Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи магістра

галузь знань *12 Інформаційні технології*
(шифр і назва галузі знань)

спеціальність *125 Кібербезпека*
(код і назва спеціальності)

освітній ступень *магістр*

освітньо-наукова програма *Кібербезпека*
(назва освітньої програми)

на тему: «Гібридний метод детектування фішингових веб сайтів»

Виконавець: студент ІІ курсу, групи КБм-21

(підпис) **Дмитро ШУТЕНКО**
(Ім'я, ПРІЗВИЩЕ)

	Ім'я, ПРІЗВИЩЕ	**Підпис**
Науковий керівник	Сергій БУЧИК	
Нормоконтроль	Олена БОГУСЛАВСЬКА	

Київ 2023

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
В.о. завідувача кафедри
кібербезпеки
та захисту інформації
_____ Сергій ТОЛЮПА
«24» жовтня 2022 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *125 Кібербезпека*
(код і назва спеціальності)

освітній ступень *магістр*

Здобувача КБм-21 (група) Шутенка Дмитра Валентиновича (прізвище ім'я по-батькові)

Тема дипломної роботи Гібридний метод детектування фішингових веб сайтів

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Рішення засідання кафедри кібербезпеки та захисту інформації факультету інформаційних технологій протокол № 3 від 20.10.2022

2. МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень	Процес виявлення фішингових елементів в веб сайтах.
Предмет досліджень	Методи виявлення фішингових веб сайтів.
Мета	Розробка системи детектування фішингових веб сайтів.
Вихідні дані для проведення роботи	Методи детектування фішингових елементів в веб сайтах.

3. ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Наукова новизна	вперше запропоновано гібридний методи виявлення фішингових елементів в веб сайтах за допомогою поєднання аналізу адреси, інформаційного наповнення веб сайту та проведення ряду евристичних перевірок за рахунок оцінки широкого спектру індикаторів компрометації фішингу та методологію оцінки системи побудованої на його основі.
Практична цінність	розробка вітчизняної системи захисту від фішингових веб сайтів у організаціях різних форм власності.

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Робота виконана у повному обсязі відповідно до теми.

5. ЕТАПИ ВИКОНАННЯ РОБОТИ

Найменування етапів робіт	Строки виконання робіт (початок-кінець)
Розробка плану для досягнення мети роботи	24.10.2022 – 23.01.2023
Аналіз літературних джерел	24.01.2023 – 14.02.2023
Розробка власного методу детектування фішингових веб сайтів	15.02.2023 – 24.04.2023
Оформлення і друк пояснювальної записки	25.04.2022 – 19.05.2023

6. РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект	Зниження збитків асоційованих з шахрайськими діями через фішингові веб сайти.
Соціальний ефект	Покращення технологій забезпечення захисту користувачів в організаціях різних форм власності.

7. ДОДАТКОВІ ВИМОГИ

Завдання видав _____ Сергій БУЧИК
(підпис) (Ім'я, ПРІЗВИЩЕ)

Завдання прийняв до виконання _____ Дмитро ШУТЕНКО
(підпис) (Ім'я, ПРІЗВИЩЕ)

Дата видачі завдання: 24.10.2022 р.
Термін подання дипломної роботи до ЕК 19.05.2023 р.

УДК. 004.432.16

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Гібридний метод детектування фішингових веб сайтів»: 94 сторінки, 15 рисунків, 3 додатки та 10 таблиць. 67 літературних джерел.

Об’єкт дослідження – процес виявлення фішингових елементів в веб сайтах.

Мета роботи – розробка системи детектування фішингових веб сайтів. Методи дослідження:

1. у розділі 1 було проведено аналіз методів соціальної інженерії та підходів до здійснення фішингових атак шляхом застосування структурного аналізу, методу порівняння та системного підходу;
2. у розділі 2 було описано цикл атак соціальної інженерії, здійснено вибір фази для детектування атак і проведено огляд методів протидії фішинговим атакам шляхом застосування методу класифікації;
3. у розділі 3 здійснено аналіз існуючих підходів для виявлення фішингових веб сайтів та запропоновано покращення; описано теоретичні засади створення системи детектування фішингових елементів у веб сайтах; пояснено принципи роботи 3-ох її компонентів, запропоновано методологію оцінки такої системи шляхом застосування функціональних методів дослідження та наукового моделювання;
4. у розділі 4 було обрано середовище виконання на методи розробки технічної реалізації запропонованої системи; описано розроблені користувацькі інтерфейси; описано етапи проведення дослідної експлуатації таких систем; для формування пропозиції щодо покращення розробленої системи було застосовано емпіричний науковий метод дослідження.

У роботі досліджено сучасні загрози та методи протидії фішинговим атакам. Проведено аналіз наукових доробок та ринку рішень, що проводять аналіз веб сайтів на предмет наявності фішингових елементів. Запропоновано гібридний метод

детектування фішингових елементів у веб сайтах. Побудовано систему виявлення фішингових атак на базі запропонованого методу.

Наукова новизна: вперше запропоновано гібридний метод виявлення фішингових елементів в веб сайтах за допомогою поєднання аналізу адреси, інформаційного наповнення веб сайту та проведення ряду евристичних перевірок за рахунок оцінки широкого спектру індикаторів компрометації фішингу та методологію оцінки системи побудованої на його основі.

Актуальність теми: Фішинг є однією з найбільш часто використовуваних типів атак для кожної організації. Традиційний набір засобів захисту користувачів, побудований на блеклістингу та аналізі декількох властивостей веб сайту не здатний протистояти сучасним загрозам. Системи побудовані на вірогідносному аналізі елементів веб сайту з використанням математичних моделей побудованих за допомогою машинного навчання та/або штучним інтелектом виявляються найефективнішим механізмом виявлення фішингових атак нульового дня. Тому представлення системи в рамках якої детектування та захист реалізований за допомогою комбінації різних підходів є досить актуальною темою для наукового дослідження.

Ключові слова: соціальна інженерія, атака, фішинг, загроза, метод, захист інформації, детектування, фішингові елементи, URL, HTML, DOM, евристика.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ПЗ	–	Програмне забезпечення
ПК	–	Персональний комп’ютер
СІ	–	Соціальна інженерія
БД	–	База даних
ІБ	–	Інформаційна безпека
АС	–	Автоматизована система
HTTPS	–	HyperText Transfer Protocol Secure
SaaS	–	Software as a Service
FBI	–	Federal Bureau of Investigation
URL	–	Uniform Resource Locator
API	–	Application programming interface
DNS	–	Domain Name System
IT	–	Information Technology
DOM	–	Document Object Model
CSS	–	Cascading Style Sheets

ЗМІСТ

ВСТУП

Актуальністю даної роботи є розробка системи в рамках якої детектування фішингових елементів у веб сайтах реалізоване за допомогою комбінації підходів блеклістингу, аналізу їх інформаційного наповнення та проведення ряду евристичних перевірок.

Основні задачі кваліфікаційної роботи:

- Проаналізувати літературні джерела, закони та підзаконні акти України, міжнародні документи та статті у сфері визначення, виявлення та протидії атакам, шо використовують методи соціальної інженерії та фішингу зокрема.
- Проаналізувати існуючі методи виявлення фішингових елементів у веб сайтах.
- Розробити гібридний метод детектування ознак фішингу в веб сайтах.
- Запропонувати оцінку якості розробленої системи виявлення фішингових веб сайтів.
- Формалізувати задачу створення технічного рішення для виявлення ознак фішингу в веб сайтах у термінах комп’ютерного моделювання.
- Обґрунтовано підібрати середовище виконання; побудувати та описати роботу кожного з компонентів системи.
- Розробити технічне рішення для детектування фішингових веб сайтів.
- Провести оцінку розробленої системи та провести дослідну експлуатацію.

Науковою новизною цієї кваліфікаційної роботи є вперше запропонований гібридний метод виявлення фішингових елементів в веб сайтах за допомогою поєднання аналізу адреси, інформаційного наповнення веб сайту та проведення ряду евристичних перевірок за рахунок оцінки широкого спектру індикаторів компрометації фішингу та методологія оцінки системи побудованої на його основі.

Події останнього десятиліття показали, що в сфері кіберзлочинності відбулися серйозні зміни. Пошук вразливостей в корпоративних мережах значно ускладнився після введення гігантами IT-індустрії кращих практик захисту корпоративних мереж

та інформаційних активів. Використовувані безпекові рішення працюють досить непогано, незважаючи на велику кількість хибних спрацювань, проте хакери адаптуються під нові технології захисту та все частіше вдаються до використання методів соціальної інженерії для отримання доступу до акаунтів користувачів в добре захищених мережах. Фішингові атаки стали однією з найпоширеніших форм кіберзлочинності, адже вони не вимагають від зловмисників високого рівня технічної оснащеності, фінансових витрат та технічних здібностей.

Об'єктом дослідження є процес виявлення фішингових елементів в веб сайтах.

Предметом дослідження є методи виявлення фішингових веб сайтів.

В даній роботі основна увага буде приділена фішинговим атакам, що реалізуються через веб сайти, які виглядають як оригінальні ресурси відомих компаній та організацій. Методи та технічні рішення, якими користуються зловмисники для досягнення своєї мети, бувають досить різноманітними, проте найчастіше їх об'єднує одна спільна деталь – ставка на людську помилку. Фішинг - це вид шахрайства, коли зловмисники намагаються отримати конфіденційну інформацію від користувачів шляхом виманювання даних, таких як логіни, паролі, номери кредитних карт та інші особисті дані.

Метою кваліфікаційної роботи є розробка системи детектування фішингових веб сайтів.

Сформовані в результаті, теоретичні та практичні рекомендації можуть бути використані, як організаційна складова в процесі роботи над створенням та впровадження даного методу розробниками інтернет браузерів та інженерами з інформаційної безпеки підприємств будь-якої форми власності, державних інституцій, тощо.

Апробація результатів роботи: матеріали кваліфікаційної роботи були презентовані та обговорювалися на V Міжнародній студентській олімпіаді "Шляхи та механізми захисту інформаційного простору України від шкідливих інформаційно-психологічних впливів" 2019 року, III міжнародній науково-практичній конференції "ПРОБЛЕМИ ТА ШЛЯХИ ЗАХИСТУ ІНФОРМАЦІЙНО-ПСИХОЛОГІЧНОЇ І ДУХОВНОЇ БЕЗПЕКИ ОСОБИ, СУСПІЛЬСТВА, ДЕРЖАВИ" 2019 року, VII

Міжнародній науково-практичній конференції “Information Technology and Interactions” 2020 року, VI Міжнародній студентській олімпіаді “Шляхи та механізми захисту інформаційного простору України від шкідливих інформаційно-психологічних впливів” 2020 року, VI Всеукраїнській науково-практичній конференції “ПЕРСПЕКТИВНІ НАПРЯМИ ЗАХИСТУ ІНФОРМАЦІЇ” 2020 року, IV Міжнародній науково-практичній конференції “Проблеми кібербезпеки інформаційно-телекомунікаційних систем” (PCSITS) 2021 року, VII Міжнародній студентській олімпіаді “Шляхи та механізми захисту інформаційного простору України від шкідливих інформаційно-психологічних впливів” 2021 року та IX Міжнародній конференції “Information Technology and Implementation” 2022 року. Основні положення дипломної роботи викладені в 8 наукових працях, серед яких: 3 реферати на міжнародних студентських олімпіадах та 5 – у матеріалах наукових конференцій, 4 з яких – міжнародні.

РОЗДІЛ 1
МІСЦЕ ФІШИНГУ СЕРЕД АТАК З ВИКОРИСТАННЯМ МЕТОДІВ СОЦІАЛЬНОЇ ІНЖЕНЕРІЇ

1.1 Соціальна інженерія як один з найрозповсюдженіших типів атак

У терміна “соціальна інженерія” існує декілька визначень, наприклад:

- Несанкціоноване отримання конфіденційної інформації або неналежних прав доступу на основі побудови довірчих відносин довіри з легітимним користувачем інформаційної системи [1].
- Змушувати людей робити те, що вони зазвичай не зробили б для незнайомця.
- Акт отримання чи спроби отримати захищені дані в іншому випадку шляхом обману особи для розкриття захищеної інформації [2].
- Практика отримання конфіденційної інформації шляхом маніпуляцій авторизованими користувачами [3].
- Наука, що вивчає методи психологічної маніпуляції над людьми для того, щоб змусити їх добровільно надавати інформацію з обмеженим доступом [4].

Соціальні інженери здатні передбачити поведінку людини, а відтак – маніпулювати її свідомістю для задоволення власних потреб [5]. У рамках соціоінженерного підходу вразливості персоналу тлумачяться як його слабкості, потреби, манії (пристрасті), захоплення. Маніпулювання ними дозволяє отримати несанкціонований доступ до інформації без руйнування та перекручування головних для нього системоутворюючих якостей. Як наслідок, це призводить до нової моделі поведінки персоналу, створення сприятливих умов реалізації загроз безпеці інформації і, як наслідок, зменшенню здатності системи захисту інформації протидіяти їх впливові. Це відображається в таких формах як, шахрайство, обман, афера, інтрига, містифікація, провокація. Використанню кожної з означених форм

маніпулювання передує визначення її змісту шляхом ретельного планування, організування та контролювання [6].

Рисунок 1.1 – Використання соціоінженерного підходу

З огляду на рис. 1.1, використання соціоінженерного підходу до оцінювання захищеності інформації в комп'ютерних системах передбачає цілеспрямований вплив на свідомість (підсвідомість) персоналу проти його волі, але за його згодою. Такий вплив дозволяє управляти поведінкою керівництва, адміністратора, користувачів через слабкості, інтереси, потреби, схильності, переконання, звички, психічний та емоційний стан. Тому маніпулювання цими вразливостями і виражається в таких формах як шахрайство, обман, афера, інтрига, містифікація, провокація. Разом з тим, використанню кожної з означених форм маніпулювання передує визначення їх сутності шляхом ретельних планування, організації та контролювання. У рамках цього підходу використання атак СІ орієнтоване на отримання "несанкціонованого" доступу до інформації при оцінюванні її захищеності шляхом негативного інформаційно-психологічного впливу на свідомість або підсвідомість персоналу. Соціальний інженер на противагу шахраю зазвичай вводить в оману, та запевняє жертву самовільно передати йому чутливу інформацію [7].

Як можемо помітити на рис. 1.2 з щорічного звіту FBI щодо кіберзлочинів зареєстрованих в США протягом 2018 - 2022 років [8], що представляє кількість жертв зареєстрованих кібератак, що використовували різні методи для порушення

властивостей інформації, атаки, що спирається на методи СІ переважають інші види в десятки разів. Пме тому в сучасному кіберпросторі в першу чергу захист від соціальних інженерів є більш пріоритетним у порівнянні з вправними хакерами.

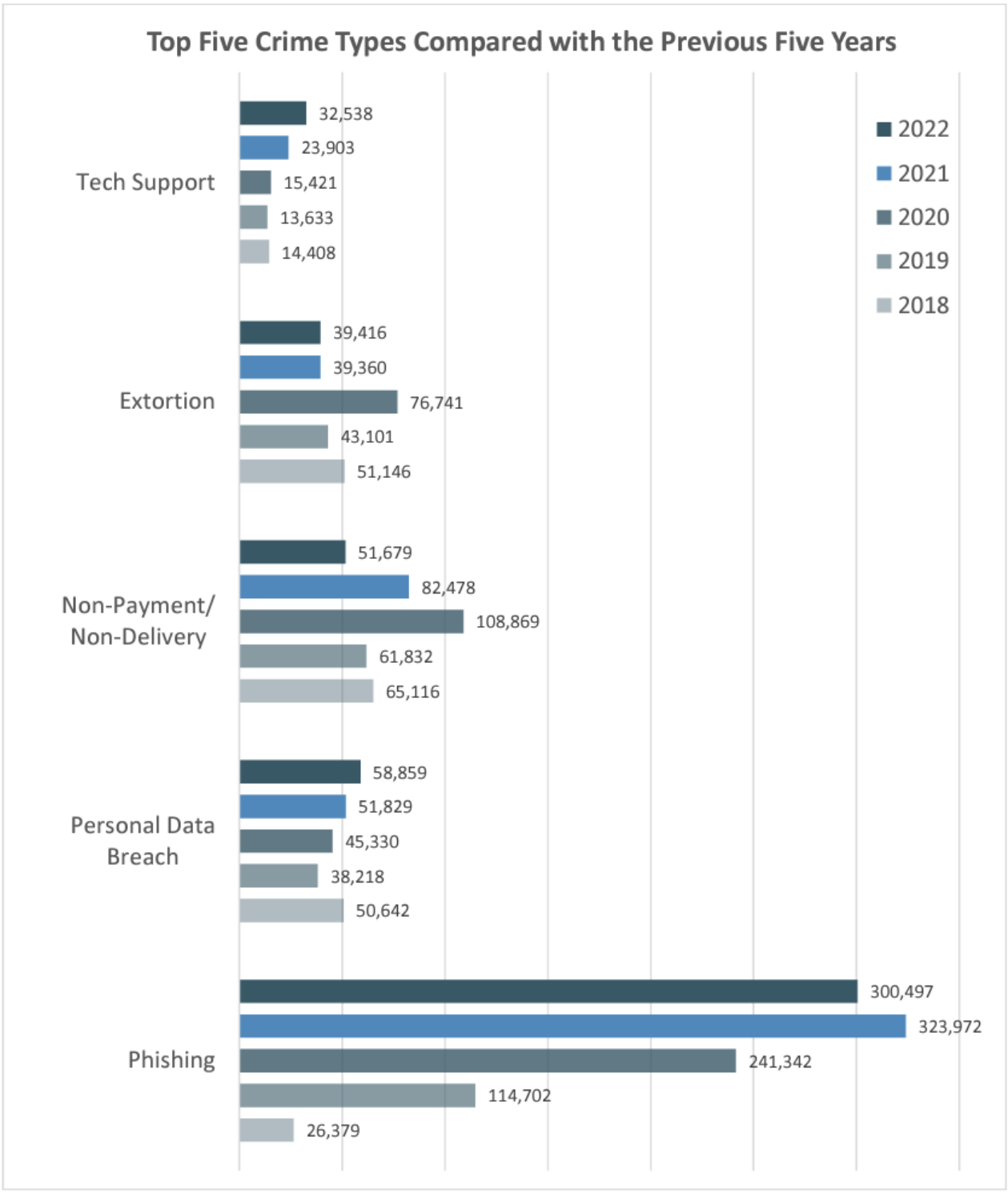

Рисунок 1.2 – Кількість жертв від атак здійснених за останні 5 років у США за категоріями

Хоча соціальна інженерія є суттєвою загрозою, організації далеко не завжди звертають на неї увагу, і не визнають потенційні великі збитки, які можуть принести атаки соціальної інженерії. У цьому розділі буде визначено актуальність теми дослідження, наведено порівняння різних тактик, якими користуються соціальні інженери, надана класифікація та аналіз підходів до проведення фішингових атак та визначення фішингу в Українському законодавстві.

1.2 Визначення актуальності теми дослідження

Сьогодні існує безліч рішень, що захищають апаратне та програмне забезпечення від вторгнення до інформаційної системи сторонніми особами та певною мірою внутрішніми агентами, але є лише обмежена кількість досліджень щодо людського фактору в інформаційній безпеці. Припускаючи, що технічні рішення для захисту інформації працюють на відмінно, певна кількість персоналу повинна мати доступ до систем для виконання своїх службових обов’язків і тим самим створює канал витоку інформації та можливість для зловмисників скомпрометувати інформаційну безпеку на підприємстві навмисно, не навмисно або шляхом маніпуляцій. Даний вид атак називають соціальною інженерією. Пом’якшення загроз, які несе ця маніпуляція, також зменшить навмисну та ненавмисну компрометацію систем та інформації. А відтак, зменшить загальний ризик.

Так само як мозок порушника виявляються безсилим перед спокусою скоєння злочину, мозок хакера прагне знайти більш витончений шлях досягнення власної мети оминаючи потужні технологічні засоби захисту. В більшості випадків це досягаються, коли ціллю зловнисника перестає бути система, а стає людина, що її використовує [7]. Історії відомі численні приклади того, як навіть найдорожчі, найскладніші та найдосконаліші системи захисту інформації опиняються безсилими перед зловмисниками, що часом використовують низько технологічні рішення задля неправомірного заволодіння інформаційними активами підприємств, державних інституцій, тощо. Загальною схемою у всіх подібних атаках було те, що на певному

етапі атаки була задіяна взаємодія людини, взаємодія, якою маніпулювали та керували, щоб отримати бажані для зловмисника результати.

Хоча основними контрзаходами проти атак соціальної інженерії є обізнаність користувачів та поєднання технічних засобів контролю [9], очевидно, що сторона, що захищається, не в змозі гарантувати цілковитий захист організації від атак з використанням методів CI. Одна з ключових причин полягає в тому, що незалежно від того, скільки навчальних програм чи технічного контролю не розгорнуто, цього завжди буде недостатньо і люди залишаються найслабшим ланкою безпеки будь-якого підприємства, а отже це і є слабкістю яку найчастіше будуть експлуатувати зловмисники, що доведено статистичними даними про методи атак на організації та державні установи в звіті компанії Positive Technologies за 2 квартал 2022 року [10]. Візуальне представлення подано на рис. 1.3 та рис. 1.4 про використовувані зловмисниками методи атак на організації та державні установи відповідно.

