Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
(рівень вищої освіти)

Метод оцінки ризику цифрових сертифікатів за допомогою машинного навчання
(тема)

Виконав: студент 2 курсу, групи СКСм-19-1

_____Власенко Д.Ю.____
(прізвище, ініціали)

Спеціальність 123 Комп'ютерна інженерія

Тип програми ____освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Освітня програма_____

___Спеціалізовані комп'ютерні системи_____
(повна назва освітньої програми)

Керівник роботи _ст.викл Адамов О.С.___
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Автоматизації проектування обчислювальної техніки
Рівень вищої освіти другий (магістерський)
Спеціальність 123 – Комп'ютерна інженерія
Тип програми Освітньо-професійна
Освітня програма Спеціалізовані комп'ютерні системи
(повна назва)

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

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Власенку Денису Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Метод оцінки ризику цифрових сертифікатів за допомогою машинного навчання»

затверджена наказом по університету від 30 10 2020 р. № 1489 Ст.
2. Термін подання студентом роботи до екзаменаційної комісії 22 12 2020 р.
3. Вихідні дані до роботи Anaconda Framework
Jupyter Notebook
Навчальний набір даних
Тесотвий набір даних
Алгоритми машинного навчання

4. Перелік питань, що потрібно опрацювати в роботі
Аналіз предметної області і постановка задачі
Реалізація машинного навчання та тестування класифікатора
Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 29 слайдів

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Видача теми роботи, її узгодження та затвердження.	01.09.2020 – 07.09.2020	
2.	Аналіз проблемної області, постановка задачі, вибір засобів реалізації.	07.09.2020 – 21.09.2020	
3.	Придбання необхідних компонентів.	21.09.2020 – 05.10.2020	
4.	Збірка пристрою.	05.10.2020 – 26.10.2020	
5.	Створення додатка на смартфон.	26.10.2020 – 02.11.2020	
6.	Написання коду для мікроконтролера.	02.11.2020 – 09.11.2020	
7.	Оформлення пояснювальної записки.	09.11.2020 – 14.12.2020	
8.	Перевірка виконаного проекту, допуск до захисту.	14.12.2020 – 21.12.2020	
9.	Захист проекту.	22.12.2020	

Дата видачі завдання 01.09.2020

Студент _____
(підпис)

Керівник роботи _____ д.т.н., проф. Чумаченко С.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить 79 сторінок, 37 рисунків, 2 таблиці, 9 лістингів, 28 джерел за переліком посилань.

ЦИФРОВИЙ СЕРТИФІКАТ, МАШИННЕ НАВЧАННЯ, КЛАСИФІКАТОР , НОРМАЛІЗАЦІЯ ДАНИХ, НАВЧАЛЬНА ВИБІРКА

У атестаційній роботі розглянуті принципи використання цифрових сертифікатів та принципи роботи центрів сертифікацій. Проаналізована проблема фішингу та кібератак з використанням цифрових сертифікатів. Розглянуто можливість впровадження машинного навчання для оцінки ризику цифрових сертифікатів.

Розроблений класифікатор для передбачення належності цифрових сертифікатів до довіреної або недовіреної групи. Створено набір даних для навчання класифікатора та для тестування його роботи. Проведена візуалізація роботи класифікатора.

ABSTRACT

The explanatory note contains 79 pages, 37 figures, 2 spreadsheets, 9 listings, 28 sources of references.

DIGITAL CERTIFICATE, MACHINE LEARNING, CLASSIFIER, DATA NORMALIZATION, TRAINING DATASET.

Principles of digital certificates usage and principles of how working certification authorities are considered in the attestation work. The problem of phishing and cyberattacks with using digital certificates is analyzed. The possibility of introducing machine learning to assess the risk of digital certificates is considered.

A classifier has been developed to predict whether digital certificates belong to trusted or non-trusted group. Training dataset and validation dataset have been created to teach the classifier and test it. Visualization of the classifier operation is performed.

ЗМІСТ

Міністерство освіти і науки України.....1
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ.....6
ВСТУП.....7
1АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 10
1.1Інфраструктура відкритих ключів.....10
1.2Цифрові сертифікати.....12
1.3Протоколи SSL та TLS.....18
1.4Центри сертифікацій.....22
1.5Проблема фішингу 27
1.6Чорні списки.....30
1.7Шахрайські сертифікати.....32
1.8MITM атаки 34
1.9Можливі засоби захисту від атак 39
1.10Машинне навчання 46
2РЕАЛІЗАЦІЯ МАШИННОГО НАВЧАННЯ 49
2.1Підготовка середовища для реалізації завдання.....50
2.2Реалізація навчання.....51
3РЕЗУЛЬТАТИ РОБОТИ КЛАСИФІКАТОРА.....71
ВИСНОВКИ.....75
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ.....76
ДОДАТОК А.....79

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

PKI – Public Key Infrastructure (інфраструктура відкритих ключів)

SSL – Secure Sockets Layer (рівень захищених сокетів);

HTTPS – HyperText Transfer Protocol Secure;

TLS – Transport Layer Security (захист на транспортному рівні);

CA – Certification authority (центр сертифікації);

EV – Extended Validation (розширена перевірка);

OV – Organization Validation (перевірка організації);

DV – Domain Validation (перевірка домену);

VPN – Virtual Private Network (віртуальна приватна мережа)

MITM – Man in the middle (атака посередника);

ARP – Address Resolution Protocol (протокол визначення адрес)

HSTS – HTTP Strict Transport Security (примусове з'єднання);

OBC – origin-bound certificate;

DNS – Domain Name System (система доменних імен);

k-NN – k-nearest neighbor (метод k-найближчих сусідів);

TP – True Positive (істинно-позитивний показник);

TN – True Negative (істинно-негативний показник);

FP – False Positive (помилково-позитивний показник);

FN – False Negative (помилково-негативний показник);

PPV – Positive Predictive Value (позитивне передбачуване значення);

NPP – Negative Predictive Value (негативне передбачуване значення);

ВСТУП

Сучасне суспільство розвивається надзвичайно швидкими темпами. Однією з основних рушійних сил цього процесу є використання комп’ютерів та інтернету в різноманітних сферах діяльності людства. Саме електронно-обчислювальна техніка стала тією матеріальною основою, яка дала такий величезний поштовх науково-технічному прогресу. З кожним днем з’являється все більше нових інформаційних технологій, які доступні

широкому спектру користувачів. Яскравим прикладом цього є інтернет, який спочатку розроблювався для військової галузі, але зараз набув розповсюдження в будь-яких сферах діяльності людини. Дуже важко уявити сучасну компанію, або приватну особу яка не користується інтернетом.

В наші часи інтернет дає надзвичайні можливості у навчанні, роботі, приватному та діловому спілкуванні, сфері розваг. Проте є і зворотна сторона цієї мережі. Якщо раніше злочинці існували тільки у реальному просторі, то з появою інтернету, з'явилося нове поле для злочинних дій у різних формах та проявах. Кіберзлочинців цікавлять паролі, банківські рахунки, персональні дані й інша приватна інформація фізичних осіб, компаній та державного сектору. Розвиток інтернету надав можливість їм здійснювати злочини не тільки на національному рівні, а і на глобальному.

Одним із видів інтернет-шахрайства є фішинг. За допомогою фішингу, злочинець викрадає конфіденційні дані, що стає можливим завдяки необізнаності користувачів інтернету в основах мережевої безпеки. Одним із методів запобігання фішингу є використання цифрових сертифікатів. Вони використовуються в функціях криптографії з відкритим ключем, найчастіше застосовуються для ініціалізації безпечного SSL з'єднання між веб-браузерами та веб-серверами. Хоча й цифрові сертифікати мають деякі недоліки, але на сьогоднішній день вони використовуються майже на всіх веб-сайтах та слугують доказом довіри до сайту. До недоліків можна віднести те, що дуже багато залежить від центру сертифікації, який видає цифровий сертифікат. Всі центри сертифікацій мають однакову степінь довіри за замовчуванням. Допоки не буде доведено, що цифровий сертифікат використовується в шахрайських цілях, веб-браузер не бачитиме різниці між сертифікатом відомого центру та підозрілого. Коли вже буде доведено шахрайство, то браузери зможуть офіційно занести скомпрометовані цифрові сертифікати до своїх чорних списків [1].

Одним із способів виявлення недовірених цифрових сертифікатів є застосування машинного навчання. За допомогою машинного навчання та

навчальної вибірки запропоновано створити класифікатор, який буде виявляти чи є сертифікат довіреним або ж ні використовуючи його характеристики.

Цілі і завдання дослідження:

– розібрати структуру цифрових сертифікатів;

– ознайомитися з центрами сертифікацій та принципами їх роботи;

– розглянути проблему фішингу;

– ознайомитися з шахрайськими цифровими сертифікатами та використанням їх у кібератаках;

– скласти навчальну та тестову вибірки для машинного навчання;

– застосувати машинне навчання та розробити класифікатор для оцінки ризику цифрових сертифікатів;

– проаналізувати роботу класифікатора і продемонструвати результати роботи класифікатора.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Інфраструктура відкритих ключів

Криптографічна система з відкритим ключем може зіграти важливу роль у наданні послуг безпеки, включаючи конфіденційність, автентифікацію та цифрові підписи.

Інфраструктура відкритих ключів (PKI) базується на асиметричній криптографії. У 1976 році два математики, Вітфілд Діффі та Мартін Геллман відкрили асиметричну криптографію. До асиметричної криптографії використовували симетричну криптографію, де той один й той самий ключ використовувався як для шифрування так і для дешифрування. Метою була концепція з двома ключами, де приватний ключ буде використовуватися для шифрування та відкритий ключ для дешифрування. На той час ця концепція була лише теорією.

У 1977 році троє математиків, Рональд Рівест, Аді Шамір та Леонард Адлеман, застосували теорії Діффі та Геллмана і розробили алгоритм шифрування під назвою RSA [2].

Інфраструктура відкритих ключів підтримує розповсюдження, відкликання та перевірку відкритих ключів, що використовуються для шифрування відкритих ключів, і дозволяє зв’язувати особисті дані із сертифікатами відкритих ключів. Також PKI дозволяє користувачам та системам безпечно обмінюватися даними через інтернет та перевіряти легітимність суб’єктів, що володіють сертифікатами. Такими суб’єктами є веб-сервери, інші автентифіковані сервери та особи. Інфраструктура відкритих ключів дозволяє користувачам автентифікувати власників цифрових сертифікатів, а також опосередковувати процес відкликання сертифікатів, використовуючи криптографічні алгоритми для захисту

процесу.

Сертифікати інфраструктури відкритих ключів містять в собі відкритий ключ, що використовується для шифрування та криптографічної автентифікації даних надісланих або отриманих від особи, якій було видано сертифікат. Інша інформація, яка включена до сертифікату PKI, включає ідентифікаційну інформацію про власника сертифікату, про PKI яка видала сертифікат, та інші дані, включаючи дату створення сертифіката та термін дії [3].

Без інфраструктури відкритих ключів конфіденційна інформація все ще може шифруватися, забезпечуючи конфіденційність, та обмінюватися між двома суб’єктами, але не було б жодного гарантії ідентичності іншої сторони. Будь-яка форма конфіденційних даних, якими обмінюються через інтернет, залежить від PKI, що дозволяє використовувати криптографію з відкритим ключем, оскільки інфраструктура відкритих ключів дозволяє автентифікований обмін відкритими ключами.

Типова інфраструктура відкритих ключів включає в собі такі ключові елементи, як:

- політику безпеки;
- центр сертифікації;
- центр реєстрації;
- сховище сертифікатів та система поширення;
- програми з підтримкою PKI.

Політика безпеки визначає основний напрямок організації щодо інформаційної безпеки, а також процеси та принципи використання криптографії. Зазвичай включає твердження про те, як організація буде поводитися з ключами та цінною інформацією, а також встановлюватиме рівень контролю, необхідний для відповідності рівням ризику.

Центр сертифікації слугує довіреною стороною, забезпечує корінь довіри для всіх сертифікатів PKI та надає послуги, які можуть бути використані для автентифікації осіб, комп'ютерів та інших організацій.

Центр реєстрації, який ще часто називають залежним або ж підлеглим центром сертифікації, займається тим, що видає сертифікати інфраструктури відкритих ключів. Такий центр реєстрації сертифікований кореневим центром сертифікації та уповноважений видавати сертифікати для певного використання.

База даних сертифікатів зберігає інформацію про видані сертифікати. На додаток до самого сертифіката, база даних включає період дії та статус кожного сертифіката PKI. Відкликання сертифіката здійснюється шляхом оновлення цієї бази даних, до якої потрібно надіслати запит для автентифікації будь-яких даних, підписаних або зашифрованих за допомогою секретного ключа власника сертифіката.

Для зберігання сертифікатів зазвичай використовують комп'ютер, проте альтернативою також може бути зберігання в пам'яті програм, які не вимагають постійного зберігання сертифікатів, а тільки коли вони будуть потрібні. Такий спосіб надає програмам, що працюють у системі доступ до сертифікатів які там знаходяться, списків анульованих сертифікатів та списків довірених сертифікатів.

1.2 Цифрові сертифікати

Цифровий сертифікат використовується для криптографічного зв’язку права власності на відкритий ключ із суб’єктом, що володіє ним. Сертифікати призначені для спільного використання відкритих ключів, які будуть використовуватися для шифрування та автентифікації. Цифрові сертифікати містять відкритий ключ, що засвідчується, ідентифікуючи інформацію про організацію, що володіє відкритим ключем, метадані, що стосуються цифрового сертифіката, та цифровий підпис відкритого ключа, створеного емітентом сертифіката.

Розповсюдження, автентифікація та анулювання цифрових сертифікатів є основними цілями інфраструктури відкритих ключів (PKI),

системи, за допомогою якої відкриті ключі розповсюджуються та автентифікуються [4].

Цифрові сертифікати використовуються в функціях криптографії з відкритим ключем, найчастіше використовуються для ініціалізації безпечного SSL з'єднання між веб-браузерами та веб-серверами (рис. 1.1).

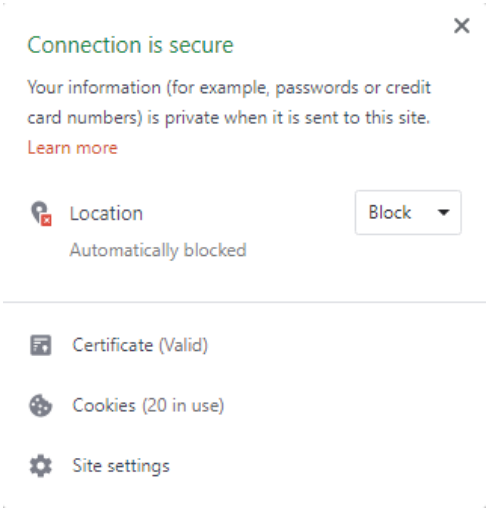

Рисунок 1.1 – Безпечне з'єднання з дійсним сертифікатом

На рисунку 1.2 зображений цифровий сертифікат, який був виданий для пошукової системи Google.

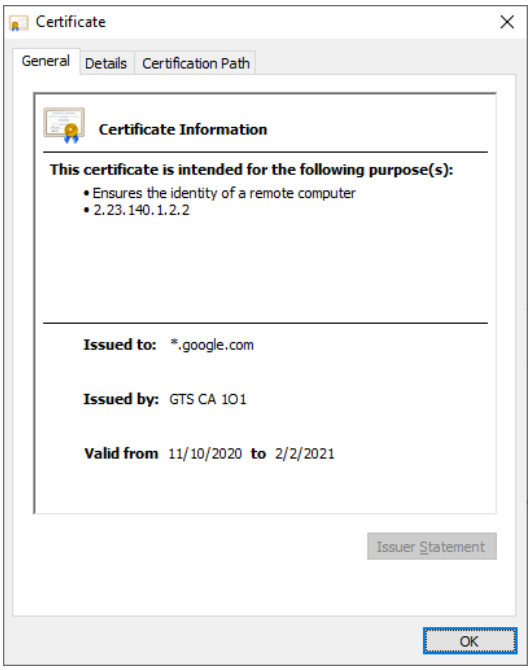

Рисунок 1.2 – Цифровий сертифікат для Google.com

Стандарт сертифікації X.509 широко використовується для структурування цифрових сертифікатів [5]. Існує три версії цього стандарту, а в даний час використовується третя версія цього стандарту (рис. 1.3).

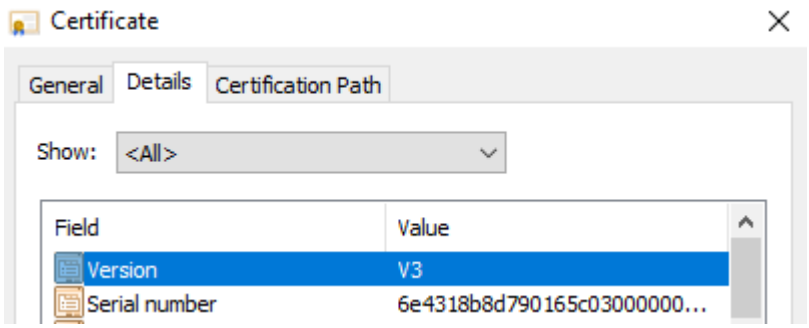

Рисунок 1.3 – Третя версія сертифікату

Цифровий сертифікат містить основні поля (рис. 1.4).

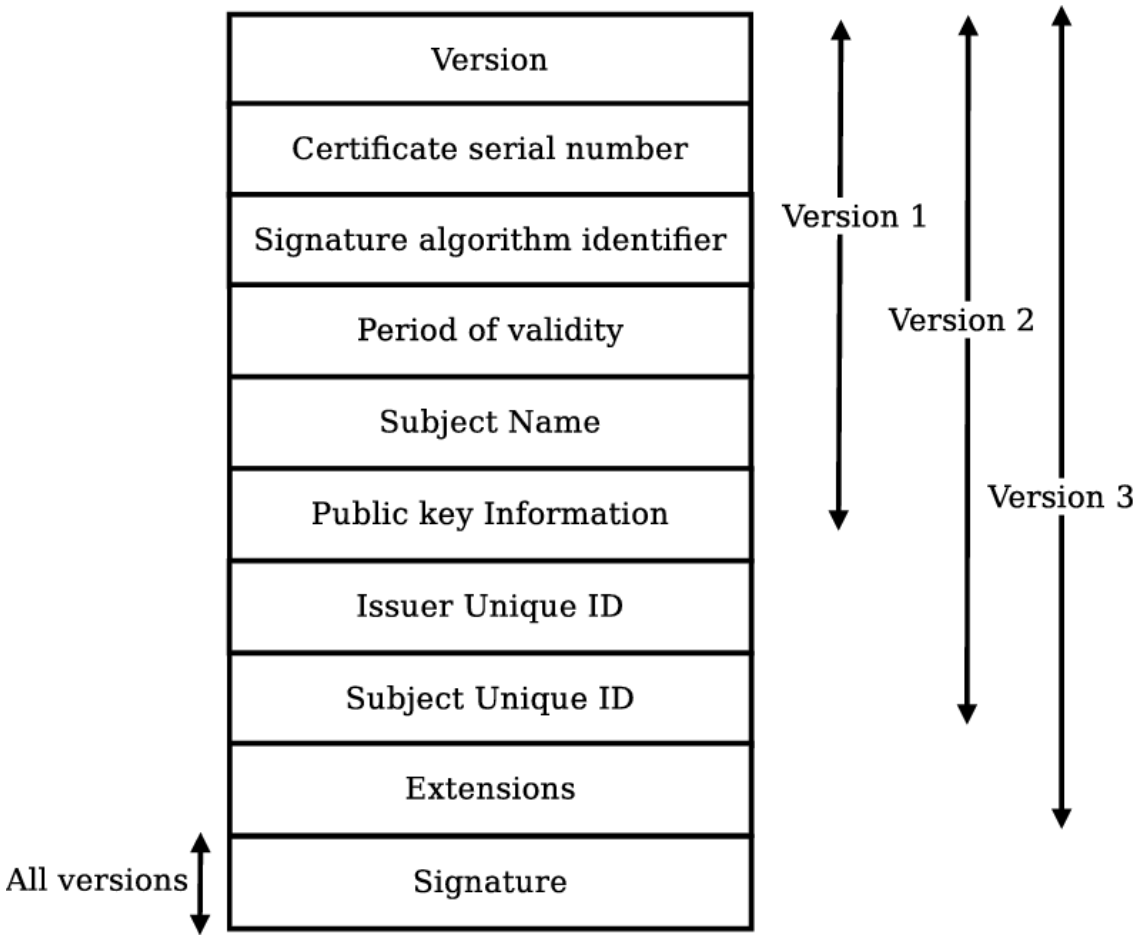

Рисунок 1.4 – Поля цифрового сертифікату

Якщо відкрити цифровий сертифікат, який видний кінцевому домену,

то можна визначити наступні поля:

1) Version;
2) Serial number;
3) Signature algorithm;
4) Issuer;
5) Valid from;
6) Valid to;
7) Subject;
8) Public key;
9) Public key parameters.

