Е. Линдеманн, пер. с англ.). Москва: НТ Пресс.

[9] Nikto: A Practical Website Vulnerability Scanner. (2021). Retrieved from: https://securitytrails.com/blog/nikto-website-vulnerability-scanner

Надійшла: червень 2022. Прийнята: липень 2022.

Автори:
Єлизавета Богданова, студентка факультету комп'ютерних наук (бакалавріат), Харківський національний університет імені В.Н. Каразіна, Україна.
E-mail: xa12850323@student.karazin.ua
Лариса Павлова, ст. викладач кафедри іноземних мов професійного спрямування факультету іноземних мов, Харківський національний університет імені В.Н. Каразіна, Україна.
ORCID ID https://orcid.org/0000-0002-5854-4209
E-mail: l.v.pavlova@karazin.ua
Каріна Погоріла, студентка факультету комп'ютерних наук (магістрат), Харківський національний університет імені В.Н. Каразіна, Україна.
ORCID ID https://orcid.org/0000-0001-8701-2394
E-mail: karina.pogorelka@gmail.com

Короткий огляд специфіки використання експлойтів.

Анотація. Розглянуто проблематику експлуатації вразливостей програмного забезпечення. Звернено увагу на існування двох іпостасей практичного застосування експлойтів: - як інструменту атаки та, як засобу тестування інформаційних систем, що потребують захисту. Наголошується, що об'єднання експлойтів в єдиний експлойт-кіт підвищує ефективність пошуку наявних вразливостей сучасних інформаційних систем. Аналіз відомих інцидентів, пов'язаних з використанням експлойтів, дозволяє стверджувати про існування зв'язку між ступенем популярності програмного продукту або пристрою та ймовірністю створення відповідних експлойтів. Представлено схему роботи експлойт-паку в цільовій інформаційній системі. Підкреслено надзвичайну важливість своєчасного випуску патчів безпеки, як ефективного засобу парірування виявлених уразливостей програмного забезпечення. Звернено увагу на те, що процес випуску патчів безпеки є базовою складовою у спектрі можливих захисних реакцій, при вирішенні подібних проблем.

Ключові слова: експлойт; програмна вразливість; патч безпеки; інформаційна безпека.

УДК 004.056.55 DOI: 10.26565/2519-2310-2022-1-03

ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ АЛГОРИТМІВ МАЛОРЕСУРСНОЇ КРИПТОГРАФІЇ У ДЕЦЕНТРАЛІЗОВАНИХ СЕРЕДОВИЩАХ

Євгеній Деменко, Олексій Нарєжній

Харківський національний університет імені В.Н. Каразіна, Харків, 61022, Україна
xa11868404@student.karazin.ua, o.nariezhnii@karazin.ua

Надійшла: липень 2022. Прийнята: липень 2022.

Анотація: *Метою даного матеріалу є ознайомлення з областю досліджень застосування малоресурсних алгоритмів криптографії для систем Інтернету речей (IoT) та можливості прямого впровадження в децентралізованих системах. За останні кілька років Інтернет речей став однією з найважливіших технологій століття. Зараз людство досягло високого рівня розвитку технологій, що дозволяє налаштовувати взаємодію між пристроями та створювати безперебійний зв'язок між людьми, процесами та речами. З появою 5G технології, IoT стали центром розвитку майже для всіх сучасних галузей. Пристрої в цій архітектурі значно менші та мають низьке енергоспоживання. Звичайні алгоритми шифрування, як правило, дорогі в обчислювальному плані через їхню складність і вимагають багато раундів, однак це може поставити під загрозу бажану цілісність. Криптографія з низьким ресурсом — це компроміс між вартістю впровадження, швидкістю, безпекою, продуктивністю та енергоспоживанням на пристроях IoT. Мотивація полегшеної криптографії полягає в тому, щоб використовувати менше пам'яті, менше обчислювальних ресурсів і менше енергоспоживання, щоб забезпечити рішення безпеки, яке може працювати на пристроях з обмеженими ресурсами. Блокові шифри мають фіксовану довжину бітів і різні кроки перетворення, які визначаються симетричним ключем. Блокові шифри дуже універсальні, що дуже корисно з точки зору IoT. Ще одна перевага полягає в тому, що цей процес має майже ідентичні методи шифрування та дешифрування. Тому його можна реалізувати з меншими ресурсами.*

Ключові слова: *малоресурсна криптографія; Інтернет речей; блокові шифри; IoT.*

1. Вступ

Сьогодні широкосмуговий Інтернет є загальнодоступним, а вартість підключення постійно знижується. Як наслідок, все більше гаджетів та різних датчиків підключаються до всесвітньої мережі Інтернет [1-2]. Всі ці тенденції створюють сприятливе підґрунтя для розвитку Інтернету речей (IoT). Однак, навколо Інтернету речей існує багато складнощів [3]. Перш за все це зв'язано зі складністю, як елементної бази, так і спеціальних алгоритмів обробки даних, що реалізуються в IoT пристроях. Процес обміну такою великою кількістю даних починається з самих пристроїв, які повинні безпечно взаємодіяти з платформою [3].

Пристрої, з яких складається система *Інтернет Речей* - це будь-який фізичний об'єкт, який можна унікально ідентифікувати (за допомогою URI або унікального ідентифікатора ресурсу) і який може надсилати/отримувати дані шляхом підключення до мережі [1]. Прикладами є транспортні засоби, промислові контролери, *RFID*-мітки, сенсорні вузли, смарт-карти, побутова техніка, тощо [2]. Вони можуть бути з'єднані між собою, з центральним сервером/мережею серверів або через хмарні сервіси.

Сутність Інтернету речей - зв'язок та обмін інформацією [1]. Однак, всі ці дані не генеруються лише для того, щоб їх десь зберігати і «забути про їх існування». Основна ціль їх використання, це автоматизація. IoT практично «стирає» розрив між цифровим та фізичним світом, однак має і зворотний бік процесу - компрометація IoT пристроїв, може мати небезпечні наслідки в «реальному» світі.

Як правило пристрій Інтернету речей, містять один або декілька датчиків, які використовуються для збору даних та підтримки мережевих інтерфейсів. Тип і номенклатура даних, що збирають ці датчики, залежать від конкретного пристрою і їх функціонального завдання. Прицьому, всі накопичені дані, та дані телеметрії (технологічна інформація) інтенсивно

циркулють в межах відповідної мережі пристроїв, що обумовлює актуальність питань забезпечення безпеки обміну інформацією.

Існуючі в даний час платформи IoT використовують переважно централізовану модель, згідно з якою вони виступають в якості «брокерів» або концентраторів для управління обміном даними між пристроями IoT [3]. Однак, багато досліджень свідчать, що IoT повинен використовувати насамперед децентралізовану модель для забезпечення безпечного обміну даними. При цьому ключовими проблемами реалізації традиційної криптографії в пристроях IoT вважаються наступні [4]:

- низький рівень наявної обчислювальної потужності (*або відсутність батареї у випадку пасивних RFID-міток*);
- обмеженість ресурсів наявної пам'яті IoT пристроїв ;
- невелика фізична площа для реалізації збірки;
- низький заряд батареї (або навпаки її відсутність);
- реакція в реальному часі.

Малоресурсна або ж *легка криптографія* є компромісом між такими категоріями, як вартість реалізації, швидкість, безпека, продуктивність та енергоспоживання на пристроях з обмеженими ресурсами. При цьому, мотивація для використання малоресурсної криптографії полягає у використанні меншого обсягу пам'яті, менших обчислювальних ресурсів та меншого енергоспоживання заради забезпечення безпеки [5].

2. Класифікація та застосування малоресурсних криптографічних примітивів

За останнє десятиліття було запропоновано низку малоресурсних криптопримітивів, які мають переваги у продуктивності порівняно з стандартними криптографічними стандартами. Ці примітиви відрізняються від звичайних алгоритмів припущеннями, що малоресурсні примітиви не призначені для широкого кола застосувань і можуть накладати обмеження на потужність зловмисника.

Малоресурсна криптографія - це розділ криптографії, метою якого є розробка алгоритмів для використання в пристроях, які не здатні забезпечити більшість існуючих кодів і мають достатні ресурси (пам'ять, потужність, розмір) для роботи [5]. Хорошо відомі чотири типи малоресурсних криптографічних примітивів, які доступні для використання:

- Малоресурсні блокові шифри (LWBC);
- Малоресурсні потокові шифри (LWSC);
- Малоресурсні хеш-функції (LWHF);
- Криптографію еліптичних кривих (ECC).

Основними факторами, за якими можна проаналізувати малоресурсні криптографічні примітиви є: розмір блоку, розмір ключа, структура та кількість раундів. ECC є ще одним із варіантів малоресурсної криптографії, причому, будучи асиметричним шифром, він має можливість забезпечувати автентифікацію та неспростування.

Властивості малоресурсної криптографії обговорювалися в *ISO/IEC 29192* в *ISO/IEC JTC 1/SC 27. ISO/IEC 29192* є новим проектом зі стандартизації малоресурсної криптографії, і проект знаходиться в процесі стандартизації. У стандарті ISO/IEC 29192 властивості малоресурсності описуються на основі цільових платформ.

Дотричаючись завдань проектування, малоресурсні алгоритми використовують зазвичай менші розміри блоків - 32, 48 або 64 біт, ніж звичайний шифр, який має більший розмір блоків - 64 або 128 біт [6]. Малоресурсні алгоритми застосовують менші розміри ключів, (менше 96 біт). Найменший розмір ключа, за даними *NIST*, становить 112 біт [6]. У стандарті *ISO/IEC 29192* [6] детально описані властивості малоресурсності, що

встановлюються на цільових платформах. По-перше, легкість апаратних засобів оцінюється за розміром мікросхеми та їх енергоспоживання і, по-друге, за обсягом потрібної пам'яті.

Поряд з продуктивністю та вартістю, безпека є невід'ємним показником для будь-якого алгоритму малоресурсної криптографії. Властивість стійкості до атак будь-якого алгоритму малоресурсної криптографії може бути виміряна за допомогою криптоаналізу. Сенс криптоаналізу заключається в пошуці слабких місць алгоритму та розробку методів дешифрування [7]. Існує чотири основних типи атак на блоковий шифр [8]: – диференціальний криптоаналіз; – лінійний криптоаналіз; – інтегральний криптоаналіз; – алгебраїчні атаки. Ці атаки базуються на використанні «відомого відкритого тексту», «тільки шифрованого тексту», «обраного шифрованого тексту», «обраного відкритого тексту», а також атаки «людина посередині», атаки «грубою силою» та атак «побічного каналу» [8]. Крім того криптографію розділяють на дві основні напряму: симетричні та асиметричні шифри. Відповідно, у табл. 1 наведено порівняння, яке дозволяє продемонструвати різницю між асиметричною та симетричною криптографією [9].

Таблиця 1 – Порівняння методів криптографії

Параметр	**Особливості різновидів реалізації**	
	Криптографія з симетричним ключем	Криптографія з асиметричним ключем
Ключ	Один загальний приватний ключ	Унікальна пара приватного та публічного ключів. Генерація відкритих ключів залежить від криптографічних алгоритмів, заснованих на односторонніх математичних функціях.
Кількість ключів	Експоненційно пропорційні кількості користувачів	Лінійно пропорційні кількості користувачів
Швидкість та складність	Це прості алгоритми, завдяки цьому процес шифрування може бути здійснений швидко.	Це набагато складніший процес, ніж шифрування з симетричним ключем, і він відбувається повільніше через те, що для використання різних ключів потрібно більше часу.
Апаратна складність	Використовує алгоритми що потребують відносно недорогого апаратного забезпечення.	Більш складна реалізація апаратного забезпечення, яка обчислює важкі алгоритми які потребують більш потужне апаратне забезпечення.
Використання	Здебільшого використовується, для передачі великих обсягів даних.	Використовується в невеликих транзакціях, в першу чергу для автентифікації та встановлення безпечного каналу зв'язку перед фактичною передачею даних.
Алгоритми	RSA, DSA, ECC	Stream cipher: Trivium, Chacha, WG-8, Espresso, Grain 128. Block Ciphers: AES, DES, 3DES, Blowfish, Twofish, Curupira, PRESENT, KATAN. TEA, Humming Bird, RECTANGLE, SIMON

В межах даного матеріалу увага зосередженна, перш за все, на криптографії з симетричним ключем, яка може має можливість широко застосовуватися на пристроях, що піддаються жорстким ресурсним обмеженням [5]. В свою чергу, асиметричні шифри набагато вимогливі до обчислювальних ресурсів, ніж їх симетричні альтернативи.