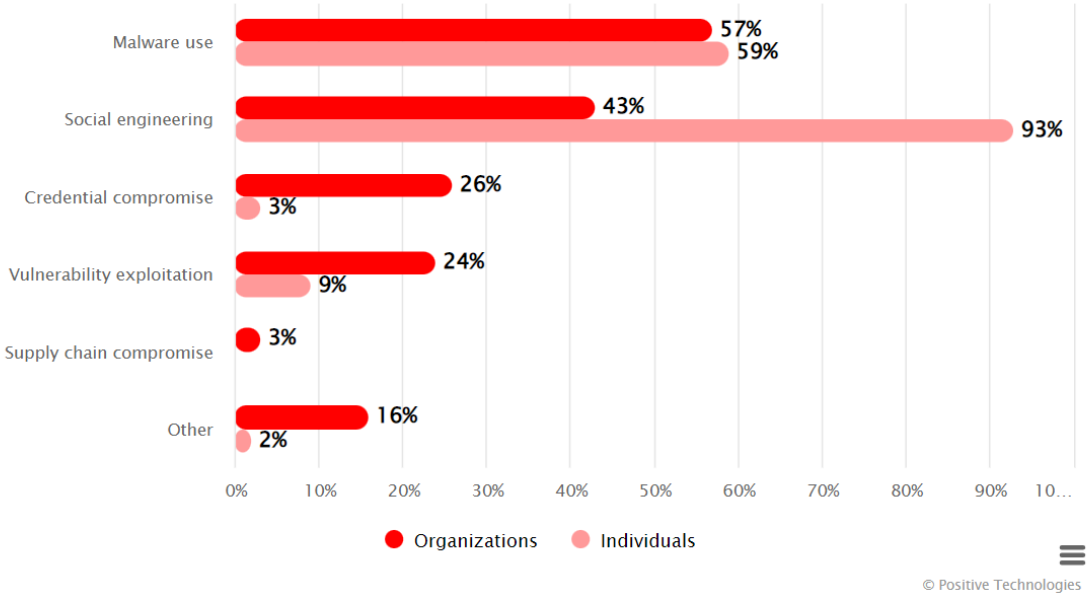

Рисунок 1.3 – Частка використання методів для атак на організації

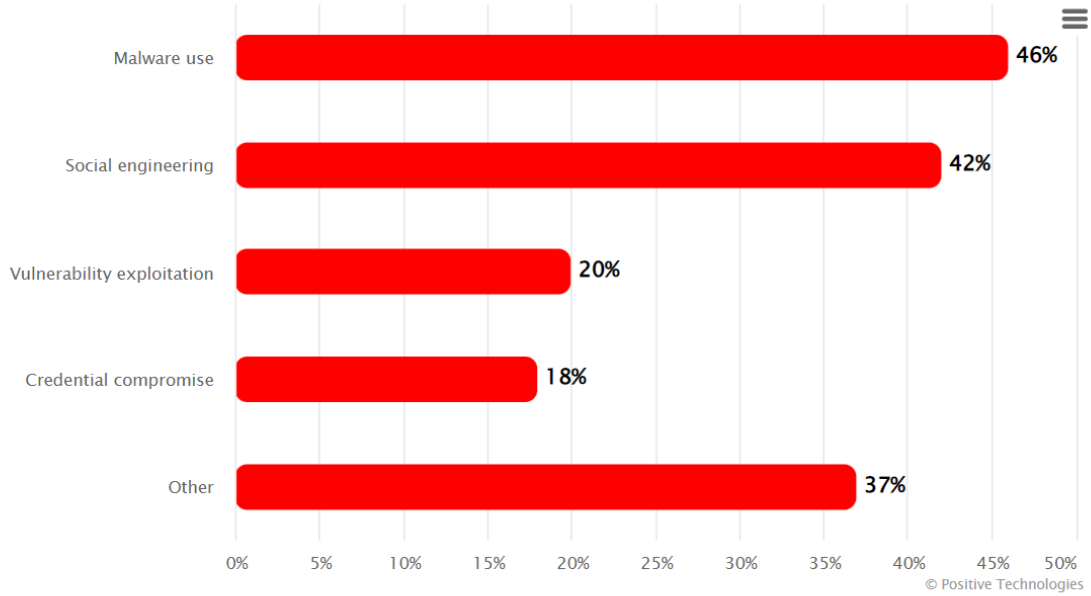

Figure 24. Methods of attacks on government (share of attacks)

Рисунок 1.4 – Частка використання методів для атак на державні установи

За словами найвідомішого хакера кінця 20 століття Кевіна Митника: “Найбільша загроза безпеці компанії – це не комп’ютерний вірус, не виправлена дірка в ключовій програмі або погано встановлений брандмауер. Насправді найбільшою загрозою можете бути ви. Я особисто переконався, що легше маніпулювати людьми, а не технологіями. Здебільшого організації не зважають на людський фактор" [7].

Як вже було зазначено, людський фактор є найслабшою ланкою в ланцюгу інформаційної безпеки, і той, над яким потрібно працювати, щоб покращити загальний стан інформаційної безпеки в цілому. Атаки з використанням СІ базуються на певних психологічних принципах і прийомах таких як нейролінгвістичне програмування, кадрування, закріплення, навіть гіпноз, тощо [11].

Проблема розробки комплексу дії для ефективної протидії атакам, що покладаються на СІ існує в урядових, бізнес та навчальних закладах по всьому світу. Не зважаючи на всі зусилля професіоналів сфери безпеки, інформація, захист якої є їх роботою все ще залишається вразливою та буде ціллю для порушників з навиками СІ допоки не буде посилена найслабша ланка безпеки – людський фактор [7].

Зараз, більше ніж будь-коли, ми маємо навчитися перестати бути самовпевненими і дізнатися більше про методи, які використовують ті, хто здійснює

атаки спрямовані на порушення цілісності, конфіденційності та доступності наших комп'ютерних систем та інформаційних ресурсів [12].

Існує чимало статей, досліджень та книг, які присвячені людському фактору або суміжним темам. Але це все ще відносно невивчена галузь наукових досліджень. Однак ці дослідження показують, що людський фактор може завдати великої шкоди організаціям, не тільки фінансовим, але й іміджу організації, що, в свою чергу, впливає на цілі та безперервність бізнес процесів і організації в довгостроковій перспективі. Той, хто не думає як реагувати у випадку настання інциденту з безпеки займає заздалегідь невдалу позицію [9].

В цілому, виникає необхідність визначення найбільш деструктивних атак, що використовують методи СІ, та розробити технічне рішення, метою якого буде автоматичне детектування та блокування спроб зловмисників експлутувати слабкості людської психології. Дипломна робота поєднує поточні дослідження соціальної інженерії та фішингу зокрема з емпіричними дослідженнями в інших областях з метою деталізованого аналізу фішинг атак та розробки технічного рішення для їх автоматичного детектування.

1.3 Визначення фішингу як найпопулярнішого методу соціальної інженерії

Одним із найбільш поширених та небезпечних видів атак соціальної інженерії є фішинг (англ. phishing, від fishing — риболовля) — вид шахрайства, метою якого є виманювання у довірливих або неуважних користувачів мережі персональних даних клієнтів онлайнових аукціонів, сервісів з переказу або обміну валюти, інтернет-магазинів. Шахраї намагаються змусити користувачів самостійно розкрити конфіденційні дані — наприклад, надсилаючи електронні листи із пропозиціями підтвердити реєстрацію облікового запису, що містять посилання на веб сайт в інтернеті, зовнішній вигляд якого повністю копіює дизайн відомих ресурсів [13]. Рис. 1.5 та рис. 1.6 ілюструють які саме типи даних зловмисники викрали при вдалих

атаках на організації приватної форми власності та приватних осіб відповідно в тому ж таки 2 кварталі 2022 року [10]. Як бачимо, зловмисників найбільше цікавлять персональні дані працівників організацій і аутентифікаційні дані приватних осіб.

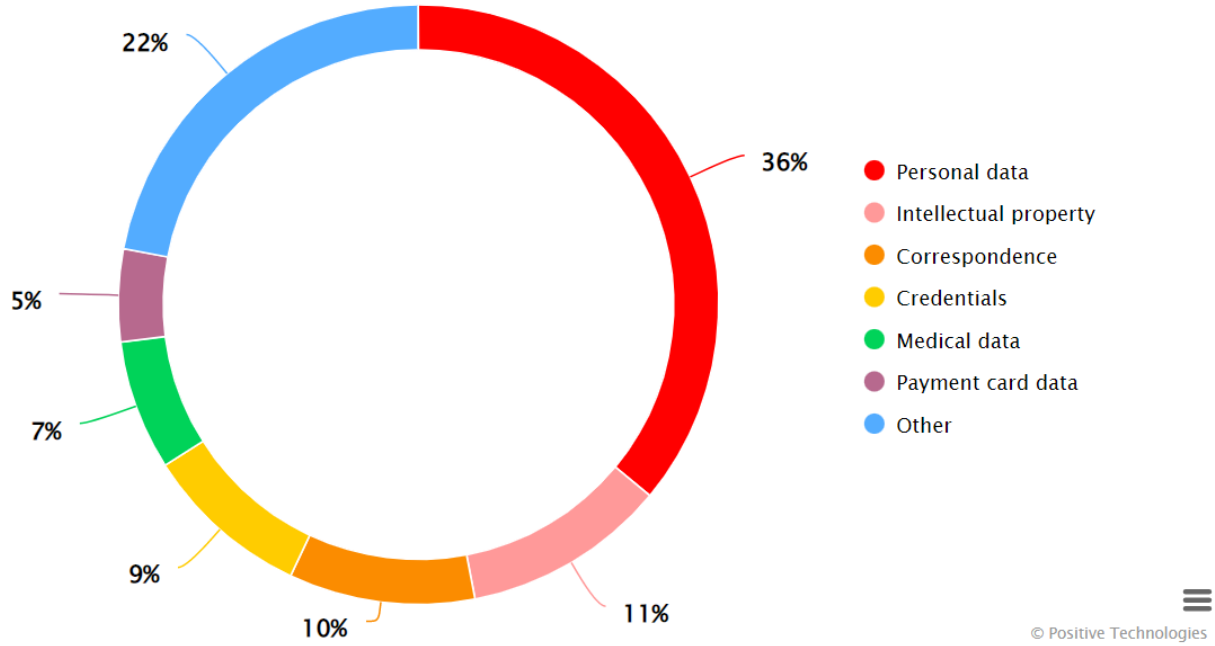

Рисунок 1.5 – Типи викрадених даних при вдалих атаках на організації приватної форми власності

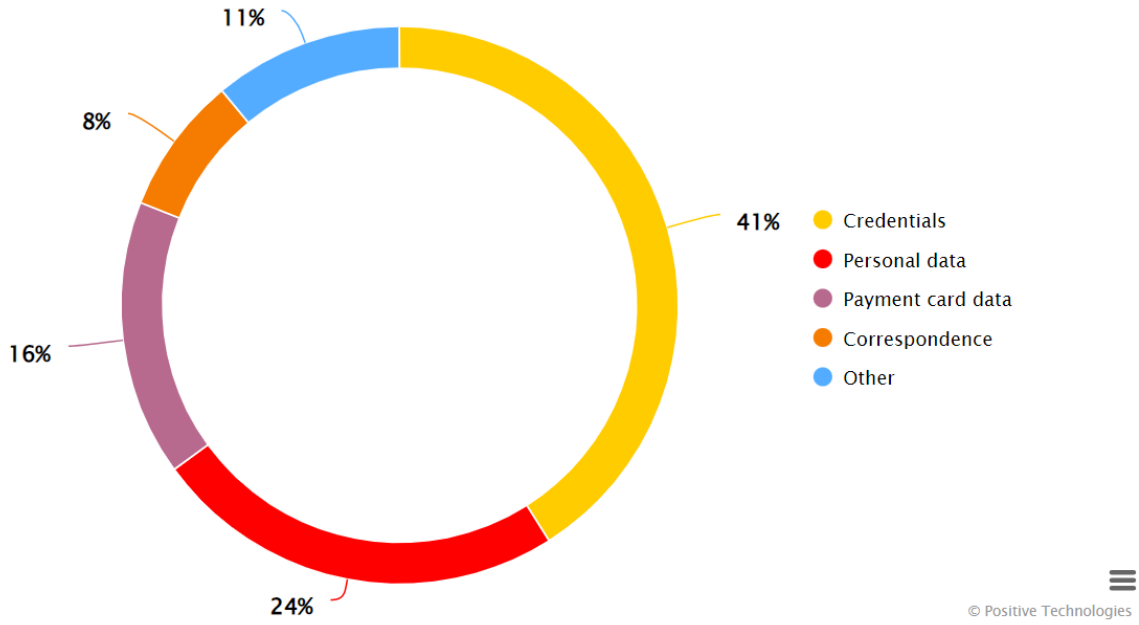

Рисунок 1.6 – Типи викрадених даних при вдалих атаках на приватних осіб

Фішинг заснований на незнанні користувачами політик корпоративної та інформаційної безпеки, процедур та правил поводження з корпоративними ресурсами та програмно-апаратними комплексами, які мають використовуватися виключно з метою виконання функціональних обов'язків [7]. Зокрема, багато хто не знає простого факту: сервіси не розсилають листів з проханнями повідомити свої ідентифікаційні дані або пройти безпідставну повторну авторизацію. Виконуючи ці і інші дії, фішинговими атаками було спричинено більше 52.000.000$ збитків відповідно до звіту FBI за 2022 рік про повідомленні кібератаки в США [8], це зображено на рис. 1.7 Приведені статистичні дані не лишають іншого варіанту, аніж як визнати фішингові атаки одними з найбільш руйнівних типів CI для організацій будь якого типу власності, державних установ та приватних осіб.

2022 CRIME TYPES continued

By Victim Loss			
Crime Type	*Loss*	*Crime Type*	*Loss*
Investment	$3,311,742,206	Lottery/Sweepstakes/Inheritance	$83,602,376
BEC	$2,742,354,049	SIM Swap	$72,652,571
Tech Support	$806,551,993	Extortion	$54,335,128
Personal Data Breach	$742,438,136	Employment	$52,204,269
Confidence/Romance	$735,882,192	Phishing	$52,089,159
Data Breach	$459,321,859	Overpayment	$38,335,772
Real Estate	$396,932,821	Ransomware	*$34,353,237
Non-Payment/Non-Delivery	$281,770,073	Botnet	$17,099,378
Credit Card/Check Fraud	$264,148,905	Malware	$9,326,482
Government Impersonation	$240,553,091	Harassment/Stalking	$5,621,402
Identity Theft	$189,205,793	Threats of Violence	$4,972,099
Other	$117,686,789	IPR/Copyright/Counterfeit	$4,591,177
Spoofing	$107,926,252	Crimes Against Children	$577,464
Advanced Fee	$104,325,444		

Рисунок 1.7 – Фінансові втрати від повідомлених атак в США за категоріями

Посилює позиції фішингу і те, що саме цим методом зловмисники найчастіше користуються аби заразити комп’ютер жертви шкідливими програмами криптографічного здирництва також відомими як програми-вимагачі. Зловмисне ПЗ цього типу шифрує дані на жорсткому диску жертви та вимагає гроші за їх розшифрування. Принаймні кількість реалізованих атак та фінансові втрати асоційовані з ними є достатнім підґрунтям для розуміння необхідності подальшого дослідження та покращення засобів автоматичного детектування фішингових атак.

1.4 Типи фішингових атак

Тема протидії фішингу на перший погляд може здаватися неосяжною, адже зловмисники використовують досить широкий спектр різноманітних атак, проте всі вони зазвичай називаються фішингом. В цьому підрозділі пропонується ознайомитися з добіркою відомих на сьогодні форм фішингових атак. Отже, за форматом проведення усі атак можна поділити на ті, які для своєї реалізація спираються переважно на технічні засоби і ті, які реалізуються завдяки методам вищезгаданої соціальної інженерії, що зображено на рис. 1.8.

Оманливий фішинг (Deceptive Phishing) – тип фішингових атаки за якої зловмисник надсилає лист електронною поштою або будь-яким іншим методом комунікації таким чином, аби інформаційне наповнення з першого погляду нагадувало листи отримані раніше від добре відомих організацій, проте в такому листі наявне посилання або прикріплений документ, при взаємодії з якими жертва перенаправляється на фішинговий сайт або заражає ПК. Використовуючи цей метод зловмисники вдаються до навичок з СІ адже вони мають переконати жертву в справжності листа та створити відповідну нагайну потребу діяти відразу, аби не давати жертві часу на роздуми. В результаті взаємодії жертви з фішинговим веб сайтом маємо передачу зловмисникам логінів та паролів до акаунту ресурсу зімітованого зловмисниками.

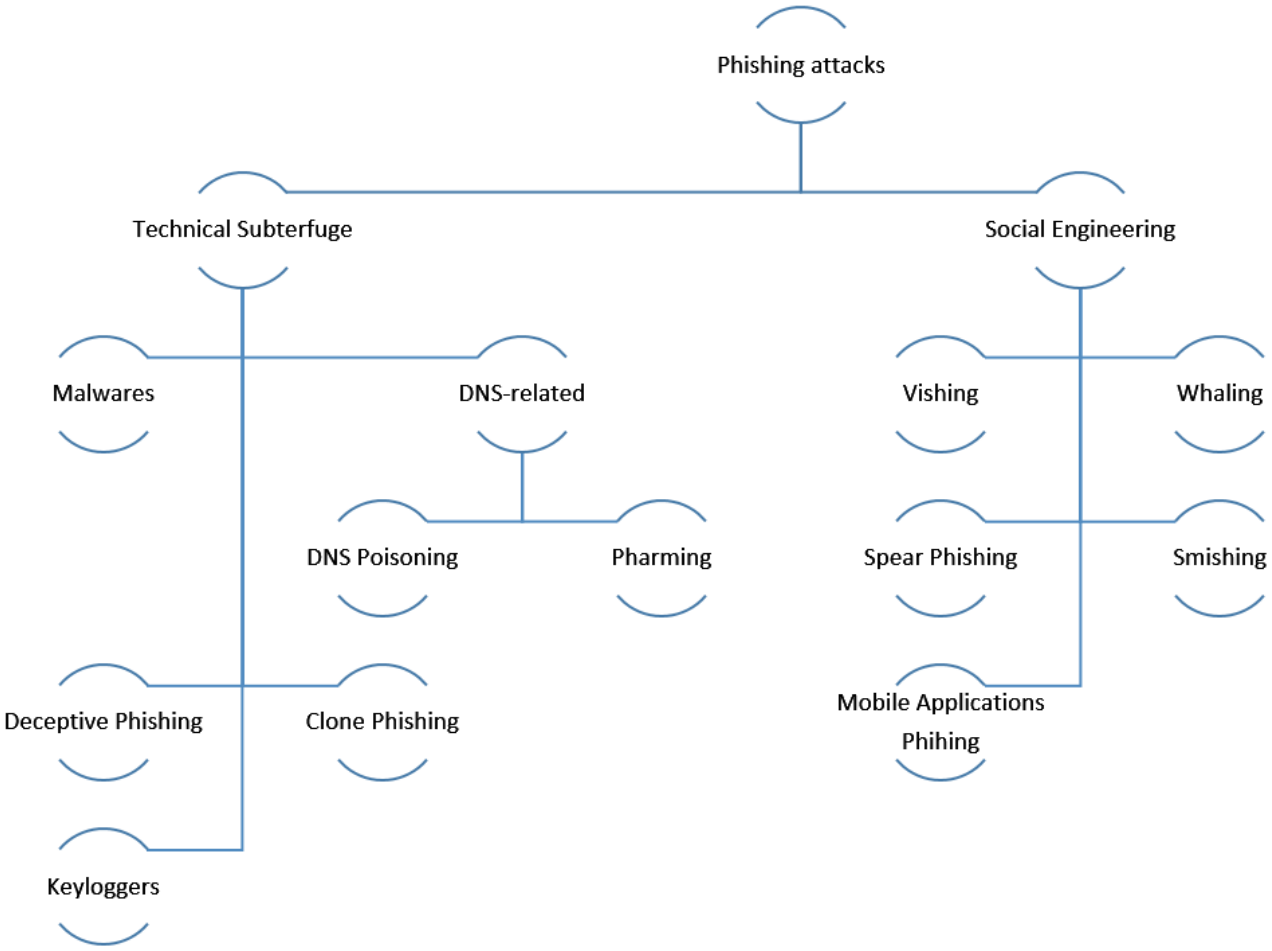

Рисунок 1.8 - Класифікація фішингових атак

Спрямований фішинг (Spear phishing) – це цілеспрямована фішингова атака. Атаки спрямованого фішингу використовують електронну пошту, щоб досягнути мети; персоналізовані електронні листи надсилають конкретній особі. Перед відправленням такого електронного листа зловмисник вивчає інтереси потенційної жертви. Найчастіше жертвами спрямованого фішингу є високі посадові особи, які мають доступ до більшої конфіденційної інформації, аніж пересічний робітник [14].