Для пояснення кожного з полів та прикладу даних, які в них містяться обраний цифровий сертифікат, який був виданий для “www.google.com”:

1) версія цифрового сертифікату, яка вказує на стандартну структуру сертифікату. Наприклад – V3;

2) серійний номер, який є унікальним та присвоєний кожному цифровому сертифікату. Він призначається емітентом для ідентифікації сертифікату. Такий вигляд має серійний номер для пошукової системи компанії Google – 6e4318b8d790165c0300000000badb8a;

3) в цьому полі вказується тип алгоритму, який використовується емітентом для підпису сертифікату. Наприклад – sha256RSA;

4) хеш алгоритм підпису, який використовується для підпису сертифікату. Метод хешу вибирається з поля, потім механізм ланцюжка сертифікатів декодує підкріплений підпис за допомогою алгоритму підпису, зазначеного в полі типу алгоритму підпису, і повертає собі підписаний хеш. Якщо обидва хеші збігаються, тоді підпис є дійсним, якщо вони відрізняються, підпис вважається недійсним. Такий вигляд має хеш алгоритм для пошукової системи Google – SHA256;

5) це поле, де записана назва організації, яка підписала та видала цифровий сертифікат. Зазвичай такою організацією є центр сертифікації, проте в деяких випадках це може й бути центр реєстрації. Використання

цього сертифіката означає довіру до організації, яка підписала цей сертифікат. У деяких випадках, коли мова йде про кореневий сертифікат центру сертифікації, емітент підписує власний сертифікат, оскільки він є найвищим органом влади. Такий вигляд має поле емітенту для Google – CN = GTS CA 1O1; O = Google Trust Services; C = US. Де CN – центр сертифікації (та назва сертифікату); O – організація; C – країна походження;

6) поле “дійсний з” вказує дату та час, коли сертифікат стає дійсним. Наприклад – Tuesday, November 10, 2020 4:34:43 PM;

7) поле “дійсний до” вказує дату та час по котру цифровий сертифікат буде дійсним, після настання цього часу сертифікат вважається недійсним. Наприклад – Tuesday, February 2, 2021 4:34:42 PM;

8) суб’єкт для якого виданий цифровий сертифікат. Наприклад – CN = *.google.com; O = Google LLC; L = Mountain View; S = California; C = US. Де CN – назва сертифікату; O – організація; L – локація; S – штат; C – країна походження;

9) містить відкритий ключ пари ключів, який пов’язаний із сертифікатом, наприклад (рис. 1.5);

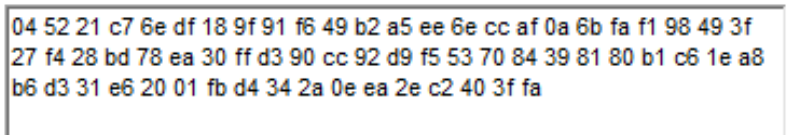

Рисунок 1.5 – Відкритий ключ

10) поле параметрів відкритого ключа дозволяє суб’єктам мати різні параметри від тих центрів сертифікацій, які видають сертифікат. Наприклад – ECDSA_P256;

На рисунку 1.6 зображений доменний сертифікат пошукової системи Google з основними полями.

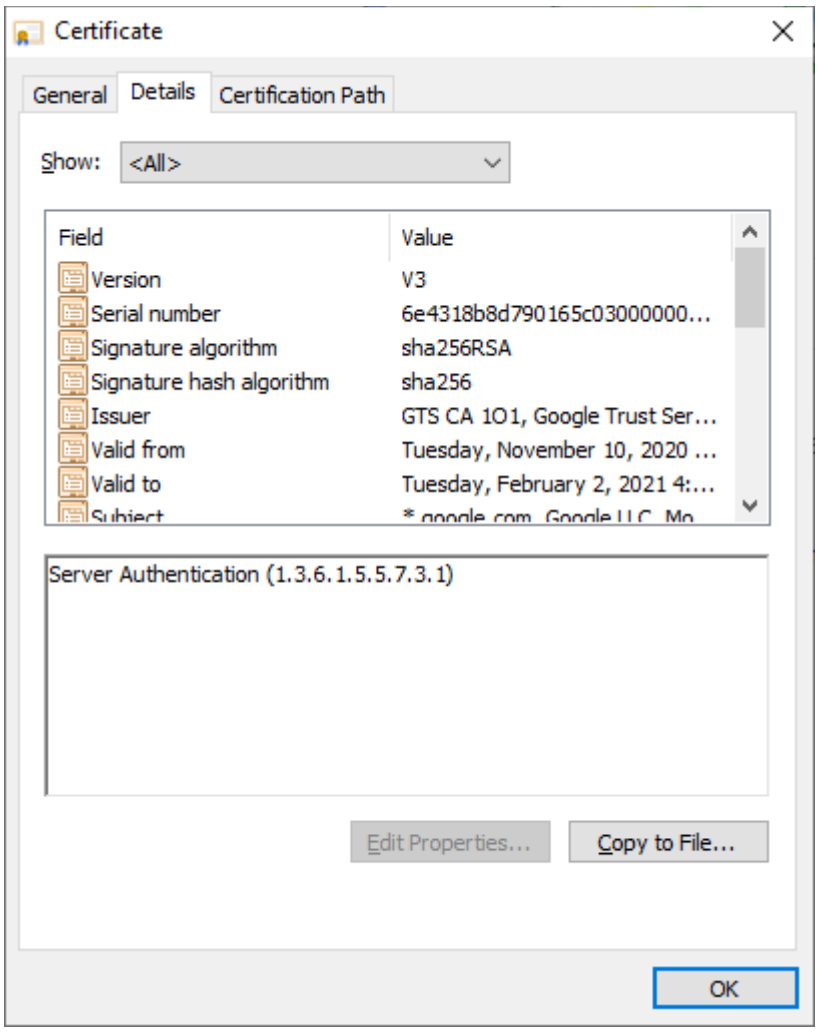

Рисунок 1.6 – Цифровий сертифікат так його основні поля

Окрім основних полів, цифровий сертифікат має ще й додаткові поля, які залежать від того, яким є цифровий сертифікат. Кореневі сертифікати мають одні поля, а доменні інші.

Додаткові поля для доменного сертифікату:

1) Enhanced Key Usage;
2) Subject Key Identifier;
3) Authority Key Identifier;
4) Authority Information Access;
5) Subject Alternative Name;
6) Certificate Policies;
7) CRL Distribution Points;
8) SCT List;

9) Key Usage;
10) Basic Constraints;
11) Thumbprint.

Для порівняння, кореневий сертифікат який є у ланцюжку найвищим, та підписав проміжний, а потім вже цей доменний сертифікат містить інші додаткові поля. Розглянуті додаткові поля кореневого цифрового сертифікату, а саме, Google Trust Services – GlobalSign Root CA-R2:

1) Subject Key Identifier;
2) CRL Distribution Points;
3) Authority Key Identifier;
4) Key Usage;
5) Basic Constraints;
6) Thumbprint;
7) Friendly name;
8) Enhanced key usage;
9) Extended Validation.

1.3 Протоколи SSL та TLS

1.3.1 SSL

SSL – це стандартна технологія для безпечного зв'язку між веб-сервером та браузером або поштовим сервером та поштовим клієнтом. Це безпечне з'єднання захищає всю інформацію, яку ми передаємо між веб-сервером та браузером (користувачем), щоб вона залишалася конфіденційною і недоторканою. SSL є галузевим стандартом і використовується мільйонами веб-сайтів по всьому світу для забезпечення безпеки даних [6].

Перевагою використання цього протоколу є використання його вбудованих функцій безпеки для захисту незахищених протоколів прикладного рівня, таких як HTTP та HTTPS. Виходячи з цього,

криптографічні алгоритми застосовуються до звичайного тексту, який повинен проходити через незахищений канал зв'язку, такий як інтернет, і забезпечують збереження конфіденційності даних по всьому каналу передачі. SSL сертифікат потрібен, для того щоб веб-сайт мав безпечне SSL з'єднання. Для створення безпечного з'єднання на веб-сайтах, рівень безпеки https забезпечується за допомогою системи ідентифікації сервера. Відкритий ключ сервера надсилається браузеру з метою захисту вмісту веб-сайту. Потім він знаходиться під контролем браузеру, для того щоб перевірити те, що сертифікат є валідним. Отже, автентифікація та перевірка розглянутого сертифіката затверджується браузером з метою отримання зворотного зв’язку від використовуваного ключа на сервері веб-сайту.

SSL працює шляхом автентифікації клієнтів та серверів використовуючи цифрові сертифікати, а також шляхом шифрування та дешифрування, відповідно за допомогою певних ключів, які потрібно перевірити в центрі сертифікації. В свою чергу, завдання центру сертифікації полягає в тому, щоб ідентифікувати сторони, відносини, адреси, банківські рахунки, та термін дії сертифікату, а також визначити справжність всього перерахованого. Саме тому, використовуючи SSL, користувач отримує впевненість у тому, що сервер справжній. Для ідентифікації користувача на веб-сайті можна використовувати програмне забезпечення на основі SSL на приймаючій стороні (наприклад Internet Explorer) стандартної техніки шифрування на основі ключів та порівняння відкритих ключів сервера (наприклад IIS). Потім користувач може вводити свою приватну інформацію, таку як номери кредитних карток або паролі, з високим рівнем безпеки та надійності.

Система SSL може використовувати комбінацію симетричного та асиметричного шифрування. Шифрування симетричного ключа швидше, ніж шифрування відкритим ключем, а з іншого боку, шифрування відкритим ключем пропонує більш надійні методи автентифікації. Захищене з'єднання SSL як "SSL Handshake" створюється, коли користувачі намагаються

отримати доступ до захищеного вмісту на веб-сайті. Сам процес та генеровані ключі у міркуванні з безпеки не видно користувачеві. При шифруванні те, що зашифровано за допомогою відкритого ключа, може бути розшифровано лише за допомогою закритого ключа та навпаки.

1.3.2 TLS

TLS – це протокол шифрування, розроблений для забезпечення наскрізної безпеки веб-комунікацій. Інженерна рада інтернету (IETF) встановила TLS як стандартний протокол для запобігання фальсифікаціям та прослуховуванню [7].

Під час користування інтернетом користувачі та веб-програми регулярно стикаються з багатьма можливими проблемами безпеки. Сюди входять автентифікація особи іншої сторони, фальсифікація даних та сторонній моніторинг. TLS використовує криптографічні методи для автентифікації клієнта або сервера у з'єднанні, допомагає забезпечити цілісність переданих даних та забезпечує захист протягом користування браузером

TLS також використовується в таких додатках, як: електронна пошта, додатки для передачі файлів, відео та аудіо конференціях. TLS сумісний зі значною кількістю протоколів, включаючи HTTP, SMTP, FTP, XMPP та багато інших. Слід зауважити, що TLS не призначений для захисту даних у кінцевих системах, а лише даних, які передаються через інтернет.

Безпека TLS розроблена для використання шифрування як з боку клієнта, так і з боку сервера, щоб допомогти забезпечити безпечне з'єднання між двома або більше комунікаційними програмами, гарантувати взаємодію між пристроями та працювати з відповідною ефективністю. Клієнт-серверний зв'язок починається із зазначення, чи буде обмін даними здійснюватися з протоколами TLS, або ж без них. Клієнт може вказати з'єднання TLS різними способами. Наприклад, клієнт може використовувати номер порту, який підтримує типи шифрування, що використовуються у

зв'язку TLS. Іншим потенційним методом є подання запиту, що стосується протоколу, для переключення на з'єднання TLS. Після того, як клієнт і сервер домовились про взаємодію за допомогою TLS, специфікація протоколу проходить два етапи: протокол рукостискання і протокол запису. Протоколи TLS використовують комбінацію симетричної та асиметричної криптографії. Симетрична криптографія створює ключі, відомі як відправнику, так і одержувачу, тоді як асиметрична криптографія генерує пари ключів - один загальнодоступний (спільний між відправником і одержувачем) і один приватний. Специфікації, необхідні для обміну повідомленнями програми, встановлені в протоколі TLS Handshake. Рукостискання TLS передбачає серію обмінів між клієнтом та сервером, які варіюються залежно від використовуваного алгоритму обміну ключами та підтримуваних наборів шифрів [8].

Як тільки метод дешифрування встановлений під час процедури рукостискання, протокол запису TLS використовує симетричну криптографію для генерації унікальних ключів сеансу для кожного з'єднання, що дозволяє продовжувати зв'язок протягом сеансу. Протокол запису також додає будь-які дані, що надсилаються, за допомогою коду автентифікації повідомлень на основі хешу. Оскільки протоколи шифрування в TLS є складними, користувачі повинні розраховувати витратити деяку потужність обчислення на процес [9]. Але TLS також має внутрішні методи для запобігання значним відставанням. Як результат, протоколи TLS не повинні помітно впливати на продуктивність веб-додатків та час завантаження.

Більшість браузерів сьогодні підтримують TLS за замовчуванням. Наприклад, Google Chrome активно застерігає користувачів від веб-сайтів, що не належать до HTTPS. У свою чергу, користувачі також стають розумнішими щодо безпеки веб-сайтів та перевірки безпечних протоколів передачі даних. Наполягаючи на обов'язковому використанні TLS у всіх веб-комунікаціях, організації та приватні особи можуть допомогти забезпечити загальний базовий рівень захисту для веб-діяльності.

1.4 Центри сертифікацій

Центр сертифікації – це організація, що видає цифрові сертифікати, які є файлами даних, що використовуються для криптографічного зв'язку об'єкта з відкрити ключем. Центри сертифікації є важливою частиною інфраструктури відкритих ключів (PKI), тому що вони видають SSL сертифікати, які веб-браузери використовують для автентифікації вмісту, надісланого з веб-серверів [10]. Мета центра сертифікації полягає в тому, щоб в кінцевому результаті зробити інтернет більш безпечним місцем для різних організацій та користувачів. Це означає, що центри сертифікацій відіграють ключову роль у цифровій безпеці.

Центри сертифікацій в якомусь сенсі схожі на орган, який видає паспорт людині. Громадянину потрібен паспорт, щоб його могли ідентифікувати, як справжню людину (переконатись, що це саме та людина за яку вона себе видає). Так і працюють центри сертифікації, вони видають сертифікати веб-сайтам. Сертифікат дозволяє організаціям, або власникам веб-сайтів довести користувачам, що вони є справжніми. Центри сертифікацій беруть плату за свою роботу, а саме за перевірку ресурсу і його власника та видачу цифрового сертифіката.

Існують нормативні стандарти, які сформовані “CA/Browser Forum”. Цей форум є галузевою групою, що регулює роботу сертифікаційних центрів та веб-браузерів. Виходячи із назви, можна зрозуміти, що більшістю членів групи є саме центри сертифікацій та постачальники веб-браузерів, але й організації-споживачі також задіяні.

Комерційні центри сертифікацій є невід'ємною частиною інфраструктури відкритих ключів та займаються тим, що:

– перевіряють доменні імена, приватних осіб та організації для того, щоб підтвердити їх справжність за допомогою офіційних записів;

– видають цифрові сертифікати для автентифікації серверів, приватних осіб та організацій (встановлюють довіру);

– ведуть списки відмінених сертифікатів, які вказують, коли сертифікат став недійсним, до закінчення терміну його дії.

Такі центри випускають три рівня SSL сертифікату, які є відповідними різним рівням довіри до цих сертифікатів [11]. Сертифікати з вищим рівнем довіри зазвичай коштують дорожче, оскільки вони потребують більше роботи з боку центра сертифікації. Три різних рівня довірених сертифікатів містять:

– сертифікати перевірки домену (DV) вимагають лише того, щоб заявник право власності на домен, для якого запитується сертифікат. DV сертифікати можна отримати дуже швидко і за низькою вартістю, або навіть безкоштовно. Наприклад, існує сервіс Let's Encrypt, за допомогою якого можна безкоштовно отримати SSL-сертифікати;

– сертифікати підтверджені організацією (OV) забезпечують наступний рівень гарантії. Центри сертифікацій зазвичай проводять певний рівень перевірки заявників, що може включати перевірку по телефону, а також використання третіх осіб для підтвердження інформації, наданої заявником. Сертифікати OV можуть бути видані, якщо заявник може довести, що він має адміністративний контроль над доменним іменем, для якого запитується сертифікат, та довести, що його організація легітимна;

– сертифікати розширеної перевірки (EV) забезпечують найвищий рівень гарантії в тому, що центр сертифікації перевірив організацію, яка покупає сертифікат. “CA/Browser Forum” визначає детальні вимоги до процесу, який повинні застосовувати органи сертифікацій під час перевірки інформації, яка надана заявником для отримання сертифіката розширеної перевірки. Наприклад, індивідуальне прохання EV сертифікату, повинне бути підтверджене шляхом “face-to-face” взаємодії із прохачем сертифікату, а також шляхом перегляду персональної заяви. Потрібен один основний документ ідентифікації, такий як паспорт, посвідчення водія, а також дві

вторинні форми ідентифікації.

Кожен цифровий сертифікат має цифровій підпис, який центр сертифікації видає сертифікату. Цей підпис слугує доказом того, що сертифікат був виданий довіреним центром сертифікації, та показує що сертифікат не був модифікований або замінений.

Ланцюг довіри складається з кількох сертифікатів, що посилаються на центр сертифікації, який видав сертифікат, та слугує ієрархічною моделлю довіри. Модель довіри, яку використовують усі публічні центри сертифікацій, складається з: кореневих сертифікатів, проміжних сертифікатів, сертифікатів сервера.

Діяльність центру сертифікації починається з кореневого сертифіката, який використовується, як основний базис довіри до всіх сертифікатів, виданим центром сертифікації (рис. 1.7).

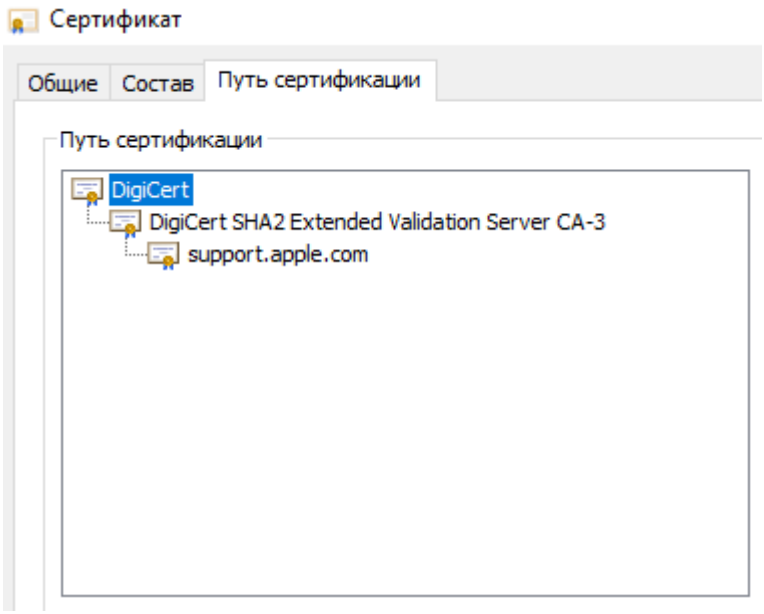

Рисунок 1.7 – Кореневий сертифікат

Кореневий сертифікат, разом з закритим ключем, який пов'язаний із цим сертифікатом, зазвичай обробляється з найвищим рівнем захисту, та зберігається в автономному режимі в захищеному стані і навіть може зберігатися на пристрої, який не працює, окрім випадків, коли потрібен сертифікат. Центр сертифікації використовуватиме цей кореневий сертифікат

для створення проміжних сертифікатів, які використовуються для підписання цифрових сертифікатів, виданих центром сертифікації. Це дозволяє довіряти виданим сертифікатам, одночасно захищаючи корінь у тих випадках, коли проміжний сертифікат закінчується або анулюється. Таким чином, якщо зловмисник скомпрометує проміжний ключ центра сертифікації, то недійсними стають лише ті сертифікати, які вони підписали [12].