Криптографія з симетричним ключем складається з основних функцій, таких як блокові або потокові шифри, а також методів застосування основної функції до пакету, яку носять назву режимом роботи блокового шифру для автентифікації чи шифрування [9]. Зусилля щодо криптографічній стандартизації малоресурсних примітивів розглядають як програмний, так і апаратний аспекти безпеки, котрі, зазвичай, мають, різні метрики. Програмні метрики включають цикли, пам'ять і цикл на байт, тоді як апаратні метрики враховують пропускну здатність, площу, співвідношення по всій площі. Тому важко отримати пряме порівняння між цими двома показниками [6].

Симетричне шифрування використовує один і той же ключ, як для шифрування, так і для розшифрування даних. Цей метод шифрування є безпечним і відносно швидким. Його основним недоліком є спільне використання ключа двома сторонами, що спілкуються. Зловмисник може розшифрувати дані, якщо має доступ до ключа. Алгоритми з симетричним ключем забезпечують конфіденційність і цілісність даних, але не гарантують автентифікацію [9]. Цей тип шифрування використовує три типи алгоритмів, заснованих на хешуванні, потоковому та блоковому шифрах.

3. Малоресурсні блокові шифри

Симетрична шифрування допомагає при проектуванні однієї і тієї ж схеми для шифрування і дешифрування з мінімальними витратами.

Блокові шифри - різновид симетричних шифрів, в яких обробляється відразу весь блок. Блокові шифри використовуються для побудови хеш-функцій та кодів автентифікації повідомлень (MAC) [10]. Полегшені блокові шифри базуються на двох типах структур: Мережа підстановки-перестановки (SPN) та Фейстеля.

Мережа Фейстеля (FN) - це багатораундовий шифр, який ділить вхідний блок на дві частини і працює тільки над половиною (дифузія) в кожному раунді шифруванні або дешифруванні. Між раундами ліворуч і праворуч половини блоку міняються місцями. Структура Фейстеля використовує свою кругову функцію лише на половині стану [10].

Таким чином, головною перевагою структури Фейстеля є використання одного і того ж програмного коду для процесу шифрування та дешифрування. Це зумовлює низьке використання пам'яті. Вона може бути реалізована на апаратних засобах з низькою середньою потужністю. Фейстелівська структура не підходить для конструкцій з малою затримкою. SPN є більш швидким, але без розкладу ключів. Відсутність ключового розкладу робить вразливим до атак. При однаковій величині запасу стійкості та однакових витратах енергії, структура SPN є більш придатною, оскільки вона вимагає меншого раунду виконання. За аналогічних умов SPN матиме менші енерговитрати.

PRESENT та *CLEFIA* - єдині два алгоритми, що затверджені стандартом ISO/IEC 29192 [11,12].

AES є класичним прикладом алгоритму на основі SPN, працює на 128-бітному блоці з 128, 192 та 256-бітними варіантами ключів [13]. Мінімальна вимога еквівалентів воріт (GE), зафіксована для AES, становить близько 2400 GE *(на 23% менше, ніж звичайна)* [13-14], що все ще є важким для деяких невеликих додатків у реальному часі. Це показує порівняно ефективну продуктивність при забезпеченні додатковими ресурсами.

Основними параметрами для оцінювання малоресурсного блочного шифру є розмір ключа, розмір блоку, тип структури та кількість раундів [5]. Малоресурсний шифр повинен відповідати чотирьом вимогам:

- Мінімальна площа кремнію або обсяг пам'яті;
- Низьке енергоспоживання;

- Менша кількість еквівалентів воріт (*GE*) для ефективної апаратної реалізації;
- Достатній рівень безпеки.

RFID-мітки можуть мати близько 1000-10000 GE, з яких можуть бути доступні 300-2100 GE для аспектів безпеки [15]. Для впровадження відповідних рішень у сфері забезпечення інформаційної безпеки (ІБ), загальна кількість доступних GE становитиме приблизно 2000-3000. При цьому, блокові шифри мають бути обмежені меншою кількістю GE для того, щоб відповідати малоресурсним додаткам.

AES, PRESENT та CLEFIA – це три шифри є обов'язкові для вибору. Слід підкреслити, що AES є найбільш широко використовуваним шифром, оскільки був встановлений в якості стандарту для шифрування в 2002 році. Він використовується багатьма пристроями IoT, незважаючи на те, що він не є малоресурсним шифром. PRESENT та CLEFIA - це два шифри, які також були стандартизовані, але як малоресурсні шифри. Характеристики цих шифри можна використовувати в якості «опорних» при оцінці властивостей інших шифрів. Наприклад, відомо [12], що для шифру CLEFIA автори змінили розмір S-box з 4-х біт до 8-ми біт, так щоб досягти кращих результатів при виконанні на програмному забезпеченні. В роботі [16] оптимізовано шифри-фіналісти конкурсу AES, же основний акцент зроблено на зменшенни обсягів займаної пам'яті за рахунок зменшення розміру коду з використанням функцій заміни макросів та інших повторень коду.

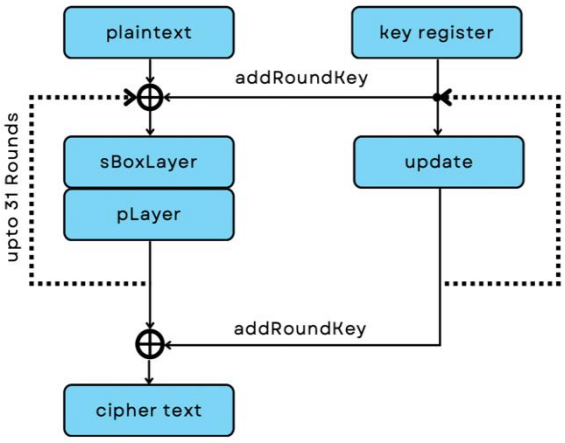


Рис. 1 – Алгоритмічний опис *PRESENT*

Використання AES призводе до високих GE [13], що робить їх нездійсненними для невеликих додатків, що працюють в реальному часі. Альтернативим рішенням для модифікації існуючого блокового шифру і створення ефективної апаратної моделі, є структура «*PRESENT*» [11]. Відповідний алгоритм (*див. рис. 1*) впроваджує малоресурсний блоковий шифр, та є більш меншим, ніж алгоритм AES. Розмір блоку процедур - 64, розмір ключа - 80 або 128, розмір S-box - 4. Один блок даних шифрується (розшифровується) за 31 раунд. Вихідні дані додатково розбиваються на блоки по 8 біт, де 4 біти цього значення складають стовпець, а ще чотири - рядок. Таким чином, вхідне значення замінюється на значення в S-Box.

Вихід з S-Box подається в блок перестановок, де біти переставляються місцями. Таким чином реалізовано 31 раунд обчислень.

Процес планування ключа буде оновлювати ключ для кожного раунду. Розшифрування виконується у зворотному порядку з інверсією S-Box.

В свою чергу, корпорація SONY представила CLEFIA, що стандартизована NIST у 2007 році. Вона базується на структурі Фейстеля і використовує 128-бітний блок з можливістю вибору ключа 128, 192, 256 біт через 18, 22, 26 раундів відповідно [12,17]. Дана реалізація демонструє високу продуктивність і стійкість до різних атак при порівняно високій вартості, оскільки найкомпактніша версія вимагає 2488 GE (тільки шифрування) для 128-бітового ключа [18]. CLEFIA має відповідні властивості [18], що робить його більш стійким до різних атак, але в той же час, це вимагає більшого об'єму пам'яті та обмежує його використання в надмалих додатках.

3.1 Порівняння методів малоресурсної криптографії

FELICS – система бенчмаркінгу з відкритим вихідним кодом, призначену для об'єктивного та послідовного оцінювання програмних реалізацій малоресурсних криптографічних примітивів для вбудованих пристроїв [19]. Фреймворк є довільно гнучким завдяки своїй модульній структурі, що дозволяє легко інтегрувати нові метрики, цільові пристрої та сценарії оцінювання. Вона складається з двох модулів, які в даний час можуть оцінювати продуктивність малоресурсних блокових і потокових шифрів на трьох широко використовуваних мікроконтролерах: 8-бітному AVR, 16-бітному MSP та 32-бітному ARM. FELICS має відносно простий користувальницький інтерфейс і призначений для використання розробниками шифрів для порівняння нових примітивів із існуючими. При цьому, отримані метрики є досить детальними та допомагають розробникам у виборі найкращого рішення, такого, що відповідає вимогам конкретного застосування.

Слід відмітити, що FELICS має реалізацію PRESENT, однак, вона не є вдалою [19]. Тому, в межах проведеного аналізу, було обрано 32-бітну реалізацію [11]. Ця реалізація була згодом оптимізована за допомогою 3-ох різних методів. В першу чергу, 4-розрядні S-box були замінені на 8-розрядні S-box для покращення продуктивності програмного забезпечення. Потім були розгорнуті всі перимутації через їх надмірну вартість для програмних реалізацій. Крім того, цикли були також повністю розгорнуті, щоб усунути всі залежності і підштовхнути шифр-код до найшвидшого рівня продуктивності. В кінці, звернення до пам'яті було зведено до мінімуму за рахунок утримання стану шифру в регістрах процесора протягом більшої частини часу виконання шифру.

Еталонна реалізація була отримана з сайту CLEFIA та адаптована до фреймворку FELICS [12, 17]. Еталонний алгоритм обчислює константи, які використовуються в планувальнику ключів, що призводить до невиправдано високого часу виконання. Тому була розроблена альтернативна версія 32-бітної реалізації, яка має попередньо обчислені значення всіх констант, що зберігаються в таблиці. Решта сім реалізацій, що були розроблені, експлуатують використання T-box. У той час як в цих реалізаціях застосовуються стандартні T-box для 8-бітового орієнтованого еталонного алгоритму та його оптимізованої 32-бітової орієнтованої версії відповідно, всі інші реалізації використовують скорочені T-box. Також, варто відмітити, що потокові шифри широко досліджуються в криптографічному середовищі через більш швидке виконання, але вони є вразливими до атак у порівнянні з блоковими шифрами.

В табл. 2 наведено стислі відомості щодо результатів порівняння деяких параметрів AES, PRESENT, CLEFIA та DES. Всі розглянуті шифри були реалізовані на модулі FELICS, для умов використання 8-розрядних мікроконтролерів AVR (сімейство 8-розрядних RISC мікроконтролерів, 100 kHz) [20].

Як і очікувалось, стандартизовані реалізації показують доволі повільні результати, за винятком PRESENT [11], де найповільніший час виконання має реалізація з невеликим обсягом коду. Це свидчить про те, що більшість оптимізацій дозволяє покращити час виконання навіть тоді, коли основна увага приділяється зменшенню розміру коду. PRESENT є «недружнім» малоресурсним шифром при націленості на програмні реалізації і тому, навіть з урахуванням декількох удосконалень та доопрацювань, все ще залишається дуже «важким» для програмного забезпечення (особливо для умов мобільних платформ). Ще одним результатом є те, що для збалансованих шифрів час виконання близький до 1000 тактів майже для кожного шифру, і лише PRESENT дає далекі від цього значення. Він має біт-орієнтовані перестановки, які важко обчислювати в програмному забезпеченні [11]. В той же час, як AES та CLEFIA підтримують блоки в 128 біт [12,17].

Таблиця 2 – Результати порівняння

Алгоритм	Алгоритм проектування шаблону	Розмір вхідного блоку	Розмір ключа	Кількість раундів	Площа (GEs)	Пропускна здатність (Kbps)	*Особливості*
AES	SPN	128	128	1,032	5440	15.53	*Високий рівень безпеки, гнучкість.*
PRESENT	SPN	64	80	31	10579	201.53	*Менша кількість воріт, менше пам'яті. Доцільний для шифрування невеликих обсягів даних.*
CLEFIA	GFN	128	128	36	27738	360.44	*Висока продуктивність та стійкість до різних атак.*

CLEFIA - алгоритм шифрування, котрий має серед досліджуваних алгоритмів найбільшу довжину блоку з довжиною блоку 128 біт [12,17], в той час як в інших алгоритмах перевага віддається довжині блоку 64 біт. Це важливо для пристроїв, що обмінюються даними в мережі Інтернет з об'єктами малої ємності. Ефективніше шифрувати блоки невеликого розміру, а також ті, що застосовують архітектуру Фейстеля. З іншого боку, збільшення розміру ключа знижує енергоефективність.

