Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра __Автоматизації проектування обчислювальної техніки_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

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

_____Метод виявлення програм-вимагачів з використанням машинного_____
_____навчання_____
(тема)

Виконала: студентка 2 курсу, групи СКСм-20-1

_____Ейхман Т.І._____
(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 __доц. Адамов О.С.__
(посада, прізвище, ініціали)

Допускається до захисту

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

2021 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

(шифр і назва)

Тип програми Освітньо-професійна

(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані комп'ютерні системи

(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«__»_____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Ейхман Тетяні Ігорівні

(прізвище, ім'я, по батькові)

1. Тема роботи (проекту)_ Метод виявлення програм-вимагачів з використанням машинного навчання

Ransomware Detection Method Using Machine Learning

затверджена наказом по університету від «_04_»__11__2021 р. № _1635 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20.12.2021
3. Вихідні дані до роботи (проекту)
Теоретичні відомості про програми-вимагачі
Статистичні дані у сфері кібербезпеки
Дані про машинне навчання
4. Перелік питань, що потрібно опрацювати у роботі
Аналіз предметної області та постановка задачі
Машинне навчання
Алгоритм кластеризації_k-means
Алгоритм кластеризації_affinity propagation

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) _14 слайдів_ _____

_____

_____

_____

_____

1. КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи (проекту)	Термін виконання етапів роботи	Примітка
1	Отримання теми проекту	04.11.2021 - 05.11.2021	
2	Аналіз предметної області	06.11.2021 - 07.11.2021	
3	Аналіз джерел з проблемної галузі	08.11.2021 - 11.11.2021	
4	Розробка підходу для реалізації поставленої	12.11.2021 - 16.11.2021	
5	Реалізація поставленої задачі	17.11.2021 - 26.11.2021	
6	Оформлення пояснювальної записки	29.11.2021 - 05.12.2021	
7	Оформлення графічного матеріалу	06.12.2021 - 10.12.2021	
8	Перевірка виконаного проекту керівником	13.12.2021 - 15.12.2021	
9	Захист проекту	16.12.2021 – 20.12.2021	

Студент _____ _____
(підпис)

Керівник роботи (проекту) ___ ____доц. Адамов О.С.___
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 66 с., 18 рис., 2 табл., 22 джерела.

ПРОГРАМА-ВИМАГАЧ, ХАКЕРСЬКА АТАКА, МАШИННЕ НАВЧАННЯ, КЛАСТЕРИЗАЦІЯ, AFFINITY PROPAGATION, КІБЕРБЕЗПЕКА.

Метою виконання кваліфікаційної роботи є практична реалізація одного з методів машинного навчання – алгоритму кластеризації affinity propagation, який можна використовувати для виявлення програм-вимагачів самостійно або як складову частину системи боротьби зі шкідливими програмами.

Для цього в роботі було зроблено аналіз програм-вимагачів, визначено їх характерні особливості та основні шляхи їх розповсюдження, а також досліджено машинне навчання як ефективний метод протидії і його роль у галузі кібербезпеки. Також було наведено існуючу реалізацію детектора аномалій, засновану на іншому алгоритмі.

Реалізований на практиці алгоритм було протестовано, а також проведено оцінку якості роботи з використанням спеціальних методів.

ABSTRACT

The explanatory note of the qualification work: 66 pages, 18 figures, 2 tables, 22 sources.

RANSOMWARE, HACKER ATTACK, MACHINE LEARNING, CLUSTERING, AFFINITY PROPAGATION, CYBERSECURITY.

The purpose of the qualification work is the practical implementation of one of the machine learning methods – the affinity propagation clustering algorithm, which can be used to detect ransomware programs independently or as an integral part of a malware control system.

For this purpose, the paper analyzed ransomware programs, identified their characteristic features and main ways of their distribution, and also investigated machine learning as an effective method of counteraction and its role in the field of cybersecurity. An existing implementation of an anomaly detector based on a different algorithm was also presented.

The algorithm implemented in practice was tested, and the quality of work was evaluated using special methods.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ОС – операційна система (англ., operating system, OS)

AI – штучний інтелект (англ., artificial intelligence)

CAPTCHA – повністю автоматизований публічний тест Тюринга для розрізнення комп'ютерів і людей (англ. Completely Automated Public Turing test to tell Computers and Humans Apart)

CSV – значення, розділені комою (англ. comma-separated values), іноді - значення, розділені символом (англ. character-separated values)

DL – глибинне навчання (англ., deep learning)

FN – хибно негативні (англ., false negatives)

FP – хибно позитивні (англ., false positives)

ML – машинне навчання (англ., machine learning)

RaaS - програма-вимагач як послуга (англ., Ransomware as a Service)

TN – істинно негативні (англ., true negatives)

TP – істинно позитивні (англ., true positives)

ВСТУП

З розвитком технологій збільшується кількість сфер нашого життя, що стають автоматизованими. На комп'ютерні системи з кожним роком покладається все більше і більше відповідальності. З ростом відповідальності і кількості важливих задач зростають ризики. Однією з нагальних проблем є збої у роботі комп’ютерних систем через дії зловмисників – хакерів, чи шкідливого програмного забезпечення.

Жертвами хакерських атак може будь хто, а збитки від них сягають мільярдів доларів. Від програм-вимагачів страждають виробництво, торговий сектор, урядові та освітні організації, заклади охорони здоров’я, комунальне господарство та ін. Нерідко такі атаки бувають цільові.

Зловмисники активно застосовують засоби автоматизації кібератак, використовуючи штучний інтелект для оптимізації своєї діяльності. Так, штучний інтелект використовують для проходження CAPTCHA, для злому паролів методом грубої сили, для пошуку вразливостей систем, для розсилки повідомлень з фішинговими посиланнями або з прикріпленими файлами, які містять в собі віруси або програми-вимагачі, тощо. Окремим феноменом виступає поява моделі обслуговування “програма-вимагач як послуга”.

Для того щоб впоратися зі зростаючим обсягом атак, необхідно розпочати активне впровадження технологій штучного інтелекту, машинного і глибокого навчання (ML/DL) для виявлення і прогнозування кіберзагроз, а також реагування на них в режимі реального часу.

В рамках виконання даної кваліфікаційної роботи проводиться аналіз програм-вимагачів, оцінка стану проблеми, визначення основ та напрямків застосування машинного навчання, дослідження існуючих методів виявлення програм-вимагачів з використанням машинного навчання, а також реалізація алгоритму кластеризації, який би допомагав в боротьбі з лавиною кібератак, детектуючи аномалії у хмарному середовищі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Загальні положення

Програма-вимагач (англ. ransomware) — це тип шкідливого програмного забезпечення, який надає злочинцям можливість віддалено заблокувати комп'ютер чи інший пристрій або зашифрувати дані на ньому. Після цього програма відображає спливаюче вікно з повідомленням, що доступ до пристрою чи даних буде повернуто після сплати викупу. В останні роки до простого блокування пристрою чи шифрування даних додались викрадення даних та загроза оприлюднення чи передачі таких даних третім особам.

Рисунок 1.1 — Приклад спливаючого вікна програми-вимагача

Слово ransomware походить з англійської від слів ransom – викуп і software – програмне забезпечення.

В якості викупу зловмисники частіше за все вимагають виплатити певну суму у якій-небудь криптовалюті. Зараз це найпопулярніший, але не єдиний спосіб. Декілька років тому платежі проводили через систему PayPal, Perfect Money, QIWI або Western Union. Відомі випадки коли викуп вимагали у формі подарункових сертифікатів Amazon або iTunes. Деякі програми-вимагачі змушували жертв робити онлайн покупки певних товарів або у конкретних магазинах. Наприклад, Archiveus Trojan надавав пароль для розшифровки даних за покупку товарів в онлайн аптеці.