Проміжні сертифікати можуть також використовуватися для видачі цифрових сертифікатів, через реєстраційні органи, суб'єкти, яким центр сертифікації може делегувати різні вимоги щодо автентифікації організації та справжності домену для суб'єкта, який прохає сертифікат. Відповідно до правил “CA / Browser Forum”, орган сертифікації за контрактом повинен вимагати від органу реєстрації дотримання та підтвердження їх відповідності правилам форуму.

Окрім проміжного сертифікату існує листовий сертифікат (leaf certificate). Цей сертифікат видається для домену прохача, він завантажується на сервер, який перевіряє домен, субдомен, тощо (залежно від сертифікату). Ці публічні сертифікати мають обмежений термін життя в 398 днів.

Існує список відмінених (анульованих) сертифікатів, що по суті є чорним списком для сертифікатів, яким більше не можна довіряти. Центр сертифікації додає сертифікат до чорного списку, щоб повідомити, що з певним сертифікатом щось не так і він більше не заслуговує на довіру. Це список, який клієнти можуть використовувати для перевірки, або сервери веб-сайтів можуть автоматично перевіряти та надавати інформацію щодо сертифікату.

Інколи центри сертифікацій поділяють за країнам або регіонам, проте, насправді лише невелика кількість найкращих організацій видають більшість цифрових сертифікатів, які використовуються в інтернеті. Центри сертифікацій можна класифікувати на декілька типів, а саме, на публічно довірені центри та на приватні центри. Довірений центр сертифікації – це комерційний центр, являє собою сторонню організацію, яка видає цифрові

сертифікати різним організаціям. Такі центри не контролюються особою чи організацією, яка прагне отримати цифровий сертифікат. Довіренні центри сертифікацій видають публічно довірені цифрові сертифікати, які відповідають нормативним стандартам “CA / Browser Forum” [13]. Провідними комерційними центрами сертифікацій є: DigiCert, Entrust Datacard, Globalsign, GoDaddy, Let’s Encrypt, Sectigo (рис. 1.8).

Рисунок 1.8 – Комерційний центр сертифікації GoDaddy

Якщо мова йде про приватні центри сертифікацій, то це внутрішній центр сертифікації, який існує в межах організації яка більше (як правило, це крупні компанії), та видає власні сертифікати. Найяскравішою відмінністю від публічно довірених центрів є те, що сертифікатам приватних центрів довіряють лише його внутрішні користувачі, клієнти та IT-системи. Приватні центри сертифікацій видають сертифікати, які обмежують доступ до вибраної групи користувачів. Оскільки ці сертифікати видаються внутрішнім центром сертифікації, а не надійним стороннім центром, то вони використовуються в внутрішньо-корпоративних мережах. Найпоширенішими способами використання є: віртуальні приватні мережі (VPN), сертифікати підпису електронної пошти, програми обміну файлами, служби закритих груп користувачів.

Варто зауважити, що окрім видачі цифрових сертифікатів для веб-

сайтів, центри сертифікацій займаються наданням інших типів цифрових сертифікатів, таких як:

– сертифікати підписання коду, що використовуються видавцями програмного забезпечення та розробниками для підписання своїх дистрибутивів програмного забезпечення. Потім кінцеві користувачі можуть використовувати їх для автентифікації та перевірки завантажень програмного забезпечення від постачальника або розробника;

– сертифікати електронної пошти, що дозволяють суб’єктам підписувати, шифрувати та автентифікувати електронну пошту, використовуючи протокол S / MIME (Secure Multipurpose Internet Mail Extension) для безпечних вкладень в електронній пошті;

– сертифікати пристрою, які можуть видаватися пристроям інтернету речей, щоб забезпечити безпечне адміністрування та автентифікацію програмного забезпечення чи оновлення вбудованого програмного забезпечення в апаратний пристрій;

– сертифікати об'єктів, які можуть використовуватися для підписання та автентифікації будь-якого типу програмного об'єкта;

– сертифікати користувача або клієнту, що використовуються приватними особами для різних цілей автентифікації, іноді їх називають сертифікатами перевірки підпису.

В останні роки, різні центри сертифікацій все частіше починають надавати більш широкий спектр послуг своїм користувачам, виходячи за рамки видачі SSL сертифікатів для веб-доменів.

1.5 Проблема фішингу

Фішинг – це такий вид шахрайства, який направлений на виманювання персональних даних. Навіть зараз фішинг залишається серйозною проблемою, яка несе в собі загрозу щодо безпеки та конфіденційності користувачів інтернету. Спроби запобігти фішингу тривають з 1995 року й

продовжуються по цей день. Фішингові атаки, як правило, передбачають зловмисника, який маскується під “справжню” мережу, щоб викрасти конфіденційну інформацію у користувачів, які нічого і не підозрюють.

Ефективність фішингу залежить від здатності зловмисника заплутати свою жертву. В цьому виді шахрайства соціальна інженерія має дуже великий вплив, чим майстерніше працює шахрай, тим більше шансів на те, щоб заманити жертву на фішинговий веб-сайт. Ефективність фішингу залежить від здатності злочинця заплутати жертву. Фішингові веб-сайти найчастіше відображають вміст, який є ідентичним до оригінального сайту, копіюють макет та відтворюють зовнішній вигляд (рис. 1.9).

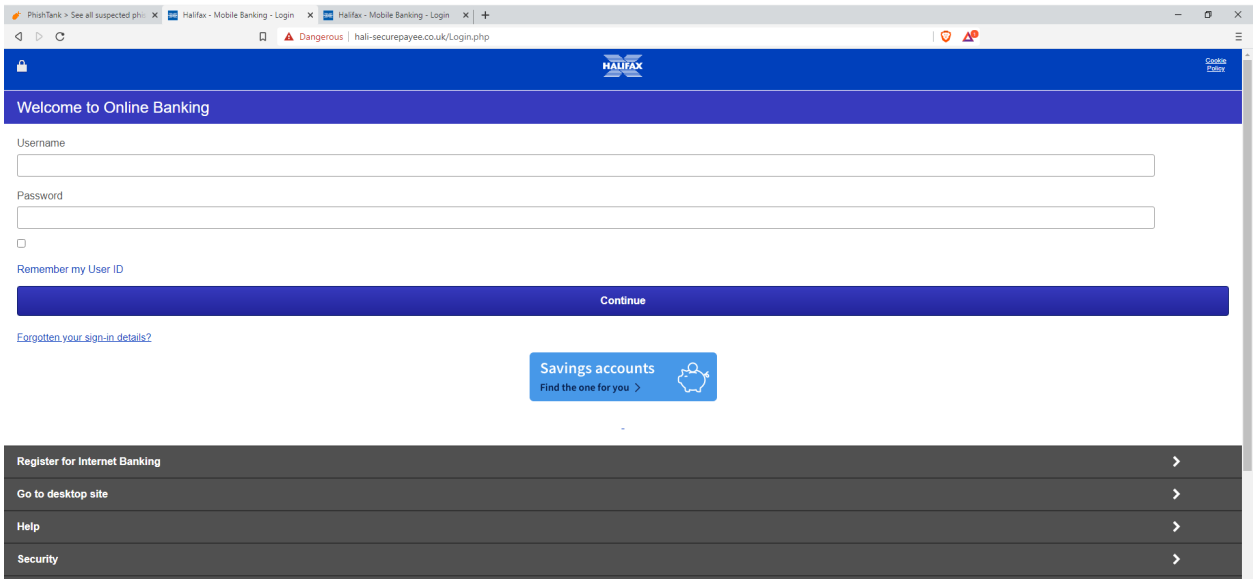

Рисунок 1.9 – Фішинговий сайт онлайн банкінгу

Візуально, майстерно підроблений сайт неможливо відрізнити від справжнього. Інколи злочинці не заморочуються і не створюють ідеальну копію сайту, а сподіваються на те що, користувач і так не помітить різниці, хоча успіх при такому підході значно менший. Фішингові атаки часто починаються з шахрайських електронних листів, які направлені користувачам, задля того, щоб заманити цих людей на шахрайський веб-сайт. Жертва натискає на гіперпосилання, яке міститься в письмі й переходить на фішинговий веб-сайт, де потім добровільно розголошує свої приватні дані,

(наприклад: паролі, дані кредитної картки), або завантажують шкідливе програмне забезпечення [14]. Способів потрапити на такий сайт дуже багато, не тільки через електронний лист. Користувач може перейти по гіперпосиланню на якомусь форумі, у мессенджері, або просто через пошукову систему.

Використовуючи протоколи SSL та TLS під час кожної важливої транзакції в інтернеті допоможуть знешкодити багато поточних спроб фішингу. В ідеалі, перед тим, як видати сертифікат, центр сертифікації повинен провести двоетапну перевірку, щодо права власності над доменом та легітимність власника веб-сайту. Веб-браузери або додатки, які не належать до браузеру (наприклад: додатки для онлайн-банкінгу) повинні перевірити видані сертифікати та повідомити результати кінцевим користувачам у відповідній формі. Отримавши результати (наприклад, через іконку замка, який відображається поруч із адресною строкою веб-браузера), кінцеві користувачі повинні мати достатню кількість інформації для прийняття обґрунтованого рішення щодо того, чи слід довіряти веб-сайту. В даний час зловмисники є частиною довгострокової тенденції до використання розповсюдженого https протоколу. Велика кількість користувачів коли бачить, що протокол https, а не http, одразу починає довіряти веб-сайту, але це хибне рішення. Як справжні, так і фішингові веб-сайти все частіше використовують криптографічно дійсні сертифікати.

Існує два компоненти для протистояння фішингу. Першим з них є безпосередньо виявлення фішингових сайтів. Другий – це ефективне повідомлення кінцевого користувача, що сайт насправді є фішинговим. Ефективне повідомлення може призвести до того, що користувач передумає вводити свої дані на сайті, або завантажувати шкідливе програмне забезпечення на комп'ютер.

Зловмисники використовують так званий “typo-squatting” (рис. 1.10).

Рисунок 1.10 – Принципи typo-squatting

Сутність полягає в тому, щоб додати, або убрати символи. Оскільки кожен може обрати ще не занятий веб-домен, то зловмисник також може зареєструвати оманливий домен, який він хоче видавати за реальний.

Наприклад, замість “monobank.com.ua” зловмисник може обрати “mono-bank.com.ua”, або замість “store.steampowered.com” обрати “ store.steampowerd.com ”. Якщо макет сайту виглядає так само, як і звичайно, то несконцентрована людина може не помітити цієї різниці. Такий технічно діючий сертифікат може бути виданий без перевірки легітимності власника веб-сайту і в кінцевому результаті представлений кінцевому користувачеві. Після перевірки на стороні клієнта, яка головним чином сфокусована на дійсності самого сертифіката, іконка замка буде відображатися у вікні веб-браузеру.

Сертифікати відкритих ключів слугують потенційно цінним джерелом інформації про власників сайтів. Інформація яка вбудована в сертифікати, набагато корисніша, ніж традиційна перевірка на стороні клієнта. Насправді, через відносно динамічний характер фішингових доменів, деякі основні характеристики їх сертифікатів можуть стати вагомим показником цих веб-сайтів [15].

1.6 Чорні списки

Одним із існуючих рішень, щодо захисту від фішингу є чорні списки. Вони складаються виробниками браузерів та довіреними третіми сторонами (такими як різні компанії, які працюють у сфері кібербезпеки). Чорні списки існують для різних функцій шкідливих веб-сайтів, таких як:

– IP-адрес (наприклад, Spamhaus) (рис. 1.11);

– доменних імен (наприклад, PhishTank);

– сертифікатів (наприклад, SSL).

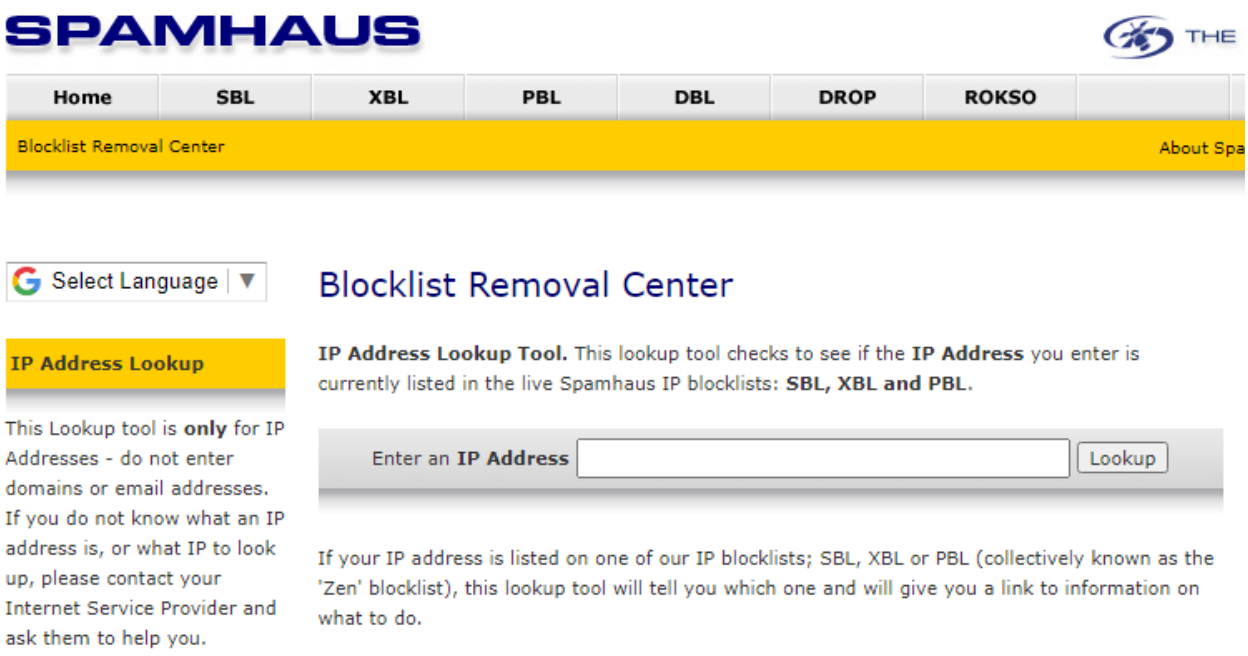

Рисунок 1.11 – Ресурс Spamhaus

Ефективний чорний список повинен одночасно задовольняти трьом вимогам: правильності, повноті та своєчасності [16].

Правильність – це точність чорного списку при розподіленні між шкідливими та нешкідливими веб-сайтами. Правильність безпосередньо впливає на взаємодію з користувачем.

Своєчасність – це затримка між публікацією фішингового сайту та часом коли сайт був доданий до чорного списку. Затримка включає ідентифікацію сайту, звіт сайту, перевірку природи сайту, а потім оновлення на стороні клієнта. Своєчасність особливо важка, коли мова йде про веб-

сайти за спір-фішингом. Спір-фішинг відрізняється від звичайного фішингу, тим що жертва зловмисника не випадкова, а конкретна та добре вивчена ціль. Оскільки спір-фішинг веб-сайти зазвичай націлені на дуже невелику частину користувачів, централізовані чорні списки можуть не отримати звіт після успішного нападу.

Повнота – це величина, яка відображає стан фішингу в чорному списку. Це функція своєчасності та правильності як одне ціле. Головною проблемою є те, що потрібно ідентифікувати фішинговий сайт, додати його до чорного списку. У відповідь на додавання сайтів до чорного списку, злочинці можуть змінювати свої URL-адреси.

1.7 Шахрайські сертифікати

Шахрайські сертифікати – це дійсні сертифікати, видані законним центром сертифікації, які, тим не менш, не заслуговують на довіру, але все ще виглядають надійними для веб-браузерів та користувачів. В умовах інфраструктури відкритих ключів існує вікно вразливості між часом видачі “шахрайського” сертифіката та часом його виявлення. Саме в це вікно такий сертифікат для користувачів вважається дійсним і на перший погляд не містить загрози, цей період триває впродовж кількох тижнів до виявлення і подальшого скасування цього сертифікату.

Надійність відкритих ключів залежить від надійності органу сертифікації, який видає цей ключ. Оскільки кореневі сертифікати доставляються кінцевим користувачам в рамках встановлення веб-браузера або операційної системи, більшість веб-браузерів роблять наступні два неявні припущення [17].

Перше припущення полягає в тому, що не існує внутрішніх обмежень в юридичному плані для організацій, які видають цифрові сертифікати. Будь-який центр сертифікації може видавати сертифікати відкритих ключів для веб-сайтів. Хоча в деяких браузерах і може існувати білий або чорний

список, але більшість надійних центрів сертифікації не підвергаються таким обмеженням.

По-друге, всі центри сертифікації мають однакову степінь довіри. До тих пір, поки ланцюжок довіри не містить “помилок” для будь-якого довіреного кореня, веб-браузер ігнорує різницю між сертифікатом від великого і відомого центру сертифікації (наприклад, VeriSign), та сертифікатом якогось менш відомого центру. Всі центри сертифікації розглядаються однаково за замовчуванням. Кінцеві користувачі та адміністратори системи також можуть управляти надійними коренями через операційну систему (наприклад, Microsoft) або в налаштуваннях веб-браузера (наприклад, Mozilla).

Ці припущення дозволяють веб-сайтам переключатися між надійними центрами сертифікації без будь-яких збоїв у роботі. Поки сертифікаційні центри старого та нового сертифікатів є довіреними, повідомлення до користувача не надходитимуть, і він навряд чи помітить таку зміну в ЦС.

Ці припущення призводять до значних ризиків безпеки. Через різницю в масштабах організації, законах, оперативна практика ЦС може суттєво відрізнятися. Центри сертифікації також стикаються з різними ризиками, які можуть містити загрозу. З огляду на ці припущення, один надійний, але скомпрометований ЦС може вплинути на цілу екосистему PKI. Зокрема, маючи “шахрайський” сертифікат, зловмисник може мати змогу здійснити MITM атаку між кінцевим користувачем та фальшивим веб-сайтом. Оскільки шахрайський сертифікат є технічно дійсним та надійним, для такої атаки не створюватиметься автоматичне виявлення та попередження щодо небезпеки. На жаль, зважаючи на велику кількість веб-сайтів та можливість націлювання на конкретну та обмежену групу жертв, може бути дуже складно виявити шахрайські сертифікати при їх першому використанні, оскільки для деяких фішингових атак потрібен лише одна успішна спроба.

1.8 MITM атаки

MITM-атака, відома як атака посередника або атака “людина посередині” – це вид кібератаки, при якій зловмисник таємно видозмінює зв'язок між двома сторонами, які вважають, що вони безпосередньо спілкуються між собою.

Можна розглянути типову атаку посередника на SSL, суть полягає перехопленні активної мережі, де зловмисник вставляє себе в канал зв'язку між клієнтом жертви та сервером, як правило, з метою прослуховування або маніпулювання приватними даними. На рисунку 1.12 зображено атаку посередника на SSL з підробленим сертифікатом, який встановлений між веб-браузером та HTTPS сервером.