Фішинг з елементами зловмисного програмного забезпечення (Malware phishing та drive-by phishing [15]) – атака, що ставить за мету обманом змусити жертв завантажити зловмисне ПЗ на їх комп'ютери. Атаки що відносяться до цього типу заманюють користувачів клікнути на посилання чи завантажити додаток який в свою чергу завантажить зловмисний код уже на самому ПК жертви без її відома. В подальшому зловмисний код може бути використаний як для приєднання до зараженого ПК до мережі ботнет, ескалації повноважень в системі до рівня супер

користувача або банально для викрадення персональних даних, збережених без додаткового захисту. Як зазначалося вище підтип “drive-by phishing” також відноситься до фішингу з елементами зловмисного ПЗ. Сценарій виглядає схожим чином: при відвідуванні скомпроментованого веб сайту ПК жертви заражається зловмисним кодом через автоматично виконуваний JavaScript код. Зловмисник може використати відомі або невідомі широкому загалу вразливості веб браузера або операційної системи для “тихого” а тому і непомітного для власника ПК завантаження вірусу або будь-якого іншого шкідницького коду.

Вішинг (англ. Vishing від поєднання Voice та Fishing) або телефонний голосовий фішинг – це такий тип атаки, коли злочинець дзвонить за номером телефону й за допомогою створення відчуття невідкладності ситуації змушує людину вчиняти дії проти своїх інтересів. Такі дзвінки зазвичай відбуваються в стресовий час, наприклад, посеред ночі, коли людина раптово прокидається і має зрозуміти, що відбувається. Ці атаки є популярними, оскільки телефонні дзвінки знеособлені, оскільки жертва не бачить співрозмовника. А це надає неабиякі переваги зловмиснику, оскільки завжди можна покласти слухавку і, наприклад, викинути сім-карту. У більшості країн вже існує обов’язкова реєстрація номеру з прив’язкою до паспортних даних особи, але навіть за такого сценарію, користувач має на це певний проміжок часу до блокування сім-карти, а це дає можливість бути знеособленим деякий період [14].

Кілоґґер (Keylogger) – шкідливе ПЗ, що використовується хакерами з метою крадіжки чутливих даних користувачів з метою подальшої компрометації їх акаунтів, шляхом відслідковування натиснутих клавіш та передачі їх зловмисникам. Таке ПЗ розроблено таким чином, аби документувати та пересилати зловмисникам дані про натискання клавіш жертвою на її ПК, що можуть включати аутентифікаційну інформацію пізніше використану з метою крадіжки ідентичності, фінансового шахрайства або несанкціонованого доступу до акаунтів та відповідно будь-яких інформаційних активів жертви. Програмне забезпечення такого типу найчастіше розповсюджується під виглядом імейл додатків, що на перший погляд не є виконуваними файлами, або через скрипти на фішингових сайтах.

СМС фішинг (Smishing) – це вид фішингу, який використовує текстові повідомлення на мобільних телефонах. Злочинці видають себе за офіційне джерело, щоб завоювати довіру жертви. Наприклад, під час СМС фішингу зловмисник може надіслати жертві посилання на веб сайт. Коли жертва відвідає цей веб сайт, на мобільний телефон буде встановлене зловмисне ПЗ. Також популярний сценарій смс-фішингу, це смс повідомлення з довільного або прихованого номеру нібито від банку з текстом «Vasha karta bude zablokovana bankom. Terminovo proidit avtorizaciu, abo zatelefonuite…» [14].

Отруєння кешу DNS або Підробка DNS (DNS Poisoning або DNS spoofing) – атака, що використовує вразливості DNS для відхилення або перенаправлення користувацьких запитів з легітимних веб сайтів на підроблені. Зловмисник перехоплює та проводить підміну DNS записів на локальному хості жертви, на мережевому обладнанні (наприклад на роутері), яке використовується жертвою для виходу в мережу Інтернет або на шляху слідування такого запиту. Атака (перенаправлення на підроблений сайт) може відбутися непомітно для жертви, адже браузери неабияк спрощують для користувачів процес пошуку необхідної адреси яка відповідає введеній адресі веб сайту. Опинившись на підробленому сайті та не перевіривши адресну строку, жертва, яка не підозрює що її запит був перенаправлений без зайвих коливань вводить необхідну для зловмисника чутливі дані, що дає можливість подальшого проведення широкого ряду інших атак та/або компроментування акаунту.

Вейлінг (Whaling або “полювання на корпоративних китів”) – це фішингова атака, що спрямована на осіб, які мають високий рівень доступу до інформації у межах організації, наприклад, її вище керівництво. Також цілями можуть бути політики або знаменитості. Термін вейлінг походить від розміру атак, а китів вибирають відповідно до їхніх повноважень у компанії. Через чітко спрямований характер вейлінг-атаки часто важче виявити, ніж стандартні фішинг-атаки. На підприємстві адміністратори безпеки можуть допомогти знизити ефективність вейлінг-атак, залучаючи співробітників корпоративного управління пройти навчання з інформаційної безпеки [14].

Фармінг (Pharming) – тип фішингових атак в рамках яких зловмисник компроментує браузер жертви та перенаправляє її запити на веб сайти клони з метою крадіжки чутливої інформації навіть коли вона вводить в пошукову стрічку легітимну адресу. Реалізація може бути виконана завдяки зловмисному коду на стороні клієнта такому як певне розширення, що працює в фоновому режимі.

Сайти-клони (Clone Phishing) – тип фішингових атак, що проводиться шляхом створення повного клону відомого веб сайту (часто включаючи розширений функціонал а не тільки форму входу), що є ідентичним до легітимного. Посилання, форми та ішні інтерактивні елементи таких веб сайтів можуть мати в собі корисне навантаження у вигляді виконуваного зловмисного коду. Крім того заповнення сфабрикованих форм жертвою являє собою пряму передачу чутливих даних зловмиснику. Досить вірогідним є сценарій подальшого зображення “помилки на стороні сервера” та прохання “перелогінитися” з фоновим перенаправленням на легітимний ресурс аби жертва все таки отримала доступ до свого аккаунту і навіть не запідозрила, що компрометація облікового запису відбулася.

Фішинг мобільних додатків (Mobile applications phishing) – тип фішингових атак, що таргетується на користувачів мобільних девайсів та полягає в тому, що зловмисники обманом змушують жертв завантажити шкідливі мобільні додатки. Часто зловмисники маскують зловмисне ПЗ під застосунки, покликані допомогти користувачам виконувати певні корисні дії, після встановлення на операційній системі Android, додаток традиційно просить дозволити йому доступ до переліку даних та функцій які виходять за рамки послуг наданих цим ПЗ, але жертва немає іншого виходу як надати ці дозволи аби користуватися повним функціоналом застосунку. Отримавши відповідні дозволи застосунок проаналізує та відправить чутливу інформацію зловмиснику для подальшої компрометації облікових записів.

В уже згаданому звіті компанії Positive Technologies за 2 квартал 2022 року [10], більше половини випадків зараження корпоративного комп’ютера шкідливим програмним забезпеченням відбулося саме через електронну пошту, при цьому звичайні користувачі інтернету в 43% випадках стали жертвами зловмисників

заразивши свій девайс відкривши фішинговий веб сайт, це проілюстровано на рис. 1.9.

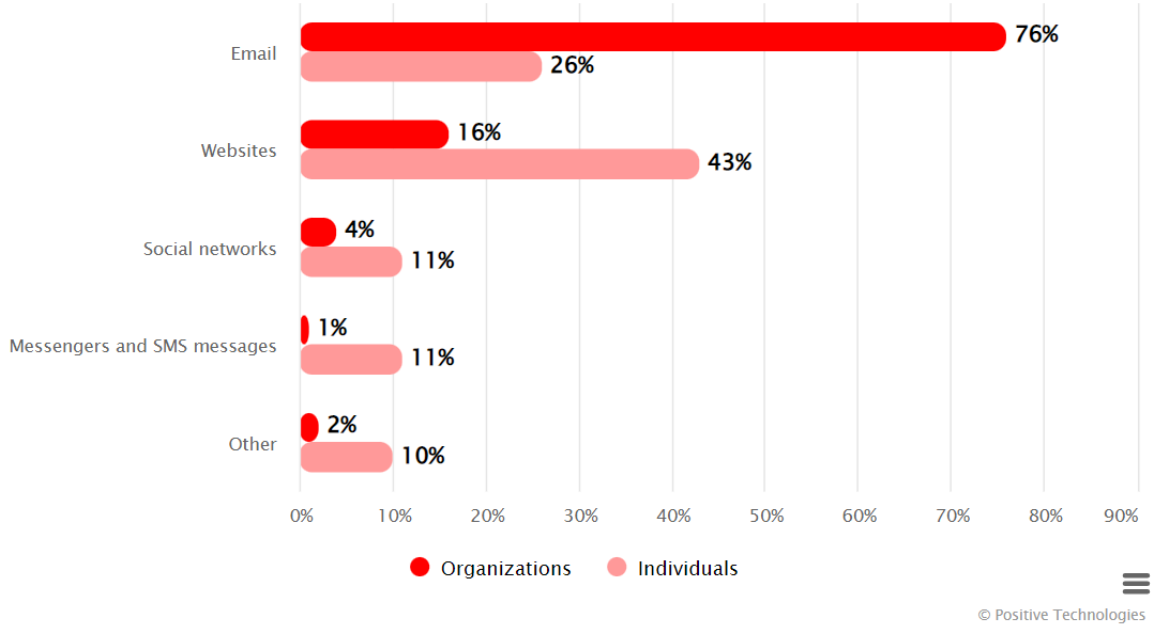

Рисунок 1.9 – Канали здійснення атак з використанням методів СІ

Спираючись на доведені статистичні дані, фокусом роботи було визначено фішингові веб сайти, адже це один з найчастіше використовуваних зловмисниками каналів здійснення атак з використанням методів СІ для зараження девайсів своїх жертв, а тому потребує вітчизняних розробок в сфері протидії такому типу загроз.

1.5 Фішинг у правовому полі України

Визначившись, що зловмисники вбачають за мету викрасти конфіденційні дані та/або іншу інформацію, якою володіють їх жертви, ми маємо спершу конкретно креслити ці поняття спираючись на наявне правове поле України. Закон України “Про інформацію” [16] окреслює конфіденційну інформацію як інформацію про фізичну особу, а також інформацію, доступ до якої обмежено фізичною або юридичною особою. Конфіденційна інформація може поширюватися за бажанням (згодою) відповідної особи у визначеному нею порядку відповідно до передбачених нею умов, а також в інших випадках, визначених законом. У сучасному світі найуразливішою

ланкою серед усіх кібератак залишається людина. Персональні дані можуть бути як конфіденційною інформацією, так і відкритою. Пункт 2 статті 11 Закону України "Про інформацію" [16] встановлює, що конфіденційними є зокрема дані про національність, освіту, сімейний стан, релігійні переконання, стан здоров'я, місце проживання і/або реєстрації, дата і місце народження. Закон України "Про захист персональних даних" [17] визначає персональні дані як відомості чи сукупність відомостей про фізичну особу, яка ідентифікована або може бути конкретно ідентифікована. Це найуживаніше визначення даного поняття - і воно є найбільш узагальненим. Перелік відомостей, які належать до персональних даних, змінюється від однієї держави до іншої. До прикладу, інформація про IP-адресу в країнах Європейського Союзу належить до персональних даних, а у США ні, проте відноситься до інформації, що пов'язана з персональними даними [18].

Фішинг відноситься до кіберзлочинів. Суб'єктами кіберзлочинів можуть бути фізичні осудні особи, які до моменту їх вчинення досягли шістнадцятирічного віку. В той же час, спеціальний суб'єкт має місце у трьох випадках:

- несанкціоновані дії з інформацією, яка оброблюється в електронно-обчислювальних машинах (комп'ютерах), автоматизованих системах, комп'ютерних мережах або зберігається на носіях такої інформації, вчинені особою, яка має право доступу до неї, де суб'єктом виступає особа, що має право доступу до інформації [19];
- порушення правил експлуатації електронно-обчислювальних машин (комп'ютерів), автоматизованих систем, комп'ютерних мереж чи мереж електрозв'язку або порядку чи правил захисту інформації, яка в них обробляється [20], де суб'єктом виступає особа, яка відповідає за експлуатацію електронно-обчислювальних машин, автоматизованих систем, комп'ютерних мереж чи мереж електрозв'язку;
- несанкціоноване втручання в роботу інформаційних (автоматизованих), електронних комунікаційних, інформаційно-комунікаційних систем, електронних комунікаційних мереж [21], вчинені особою або групою осіб, що не мають доступу до інформаційних (автоматизованих), електронних комунікаційних, інформаційно-комунікаційних систем, електронних комунікаційних мереж.

Суб’єктивна сторона кіберзлочинів зазвичай характеризується прямим умислом і корисливим мотивом хоча діяння, передбачене статтями 362 та 363 Кримінального кодексу України, може вчинятися як умисно, так і через необережність [19,20]. Відповідно до статті 361 Кримінального кодексу України, несанкціоноване втручання в роботу інформаційних (автоматизованих), електронних комунікаційних, інформаційно-комунікаційних систем, електронних комунікаційних мереж трактується виключно як умисне злодіяння [21].

Об’єктом злочину є суспільні відносини, на які посягають злочини, а предметом злочину слід вважати будь-які речі матеріального світу, з певними властивостями яких кримінальний закон пов’язує наявність у діях особи ознак конкретного складу злочину.

Кримінальна відповідальність:

- штраф;
- обмеження волі;
- позбавлення волі з позбавленням права обіймати певні посади чи займатися певною діяльністю.

Кожен з вищезазначених видів покарань деталізовано у відповідних статтях. Відтак, можемо констатувати наявність фундаментальної правової бази для визначення фішингової діяльності як такої, за яке передбачене покарання.

Висновок до першого розділу

Дані отримані за останні п'ять років неабияк красномовно підкреслюють, що дослідження атак, що використовують методи СІ, а саме фішинг зараз дуже актуальні. Одна з ключових причин полягає в тому, що незалежно від того, скільки безпекових тренінгів чи технічних контролів не розгорнуто, цього завжди буде недостатньо і люди залишаються найслабшим ланкою безпеки підприємства будь-якої форми власності, державних інституцій, тощо.

“Безпека – це не продукт, а радше процес. Безпека – це не технологічна проблема, це проблема людей та управління ними”, – каже американський

криптограф та експерт в області комп’ютерної безпеки Брюс Шнайер. З ним важко не погодитись, адже не сьогодні не існує жодного технічного рішення, яке б гарантувало 100% захист мережі від широкого спектру кібератак, особливо від тих, які покладаються на методи СІ.

Беручи до уваги запропоновані статистичні дані, фокусом роботи було визначено фішингові веб сайти, адже це один з найчастіше використовуваних зловмисниками каналів здійснення атак, що покладаються на методи СІ, а тому потребує вітчизняних розробок в сфері протидії такому типу загроз.

РОЗДІЛ 2
ЗАХИСТ ВІД ФІШИНГОВИХ АТАК

2.1 Цикл атак соціальної інженерії та фішингу зокрема

Для ефективної протидії фішинговим атакам, слід визначити потенційні етапи для їх блокування. Автори [22] пропонують розглядати атаку соціальної інженерії як цикл. Він складається з чотирьох фаз: збору інформації, розвитку відносин, експлуатації та виконання.

1. Збір інформації (дослідження) може виконуватися за допомогою різних технік і охоплює інформацію про організацію – наприклад телефонні списки, організаційні діаграми, – або про передбачувані цілі атаки – тобто особиста інформація.

2. Для розвитку відносин необхідна довіра. Людина за своєю природою заслуговує довіри, і відносини можуть бути легко побудовані за допомогою належних знань, отриманих на попередньому етапі. Відносини можуть бути використані для виконання атаки або надання зловмиснику додаткової інформації про ціль.

3. Під час експлуатації довірений зловмисник може впливати на ціль, щоб змусити розкрити інформацію або виконати якусь дію в інтересах зловмисника.

4. Фаза виконання може бути інтерпретована як виконання останнього кроку в атаці, якщо отриманий доступ або інформація не є кінцевою метою, і зловмиснику все ще потрібно виконати заключний акт, використовуючи отриману довіру та інформацію, наприклад, увійти в інформаційну систему для викрадення, зміни або видалення файлів.

Як бачимо, ця робота пропонує розгляд фаз проведення атаки з використанням методів СІ, відмінних від фішингу, тому має бути вдосконалена та доповнена аби відповідати реаліям сучасних фішинг атак. В таблиці 2.1.1 пропонуються міркування щодо узагальнення фаз сучасних фішингових атак.

Таблиця 2.1.1

Фази сучасних фішингових атак

Фаза 1: Підготовка	**Фаза 2: Реалізація**	**Фаза 3: Використання**	**Фаза 4: Просування**
При реалізації зловмисником атаки цільового фішингу першим етапом є збір інформації про жертву. В фазі підготовки зловмисник проводить підбір каналу комунікації та передачі фальшивого корисного навантаження відповідно до профілю жертви. В рамках цієї ж фази відбувається підбір та налаштування або розробка технічного рішення, якщо таке буде використано.	В фазі реалізації фішингової атаки відбувається взаємодія між зловмисником, який вдає з себе довірену сторону та жертви напряму (переписка, листування, розмова) або опосередковано через використовуване зловмисне ПЗ. Протягом цієї ж фази відбувається експлуатація людських вразливостей та отримання та передача на підконтрольну зону бажаних у багатьох випадках аутентифікаційних даних. Неписаними кращими практиками фішингу в фазі реалізації атаки також є створення ілюзії, що витоку даних не відбулося, реалізація цього сценарію залежить як від типу комунікації так і від ряду інших факторів.	Отримавши необхідні аутентифікаційні дані зловмисник напряму або з використанням стороннього ПЗ виконує поставлені цілі такі як спустошення рахунків жертви, крадіжка ідентичності, шифрування даних і т.д. Не виключено також і те, що видимих для жертви не відбудеться, при цьому зловмисна програма продовжить виконуватися на зараженому девайсі в фоновому режимі якщо у зловмисника є більш довготривала стратегія.	Опціональна фаза просування є логічним продовженням діяльності певного типу зловмисників які мають на меті зараження більш ніж одного девайсу та використовують або розробляють зловмисне ПЗ таким чином, аби воно ширилось схожим до черв’яка чином.

Беручи за основу запропоновану узагальнену модель проведення фішинг атак та поставлену задачу кваліфікаційної роботи в наступних розділах будуть запропоновані підходи до детектування та блокування підозрілих веб сайтів в фазі реалізації попереджуючи розповсюдження зловмисного коду та подальше зараження

мережі та/або інших девайсів. Слід зазначити, що працівники відділу інформаційної безпеки організацій різних форм власності обов’язково мають впроваджувати необхідні моніторингові рішення, що відповідають специфіці їх роботи аби унеможливлювати розвиток атаки від 3-ї до 4-ї фази, але ця проблематика виходить за скоуп даної роботи.

2.2 Підходи до детектування фішингових веб сайтів

Визначивши фазу реалізації фішингової атаки, як таку, потягом якої слід проводити детектування та блокування зловмисних веб сайтів доцільно оглянути вітчизняні та закордонні підходи, що використовуються для вирішення цієї задачі. В статті [14] вітчизняний автор виокремлює два основних підходи до виявлення фішингу: навчання користувачів і за допомогою програмних засобів. Пропонується провести огляд запропонованих ним рішень.

Навчання користувачів: користувачів можна навчити краще розуміти природу фішингових атак, що в результаті допоможе коректно розрізняти фішингові і справжні повідомлення, проте автор знаходить певні суперечності і зазначає, що навчання користувачів часто розглядається як превентивний захід. В кінці-кінців робить висновок, що навчання має на меті розпізнавання користувачами фішингових атак, тому пропонує відносити його до підходів до виявлення фішингу. Не зважаючи на розповсюдженість та ефективність цього методу протидії фишинговым атакам, він, в свою чергу покладається на той же людський фактор, який і є першопричиною виникнення цієї загрози. Зважаючи на те, що метою кваліфікаційної роботи є розробка системи детектування та блокування фішингових веб сайтів в автоматичному режимі дана проблематика виходить за її скоуп.

За допомогою програмних засобів: цей підхід має на меті заповнити прогалину, яка виникає через помилку користувача або незнання, та розрізняти програмним способом фішингові та легітимні повідомлення. Цей недолік вимагає вирішення, оскільки навчання користувачів є дорожчим, ніж автоматична класифікація, та не

завжди є можливим, наприклад, коли бази користувачів є завеликими (PayPal, eBay, Amazon тощо).

Ефективність виявлення може бути покращена за рахунок навчання класифікатора (як людини, так і ПЗ). У випадку з навчанням користувачів якість виявлення може бути покращена за рахунок їхнього індивідуального досвіду або за допомогою зовнішніх навчальних програм. У випадку програмної класифікації ефективність може бути підвищена в процесі в процесі "навчання" класифікатора, побудованого на алгоритмах машинного навчання, або вдосконаленням правил виявлення в системі на основі правил.

Деталізуючи, Лаптєв [14] відносить до програмних засобів наступне: чорні списки, евристику, візуальну схожість та машинне навчання. В подальшому деталізує кожен з них наступним чином:

Чорні списки: це постійно оновлюванні списки, що містять раніше виявлені фішингові URL-адреси. Недоліком даної методології є затримка в оновленні списків. Для того, щоб щойно створений фішинговий сайт потрапив у список, необхідний час. Цієї затримки між відправленням даних і додаванням сайту до списку може бути достатньо для досягнення зловмисниками своїх цілей. Білі списки. Ці списки є чимось протилежним до чорних. Якщо є певна URL адреса, її порівнюють з легітимною адресою зі базою даних "білого списку". База даних "білого списку" здебільшого містить список популярних справжніх URL-адрес та їхні важливі дані. Як і у випадку з чорним списком, для завантаження нової відомої URL адреси може знадобитися певний час, через який зловмисник, безсумнівно, може досягти своїх цілей.