Зазвичай, програма-вимагач потрапляє на комп'ютер під виглядом необхідних та зовсім нешкідливих файлів: документу в листі, що надійшов на електронну пошту, якогось файлу в особистій переписці і т.д. Так, в 2020 році одним з найпоширеніших шляхів зараження стала електронна пошта – на неї припадає більше половини всіх спроб зараження [7]. Але в останні роки, з поширенням цілеспрямованих атак (наприклад, при здійсненні атаки на конкретну компанію), зловмисники намагаються знайти вразливості системи – програмного забезпечення чи інфраструктури, через які можна потрапити в цільову мережу.

Програми-вимагачі можна розділити на категорії залежно від методу, за допомогою якого вони потрапляють на пристрій жертви:

– за допомогою хробаків (cryptoworm) – автономне зараження усіх доступних комп'ютерів;

– через модель обслуговування «програма-вимагач як послуга» - зазвичай в такому випадку шкідливі програми потрапляють на пристрій жертви через спам листи, комплект експлойтів або напіввручну;

– automated active adversary – зловмисники за допомогою засобів автоматичного сканування мережі Інтернет шукають IT-системи зі слабким захистом, а знайшовши таку систему, зловмисники розгортають в ній програму-вимагач [19].

1.2 Короткі історичні відомості про програми-вимагачі

Програми-вимагачі стали загальновідомими в середині 2000-х років. Але вважається, що перша атака програми-вимагача відбулася у 1989 році. AIDS Trojan або PC Cyborg була ініційована дослідником СНІДу, доктором наук Джозефом Поппом. Він роздав 20000 дискет дослідникам СНІДу (понад 90 країн світу), стверджуючи, що диски містять програму, яка аналізує ризик зараження СНІДом за допомогою анкетування. Однак дискета також містила шкідливу програму, яка активувалась лише після досягнення порогу в 90 запусків комп'ютера. На екрані користувача відображалось повідомлення з вимогою сплатити 189 доларів США та ще 378 доларів США за користування програмою [2].

Саме AIDS Trojan став натхненням для появи більш складних програм-вимагачів. Від простого симетричного шифрування вони перейшли до більш складних алгоритмів, а стрімке поширення мережі Інтернет спростило їх розповсюдження.

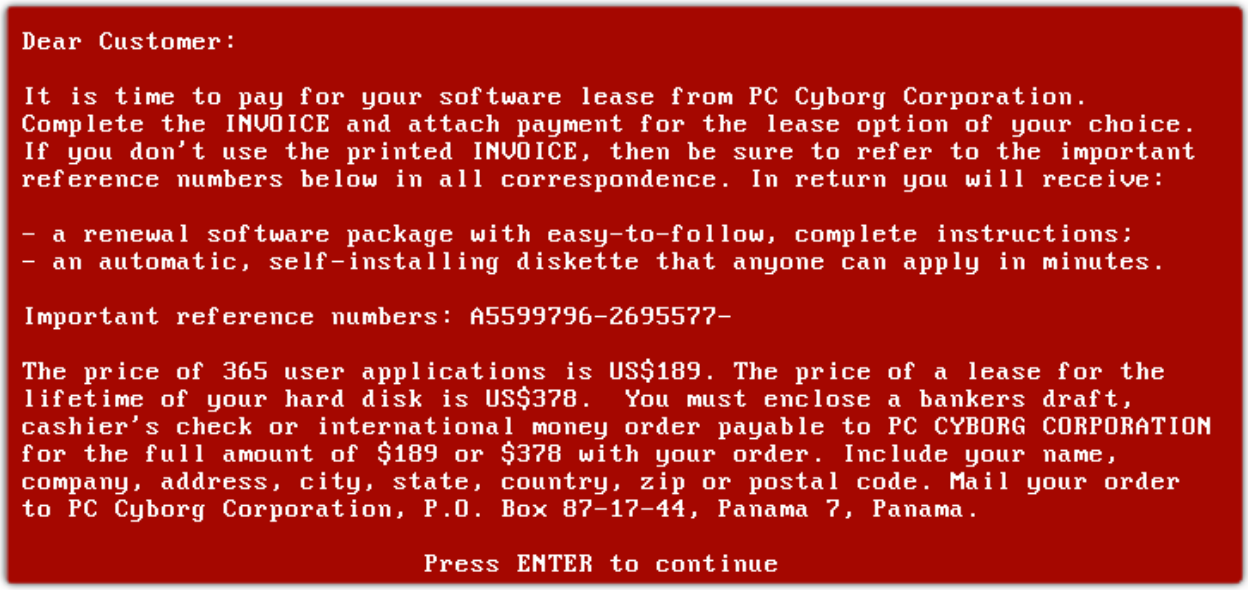

Рисунок 1.2 — Повідомлення з вимогою у AIDS Trojan

Новий етап в розвитку програм-вимагачів розпочався з появою вимагача Chimera у 2015 році. На поштову скриньку жертви надходив лист з посиланням, яке відкривало і одразу ж скачувало файл з хмарного сервісу Dropbox. Після завантаження на комп'ютер жертви, розпочинався процес шифрування файлів користувача – зображень, мультимедійних файлів, документів і т.д. Особливістю Chimera є те, що вона не тільки шифрувала файли, а й погрожувала опублікувати їх в мережі Інтернеті, якщо викуп не буде сплачено. Тепер, навіть якщо жертва має неушкоджені копії файлів і може їх відновити, страх оприлюднення таких файлів все одно спонукає її сплатити викуп.

Така тактика виявилась доволі успішною і її перейняли інші розробники програм-вимагачів.

1.3 Сучасні тенденції

Відповідно до даних веб-сервісу statista.com, протягом останніх 5 років кількість атак з використанням програм вимагачів значно зросла.

Рисунок 1.3 — Кількість атак з використанням програм-вимагачів у світі за 2014-2020 рр.

Середня вартість викупу за випадок теж зростає. Так, у 2018 році ця сума становила 4,3 тисячі доларів США, а у 2020 році сягнула 8,1 тисяч доларів США. Відповідно до даних Kaspersky, в 2020 році 56% жертв заплатили викуп щоб повернути свої дані, але навіть виплата не гарантує повного повернення всіх даних [6]. Для більшості компаній неможливість доступу до даних або зупинка роботи системи чи програмного забезпечення, навіть протягом декількох годин, спричиняє мільйонні збитки. Окрім цього, характерною тенденцією 2020 року стало поширення нової техніки –

викрадення даних перед їх шифруванням, а потім погроза оприлюднення цих даних, а також їх реальне оприлюднення у разі несплати викупу. Наприкінці року аж 15 груп програм-вимагачів використовували таку тактику [7].

Не останню роль у зростанні популярності та збільшенні збитків від атак з використанням програм-вимагачів зіграла поява моделі обслуговування “програма-вимагач як послуга” (Ransomware-as-a-Service (RaaS)). По аналогії зі стандартними (застосунок як послуга, інфраструктура як послуга чи платформа як послуга) моделями надання послуг хмарних сервісів, цей сервіс надає замовнику інструменти, необхідні для здійснення атаки. При цьому замовнику не потрібно витрачати час на розробку власних рішень чи мати спеціальні знання – достатньо просто знайти пропозицію, що підходить і заплатити. Такі пропозиції можна знайти в даркнеті. Пропозиції є для будь-яких цілей та на будь-який гаманець.

Зазвичай, клієнтам пропонується чотири основні види моделі оплати таких послуг:

– щомісячна передплата за фіксованою ціною;

– партнерські програми, які дещо сході на щомісячну підписку, але з відсотком прибутку (зазвичай 20-30%), що надходить оператору;

– одноразова плата за ліцензію без розподілу прибутку;

– чистий розподіл прибутку [4].