Звичайно, чим більший розмір ключа, тим краще забезпечується безпека. Однак, в умовах роботи IoT, більш вдалими слід вважати ключі від 80 біт до 128 біт (принаймні поки). Вибір структур простим способом, який не потребує занадто багато енергії, підвищує ефективність. Енергоємні структури, такі як процеси редукції та змішані удари, що використовуються в алгоритмах CLEFIA підтверджує цю ситуацію.

PRESENT має певну перевагу над CLEFIA: - нелінійний S-box використовує 4-бітову структуру, що призводить до меншого GE і меншого енергоспоживання. Додаткові властивості S-box допомагають PRESENT досягти бажаного лавинного ефекту, а результати роботи [11] свідчать про те, що PRESENT має компактний S-box. Також в PRESENT є 16 S-box, які розділені на чотири групи. Деякі важливі відмінності цих S-box наведені нижче [11]:

1. Вхідний біт до S-box надходить з 4 чітко визначених S-box тієї ж групи.
2. Вхідні біти до групи з чотирьох S-box надходять з 16 різних S-box.
3. Чотири вихідні біти з певного S-box надходять у чотири чітко визначені S-box, кожен з яких належить до окремої групи S-box у наступному раунді.
4. Вихідні біти S-box у різних групах подаються до різних S-box.

Апаратна реалізація AES для IoT, з точки зору ІБ, може залучати деякі апаратні атаки. Тому важливо спостерігати за цими спробами і своєчасно знаходити потрібні рішення. В цілому, AES та CLEFIA є двома найбільш вдалими прикладами шифрів, які витрачають багато ресурсів (на розмір коду) для досягнення їх більш швидкої роботи.

4. Висновки

Зростання масштабів використання IoT, обумовлює потребу в більш широкому запровадженні механізмів (алгоритмів) малоресурсного шифрування.

Для пристроїв із певними ресурсними обмеженнями, наявні стандарти криптографічних алгоритмів можуть бути занадто складними та/або занадто енерговитратними. Крім того, кіберзлочинці можуть скористатися недоліками паролів, які відносно легко підбираються, якщо немає жорстко декларованих вимог до паролів, які створюються користувачами.

Для пристроїв з обмеженими ресурсами, зокрема пристроїв IoT, малоресурсна криптографія є ефективним напрямом забезпечення безпеки їх мережевої взаємодії.

Спираючись на результаті проведеного аналізу, можна констатувати, що AES, PRESENT та CLEFIA, є найбільш експериментально дослідженими та широко адаптованими блоковими шифрами. Однак, з появою нових алгоритмів малоресурсної криптографії, що є об'єктивним процесом, з'являються і нові методики та різновиди атак.

В цілому, бажаний малоресурсний алгоритм повинен забезпечувати баланс між вартістю, продуктивністю та безпекою. При цьому слід мати на увазі, що оптимізувати всі три цілі одночасно, дуже важко.

В індустрії IoT не існує AES-подібного стандарту для малоресурсних алгоритмів. З цієї причини найближчим часом можна очікувати інтенсифікацію розробок нових алгоритмів шифрування для нових IoT. Безумовно, безпечна мережева взаємодія має велике значення в сфері IoT, однак, крім питані ІБ, ефективне застосування IoT, є не менш важливим аспектом. Тому при розробці малоресурсних криптографічних алгоритмів, параметри їх енергоспоживання будуть вкрай актуальні.

Слід зазначити, що S-box PRESENT реалізує дуже компактну реалізацію, що споживає лише 21 GE для одного 4-бітового S-box. Крим того, з точки зору вимог до обсягів використовуваної пам'яті, цей алгоритм (в порівнянні з іншими) найкращім чином підходить для вирішення питань забезпечення малоресурсного криптографічного захисту даних при здійсненні мережевої взаємодії IoT (принаймні у найближчій перспективі).

Список літератури

[1] Xia, F., Yang, L. T., Wang, L., & Vinel, A. (2012). Internet of Things. International Journal of Communication Systems, 25(9), 1101–1102. https://doi.org/10.1002/dac.2417

[2] Jia, X., Feng, Q., Fan, T., & Lei, Q. (2012). RFID technology and its applications in Internet of Things (IoT). 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet). https://doi.org/10.1109/cecnet.2012.6201508

[3] Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645–1660. https://doi.org/10.1016/j.future.2013.01.010

[4] Suo, H., Wan, J., Zou, C. and Liu, J. (2012) Security in the Internet of Things: A Review. IEEE International Conference on Computer Science and Electronics Engineering, Hangzhou, 23-25 March 2012, 648-651. https://doi.org/10.1109/ICCSEE.2012.373

[5] McKay, K. A., Bassham, L., Turan, M. S., & Mouha, N. (2017). Report on lightweight cryptography. https://doi.org/10.6028/nist.ir.8114

[6] ISO/IEC 29192-2:2012. (2012). Information technology – Security techniques – Lightweight cryptography – Part 2: Block ciphers. Retrieved from https://www.iso.org/obp/ui#iso:std:iso-iec:29192:-2:ed-2:v1:en.

[7] Banik, S., Bogdanov, A., Isobe, T., Shibutani, K., Hiwatari, H., Akishita, T., & Regazzoni, F. (2015). Midori: A Block Cipher for Low Energy. Advances in Cryptology – ASIACRYPT 2015, 411–436. https://doi.org/10.1007/978-3-662-48800-3_17

[8] Eisenbarth, T., Gong, Z., Güneysu, T., Heyse, S., Indesteege, S., Kerckhof, S., Koeune, F., Nad, T., Plos, T., Regazzoni, F., Standaert, F.-X., & van Oldeneel tot Oldenzeel, L. (2012). Compact Implementation and Performance Evaluation of Block Ciphers in ATtiny Devices. Progress in Cryptology - AFRICACRYPT 2012, 172–187. https://doi.org/10.1007/978-3-642-31410-0_11

[9] Chandra, S., Paira, S., Alam, S. S., & Sanyal, G. (2014). A comparative survey of Symmetric and Asymmetric Key Cryptography. 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE). https://doi.org/10.1109/icecce.2014.7086640

[10] Diehl, W., Farahmand, F., Yalla, P., Kaps, J.-P., & Gaj, K. (2017). Comparison of hardware and software implementations of selected lightweight block ciphers. 2017 27th International Conference on Field Programmable Logic and Applications (FPL). https://doi.org/10.23919/fpl.2017.8056808

[11] Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B., Seurin, Y., & Vikkelsoe, C. (2007). PRESENT: An Ultra-Lightweight Block Cipher. Cryptographic Hardware and Embedded Systems - CHES 2007, 450–466. https://doi.org/10.1007/978-3-540-74735-2_31

[12] Shirai, T., Shibutani, K., Akishita, T., Moriai, S., & Iwata, T. (2007). The 128-bit blockcipher CLEFIA. In International workshop on fast software encryption, 181-195. Springer, Berlin, Heidelberg.

[13] Federal Information Processing Standards Publication 197 Announcing the ADVANCED ENCRYPTION STANDARD (AES). (2001). Retrieved from https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[14] Moradi, A., Poschmann, A., Ling, S., Paar, C., & Wang, H. (2011). Pushing the Limits: A Very Compact and a Threshold Implementation of AES. Advances in Cryptology – EUROCRYPT 2011, 69–88. https://doi.org/10.1007/978-3-642-20465-4_6

[15] Juels, A., & Weis, S. A. (2005). Authenticating Pervasive Devices with Human Protocols. Advances in Cryptology – CRYPTO 2005, 293–308. https://doi.org/10.1007/11535218_18

[16] Grossschadl, J., Tillich, S., Rechberger, C., Hofmann, M., & Medwed, M. (2007). Energy Evaluation of Software Implementations of Block Ciphers under Memory Constraints. 2007 Design, Automation & Test in Europe Conference & Exhibition. https://doi.org/10.1109/date.2007.364443

[17] Akishita, T., & Hiwatari, H. (2012). Very Compact Hardware Implementations of the Blockcipher CLEFIA. Selected Areas in Cryptography, 278–292. https://doi.org/10.1007/978-3-642-28496-0_17

[18] Hatzivasilis, G., Fysarakis, K., Papaefstathiou, I., & Manifavas, C. (2017). A review of lightweight block ciphers. Journal of Cryptographic Engineering, 8(2), 141–184. https://doi.org/10.1007/s13389-017-0160-y

[19] Dinu, D., Biryukov, A., Großschädl, J., Khovratovich, D., Le Corre, Y., & Perrin, L. (2015). FELICS - Fair Evaluation of Lightweight Cryptographic Systems. Retrieved from https://csrc.nist.gov/csrc/media/events/lightweight-cryptography-workshop-2015/documents/papers/session7-dinu-paper.pdf

[20] Meiser, G., Eisenbarth, T., Lemke-Rust, K., & Paar, C. (2008). Efficient implementation of eSTREAM ciphers on 8-bit AVR microcontrollers. 2008 International Symposium on Industrial Embedded Systems. https://doi.org/10.1109/sies.2008.4577681

Received: on July 2022. Accepted: on July 2022.

Authors:
Eugene Demenko, CSD Student, V. N. Karazin Kharkiv National University, Kharkov, Ukraine.
E-mail: xa11868404@student.karazin.ua
Oleksii Nariezhnii, Ph.D., Associate Professor, V. N. Karazin Kharkiv National University, Kharkov, Ukraine.
ORCID ID https://orcid.org/0000-0003-4321-0510
E-mail: o.nariezhnii@karazin.ua

Research of application of low-resource cryptography algorithms in decentralized environments.

Abstract. The purpose of this material is to analysis of the application of low-resource cryptography algorithms for Internet of Things (IoT) systems and the possibility of their implementation in decentralized systems. Over the past few years, the Internet of Things has become one of the most important technologies of the century. Modern IT developments has reached a high level of technological development, which allows you to customize the interaction between IoT devices and provide connection between people. With the appearance of 5G technologies, the IoT has become the center of development, for almost to all modern industries. Devices in this architecture are significantly smaller and have low power consumption. Conventional encryption algorithms tend to be computationally expensive due to their complexity and require many processing rounds. Low-resource cryptography is a compromise between implementation cost, speed, security, performance, and power consumption on IoT devices. The motivation for lightweight cryptography is to use less memory, less computing resources, and less power consumption to provide a security solution that can run on resource-constrained devices. Block ciphers have a fixed length (of bits) and special transformation stages, which are determined by a symmetric key. Block ciphers are quite versatile, which is very useful from an IoT perspective. Another advantage is that block ciphers has nearly proportional encryption and decryption methods. Therefore, it can be implemented with fewer resources.

Keywords: low-resource cryptography; Internet of things; block ciphers; IoT.

УДК 621.327:621.391 DOI: 10.26565/2519-2310-2022-1-04

ВИКОРИСТАННЯ ПАРАМЕТРІВ ДОВЖИН СЕРІЙ, ЯК ЕЛЕМЕНТА МІЖБЛОЧНОГО МУЛЬТИПЛЕКСУ ДАНИХ СТЕГАНОАЛГОРИТМУ

Микита Гончаров, Юлія Лєсная

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

Надійшла: липень 2022. Прийнята: липень 2022.

Анотація: *Розглянуто особливості використання параметрів довжин серій та кількості сформованих опорних блоків, як елементів комплексного ключа екстрактора даних, гібридного стеганоалгоритму. Наведено результати атаки (зламу) тестових зображень, які отримані для стеків вибірки різної довжини (різної бази перестановок діючих параметрів серій). Зроблено висновок про провідну роль параметра «довжина серій» при реалізації процедур міжблокового мультиплексування стеганоконтента. Наголошено, що одночасне використання дворівневого мультиплексу даних, значно розширює можливості протистояння спробам атак контенту. Встановлено, що застосування блоків з більшою розмірністю, істотно зменшує роль параметра «довжин серій», як основного елемента для руйнування структури вихідних зображень. Констатується, що збільшення довжини стека вибірки серій, розширює потенційну комбінаторику мультиплексування для діючих пар параметрів серій, та в більшої мірі руйнує кореляційні зв'язки елементів вихідного масиву даних. За результатами моделювання зроблено висновок, що використання різних способів розгортки серій забезпечує ще одну позицію в структурі ключа екстрактора даних.*

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

1. Вступ

Ця робота відображає деякі результати, отримані в ході проведення моделювання основних процедур міжблокового мультиплексування даних, у рамках відпрацювання загальної концепції малоресурсного гібридного стеганографічного алгоритму [1]. На даному етапі робіт основна увага приділена дослідженню одержуваних ефектів, при використанні в якості елементів мультиплексування, діючих параметрів масиву серій [2]: – кількості опорних блоків (ОБ) зображень та параметра довжин серій ОБ. Можливість формування базового масиву серій ОБ певною мірою, забезпечується за рахунок проведення відповідних процедур «згладжування» зображень (*або передобробки вихідних даних*), що реалізованих на першому етапі роботи дослідного алгоритму [1]. У рамках моделювання процедур першого етапу було досліджено три варіанта згладжування вихідних зображень [1], які дозволяють отримати необхідний результат [2] за кількістю блоків ідентичного змісту (*тобто серій ОБ*) для різних типів вихідних даних *(у даному випадку, тестових зображень)*.