Евристичні методи виділяють певні характеристики веб сторінки для того, щоб визначити легітимність веб сайту, а не залежати від будь-яких попередньо скомпільованих списків. Це перевага евристичних методів, над "списками". Більшість цих характеристик витягуються з URL-адреси та дерево об'єктної моделі документа HTML (DOM) даної веб сторінки. Вилучені характеристики порівнюються з вже відомими, що були зібрані з фішингових та справжніх сторінок, щоб визначити їх легітимність. Деякі з цих підходів використовують евристики для обчислення оцінки підробки даної веб сторінки, щоб перевірити її справжність, як зазначено в

[24]. Сучасні веб браузери та поштові клієнти побудовані з механізмами захисту від фішингу, такими як евристичні тести з метою виявлення фішингових атак. Так само евристичні тести виявлення фішингу можуть бути включені в антивіруси.

Методи візуальної схожості. Це методи розрізнення фішингових сайтів та легітимних сайтів за зовнішнім виглядом сайтів. Зазвичай фішингові сайти є майже точними копіями справжніх, щоб у користувача не виникали сумніви щодо легітимності ресурсу. З метою не бути виявленими зловмисники, як правило, вставляють зображення, Flash, ActiveX і Java-аплет замість HTML-тексту. Методи виявлення на основі візуальної схожості можуть швидко розпізнавати перераховані об'єкти на веб сторінках фішингових сайтів. Методи, засновані на візуальній схожості, використовують підпис, щоб розрізняти фішингові сторінки. Щоб зробити підпис потрібно вибирати спільні компоненти з усього сайту, а не з окремої сторінки веб сайту. Таким чином, одного підпису достатньо, щоб ідентифікувати різні цільові веб сторінки окремого веб сайту або унікальні форми веб сайту. Даний метод використовує для порівняння дерево об'єктної моделі документа HTML (DOM), схожість каскадної таблиці стилів (CSS), візуальне сприйняття, візуальні особливості, піксельні та гібридні підходи.

Машинне навчання забезпечує спрощені та ефективні методи аналізу даних, останнім часом демонструючи багатообіцяючі результати у проблемах класифікації в реальному часі. Ключовою перевагою машинного навчання є можливість створювати гнучкі моделі для конкретних завдань, таких як виявлення фішингу. Оскільки фішинг є проблемою класифікації, моделі машинного навчання можна використовувати як потужний інструмент. Моделі машинного навчання можуть швидко адаптуватися до змін, щоб визначити моделі шахрайських операцій, які допомагають розробити систему ідентифікації на основі навчання.

Провівши огляд запропонованих Лаптєвим у [14] підходів для виявлення фішингу було визначено, що список використовуваних підходів може бути розширено, спираючись як на дослідження закордонних науковців [25-49, 51, 52] так і на принципи роботи лідерів світового IT ринку. Запропонована схема, що включає розширений перелік підходів до детектування фішингу наведена на рис. 2.1.

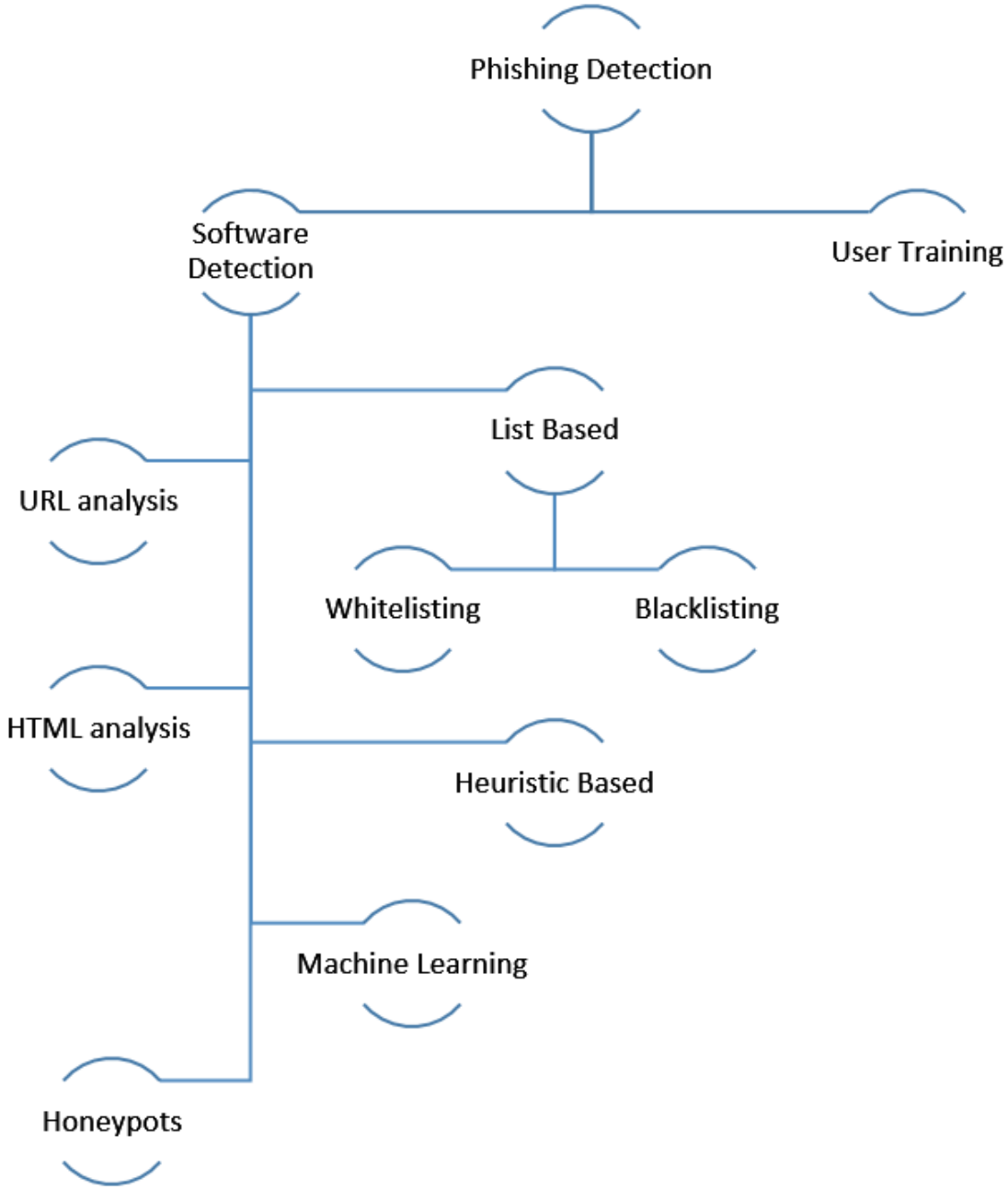

Рисунок 2.1 – Підходи до детектування фішингу

В дослідженні [29] автор пропонує використовувати адресу веб сайтів як один із типів вхідних даних для алгоритму класифікації веб сайтів на фішингові та легітимні. Його алгоритм використовує атрибути URL, такі як: назви директорій, довжина, доменне ім'я, кількість спеціальних символів, назва файлу і т.д. Цей метод має ряд переваг над популярними до цього традиційних методів вайтлістингу та блеклістингу, як зазначає [32] адже вони працюють доти, доки зібрана база даних відповідає реальній ситуації в кіберпросторі, шо неабияк змінилося за останні 20

років завдяки експоненціальному зростанню кількості сайтів та полегшенню процедури реалізації фішингових атак.

В дослідженнях [33, 34] пропонується підхід виявлення фішингових веб сайтів, що заснований на візуальній подібності і включає оцінку візуальної схожості підозрілого веб сайту та інших за допомогою схожості стилю та макету (DOM аналіз) і TrustPage, який використовує пошук Google і оцінку користувачів для визначення візуально подібної сторінки відповідно. На рис. 2.1 він представлений як HTML аналіз.

Відносно новим підходом до детектування фішингу є пастки більше відомі як “honeypots” [35]; вони покликані імітувати існуючі робочі системи та приваблювати о себе зловмисників аби збирати дані та вивчати патерни їх діяльності з метою розробки відповідної системи протидії та автоматичного блокування підозрілої активності. Як зазначає автор [36], пастки можуть бути використані для профілювання користувачів соціальних мереж та детектування фішингових дій але при цьому потребують значної тренувальної бази та постійного моніторингу спеціалістами з інформаційної безпеки через велику кількість хибних спрацювань. Відтак, підхід пасток не можуть бути ефективно використаний для задачі автоматичного детектування фішингових веб сайтів, а тому не буде внесений у скоуп цієї роботи.

2.2.1 Детектування фішингових веб сайтів за допомогою URL аналізу

Аналіз URL з метою детектування фішингових веб сайтів, як згадувалося в попередньому розділі може бути здійсненим різними способами, пропоную проаналізувати та виявити ті підходи які можуть бути ефективним в сучасних реаліях. В Таблиці 2.2.1 наведено порівняльний аналіз вайтлістингу, блеклістингу та проактивного аналізу адреси веб сайтів.

Таблиця 2.2.1

Підходи до детектування фішингових веб сайтів за допомогою URL аналізу

Підхід	**Оцінка ефективності**
Вайтлістинг	Підхід білих списків в сучасному світі виявляється досить обмежуючим в зв'язку з високою варіативністю задач та непередбачуваністю способів їх вирішення працівниками. Відтак встановлення обмежень як для працівників державного сектору, так і для приватних компаній матиме ефект ускладнення ряду як технічних так і організаційних процесів, а відтак суттєвого зниження ефективності роботи.
Блеклістинг	Підхід використання чорних списків справді вважався досить дієвим способом 10 років тому, коли кількість виявлених фішингових веб сайтів була в 10 разів меншою від останніх даних за 2022 рік [37]. Відтоді база фішингових доменів збільшилася удесяторо, і вірогідніше продовжить експоненційне зростання. Недоліком підходу з використання чорних списків є те, що більшість фішингових сайтів не орієнтовані на постійне використання і часто існують менше 20 годин [38], або ж постійно змінюють URL-адреси, саме тому підхід із внесенням підозрілої URL-адреси у чорний список не здатний розпізнати більшу частину сучасних випадків фішингових атак.
Проактивний аналіз URL	Знаючи, що фішингові URL-адреси часто містять підозрілі ключові слова або символи, за якими їх можна ідентифікувати, наприклад, багато фішингових URL-адрес містять неправильне написання офіційних доменних імен або використовують варіанти поширених ключових слів, таких як "login" або "banking". Для виявлення паттернів такого роду можна використовувати інструменти аналізу URL-адрес. Такий підхід описаний в роботі [39], де авторами запропонувано рішення для виявлення фішингу, засноване на використанні шести функцій URL-адреси, використанні SVM (Support Vector Machines) та індексу схожості для виконання процедури виявлення. Результат показав, що метод виявий фішингові URL-адреси з точністю 95,8% для 2 000 записів.

Спираючись на проведений аналіз робимо висновок, що підхід чорних списків може використовуватися у випадку наявності віддаленої та постійно оновлюваної

бази даних фішингових веб сайтів аби не перенавантажувати таким чином хости надлишковими копіями. Таким чином постановка задачі при використанні цього методу полягає в створенні та автоматизованій підтримці актуальності такої БД в режимі 24/7. Створення такого рішення потребує досить значних обчислювальних потужностей а також інтеграції з численними безпековими інтернет сервісами, тому більш раціональним варіантом вирішення поставленої задачі є пошук вендора, який вже проводить таку діяльність і залучення існуючої БД. Навіть більш прогресивною ідеєю є використання моделі SaaS або API, аби перевірка посилання відбувалася на стороні такого сервіс провайдера, що досить суттєво знизить використання ресурсів мережі, таке рішення з високою вірогідністю буде краще оптимізованою та якісно підтримувано на стороні вендора.

В реаліях комерційних підприємств або державних установ відповідні провайдери мають пройти сертифікацію Державної служби спеціального зв'язку та захисту інформації України або іншого сертифікатора, визнаного в Україні, що дозволить їм проводити таку діяльність. В дослідницьких цілях розробки гібридного методу виявлення фішингових елементів у веб сайтах пропонується використати публічний Safe Browsing APIs (v4) від Google, що був розроблений з метою виявлення небезпечних веб ресурсів [40]. Імплементація цього програмного інтерфейсу як одного з компонентів системи детектування фішингових веб сайтів буде розписана детальніше у розділі 3.1.

2.2.2 Детектування фішингових веб сайтів за допомогою аналізу їх змісту

При розгляді підходу детектування фішингових елементів, що базується на аналізі змісту веб сайту, доцільним є ознайомитися з вже існуючими дослідницькими матеріалами за цією тематикою.

Так у [41] автори пропонують підхід, в якому фішингові атаки напряму виявляються шляхом дослідження вмісту сайту. Ознаки, що використовуються в цьому підході можуть, включати в себе: поля для вводу паролю, орфографічні помилки, джерела зображень, посилання, вбудовані посилання тощо.

В дослідженні [42] крім вищезгаданого використовуються також лексичні ознаки, рейтинг сторінки у пошуковій системі, інформація сервісу WHOIS та контент сторінки, який включає в себе HTML, JavaScript, iframe, зображення та джерело їх походження серед інших.

Деякі дослідники в своїх працях вивчали підходи засновані на основі нечіткої логіки та використання серії хешів веб сайтів для ідентифікації фішингових сайтів [43]. Тим не менш, проведені експерименти, що стосуються підходу, заснованому на нечіткому хешуванні, показали, що він хоча і може ідентифікувати наявні атаки, але цю ідентифікацію можна легко уникнути, реструктуризувавши елементи HTML без зміни зовнішнього вигляду сайту.

Інструмент GoldPhish описаний в [44] реалізує підхід нечіткого хешування, а як пошукову систему використовує Google. Було помічено, що шахрайські веб сторінки існують лише протягом невеликого періоду часу, це в результаті дає таким сайтам більш низький рейтинг під час інтернет-запитів, що робить їх основою для антифішингового підходу на основі вмісту. Алгоритм роботи GoldPhish був описаний в [45] трьома кроками:

1. створення знімку (підпису) поточного веб сайту користувача у його веб браузері;
2. перетворення отриманого зображення в текстовий формат, зрозумілий для комп’ютера, використовуючи оптичне розпізнавання символів;
3. введення тексту, отриманого на другому етапі, у відому пошукову систему для отримання і аналізу результатів та оцінки рейтингу сторінки.

Цей інструмент практично не призводить до отримання помилкових спрацьовувань і надає захист від фішингової атаки нульового дня, що є його основною перевагою. Недоліком інструменту GoldPhish є час, необхідний для відтворення веб сторінки. Він є також сприйнятливим до алгоритму Google PageRank і атак пошукових служб Google, зазначається в [44].

Система детектування фішингових веб сайтів, запропонована в [46] побудована на 5 модулях, що виконують наступні функції:

1. білі списки (за URL);

а) пошук сторінки для входу (аутентифікації);

2. перевірка кількості посилань в HTML (відсутність тегу <a> – підозра фішингу);

3. перевірка адрес посилань в <footer> (атрибут “href” тегу <a> пустий або “#[будь що]” – підозра фішингу);

4. інформація про авторські права та контент тегу <title> (перевірка значень в білих списках);

5. “самоідентичність” сайту (якщо більша кількість посилань спрямовані на ресурси неафілійовані з доменом – підозра фішингу).

Існує також тип підходів виявлення фішингових веб сайтів, що заснований на візуальній подібності і включає оцінку візуальної схожості підозрілого веб сайту та інших за допомогою схожості стилю та макету (DOM) [47] і TrustPage [48], які використовують пошук Google і оцінку користувачів для визначення візуально подібної сторінки. В першому випадку візуальні функції, такі як рівень блоку (текст і зображення), схожість макету і CSS порівнюються з відповідними ознаками легітимного веб сайту. Кожній функції призначаються вага відповідно до пріоритету, який грає роль в розробці легітимного веб сайту. Підозрілий веб сайт класифікується як фішинговий, якщо його візуальна подібність перевищує кастомізоване порогове значення. Недоліком такої техніки є високий час відповіді, адже ця схема використовує чималу базу даних достовірних зображень, в той час як візуальне порівняння підозрілого веб сайту з елементами бази даних зображень є досить довготривалим процесом, що досліджено в [49].

Проаналізувавши методи запропоновані в дослідженнях [41-49], пропонується покращити метод виявлення фішингових веб сайтів шляхом проведення індивідуального аналізу елементів DOM (HTML), відмовившись від ідеї створення знімку або підпису сайту, що має призвести до зниження часу аналізу за рахунок підвищення швидкості відтворення веб сайту. Імплементація цього інтерфейсу як одного з компонентів системи детектування фішингових веб сайтів буде розписана детальніше у розділі 3.2.

Використовуючи детектування фішингових веб сайтів лише за допомогою аналізу їх вмісту не є достатньо ефективним в сучасних реаліях і потребує комбінування з іншими методами [50]. Кращих результатів досягають евристичні підходи, що комбінують аналіз адреси, контенту, інформації про домен та інших аномалій веб сайтів, вони будуть детальніше описані в розділі 2.2.3.

2.2.3 Евристичні підходи до детектування фішингових веб сайтів

Евристичні методи зазвичай використовують перевірку різних ознак, щоб виявити можливі фішингові веб сайти, такі як URL, інформаційне наповнення веб сторінки, мова, використання JavaScript функцій, заголовків, DNS записів та інших параметрів. В більшості випадків задача полягає в виявленні так званих сигнатур або ж паттернів, що будуть універсальними “зліпками” фішингових веб сайтів і в подальшому будуть порівнюватися з підозрілими сайтами на предмет схожості. Використання машинного навчання та інших аналітичних технік дозволяє де в чому автоматизувати процес виявлення фішингових веб сайтів та значно покращити ефективність виявлення [15, 25, 27, 42, 45, 49, 54-57].

При розгляді підходу детектування фішингових веб сайтів, що базується на евристичних перевірках, доцільно ознайомитися з вже існуючими роботами за цією тематикою.

В роботі [41] описані різного роду перевірки, які можливо провести на стороні клієнта для виявлення веб ресурсів, що мають на меті викрадення ідентичності, пропоную розглянути ті, що будуть корисними для вирішення задачі кваліфікаційної роботи:

1. Перевірка паролю: у випадку коли веб сайти запитують користувачів вводити паролі чи PIN-коди (або будь які інші чутливі дані) відбувається перевірка протоколу передачі http(s) і у випадку використання https додаткова перевірка валідності виданого сертифікату.

2. Перевірка усіх параметрів POST запиту: підроблені веб сторінки можуть змінити використання тегів та текстового наповнення на зображення, що не буде

вявлено звичайном парсером. Автори пропонують хешувати всі вихідні дані та перевіряти з базою даних паролів та іншою чутливою інформацією.

В докторській дисертації "An Intelligent Phishing Detection and Protection Scheme using a fusion of Images, Frames and Text" [51] автор проводить надзвичайно деталізований огляд індикаторів компрометації з відсотковим значеннями того, як вино впливають на визначення фішингу. В рамках підходу евристичного виявлення фішингових веб сайтів пропонується використати наступні перевірки:

1. Час існування домену: використовуючи сервісу WHOIS можливо визначити дату реєстрації а відтак вік домену. Автор пропонує вважати, що домени зареєстровані більше ніж 6 місяців тому майже у всіх випадках буде легітимним, відповідно менше 6 місяців – фішинг.

2. Записи DNS: використовуючи сервіс WHOIS слід звернути увагу на DNS записи, їх відсутність або прихованість власником домену підвищує вірогідність того, що досліджуваний домен використовується для фішингових цілей.

3. Вичерпання терміну реєстрації (власності) домену: роблячи припущення, що легітимні домени зазвичай викуплені на багато років вперед, автор стверджує, що домен до кінця реєстрації якого залишилося менше років вірогідно є фишинговим.

4. Використання <iframe> перенаправлення: ця функція використовується для перевірки тегу HTML, який використовується для відображення вбудованих веб сторінок на поточному веб сайті. Зловмисник може скористається цією функцією, створюючи тим самим "невидимий тег без рамки".

5. Вимкнена функція використання правої кнопки миші: ця функція використовується для перевірки, чи вимкнено функцію клацання правою кнопкою миші за допомогою JavaScript, щоб користувачі не могли зберегти або переглянути вихідний код веб сторінки. Іноді функція правої кнопки миші також вимкнена на шахрайському ресурсі, який відкривається у вікні браузера меню.

6. Переадресація веб сайту: ця перевірка використовується для визначення кількості перенаправлень, використовуваних веб сайтом. Як правило, легітимний сайт робить це один раз, тоді як фішинговий сайт робить це більше чотирьох разів, зазначає автор.

7. Джерело походження піктограми (фавікону): перевірка джерела завантаження "shortcut icon", якщо вона завантажується з домену, відмінного від того, що показано в адресному рядку – сайт ймовірно скопійований і є фішингом.