-= PACKAGES COMPARISON =-

	Package #TEST	Package #STANDARD	Package #PREMIUM	Package #ELITE
Subscription	1 Month	6 Months	12 Months	12 Months
Darknet C2 Dashboard	Yes	Yes	Yes	Yes
Features: Delayed Start, Delayed Encryption, Mutex, Task Manager/Registry Editor Disabler, UAC Bypass, Desktop Wallpaper Changer	Yes	Yes	Yes	Yes
Offline Encryption	No	Yes	Yes	Yes
Support	No	Yes	Yes	Yes
Real-Time Client Manager	No	Yes	Yes	Yes
Dropper	No	Buy	Yes	Yes
Clone	No	Buy	Buy	Yes
FUD+Obfuscator	Buy	Buy	Buy	Yes
Unkillable Process	No	Buy	Buy	Yes
FUD Stub #	1	1 (100% private FUD stub)	2 (100% private FUD stub)	**12 (100% private FUD stub)**
Price	120 USD	490 USD	900 USD	1900 USD

Рисунок 1.4 — Пропозиція RaaS з переліком наявних тарифних планів

Компанія BlackFog Enterprise займається розробкою та оптимізацією технології, яка перешкоджає вилученню даних з пристроїв кінцевих користувачів. Вона також збирає данні про випадки атак програм-вимагачів та кожного місяця створює звіти, в яких подається інформація про найбільші атаки, висвітлюються основні тенденції – порівняння з попередніми періодами, основні шляхи зараження та техніки вилучення , надається порівняльний аналіз подій за обраний звітний період. Звіти створюються для ознайомлення, аналітики та оцінки ризиків, пов’язаних з кіберзлочинністю.

За даними компанії BlackFog, останнім часом найбільш визначними програмами-вимагачами у світі є Conti та REvil. Трохи відстають від них CL0P та Darkside. Ці чотири програми вже не перший місяць поспіль залишаються найбільш впливовими.

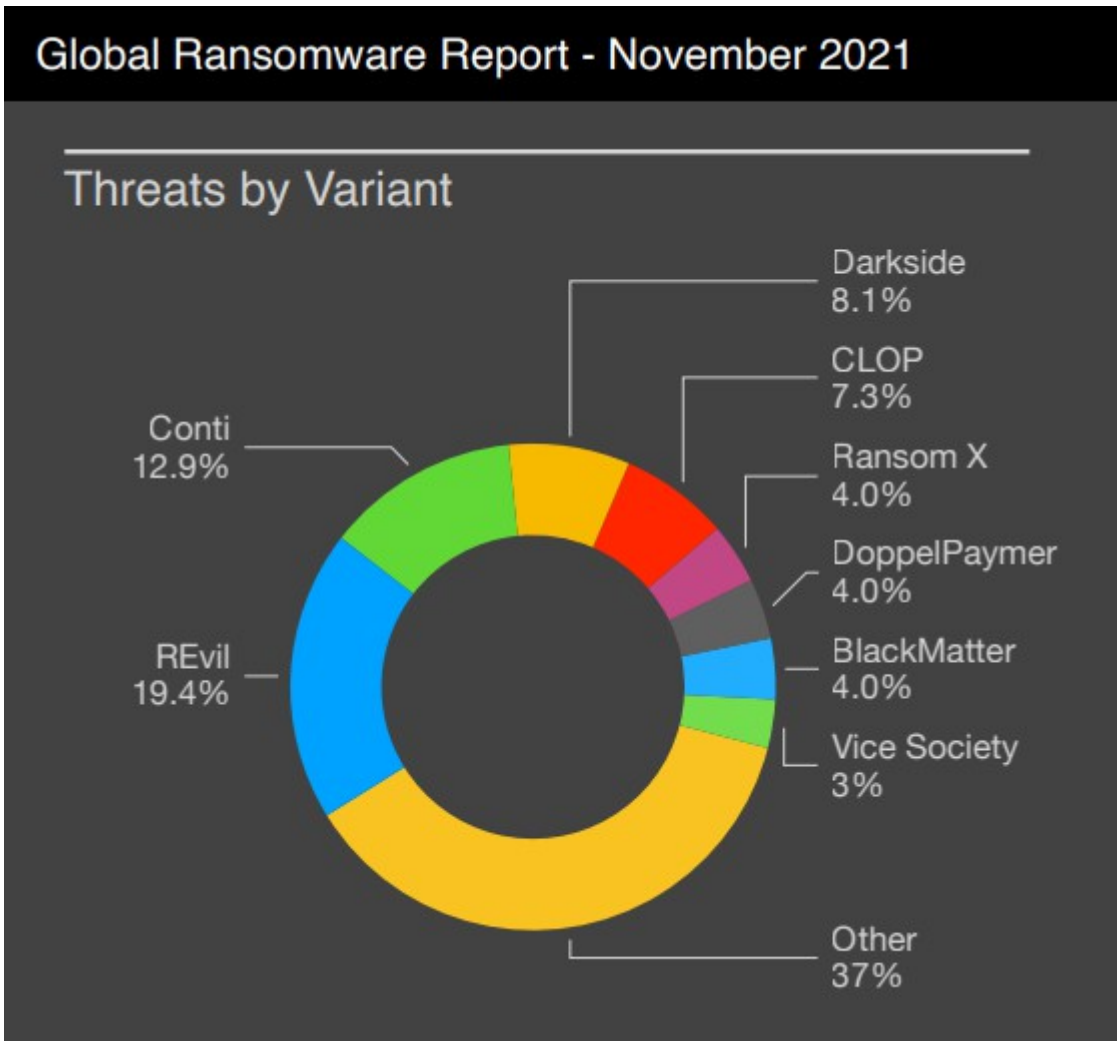

Рисунок 1.5 — Найвпливовіші програми-вимагачі відповідно даних компанії BlackFog Enterprise

Оригінальний Conti - це вірус, керований людиною. Він здатний автоматично проникати в систему; однак ним також може керувати людина-оператор. Програмне забезпечення використовує власну реалізацію AES-256, яка використовує до 32 окремих логічних потоків, що робить її набагато швидшою, ніж більшість програм-вимагачів.

REvil поширюється за допомогою експлойт-наборів, експлоїтів вразливостей та інсталяторів програмного забезпечення і працює як програма- вимагач як послуга.

CL0P зазвичай потрапляє в систему жертви через фішинг-кампанії

через zip-файли та файли docx із шкідливими макросами. Ця програма-вимагач перериває процеси системи, пов’язані із резервним копіюванням даних та контролем безпеки.

Darkside – керована людиною програма-вимагач, що використовує тактику подвійного вимагання. Атакує використовуючи фішингові електронні листи, що містять посилання на відомі довірені служби Shopify, Google Drive або DropBox. Вони містять файл .LNK, який зв’язується з інфраструктурою EMPIRE C2, для захоплення системи через бекдор на базі PowerShell, що містить функції реєстрації клавіш та захоплення екрана, а також можливість виконувати довільні команди.

Оцінка обсягів діяльності зловмисників, а також аналіз окремих атак дають змогу стверджувати, що зловмисники активно застосовують засоби автоматизації кібератак, в тому числі використовують технології штучного інтелекту і машинного навчання для їх вдосконалення і трансформації, пошуку вразливостей, а також для обходу відомих засобів захисту.

Завдяки успішній діяльності злочинців та широкому висвітленню в засобах масової інформації, організатори атак з використанням програм-вимагачів, систематично збільшували суми викупу, отримуючи більше грошей, здобуваючи відомість та репутацію.

Дослідники Касперського передбачають ще більший ріст атак з використанням програм-вимагачів у майбутньому. Отже, на сьогодні програми-вимагачі представляють собою серйозну проблему світового масштабу.

1.4 Особливості поведінки програм-вимагачів

Незважаючи на велику кількість різних програм-вимагачів, всі вони мають на меті зараження пристрою жертви та обмеження доступу до зараженого пристрою або до даних. Типовою поведінкою для програм вимагачів є підвищення прав доступу, зміна назв файлів та їх шифрування,

переміщення чи їх заміна і т. ін. Таким чином, різні програми-вимагачі, потрапивши на пристрій, виконують схожі дії, здійснюють атаку за схожими алгоритмами. Саме за цими рисами в поведінці програми-вимагачі можна детектувати. Окрім того є ряд дій, до яких вдаються злочинці і до яких нобхідно приділити більше уваги.

Перш за все варто зазначити про криптографічно підписаний код (наявність цифрового підпису) у шкідливих програм, в тому числі програм-вимагачів.