В якості тестових зразків даних використані напівтонові зображення трьох різних типів: - зображення типу *«портрет»*; - зображення типу *«пейзаж»* та зображення типу «*мнемосхема»*. Основна відмінність між ними полягає в характерних значеннях ймовірності перепаду яскравості між сусідніми елементами (пікселями) зображень кожного типу [3-5].

2. Основна частина

Для демонстрації отриманих ефектів, використана *полегшена* версія дослідного алгоритму, що передбачає підтримку виключно міжблокового рівня мультиплексу діючих параметрів масиву довжин серій (*ОБ та їх довжин серій)*, який було сформовано за результатами етапу згладжування тестових зображень. Маска перестановок, котра використовується для цих параметрів, визначається відповідною позицією елемента в структурі композиційного ключа екстрактора даних, та забезпечує міжблоковий рівень мультиплексування діючих параметрів масиву довжин серій для обраної розмірності ОБ [1-2].

Для імітації шифрування контенту використано короткий стек вибірки серій ОБ, з довжиною в 4 серії. Реалізований механізм перестановок діючих параметрів серій ОБ (*на вузької базі*) наведено на рис.1.

Зразки використаних тестових зображень наведенні на рис. 2-5*(з)*.

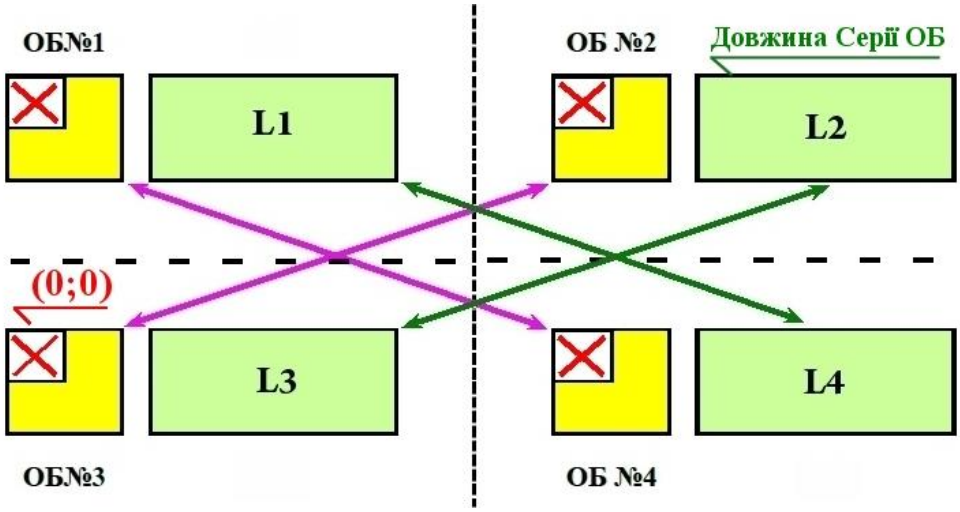


Рис. 1 – Тестова маска перестановок на вузькій базі вибірки

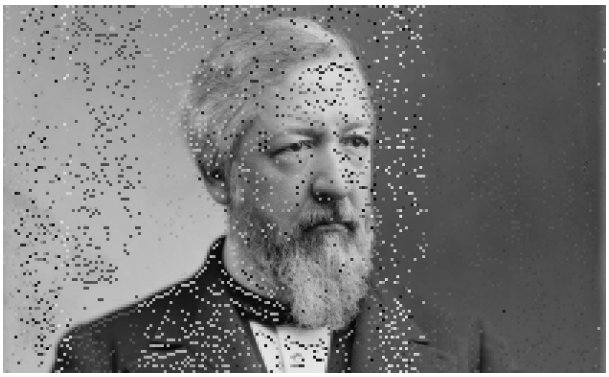
В межах проведеного моделювання передбачалося, що атакуючий правильно визначив діючий параметр довжини стека *(тобто, базу вибірки*) та принцип вибірки серій (*спосіб розгортки серій*), проте послідовно помилявся у визначенні діючих значень 2-х ключових параметрів, що залишилися: 1 – параметра зсуву ОБ *(жовті блоки на рис.1*); 2 – параметра довжини серій ОБ (*зелені блоки* - ***L)***. Відповідно до двох зазначених варіантів розвитку атаки на рис. 2-4*(а-е)* наведені результати несанкціонованого вилучення контенту при хибному доборі зазначених ключових параметрів:

- при неправильному доборі параметра зсуву ОБ – зразки *(а, в, д)*;
- при неправильному параметрі ***L*** – зразки (*б, г, е*).

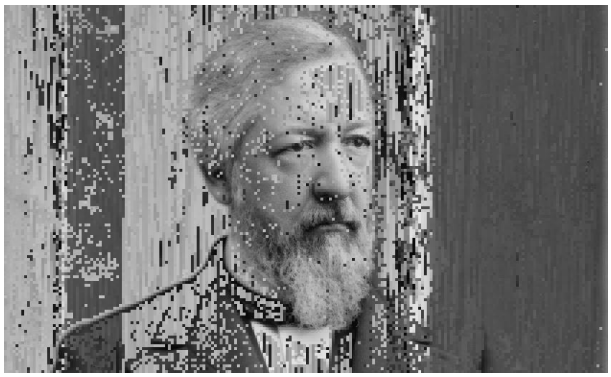
На рис. 5 представлена візуалізація отриманої різниці між вихідним та відновленим *(тобто. нелегітимно вилученим*) зображеннями для зазначених вище результатів атаки при різних розмірностях блоків *(зразки (а, в, д) проти зразків (б, г, е))*. При цьому, чим яскравіше (*біліше*) точка чи фрагмент зображень на рис. 5, то тем більше ступінь відмінності «зламаного» зображення від його оригіналу, і відповідно, чим темніше зазначений елемент/фрагмент, тим ближчі його параметри до вихідних значень оригіналу.

Ширину потенційної бази вибірки досліджуваних параметрів серій (*ОБ та* ***L)*** для різних типів зображень, різних розмірностей блоків та значень **Pz** (*значення порога загрубления яскравості елементів* зображення [1]) можна оцінити за діаграмами на рис 2-4 (*ж*). Значення параметра **Pz** = 3 вибиралося, як компроміс між урахуванням особливостей обраних тестових зображень та особливостями зорової системи людини [4-5], наприклад, властивістю зорового апарату людини виявляти на зображенні різні регулярні структури [3].

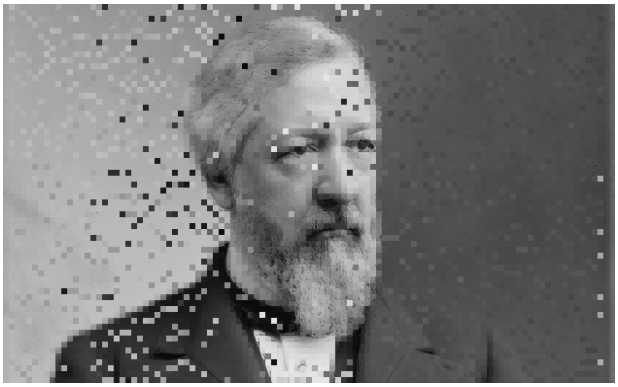
Вибір стека малої довжини обумовлений припущенням того, що *неправильний* підбір параметрів вилучення контенту *(спроба атаки*) на стеку з більш широкою базою буде призводити до більш руйнівних результатів при відновленні вихідних даних. Іншими словами, використання широкої бази перестановок поточних параметрів серій ОБ, призводить до більшого порушення просторової кореляції між елементами зображень [3-5].



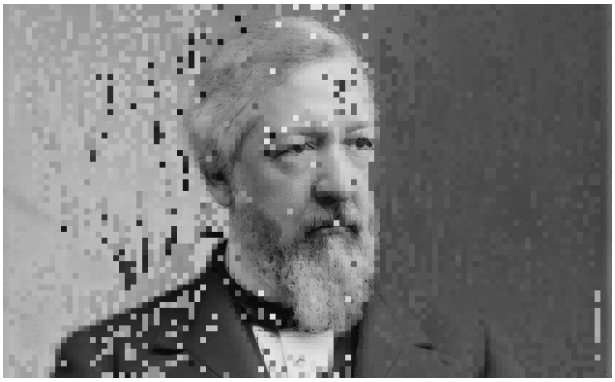
а) тіх ОБ розмірністю 4×4 ел.;



б) тіх Довжин серій, ОБ 4×4 ел.;

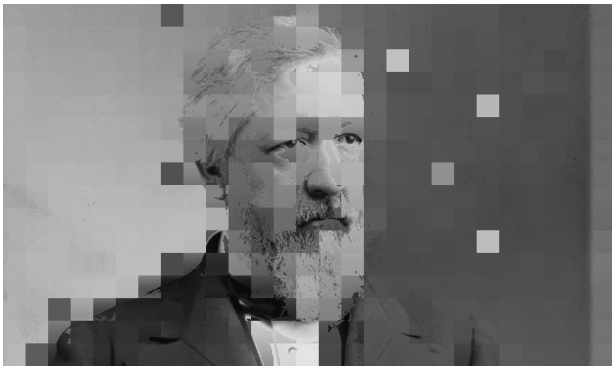


в) тіх ОБ розмірністю 8×8 ел.;

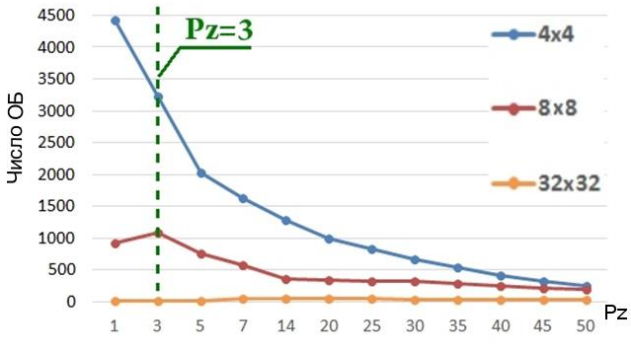


г) тіх Довжин серій, ОБ 8×8 ел.;

д) тіх ОБ розмірністю 32×32 ел.;



е) тіх Довжин серій, ОБ 32×32 ел.;



ж) кількість ОБ різної розмірності;

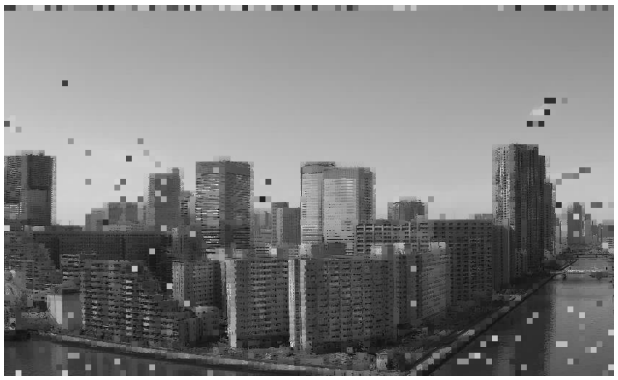
з) вихідне тестове зображення;

Рис. 2 – Результати атаки тестового зображення типу *«портрет»*
для різних розмірностей ОБ *(2-й Вар. згладжування)*

*Прим.: - «**тіх**», це скорочення терміну мультиплексування.*

а) тіх ОБ розмірністю 4×4 ел.;

б) тіх Довжин серій, ОБ 4×4 ел.;

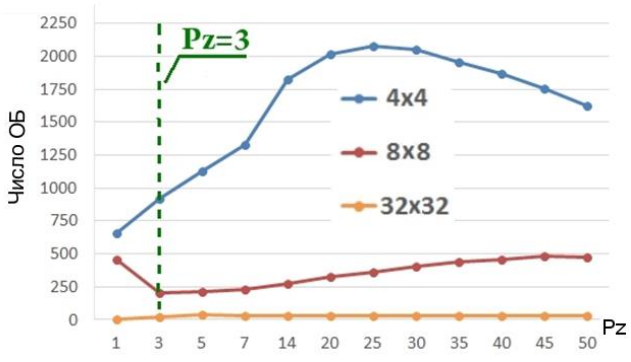


в) тіх ОБ розмірністю 8×8 ел.;