8. Використання функції “onMouseOver”: ця функція використовується, щоб перевірити, чи використовується JavaScript для показу користувачам підробленої URL-адреси в пошуковій стрічці браузера. Ця функція використовується для приховування реальної URL-адреси на якій перебуває користувач змінюючи її на бажану. Таким чином детектування використання цієї функції практично гарантує детектування фішингу.

В роботі [52] автор дослідив фішингові елементи в веб сайтах і запропонував ряд важливих перевірок, частина з яких буде використана в даній роботі:

1. Перенаправлення в URL за допомогою “//”.
2. Використання нестандартного порту (відмінного від 80 та 443).
3. Використання фрази “https” в доменній частині URL.
4. Значення SFH (Server Form Handler) "about: blank" або пусте – фішинг, значення що включає інший домен – підозра на фішинг.
5. URL-запити виконувані веб сайтом: якщо більша частина ресурсів завантажується з сторонніх доменів – підозра на фішинг.

З метою систематизації евристичних підходів для виявлення фішингових веб сайтів запропонована зведена Таблиця 2.2.3 перевірок елементів, характеристик та атрибутів веб сайту, що включає інформацію як про його адресу, внутрішнє наповнення, використання функцій і т.д. В таблицю також включені результати попереднього дослідження за цією тематикою з роботи [35].

Таким чином маємо чималу кількість ідентифікаторів компрометації для детектування фішингових елементів у веб сайтах. Багато в чому дослідження [35, 51, 52] пропонують схожі коефіцієнти для класифікування певних атрибутів, значень як підозрілих, проте все ж вони різняться.

Таблиця 2.2.3

Евристичні підходи для детектування фішингових веб сайтів

Адресний рядок	**Інформаційне наповнення та JavaScript код**	**Домен**	**Нетипові особливості**
Використання IP адреси	Кількість перенаправлень	Веб трафік	Використання нестандартного порту
Використання сервісів скорочення URL	Посилання на ресурси третіх сторін Google Analytics, Facebook, Cloudflare, etc.	Кількість посилань на ресурс з сторонніх доменів	Переважаюча кількість сторонніх доменів в URL-запитах для завантаження ресурсів
Додавання до домену суфіксу або префіксу з “-”	Використання перенаправлення з <iframe>	Позиція PageRank в популярних браузерах	Значення Server Form Handler пусте або "about: blank"
Використання https	Наявність сторінки з формою для авторизації та/або полів для введення даних	Google Index	
Валідність сертифікату та його емітент	Налаштування рядка стану	DNS записи (whois)	
Довгий URL	Відключення реагування на правий клік миші	Час існування домену	
Перенаправлення в URL за допомогою “//”	Кількість тегів <a> в HTML	Термін реєстрації (власності) домену	
Використання фрази “https” в доменній частині адреси	Значення атрибуту “href” тегів <a> в <footer>	Кореляція між інформацією про власника домену та URL	

Продовження табл. 2.2.3

Наявність символу “@’ в URL	Співпадіння контент тегу <title> та інформації про авторські права з назвою домену		
	Посилання в <Meta>, <Script> і <Link> тегах на сторонні ресурси		
	Кількість посилань спрямовані на ресурси домени третіх сторін		
	Використання JavaScript функції onMouseOver для маніпуляції адреси в пошуковій стрічці браузера		
	Джерело походження піктограми		
	Параметри POST запиту		

Сучасні дослідження за цією тематикою все частіше направлені на створення алгоритмів машинного навчання, нейронних мереж та моделей, здатних аналізувати вплив вищезгаданих та інших ідентифікаторів компрометації на прийняття рішення про валідність веб сайтів. Доцільність використання методів машинного навчання для досягнення мети кваліфікаційної роботи буде досліджено в розділі 2.2.4.

2.2.4 Використання машинного навчання для детектування фішингових веб сайтів

Як зазначалося в розділі 2.2.3, оскільки евристичні методи зосереджені на паттернах або сигнатурах, що складаються з подібних шаблонів, вони більш схильні ідентифікувати фішингові елементи, які ще не були додані в чорні списки великих вендорів з більшою ймовірністю, що робить їх більш гнучкими, але в той же час створює ризик неправильної ідентифікації легітимних веб сайтів і певного відсотку

хибних спрацювань, що порушує нормальний робочий процес як АС так і створює додаткове навантаження на усю систему, що в свою чергу призводить до її уповільнення і даремної витрати як обчислювальних потужностей, так і часу.

Відтак на шляху еволюції антифішингових засобів з розвитком нейронних мереж з'явилися методи детектування фішингу, що покладаються на машинне навчання. Машинне навчання застосовується в багатьох сферах, включаючи кібербезпеку для оцінки методів, які застосовуються для виявлення вторгнень, зловмисного програмного забезпечення, спаму та фішингу [35]. Архітектура моделей глибокого навчання складаються з нелінійних операцій на кількох рівнях, таких як нейронні мережі з прихованими шарами, або складних реляційних методів у багаторазових підходах [54].

В праці [55] було розроблено два набори функцій для веб фішингової взаємодії та оригінального вмісту. Також була розроблена схема, засновану на мережі глибоких переконань (deep belief network або DBN). Тест, в рамках якого використовувалися реальних IP-потоки від інтернет сервіс провайдера, показав, що запропонована модель на основі DBN здатна досягти приблизно 90% ефективності виявлення фішингових веб сайтів.

Ефективність моделей машинного навчання для виявлення фішингу може залежати від різних факторів, таких як кількість та якість навчальних даних, специфічні функції, які використовуються для виявлення фішингових веб сайтів, і здатність виявляти фішингові атаки нульового дня. Пропоную ознайомитися з перевагами та проблемами при використанні підходів глибинного навчання в контексті детектування фішингових веб сайтів представленими в роботах [56, 57], що наведені в таблиці 2.2.4.

Таблиця 2.2.4

Використання глибинного навчання для детектування фішингових веб сайтів

Переваги	**Проблеми**
Доступний режим за якого кількість ітерацій та завдань спочатку невідома.	Незбалансовані дані: проблема, яка виникає під час навчання та здебільшого під час класифікації, якщо ознак одного класу більше, ніж інших.
Навчання може бути проведено з попередньо некласифікованими даними.	Недостатня кількість та/або якість даних для навчання: проблема, яка виникає, коли обмежена кількість даних доступна для методів перехресної перевірки, які в основному застосовуються шляхом поділу доступних даних на два набори, один для навчання, а інший для перевірки, щоб перевірити поведінку моделі.
Алгоритм здатний вивчати концепції низького, середнього та високого рівнів з незначним коригуванням оператора, що досить корисно з точки зору характеристики складних функцій, необхідних для завдань глибинного навчання.	Перелив даних: проблема виникає у великих даних, оскільки обсяг даних в декотрих випадках може зростати експоненціально, і прогнозується, що кількість інформації, яка містить великі дані, збільшуватиметься темпами швидшими та темпи її обробки.
Алгоритм може використовувати взаємодії, які існують у величезній кількості завдань. Ці взаємодії існують тому, що все, що пропонує завдання алгоритму — це різноманітний погляд на ту саму базову реальність.	Частковість даних: іноді набір даних використовується для вирішення певного завдання, але дані стають частковими, коли частина з них втрачається або тому, що деякі їх змінні чи функції не ідентифіковані.
Алгоритм може навчитися багатогранній, дуже варіативній функції з декількома відмінностями, набагато більшими, ніж кількість екземплярів навчання.	Високий рівень вимірювання: інформація в додатку реального світу часто переповнена визначенням конкретної точки зору проблеми, яку може обробити алгоритм.

Як бачимо, більша кількість зазначених в [57] проблем напряму пов’язані з дата сетами необхідними для тренування та тестування моделей. Для вирішення поставленої задачі необхідно мати величезний датасет (більше 20000 об’єктів) з наступними даними про кожен із веб сайтів:

1. URL адреса.
2. Контент: HTML, JavaScript, і т.д.
3. Повні записи whois на момент існування.
4. Історичні дані індексування даного сайту популярними пошуковими системами.
5. Історичні дані про трафік в рамках домену та/або субдомену.

Формування датасету є одним з ключових етапів при розробці алгоритму глибинного навчання для виявлення фішингових веб сайтів. Для створення надійного датасету потрібно зібрати велику кількість даних, що включає в себе як фішингові, так і легітимні веб сайти. Отримані дані потрібно перевірити на коректність та додати до датасету. Якщо з легітимними сайтами складнощів не виникає, то от з фішинговими це є досить важко адже як зазначалося в розділі 2.2.1 шахрайські веб сторінки в середньому існують протягом 20 годин до моменту їх детектування, аналізу великими гравцями, додавання в чорні списки та блокування на рівні інтернет сервіс провайдера.

Враховуючи велику кількість даних, що потрібно зібрати та перевірити на коректність, формування датасету може зайняти значний час і вимагає значних ресурсів, які виходять за рамки кваліфікаційної роботи. Крім того, навчання моделі на такому датасеті потребує значних обчислювальних ресурсів та часу, що є досить складним завданням для дослідників з обмеженими ресурсами. Тому, хоча формування датасету є важливим етапом при проведенні навчання моделі, воно не є головною метою даної роботи. Замість цього, подальше дослідження зосереджується на евристичних підходах виявлення фішингових веб сайтів, які можуть допомогти досягти більш високої точності виявлення фішингових атак нульового дня ефективніше за чорні списки без використання великого датасету.

Висновок до другого розділу

Фішинг є серйозною загрозою безпеки інфраструктури практично кожної організації – бізнесу будь-якого розміру, державних інституцій, банків, підприємств промисловості та багатьох інших суб'єктів господарської діяльності.

В розділі описаний цикл атак соціальної інженерії та запропонована узагальнена модель проведення фішинг атак. В результаті проведеного аналізу запропоновані підходи до детектування та блокування підозрілих веб сайтів в фазі реалізації фішингових атак попереджуючи розповсюдження зловмисного коду та подальше зараження мережі.

Визначивши фазу реалізації фішингової атаки, як таку, потягом якої слід проводити детектування та блокування зловмисних веб сайтів було проведено огляд вітчизняних та закордонних підходів, що використовуються для вирішення задачі детектування фішингових елементів у веб сайтах.

Проаналізувавши методи детектування фішингових елементів за допомогою URL аналізу, пропонується використати інтерфейс заснований на чорних списках Safe Browsing APIs (v4) від Google, що був розроблений з метою виявлення відомих небезпечних веб ресурсів.

Проаналізувавши методи детектування фішингових елементів за допомогою аналізу контенту веб сайтів, пропонується проводити індивідуальний аналіз елементів DOM (HTML), відмовившись від ідеї створення знімку або підпису сайту, що має призвести до зниження часу аналізу за рахунок підвищення швидкості відтворення веб сайту.

Проаналізувавши евристичні підходи до детектування фішингових веб сайтів було створено попередній перелік перевірок, що їх необхідно провести з метою виявлення підозрілих елементів у підроблених сайтах, які ще не потрапили до чорних списків популярних вендорів.

Сучасні дослідження в кібербезпеці все частіше направлені на створення алгоритмів машинного навчання, нейронних мереж та моделей, здатних аналізувати

вплив ідентифікаторів компрометації на прийняття рішень про блокування. Враховуючи кількість даних, що потрібно зібрати та валідувати, формування датасету для створення моделі виявлення фішингових сайтів потребує значних обчислювальних ресурсів та часу, що є досить складним завданням для такого типу дослідження з обмеженими ресурсами, тому виходить за скоуп кваліфікаційної роботи. Подальше дослідження зосереджується на евристичних підходах виявлення фішингових веб сайтів, які можуть допомогти досягти більш високої точності виявлення фішингу без використання великого датасету.

РОЗДІЛ 3
СИСТЕМА ДЕТЕКТУВАННЯ ФІШИНГОВИХ АТАК

3.1 Компонент 1: Перевірка URL

Як зазначалося в розділі 2.2.1 з метою оптимізації використання таких ресурсів як час, обчислювальні можливості та фінансові ресурси для виконання задачі проактивного аналізу адреси веб сайту було запропоновано використати Safe Browsing APIs (v4) від Google, що був розроблений з метою виявлення небезпечних веб ресурсів, включаючи фішинг. В таблицях 3.1.1 та 3.1.2 проаналізовані доступні прикладні програмні інтерфейси Google [40] з метою визначення рішення, що найкраще підійде для виконання поставленої задачі.

Таблиця 3.1.1

Прикладний програмний інтерфейс Google Safe Browsing Lookup API (v4)

Опис	Програмний інтерфейс Lookup API (v4) дозволяє клієнтським програмам надсилати URL-адреси на сервер Google Safe Browsing для перевірки їх статусу в постійно оновлюваній базі даних. Lookup API достатньо простий і легкий у використанні, оскільки він спрощує задачу створення та підтримки бази даних на стороні клієнта, надаючи можливість проводити перевірку на стороні надійного сервіс провайдера.
Переваги	Простота використання: надсилаючи HTTP-запит формату POST з метаданими, що серед іншого включають URL-адреси сайтів для перевірки, сервер у відповідь повідомляє про статус цих URL-адрес відповідно до інформації в своїй базі даних (безпечні чи небезпечні) клієнтському ПЗ. Швидкість: середня затримка відповіді на стороні сервера за звичайних умов - 0.002 секунди; також певний час займатиме парсинг даних, формування та відправка HTTP-запиту, отримання відповіді та зворотній парсинг на стороні клієнта. Таким чином, очікується, що процес аналізу посилання займатиме менше 1 секунди і буде непомітним для користувача клієнтського ПЗ.

Продовження табл. 3.1.1

	Економічна обґрунтованість: використання Lookup API безкоштовне, витрати на інфраструктуру та підтримку клієнтського ПЗ мінімальні.
Недоліки	Конфіденційність: URL-адреси не хешуються, та не шифруються, відтак сервер та будь-хто, хто аналізує трафік знає, які URL-адреси відправляються на перевірку. Час відповіді: кожен запит на пошук обробляється сервером. що може бути перевантажений іншими запитами, відтак середня швидкість відповіді буде подовжуватися адже вендор не дає гарантії на найдовший час відповіді на запит.

Таблиця 3.1.2

Прикладний програмний інтерфейс Google Safe Browsing Update API (v4)

Опис	Програмний інтерфейс Update API (v4) дозволяє клієнтським програмам завантажувати хешовані версії списків безпечного перегляду для локальної перевірки URL-адрес на стороні клієнта. Update API розроблено для клієнтів, яким потрібні відповіді на запити з високою частотою та малою затримкою. Кілька веб браузерів і програмних платформ використовують цей програмний інтерфейс для захисту великої кількості користувачів від різного типу загроз включаючи фішинг та зловмисне програмне забезпечення.
Переваги	Конфіденційність: обмін даними із сервером виконується нечасто (тільки після збігу локального хеш-префіксу) а використання хешованих URL-адрес, практично унеможливлює можливість для серверу та всіх “мережевих слухачів” дізнатися фактичні URL-адреси, які запитує клієнтське ПЗ. Час відповіді: так як підтримка локальної бази даних покладена на клієнтське ПЗ, а отже містить копії списків безпечного перегляду — користувачам клієнтського ПЗ не потрібно відправляти на сервер запит сервер кожного разу, коли вони здійснюють перевірку URL. Таким чином час перевірки цілком та повністю контролюється розробниками клієнтського ПЗ і фактично не залежить від вендора.
Недоліки	Технічна реалізація: необхідно налаштувати локальну базу даних, а потім завантажити та періодично оновлювати локальні копії списків безпечного перегляду (зберігаються як хеші SHA256 змінної довжини).

Продовження табл. 3.1.2

	Комплексні перевірки URL-адрес: для розробки системи необхідні знання процесів канонізації URL-адрес, створення виразів суфіксів/префіксів і обчислення SHA256 хешів (для порівняння з локальними копіями списків безпечного перегляду, а також зі списками безпечного перегляду, що зберігаються на сервері). Це необхідно адже сервіс не надає можливості завантаження самих URL-адрес в чистому вигляді. Додаткові витрати: ускладнена технічна реалізація вимагає наявності додаткової інфраструктури та впровадження процесів для автоматизації обновления контенту БД а також більшої кваліфікації як розробників так і спеціалістів з ІБ.

Висновок проведеного аналізу: при написанні кваліфікаційної роботи з метою оптимізації використання ряду ресурсів буде використано Lookup API (v4) з тілом запиту, наведеним у Додатку А, проте при розробці ліцензійних рішень для захисту корпоративних або державних мереж рекомендовано виділяти більші ресурси та використовувати Update API (v4). Імплементація обраного прикладного програмного інтерфейсу як одного з компонентів системи детектування фішингових веб сайтів описана в розділі 4.2.

3.2 Компонент 2: Аналіз контенту DOM

Проаналізувавши методи запропоновані в дослідженнях [41-50], пропонується покращити метод виявлення фішингових веб сайтів шляхом проведення індивідуального аналізу елементів DOM (HTML), таким чином відмовившись від ідеї створення знімку або підпису сайту, що має призвести до зниження часу аналізу за рахунок підвищення швидкості відтворення веб сайту.

Як відомо, в більшості випадків фішингові сайти використовуються для отримання аутентифікаційних даних користувачів. Механізм передачі цих даних при будь-якому сценарії починається з заповненні жертвою форми. Таким чином наявність в DOM сайту форми дає підстави для подальшої перевірки. В свою чергу

форма має ряд параметрів таких як "action", а також та внутрішніх елементів таких як <input> чи <label> з їхніми атрибутами, такими як "type", "novalidate" тощо. Для більш зручного представлення цієї інформації спершу пропоную проаналізувати спрощену об'єктну модель документа веб сайту, яка зображує важливі для дослідження елементи, це проілюстровано на рис. 3.1.

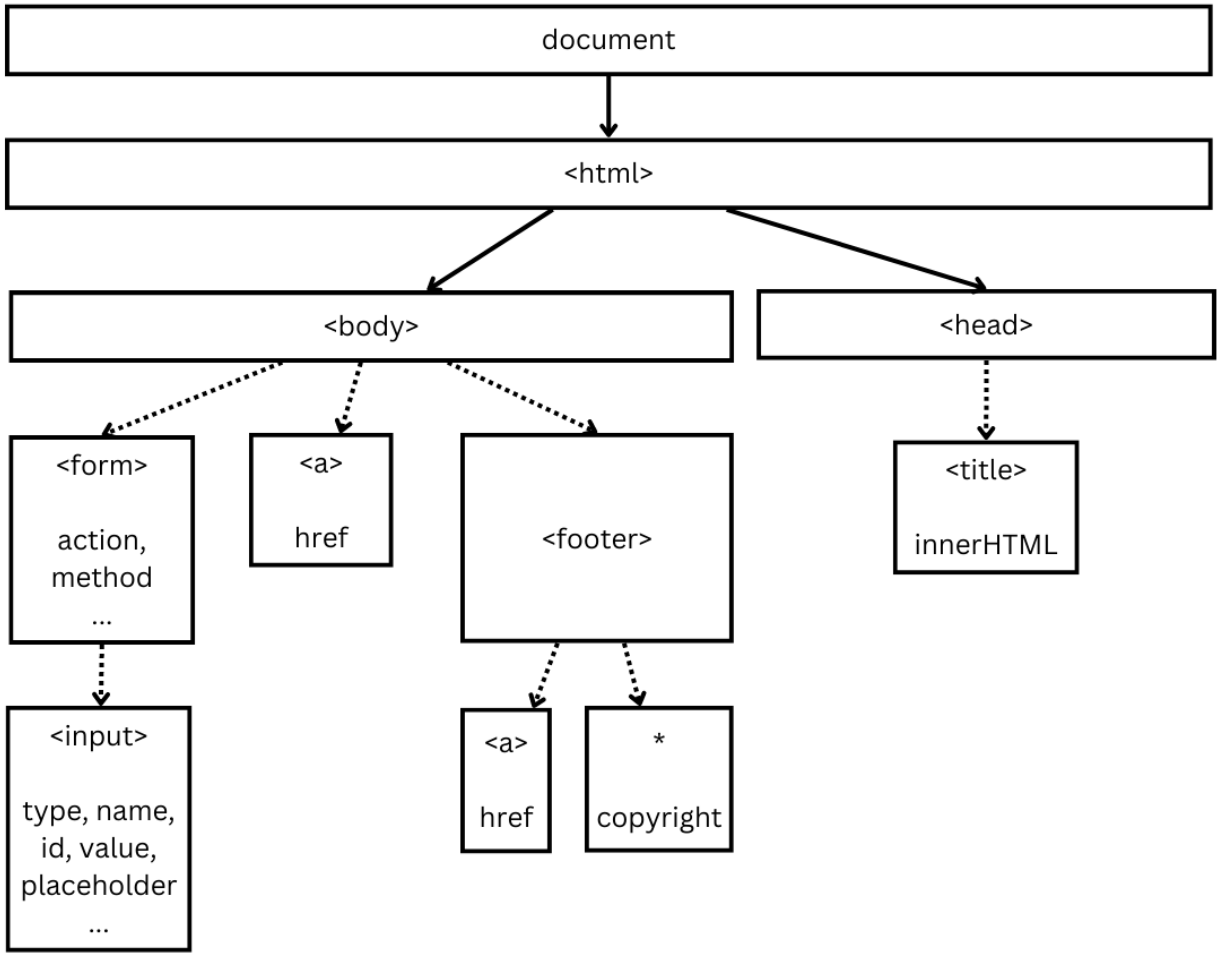

Рисунок 3.1 – Спрощена об'єктна модель документа веб сайту

Таким чином виникає необхідність створення веб кроллера, що буде зчитувати елементи веб сайту для подальшого аналізу їх атрибутів та значень і прийняття рішення щодо легітимності аналізованого ресурсу. Базова логіка роботи такого технічного рішення пропонується така, як це показано на рис. 3.2.