Зловмисники можуть спробувати звести до мінімуму можливість їх детектування за допомогою захисного програмного забезпечення, підписавши свою програму-вимагач сертифікатом Authenticode, який будь-хто може купити (або вкрасти). Підписані програми повинні забезпечувати гарантію того, що код не був змінений з моменту його випуску компанією-розробником програмного забезпечення, але це не дає ніяких гарантій того, що програмне забезпечення взагалі має бути запущено. На жаль, деякі засоби безпеки вважають наявність цифрового підпису гарантією безпечності та ознакою того, що програмі можна довіряти. Якщо програма-вимагач має належний цифровий підпис, засоби захисту від шкідливих програм або програм-вимагачів можуть аналізувати програму-вимагач не так ретельно, як інші файли, у яких відсутній дійсний цифровий підпис. Програмне забезпечення для захисту кінцевих точок може навіть віддати перевагу довіряти реально шкідливій програмі лише на основі того, що в неї сертифікат.

Нові сертифікати підпису коду цифрового підпису зазвичай коштують недорого. На додаток до обміну інформацією про оплату центр сертифікації вимагає, щоб особа або організація, які купують сертифікат, надали контактні дані. Центр сертифікації зв'язується з покупцем по електронній пошті і телефону, щоб підтвердити їх існування. Хоча це є перешкодою і ризиком для багатьох авторів шкідливих програм, все більше організованих злочинців докладають зусиль для забезпечення того, щоб їх шкідливе ПЗ

було підписано кодом з дійсним сертифікатом Authenticode, щоб запобігти виявлення і гарантувати успіх.

Емітенти сертифікатів швидко відкликають сертифікат підпису, коли отримують повідомлення про те, що сертифікат використовується при здійсненні кіберзлочину. Як тільки центр сертифікації скасує цифровий сертифікат, програмному забезпеченню, що здійснює захист на стороні кінцевих користувачів стане дуже легко знаходити і поміщати в карантин всі шкідливі програми, підписані відкликаним сертифікатом.

Ще однією неприємною можливістю шкідливих програм є багатопоточність.

Комп'ютери тепер оснащені одним або декількома багатоядерними процесорами з технологією одночасної багатопоточності (SMT) або Hyper-Threading (HT). Такі досягнення в області мікропроцесорного обладнання забезпечують величезні переваги в продуктивності для повсякденних бізнес-операцій, оскільки вони забезпечують паралельне виконання і більш ефективне використання системи для підвищення продуктивності.

Деякі програми-вимагачі спеціально розроблені для ефективного використання сучасного апаратного забезпечення ЦП і розпаралелюють окремі завдання, щоб забезпечити більш швидкий а, отже, більш відчутний вплив до того, як жертви виявлять, що вони піддаються атаці. Ці атаки можуть забезпечити більш високу пропускну здатність і меншу затримку, оскільки дані на більш швидкому носії можуть бути вилучені одним потоком, в той час як інший потік витягує дані з більш повільного носія, при цьому жоден з потоків не очікує завершення іншого.

Наприклад, програма-вимагач Sodinokibi має доступ до сховища (читання вихідного документа, запис зашифрованого документа), впровадження великих двійкових об'єктів і перейменування документів в декількох окремих потоках. А програми-вимагачі Locker Giga і Mega Cortex запускають підпроцеси для кожних декількох документів, щоб прискорити і ускладнити виявлення і зупинку їх атак.

Підвищення привілеїв (і бічне переміщення) хоча гарною практикою є надання облікових записів користувачів-і, отже, додатків, які вони запускають, - обмежених прав доступу, в умовах сучасних загроз це мало допомагає. Навіть якщо користувач, який увійшов до системи, має стандартні обмежені привілеї та дозволи, сьогоднішні вимагачі використовують експлойти для підвищення власних привілеїв та зловживання вкраденими обліковими даними адміністратора, щоб переконатися, що атака виконується за допомогою привілейованого облікового запису. Деякі приклади:

EternalBlue – це експлойт, розроблений Агентством національної безпеки США, до якого отримали доступ хакери і виуористали його потім в 2017 році в рамках Всесвітньої атаки вимагачів WannaCry. У поєднанні з технологією впровадження коду DoublePulsar експлойт дозволяє встановлювати шкідливе ПЗ з найвищими привілеями на кінцевій точці, незалежно від привілеїв, з якими увійшов в систему користувач.

Для придушення запиту контролю доступу користувачів (UAC), який зазвичай виникає під час підвищення привілеїв, деякі програми-вимагачі використовують експлойт обходу контролю доступу, який задає шлях до програми-вимагача в певному розділі реєстру.

Шкідлива програма, яка успішно використовує цю вразливість, може запускати довільний код в режимі ядра. Наприклад, шкідлива програма може встановлювати програми; переглядати, змінювати або видаляти дані; або створювати нові облікові записи з повними правами користувача, незалежно від привілеїв увійшов в систему користувача.

Як тільки зловмисники компрометують сервер або кінцеву точку, багато активних зловмисників зловживають існуючими засобами Windows, а також засобами безпеки з відкритим вихідним кодом або засобами тестування на проникнення. Наприклад, вони могли б використовувати TASKKILL.EXE для завершення процесів, що належать програмному забезпеченню endpoint protection. Вони можуть навіть впровадити такий інструмент, як Process Hacker, щоб в інтерактивному режимі зробити машину

своєю власною. Вони встановлять інструмент віддаленого доступу (RAT), такий як CobaltStrike, Meterpreter або PowerShell Empire, для забезпечення гнучкості і збереження сталості на своїй машині foothold до завершення місії.

Захопивши пристрій, багато зловмисників намагаються отримати облікові дані адміністратора локального домену, використовуючи інструмент після експлуатації, такий як Mimikatz. Зловмисники можуть додати новий обліковий запис адміністратора домену в Active Directory (AD), призначену тільки для них, на випадок, якщо справжні адміністратори домену змінять паролі.

Після цього зловмисники шукають – сервери або інші цінні об'єкти. Багато зловмисників витрачають час на інтерактивний пошук файлових серверів і тих, які використовуються для резервного копіювання даних, оскільки пошкоджена або пошкоджена резервна копія підвищує ймовірність того, що жертва погодиться заплатити викуп.

Для автоматичного розповсюдження програм-вимагачів на однорангові кінцеві точки і сервери зловмисники можуть використовувати надійну утиліту подвійного призначення, таку як PsExec від Microsoft SysInternals. Зловмисник створює скрипт, який перераховує зібрані цільові машини і об'єднує їх разом з PsExec, обліковим записом привілейованого домену і програмою-вимагачем. Цей скрипт послідовно копіює і запускає програму-вимагача на однорангові комп'ютери. Це займе менше години, залежно від кількості цільових машин. До того часу, коли жертва помічає, що відбувається, вже занадто пізно, так як ці атаки зазвичай відбуваються посеред ночі, коли IT-персонал спить.

В якості альтернативи PsExec також були помічені активні зловмисники, що використовують сценарій входу і виходу з системи через об'єкт групової політики (об'єкт групової політики) або зловживають інтерфейсом управління Windows (WMI) для масового поширення програм-вимагачів всередині мережі.

Були помічені зловмисники, які використовують вкрадені облікові дані

або використовують уразливості в рішеннях віддаленого моніторингу та управління (RMM), таких як kaseya, ScreenConnect і bomgar. Ці рішення RMM зазвичай використовуються постачальником керованих послуг (MSP), який віддалено керує IT-інфраструктурою своїх клієнтів та/або системами кінцевих користувачів. Рішення RMM зазвичай працюють з високими привілеями і після злому надають віддаленому зловмиснику доступ "з клавіатури", що призводить до небажаних ситуацій із заручниками даних. Володіючи таким доступом, зловмисники можуть легко поширювати програми-вимагачі в мережах віддалено, потенційно вражаючи відразу декількох клієнтів MSP.