г) тіх Довжин серій, ОБ 8×8 ел.;

д) тіх ОБ розмірністю 32×32 ел.;

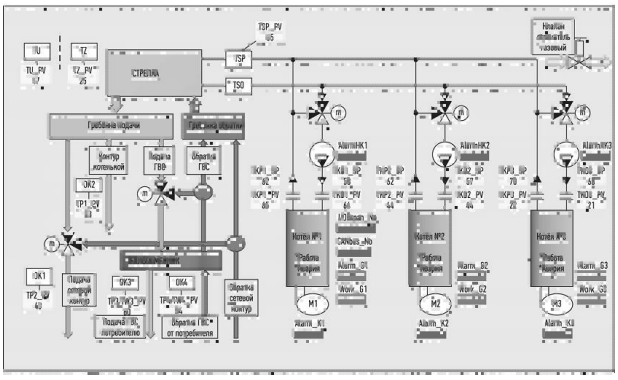
е) тіх Довжин серій, ОБ 32×32 ел.;



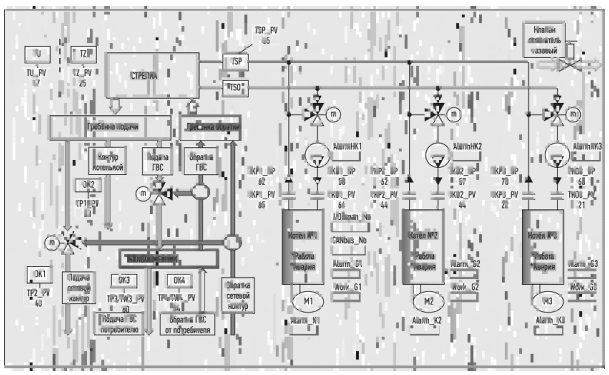
ж) кількість ОБ різної розмірності;

з) вихідне тестове зображення;

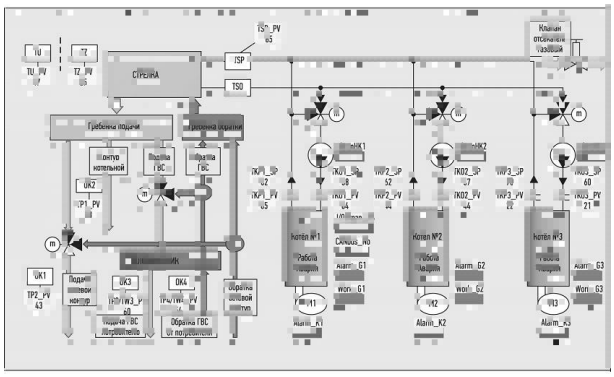
Рис. 3 – Результати обробки зображення типу *«пейзаж»*
для різних розмірностей ОБ *(2-й Вар. згладжування)*



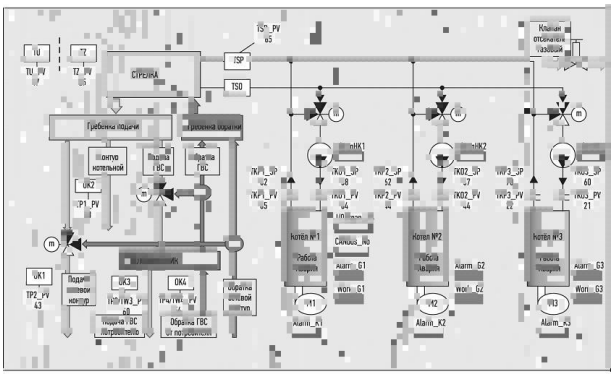
а) тіх ОБ розмірністю 4×4 ел.;



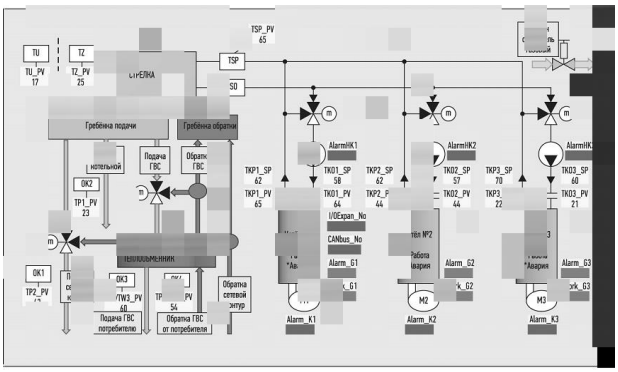
б) тіх Довжин серій, ОБ 4×4 ел.;



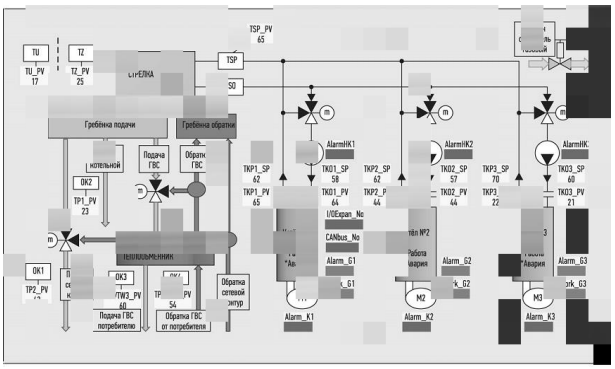
в) тіх ОБ розмірністю 8×8 ел.;



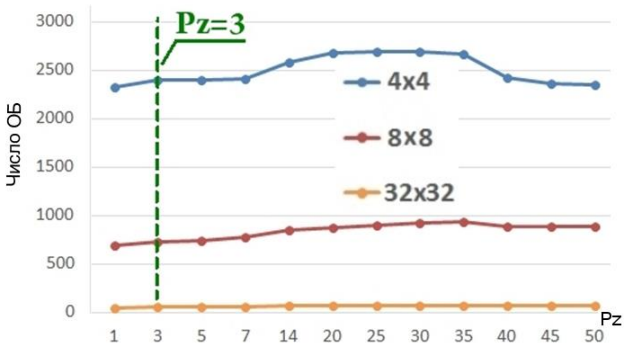
г) тіх Довжин серій, ОБ 8×8 ел.;



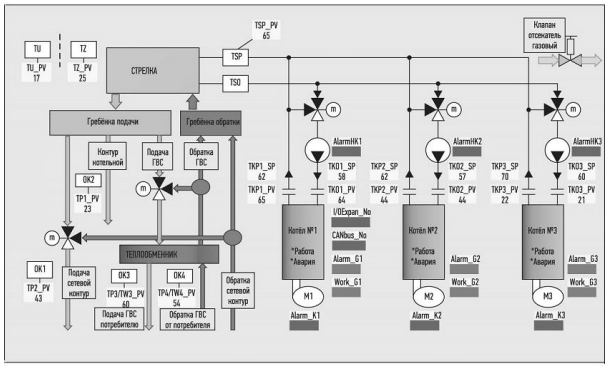
д) тіх ОБ розмірністю 32×32 ел.;



е) тіх Довжин серій, ОБ 32×32 ел.;



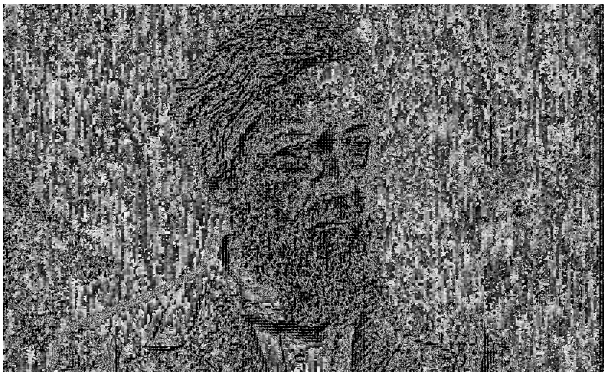
ж) кількість ОБ різної розмірності;



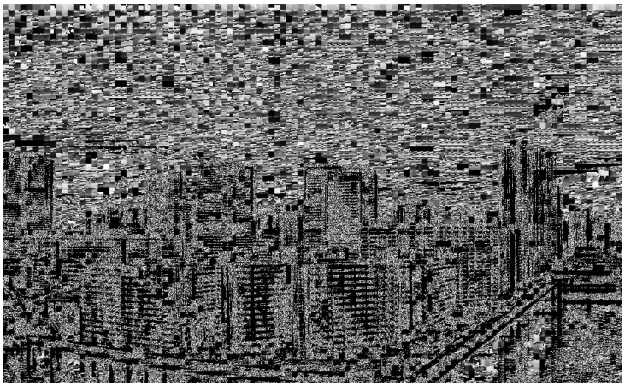
з) вихідне тестове зображення;

Рис. 4 – Результати атаки зображення типу «*мнемосхема*»
для різних розмірностей ОБ *(3-й Вар. згладжування)*

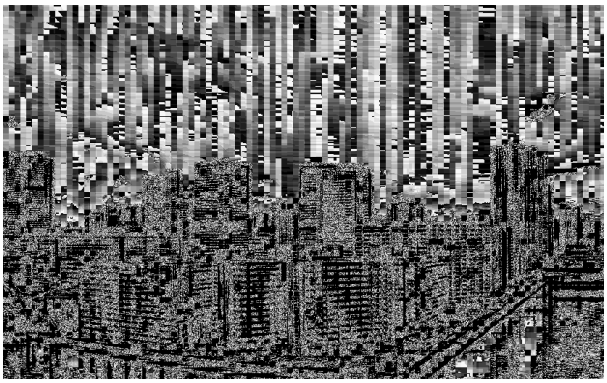
а) тіх ОБ, «портрет», 4×4 ел.;



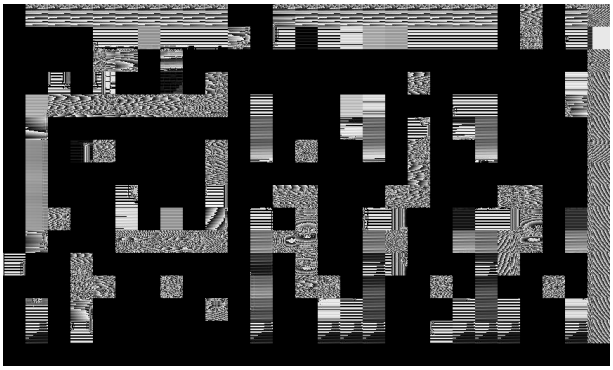
б) тіх Серій, «портрет», 4×4 ел.;



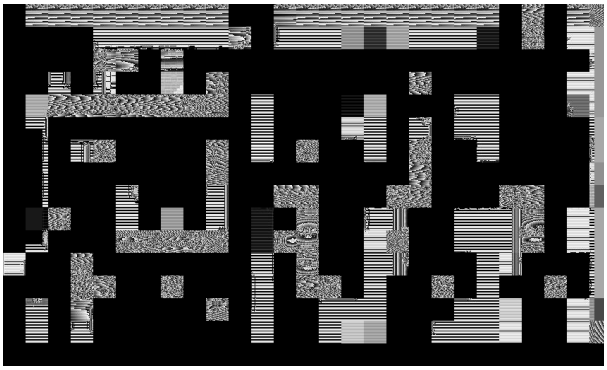
в) тіх ОБ, «пейзаж», 8×8 ел.;



г) тіх Серій, «пейзаж», 8×8 ел.;



д) тіх ОБ, «мнемосхема», 32×32 ел.;



е) тіх Серій, «мнемосхема», 32×32 ел.;

Рис. 5 – Візуалізація різниці вихідних та «атакованих» зображень
для різних розмірностей та комбінацій підбору діючих параметрів мультиплексу
((а-г) - 2-й Вар. згладжування; (д-е) - 3-й Вар. згладжування)

Таким чином, якщо на малій довжині стека одержуваний ефект (*тобто руйнування вихідних даних)* буде помітним, то при використанні стеку з більшою комбінаторикою перестановок цей процес стане ще більш очевидним. Відповідна тестова маска перестановок параметрів серій ОБ, що діють, на довгому стеку вибірки, представлена на рис. 6. В даному випадку базовий масив серій ОБ, був розділений на дві частини (напівстккеки) рівної довжини, між елементами яких проводяться відповідні маніпуляції.

Характерні результати атаки (*тобто, спроб нелегітимного вилучення стеганоконтенту*) для 2-х вибраних типів тестових зображень при використанні стеків вибірки різної довжини *(див. рис.1 та 6)*, представлені нижче, на рис.7.