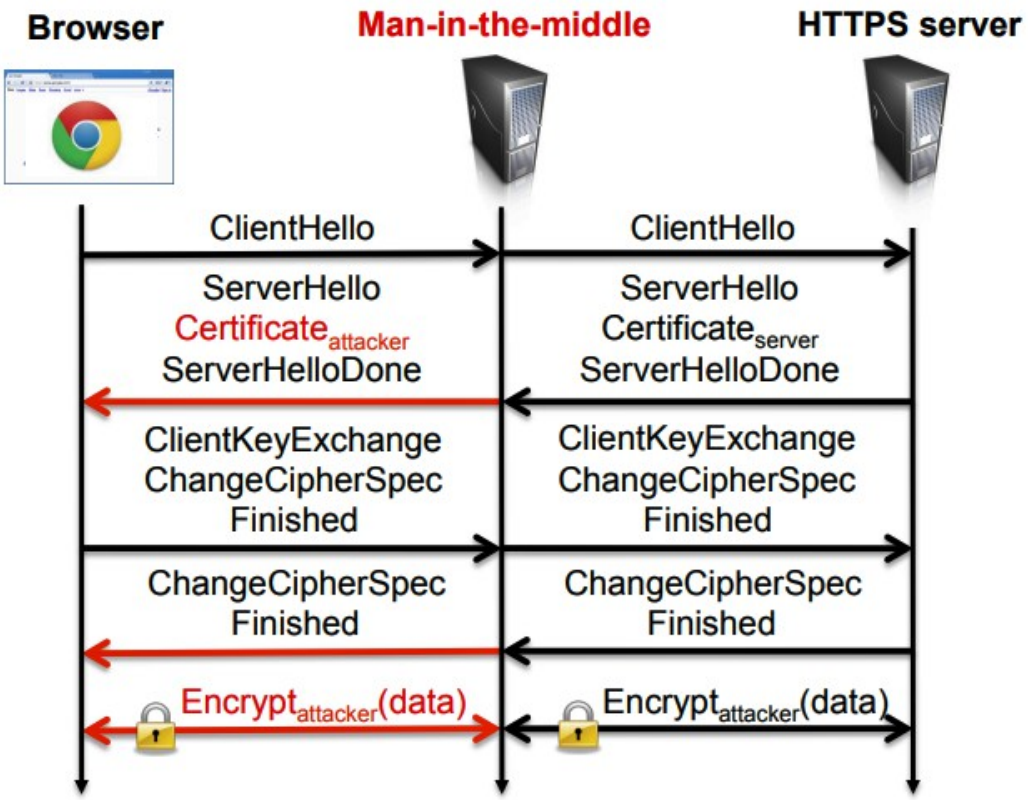

Рисунок 1.12 – Атаку посередника на SSL з підробленим сертифікатом

Зловмисник встановлює два окремих SSL з'єднання з клієнтом та

сервером і передає повідомлення між ними таким чином, що ні клієнт, ні сервер не знають про його присутність. Ця установка дозволяє зловмиснику записувати всі повідомлення і навіть вибірково змінювати передані дані.

В даному прикладі розглянуто атаку на SSL через HTTPS, оскільки це найпоширеніша модель реалізації протоколу SSL і використовується практично у всіх системах додатків для онлайн-банкінгу, службах електронної пошти для забезпечення шифрування каналу зв'язку. Ця технологія покликана забезпечити збереження даних від перехоплення третіми сторонами за допомогою простого аналізатора пакетів [18].

Базові принципи атаки на SSL:

1) спочатку зловмисник проникає в транспортний шлях між клієнтом та сервером, наприклад, налаштувавши шкідливу точку доступу Wi-Fi. Навіть в інших довірених мережах, атакуючий локальну мережу може перенаправляти весь трафік клієнта до себе, використовуючи такі експлойти, як отруєння ARP, підробка DNS, викрадення BGP тощо. Зловмисник також може налаштувати себе як проксі-сервер клієнта, використовуючи протоколи автоконфігурації (PAC / WPAD). На даний момент зловмисник отримав контроль над трафіком клієнта і діє як сервер ретрансляції між клієнтом і сервером;

2) коли атакуючий виявляє SSL повідомлення “ClientHello” клієнта, яке надсилається від клієнта, зловмисник точно визначає, що клієнт ініціює SSL з'єднання. Атакуючий розпочинає видавати за себе сервер своєї жертви та встановлює SSL-зв’язок з клієнтом. Під час “рукостискання” з клієнтом, зловмисник використовує підроблений SSL сертифікат;

3) паралельно попередньому кроку, атакуючий створює окреме SSL з'єднання із легітимним сервером, видаючи себе за клієнта. Після встановлення обох SSL з'єднань зловмисник передає всі зашифровані повідомлення між ними (розшифровує повідомлення від клієнта, а потім повторно шифрує перед відправкою на сервер). Тепер зловмисник може

читати і навіть модифікувати зашифровані повідомлення між клієнтом та сервером.

Як тільки клієнт прийме підроблений SSL сертифікат, то його дані будуть зашифровані за допомогою відкритого ключа зловмисника, який він зможе розшифрувати. Важливо зауважити, що незалежно від того що підроблений сертифікат міг видати надійний центр сертифікації, кроки атаки будуть такими самими. Якщо один із довірених центрів сертифікації став шахрайським, або був змушений іншим чином видати сертифікат для зловмисника, браузер автоматично прийме підроблений сертифікат. Бували випадки, коли зловмисники компрометували центри сертифікацій, щоб отримати дійсні сертифікати. Більше того, навіть якщо зловмисник не має надійного сертифіката то дуже багато користувачів просто ігнорують попередження SSL-сертифіката, представлені браузером. Навіть гірше, деякі небраузерні програмні засоби та мобільні додатки можуть містити несправний код перевірки SSL-сертифіката, який мовчки приймає недійсні сертифікати. Також важливо зауважити, що існує велика кількість автоматизованих інструментів, які можуть відтворювати атаку посередника на SSL. Ці інструменти знаходяться у загальному доступу в інтернеті (наприклад, sslsniff) і значно знижують рівень потрібної технічної досвідченості від зловмисника, який необхідний для здійснення таких атак.

Розглянемо процес комунікації по HTTPS на прикладі підключення користувача до облікового запису на Google [19]. Цей процес включає в себе кілька окремих операцій:

1) клієнтський браузер звертається до http://mail.google.com на порт 80 за допомогою HTTP;

2) сервер перенаправляє клієнтську HTTPS-версію цього сайту, використовуючи HTTP code 302 переспрямування;

3) клієнт підключається до https://mail.google.com на порт 443;

4) сервер пред'являє клієнту свій сертифікат відкритих ключів для перевірки автентичності сайту;

5) клієнт звіряє даний сертифікат зі своїм списком довірених центрів сертифікації;

6) створюється зашифроване з'єднання.

З усіх цих дій найбільш вразливою бачиться операція переспрямування на HTTPS через код відповіді HTTP 302. Для здійснення атаки на точку переходу від незахищеного до захищеного каналу можна використати спеціальний інструмент SSLStrip. З використанням даного інструменту процес атаки виглядає наступним чином:

1) перехоплення трафіку між клієнтом і веб сервером;

2) у момент виявлення HTTPS URL адреси інструмент SSLstrip підміняє його HTTP-посиланням, зіставляючи всі зміни;

3) атакуюча машина надає сертифікати веб-серверу і уособлює клієнта;

4) трафік приймається з захищеного веб-сайту і надається клієнтові.

В результаті чого атакуючий отримує доступ до даних, які клієнт відправляє на сервер. Ними можуть бути паролі від облікових записів, номери банківських карт або будь-яка інша інформація, яка зазвичай передається в прихованому вигляді. Потенційним сигналом проведення даної атаки для клієнта може стати відсутність позначення захищеного HTTPS-трафіку в браузері. Для сервера же така підміна залишиться зовсім непоміченою, тому що немає ніяких змін в SSL-трафіку.

Можливі наступні сценарії атаки, а саме: отруєння кешу ARP, атака на системи з відкритим ключем, впровадження шкідливого коду, Downgrade Attack, публічні засоби комунікації.

У разі системи з відкритим ключем злочинець може перехопити повідомлення обміну відкритими ключами між клієнтом і сервером та змінити їх. Щоб залишитися непоміченим, злочинець має перехопити всі повідомлення між клієнтом і сервером та використовуючи відповідні ключі зашифрувати і розшифровувати їх. Подібні дії можуть здатися занадто

складними для проведення атаки, але вони несуть в собі реальну загрозу для небезпечних мереж, таких як: електронний бізнес, інтернет-банкінг.

Коли мова йде про впровадження коду в MITM атаці, то цей сценарій підходить для захоплення вже авторизованої сесії. Зловмисник може робити так звані ін'єкції коду до веб-сторінок, SQL запитів, а також модифікувати бінарні файли, які завантажив користувач. Це робиться задля того, щоб отримати доступ облікового запису.

Downgrade Attack – це такий вид атаки, коли зловмисник використовую старі версії протоколів, які є більш вразливими, але все ще підтримуються з різних причин. В даному випадку, злочинець може атакувати такі протоколи, як PPTP, SSH, IPsec. Для захисту такого виду атаки, небезпечні протоколи повинні бути відключені, як мінімум на одному боці, підтримки і використання безпечних протоколів за замовчуванням недостатньо.

Атака «Отруєння кешу ARP» полягає в тому, що існує вразливість в протоколі ARP (рис. 1.13).

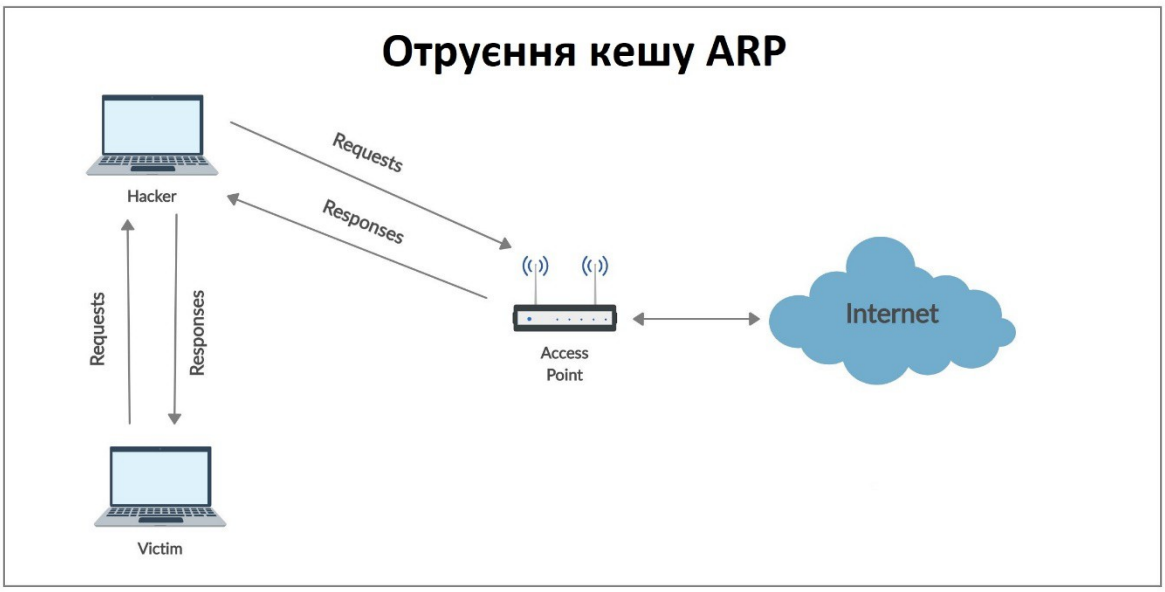

Рисунок 1.13 – Отруєння кешу ARP

На відміну від протоколу DNS, який є можливість налаштувати на прийом тільки захищених динамічних оновлень, пристрої, що використовують ARP, будуть отримувати оновлення в будь-який час. Така

властивість ARP-протоколу дозволяє пристрою відправляти пакет ARP-відповіді на інший вузол, задля вимагання від нього оновлення ARP-кешу. Відправлення ARP-відповіді без генерування будь-яких запитів називається відправкою ARP звернених до себе. У результаті вдало спрямованих ARP-пакетів, які звернені до себе, можуть стати вузли, які вважають, що взаємодіють з одним вузлом, але насправді взаємодіють з вузлом атакуючого.

Публічні засоби комунікацій є найменш використовуваним видом атаки, він може бути задіяним тільки у тому разі, коли зловмисник є власником ресурсу та має повний контроль над інформацією, якою обмінюються жертви. В такому випадку атака може бути проведена в будь-який час, а все що потрібно, щоб користувачі на ментальному ріні ігнорували базові принципи щодо кібербезпеки. Найпоширенішими засобами комунікації є різні сервіси електронної пошти та соціальні мережі.

Можна проаналізувати мережевий трафік, щоб визначити, чи була здійснена атака або ні, та визначити джерело атаки в разі підтвердження, що атака дійсно була. При аналізі дуже вагомими показниками вважаються:

– IP-адреса сервера;
– DNS-ім'я сервера;
– цифровий сертифікат сервера;
– чи є сертифікат власно підписаним;
– чи був сертифікат виданий надійним центром сертифікації;
– чи був сертифікат скасованим;
– чи замінювався сертифікат нещодавно;
– чи отримували користувачі такий же сертифікат в іншому місці.

1.9 Можливі засоби захисту від атак

1.9.1 Примусове захищене з'єднання

Примусове з'єднання (HSTS) – це механізм, який автоматично активує

з'єднання через протокол HTTPS (рис. 1.14).

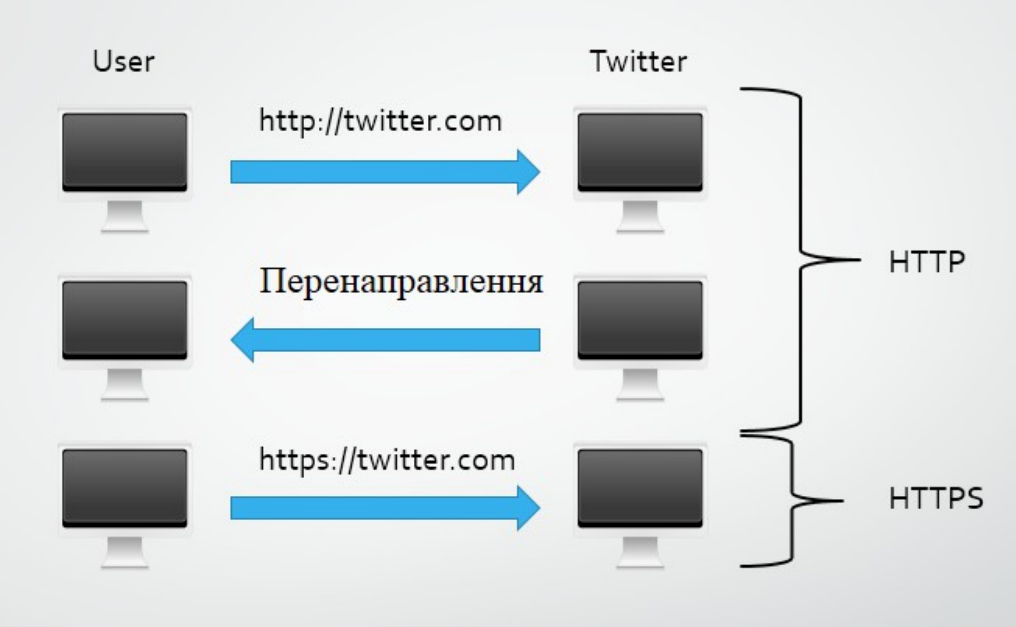

Рисунок 1.14 – Примусове захищене з'єднання

Якщо є наявність примусового з'єднання, то веб-сайт може перешкоджати мережевим зловмисникам робити махінації з SSL. Менш очевидною перевагою безпеки HSTS є те, що браузери просто не спрацьовують, детектуючи недійсні сертифікати, і не дають користувачам можливість ігнорувати помилки SSL.

Ця функція забороняє користувачам приймати ненадійні сертифікати, однак HSTS не призначений для захисту від шкідливого програмного забезпечення або професійних зловмисників, які використовують підроблені сертифікати, які будуть прийняті браузером.

1.9.2 Механізм прив'язки відкритих ключів

Механізм прив'язки відкритих ключів для HTTP дозволяє веб-сайтам визначати власні відкриті ключі за допомогою заголовку HTTP (рис. 1.15), та вказує веб-браузерам відхиляти будь які сертифікати з невідомими відкритими ключами.

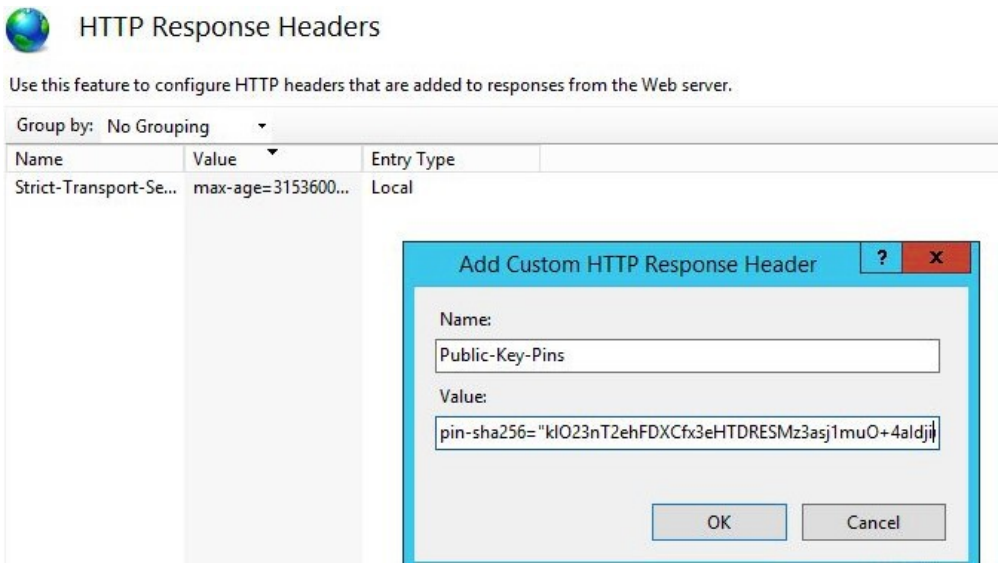

Рисунок 1.15 – Прив'язка відкритого ключа

Механізм прив'язки відкритих ключів забезпечує захист від атаки посередника на SSL, у якій використовують несанкціоновані, але, можливо, надійні сертифікати. Цей механізм автоматично відхиляє шахрайські сертифікати, навіть якщо б клієнт їм довіряв.

Як захист примусового надійного з'єднання, так і механізм прив'язки відкритих ключів вимагає того, щоб користувачі спочатку повинні безпечно відвідати легітимний веб-сайт, перед підключенням із ненадійних мереж. Ця вимога скасовується, якщо прив'язка відкритих ключів попередньо завантажена у браузер, наприклад, у Google Chrome, хоча такий підхід може не мати масштабу для всієї мережі.

Варто відзначити, що попередньо завантажений механізм прив'язки відкритого ключа Google Chrome успішно виявив кілька гучних інцидентів центрів сертифікацій, в яких помилково видані сертифікати використовувались для нападу на домени Google.

Однак у поточних реалізаціях прив'язки відкритого ключа Google Chrome не відхиляє сертифікати які видані місцевим довіреним підписантом, такі як антивіруси, системи корпоративного контролю та шкідливе програмне забезпечення.

Схожим рішенням є TACK, механізм для захисту користувачів від

шахрайства з використанням SSL сертифікатів з мінімальними накладними витратами або інфраструктурою. На відміну від механізму прив'язки відкритих ключів, TACK закріплює ключ підпису, обраний сервером, окремо від приватного ключа, що відповідає сертифікату сервера, та може мати недовгий термін дії. TACK дозволяє веб-сайтам з декількома серверами SSL та декількома відкритими ключами прив'язувати один і той же ключ підписання. Як тільки браузер отримає розширення TACK, йому потрібно буде підписати майбутні підключення до того самого сайту тим же ключем підпису TACK, або в іншому випадку відхилити підключення.

Іншим рішенням слугує DVCert, який надає список прив'язаних сертифікатів за модифікованим протоколом PAKE, намагаючись виявити атаки посередника на SSL, але також вимагає модифікації існуючих клієнтів та серверів.

Концепція прив'язки відкритого ключа (або прив'язки сертифіката) раніше також реалізовувалась як захист на стороні клієнта. Такі додатки для Firefox, як Certificate Patrol та Certlock, були розроблені для попередження користувачів, коли раніше відвіданий веб-сайт починає використовувати інший сертифікат. Однак без явних сигналів сервера може бути важко точно відрізнити реальні атаки від законних змін сертифікатів або альтернативних сертифікатів [20].

1.9.3 Origin-Bound сертифікати

По суті Origin-Bound сертифікат (OBC) – це самопідписаний сертифікат, який браузери використовують для автентифікації клієнта TLS. На відміну від звичайних сертифікатів та їх використання в автентифікації клієнта TLS, такі сертифікати не вимагають жодної взаємодії з користувачем. Ця властивість випливає із спостереження, що оскільки браузер генерує та зберігає лише один сертифікат на клієнтську вимогу, браузеру завжди ясно, який сертифікат він повинен використовувати. Для прийняття рішення не потрібно жодних конфігурацій користувача.

Origin-Bound сертифікат може заблокувати більшість існуючих наборів інструментів для MITM-атак, оскільки зловмисники не можуть видавати себе за клієнта, без викрадення самопідписаного приватного ключа із законного браузера. Однак це не заважає серверу зловмисника постачати користувачу кешований шкідливий JavaScript файл, який згодом виконається на веб-сайті жертви та потенційно видалить дані шляхом повторного підключення до легітимного серверу.