Важливо включити багатофакторну аутентифікацію (MFA) в засобах централізованого управління і залишити включеної захист від несанкціонованого доступу в програмному забезпеченні захисту на стороны пристроїв коритсувача. Активні зловмисники спробують відключити локальні служби захисту за допомогою таких інструментів, як Process Hacker, але також спробують увійти на центральні портали безпеки, щоб відключити захист по мережі.

Деякі програми-вимагачі, потрапивши на пристрій жертви, шукають мережеві диски, віддалений доступ або файові сервери. У разі успіху вони. Використовуючи ресурси і привілей облікового запису пристрою, який першим заразили, програми-вимагачі завдають шкоди мережевим системам, а не пристрою кінцевого користувача. Щоб гарантувати, що жертви заплатять викуп, вимагачі намагатимуться зашифрувати якомога більше документів, іноді навіть ризикуючи або навмисно пошкоджуючи кінцеву точку. Ці документи можуть зберігатися на локальних стаціонарних і знімних дисках, а також на підключених віддалених загальних дисках. Програма-вимагач може навіть спочатку визначити пріоритетність певних дисків або розмірів документів, щоб забезпечити успіх, перш ніж її зловить програмне забезпечення для захисту кінцевих точок або помітять жертви. Наприклад, програма-вимагач може бути запрограмована на одночасне шифрування

декількох документів за допомогою декількох потоків, пріоритизацію невеликих документів або навіть спочатку атакувати документи на підключених віддалених загальних дисках.

У мережевих середовищах кінцеві точки - як в локальних, так і в віддалених офісах – зазвичай підключаються до парку файлових серверів. Ділові документи, включаючи креслення та інші файли, зберігаються на одному або декількох центральних файлових серверах, які доступні з кінцевих точок клієнта через одну або кілька загальних папок. Зазвичай кінцева точка може мати кілька зіставлень дисків з різними загальними ресурсами файлового сервера, що містять документи з окремих офісів, відділів, груп і проектів для поділу даних.

Програма-вимагач завдає найбільшої шкоди організації, коли спочатку шифрує ці підключені мережеві диски, оскільки вона негайно зачіпає більшість співробітників, незалежно від того, де вони географічно розташовані. Коли співробітники не можуть виконувати свою роботу, це руйнує всю організацію, чинячи тиск на керівництво, вимагаючи виплати викупу. І хоча більшість підприємств створюють резервні копії своїх даних, більшість резервних копій створюються періодично і не завжди є актуальними. Крім того, відновлення декількох серверів з резервної копії може зайняти багато днів, залежно від розміру даних і кількості порушених серверів. Фінансові наслідки таких затримок можуть швидко зрости.

Важливо відзначити, що самі файлові сервери часто не заражені вимагачами. Загроза зазвичай запускається на одній або декількох скомпрометованих кінцевих точках, використовуючи привілейований обліковий запис Користувача з дозволами рівня адміністратора для віддаленої атаки на документи. Таким чином, навіть якщо файловий сервер захищений антивірусним програмним забезпеченням, сама загроза фактично не запущена на сервері.

Що ще гірше, багато рішень для захисту кінцевих точок не перевіряють зміни документів на віддалених загальних дисках зберігання, навіть якщо

вони пропонують захист від вимагачів. Погана реалізація і часткове покриття накопичувачів-через проблеми з продуктивністю або технічної заборгованості - є основними причинами, через які більшість продуктів безпеки і навіть рішень для захисту від вимагачів не можуть виявити атаку вимагачів.

Microsoft Windows пропонує можливість виконувати поновлення ситеми, повернення її у попередній стан. Це досягається за допомогою служби тіньових томів (VSS). Windows зберігає копії документів (і системних файлів) попередніх версій, які були змінені – не тільки в одній попередній версії, але, можливо, у багатьох попередніх версіях. Ці файли попередньої версії легко відновлюються, навіть якщо поточна версія була успішно видалена. VSS вимагає, щоб томи, які він захищає, були відформатовані у файловій системі NTFS.

Як правило, вимагачі перейменовують документи під час своєї атаки, порушуючи зв'язок з файлом попередньої версії, що зберігається в VSS. Теоретично ви можете перейменувати зашифрований документ назад у вихідне ім'я файлу, відновивши таким чином зв'язок, а потім відновити оригінал з VSS, щоб усунути наслідки атаки вимагачів. На жаль, зловмисники зазвичай видаляють тіньові копії томів під час атаки за допомогою утиліти Windows, званої VSSADMIN.EXE . Ця утиліта надає інтерфейс командного рядка зі службою тіньового копіювання томів і вимагає підвищених прав адміністратора. Зловмисники зазвичай крадуть або вже володіють привілейованими обліковими даними або використовують експлойт для підвищення привілеїв.

Альтернативний спосіб видалення тіньових копій томів - за допомогою інструментарію управління Windows (WMI). Windows включає в себе утиліту командного рядка під назвою WMIC.EXE для доступу до WMI, який використовується багатьма додатками в якості інтерфейсу сценаріїв. Просте блокування або використання групової політики для обмеження WMIC.EXE швидше за все, це порушить деякі повсякденні завдання.

Поряд з видаленням резервних копій документів та інших файлів, що зберігаються в службі тіньових томів Windows, вимагачі також можуть спробувати перешкодити жертвам і Windows приступити до процедури відновлення для відновлення комп'ютера. Для досягнення цієї мети він зловживає командою Windows bcdedit.EXE, який дозволяє маніпулювати даними конфігурації завантаження Windows (BCD). Такі програми-вимагачі зазвичай встановлюють наступні параметри:

1.recoveryenabled немає-вимкніть функцію діагностики та відновлення Windows, щоб вона більше не запускалася автоматично після третьої невдалої завантаження вашого комп'ютера.

2. політика завантаження ігнорує всі помилки-ігнорує помилки при невдалому завантаженні, невдалому завершенні роботи або невдалої контрольної точки. Комп'ютер спробує завантажитися в звичайному режимі після виникнення помилки.

Програми-вимагачі, такі як LockerGoga і MegaCortex, зловживають CIPHER.EXE інструмент командного рядка від Microsoft, щоб переконатися, що жертви вимагачів не зможуть відновити видалені документи зі своїх накопичувачів. Цей інструмент є частиною Windows з Windows 2000 і призначений для управління законно зашифрованими даними з використанням файлової системи шифрування (EFS).

CIPHER.EXE також надає можливість безповоротно перезаписати (або" стерти") всі видалені дані на накопичувачі – як зловживають деякі програми-вимагачі. Ця функція призначена для підвищення безпеки, гарантуючи, що навіть зловмисник, який отримав повний фізичний контроль над комп'ютером Windows, не зможе відновити раніше видалені дані. В руках зловмисників-вимагачів, CIPHER.EXE посилює і без того велику проблему.

Замість того щоб виконувати шифрування на місці, деякі програми-вимагачі спочатку створюють зашифровану копію документа, який вони атакують, а потім видаляють вихідний файл. У порівнянні з шифруванням на місці, в цьому випадку зашифровані копії зберігаються в іншому місці на

накопичувачі. Засіб відновлення даних може відновити вихідні файли до тих пір, поки сектори видалених файлів, не позначені як вільні, ще не були перезаписані. Щоб перешкодити цьому способу відновлення, програма-вимагач Dharma встановлює розмір файлу кожного з атакованих документів рівним 0 байтам перед видаленням.

Інструменти відновлення даних, такі як Recuva, можуть швидко перераховувати видалені файли і їх розподіл в кластері, а також можуть витягувати (копіювати) ці файли з пошкодженого накопичувача. Він може зробити це завдяки таблиці основних файлів (MFT), в якій часто все ще зберігаються записи про видалені документи, включаючи фізичне розташування цих файлів на накопичувачі. У записах видалених документів встановлено біт "видалено", і засіб відновлення може зосередитися на цьому Біті, щоб допомогти жертвам повернути деякі дані. Але встановлення розміру файлу рівним 0 після видалення призведе до оновлення запису файлу в таблиці основних файлів (MFT), що ускладнить відновлення видалених документів засобами відновлення даних.