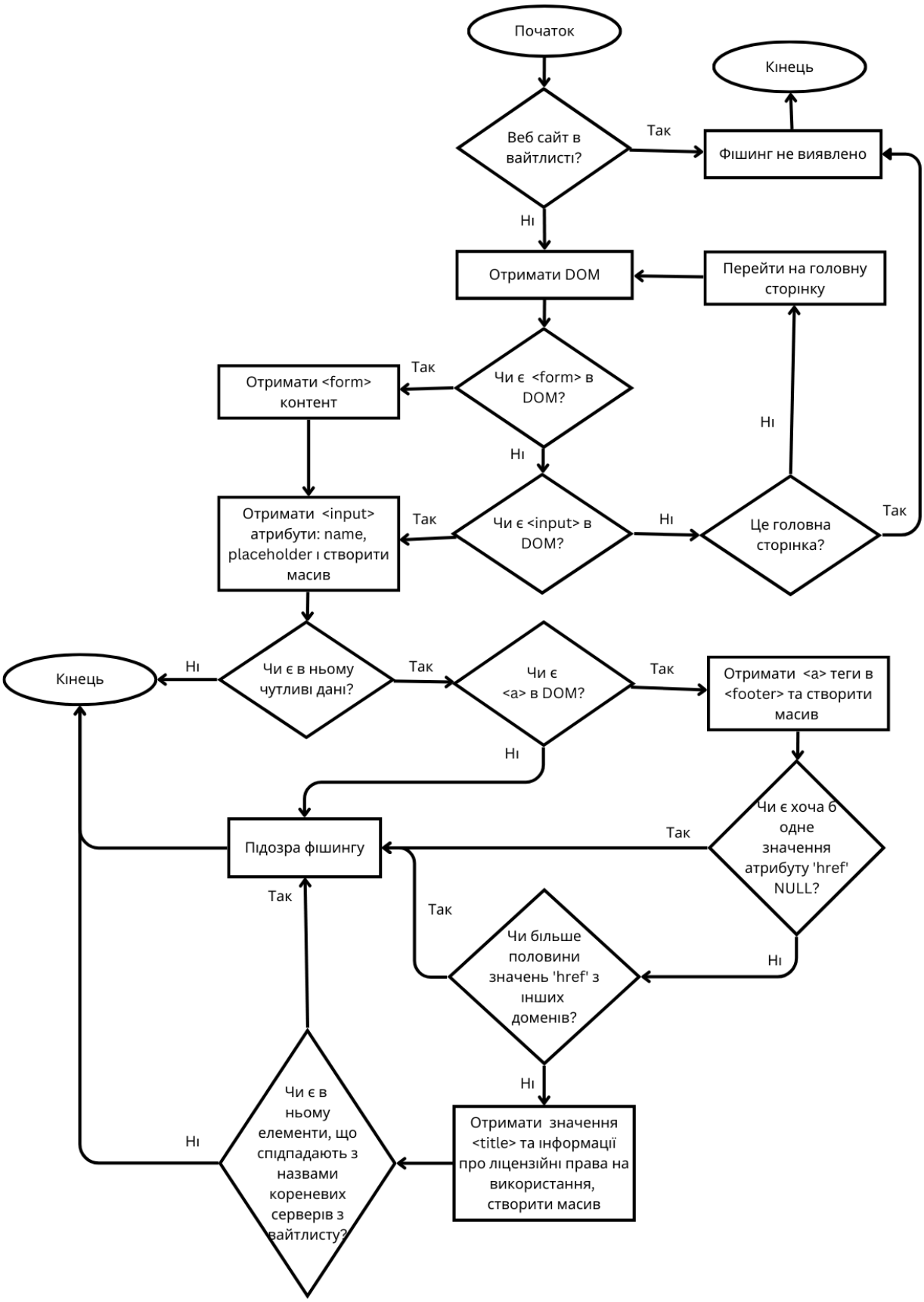

Рисунок 3.2 – Алгоритм виявлення фішингових веб сайтів на основі DOM

Запропонований алгоритм частково залучає Компонент 1, а саме первинну перевірку посилання на предмет його перебування в чорному списку Google Safe Browsing серед відомих загроз. В подальшому алгоритм можна розширити більшою кількістю перевірок.

3.3 Компонент 3: Евристичний аналіз веб сайту

Евристичний аналіз веб сайту з метою виділення фішингових елементів буде являти собою синтез аналізу адреси сайту, його інформаційного наповнення, заголовків, форм, полей, їх параметрів, атрибутів та значень використовуваних JavaScript функцій, інформації про домен, протоколу передачі даних, сертифікату якщо такий є та інших нетипових особливостей пов’язаних з третіми веб сайтами.

Достатньо важливо зазначити, що перевірка має бути реалізована на стороні сервера, аби зняти навантаження з клієнта і не витрачати значну кількість обчислювальних потужностей і не перевантажувати мережу. Отже 3 компонент системи детектування фішингових веб сайтів буде логічним продовженням перших двох адже він також вимагає здійснення перевірки як адреси веб сайту так і його DOM контенту.

В рамках цієї роботи було обрано некласичний евристичний підхід індивідуальних перевірок, запропонованих в таблиці 2.2.3 аби оминути проблему створення так званих сигнатур або ж паттернів фішингових веб сайтів та формування датасету з огляду на доступні обчислювальні потужності та часові ресурси. Для вирішення проблеми призначення коефіцієнтів важливості кожного атрибуту для визначення фішингових веб сайтів пропонується створити зважену матрицю атрибутів та розподіл ймовірності виявлення фішингу між можливими значеннями кожного атрибуту. В таблиці 3.3.1 запропонований зразок зваженого розподілу атрибутів.

Таблиця 3.3.1

Зважений розподіл атрибутів

Група атрибутів	**Назва атрибуту**	**Вага**
Адресний рядок (URL)	1. Використання сервісів скорочення URL	$\frac{0.8}{24}$
	2. Додавання до домену суфіксу або префіксу символу “-”	$\frac{1.2}{24}$
	3. Довгий URL	$\frac{0.8}{24}$
	4. Використання протоколу https для передачі даних	$\frac{0.4}{24}$
	5. Перенаправлення в URL за допомогою “//”	$\frac{0.8}{24}$
	6. Використання фрази “https” в доменній частині адреси	$\frac{0.8}{24}$
	7. Наявність символу “@” в URL	$\frac{0.4}{24}$
Інформаційне наповнення веб сайту та JavaScript код	8. Кількість перенаправлень	$\frac{0.8}{24}$
	9. Використання перенаправлення з <iframe>	$\frac{1.1}{24}$
	10. Наявність сторінки з формою для авторизації та/або полів для введення даних	$\frac{2.2}{24}$
	11. Налаштування рядка стану / Використання JavaScript функції "onMouseOver" для маніпуляції адреси в пошуковій стрічці браузера	$\frac{1.6}{24}$
	12. Реагування на правий клік миші	$\frac{1.2}{24}$
	13. Кількість тегів <a> в HTML	$\frac{1.6}{24}$

Продовження табл. 3.1.2

	14. Значення атрибуту “href” тегів <a> в <footer>	$\frac{1}{24}$
	15. Співпадіння контенту тегу <title> та інформації про авторські права з назвою домену	$\frac{0.8}{24}$
	16. Кількість посилань спрямованих на ресурси інших доменів	$\frac{0.8}{24}$
	17. Джерело походження піктограми	$\frac{1.6}{24}$
Домен	18. DNS записи	$\frac{0.8}{24}$
	19. Час існування домену (субдомену)	$\frac{0.4}{24}$
	20. Термін реєстрації домену (субдомену)	$\frac{0.5}{24}$
	21. Кореляція між інформацією про власника домену та URL	$\frac{0.4}{24}$
	22. Google PageRank	$\frac{1}{24}$
Нетипові особливості	23. Кількість сторонніх доменів в запитах для завантаження ресурсів	$\frac{1.4}{24}$
	24. Значення Server Form Handler пусте або "about: blank"	$\frac{1.6}{24}$

Виявивши атрибути для перевірки, наступним кроком є створення вірогіднісного розподілу впливу значень атрибутів на процес виявлення фішингу для кожного з 24 атрибутів. Це буде показано в таблиці з 3.3.2 синтезуючи підходи, описані в роботах [51-57].

Таблиця 3.3.2

Вірогіднісний розподіл впливу значень атрибутів на процес виявлення фішингу

Назва атрибуту	**Можливі значення**	**Ймовірність фішингу**
1. Використання сервісів скорочення URL	Сервіс скорочення URL використано	$\frac{7}{10}$
	Сервіс скорочення URL не використано	$\frac{3}{10}$
2. Додавання до домену суфіксу або префіксу символ “-”	Символ “-” використано	$\frac{6}{10}$
	Символ “-” не використано	$\frac{4}{10}$
3. Довгий URL	Довжина більша 54 символів	$\frac{7}{10}$
	Довжина менша 54 символів	$\frac{3}{10}$
4. Використання протоколу https для передачі даних	Дані передаються за допомогою протоколу http	$\frac{7}{10}$
	Дані передаються за допомогою протоколу https	$\frac{3}{10}$
5.Перенаправлення в URL за допомогою “//”	Перенаправлення в URL за допомогою “//” використано	$\frac{6}{10}$
	Перенаправлення в URL за допомогою “//” не використано	$\frac{4}{10}$
6. Використання фрази “https” в доменній частині адреси	Фраза “https” використана в доменній частині адреси	$\frac{9}{10}$
	Фраза “https” не використана в доменній частині адреси	$\frac{1}{10}$

Продовження табл. 3.3.2

7. Наявність символу “@” в URL	Наявність символу “@” в URL	$\frac{7}{10}$
	Відсутність символу “@” в URL	$\frac{3}{10}$
8. Кількість перенаправлень	Кількість редіректів ≤ 1	$\frac{2}{10}$
	Кількість редіректів 2 ≤ 4	$\frac{3}{10}$
	Кількість редіректів > 5	$\frac{5}{10}$
9. Використання перенаправлення з <iframe>	Перенаправлення з <iframe> використовується	$\frac{7}{10}$
	Перенаправлення з <iframe> не використовується	$\frac{3}{10}$
10. Наявність сторінки з формою для авторизації та/або полів для введення даних	Сайт має <form> або <input>	$\frac{8}{10}$
	Форми для авторизації та полів немає	$\frac{2}{10}$
11. Налаштування рядка стану / Використання JavaScript функції onMouseOver для маніпуляції адреси в пошуковій стрічці браузера	Проведена кастомізація	$\frac{6}{10}$
	Налаштування за замовчуванням	$\frac{4}{10}$
12. Реагування на правий клік миші	Відключено	$\frac{7}{10}$
	Налаштування за замовчуванням	$\frac{3}{10}$

Продовження табл. 3.3.2

13. Кількість тегів <a> в HTML	<a> теги відсутні	$\frac{8}{10}$
	Кількість тегів <a> в HTML ≥ 1	$\frac{2}{10}$
14. Значення атрибуту “href” тегів <a> в <footer>	Посилання на сторінки в рамках домену	$\frac{2}{10}$
	Посилання на сторінки інших доменів	$\frac{3}{10}$
	Пусті посилання (“#”)	$\frac{5}{10}$
15. Співпадіння контенту тегу <title> та інформації про авторські права з назвою домену	Високий показник співпадіння ($\geq 75\%$)	$\frac{1}{10}$
	Часткове співпадіння ($< 75\%$)	$\frac{2}{10}$
	Повне неспівпадіння	$\frac{7}{10}$
16. Кількість посилань спрямованих на ресурси інших доменів	Більше половини посилань спрямовані на ресурси інших доменів	$\frac{7}{10}$
	Менше половини посилань спрямовані на ресурси інших доменів	$\frac{3}{10}$
17. Джерело походження піктограми	Той самий домен	$\frac{3}{10}$
	Інший домен	$\frac{7}{10}$
18. DNS записи	Приховані, невідомі, пусті	$\frac{8}{10}$
	Повні	$\frac{2}{10}$

Продовження табл. 3.3.2

19. Час існування домену (субдомену)	Більше 1 місяця	$\frac{1}{10}$
	Менше 1 місяця	$\frac{9}{10}$
20. Термін реєстрації домену (субдомену)	Більше 1 року	$\frac{1}{10}$
	Менше 1 року	$\frac{9}{10}$
21. Кореляція між інформацією про власника домену та URL	Співпадіння (> 25%)	$\frac{1}{10}$
	Невідповідність	$\frac{9}{10}$
22. Google PageRank	Показник = 0	$\frac{9}{10}$
	Показник ≥ 1	$\frac{1}{10}$
23. Кількість сторонніх доменів в запитах для завантаження ресурсів	Кількість сторонніх доменів в запитах для завантаження ресурсів ≥ 50%	$\frac{7}{10}$
	Кількість сторонніх доменів в запитах для завантаження ресурсів < 50%	$\frac{3}{10}$
24. Значення Server Form Handler	Значення Server Form Handler NULL, "about: blank" або “#”	$\frac{8}{10}$
	Валідне значення Server Form Handler	$\frac{2}{10}$

Як зазначалося в роботі [35], для розрахунку індивідуальної ентропії атрибутів буде використаний видозмінена концепція Клода Шеннона $H_{attribute} = p_1 \log_s(1/p_1) + p_2 \log_s(1/p_2) + ... + p_s \log_s(1/p_s)$, де p1, p2, ps 3 – вірогідніснi показники для різних значень атрибутів, а s – кількість таких значень.

Загальна ентропія системи при цьому виражається як $H = \sum_{i=1}^{k} H_i * w_i$, де H_i, w_i – ентропія і вага кожного i-го атрибута відповідно, k – кількість атрибутів.

Робиться припущення, що якісна оцінка ефективності такого рішення буде покращуватися зі збільшенням кількості атрибутів та правильністю присвоєння ймовірностей впливу на процес детектування фішингу. Показник підозрілості сайту розраховується як $\sum_{i=1}^{k} w_i * p_i$, де w_i , p_i – вага і ймовірність впливу конкретного значення кожного атрибута на процес виявлення фішингу відповідно, k – кількість атрибутів.

Використовуючи ці метрики можна визначити ентропію вибраної моделі з конкретною кількістю атрибутів та їх значень а також оцінити підозрілість веб сайтів, що і закладено в 3 компонент гібридної системи виявлення фішингових веб сайтів.

Висновок до третього розділу

В результаті синтезування підходів до детектування фішингових елементів в веб сайтах, враховуючи наявні ресурси та експертність в даному питанні було вирішено розробити систему побудовану на гібридному методі, що включає комбінацію підходів проактивного аналізу адреси (компонент 1), інформаційного наповнення та клієнтської частини сайту (компонент 2) та різноманітних евристичних перевірок (компонент 3). Запропоновано методологію визначення ентропії розробленої моделі зі змінною кількістю атрибутів та їх значень а також оцінки підозрілості веб сайтів на базі даної системи.

РОЗДІЛ 4
ТЕХНІЧНА РЕАЛІЗАЦІЯ СИСТЕМИ ДЕТЕКТУВАННЯ ФІШИНГОВИХ ВЕБ САЙТІВ

4.1 Вибір середовища виконання, мов програмування та фреймворків

Для вибору середовища виконання, мов програмування та фреймворків слід відштовхуватися від запропонованої узагальненої моделі проведення фішинг атак з використанням веб сайтів, що була описана в розділі 2.1. Як зазначалося, дана робота ставить на меті детектування фішинг атаки в фазі реалізації попереджуючи розповсюдження зловмисного коду та подальше зараження мережі та/або інших девайсів. Отже, середовище детектування а відтак і середовище виконання - веб браузер. Частина евристичних перевірок зазначених в розділі 3.3, що не потребують значних обчислювальних потужностей і виконуються за долі секунди буде виконана на стороні клієнта за допомогою мови програмування JavaScript.

Переваги використання веб середовища для розробки системи детектування фішингових веб сайтів та вибору мови програмування JavaScript в рамках даної роботи вони полягають в наступному [59]:

- Розподіленість – користувач може працювати з системою з будь-якого місця, пов’язаного з веб сервером по мережі, перебуваючи в будь-якій точці світу.
- Переносимість – веб клієнти (браузери) існують для будь-яких платформ, від настільних комп’ютерів до стільникових телефонів. Веб сервери використовуються для більшості платформ, веб додатки зазвичай пишуться на кросплатформенних мовах програмування.
- Швидкість – JavaScript має тенденцію бути дуже швидким, оскільки запускається відразу в браузері клієнта. Крім того, JavaScript немає необхідності компілювати код перед запуском.
- Навантаження сервера – JavaScript працює на стороні клієнта, тому загалом зменшує попит на сервери, а простим програмам сервер може взагалі не знадобитися.

Проте обмежитися лише клієнтською частиною не має змоги через конфлікт з політикою спільного використання ресурсів з різних джерел (Cross-Origin Resource Sharing) під час запиту HTML сторонніх доменів з локального ПК [60]. Для подолання цього конфлікту необхідне використання серверної частини системи, що буде робити запит інформаційного наповнення веб сайтів та інші дані з доменів відмінних від того, на якому буде виконуватись клієнтська частина. Серверна частина системи буде розроблена на мові програмування Python.

Переваги використання Python як мови програмування для написання серверної частини згідно з [61] полягають в наступному:

- Якість програмного забезпечення - концентрація Python на читабельності, узгодженості і якості ПЗ в цілому відрізняє його від інших інструментів в світі мов написання сценаріїв. Python має розвинену підтримку механізмів багаторазового застосування ПЗ, таких як об'єктно орієнтоване і функціональне програмування.
- Інтеграція компонентів – в сценаріях Python можна легко взаємодіяти з іншими частинами додатку, використовуючи різні механізми інтеграції. Така інтеграція дозволяє використовувати Python в якості інструмента для налаштування та розширення взаємодії з іншими продуктами.

Отже Python є необхідним інструментом у розробці серверної частини додатку. На ньому пишеться багато програмних платформ, які визначають структуру програмної системи; програмне забезпечення, яке полегшує розробку програмного проекту – фреймворки. Фреймворки визначають структуру, задають правила і надають необхідний набір інструментів для розробки.

Для зручності модульної розробки системи, переважно побудованої на мові програмування Python було прийняте рішення використати мікро веб фреймворк Flask. Цей фреймворк надає всі засоби, бібліотеки і технології, необхідні для створення як веб додатків так і веб сайтів, а отже підходить для досягнення поставленої мети.

4.2 Опис системи детектування фішингових веб сайтів

Як було визначено в розділі 3, система детектування фішингових веб сайтів складається з трьох компонентів: перевірки URL за допомогою Google API, індивідуального аналізу DOM веб сайту та ряду евристичних перевірок, що стосуються використовуваних JavaScript функцій, інформації про домен і т. д. Кожен компонент має мати окремий користувацький інтерфейс для вводу даних. Перевірки, що займають більше долі секунди мають виконуватися на серверній частині сайту для оптимізації використовуваних ресурсів.

Використовуючи Flask дуже зручно підтримувати структуру такого проекту адже за замовчуванням веб фреймворк “слухає” зміни файлів в конкретних директоріях і відображає це в консолі серверної частини. За замовчуванням, шаблони і статичні файли зберігаються в внутрішніх директоріях всередині дерева вихідних текстів на Python, названі "templates" та "static" відповідно [61]. Основна логіка роботи системи з описом роботи інтерфейсів, перенаправленням та імпортуванням функцій та бібліотек знаходиться в “app.py” і відображена в Додатку Б. Всі додаткові Python бібліотеки завантажуються з “./venv/Include/Lib/site-packages/” і самописного файлу “shutenko_functions.py”. Розробляючи веб застосунок, була використана рекомендована в документації [65] файлова структура проекту – результат відображений в Додатку В. На рис. 4.1 зображена головна панель системи.

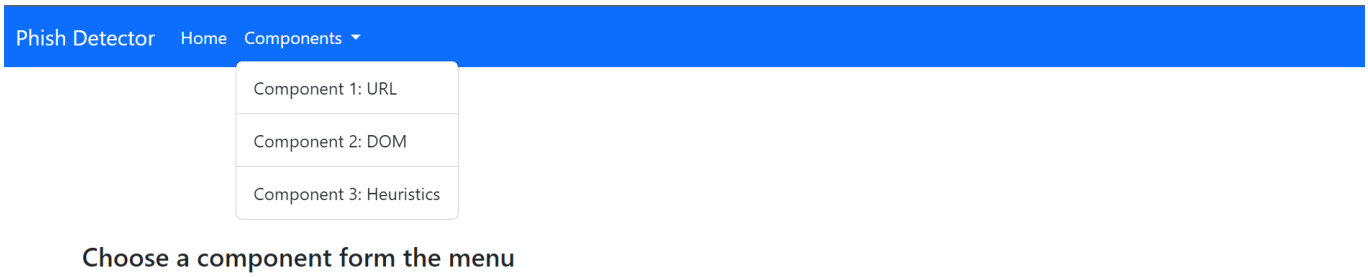

Рисунок 4.1 – Головна панель системи

Як бачимо користувач має можливість обрати конкретний компонент системи для перевірки веб сайту одним з трьох підходів. Приклад використання 2 компоненту системи наведено на рис. 4.2.