Крім того, OBC вимагає змін коду мережевого стеку на серверах, в той час як примусове захищене з'єднання та механізм прив'язки відкритих ключів цього не роблять, і викликає додаткові обчислювальні витрати на генерацію сертифікатів клієнта. Веб-сайти повинні оцінити, чи є це прийнятним компромісом.

1.9.4 Логи аудиту цифрових сертифікатів

Існує ідея ведення криптографічно-незворотних записів усіх законно виданих цифрових сертифікатів, таким чином щоб невірно видані цифрові сертифікати можна легко виявити, тоді як незаписані сертифікати просто відхилятимуться.

Проект прозорості сертифікатів, ініціатором якого є Google, працює за принципом, що не робить ніяких додаткових перевірок власності доменів, та не запобігає випуску цифрових сертифікатів, а дозволяє будь-кому дізнатися про всі цифрові сертифікати, які були випущені підтверджуючим центром. Якщо такі центри будуть дотримуватись цього стандарту, то тоді стане просто неможливо випустити цифровий сертифікат так, щоб про це не дізнався власник домену.

Інформація про кожен випущений цифровий сертифікат буде записуватись в лог, який в свою чергу буде доступний тільки для додавання запису. Редагування або ж видалення не буде доступним. Сутність полягає в тому, що будь-хто може отримати доступ до цього логу та ознайомитися з інформацією щодо випущеного цифрового сертифікату [21].

На даному етапі вже існує декілька таких логів. Моніторинг логів дозволить відстежити всі випуски всіх цифрових сертифікатів для домену та не дозволить пропустити помилковий.

Для того, щоб лог працював правильно, та буде доступним тільки додавання записів, використовується дерево Меркла, ще відоме як хеш-дерево (рис. 1.16).

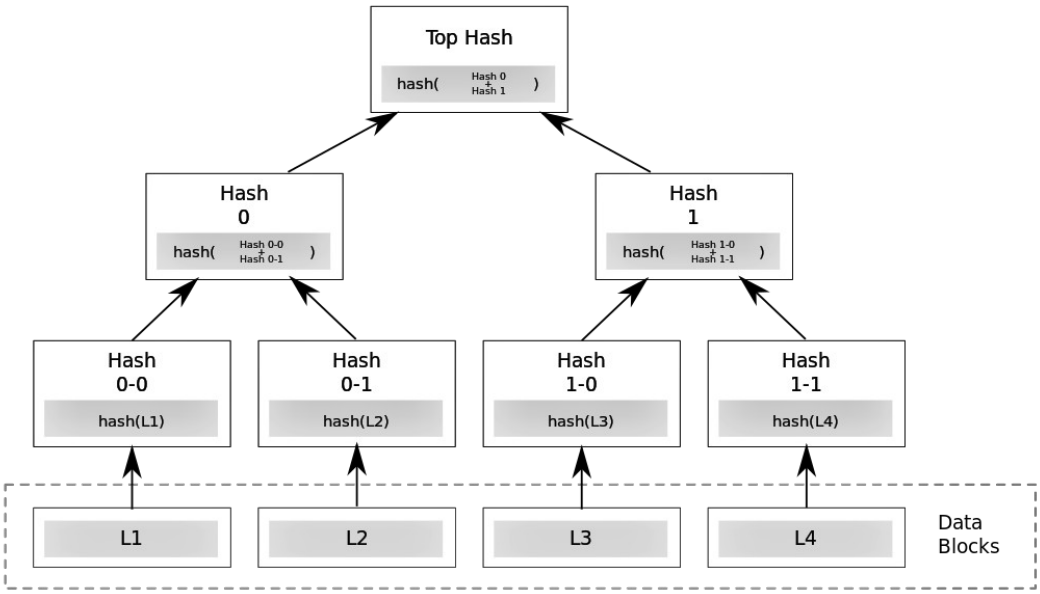

Рисунок 1.16 – Приклад бінарного дерева хешування

Таким чином, це дозволяє перевірити що будь-яка нова версія логу буде включати в себе попередню. За таким же принципом працює і біткойн з системою блокчейну.

Коли сертифікат буде доданий до логу, то повернеться підписаний штамп. Цей штамп часу показує, що цифровий сертифікат буде доданий до логу. Браузер не повинен приймати сертифікат, якщо дійсний штамп часу відсутній.

Також активну участь в процесі приймають спостерігачі та аудитори. Спостерігачі повинні відстежувати записи, які додаються до логу та звіряти новий хеш за своїми розрахунками.

Сутність полягає в тому, щоб знайти ті сертифікати, які були

скомпрометовані, або ж незвичні сертифікати, які потребують окремої уваги. Аудитори займаються тим, що перевіряють криптографічну послідовність та наявність того, що сертифікат додався до логу. Функції аудитора можуть виконувати браузер або ж сторонній сервіс.

Якщо буде така ситуація, що браузери будуть перевіряти логи, та дивитися чи є інформація про сертифікат в них, та в свою чергу не будуть приймати ті цифрові сертифікати, де інформація відсутня, то випустити цифровий сертифікат для чужого домену стане дуже важко. Для того щоб виявити несанкціоновану видачу сертифікату, який відповідає вимогам прозорості сертифікатів, власник домену змушений буде контролювати логи. Таким чином потрібно буде або ж самостійно підтримувати сервер, або користуватися послугами сторонніх служб, які будуть повідомляти власника цього домену, про цифрові сертифікати які були видані.

Принцип цієї системи зображений на рисунку 1.17.

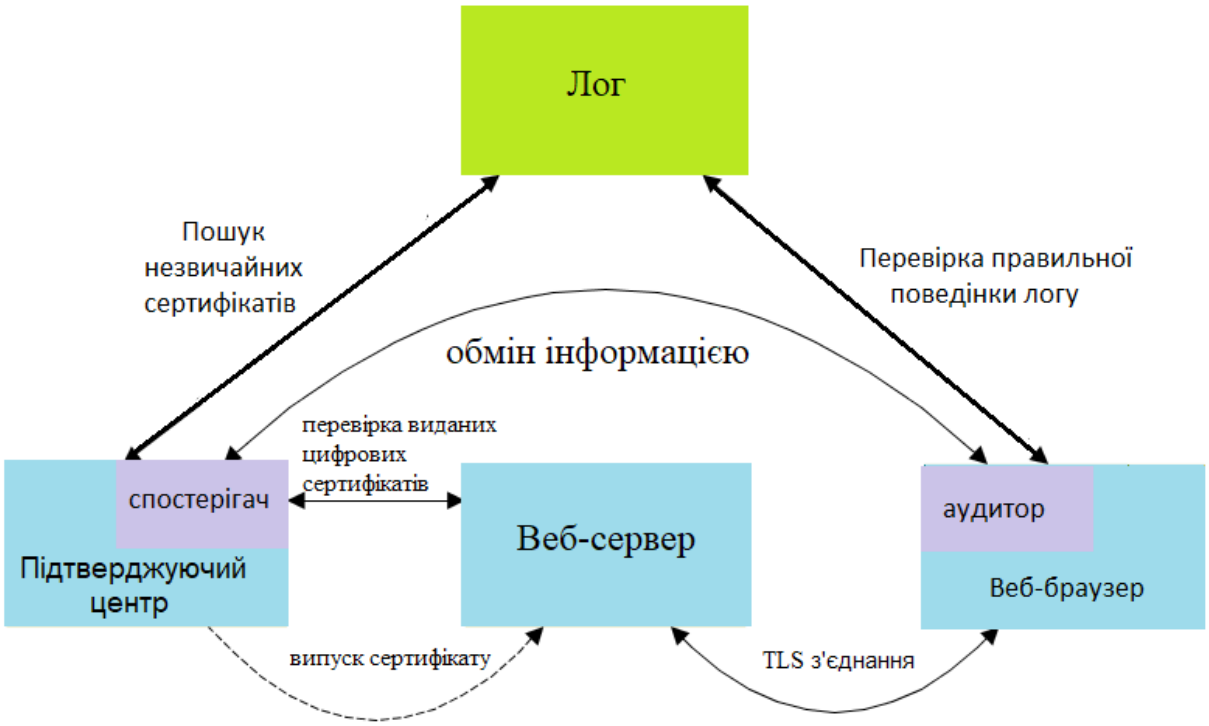

Рисунок 1.17 – Принцип роботи Certificate Transparency

1.9.5 Автентифікація на основі DNS

Автентифікація іменованих об'єктів на основі DNS (DANE) дозволяє

оператору домену підписувати SSL сертифікати для веб-сайтів у своєму домені. Як і в випадку захисту з прив'язкою відкритих ключів, автентифікація іменованих об'єктів на основі DNS може дозволити веб-сайтам вказувати браузерам приймати лише певні сертифікати.

Цей підхід заважає будь-яким недобросовісним центрам сертифікацій видавати надійні сертифікати для будь-якого домену в Інтернеті. Іншим варіантом може бути, щоб сертифікаційний орган, який уповноважує записи DNS, може визначити, що сертифікати веб-сайту повинні видаватись певним центром сертифікації. Однак ці підходи принципово покладаються на набір специфікацій, що забезпечують безпеку інформації, яка надається засобами DNS в IP-мережах, для запобігання підробки та модифікації запитів DNS [22].

Допоки такий підхід не буде широко використовуватись в інтернеті, веб-сайти повинні розглядати інші альтернативні засоби захисту.

1.10 Машинне навчання

1.10.1 Модель роботи машинного навчання

Машинне навчання – це галузь штучного інтелекту, орієнтована на створення додатків, які навчаються на даних та покращують їх точність з часом, будучи не запрограмовані на це [23]. В науці про дані, алгоритм – це послідовність етапів статистичної обробки. У машинному навчанні алгоритми “навчаються” знаходити закономірності та особливості у великих обсягах даних, щоб приймати рішення та прогнози на основі нових даних. Чим кращим буде алгоритм, тим точнішими стануть рішення та прогнози в міру обробки більшої кількості даних.

Існує чотири основних етапи побудови моделі для машинного навчання. Зазвичай вони виконуються вченими з даних, тісно співпрацюючи з бізнес-професіоналами, для яких модель розробляється.

Першим кроком у машинному навчанні є вибір та підготовка

навчальних даних. Навчальні дані – це набір даних, які модель машинного навчання сприймає для вирішення проблеми, заради якої це все робиться. У деяких випадках навчальні дані позначаються, щоб визначити ознаки та класифікації, які модель повинна буде визначити. Інші дані не маркуються, і модель повинна буде самостійно витягти ці ознаки та призначити класифікації. У будь-якому випадку дані навчання повинні бути належним чином підготовлені, також вибірку слід розділити на навчальну, яка буде використана для навчання класифікатора, та на тестову, що використовується для тестування та вдосконалення [24].

Наступним кроком є вибір алгоритму для запуску набору навчальних даних. Тип алгоритму залежить від того позначені дані або ж ні, типу даних у наборі навчальних даних та від проблеми, яку потрібно вирішити.

Найпоширеніші типи алгоритмів машинного навчання з позначеними даними:

– лінійна та логістична регресія, які використовуються для розуміння взаємозв’язків у даних. Лінійна регресія використовується для прогнозування значення залежної змінної на основі значення незалежної змінної. Логістичну регресію можна використовувати, коли залежна змінна має двійковий характер;

– дерева рішень, які використовують класифіковані дані для вироблення рекомендацій на основі набору правил прийняття рішень. Наприклад, дерево рішень, яке рекомендує робити ставки на певного коня для перемоги, може використовувати дані про коня (наприклад, вік, відсоток виграшів, родовід) та застосовувати правила до цих факторів, щоб рекомендувати дію чи рішення;

– алгоритм на основі екземплярів, гарним прикладом такого алгоритму є k-NN. Він використовує класифікацію, щоб оцінити, наскільки ймовірно, що точка даних є членом тієї чи іншої групи, виходячи з її близькості до інших точок даних.

Алгоритми з непозначеними даними:

– алгоритми кластерізації, які працюють за методом розглядання кластерів як груп. Кластерізація фокусується на виявленні груп подібних записів та маркуванні записів відповідно до групи, до якої вони належать. Це робиться без попереднього знання про групи та їх характеристик;

– алгоритми асоціацій, які знаходять закономірності та взаємозв'язки в даних і визначають часті взаємозв'язки "if-then", які називаються правилами асоціацій. Вони подібні до правил, які використовуються при видобутку даних.

Після вибору алгоритму йде стадія навчання. Навчання алгоритму є ітераційним процесом, він передбачає запуск змінних через алгоритм, порівняння результату з результатами, які він повинен був отримати, коригування ваг та упереджень в алгоритмі, що може дати більш точний результат. Потім йде запуск змінних знову, допоки алгоритм не почне повертати правильний результат більшу частину часу [25].

Останній крок полягає у використанні моделі з новими даними. З новими даними можна перевірити роботу класифікатора.

1.10.2 Методи машинного навчання

Методи машинного навчання діляться на три основні категорії, а саме: навчання з учителем, напівавтоматичне навчання та навчання без учителя.

Навчання з учителем полягає в тому, що є приклади вхідних даних та бажаних вихідних результатів, які задані “вчителем”, а метою слугує навчання загального правила, яке відображає входи на виходи. Навчання з учителем вимагає менше навчальних даних, ніж інші методи машинного навчання, і полегшує навчання, оскільки результати моделі можна порівняти з фактично позначеними результатами.

Напівавтоматичне навчання є золотою серединою між навчанням з учителем та без учителя. Під час навчання, цей метод використовує менший набір позначених даних для керівництва класифікацією та вилучення ознак із більшого сету з непозначеними даними. Використовуючи такий метод можна

вирішити проблему недостатньої кількості позначених даних.

Навчання без вчителя працює за принципом, що використовує немарковані дані і використовує алгоритми для вилучення значущих функцій, необхідних для маркування, сортування та класифікації даних у режимі реального часу, без участі людини. Навчання без вчителя більше про виявлення закономірностей та взаємозв’язків у даних, які люди могли б пропустити.

2 РЕАЛІЗАЦІЯ МАШИННОГО НАВЧАННЯ

2.1 Підготовка середовища для реалізації завдання

Метою дипломної роботи є впровадження машинного навчання для детектування того, чи є цифровий сертифікат довіреним або ж ні. Якщо сертифікат шахрайський, то зловмисник може його використати для фішингу і знизити ризик виявлення користувачем. Коли жертва буде знаходитись на фішинговому веб-сайті, то браузер не зможе її попередити про фішинг, та буде вважати, що веб-сайт легітимний.

Для боротьби з фішинговими веб-сайтами та сертифікатами існують репозиторії які використовуються браузерами та операційними системами. Попередження користувача може сильно вплинути на його рішення, щодо того відвідувати такий сайт чи ні (рис. 2.1).

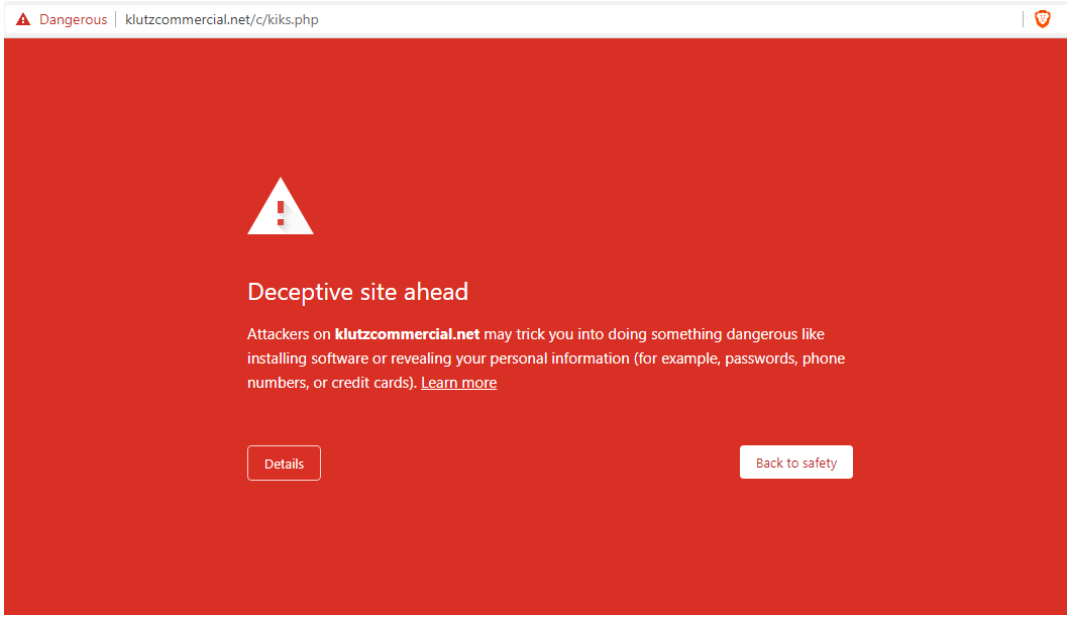

Рисунок 2.1 – Попередження користувача веб-браузером

Для реалізації роботи був використаний, такий дистрибутив, як Anaconda (рис. 2.2).

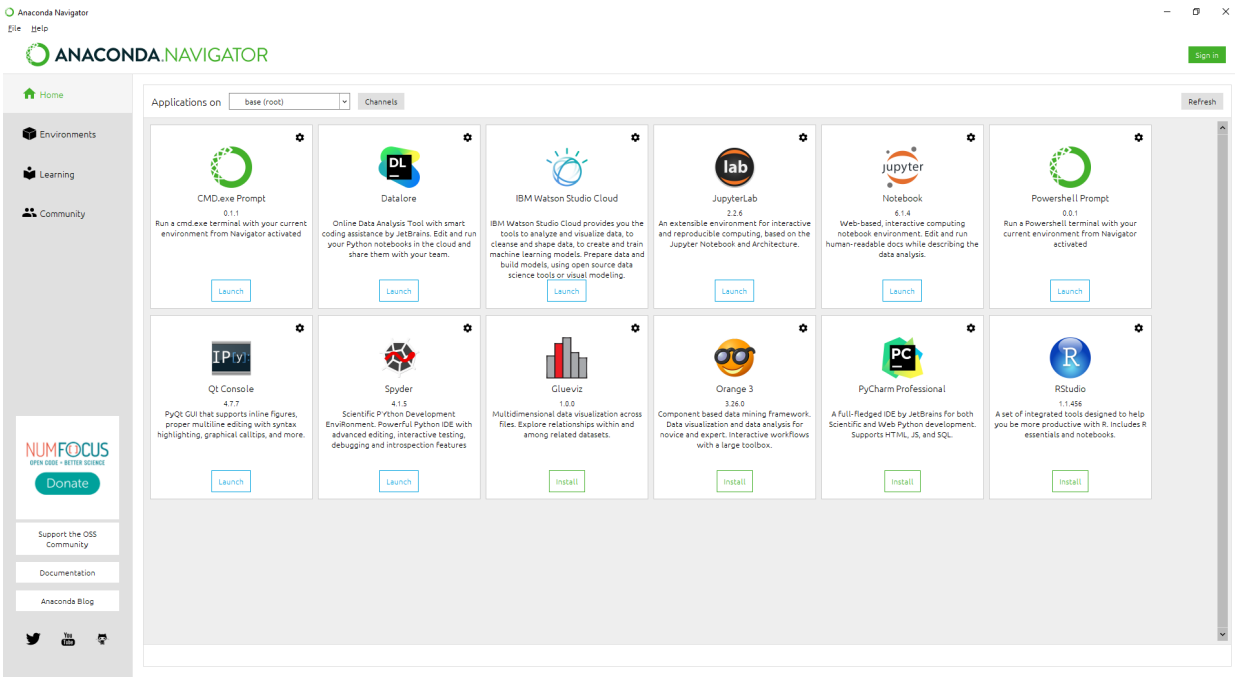

Рисунок 2.2 – Графічний інтерфейс Anaconda

Цей дистрибутив корисний тим, що він є самим популярним дистрибутивом Python та містить більш ніж 1000 різних пакетів, а також включає в собі широко використовувані бібліотеки, такі як pandas, numpy та інші. Використовуючи дистрибутив Anaconda можна швидко приступити до роботи не думаючи про те, що потрібно буде встановлювати різні відсутні пакети та бібліотеки, які потрібні для виконання завдання.

Програма, яка буде використовуватись для виконання завдання, називається Jupyter Notebook. Цей інструмент гарно себе зарекомендував у таких напрямах, як наука про дані та машинне навчання. Великим плюсом Jupyter Notebook є те, що проект можна зобразити в інтерактивному виді, він об'єднує код та відображає всю роботу в одному документі, дозволяє все зручно візуалізувати.