Windows підвищує продуктивність системи за рахунок використання кешування запису на стаціонарних, знімних і підключених до мережі пристроях зберігання даних. Але жертви програм-вимагачів, які знаходяться в паніці і фізично відключають свій комп'ютер, можуть привести до втрати або пошкодження даних (що все ж краще, ніж дозволити зашифрувати всі ваші документи і зберігати їх для викупу). Це також є причиною того, що до Windows 10 v1809 вам доводилося виконувати процедуру безпечного видалення обладнання, наприклад, для портативних USB-накопичувачів, на яких зберігаються дані і файли, такі як зовнішні жорсткі диски і флеш-накопичувачі.

Файлові сервери зазвичай встановлюються з RAID-сховищем, часто з кешем запису з батарейним живленням. На цих машинах очищення буфера кешу запису зазвичай відключається для подальшої оптимізації продуктивності, оскільки кеш запису з батарейним живленням дозволяє

машині очищати свій буфер в разі збою живлення без ризику втрати або пошкодження даних.

Ціною деякої продуктивності деякі програми-вимагачі (такі як WannaCry, GandCrab і BitPaymer) гарантують, що записані дані зашифрованих документів негайно зберігаються на накопичувачі. Вони роблять це або шляхом виклику функції FlushFileBuffers, або за допомогою наскрізного запису.

Деякі програми-вимагачі, такі як GandCrab і Sodinokibi, зловживають Windows PowerShell для запуску сценарію PowerShell з Інтернету, який налаштований на Автоматичний запуск програми-вимагача через кілька днів, через що здається, що атака відбувається з нізвідки. У цьому сценарії сама атака на Шифрування файлів виконується довіреною Windows POWERSHELL.EXE процес, що змушує програмне забезпечення endpoint protection вважати, що довірена програма змінює документи. Для досягнення тієї ж мети вимагачі, такі як Ryuk, можуть впроваджувати свій шкідливий код в довірений запущений процес, такий як SVCHOST.EXE . І програма-вимагач megacortex використовує Windows RUNDLL32.EXE додаток для шифрування документів з довіреного процесу.

Програми-вимагачі, такі як BitPaymer, можуть запускатися з альтернативного потоку даних NTFS (ADS) в спробі сховатися як від користувачів-жертв, так і від програмного забезпечення для захисту кінцевих точок.

З точки зору активності файлової системи, в залежності від того, як вона шифрує документи, програми-вимагачі можна розділити на дві групи: ті, що перезаписують файли і ті, які копіюють файли.

Перезапис файлів полягає у тому, що зашифровані файли зберігаються в тому самому місці, де вони знаходились до зараження.

Програма-вимагач зчитує оригінальний документ, перезаписує його зашифровану версію зверху оригінального документу, а потім змінює ім’я файла. Деякі програми-вимагачі спочатку змінюють ім’я файла, а потім

перезаписують його зашифровану версію.

В такому разі повернути файл звичайними утилітами для поновлення файлів не вдасться.

Копіювання файлів передбачає розміщення зашифрованих копій оригінальних файлів на вільному дисковому просторі.

В разі копіювання програма зчитує файл, створює його зашифровану версію з іншим розширенням чи назвою, а оригінальний файл видаляє.

В такому випадку можна спробувати поновити файл за допомогою програм для відновлення видалених файлів.

Деякі програми-вимагачі після шифрування зберігають ключ-дешифратор (Key blob) у вигляді файлу або декількох файлів на пристрої жертви (залежно від сімейства програм-вимагачів). Як правило, це великий двійковий об'єкт файл.

Програма-вимагач зазвичай перейменовує документи під час шифрування. Точний час, коли відбувається процес перейменування (це може статися до або після того, як програма-вимагач зашифрує файл), зазвичай залишається узгодженим в сімействі програм-вимагачів. А деякі програми-вимагачі (зокрема, RobbinHood) навіть змінюють повне ім'я файлу документа. У будь-якому випадку, перейменування файлів допомагає зловмиснику кількома способами:

1. Робить шкоду більш помітною для жертв, включаючи як адміністраторів, так і користувачів, оскільки змінюється значок типу файлу документа в Провіднику і додатках.

2. Запобігає подвійному шифруванню документів у разі повторного запуску програми-вимагача на тому ж комп'ютері. Якщо програма-вимагач змогла зашифрувати документ двічі, засіб дешифрування може виявитися нездатним відновити вихідний файл.

3. Запобігає шифруванню файлу іншими програмами-вимагачами, оскільки програми-вимагачі зазвичай шифрують документи тільки з певним розширенням імені файлу. Це, безумовно, ускладнило б розшифровку, навіть

якщо жертва заплатила обом зловмисникам, оскільки жертві потрібно було б знати, яка програма-вимагач запустилася першою.

4. Порушує зв'язок типу файлу з батьківським додатком файлу, а також забороняє користувачеві відновлювати свої файли з більш ранніх версій в службі тіньового копіювання томів Windows.

5. Ускладнює відновлення видалених документів за допомогою спеціального програмного забезпечення для відновлення. Програмне забезпечення для відновлення зазвичай сканує таблицю основних файлів (MFT) на наявність віддалених файлів з певним розширенням імені файлу. У ситуаціях, коли програма-вимагач зашифрувала копію документа замість того, щоб виконувати шифрування на місці, програмне забезпечення для відновлення повинно замість цього виконати повний аналіз поверхні диска, що займає набагато більше часу. Але оскільки файлова система буде повторно використовувати сектори, раніше зайняті віддаленими файлами, результати відновлення можуть бути обмеженими.

6. Деякі програми-вимагачі змінюють розширення імені файлу на адресу електронної пошти зловмисника, щоб жертви могли чітко бачити, з ким зв'язатися для отримання викупу і засоби дешифрування, хоча немає гарантії, що всі зашифровані документи можуть бути правильно розшифровані до їх початкового стану. Багато зловмисників-вимагачів стають поганими розробниками програмного забезпечення.

Нерідко злочинці окрім зараження хочуть додатково повідомити про свою присутність. В таких випадках вони можуть змінювати заставку на робочому столі жертви на своє повідомлення з погрозами, вимогою про сплату викупу, платіжною інформацію і т. ін. В такому випадку на жертву атаки створюється додатковий тиск, тому вона ймовірніше заплатить злочинцям.

2 МАШИННЕ НАВЧАННЯ ЯК ЗАСІБ ДЛЯ ВИЯВЛЕННЯ ПРОГРАМ-ВИМАГАЧІВ

2.1 Машинне навчання як інструмент

Машинне навчання (англ. machine learning) — це підгалузь штучного інтелекту в галузі інформатики, яка часто застосовує статистичні прийоми для надання комп'ютерам здатності «навчатися» (тобто, поступово покращувати продуктивність у певній задачі) з даних, без того, щоби бути програмованими явно [8].

Рисунок 2.1 — Машинне навчання у системі технологій штучного інтелекту

Машинне навчання може бути спонтанним і застосовуваним для навчання та встановлення базових характерів поведінки різних суб'єктів, а потім застосовуваним для пошуку виразних аномалій. В межах галузі аналізу даних машинне навчання є методом, який застосовують для винаходження

складних моделей та алгоритмів, які слугують прогнозуванню — в комерційному застосуванні це відоме як передбачувальна аналітика [8].

Машинне навчання розширює можливості з попередження та ліквідації кіберзагроз, допомагає знайти слабкі місця, а також зміцнити інфраструктуру. Досліджуючи матеріал вже існуючих програм-вимагачів, можна виявити певні характерні особливості атаки, схожі сценарії зараження. Наприклад, спосіб, яким вони потрапляють на пристрій жертв, вразливості, які для цього використовуються, процедуру шифрування, та ін. Аналізуючи всі ці дані, можна обачити певні шаблони в поведінці програм-вимагачів. Машинне навчання може допомогти як в задачах аналізу і сортування вже існуючих даних, так і в виявленні загроз в режимі реального часу на основі вже існуючих даних. Об’єми даних, які може обробити комп’ютер, в рази перевищують людські можливості.