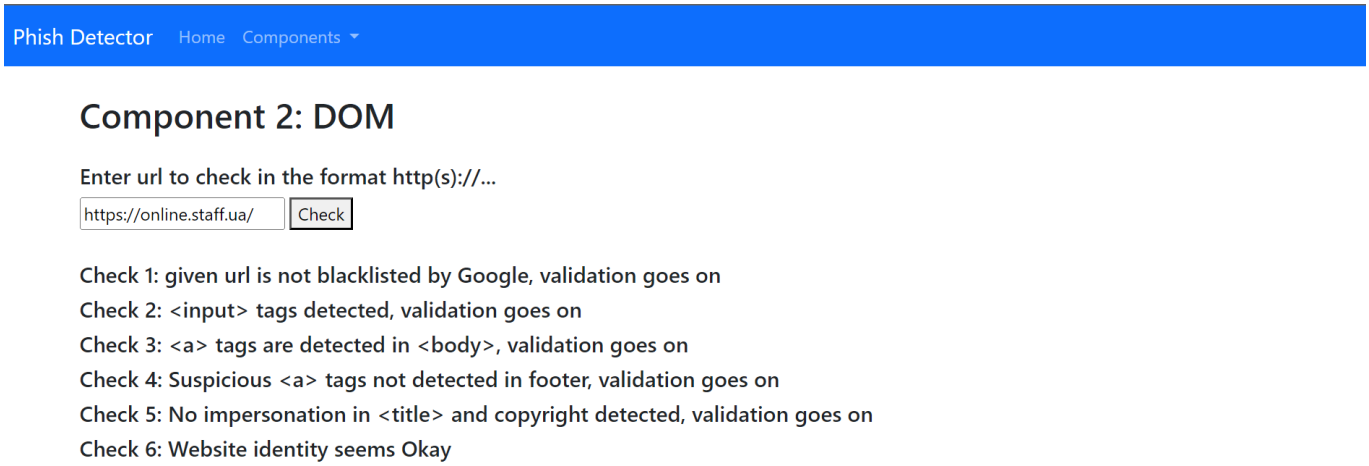

Рисунок 4.2 – Компонент 2: Аналіз інформаційного наповнення веб сайту

Система також здатна обробляти деякі типи помилок, для цього в головному конфігураційному файлі “app.py” передбачена логіка, додатково використовуються відповіді від Google API, описаному у розділі 3.1, приклад такої відповіді наведено на рис. 4.3.

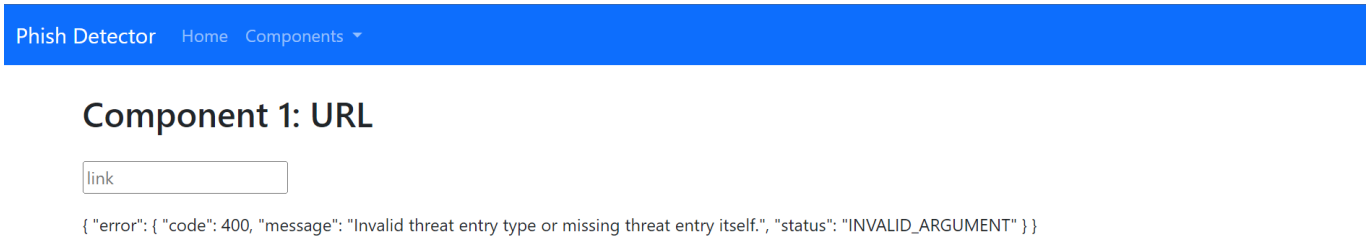

Рисунок 4.3 – Обробка помилки при використанні компоненту 1

4.3 Дослідна експлуатація системи виявлення фішингових веб сайтів

Проведення дослідної експлуатації системи виявлення фішингових веб сайтів відбувається в чотири етапи:

1. Збір даних – Необхідно зібрати набір снепшотів як валідних так і фішингових веб сайтів з різних джерел. Ці дані слугують основою для проведення евристичних перевірок і оптимізації коефіцієнтів, запропонованих в розділі 3.3.

2. Тестування – Систему необхідно піддавати тестуванню на реальних веб сайтах, які містять фішингові елементи з датасету 1. Це дозволяє оцінити, наскільки ефективно система виявляє ці елементи та як швидко реагує на нові фішингові техніки.

3. Оцінка результатів – Після тестування системи проводиться оцінка її результатів. Виявлені фішингові елементи порівнюються з наявними даними про фішинг, і проводиться аналіз точності, чутливості та інших особливостей системи.

4. Вдосконалення – Базуючись на результатах дослідної експлуатації, система може піддаватися вдосконаленню та оптимізації. Можуть бути покращені існуючі або впроваджені нові алгоритми, методи, пришвидчена обробка даних з метою підвищення ефективності та надійності системи я також переоцінки вагових та імовірнісних коефіцієнтів запропонованих в таблицях 3.3.1 та 3.3.2.

Українські дослідники, серед яких [66] та [67] а також іноземні автори [15, 25, 27, 42, 45, 49, 54-57] проводили дослідну експлуатацію переважно використовуючи датасети для машинного навчання що включають значення {-1}, {0} та {1} що відповідно виражають різні стани атрибутів і не враховують певні евристичні правила, запропоновані в розділі 3.3. Відтак для проведення комплексної дослідної експлуатації системи детектування фішингових веб сайтів з різною кількістю атрибутів необхідно зібрати унікальний набір даних, що не є метою кваліфікаційної роботи.

Оцінка ефективності запропонованого рішення, що в випадку розробленої системи є комбінацією її ентропії та показнику підозрілості показала, що запропонований розподіл ймовірностей впливу значень 24 атрибутів (розділ 3.3) на

виявлення фішингових елементів у веб сайтах має наступні результати, надані в Таблиці 4.3.1.

Таблиця 4.3.1

Індивідуальні зважені значення ентропії атрибутів

Назва атрибуту	**Зважене значення ентропії**
1. Використання сервісів скорочення URL	$H_1 = 0.026$
2. Додавання до домену суфіксу або префіксу символу “-”	$H_2 = 0.049$
3. Довгий URL	$H_3 = 0.026$
4. Використання протоколу https для передачі даних	$H_4 = 0.015$
5. Перенаправлення в URL за допомогою “//”	$H_5 = 0.033$
6. Використання фрази “https” в доменній частині адреси	$H_6 = 0.014$
7. Наявність символу “@” в URL	$H_7 = 0.015$
8. Кількість перенаправлень	$H_8 = 0.062$
9. Використання перенаправлення з <iframe>	$H_9 = 0.04$
10. Наявність сторінки з формою для авторизації та/або полів для введення даних	$H_{10} = 0.08$
11. Налаштування рядка стану / Використання JavaScript функції onMouseOver для маніпуляції адреси в пошуковій стрічці браузера	$H_{11} = 0.062$
12. Реагування на правий клік миші	$H_9 = 0.044$
13. Кількість тегів <a> в HTML	$H_{13} = 0.049$
14. Значення атрибуту “href” тегів <a> в <footer>	$H_{14} = 0.062$
15. Співпадіння контенту тегу <title> та інформації про авторські права з назвою домену	$H_{15} = 0.038$
16. Кількість посилань спрямованих на ресурси інших доменів	$H_{16} = 0.029$

Продовження табл. 4.3.1

17. Джерело походження піктограми	$H_{17} = 0.059$
18. DNS записи	$H_{18} = 0.023$
19. Час існування домену (субдомену)	$H_{19} = 0.007$
20. Термін реєстрації домену (субдомену)	$H_{20} = 0.01$
21. Кореляція між інформацією про власника домену та URL	$H_{21} = 0.007$
22. Google PageRank	$H_{22} = 0.019$
23. Кількість сторонніх доменів в запитах для завантаження ресурсів	$H_{23} = 0.051$
24. Значення Server Form Handler пусте або "about: blank"	$H_{24} = 0.065$

Загальна ентропія розробленої системи з 24-ма атрибутами та їх ймовірністими значеннями $H = 0.885$. Показник підозрілості сайту залежить від кожного конкретного випадку відповідно. Середній час перевірки URL трьома компонентами системи надано в таблиці 4.3.2.

Таблиця 4.3.2

Середній час перевірки веб сайту

Назва компоненту	Компонент 1: URL	Компонент 2: DOM	Компонент 3: Евристичні правила
Середній час виконання коду	$t_1 = 0.05$ секунд	$t_2 = 0.15$ секунд	$t_3 = 0.45$ секунд

В подальшому ентропію розробленої системи можна збільшити розширивши кількість аналізованих атрибутів. Середній час перевірки можна зменшити шляхом розробки додаткового модуля взаємодії з локальною базою даних, що міститиме хеші даних, які в поточній реалізації утримуються від третіх сторін за допомогою прикладних програмних інтерфейсів. Порівняння ефективності використання

розробленої системи буде можливе після створення нової моделі з розширеною кількістю атрибутів та проведеною дослідною експлуатацією з використанням коректного набору даних.

Висновок до четвертого розділу

Для успішної технічної реалізації системи детектування фішингових веб сайтів середовищем детектування а відтак виконання був обраний веб браузер. Частина евристичних перевірок, що не потребують значних як обчислювальних потужностей так і часових ресурсів виконана на стороні клієнта за допомогою мови програмування JavaScript. Для подолання конфлікту політики спільного використання ресурсів з різних джерел (Cross-Origin Resource Sharing) під час завантаження DOM сайтів, що будуть перевірятися прийняте рішення розробки серверної частини системи, що буде робити запит інформаційного наповнення веб сайтів та інші дані з доменів відмінних від того, на якому буде виконуватись клієнтська частина. Серверна частина системи була розроблена на мові програмування Python. Описані переваги та причини вибору зазначених мов програмування та мікро веб фреймворку Flask для орієнтації на модульну архітектуру веб застосунку.

В результаті була розроблена система детектування фішингових веб сайтів, що складається з трьох компонентів: перевірки URL за допомогою Google API, індивідуального аналізу DOM веб сайту та ряду евристичних перевірок, що стосуються використовуваних JavaScript функцій, інформації про домен і т. д. Користувацький інтерфейс для вводу даних передбачено для кожен компоненту системи. Описана файлова структура технічного рішення.

Зазначені етапи проведення дослідної експлуатації системи, розрахована загальна ентропія розробленої системи та середній час виконання перевірок для всіх її компонентів. Надані рекомендації для покращення системи та проведення подальших досліджень в рамках тематики детектування фішингових веб сайтів.

ВИСНОВКИ

В першому розділі дипломної роботі було висвітлено актуальність такої проблеми як реалізація фішингових атак за допомогою підроблених веб сайтів. Фішинг є серйозною загрозою безпеки інфраструктури для організацій будь якої форми власності, державних інститутів, банків, підприємств промисловості та багатьох інших. Постійна еволюція підходів, що використовуються зловмисниками та збільшення кількості атак нульового дня вимагає нових засобів захисту для збереження високого рівня захищеності інформаційних систем. Традиційні засоби протидії таким атакам побудовані на списках не здатні належним чином протистояти даним загрозам, відтак у роботі проаналізовані складніші підходи до їх детектування.

В другому розділі дипломної роботі було визначено, що детектування та блокування фішингових атак доцільно проводити в фазі їх реалізації. Було проведено огляд та аналіз вітчизняних та закордонних підходів, що використовуються для вирішення задачі детектування фішингових елементів у веб сайтах та роз’яснено доцільність їх використання для створення гібридного методу детектування фішингових веб сайтів.

Третій розділ дипломної роботі було присвячено аналізу та роз’ясненню використання та/або покращення відомих підходів до детектування фішингових елементів у веб сайтах.

Проаналізувавши методи детектування фішингових елементів за допомогою URL аналізу, був розроблений 1-й компонент гібридного методу детектування фішингових веб сайтів – інтеграція з прикладним програмним інтерфейсом Safe Browsing APIs (v4) від Google, що використовується як перший рівень виявлення відомих небезпечних веб ресурсів.

Проаналізувавши методи детектування фішингових елементів за допомогою аналізу контенту веб сайтів, було запропоновано проводити індивідуальний аналіз елементів DOM (HTML), відмовившись від ідеї створення степшоту сайту, що

призвело до зниження часу аналізу за рахунок підвищення швидкості відтворення, що стало 2-м компонентом гібридного методу детектування фішингових веб сайтів.

Проаналізувавши евристичні підходи до детектування фішингових веб сайтів було запропоновано перелік перевірок для виявлення підозрілих елементів у підроблених сайтах, які ще не потрапили до чорних списків популярних вендорів, що в результаті стало 3-м агрегованим компонентом гібридного методу детектування фішингових веб сайтів. На базі цього методу було запропоновано модель, що складається з 24 атрибутів, кожен з яких має більше як 2-а можливі значення. Також запропоновано методологію оцінки ентропії системи зі змінною кількістю атрибутів, та коефіцієнту підозрілості веб сайту.

В результаті проведеної дослідної експлуатації була розрахована загальна ентропія розробленої системи з 24-ма атрибутами, їх ймовірнісними значеннями та середній час виконання перевірки веб сайту кожним з розроблених компонентів. Надано рекомендації для підвищення значення ентропії системи та виявлено сфери подальших досліджень в рамках тематики, а саме створення набору даних для проведення тренування моделі, покращення коефіцієнтів та розширення кількості атрибутів за допомогою методів машинного навчання.

Поставлені завдання були виконані у повному обсязі. Розроблена система виявлення фішингових атак, побудована на базі вперше запропонованого гібридного методу детектування фішингових веб сайтів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Фесенко А. О., Шутенко Д. В. Заходи, що проводяться для захисту від застосування соціальної інженерії, зокрема фішингу: матеріали VI Міжнародної студентської олімпіади “Шляхи та механізми захисту інформаційного простору України від шкідливих інформаційно-психологічних впливів”. — 2020.

2. Фесенко А. О., Шутенко Д. В. Підходи до управління інформаційною безпекою в контексті атак що використовують методи соціальної інженерії: матеріали VI Міжнародної науково-практичної конференції “Проблеми кібербезпеки інформаційно-комунікаційних систем” (PCSITS). — 2021.

3. Фесенко А. О., Шутенко Д. В. Експлуатація вразливостей людини та захист від негативного інформаційно-психологічного впливу: матеріали VII Міжнародної студентської олімпіади “Шляхи та механізми захисту інформаційного простору України від шкідливих інформаційно-психологічних впливів”. – 2021.

4. А. О. Фесенко, Д. В. Шутенко. Матеріали 4-ї Всеукраїнської науково-практичної конференції “Перспективні напрями захисту інформації” – Одеса, Одеська національна академія зв’язку імені О. С. Попова, 2020. – “Заходи, що проводяться для захисту від застосування соціальної інженерії, зокрема фішингу”. — 2020.

5. Whitepaper “Social Engineering: Exploiting the Weakest Links” by The European Network and Information Security Agency. URL: https://www.enisa.europa.eu/publications/archive/social-engineering/at_download/fullReport

6. Оцінювання захищеності інформації в комп’ютерних системах за соціоінженерним підходом / Мохор В.В., Цуркан О.В., Цуркан В.В., Герасимов Р.П. [Електронний ресурс]. – Режим доступу: http://ceur-ws.org/Vol-2067/paper13.pdf

7. Kevin D. Mitnick / William L. Simon / Steve Wozniak // The Art of Deception: Controlling the Human Element of Security. – 2002. – 577 p.

8. Internet Crime Report 2022 / FEDERAL BUREAU OF INVESTIGATION // Internet Crime Complaint Center. URL: https://www.ic3.gov/Media/PDF/AnnualReport/2022_IC3Report.pdf

9. Шутенко Д. В., Фесенко А. О. Характеристика основних методів соціальної інженерії: матеріали V Міжнародної студентської олімпіади “Шляхи та механізми захисту інформаційного простору України від шкідливих інформаційно-психологічних впливів”. – 2019.

10. Cybersecurity Threatscape Q2 2022 report / Positive Technologies. URL: https://www.ptsecurity.com/ww-en/analytics/cybersecurity-threatscape-2022-q2/

11. Біленко А.А., Гайдур Г.І. ДОСЛІДЖЕННЯ МЕТОДИКИ ПРОТИДІЇ СОЦІАЛЬНОМУ ІНЖИНІРИНГУ ДЛЯ ЗАХИСТУ ІНФОРМАЦІЇ НА ОБ’ЄКТАХ ІНФОРМАЦІЙНОЇ ДІЯЛЬНОСТІ [Електронний ресурс]. – Режим доступу: http://www.dut.edu.ua/uploads/p_422_79548521.pdf

12. Шутенко Д. В., Фесенко А. О. Аналіз атак з використанням соціальної інженерії: матеріали VII Міжнародної науково-практичної конференції “Information Technology and Interactions”. – 2020.

13. Bernard Oosterloo, Peter Geurtsen, Theo Thiadens, Irene van Santen, Jeffrey Hicks, Peter Wemmenhove. Managing Social Engineering Risk - Making social engineering transparent. URL: https://essay.utwente.nl/59233/1/scriptie_B_Oosterloo.pdf

14. Лаптєв Сергій. УДОСКОНАЛЕНИЙ МЕТОД ЗАХИСТУ ПЕРСОНАЛЬНИХ ДАНИХ ВІД АТАК ЗА ДОПОМОГОЮ АЛГОРИТМІВ СОЦІАЛЬНОЇ ІНЖЕНЕРІЇ. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 4(16), 45-62 с. [Електронний ресурс]. – Режим доступу: https://doi.org/10.28925/2663-4023.2022.16.4562

15. Dutta AK. Detecting phishing websites using machine learning technique. PLoS One. Oct 11;16(10):e0258361. doi: 10.1371/journal.pone.0258361. PMID: 34634081; PMCID: PMC8504731. – 2021.

16. Про інформацію: Закон України від 02.10.92, No № 2658-XII. [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2657-12

17. Про захист персональних даних: Закон України від 23.02.2012, № 4452-VI. [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2297-17

18. Шутенко Д., Фесенко А. Заходи та засоби для захисту від соціальної інженерії, зокрема фішингу. Київський національний університет імені Тараса Шевченка. [Електронний ресурс]. – Режим доступу: http://10.23.8.5:8080/xmlui/handle/123456789/1196

19. Про Несанкціоновані дії з інформацією, яка оброблюється в електронно-обчислювальних машинах (комп'ютерах), автоматизованих системах, комп'ютерних мережах або зберігається на носіях такої інформації, вчинені особою, яка має право доступу до неї: Стаття 362 в редакції Закону № 2289-IV від 23.12.2004; із змінами, внесеними згідно із Законами № 770-VIII від 10.11.2015, № 2617-VIII від 22.11.2018. [Електронний ресурс]. – Режим доступу: https://protocol.ua/ua/kriminalniy_kodeks_ukraini_stattya_362/

20. Про Порушення правил експлуатації електронно-обчислювальних машин (комп'ютерів), автоматизованих систем, комп'ютерних мереж чи мереж електрозв'язку або порядку чи правил захисту інформації, яка в них оброблюється: Стаття 363 в редакції Закону № 2289-IV від 23.12.2004 ; із змінами, внесеними згідно із Законом № 2617-VIII від 22.11.2018. [Електронний ресурс]. – Режим доступу: https://protocol.ua/ua/kriminalniy_kodeks_ukraini_stattya_363/

21. Про Несанкціоноване втручання в роботу інформаційних (автоматизованих), електронних комунікаційних, інформаційно-комунікаційних систем, електронних комунікаційних мереж: Стаття 361 в редакції Законів № 908-IV від 05.06.2003 , № 2289-IV від 23.12.2004 ; із змінами, внесеними згідно із Законом № 770-VIII від 10.11.2015 ; в редакції Закону № 2149-IX від 24.03.2022. [Електронний ресурс]. – Режим доступу: https://protocol.ua/ua/kriminalniy_kodeks_ukraini_stattya_361/

22. ІСТОРІЯ ІНФОРМАЦІЙНО-ПСИХОЛОГІЧНОГО ПРОТИБОРСТВА: підручник / Я.М.Жарков, Л.Ф.Компанцева, В.В.Остроухов В.М.Петрик, М.М.Присяжнюк, Є.Д.Скулиш. Київ: Науково-видавничий відділ Національної академії СБ України. – 2021. – 212 с.

23. Thapa, Chandra, Jun Tang, Alsharif Abuadbba, Yansong Gao, Seyit Ahmet Çamtepe, Surya Nepal, Mahathir Almashor and Yifeng Zheng. Evaluation of Federated Learning in Phishing Email Detection. Sensors (Basel, Switzerland) 23. – 2020. URL: https://arxiv.org/abs/2007.13300

24. Szafranski, R. Theory of Information Warfare: Preparing For 2020. Airpower Journal. URL: http://www.airpower.au.af.mil/airchronicles/apj/apj95/spr95_files/szfran.htm.