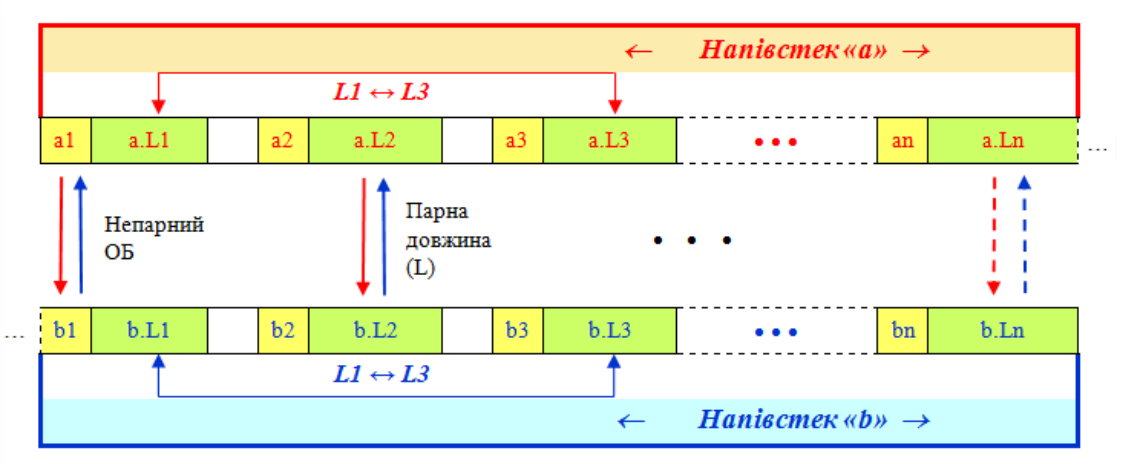


Рис. 6 – Тестова маска перестановок на широкій базі вибірки параметрів серій

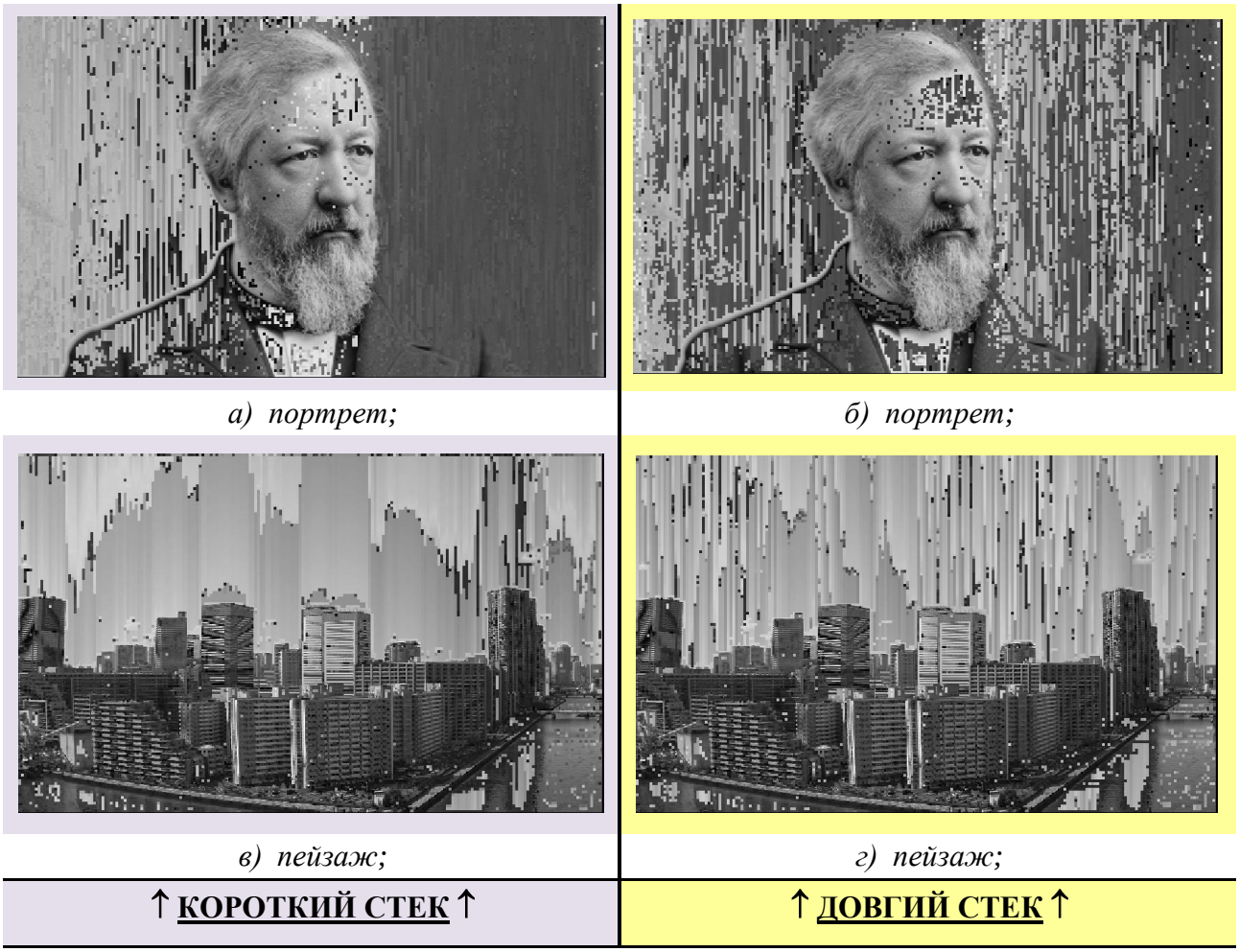


Рис. 7 – Результати «атаки» тестових зображень для стеків різної розмірності при $P_Z = 7$ (для *ОБ 4×4 ел.*)

Як було зазначено вище, основною метою представленого матеріалу є демонстрація можливості використання метода кодування довжин серій, як основи механізму міжблокового мультиплексування діючих параметрів довжин серій ОБ для забезпечення протидії спробам нелегітимної екстракції стеганоконтенту. Наочним підтвердженням цієї можливості є попарне порівняння зразків зображень, які представлені на рис. 2-4 *(а-е)*, де добре проявля-

ються вертикальні регулярні структури блоків (*т.зв. доріжки*), що є наслідком хибного підбору використаних ключових параметрів на довжині стека *всього* в 4 серії. А порівняння тестових зразків, що представлені на рис.7, переконливо демонструє різницю в наслідках атаки контенту, для захисту якого використовуються стеки різної довжини, що формують різне комбінаторне поле для діючих параметрів серій ОБ.

Важливо підкреслити, що на даному циклі моделювання було вимкнуто функцію внутрішньоблокового мультиплексування даних (*зсув значень (0;0) на рис.1*), що добре видно по практично неспотвореним високодетальним областям тестових зображень на рис.7 (*фрагменти обличчя з зачіскою, очима і бородою, та частини зображень, з деталями будівель*). Крім того, варто звернути увагу на те, що спосіб формування базового масиву серій може бути дуже різним, а його конкретна реалізація, також є одним із елементів використовуваної ключової послідовності. Як видно із рис. 2-4 *(б, г),* у даному випадку була використана розгортка по стовпцях (*про що свідчать вертикальні доріжки з помилково відновлених блоків на рис.7).* При цьому поєднання різних способів вибірки діючих пар довжин серій (*ОБ і параметру його довжини*) та різних принципів організації самої розгортки серій (*по стовпцях, зигзаг тощо*), додатково розширює спектр можливих станів відповідного елемента в структурі складеного ключа екстрактора даних (*див. рис.2 в роботі* [1]).

3. Висновки

1. Використання параметра «довжин серій» ОБ, дозволяє отримати набагато більш суттєвіший ефект, ніж при реалізації міжблокового мультиплексу тільки за рахунок зсуву ОБ (*див. рис. 2-5, зразки (а, в, д) проти зразків (б, г, е)).*

2. Результати експериментів підтвердили припущення, що поєднання одразу 2-х зазначених параметрів серій [4], помітно ускладнює процес підбору діючих компонентів складеного ключа екстрактора даних. Причому основну роль відіграє саме параметр довжин серій (*інтенсивність та кількість блоків білого кольору на рис. 5 у зразках (а, в, д) проти (б, г, е)).*

3. Одночасне використання дворівневого мультиплексу даних *(ОБ і параметра довжин серій на 1-му рівні, та значень середньої яскравості ОБ (елемент (0; 0) на рис. 1) на 2-му рівні мультиплексу),* значно розширює можливості протистояння спробам атак контенту.

4. Збільшення довжини стека вибірки серій розширює комбінаторику мультиплексу діючих параметрів серій ОБ, та в більшої мірі руйнує кореляційні зв'язки елементів вихідного масиву даних. Цей ефект виразно підтверджується значним збільшенням щільності розміщення серій блоків різного відтінку у фонових областях тестових зображень, що використовують широку базу перестановок *(див. порівняння зображень в різних стовбцях на рис 7).*

5. Збільшення розмірності ОБ для всіх типів зображень призводить до зменшення загальної кількості серій, що звужує базу можливих перестановок (рис. 2-4(*ж*)). За сукупністю показників, найбільш збалансованою розмірністю блоків є діапазон від 4 до 8 елементів.

6. Застосування блоків більшої розмірності (див. Рис. 2-4 *(д-е)*) значною мірою зменшує роль параметра довжин серій, як основного елемента для «руйнування» вихідної структури зображень. У даному випадку кількість серій, що формуються, практично для всіх типів зображень, знаходиться в одному діапазоні (*нижній помаранчевий графік на рис. 2-4 (ж)).*

7. Використовуваний варіант згладжування вхідних даних значно меншою мірою визначає кількість отриманих серій, ніж це робить параметр загрублення **Pz**. При заданих розмірностях блоків збільшення порогового значення яскравості сусідніх елементів (*зміщення пунктирної зеленої лінії «***Pz***» вправо на рис. 2-4(ж))* помітно змінює можливу комбінаторику перестановок, особливо для блоків малої та середньої розмірності *(4×4 та 8 ×8 ел.).*

8. Використання для контенту та контейнера блоків різної розмірності (*тобто режиму «несиметричної обробки»*) створює хороші вихідні співвідношення для подальшої інкапсуляції контенту.

9. При взаємному порівнянні результатів впливу параметра **Pz** і розмірності блоків на кількість та довжину формованих серій ОБ, безумовним лідером є параметр розмірності блоків (рис. 2-4(*ж*)).

10. Збільшення значення **Pz** в області малої візуальної помітності (*тобто до 7 градацій яскравості*) для різних типів даних, за вектором процесу, що спостерігається (*збільшення або зменшення кількості ОБ*), дає різні результати [2], що пояснюється відмінністю статистичних властивостей обраних тестових зображень [4]. При цьому у всіх випадках спостерігається збільшення помітності спотворень відновлюваних зображень.

11. Впровадження різних способів розгортки серій забезпечує ще одну позицію в структурі ключа екстрактора даних.

Список літератури

[1] Лесная, Ю., Гончаров, Н., & Малахов, С. (2021). Отработка концепта многоуровневого мультиплекса данных гибридного стеганоалгоритма. Збірник наукових праць SCIENTIA. (Vol.2), 48-55. Вилучено з https://ojs.ukrlogos.in.ua/index.php/scientia/article/view/17666

[2] Гончаров О., Лєсная Ю., Погоріла К., Богданова Є., Малахов С. Дослідження параметру «серій опорних блоків», як елементу композитного ключа екстрактора даних стеганоалгоритму // Problems of science and practice, tasks and ways to solve them. Proceedings of the XX International Scientific and Practical Conference. Warsaw, Poland. 2022. Pp. 779-785. Вилучено з https://isg-konf.com/problems-of-science-and-practice-tasks-and-ways-to-solve-them-two/

[3] Ярославский Л. П. (1979). Введение в цифровую обработку изображений. Москва: Сов. Радио.

[4] Прэтт У. (1985). Цифровая обработка изображений (Д. С. Лебедева, пер. с англ.). т. 1,2. Москва: Мир.

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

Received: on July 2022. Accepted: on July 2022.

Authors:
Mykyta Honcharov, student of the Department of Security of Information Systems and Technologies, V. N. Karazin Kharkiv National University, Ukraine.
ORCID ID https://orcid.org/0000-0002-9790-7260
E-mail: worldxdark@gmail.com
Yuliia Liesnaia, 4rd year student, Faculty of Computer Science, V.N. Karazin Kharkiv National University, Ukraine.
E-mail: xa12284109@student.karazin.ua

Using the parameters of the series lengths as an element of the interblock data multiplex of the steganoalgorithm.