2.2 Реалізація навчання

Першим кроком у виконанні роботи є збір даних, які будуть використовуватись для машинного навчання. Даними будуть слугувати цифрові сертифікати з їх різними характеристиками. Для того, щоб

імпортувати набір даних у Jupyter Notebook, та щоб вони коректно зчитувались, потрібно створити таблицю з цифровими сертифікатами та їх характеристиками у форматі “.csv” (рис. 2.3).

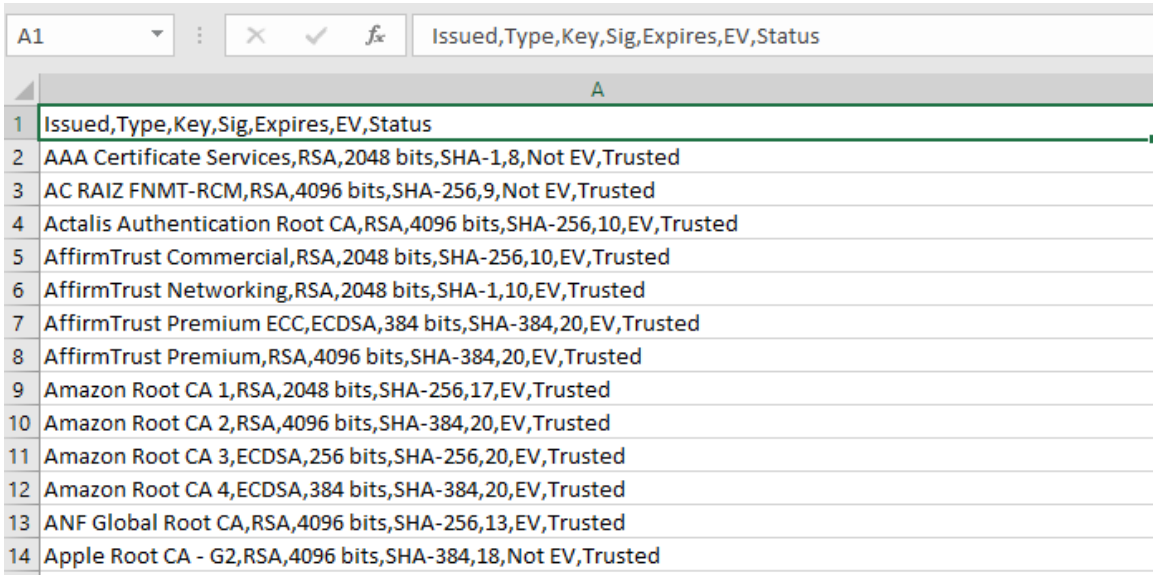

Рисунок 2.3 – Характеристики цифрових сертифікатів у “.csv” форматі

В такому форматі всі стовбці майбутньої таблиці будуть розподілені комами, та не мають структурованого вигляду, але після загрузки до проекту у Jupyter Notebook та імпорту “.csv” файлу, всі данні які знаходяться в файлі будуть відображені в виді класичної таблиці.

Для подальшого машинного навчання потрібно обрати характеристики цифрових сертифікатів. В даній роботі використані наступні характеристики:

1) Issued – ким виданий цифровий сертифікат;
2) Type – тип алгоритму цифрового підпису: RSA або ECDSA;
3) Key – розмір ключа: 2048 біт, 4096 біт, 384 біт, 256 біт, 1024 біт;
4) Sig – алгоритм підпису: SHA-1, SHA-256, SHA-384, SHA-512;
5) Expires – термін дії цифрового сертифікату (скільки років ще буде діяти сертифікат з листопаду 2020 року) ;
6) EV – розширена перевірка організації для якої видається цифровий сертифікат. Якщо перевірка була проведена, то встановлюється значення EV, коли не було перевірки – Not EV;

7) Status – статус цифрового сертифікату. Якщо сертифікату можна довіряти, то встановлюється значення Trusted, відповідно, коли сертифікат не заслуговує на довіру – Non Trusted.

У лістингу 2.1 зображено імпорт “.csv” файлу, побудова таблиці, та трансформація строкових даних у категоріальні.

Лістинг 2.1 – Імпорт таблиці в Jupyter Notebook, побудова та трансформація строкових даних у категоріальні

```
# Load CSV
import pandas as pd
import numpy as np
filename = 'Dataset.csv'
# Loading with NumPy
#raw_data = open(filename, 'rt')
#data = numpy.loadtxt(raw_data, delimiter=",")
# Loading with Pandas
data = pd.read_csv(filename)
print(data.shape)

# Transforming 'object' data to 'categorical' to get numerical (ordinal numbers) representation

data['Issued'] = data['Issued'].astype('category')
data['Type'] = data['Type'].astype('category')
data['Key'] = data['Key'].astype('category')
data['Sig'] = data['Sig'].astype('category')
data['EV'] = data['EV'].astype('category')
data['Status'] = data['Status'].astype('category')

data['Issued_code'] = data['Issued'].cat.codes
data['Type_code'] = data['Type'].cat.codes
data['Key_code'] = data['Key'].cat.codes
data['Sig_code'] = data['Sig'].cat.codes
data['EV_code'] = data['EV'].cat.codes
data['Status_code'] = data['Status'].cat.codes

#pd.options.display.max_rows=1000
pd.options.display.max_rows=100
data
```

Після виконання коду, на виході зображена таблиця (рис. 2.4). Ця таблиця містить великий набір цифрових сертифікатів (196 різних екземплярів).

	Issued	Type	Key	Sig	Expires	EV	Status	Issued_code	Type_code	Key_code	Sig_code	EV_code	Status_code
0	AAA Certificate Services	RSA	2048 bits	SHA-1	8	Not EV	Trusted	1	1	1	0	1	1
1	AC RAIZ FNMT-RCM	RSA	4096 bits	SHA-256	9	Not EV	Trusted	2	1	4	1	1	1
2	Actalis Authentication Root CA	RSA	4096 bits	SHA-256	10	EV	Trusted	4	1	4	1	0	1
3	AffirmTrust Commercial	RSA	2048 bits	SHA-256	10	EV	Trusted	5	1	1	1	0	1
4	AffirmTrust Networking	RSA	2048 bits	SHA-1	10	EV	Trusted	6	1	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
191	TRIAL PKIoverheid Organisatie TEST CA - G2	RSA	4096 bits	SHA-256	0	Not EV	Non Trusted	149	1	4	1	1	0
192	XRamp Global Certification Authority	RSA	2048 bits	SHA-1	8	Not EV	Non Trusted	171	1	1	0	1	0
193	TU Delft CA	RSA	2048 bits	SHA-1	0	Not EV	Non Trusted	151	1	1	0	1	0
194	Certification Authority of WoSign	RSA	2048 bits	SHA-256	9	Not EV	Non Trusted	31	1	1	1	1	0
195	COMODO RSA Domain Validation Secure Server CA	RSA	2048 bits	SHA-256	0	Not EV	Non Trusted	30	1	1	1	1	0

196 rows × 13 columns

Рисунок 2.4 – Таблиця з колекцією даних

Можна побачити, що всі строкові дані перетворені в категоріальні дані. Категоріальні дані – це такі дані, які не мають кількісного вираження, в них кожна одиниця призначається певній категорії на основі деякої якісної властивості [26].

У таблиці категоріальними даними є стовпці: Issued, Type, Key, Sig, Expires, EV, Status. Цим стовпцям присвоюються нові відповідні стовпці, а саме: Issued_code, Type_code, Key_code, Sig_code, EV_code, Status_code. Нові стовпці мають числа, що кодують ці категорії. Тільки стовпець Expires не потрібно перетворювати, тому що всі дані в ньому, це числа.

Наступним кроком потрібно нормалізувати дані, щоб всі закодовані числа були в межах від 0 до 1. У лістингу 2.2 зображена нормалізація даних у таблиці.

Лістинг 2.2 – Нормалізація даних у таблиці з колекцією даних

```
from sklearn.preprocessing import MinMaxScaler
min_max_scaler = MinMaxScaler()

X_raw = data[['Issued_code', 'Type_code', 'Key_code',
'Sig_code', 'Expires', 'EV_code']].values
X = min_max_scaler.fit_transform(X_raw)
y = data['Status_code'].values
```

```
    feature_names = data[['Issued_code', 'Type_code',
'Key_code', 'Sig_code', 'Expires',
'EV_code']].columns.values

    status_names = data['Status'].cat.categories
    issued_names = data['Issued'].cat.categories
    type_names = data['Type'].cat.categories
    key_names = data['Key'].cat.categories
    sig_names = data['Sig'].cat.categories
    EV_names = data['EV'].cat.categories
```

На “X” подаються наступні значення: Issued_code, Type_code, Key_code, Sig_code, Expires, EV_code. Ці значення будуть впливати на машинне навчання і слугують вхідними даними. На “y” подається Status_code, який показуватиме результат, сертифікат заслуговує на довіру чи ні.

Щоб перевірити нормалізацію, достатньо буде вивести “X” (рис. 2.5).

```
B [3]: X

Out[3]: array([[0.00568182, 1.        , 0.25      , 0.        , 0.30769231,
                1.        ],
               [0.01136364, 1.        , 1.        , 0.33333333, 0.34615385,
                1.        ],
               [0.02272727, 1.        , 1.        , 0.33333333, 0.38461538,
                0.        ],
               ...,
               [0.85795455, 1.        , 0.25      , 0.        , 0.        ,
                1.        ],
               [0.17613636, 1.        , 0.25      , 0.33333333, 0.34615385,
                1.        ],
               [0.17045455, 1.        , 0.25      , 0.33333333, 0.        ,
                1.        ]])
```

Рисунок 2.5 – Нормалізація даних

На рисунку 2.5 можна побачити, що всі дані коректно нормалізовані та знаходяться в межах від 0 до 1.

Після нормалізації всіх даних можна перейти до машинного навчання на основі імпортованого набору даних (навчальної вибірки). Для вирішення цієї задачі буде використаний алгоритм kNN (k-Nearest Neighbors). Алгоритм класифікації ближніх сусідів. Задачею класифікації слугує те, що об'єкт

відноситься к одному з заздалегідь обраних класів виходячи з його формалізованих ознак. Об'єкти представляються у вигляді вектору в N-вимірному просторі, де вимірюванням слугує опис ознак одного з об'єктів [27].

У випадку коли потрібно класифікувати цифрові сертифікати вимірюванням будуть слугувати вхідні дані, а саме: ким був виданий цифровий сертифікат, тип алгоритму цифрового підпису, розмір ключа, алгоритм підпису, термін дії цифрового сертифікату, чи була здійснена розширена перевірка організації, яка замовила сертифікат. Вихідними даними буде слугувати значення того чи є сертифікат довіреним або ж ні

Для навчання класифікатора буде використаний складений раніше датасет з атрибутами цифрових сертифікатів.

У лістингу 2.3 відбувається машинне навчання класифікатора на основі імпортованого набору даних.

Лістинг 2.3 – Машинне навчання

```
from sklearn import neighbors
# create the model
knn = neighbors.KNeighborsClassifier(n_neighbors=7)
# fit the model
knn.fit(X, y)
#predict
y_predict=knn.predict(X)
y_predict
```

Результати машинного навчання класифікатора у вигляді масиву зображені на рисунку 2.6. Як можна побачити в цьому масиві знаходиться правильна кількість цифрових сертифікатів, а саме така ж сама кількість яка була і в наборі даних (196 цифрових сертифікатів).

Результати які зображені в масиві:

1) 1 – Trusted;
2) 0 – Non Trusted.

```
Out[6]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1],
              dtype=int8)
```

Рисунок 2.6 – Результати навчання датасету

Якщо подивитися на масив та на датасет, то неозброєним оком можна помітити при навчанні з'явилися похибки. Набор даних був складений таким чином, що останні 38 сертифікатів були Non Trusted. Дивлячись на результати, можна побачити, що частина Non Trusted сертифікатів помічна як Trusted і навпаки, деякі Trusted сертифікати зображені, як Non Trusted (рис. 2.7).

```
Out[6]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1],
              dtype=int8)
```

Рисунок 2.7 – Похибки які з'явилися при навчанні

Список “Non Trusted” цифрових сертифікатів, які були класифіковані, як “Trusted”:

1) Class 3 Public Primary Certification Authority, RSA, 1024 bits, SHA-1, 8, Not EV, Non Trusted;

2) DigiNotar Root CA G2, RSA, 4096 bits, SHA-256, 9, Not EV, Non Trusted;

3) TRIAL PKIoverheid Organisatie TEST CA - G2, RSA, 4096 bits, SHA-256, 0, Not EV, Non Trusted;

4) XRamp Global Certification Authority, RSA, 2048 bits, SHA-1, 8, Not EV, Non Trusted;

5) Certification Authority of WoSign,RSA,2048 bits,SHA-256,9,Not EV,Non Trusted;

6) COMODO RSA Domain Validation Secure Server CA, RSA, 2048 bits, SHA-256, 0, Not EV, Non Trusted.

Список “Trusted” цифрових сертифікатів, які були класифіковані, як “Non Trusted ”:

1) Certinomis - Autorite Racine, RSA, 4096 bits, SHA-1, 8, Not EV, Trusted;

2) DST Root CA X3, RSA, 2048 bits, SHA-1, 1, Not EV, Trusted;

3) GeoTrust Global CA, RSA, 2048 bits, SHA-1, 2, Not EV, Trusted;

4) Hongkong Post Root CA 1, RSA, 2048 bits, SHA-1, 2, Not EV, Trusted;

5) Sonera Class2 CA, RSA, 2048 bits, SHA-1, 1, Not EV, Trusted;

6) Trustis FPS Root CA, RSA, 2048 bits, SHA-1, 3, Not EV, Trusted;

7) VRK Gov Root CA, RSA, 2048 bits, SHA-1, 3, Not EV, Trusted.

За допомогою наступного скрипта можна візуалізувати залежність між різними характеристиками цифрових сертифікатів (лістинг 2.4).

Лістинг 2.4 – Візуалізація даних за допомогою побудови 2D графіку

```
import matplotlib.pyplot as plt
x_index = 0
y_index = 4

formatter = plt.FuncFormatter(lambda i, *args:
status_names[int(i)])
plt.scatter(X[:, x_index], X[:, y_index], c=y,
cmap=plt.cm.get_cmap('Paired', 2))

plt.colorbar(ticks=[0, 1], format=formatter)
plt.clim(-0.5, 1.5)
plt.xlabel(feature_names[x_index])
plt.ylabel(feature_names[y_index]);
plt.show()
```

Для того, щоб змінити атрибути на інші, достатньо лише задати

значення для осі “x” та “y” використовуючи порядковий номер атрибуту.

Порядкові номера для атрибутів:

1) Issued_code – 0;
2) Type_code – 1;
3) Key_code – 2;
4) Sig_code – 3;
5) Expires – 4;
6) EV_code – 5.

На рисунку 2.8 зображена залежність між тим, ким був виданий цифровий сертифікат та терміну його дії.

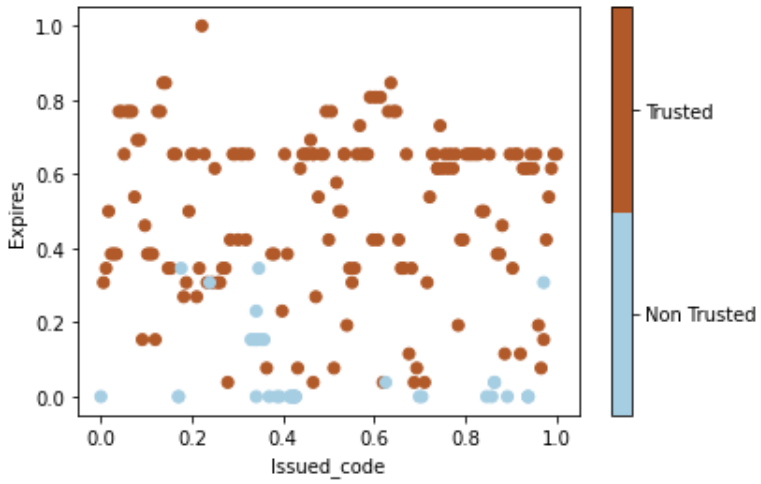

Рисунок 2.8 – Залежність між тим, ким був виданий сертифікат та терміном його дії

На рисунку 2.8 кожна точка відповідає певному цифровому сертифікату. Всі коричневі точки – довірені сертифікати, а сині точки – недовірені. Розглядаючи цей графік стає очевидним, що на класифікацію об'єктів впливає термін дії цифрового сертифікату. Чим більше цей термін, тим більше ймовірність, що сертифікат довірений.

Також є можливість візуалізувати не тільки у вигляді двомірного графіку, а й побудувати тривимірну модель (ліст. 2.5).

Лістинг 2.5 – Візуалізація даних за допомогою побудови 3D графіку

```
        from mpl_toolkits.mplot3d import Axes3D
        fig = plt.figure(1, figsize=(10, 8))
        plt.clf()
        ax = Axes3D(fig, rect=[0, 0, 1, 1], elev=30, azim=100)
        ax.scatter(X[:, 0], X[:, 3], X[:, 4], lw=2, c=y,
cmap='Paired')
        ax.set_xlabel(feature_names[0])
        ax.set_ylabel(feature_names[3]);
        ax.set_zlabel(feature_names[4]);
        plt.show()
```

Аналогічно з двомірною моделлю, порядкові номери атрибутів можна задавати для осей “x”, “y” та “z”. Побудована модель зображена на рисунках 2.9 – 2.10.

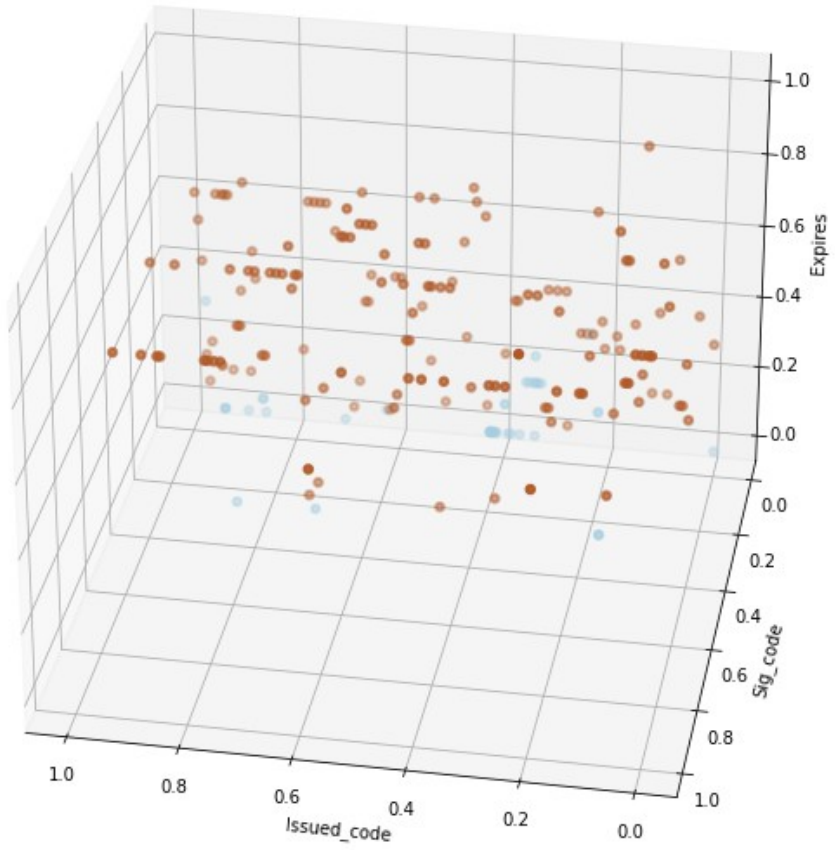

Рисунок 2.9 – Побудована 3D модель

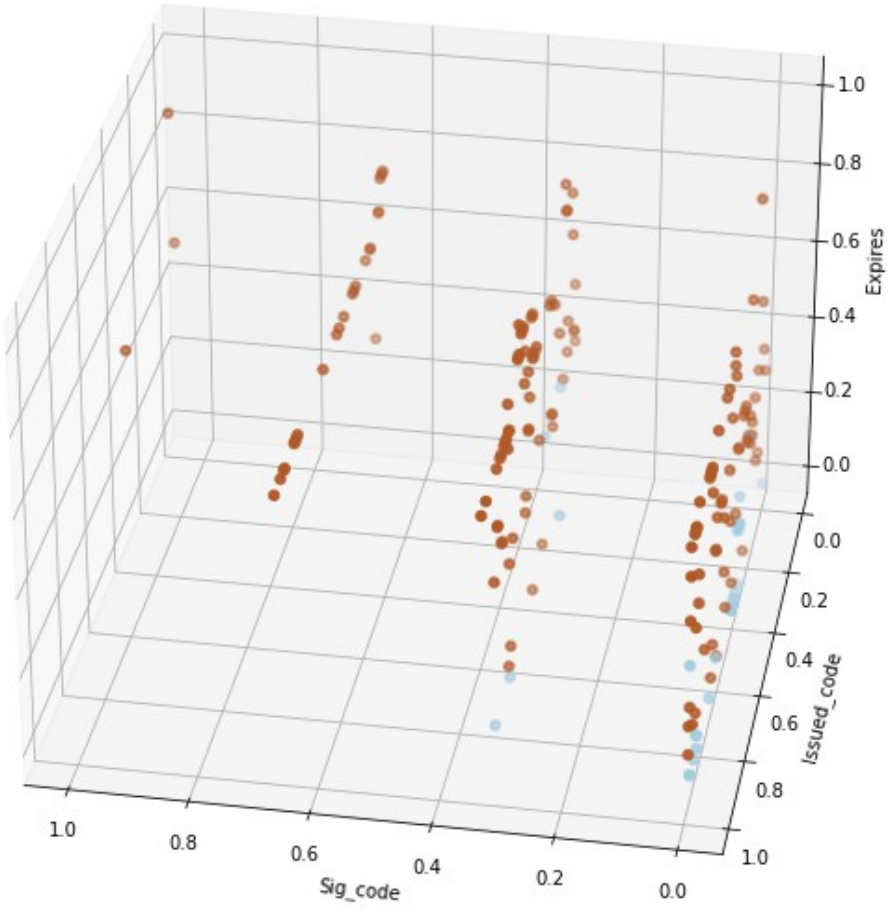

Рисунок 2.10 – Побудована 3D модель

Аналізуючи дані побудовані моделі, можна побачити як впливає термін дії цифрового сертифіката, алгоритм підпису та ким був виданий сертифікат.