25. Abdulhamit Subasi, Emir Kremic, Comparison of Adaboost with MultiBoosting for Phishing Website Detection, Procedia Computer Science, Volume 168. – 2020. – 272-278 p. ISSN 1877-0509. URL: https://doi.org/10.1016/j.procs.2020.02.251

26. M.A. Adebowale, K.T. Lwin, E. Sánchez, M.A. Hossain, Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text, Expert Systems with Applications, Volume 115. – 2019. – 300-313 p, ISSN 0957-4174. URL: https://doi.org/10.1016/j.eswa.2018.07.067.

27. Dutta AK. Detecting phishing websites using machine learning technique. PLoS One. 2021 Oct 11;16(10):e0258361. doi: 10.1371/journal.pone.0258361. PMID: 34634081; PMCID: PMC8504731. – 2021.

28. Aldawood, Hussain & Skinner, Geoff. A Critical Appraisal of Contemporary Cyber Security Social Engineering Solutions: Measures, Policies, Tools and Applications. 10.1109/ICSENG.2018.8638166. – 2018.

29. A. Le, A. Markopoulou, and M. Faloutsos. Phishdef: Url names say it all. 2011 Proceedings IEEE INFOCOM. – 2020. – pp. 191–195.

30. S. C. Jeeva and E. B. Rajsingh. Intelligent phishing url detection using association rule mining. Hum.-Centric Comput. Inf. Sci., vol. 6, no. 1, p. 10. – 2016.

31. Y. Li, Z. Yang, X. Chen, H. Yuan, and W. Liu. A stacking model using URL and HTML features for phishing webpage detection. Future Gener. Comput. Syst., vol. 94. – 2020. – pp. 27–39.

32. Rao, Routhu Srinivasa and Alwyn Roshan Pais. Detecting Phishing Websites using Automation of Human Behavior. Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security. – 2017.

33. Wenyin L., Huang G., Xiaoyue L., Min Z., Deng X. Detection of Phishing Webpages based on Visual Similarity. L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, X. Deng. 14th International Conference on World Wide Web (WWW): Special Interest Tracks and Posters. – 2005.

34. Ronda T., Saroiu S.,Wolman A. Itrustpage: a user-assisted anti-phishing tool / T. Ronda, S. Saroiu, A. Wolman // ACM SIGOPS Operating Systems Review, Vol. 42. – 2008. – pp. 261-272.

35. Buchyk S., Shutenko D., Toliupa S., Phishing Attacks Detection. IX International Scientific Conference “Information Technology and Implementation" (IT&I-2022), Workshop Proceedings, Kyiv, Ukraine, November 30 - December 02, 2022., Kyiv, Ukraine, pp. 193–201. URL: https://ceur-ws.org/Vol-3384/Short_7.pdf.

36. Chizari, Hassan & Zulkurnain, Ahmad & Hamidy, Ahmad & Husain, Affandi. (2015). Social Engineering Attack Mitigation. International Journal of Mathematics and Computational Science. 1. pp. 188-198.

37. Number of unique phishing sites detected worldwide from 3rd quarter 2013 to 3rd quarter 2022. Statista. URL: https://www.statista.com/statistics/266155/number-of-phishing-domain-names-worldwide/

38. Moore T., Clayton R. The Economics of Online Crime / T. Moore, R. Clayton // Journal of Economic Perspectives—Volume 23, Number 3. —2009. – p 3–20.

39. Zouina, M. and Outtaj, B. A novel lightweight URL phishing detection system using SVM and similarity index. Human-centric Computing and Information Sciences, 7(1). – 2017. – pp. 17.

40. Overview: What are the Safe Browsing APIs? URL: https://developers.google.com/safe-browsing/v4

41. Chou N., Ledesma R., Teraguchi Y., Boneh D., Mitchell J. C. Client-side defense against Web-based identity theft / N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, J. C. Mitchell // In Proceedings of the Network and Distributed System Security Symposium. – 2004. URL: https://crypto.stanford.edu/SpoofGuard/webspoof.pdf

42. Zhang Y., Egelman S., Cranor L., Hong J. Phinding phish: Evaluating anti-phishing tools / Y. Zhang, S. Egelman, L. Cranor, J. Hong // In Proceedings of the 14th Annual Network and Distributed System Security Symposium. – 2007.

43. Aburrous M., Alamgir M., Prasad K.. Intelligent Phishing Website Detection System using Fuzzy Techniques / M. Aburrous, M. Alamgir Hossain, K. Prasad Dahal // Conference: Information and Communication Technologies: From Theory to Applications. – 2008. – p. 1-6.

44. Dunlop M., Shelly D., Groat S. GoldPhish: Using Images for Content-Based Phishing Analysis [Electronic resource] / M. Dunlop, D. Shelly, S. Groat // Internet Monitoring and Protection (ICIMP) – 2010 – p. 123 – 128 . URL: https://www.researchgate.net/publication/224142945_GoldPhish_Using_Images_for_Content-Based_Phishing_Analysis

45. Afroz S., Greenstadt R., Phishzoo: An automated web phishing detection approach based on profiling and fuzzy matching. / S. Afroz, R. Greenstadt // Technical Report DU-CS-09-03, Drexel University. – 2009.

46. Routhu Srinivasa Rao, Syed Taqi Ali, PhishShield: A Desktop Application to Detect Phishing Webpages through Heuristic Approach, Procedia Computer Science, Volume 54, 2015, Pages 147-156, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2015.06.017. URL: https://www.sciencedirect.com/science/article/pii/S1877050915013411

47. Wenyin L., Huang G., Xiaoyue L., Min Z., Deng X. Detection of Phishing Webpages based on Visual Similarity / L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, X. Deng // 14th International Conference on World Wide Web (WWW): Special Interest Tracks and Posters. – 2005.

48. Ronda T., Saroiu S.,Wolman A. Itrustpage: a user-assisted anti-phishing tool / T. Ronda, S. Saroiu, A. Wolman // ACM SIGOPS Operating Systems Review, Vol. 42. – 2008. – pp 261-272.

49. Rao R.S, Ali S.T., PhishShield: A Desktop Application to Detect Phishing Webpages through Heuristic Approach [Electronic resource] / R.S. Rao, S.T. Ali // Procedia

Computer Science Volume 54. – 2015. – pp. 147-156. URL: https://doi.org/10.1016/j.procs.2015.06.017

50. Shutenko Dmytro. SOFTWARE-BASED PHISHING WEBSITE DETECTION: A COMPREHENSIVE OVERVIEW. Shutenko Dmytro, Buchyk Serhii. Проблеми кібербезпеки інформаційно-телекомунікаційних систем: Збірник матеріалів доповідей та тез; м. Київ, 27 квітня 2023 року; Київський національний університет імені Тараса Шевченка / Редкол.: В.В. Ільченко, д.ф-м.н., проф., (голова); та ін. – К.: ВПЦ "Київський університет". – 2023. – 166 с.

51. Moruf Akin Adebowale, An Intelligent Phishing Detection and Protection Scheme using a fusion of Images, Frames and Text. Moruf Akin Adebowale, Professor Alamgir Hossain, Dr. Khin Lwin, Dr. George, Wilson. University at Chelmsford in Fulfilment. – 2020. URL: https://arro.anglia.ac.uk/id/eprint/706745/1/Adebowale_2020.pdf

52. Rami M. Mohammad, Phishing Websites Features / Rami M. Mohammad, Fadi Thabtah, Lee McCluskey. University of Huddersfield. – 2015. URL: http://eprints.hud.ac.uk/id/eprint/24330/6/MohammadPhishing14July2015.pdf

53. Apruzzese, G., Colajanni, M., Ferretti, L., Guido, A. and Marchetti, M. 'On the effectiveness of machine and deep learning for cyber security'. 10th International Conference on Cyber Conflict (CyCon), Tallinn, Estonia, 29 May-1 June 2018: IEEE. – 2018. – pp. 371-390.

54. Montavon, G., Samek, W. and Müller, K.-R. (2018) 'Methods for interpreting and understanding deep neural networks', Digital Signal Processing, 73. – 2018. – pp. 1-15.

55. Ping Yi, Yuxiang Guan, Futai Zou, Yao Yao, Wei Wang, Ting Zhu. Web Phishing Detection Using a Deep Learning Framework. Wireless Communications and Mobile Computing, vol. 2018, Article ID 4678746. – 2018. – 9 p. URL: https://doi.org/10.1155/2018/4678746

56. Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y. and Alsaadi, F. E. A survey of deep neural network architectures and their applications. Neurocomputing, 234. – 2017. – pp. 11-26.

57. Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. and Lew, M. S. Deep learning for visual understanding: A review. Neurocomputing, 187. – 2016. – pp., pp. 27-48.

58. Програмування. Основи програмування в середовищі Microsoft Visual C++ : конспект лекцій / С. С. Бучик, Р. В. Нетребко. – Житомир : ЖВІ. – 2016. – 204 с.

59. Переваги та недоліки JavaScript [Електронний ресурс]. – Режим доступу: https://hackit-ukraine.com/627-the-advantages-and-disadvantages-of-javascript

60. Chen, Jianjun, et al. We Still Don't Have Secure Cross-Domain Requests: an Empirical Study of CORS. 27th USENIX Security Symposium (USENIX Security 18). – 2018.

61. Лузанов Є. Онлайн середовище для організації спільної роботи над проєктами. Лузанов Є., Росінська Г. НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ. – 2021.

62. Aldawood, Hussain & Skinner, Geoff. Contemporary Cyber Security Social Engineering Solutions, Measures, Policies, Tools and Applications: A Critical Appraisal. – 2019 . URL: https://www.researchgate.net/publication/333531483_Contemporary_Cyber_Security_Social_Engineering_Solutions_Measures_Policies_Tools_and_Applications_A_Critical_Appraisal

63. Бурячок, В. Л., Толубко, В. Б., Хорошко, В. О., Толюпа, С. В. Інформаційна та кібербезпека: соціотехнічний аспект : підручник. ДУТ. – 2015.

64. Shabuddin, Shafaizal & S Sani, Nor & Zainol Ariffin, Khairul Akram & Aliff, Mohd. Feature Selection for Phishing Website Classification. International Journal of Advanced Computer Science and Applications. 11. – 2020. – 587-595.

65. Project Layout: Flask documentation . URL: https://flask.palletsprojects.com/en/2.3.x/tutorial/layout/

66. Тернопольська С. Розпізнавання фішингових сайтів з використанням методів машинного навчання. Тернопольська С. Стьопочкіна І. НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО". – 2019.

67. Покидко О. Автоматизація моніторингу витоку інформації. Покидко О., Кареліна О. Тернопільський національний технічний університет імені Івана Пулюя. – 2021.

ДОДАТОК А

Тіло запиту Lookup API (v4)

```
{
        'client':
        {
            'clientId': 'Dmytro_Shutenko',
            'clientVersion': '1.0.3'
        },
            'threatInfo':
            {
            'threatTypes': ['MALWARE', 'SOCIAL_ENGINEERING',
'UNWANTED_SOFTWARE'],
            'platformTypes': ['ANY_PLATFORM'],
            'threatEntryTypes': ['URL'],
            'threatEntries': [ {'url': '[посилання для перевірки]'} ]
            }
                                    }
```

ДОДАТОК Б

app.py

```
import os
from flask import Flask, render_template, request
from bs4 import BeautifulSoup
import requests
import re
import tldextract
import validators
from shutenko_functions import heuristic_checks


app = Flask(__name__)

whitelist = ["facebook", "google", "meta"] # test

def component_2_check_blacklisted(input_url):
    print("\n input_url: ", input_url)
    url = "https://safebrowsing.googleapis.com/v4/threatMatches:find?key=AIzaSyD5_odg6Etmv_f5coTMEHJpm6GeUDIXFcs"
    payload = "{ 'client': {'clientId':'Dmytro_Shutenko','clientVersion':'1.0.2'}, 'threatInfo': {'threatTypes':['MALWARE', 'SOCIAL_ENGINEERING', 'UNWANTED_SOFTWARE'], 'platformTypes':['ANY_PLATFORM'],'threatEntryTypes': ['URL'], 'threatEntries': [ {'url': '" + input_url + "'} ] } }"
    headers = {'Content-Type': 'application/json'}

    response = requests.request("POST", url, headers=headers, data=payload)
```

```
        print("\n Google API matches: ", response.text)

        if "matches" in response.text:
            return True
        else:
            return False


    def component_2_check_2(soup):
        inputs = soup.findAll("input", {"name" : re.compile("name", re.IGNORECASE)})
# finds all the input tags that have attribute "name" and contain value "name"
        inputs.append( soup.find_all("input", attrs={"name": re.compile("mail")}) )
        inputs.append( soup.find_all("input", attrs={"name": re.compile("login")}) )
        inputs.append( soup.find_all("input", attrs={"name": re.compile("id")}) )
        inputs.append( soup.find_all("input", attrs={"name": re.compile("phone")}) )
        inputs.append( soup.find_all("input", attrs={"name": re.compile("code")}) )
        inputs.append( soup.find_all("input", attrs={"name": re.compile("pass")}) )
        inputs_ = list( filter(None, inputs) ) # remove all empty []
        # print("inputs_: ", inputs_)
        if inputs_:
            return inputs_[0] # return 0-th element
        else:
            return inputs_


    def component_2_check_a_tags_in_body(soup):
        return soup.find_all("a")


    def footer_a_tags_suspicious(footer):
```

```
        footer_a_tags_suspicious_list = footer.find('a', attrs={'href': '', 'href': '#'}) # , 'href': re.compile('^#')
        print("\n footer_a_tags_suspicious: ", footer_a_tags_suspicious_list)
        return footer_a_tags_suspicious_list


    def title_and_copyright_check_impersonation(url, soup): # checks title, copyright and url correlation
        title = soup.title.string
        print("\n title: ", title)
        title = str(title.lower()).split(" ")
        for word in title:
            if word in url:
                return False

        copyright = soup.find_all("copyright")
        if copyright:
            print("\n copyright: ", copyright)
            copyright = str(copyright.lower()).split(" ")
            for word in copyright:
                if word in url:
                    return False
        else:
            copyright = soup.find_all("©")
            print("\n copyright: ", copyright)
            copyright = str(copyright.lower()).split(" ")
            for word in copyright:
                if word in url:
                    return False
        return True
```

```
def check_identity(url, soup):
    extracted = tldextract.extract(url)
    domain = str(extracted.domain) + "." + str(extracted.suffix)
    print("\n domain: " + domain + "\n")
    a_tags = component_2_check_a_tags_in_body(soup)
    a_tags_refferencing_outside = 0
    for a_tag in a_tags:
        if str(domain) not in str(a_tag) and a_tag.get("href")[0] != '/' and a_tag.get("href")[0] != '#':
            a_tags_refferencing_outside+=1
            # (str(domain) + " is not in " + str(a_tag))
        # else:
            # print(str(domain) + " is in " + str(a_tag))
    # print("\n Number of <a> tags refferencing outside:", a_tags_refferencing_outside)
    if a_tags_refferencing_outside > len(a_tags)/2:
        print("more than half of <a> tags point to other domains, PHISHING SUSPECTED")
        return False
    else:
        print(str(a_tags_refferencing_outside) + " out of " + str(len(a_tags)) + " <a> tags point to other domains \n")
        return True



@app.route('/')
```

```
def main():
    return render_template("index.html")


@app.route("/component1")
def component_url():
    return render_template("component_url.html")


@app.route("/component2", methods=["GET", "POST"])
def component_dom():
    if request.method == "POST":
        url = request.form.get("url")
        if not validators.url(str(url)):
            return render_template("component_dom.html",
                input_url = str("URL: " + url),
                check_1 = str("invalid url, try again") )

        if component_2_check_blacklisted(url): # if match found
            print("Blacklisted url: ", url)
            return render_template("component_dom.html",
                input_url = str(url),
                check_1 = str("Check 1: given url is blacklisted by Google, validation stopped, PHISHING DETECTED") )

        response = requests.get(url)
        soup = BeautifulSoup(response.content, 'html.parser')
        # app.logger.debug(soup.prettify())

        inputs = component_2_check_2(soup) # component with input presence checks
```

```
            print("inputs:", inputs)

            if len(inputs) > 0:
                print("\n Check 2: <input>s detected, validation goes on")

                if component_2_check_a_tags_in_body(soup):
                    footer_tag = soup.select('footer, div#footer, .footer')[0]
                    if footer_tag:
                        print("\n footer_tag: ", footer_tag)
                        if footer_a_tags_suspicious(footer_tag):
                            # for footer_a_tag_suspicious in footer_a_tags_suspicious:
                            #     print("tag", footer_a_tag_suspicious, " is suspicious")
                            return render_template("component_dom.html",
                                check_1 = str("Check 1: given url is not blacklisted by Google, validation goes on"),
                                check_2 = str("Check 2: <input> tags detected, validation goes on"),
                                check_3 = str("Check 3: <a> tags are detected in <body>, validation goes on"),
                                check_4 = str("Check 4: Suspicious <a> tags are not detected in footer, validation stopped, PHISHING SUSPECTED") )
                        else:
                            print("\n no suspicious <a> tags found")

                            if title_and_copyright_check_impersonation(url, soup):
                                print("Phishing Suspected (impersonation)")
                                return render_template("component_dom.html",
                                check_1 = str("Check 1: given url is not blacklisted by Google, validation goes on"),
                                check_2 = str("Check 2: <input> tags detected, validation goes on"),
```

```
                check_3 = str("Check 3: <a> tags are detected in <body>, validation
goes on"),
                check_4 = str("Check 4: Suspicious <a> tags not detected in footer,
validation goes on"),
                check_5 = str("Check 5: Website impersonation in <title> and
copyright detected, validation stopped, PHISHING SUSPECTED") )
            else:
                print("No impersonation detected")

                if check_identity(url, soup):
                    return render_template("component_dom.html",
                check_1 = str("Check 1: given url is not blacklisted by Google,
validation goes on"),
                check_2 = str("Check 2: <input> tags detected, validation goes on"),
                check_3 = str("Check 3: <a> tags are detected in <body>, validation
goes on"),
                check_4 = str("Check 4: Suspicious <a> tags not detected in footer,
validation goes on"),
                check_5 = str("Check 5: No impersonation in <title> and copyright
detected, validation goes on"),
                check_6 = str("Check 6: Website identity seems Okay") )
                else:
                    return render_template("component_dom.html",
                check_1 = str("Check 1: given url is not blacklisted by Google,
validation goes on"),
                check_2 = str("Check 2: <input> tags detected, validation goes on"),
                check_3 = str("Check 3: <a> tags are detected in <body>, validation
goes on"),
                check_4 = str("Check 4: Suspicious <a> tags not detected in footer,
validation goes on"),
```

```
                    check_5 = str("Check 5: No impersonation in <title> and copyright
detected, validation goes on"),
                    check_6 = str("Check 6: Website identity check failed, more than
half <a> tags point to other domains, PHISHING SUSPECTED") )

                return render_template("component_dom.html",
                    check_1 = str("Check 1: given url is not blacklisted by Google,
validation goes on"),
                    check_2 = str("Check 2: <input> tags detected, validation goes on"),
                    check_3 = str("Check 3: <a> tags are detected in <body>, validation
goes on"),
                    check_4 = str("Check 4: Suspicious <a> tags not detected in footer,
validation goes on"),
                    check_5 = str("Check 5: No impersonation in <title> and copyright
detected, validation goes on") )
            else:
                print("no footer detected")
        else:
            print("no <a> tags detected in <body>")
            return render_template("component_dom.html",
                check_1 = str("Check 1: given url is not blacklisted by Google, validation
goes on"),
                check_2 = str("Check 2: <input> tags detected, validation goes on"),
                check_3 = str("Check 3: <a> tags are not detected, validation stopped,
PHISHING SUSPECTED") )
    else:
        print("no <input> tags detected in <body>")
        return render_template("component_dom.html",
            check_1 = str("Check 1: given url is not blacklisted by Google, validation
goes on"),
```

```
                check_2 = str("Check 2: no <input> tags detected, validation stopped") )
        else:
            return render_template("component_dom.html") # default, no form submission


    @app.route("/component3", methods=["GET", "POST"])
    def component_heuristics():
        if request.method == "POST":
            url = request.form.get("url")
            if not validators.url(str(url)):
                return render_template("component_heuristics.html",
                    input_url = str("URL: " + url),
                    check_1 = str("invalid url, try again") )

            if component_2_check_blacklisted(url): # if match found
                print("Blacklisted url: ", url)
                return render_template("component_dom.html",
                    input_url = str(url),
                    check_1 = str("Check 1: given url is blacklisted by Google, validation stopped, PHISHING DETECTED") )

            response = requests.get(url)
            soup = BeautifulSoup(response.content, 'html.parser')

            heuristic_checks(url, soup) # too much code -> import from outer file
        else:
            return render_template("component_heuristics.html") # default, no form submission
```

```
if __name__ == '__main__':
    app.run()
```

ДОДАТОК В

Файлова структура проекту

C:\Github\Hybrid-system-for-phishing-website-detection\venv>

```
├────Include
├────Lib
│   └────site-packages
│       ├──── [всі використовувані бібліотеки та їх залежності]
├────Scripts
├────static
│   ├────css
│       ├──── [всі використовувані стилі]
│   ├────img
│       ├──── [всі використовувані зображення]
│   └────js
│       ├──── [всі використовувані скрипти клієнтської частини]
├────templates
│       ├──── [всі використовувані HTML спіппети]
├────app.py
├────shutenko_functions.py
├────requirements.txt
├────pyvenv.cfg
└────__pycache__
```