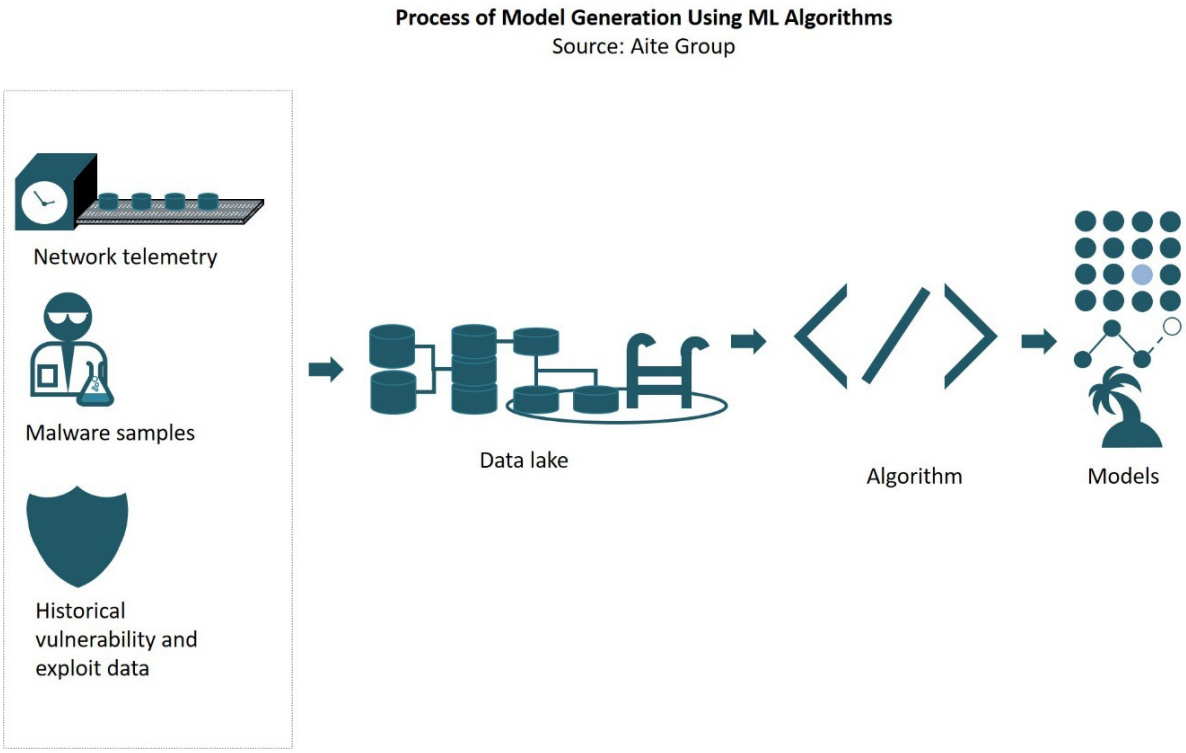

Рисунок 2.2 — Процес створення моделі з використанням алгоритмів машинного навчання

2.2 Основні задачі машинного навчання в кібербезпеці

Підмножина штучного інтелекту, машинне навчання використовує алгоритми, породжені попередніми наборами даних та статистичним аналізом, щоб зробити припущення про поведінку комп'ютера. На основі цього комп'ютер може коригувати свої дії - і навіть виконувати функції, для яких він явно не запрограмований. Таким чином, модель машинного навчання можна натренувати для ідентифікації потенційних загроз, в тому числі і загроз нульового дня.

Основним засобом захисту від кіберзагроз є антивірус. Але антивірусні рішення часто працюють на основі сигнатур - характерних ознак атаки або вірусу, що використовуються для їх виявлення: «синтаксичні» сигнатури, взяті безпосередньо з тіла або засновані на поведінці чи аномаліях. Для захисту від найновіших загроз необхідно постійно досліджувати нові віруси та атаки, постійно оновлювати сигнатури. Це може бути проблемою, наприклад, якщо визначення сигнатур вірусів відстає або через невдале оновлення, або через відсутність знань у постачальника антивірусів. Таким чином, якщо для атаки використовується нова, раніше не досліджена програма-вимагач, система захисту на основі сигнатур просто не матиме можливості зупинити таку атаку.

При використанні засобів штучного інтелекту, відбувається встановлення певної базової лінії поведінки для кінцевої точки шляхом повторюваного навчального процесу. Якщо трапляється щось незвичне, штучний інтелект може відмітити це і вжити заходів – надіслати сповіщення відповідальним особам або навіть повернути систему в безпечний стан після атаки [11]. Завдяки цьому забезпечується активний захист системи від загроз ще до оновлення баз даних сигнатур. За допомогою машинного навчання можна виявляти навіть незначні відхилення від типової поведінки, які людина може просто не помітити або не приділити їм достатньої уваги.

Отже, машинне навчання може бути корисним у напрямках:

– класифікації даних на основі заданих параметрів;

– кластеризації даних на основі їх подібності або аномальності, якщо вони не відповідають заданим параметрам;

– рекомендації підходів та рішень на основі минулих випадків;

– генерування варіантів дій, що можна застосувати до нововиявлених даних;

– прогнозування на основі вже існуючих наборів даних та результатів [10].

Конкретно, можна виділити ряд задач, які можна «доручити» моделі машинного навчання. Наприклад:

– фішинг та фільтрація спаму з класифікацією за заданими параметрами;

– аналіз кластеризованих даних для отримання додаткової інформації про методи атак;

– реагування на інциденти та управління ризиками за допомогою рекомендаційних методів;

– пентестування з генеруванням вхідних даних для перевірки на наявність вразливостей;

– профілактика та моделювання загроз за допомогою прогнозів на основі раніше зібраних даних.

2.3 Досвід запровадження машинного навчання в кібербезпеці

На сьогодні вже чимало компаній використовують машинне навчання у складі своєї системи захисту від кіберзагроз.

Microsoft використовує можливості машинного навчання для профілактичного захисту, виявлення порушень та загроз, автоматизованого розслідування та реагування в своєму продукті Windows Defender Advanced Threat Protection, що є складовою ОС Windows 10.

Варто згадати ще один продукт компанії Microsoft Azure Machine

Learning дає організаціям можливість будувати, навчати та управляти власними моделями ML, які в подальшому можна використовувати як частину системи безпеки компанії.

Компанія Blackberry, яка останнім часом перейшла на розробку програмного забезпечення та продуктів в сфері кібербезпеки. Серед спеціальностей компанії – рішення для кібербезпеки, які використовують штучний інтелект і машинне навчання для запобігання загрозам кібербезпеки та автоматизації можливостей реагування на загрози клієнтів. Нещодавно вона придбала фірму AI Cylance, що займається кібербезпекою з використанням машинного навчання. Cylance допомагає запобігти загрозам, перш ніж вони можуть заподіяти шкоду, прогнозуючи та захищаючи від безфайлових атак, зловмисного програмного забезпечення та загроз нульового дня. Технологія Cylance функціонує шляхом профілювання мільярдів зразків файлів, оцінки файлів на наявність загроз, диктування того, чи існує загроза чи ні, та введення в карантин заражених файлів.

Платформа AI Darktrace допомагає тисячам компаній у різних галузях промисловості виявляти та боротися з кіберзагрозами в режимі реального часу, аналізує мережеві дані для розрахунків та виявлення закономірностей. Технологія машинного навчання використовує дані, щоб допомогти організаціям виявити відхилення від типової поведінки та детектувати загрози.

Компанія Versive допомагає компаніям та організаціям визначати найважливіші загрози, тим самим надаючи командам можливість краще розподіляти свій час та пріоритезувати задачі залежно від рівня їх важливості і невідкладності. Це дозволяє економити час, який в іншому випадку може бути витрачений на ті сповіщення, які не потребують негайної уваги. Versive Security Engine (VSE) використовує штучний інтелект для відокремлення критичних ризиків від рутинної мережевої діяльності, виявляючи ланцюжки дій, що призводять до атак, та допомагаючи командам безпеки перевершити ці атаки.