Щоб протестувати машинне навчання, необхідно, окрім навчального набора даних, скласти ще й тестовий [28]. Тестовий набір даних не буде таким об'ємним, як навчальний. В цьому немає сенсу, достатньо лише декілька цифрових сертифікатів, щоб подивитися як класифікатор їх буде класифікувати, довіреними або ж недовіреними. Складений тестовий набір даних, міститиме 10 цифрових сертифікатів, з яких 6 – довірені, та використовуються на різних довірених сайтах. Останні 4 цифрових сертифікати є недовіреними, та використовуються на фішингових веб-сайтах.

За допомогою ресурсу FishTank можна знайти фішингові веб-сайти та подивитись на реальні приклади своїми очима, також можна досліджувати, які сертифікати використовуються фішинговими сайтами. Найбільший

інтерес представляють сайти з протоколом https. Зараз середньостатистичний користувач інтернету вже знає, що веб-сайти з протоколом http небезпечні, та завжди обиратиме сайт з протоколом https. Приклад виявлених фішингових сайтів та завантажених на FishTank (рис. 2.11).

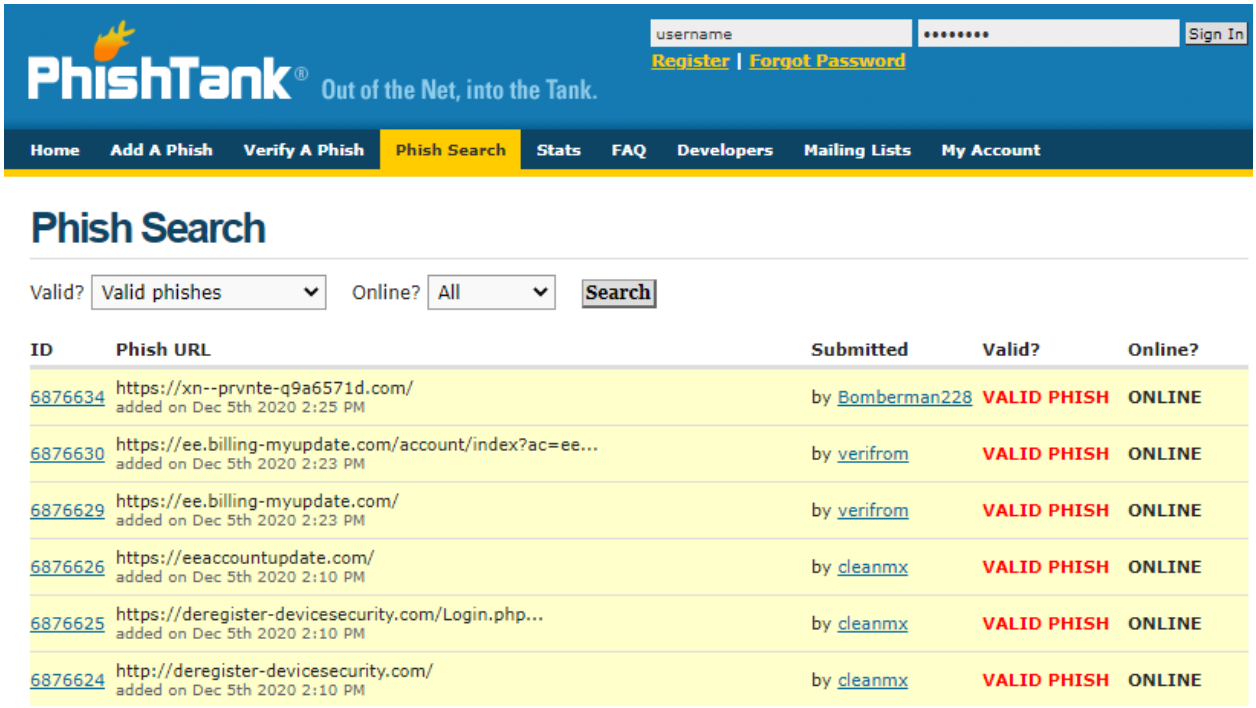

Рисунок 2.11 – Репозиторій фішингових веб-сайтів

Аналогічно з навчальним набором даних потрібно підтягнути й тестову вибірку у вигляді таблиці (лістинг 2.6).

Лістинг 2.6 – Імпорт тестового набору побудова таблиці та трансформація строкових даних у категоріальні

```
        # Load CSV
        import pandas as pd
        import numpy as np

        filename = 'Testset.csv'
        # Specify the names of attributes if the header is not
availabel in a CSV file
        #names = ['Issued_by', 'Type', 'Key_size', 'Sig_alg',
'EV_policy', 'Status' ]
        # Loading with Pandas
        datatest = pd.read_csv(filename)
print(datatest.shape)

        # Transforming 'object' data to 'categorical' to get
numerical (ordinal numbers) representation
```

```
        datatest['Issued'] =
datatest['Issued'].astype('category')
        datatest['Type'] = datatest['Type'].astype('category')
        datatest['Key'] = datatest['Key'].astype('category')
        datatest['Sig'] = datatest['Sig'].astype('category')
        datatest['EV'] = datatest['EV'].astype('category')
        datatest['Status'] =
datatest['Status'].astype('category')

        datatest['Issued_code'] = datatest['Issued'].cat.codes
        datatest['Type_code'] = datatest['Type'].cat.codes
        datatest['Key_code'] = datatest['Key'].cat.codes
        datatest['Sig_code'] = datatest['Sig'].cat.codes
        datatest['EV_code'] = datatest['EV'].cat.codes
        datatest['Status_code'] = datatest['Status'].cat.codes

        #pd.options.display.max_rows=1000
        pd.options.display.max_rows=100
        datatest
```

На рисунку 2.12 зображена таблиця з тестовим набором. Всі строкові дані перетворені в категоріальні. Сама таблиця має таку ж структуру, як і в навчальному наборі.

	Issued	Type	Key	Sig	Expires	EV	Status	Issued_code	Type_code	Key_code	Sig_code	EV_code	Status_code
0	DigiCert High Assurance EV Root CA	RSA	2048 bits	SHA-1	11	EV	Trusted	5	1	0	0	0	1
1	Apple Root CA	RSA	2048 bits	SHA-1	14	Not EV	Trusted	1	1	0	0	1	1
2	ComSign Global Root CA	RSA	4096 bits	SHA-256	16	Not EV	Trusted	4	1	2	1	1	1
3	Go Daddy Class 2 Certification Authority	RSA	2048 bits	SHA-1	14	EV	Trusted	7	1	0	0	0	1
4	SSL.com EV Root Certification Authority ECC	ECDSA	384 bits	SHA-256	20	EV	Trusted	8	0	1	1	0	1
5	Baltimore CyberTrustRoot	RSA	2048 bits	SHA-1	4	EV	Trusted	2	1	0	0	0	1
6	GlobalSign Root CA	RSA	2048 bits	SHA-1	7	EV	Trusted	6	1	0	0	0	1
7	Actalis Domain Validation Server CA G3	RSA	2048 bits	SHA-1	1	Not EV	Non Trusted	0	1	0	0	1	0
8	cPanel inc. Certification Authority	RSA	2048 bits	SHA-1	1	Not EV	Non Trusted	10	1	0	0	1	0
9	CNNIC ROOT	RSA	2048 bits	SHA-1	3	Not EV	Non Trusted	3	1	0	0	1	0
10	TU Delft CA	RSA	2048 bits	SHA-256	7	Not EV	Non Trusted	9	1	0	1	1	0

Рисунок 2.12 – Таблиця з колекцією даних

Наступним кроком буде нормалізація даних у тестовій вибірці та виклик методу knn.predict, для того щоб перевірити як класифікатор передбачує наявність того, довірений сертифікат або ні (лістинг 2.7).

Лістинг 2.7 – Нормалізація даних та перевірка метода передбачення

```
    # call the "predict" method:
X_test = datatest[['Issued_code', 'Type_code', 'Key_code',
'Sig_code', 'Expires', 'EV_code']].values
    X_test2 = min_max_scaler.fit_transform(X_test)
    y_test = datatest['Status_code'].values
    testresult = knn.predict(X_test2)
    testfeature_names = datatest[['Issued_code',
'Type_code', 'Key_code', 'Sig_code', 'Expires',
'EV_code']].columns.values
    teststatus_names = datatest['Status'].cat.categories
    testissued_names = datatest['Issued'].cat.categories
    testtype_names = datatest['Type'].cat.categories
    testkey_names = datatest['Key'].cat.categories
    testsig_names = datatest['Sig'].cat.categories
    testEV_names = datatest['EV'].cat.categories
```

Всі дані у тестовій вибірці були нормалізовані, це можна перевірити вивівши X_test2 (рис. 2.13).

```
B [9]: X_test2

Out[9]: array([[0.5       , 1.        , 0.        , 0.        , 0.52631579,
                0.        ],
               [0.1       , 1.        , 0.        , 0.        , 0.68421053,
                1.        ],
               [0.4       , 1.        , 1.        , 1.        , 0.78947368,
                1.        ],
               [0.7       , 1.        , 0.        , 0.        , 0.68421053,
                0.        ],
               [0.8       , 0.        , 0.5       , 1.        , 1.        ,
                0.        ],
               [0.2       , 1.        , 0.        , 0.        , 0.15789474,
                0.        ],
               [0.6       , 1.        , 0.        , 0.        , 0.31578947,
                0.        ],
               [0.        , 1.        , 0.        , 0.        , 0.        ,
                1.        ],
               [1.        , 1.        , 0.        , 0.        , 0.        ,
                1.        ],
               [0.3       , 1.        , 0.        , 0.        , 0.10526316,
                1.        ],
               [0.9       , 1.        , 0.        , 1.        , 0.31578947,
                1.        ]])
```

Рисунок 2.13 – Нормалізація даних у тестовій вибірці

Для того, щоб перевірити результати роботи навченого класифікатора, який проаналізував тестову вибірку треба вивести testresult (рис 2.14).

```
B [10]: testresult
Out[10]: array([1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1], dtype=int8)
```

Рисунок 2.14 – Результат тестування

В результаті тестування можна побачити, що як і передбачалось класифікатор не працює ідеально. Проаналізувавши тестову вибірку, можна сказати, що 10 із 11 цифрових сертифікатів були класифіковані вірно, лише 1 екземпляр не відповідає дійсності.

Всі 7 довірених цифрових сертифікатів визначились як “Trusted”:

1) DigiCert High Assurance EV Root CA, RSA, 2048 bits, SHA-1, 11, EV, Trusted;
2) Apple Root CA, RSA, 2048 bits, SHA-1,14, Not EV, Trusted;
3) ComSign Global Root CA, RSA, 4096 bits, SHA-256, 16, Not EV, Trusted;
4) Go Daddy Class 2 Certification Authority, RSA, 2048 bits, SHA-1, 14, EV, Trusted;
5) SSL.com EV Root Certification Authority ECC, ECDSA, 384 bits, SHA-256, 20, EV, Trusted;
6) Baltimore CyberTrustRoot ,RSA, 2048 bits, SHA-1, 4, EV, Trusted;
7) GlobalSign Root CA, RSA, 2048 bits, SHA-1, 7, EV, Trusted.

Щодо недовірених сертифікатів, 3 із 4 цифрових сертифікатів вірно визначились як “Non Trusted”:

1) Actalis Domain Validation Server CA G3, RSA, 2048 bits, SHA-1, 1, Not EV, Non Trusted;
2) cPanel inc. Certification Authority, RSA, 2048 bits, SHA-1, 1, Not EV, Non Trusted;
3) CNNIC ROOT, RSA, 2048 bits, SHA-1, 3, Not EV, Non Trusted.

Лише 1 “Non Trusted” цифровий сертифікат визначився як “Trusted”:

1) TU Delft CA, RSA, 2048 bits, SHA-256, 7, Not EV, Non Trusted.

Саме цей цифровий сертифікат через похибку класифікатора був

класифікований невірно.

Щоб продемонструвати, що класифікатор в цілому працює гарно, можна розглянути недовірений сертифікат, та проаналізувати веб-сайт. Для прикладу був взятий перший за списком недовірений сертифікат, виданий – Actalis Domain Validation Server CA G3 (8 за списком у тестовій вибірці).

Був знайдений такий веб-сайт (рис. 2.15).

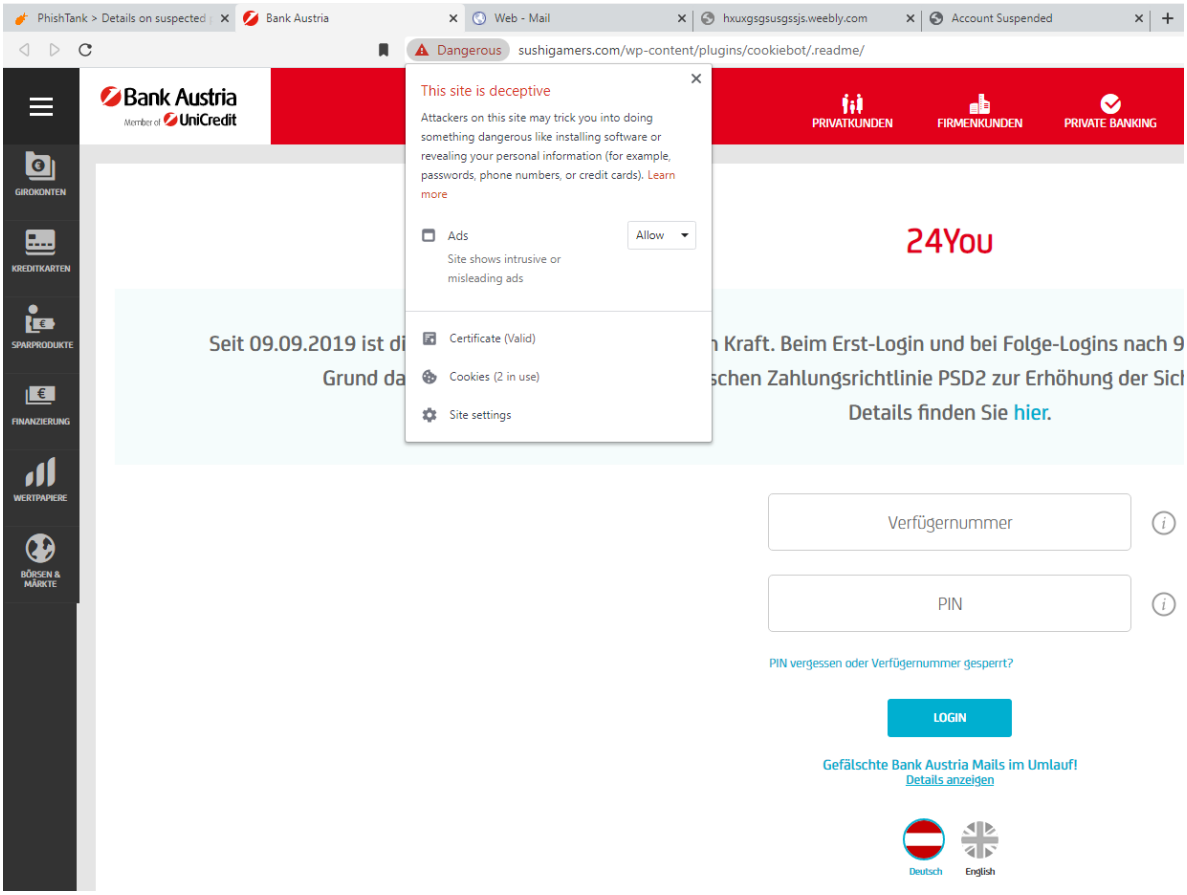

Рисунок 2.15 – Фішинговий веб-сайт

Можна побачити, що браузер попереджує користувача, що цей сайт небезпечний, але він вже знаходиться у репозиторії, як фішинговий, тому є попередження. Якщо б це був зовсім “свіжий” сайт, то такого б попередження не було. Сам сайт відтворений доволі майстерно, але має поганий URL, який зразу наводить на думку, що щось тут не так.

Дуже дивно, що сайту для онлайн банкінгу має URL:

https://www.sushigamers.com/wp-content/plugins/cookiebot/.readme/.

Проте можна зробити припущення, що сторінка з онлайн банкінгом повинна була відкриватись, коли користувач хотів щось придбати, або заплатити за якусь послугу.

Цей фішинговий сайт був розроблений задля того, щоб красти банківські дані користувачів банку Австрії. Якщо користувач введе свої дані, то вони будуть відправлені злочинцю, який зможе їх використати в своїх цілях, та вкрасти гроші з облікового запису користувача, тощо.

Варто відзначити, що хоча й цей веб-сайт фішинговий, проте він має дійсний цифровий сертифікат, саме той, який розглянутий в тестовій вибірці (рис 2.16).

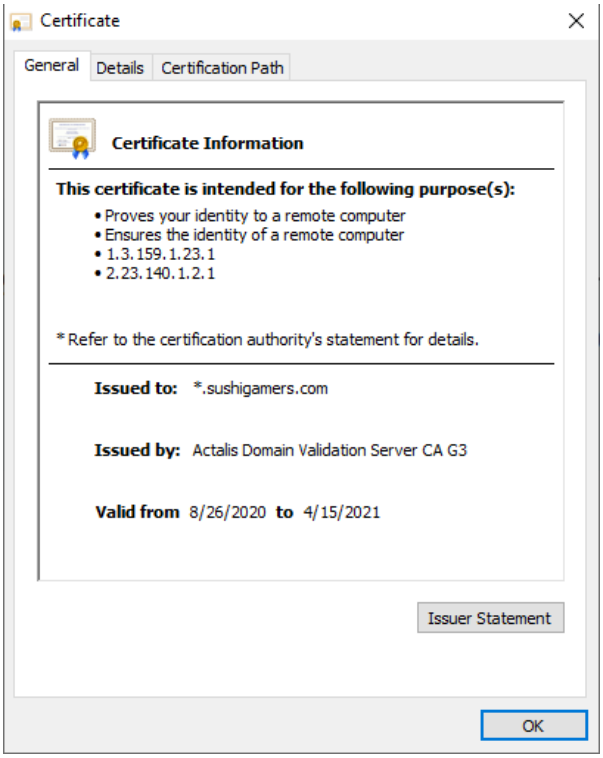

Рисунок 2.16 – Цифровий сертифікат

Таким чином можна впевнено сказати, що класифікатор вірно вказав на те, що цей сертифікат недовірений, а для того, щоб зовсім не залишилось сумнівів треба перевірити його на FishTank (рис. 2.17).