Annotation. The peculiarities of using the parameters of runs lengths and the number of generated anchor blocks as elements of the composite key of the data extractor for a hybrid steganoalgorithm, are considered. The results of attacking (hacking) test images obtained for sample stacks of different lengths (forming a different base of permutations of valid runs parameters) are presented. It has been concluded that the "runs length" parameter plays the leading role in implementing inter-block multiplexing procedures for steganocontent. It has been emphasized that the simultaneous use of 2-level data multiplexing significantly extends the capabilities to withstand content attack attempts. It has been found that the use of blocks of higher dimensionality, significantly reduces the role of the "runs length" parameter in breaking the structure of the original images. It is stated that increasing the length of a sample stack of runs expands the potential combinatorial multiplexing of the valid pairs of runs parameters and destroys the correlation between the elements of the original data set to a greater extent. Based on the simulation results, it has been concluded that the introduction of different methods of runs scanning provides additional position in the structure of the key of the data extractor.

Keywords: image; steganography; container; content; image smoothing; visual visibility of distortions; series coding; multiplexing.

УДК 621.391 DOI: 10.26565/2519-2310-2022-1-05

ОСОБЛИВОСТІ ІНТЕГРАЦІЇ СИСТЕМ ЗАХИСТУ ВІД НЕСАНКЦІОНОВАНИХ ДІЙ В СУЧАСНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

Ольга Мелкозьорова, Юлія Лєсная, Сергій Малахов

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

Надійшла: серпень 2022. Прийнята: серпень 2022.

Анотація: *Метою даного матеріалу є стислий розгляд основних варіантів інтеграції елементів систем (підсистем) захисту від несанкціонованих дій (НСД) до складу інформаційних систем різного призначення. Відзначено, що ступінь і спосіб взаємної інтеграції основних систем (базової системи (тієї, що захищається) та, власне системи/підсистеми захисту) є наслідком проекції реалізованих ієрархічних відносин між ними. Звернено особливу увагу на те, що залежно від умов експлуатації і цільового призначення базової системи, можлива значна ре конфігурація логіки їх взаємовідносин у частині глибини взаємного контролю та можливостей блокування заданих функцій управління (критичних процедур або процесів). Підкреслено, що при загальній схожості базових ідей та цільових установок, особливості проектування підсистем (засобів) захисту від НСД, у кожному конкретному випадку мають свою яскраво виражену специфіку та обмеження. Акцентовано увагу на те, що заявлений рівень легітимації основних процедур управління визначає потрібний рівень інтеграції відповідних систем (систем, котрі поєднуються). Зроблено висновок, що рівень функціональної залежності підсистеми захисту від НСД, від поточних режимів роботи базової системи та дій персоналу, визначається переліком і змістом покладених на неї задач.*

Ключові слова: *інформаційні системи; делегування повноважень; ланки управління; сумісні дії; санкціонування;НСД; інформаційна безпека.*

1. Вступ

У ході розробки та модернізації інформаційних систем (ІС) різного призначення, котрі забезпечують виконанні особливо відповідальних технологічних процедур і процесів, питання щодо підтримки заданого рівня контролю та безпеки реалізації найбільш значущих *(критичних*) процедур управління є найбільш принциповими. Як свідчить відомий досвід, саме від реалізованого рівня легітимації і контролю виконання критичних процедур управління, зрештою, залежить фактичний рівень безпеки застосування відповідних систем та/або комплексів автоматизації. Ця ситуація однаковою мірою характерна, як при вирішенні завдань управління технічними засобами і технологічними процесами, так і під час реалізації систем, де об'єктами управління виступають безпосередньо люди [1-2].

В якості прикладів необхідності використання підсистем *(або засобів*) захисту від несанкціонованих дій (НСД) можна навести наступні:

– підтвердження процедур делегування *(тобто, тимчасової передачі)* повноважень управління заданим категоріям персоналу базових ІС, котрі задіяні при виконанні критичних циклів управління;

– санкціонування виконання особливо важливих процедур управління *(наприклад, підтвердження індикативних грошових транзакцій, дистанційне підтвердження виконання процедури знищення спеціальних баз даних тощо)*;

– зміна поточних режимів роботи контрольованих технічних засобів та/або критичних технологічних процесів (*наприклад, зміна поточного режиму роботи енергоагрегату)*;

– санкціонування доступу персоналу до «контрольованих» вантажів та критичних об'єктів техногенної інфраструктури *(віддалене спільне розблокування електронних пломб та елементів доступу до технічних об'єктів*);

– санкціонування доступу обслуговуючого персоналу до зміни діючих параметрів роботи елементів підсистеми захисту від НСД на об'єктах базової ІС *(у тому числі розблокування кодоблокуючих пристроїв підсистеми (або засобів) захисту від НСД);*
– централізоване «скидання» поточних блокувань кодоблокуючих пристроїв підсистеми захисту від НСД, які здійснені внаслідок фіксації спроб реалізації несанкціонованих дій (*наприклад, при підтвердженні фактів ненавмисного порушення порядку реалізації контрольованих процедур управління в межах передбачених циклів управління базової ІС*);
– централізоване блокування апаратури та/або доступу до даних на контрольованих ланках управління базової системи (*або окремих об'єктів базової ІС)* при спробах компрометації елементів підсистеми та/або засобів захисту від НСД;
– підтвердження процедури видачі разових повноважень персоналу нижніх ланок управління базової ІС, що мають тимчасові чи інші обмеження *(наприклад, з геолокації та/або кратності їх використання)*;
– санкціонування змін в діючій конфігурації структури системи управління базової ІС *(наприклад, активація режиму роботи через інстанцію або запуск гіпервізорів функціонального VR-розширення окремих елементів базової ІС*);
– спільне «скидання» параметрів *(програмованих уставок)* таймерів контролю виконання критичних процедур та ін.

Підтримка розглянутого вище функціоналу забезпечується шляхом комплексної інтеграції до складу базової ІС, відповідних елементів підсистеми захисту від НСД (*чи засобів санкціонування повноважень*). При цьому під «легітимацією процедур управління» слід розуміти процес надання необхідного рівня гарантій щодо безумовної відповідності реалізованих процедур управління вимогам технічної, експлуатаційної та нормативної документації.

Забезпечення зазначених гарантій виконується шляхом реалізації комплексу організаційно-технічних заходів, які передбачають глибоку взаємну інтеграцію процесів інформаційної взаємодії систем, що сполучаються [1-2].

2. Основна частина

Розглянемо найбільш характерні варіанти інтеграції елементів двох систем, які забезпечують різний рівень взаємної транспарентності і ієрархічних відносин між базовою ІС та підсистемою (та/або комплексом засобів) захисту від НСД.

На рис. 1 представлені найбільш характерні варіанти інтеграції елементів системи захисту від НСД до складу базової ІС. Із представлених схем слід, що підвищення вимог до рівня функціональних можливостей і ступеня автономності навіть найпростішої реалізації комплексу засобів захисту від НСД *(див. рис. 1(а))*, поетапно підвищує статус цих засобів на рівень окремої підсистеми у складі базової ІС (*рис. 1(б)) – 1(в)*), доводячи її можливості до рівня незалежної системи (рис. 1(г)), яка має абсолютний пріоритет, стосовно контролю заданого переліку функціональних завдань базової ІС.

У варіанті, наведеному на рис. 1(а), елементи (*або програмне забезпечення)* комплексу засобів захисту від НСД, є складовою частиною окремих підсистем у складі базової ІС. Такий варіант реалізації захисту є не більш, ніж продовженням частини функцій наявних підсистем ІС, у які вони інтегровані, та не передбачає будь-якого «зовнішнього» (*технологічного*) входу. Така реалізація захисту від НСД можлива на етапі модернізації спеціального про-

грамного забезпечення відповідних підсистем базової ІС, але її можливості обмежуються рівнем логічних блокувань, із усіма наслідками, що звідси випливають…

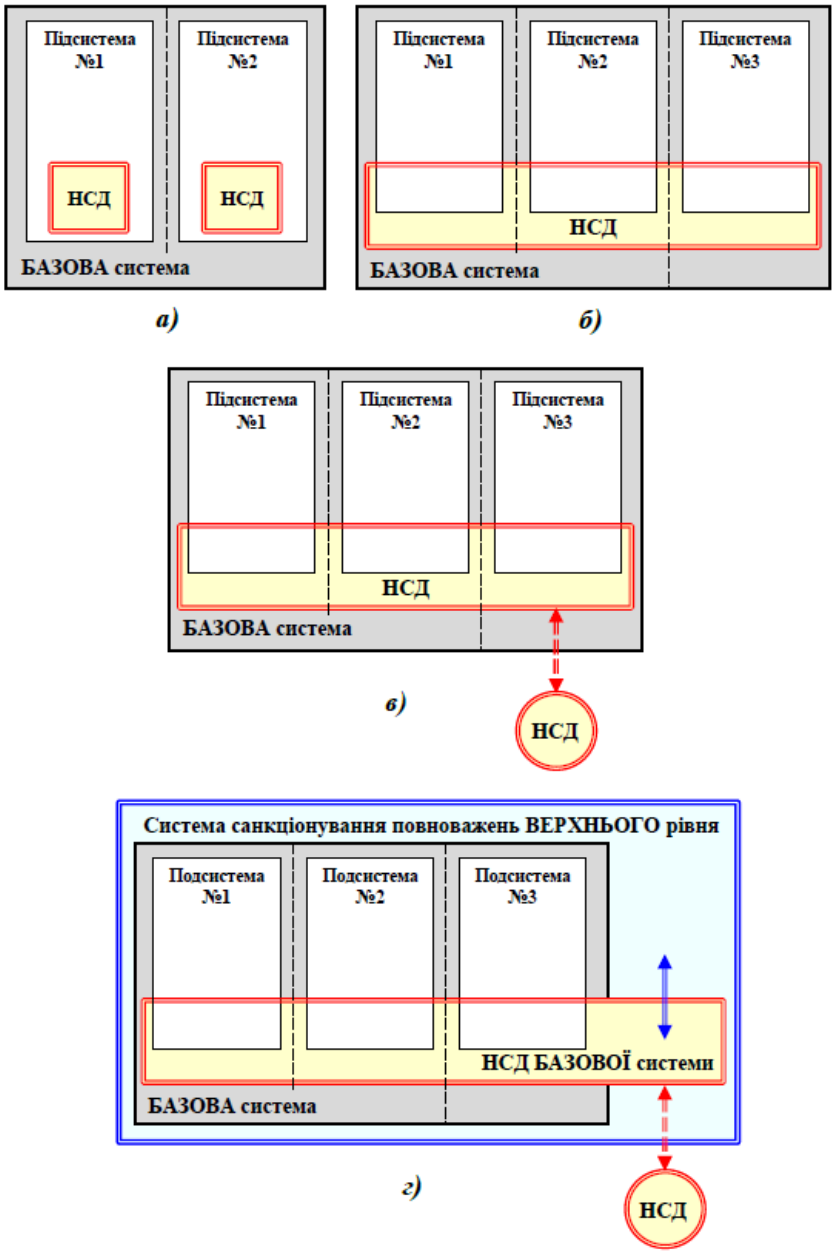


Рис. 1 – Варіанти інтеграції систем

Варіант взаємодії, представлений на рис.1(б) декілька розширює можливості комплексу засобів захисту від НСД до рівня однієї з підсистем в складі базової ІС. У цьому випадку логіка роботи складових елементів підсистеми захисту підпорядковується загальній логіці роботи базової ІС. Відповідно, перебуваючи на вищому ієрархічному рівні, базова ІС є провідною в частині регламентації складу функціональних завдань для підсистеми захисту від НСД, забезпечуючи можливість її діагностики та зміни параметрів налаштування. Принциповою відмінністю варіанта *1(б)* від *1(а)* є те, що в даному випадку засоби підсистеми захисту від НСД контролюють «точки» сполучення *(інтерфейси взаємодії*) складових підсистем базової ІС. За рахунок цього, підвищується загальний рівень безпеки та усувається можливість створення обхідних ланцюгів (*в т.ч. можливість імітації або програмної емуляції окремих елементів*) у найважливіших сегментах базової системи.

У даному випадку виконавчі елементи підсистеми захисту можуть вводити необхідні блокування *(або навпаки ініціювати запуск відповідних процесів, наприклад, знищення даних)* на рівні взаємодії окремих підсистем базової ІС, залежно від заданих для неї (підсистеми захисту) критеріїв оцінки поточного ступеня загроз, щодо реалізації НСД.