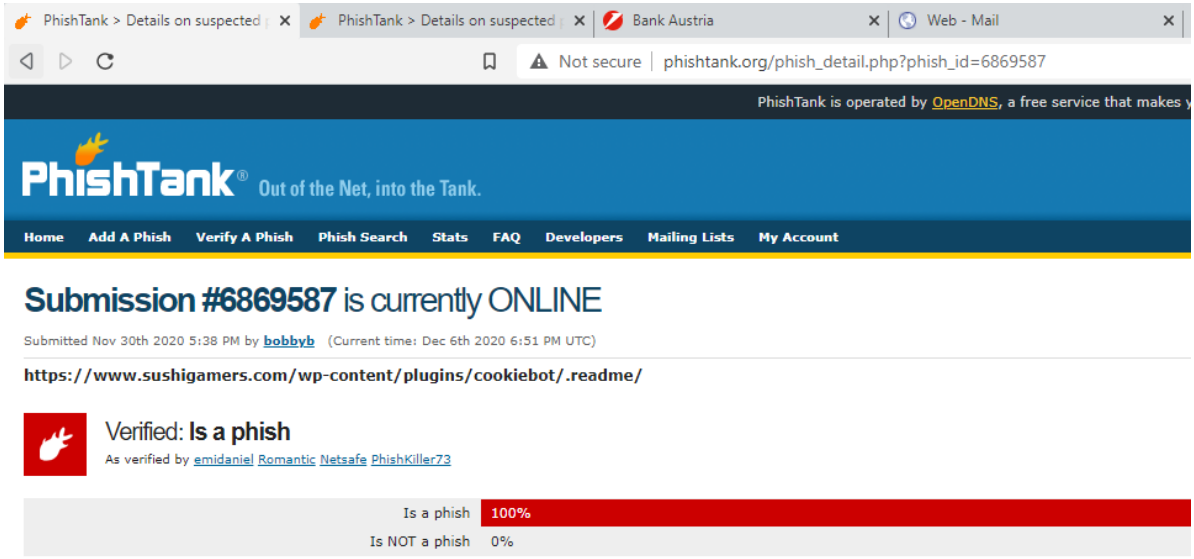

Рисунок 2.17 – Перевірка сайту на FishTank

Повертаючись до класифікатора, можна відобразити границю прийняття рішення класифікатора (лістинг 2.8).

Лістинг 2.8 – Границя прийняття рішення класифікатора

```
from matplotlib.colors import ListedColormap

n_neighbors = 7
h = .02  # step size in the mesh

# Create color maps
cmap_light = ListedColormap(['cyan', 'red'])
cmap_bold = ListedColormap(['blue', 'orange'])

# Get '1: Lifetime' and '2: Country' attributes only
x_index = 0
y_index = 4
X2 = X[:,[x_index, y_index]]

for weights in ['uniform', 'distance']:
 # we create an instance of Neighbours Classifier and fit
the data.
     knn = neighbors.KNeighborsClassifier(n_neighbors,
weights=weights)
     knn.fit(X2, y)

     # Plot the decision boundary. For that, we will assign
a color to each
     # point in the mesh [x_min, x_max]x[y_min, y_max].
     x_min, x_max = X2[:, 0].min() - 1, X2[:, 0].max() + 1
     y_min, y_max = X2[:, 1].min() - 1, X2[:, 1].max() + 1
     xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
```

```
                            np.arange(y_min, y_max, h))
        Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])

        # Put the result into a color plot
        Z = Z.reshape(xx.shape)
        plt.figure()
        plt.pcolormesh(xx, yy, Z, cmap=cmap_light)

        # Plot also the training points
        plt.scatter(X2[:, 0], X2[:, 1], c=y, cmap=cmap_bold,
                    edgecolor='k', s=35)
        plt.xlim(xx.min(), xx.max())
        plt.ylim(yy.min(), yy.max())
        plt.title("2-Class classification (k = %i, weights =
'%s')"
                  % (n_neighbors, weights))
        plt.xlabel(feature_names[x_index])
        plt.ylabel(feature_names[y_index]);

        plt.show()
```

Таким чином можна розглянути границю прийняття рішення класифікатора (рис 2.18 – 2 .19). Розглядаючи ці рисунки потрібно сказати, що:

– всі помаранчеві крапки, які потрапили до лазурної зони, це довірені цифрові сертифікати, які класифікатор правильно розцінив як “Trusted”;

– всі помаранчеві крапки, які потрапили до червоної зони, це довірені цифрові сертифікати, які класифікатор неправильно розцінив як “Non Trusted”;

– всі сині крапки, які потрапили до лазурної зони, це недовірені цифрові сертифікати, які класифікатор неправильно розцінив як “Trusted”;

– всі сині крапки, які потрапили до червоної зони, це недовірені цифрові сертифікати, які класифікатор правильно розцінив як “Non Trusted”.

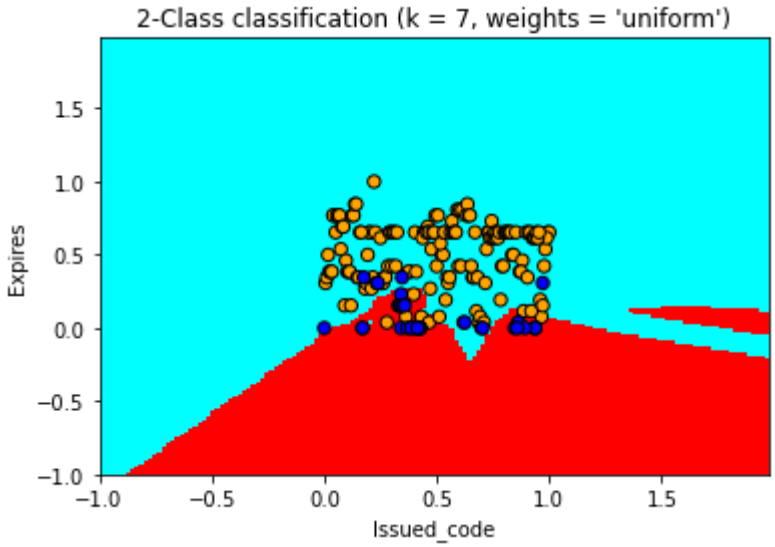

Рисунок 2.18 – Границя прийняття рішення класифікатора

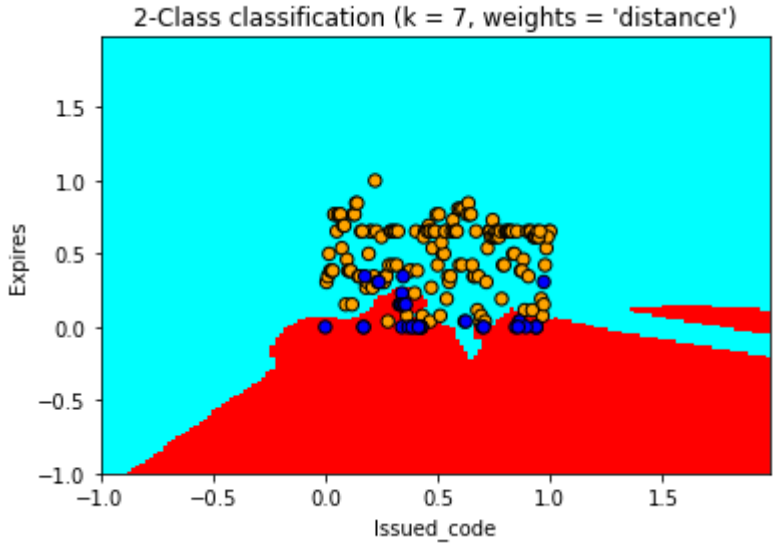

Рисунок 2.19 – Границя прийняття рішення класифікатора

На рисунку 2.18 зображений бінарний класифікатор на основі алгоритму k-NN. Кількість сусідів k=7; вагома функція – рівномірна (всі точки враховуючи всіх сусідів мають однакову вагу).

На рисунку 2.19 зображений бінарний класифікатор на основі алгоритму k-NN. Кількість сусідів k=7; вагома функція – віддалена (ближчі сусіди матимуть більший вплив, ніж сусіди, що знаходяться далі).

3 РЕЗУЛЬТАТИ РОБОТИ КЛАСИФІКАТОРА

Для того, щоб дати оцінку класифікатору мають враховуватися наступні показники, а саме:

– істинно-позитивний показник (TP) – кількість недовірених цифрових сертифікатів, які визначились як “Non Trusted”;

– істинно-негативний показник (TN) – кількість довірених цифрових сертифікатів, які визначились як “Trusted”;

– помилково-позитивний показник (FP) – кількість довірених цифрових сертифікатів, які визначились як “Non Trusted”;

– помилково-негативний показник (FN) – кількість недовірених цифрових сертифікатів, які визначились як “Trusted”;

Для розрахунку повноти, або ж частки істинно-позитивного показника (TPR) використовується формула:

, (3.1)

де – істинно-позитивний показник;

– помилково-негативний показник.

Для розрахунку частки істинно-негативного показника (TNR) використовується формула:

, (3.2)

де – істинно-позитивний показник;

– помилково-позитивний показник.

Для розрахунку частки помилково-позитивного показника (FPR) використовується формула:

, (3.3)

де – істинно-позитивний показник;

– помилково-позитивний показник.

Для розрахунку частки помилково-негативного показника (FNR) використовується формула:

, (3.4)

де – істинно-позитивний показник;

– помилково-негативний показник.

Для розрахунку точності, або ж позитивного передбачуваного значення (PPV) використовується формула:

, (3.5)

де – істинно-позитивний показник;

– помилково-позитивний показник.

Для розрахунку негативного передбачуваного значення (NPV) використовується формула:

, (3.6)

де – істинно-позитивний показник;

– помилково-негативний показник.

Для розрахунку точності використовується формула:

, (3.7)

де – істинно-позитивний показник;

– істинно-позитивний показник;

– помилково-позитивний показник.

Для розрахунку F-міри, а саме середнього гармонійного значення між повнотою (TPR) та точністю (PPV) використовується формула:

, (3.8)

де – позитивне передбачуване значення;

– істинно-позитивний показник.

Наступним кроком буде підрахунок всіх згаданих вище показників, для оцінки роботи класифікатора (лістинг 3.1).

Лістинг 3.1 – Оцінка роботи класифікатора

```
TP = 0
TN = 0
FP = 0
FN = 0

for i in range (0, len(y)):

    if (y[i] == 1):
        if (y[i] == y_predict[i]):
            TN+=1
        else:
            FP+=1
    else:
        if (y[i] == y_predict[i]):
            TP+=1
        else:
            FN+=1


print("TP =", TP, "TN =", TN, "FP =", FP, "FN =", FN)

TPR = TP / (TP+FN)
TNR = TN / (TN+FP)
FPR = FP / (FP+TN)
FNR = FN / (TP+FN)
PPV = TP / (TP+FP)
NPV = TN / (TN+FN)
Accuracy = (TP+TN) / (TP+TN+FP+FN)
Fmeasure = 2*PPV*TPR / (PPV + TPR)

print("TPR =", TPR, "TNR =", TNR)
print("FPR =", FPR, "FNR =", FNR)
print("PPV =", PPV, "NPV =", NPV)
print("F-measure =", Fmeasure)
print("Accuracy =", Accuracy)
```

Після виконання коду отримаємо такі результати (рис. 3.1).

```
TP = 32 TN = 151 FP = 7 FN = 6
TPR = 0.8421052631578947 TNR = 0.9556962025316456
FPR = 0.04430379746835443 FNR = 0.15789473684210525
PPV = 0.8205128205128205 NPV = 0.9617834394904459
F-measure = 0.8311688311688312
Accuracy = 0.9336734693877551
```

Рисунок 3.1 – Результат роботи класифікатора

Результати роботи класифікатора зображені в таблицях 3.1 – 3.2.

Таблиця 3.1 – Результати роботи класифікатора

Цифрові сертифікати	Кількість цифрових сертифікатів
Недовірені сертифікати визначені як недовірені (TP)	32
Довірені сертифікати визначені як довірені (TN)	151
Довірені сертифікати визначені як недовірені (FP)	7
Недовірені сертифікати визначені як довірені (FN)	6

Таблиця 3.2 – Результати роботи класифікатора

Показники	Результат	Результат у відсотках
Істинно позитивний показник (TPR)	0.84	84 %
Істинно-негативний показник (TNR)	0.96	96 %
Помилково-позитивний показник (FPR)	0.04	4 %
Помилково-негативний показник (FNR)	0.16	16 %
Позитивне передбачуване значення (PPV)	0.93	93 %
Негативне передбачуване значення (NPV)	0.96	96 %
Точність (Accuracy)	0.93	93 %
Середньо-гармонійне значення (F-measure)	0.83	83 %

ВИСНОВКИ

В даній атестаційній роботі були розглянуті принципи роботи цифрових сертифікатів та центрів сертифікацій, які їх видають. Окрема увага надавалась використанню цифрових сертифікатів у шахрайських цілях. Були наведені реальні приклади фішингу з використанням шахрайського сертифікату для отримання персональних даних користувача, а саме фішинговий сайт банку.

Запропонований один із методів виявлення недовірених цифрових сертифікатів, а саме метод оцінки ризиків цифрових сертифікатів з використанням машинного навчання.

Були створені дві вибірки з цифровими сертифікатами: навчальна вибірка, яка містить 196 сертифікатів, та тестова вибірка, яка містить 11 сертифікатів. За допомогою навчальної вибірки було здійснено навчання класифікатора. Тестова вибірка складена для перевірки роботи класифікатора. Дані про всі цифрові сертифікати, які використовувались в навчальній вибірці, збирались з актуального і довіреного репозиторію компанії Apple. Для чистоти експерименту дані про недовірені цифрові сертифікати, які використовувались в тестовій вибірці, збирались зі справжніх фішингових веб-сайтів. Дані про довірені цифрові сертифікати в тестовій виборці збирались з довірених і популярних веб-сайтів. Для машинного навчання був обраний алгоритм k-NN.

В результаті роботи був отриманий класифікатор, який, маючи тестову вибірку, може зробити передбачення щодо цифрового сертифікату – чи є він довіреним або ж ні. Результати роботи класифікатора були проаналізовані. Класифікатор показав гарні результати, але ще є шляхи для вдосконалення.

Як варіант покращення роботи класифікатора можна розширювати навчальну вибірку, маючи більше екземплярів цифрових сертифікатів можна покращити результат передбачення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Metcalf L. Blacklist Ecosystem Analysis: Spanning Jan 2012 to Jun 2014 /L. Metcalf, J. Spring // Proceedings of the 2nd ACM Workshop on Information Sharing and Collaborative Security. – 2015. – January. – P. 13– 22.

2. Rivest, R. A method for obtaining digital signatures and public-key cryptosystems / R. Rivest, A. Shamir, L. Adleman // Communications of the ACM. 21 (2). – 1978. – P. 120–126.

3. Huang Y. Decentralized Public Key Infrastructure (DPKI): What is it and why does it matter? / Y. Huang // Hacker Noon. Retrieved 2019-05-22.

4. Merkle R. C. A digital signature based on a conventional encryption function / R. C. Merkle // In Proc. of CRYPTO’87. 1988. – Springer. – P. 369–378.

5. Representation and verification of domain-based application service identity within Internet Public Key Infrastructure using X.509 (PKIX) certificates in the context of transport layer security (TLS) [Електронний ресурс] – Режим доступу: http://tools.ietf.org/html/rfc6125, 2011. – Дата доступу: 20.10.2020 р. – Загол. з екрану.

6. Oppliger R. I. SSL and TLS: Theory and Practice. Second Edition. – Artech House, 2016. – 301 p.

7. Barnes R.L. DANE: Taking TLS authentication to the next level using DNSSEC / R. L. Barnes // IETF J. Oct. – 2011. – P. 5-11.

8. Dierks T. The Transport Layer Security (TLS) Protocol Version 1.2 / T. Dierks, E. Rescorla // RFC 5246. – August. – 2008. – P. 17-23.

9. Hoffman-Andrews J. ETS Isn't TLS and You Shouldn't Use It. / J. Hoffman-Andrews // Electronic Frontier Foundation. – 26 February. – 2019.

10. Fromknech C. A Decentralized Public Key Infrastructure with Identity Retention / C. Fromknech // IACR Cryptology ePrint Archive. – November 2014.

11. Henry W. Trusted Third Party Service [Електронний ресурс] / W. Henry – Режим доступу: http://www.businessdictionary.com/definition/Trusted-Third-Party-Services-TTP-Services.html − Дата доступу: 01.11.2020 р. – Загол. з екрану.

12. Market share trends for SSL certificate authorities [Електронний ресурс] / W3Techs. Режим доступу: https://w3techs.com/technologies/history_overview/ssl_certificate − Дата доступу: 10.11.2020 р. – Загол. з екрану.

13. CA/Browser Forum. Baseline Requirements Certificate Policy for the Issuance and Management of Publicly-Trusted Certificates [Електронний ресурс]. Режим доступу: https://cabforum.org/wp-content/uploads/CAB-Forum-BR1.3.0.pdf. 2015. Дата доступу: 09.11.2020 р. – Загол. з екрану.

14. Dong Z. Phishing in smooth waters: The state of banking certificates in the us / Z. Dong, K. Kane, L. Camp // TPRC Conference Paper. – 2014. – P. 15-20.

15. Pan Y. Anomaly based web phishing page detection / Y. Pan, X. Ding // Proc of ACSAC. 06. 2006. – P. 381–392.

16. Sheng S. An Empirical Analysis of Phishing Blacklists [Електронний ресурс] / S. Sheng, L. F. Cranor, J. Hong, B. Wardman, G. Warner, C. Zhang // Proc. of CEAS. – 09 Jul. – 2009. Режим доступу: https://www.researchgate.net/publication/228932769_An_Empirical_Analysis_of_Phishing_Blacklists. Дата доступу: 10.11.2020 р. – Загол. з екрану.

17. Franco R. Better Website Identification and Extended Validation Certificates in IE7 and Other Browsers [Електронний ресурс] / R. Franco // IEBlog Режим доступу: https://link.springer.com/chapter/10.1007/978-3-540-88313-5_27. Дата доступу: 12.11.2020 р. – Загол. з екрану.

18. Blythe J. Targeted risk communication for computer security / J. Blythe, J. Camp, V. Garg // IUI. ACM. – 2011. – P. 295–298.

19. Callegati F. IEEE Xplore - Man-in-the-Middle Attack to the HTTPS Protocol / F. Callegati, W. Cerroni, M. Ramilli // IEEE Security & Privacy Magazine. – 2009. – P. 78–81.

20. Patange T. How to defend yourself against MITM or Man-in-the-middle attack [Електронний ресурс] / T. Patange. Режим доступу: https://hackerspace.kinja.com/how-to-defend-yourself-against-mitm-or-man-in-the-middl-1461796382. Дата доступу: 12.11.2020 р. – Загол. з екрану.

21. Moore T. Evil searching: Compromise and recompromise of Internet hosts for phishing / T. Moore, R. Clayton // Financial Cryptography and Data Security. Springer. 2009. – P. 256–272.

22. Domain Name System (DNS) Parameters. IANA. DNS RCODEs. [Електронний ресурс]. Режим доступу: https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml/ Дата доступу: 12.11.2020 р. – Загол. з екрану.

23. Флах П. Машинное обучение. – М.: ДМК Пресс, 2015. – 400 с.

24. James G. An Introduction to Statistical Learning. Springer / G. James, D. Witten, T. Hastie, R. Tibshirani. Springer Science+Business Media New York. – 2013. 426 p.

25. Alpaydin E. Introduction to Machine Learning(Fourth ed.) / E. Alpaydin // MIT. 2020. – P. 13–18.

26. Mohri M. Rostamizadeh, Afshin; Talwalkar, Ameet Foundations of Machine Learning / M. Mohri, A. Rostamizadeh, A. Talwalkar // USA, Massachusetts: MIT Press. – 2012. – P. 30-37.

27. Wang X. An algorithm for L1 nearest neighbor search via monotonic embedding / X. Wang, S. Dasgupta // Advances in Neural Information Processing Systems. 2016. – P. 983–991.

28. Adamov A. AI and Cybersecurity. Part 2 - Detecting Phishing URLs with ML [Електронний ресурс]. Режим доступу: https://www.nioguard.com/2020/03/ai-and-cybersecurity-part-2.html. Дата доступу: 12.11.2020 р. – Загол. з екрану.