Такими параметрами можуть бути:

– перевищення заданої кількості спроб введення нелегітимних *(в т.ч. помилкових)* команд управління в межах здійсненні циклів управління базової ІС;

– порушення порядку спільних дій персоналом базової ІС під час реалізації особливо важливих *(критичних)* процедур управління;

– перевищення встановлених значень процедурних тайм-слотів, в т.ч. при реалізації *критичних* процедур управління базової ІС;

– порушення порядку доступу до елементів підсистеми захисту від НСД *(в т.ч. порушення цілісності інтерфейсів та/або порядку обміну інформацією між елементами підсистеми захисту, що розміщуються на різних рівнях (та/або підсистемах) базової ІС)* та ін.

Варіант інтеграції, що представлений на рис.1(в), знижує ступінь функціональної залежності підсистеми захисту від особливостей поточних режимів роботи і дій персоналу базової системи. У даному випадку система захисту структурно є підсистемою базової ІС, проте має незалежний від неї канал взаємодії *(входу в систему захисту)* із «зовнішніми» елементами цієї ж підсистеми. Залежно від особливостей реалізації базової ІС (*топологія, параметри розміщення інфраструктури, характеристики мобільності основних елементів та ін.*), у якості таких «зовнішніх» елементів може виступати автономна консоль обслуговування підсистеми захисту від НСД (*червоне коло, що позначено «НСД» на рис.1(в)*). Поява цього елемента в загальній структурі засобів управління «виводить» з функцій базової системи такі можливості, як:

– локальне управління параметрами роботи підсистеми захисту від НСД (*в межах діючих повноважень для обслуговуючого її персоналу*);

– робота з даними log-файлів підсистеми захисту *(тобто функції аудиту інцидентів)*;

– розблокування (*локально або спільно з персоналом даної ланки управління базової ІС*) кодоблокуючих пристроїв підсистеми захисту від НСД;

– можливість взаємодії з персоналом вищої ланки управління базової ІС, минаючи залучення персоналу цієї ланки (*об'єкта*) управління

– оновлення програмного забезпечення елементів підсистеми захисту та ін.

Таким чином, в даному випадку, базова система втрачає можливість адміністрування функцій для підсистеми захисту від НСД, зберігаючи лише логічну взаємодію спеціальних алгоритмів у циклах контролю виконання критичних процедур управління.

Зрештою, рис.1(*г*) відображає варіант інтеграції систем, при якому підсистема захисту від НСД фактично виведена за рамки функціональної залежності від базової ІС, а їх взаємодію слід розглядати, як завдання з поєднання двох практично незалежних систем. В даному випадку система захисту від НСД, що інтегрується з базовою ІС, може мати не тільки локальний канал взаємодії зі своїми «зовнішніми» елементами (рис.1(*в*)), але й бути частиною іншої, ієрархічно більш високорівневої системи, яка вирішує завдання загального контролю та легітимації циклів управління для цілої множини систем, що об'єднані спільними цілями та/або функціональними задачами.

У загальному випадку, розгляд тематики питань, що розглядаються, диктує необхідність проведення відповідного аналізу загроз, щодо забезпечуваного рівня легітимації про-

цедур формування і виконання найбільш важливих (критичних) команд управління. Існування подібних загроз багато в чому обумовлено виникненням відповідних передумов здійснення НСД, характерних для різних умов експлуатації та застосування за призначенням, як самої базової системи в цілому, так і окремих засобів автоматизації, що входять до її складу [3]. При цьому, визначення передумов здійснення НСД та подальший всебічний аналіз першопричин їх появи, слід проводити спираючись на результати широкої систематизації досвіду розробки і експлуатації відповідних ІС, та засобів захисту від НСД, з одного боку, і аналітичного прогнозу їх подальшого розвитку, з іншої сторони.

3. Висновки

1. При загальній схожості базових ідей та цільових установок, особливості проектування підсистем (засобів) захисту від НСД, у кожному конкретному випадку мають свою яскраво виражену специфіку та обмеження.

2. Декларований рівень легітимації основних процедур управління визначає необхідний рівень інтеграції аналізованих систем.

3. Рівень функціональної залежності підсистеми захисту від НСД від поточних режимів роботи базової системи та дій персоналу визначається переліком та змістом покладених на неї задач, а також ступенем автономізації основних функцій захисту.

4. Розміщення виконавчих елементів підсистеми захисту від НСД у місцях сполучення основних підсистем базової ІС, при одночасному підвищенні рівня автономізації її основних функцій, є найбільш правильною стратегією при проектуванні та створенні таких систем.

5. Оперативне розблокування виконавчих елементів підсистеми захисту від НСД, можливе шляхом реалізації спільних дій персоналу скомпрометованої системи та представників органів управління верхніх ланок управління базової ІС (*режим спільного розблокування*).

6. Актуальність розглянутої проблематики обумовлена впливом 5 основних факторів:
   – суттєвими змінами у змісті та формах реалізації процедур управління;
   – тенденцією до розосередження основних елементів систем, що захищаються;
   – підвищенням вимог до рівня компетентності персоналу;
   – високим ступенем технологічних ризиків сучасних техногенних комплексів (*енерговузли, транспортні системи, хімічні виробництва та ін.*) при збереженні значного рівня внутрішніх загроз (*інсайд, саботаж і т.п.*);
   – масштабністю та швидкоплинністю настання наслідків за успішної реалізації НСП, безвідносно їх субстантивного змісту.

Список літератури

[1] Сербин, В. & Малахов, С. Захист від несанкціонованих дій в сучасних інформаційних системах. Проблеми інформатизації: матеріали 7-ї міжнар. наук.-техн. конф., 13-15 листопада 2019 р. Харків, Україна. Вилучено з http://repository.kpi.kharkov.ua/handle/KhPI-Press/42752

[2] Сербин, В. & Малахов, С. Особливості інтеграції підсистем захисту від несанкціонованих дій в сучасних інформаційних системах і комплексах автоматизації. Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: матеріали 11-ї міжнар. наук.-техн. конф., 08-09 квітня 2021 р. Харків, Україна: ДП «Південний державний проектно-конструкторський та науково-дослідницький інститут авіаційної промисловості». Вилучено з http://repository.kpi.kharkov.ua/handle/KhPI-Press/52020

[3] Мелкозьорова, О., Лєсная, Ю., & Малахов, С. (2022). Особливості забезпечення захисту від НСД в сучасних інформаційних системах. InterConf, (97). Вилучено з https://ojs.ukrlogos.in.ua/index.php/interconf/article/view/18428

Received: on August 2022. Accepted: on August 2022.

Authors:

Olha Melkozerova, Ph.D., Associate Professor Department of Security of Information Systems and Technologies, V.N. Karazin Kharkiv National University, Ukraine.

ORCID ID https://orcid.org/0000-0002-1134-2925
E-mail: olha.melkozerova@karazin.ua

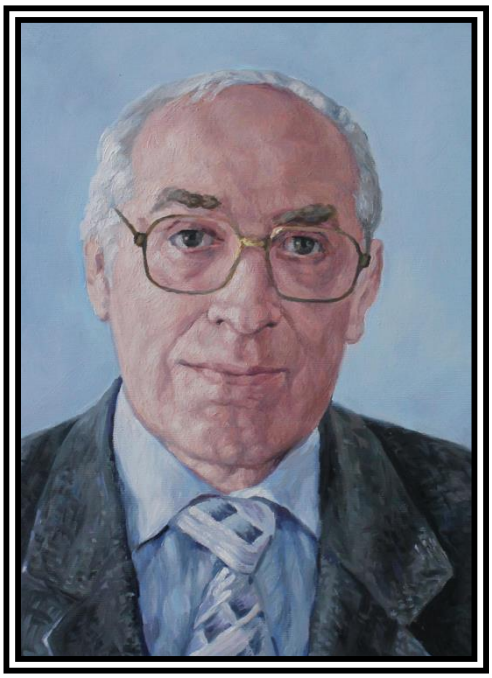
Yuliia Liesnaia, student of the Department of Security of Information Systems and Technologies, V.N. Karazin Kharkiv National University, Ukraine.
E-mail: xa12284109@student.karazin.ua

Serhii Malakhov, Ph.D., Senior Researcher, Computer Science Department, V.N. Karazin Kharkiv National University, Ukraine.
ORCID ID https://orcid.org/0000-0001-8826-1616
E-mail: mailgate@meta.ua

Peculiarities of the integration of systems of protection against unsanctioned actions in modern information systems.

Abstract. The purpose of this material is a brief review of the main options for integrating elements of systems (subsystems) of protection against unsanctioned activities (NSA) into information systems (IS) for various purposes. It is noted that the degree and method of mutual integration of the main systems are the result of the projection of the realized hierarchical relations between them. Attention is drawn to the fact that, depending on the operating conditions and the purpose of the base system, a significant reconfiguration of the logic of their relationship is possible, regarding the depth of mutual control and the possibilities of blocking the specified control functions (and/or critical processes). It is emphasized that with the general similarity of the basic ideas, the specific features of the design of protection subsystems against NSA in each case have their own specifics and limitations. Attention is focused on the fact that the declared level of legitimation of management procedures determines the required level of integration of interfaced systems (device). It is concluded that the level of functional dependence of the protection subsystem on the current modes of operation of the basic information system and the actions of its personnel is determined by the content of the (NSA) tasks assigned to it.

Keywords: information systems; delegation of authority; control link; joint actions; sanctioning; unsanctioned activities (NSA); information security.



ДОЛГОВ Віктор Іванович

(21.11.1938 – 21.07.2022)

Редакційна колегія журналу «Комп'ютерні науки та кібербезпека» з глибоким сумом сповіщає, що важка хвороба забрала життя Долгова Віктора Івановича, професора кафедри безпеки інформаційних систем і технологій, доктора технічних наук , Заслуженого діяча науки і техніки УРСР.

Віктор Іванович Долгов народився 21 листопада 1938 року у м. Ленінград (*нині – Санкт-Петербург*) в родині військовослужбовця. У 1961 році він закінчив з відзнакою Харківське вище авіаційно-інженерне військове училище за спеціальністю «*Спеціальні системи радіоуправління і контролю*», а у 1966 році – вечірнє відділення Харківського державного університету (*нині – Харківський національний університет імені В.Н.Каразіна*) за спеціальністю «Математика».

Від 1961 року Віктор Іванович працював у Харківському вищому військовому командно-інженерному училищі (*нині Харківський університет Повітряних Сил*), там же у 1972 закінчив докторантуру. Віктор Іванович Долгов – д.т.н. (1973), звання професора отримав у 1978 р. У 1976–1985 рр. Віктор Іванович очолював кафедру зв'язку Харківського вищого військового командно-інженерного училища ім. М.І. Крилова; у 1985–1992рр. він обіймав посаду заступника начальника з навчальної та наукової роботи.

У 1991 році Віктору Івановичу було присвоєно почесне звання «Заслужений діяч науки і техніки Української РСР». З 1991року В.І.Долгов працював професором кафедри № 42 Харківського військового університету. Пізніше професор Долгов викладав у Харківському державному технічному університеті радіоелектроніки: 1995-1998 рр. – на кафедрі ЕОМ, 1999-2001рр. – на кафедрі безпеки інформаційних технологій.

З 2014 року і до останніх днів Віктор Іванович Долгов працював у Харківському національному університеті імені В.Н. Каразіна професором кафедри безпеки інформаційних систем і технологій, викладав курс «Криптографічні методи в кібербезпеці».

Віктор Іванович - автор 6 підручників і монографій, більш ніж 200 статей, понад 80 винаходів і патентів.

Багато сил віддавав Віктор Іванович навчанню та вихованню студентів і аспірантів, сприяв їх науковому зростанню. Безпосередньо ним було підготовлено більше 20 кандидатів та 4 доктора технічних наук.

Віктор Іванович був майстром спорту з класичної боротьби, з молодості захоплювався альпінізмом. Неодноразово з групою спортсменів він підкорював Кавказькі гори.

21 липня 2022 року, після тривалої мужньої боротьби з хворобою, Віктор Іванович пішов з життя. Для більшості з тих, хто особисто знав Віктора Івановича, був Вчителем, прикладом справжнього вченого й допитливого дослідника, глибоко інтелігентною та порядною людиною.

ISSN 2519-2310 DOI: 10.26565/2519-2310-2022-1

Journal Requisites

Publishing, cover design: V.N. Karazin Kharkiv National University, 2022

Наукове видання

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА

Випуск 1(21) 2022

Міжнародний електронний науково-теоретичний журнал

Англійською, українською, *та ін. мовами*

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

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

V. N. Karazin Kharkiv National University Publishing

CS&CS

2022