SQRRL розробила платформу для полювання на кіберзагрози. Вона здійснює моніторинг мережі компанії, щоб знайти код, який може обійти існуючі заходи безпеки. Продукт використовує машинне навчання, щоб створити карту загроз, яка діє як візуальне представлення комп'ютерної мережі та показує слабкі місця, через які можуть проникнути загрози.

Компанія Chronicle зі своїм продуктом Backstory допомагає аналізувати великі обсяги даних у сфері кібербезпеки (наприклад, активність внутрішньої мережі, відомі невдалі домени та підозри на зловмисне програмне забезпечення) і використовує машинне навчання, щоб перетворити їх у більш легкі для сприйняття дані.

Штучний інтелект платформи Tessian запобігає порушенням, фішингу та втраті даних від шкідливих електронних листів. Компанія створює гнучкі фільтри електронної пошти, які усувають шкідливу та підозрілу активність як у вхідній, так і в вихідній електронній пошті. Платформа також оснащена інформаційною панеллю, що дозволяє моніторити стан інфраструктури в режимі реального часу, завдяки чому команди безпеки мають усі актуальні дані і можуть вчасно реагувати на небезпеку.

Компанія Blackfog розробила технологію ADX (Anti Data Exfiltration) – захисту від ексфільтрації (витоку) даних. Витік даних є серйозною загрозою для організацій, а погроза оприлюднення даних є одним з основних важелів, за допомогою яких злочинці отримують викуп. Запобігання витоку даних залишається слабким місцем для багатьох організацій, незважаючи на наявність складного набору інструментів кібербезпеки, включаючи рішення щодо запобігання втраті даних DLP (Data Leakage Protection). Просте проникнення в мережу або пристрій не є успішною кібератакою. Атака буде успішною тільки якщо будуть вкрадені конфіденційні дані. Без ексфільтрації даних не відбуватиметься втрати, витоку даних, а також вимагання викупу. ADX працює, досліджуючи вихідні дані на кінцевих пристроях. Замість порівняння трафіку зі словником сигнатур атак рішення ADX використовують поведінкову аналітику для виявлення незвичайної

поведінки, орієнтованої на користувача. Технологія відстежує і блокує дії, типові для програм-вимагачів, а також обмежує можливості користувачів, (включаючи привілейованих користувачів і адміністраторів) відправляти конфіденційні дані за межі мережі.

Human Security застосовує машинне навчання та штучний інтелект для прогнозування порушень безпеки та зупинки шкідливої роботи ботів. Програмний продукт прогнозує наявність автоматизованого трафіку ботів і автоматично блокує трафік, перш ніж він стане небезпечним для фінансових транзакцій або внутрішньої інфраструктури.

Ці компанії – лише невеличка частина тих, хто на даний час активно інтегрує технології машинного навчання в системи захисту від кіберзагроз.

2.4 Основні алгоритми машинного навчання

Машинне навчання традиційно поділяють на три основні групи: навчання з вчителем (supervised learning), навчання без вчителя (unsupervised learning), навчання з підкріпленням (reinforcement learning).

Метод навчання без вчителя заснований на тому, що система обробляє велику кількість нерозмічених даних, шукаючи взаємозв’язки і закономірності в них, упорядкувати чи класифікувати отримані дані. Система навчається без будь-якого втручання ззовні. Навчання без вчителя представлене алгоритмами, які вирішують задачі кластеризації, зменшення розмірності і пошуку правил.

При навчанні з вчителем на вхід системи подаються розмічені дані, тобто бажаний результат вже відомий і необхідно лише встановити залежність між вхідними та вихідними даними. Вчителем в цьому випадку зазвичай є людина, яка попередньо підготувала дані та перевіряє результат. Прикладами алгоритмів навчання з вчителем є метод опорних векторів, регресія та класифікація.

Навчання з підкріпленням вирізняється тим, що не має фактичного

набору вхідних даних, а навчання відбувається через дію. На кожну дію реагує середовище, надаючи зворотній зв’язок. Він може бути як позитивним, так і негативним. Система аналізує отриману від середовища інформацію і залежно від цієї інформації коригує свої дії. Наприклад, алгоритм Q-навчання, метод часових різниць.

Рисунок 2.3 — Види методів та приклади алгоритмів машинного навчання

На практиці для вирішення складних задач можуть використовувати комбінацію вищевказаних методів або проміжні алгоритми. Також на даний час ведеться робота над іншими методами навчання. Наприклад, навчання з частковим залученням вчителя використовується у випадках, коли частина даних для тренування нерозмічена, поєднуючи в собі ознаки навчання з вчителем та навчання без вчителя. Мета-навчання передбачає створення моделі, яка залежно від поставленої задачі та набору даних сама обирає доречний алгоритм і налаштовує необхідні параметри обраного алгоритму для вирішення задачі. Також такий метод машинного навчання може застосовувати комплекс алгоритмів для корекції чи мінімізаціїї похибок, які зазвичай виникають у алгоритмів, якщо їх використовувати окремо.

Для отримання якісної моделі машинного навчання необхідні дані, властивості чи характеристики, які є ключовими для результату, а також алгоритм.

Вибір алгоритму залежить від поставлених задач, об’єму та якості вхідних даних, а також відповідно до вимог до самої роботи алгоритму – швидкості, якості, необхідних ресурсів.

На практиці здебільшого знайти вибірку великого об’єму з нерозміченими даними простіше, ніж з розміченими. А розмітка даних – дуже довготривалий та ресурсномісткий процес. Перевагою навчання без

вчителя є саме можливість використання нерозмічених даних. Але методи навчання без вчителя вимагають більші об’єми даних та часу для навчання, а також потребують більше обчислювальних потужностей та використовують при роботі більше пам’яті. Також такі алгоритми дуже чутливі до викидів та аномалій в даних, які подаються на вхід.

Великим класом задач, які вирішують алгоритмами є задачі кластеризації. Кластерний аналіз – задача розбиття заданої вибірки об'єктів (ситуацій) на підмножини, які називаються кластерами, так, щоб кожен кластер складався з схожих об'єктів, а об'єкти різних кластерів істотно відрізнялися. Завдання кластеризації відноситься до статистичної обробки. Відповідний алгоритм кластеризації та вибору параметрів (включаючи такі параметри, як функція відстані, порогове значення щільності або кількість очікуваних кластерів) залежать від конкретного набору даних та мети використання результатів [17]. Кількість кластерів заздалегідь не відома. Деякі алгоритми потребують вказання кількості кластерів на початку роботи алгоритму і мають для цього спеціальні формули чи порядок розрахунку кількості кластерів; інші визначають кількість кластерів самостійно в процесі роботи.

Існує велика кількість різних алгоритмів кластеризації. Наприклад:

- k-means;
- ієрархічна кластеризація;
- EM-алгоритм;
- DBSCAN;
- BIRCH;
- Affinity Propagation;
- FOREL.

В результаті кластеризації алгоритм видає групи об’єктів, об’єднані за якоюсь ключовою характеристикою чи декількома характеристиками. Важливою є стійкість знайденого кластерного рішення. Перевірка стійкості кластеризації зводиться до перевірки її достовірності – стійка типологія

повинна зберігатися при зміні методів кластеризації. Результати ієрархічного кластерного аналізу можна перевіряти ітеративним кластерним аналізом методом k- means. Якщо при порівнянні групи збігаються більше, ніж на 70% (понад 2/3 збігів), то кластерне рішення приймається [17].

Не існує універсального алгоритму кластеризації, який би однаково гарно впорався з поставленою задачею незалежно від наявної вибірки даних. Під кожну задачу необхідно підбирати свій алгоритм. За допомогою модуля sklearn.cluster можна виконати кластеризацію нерозмічених даних. Результати роботи, а також затрачений час для кожного з реалізованих алгоритмів на різних вибірках даних представлені в документації модуля, можуть допомогти з вибором конкретного алгоритму для вирішення поставленої задачі.

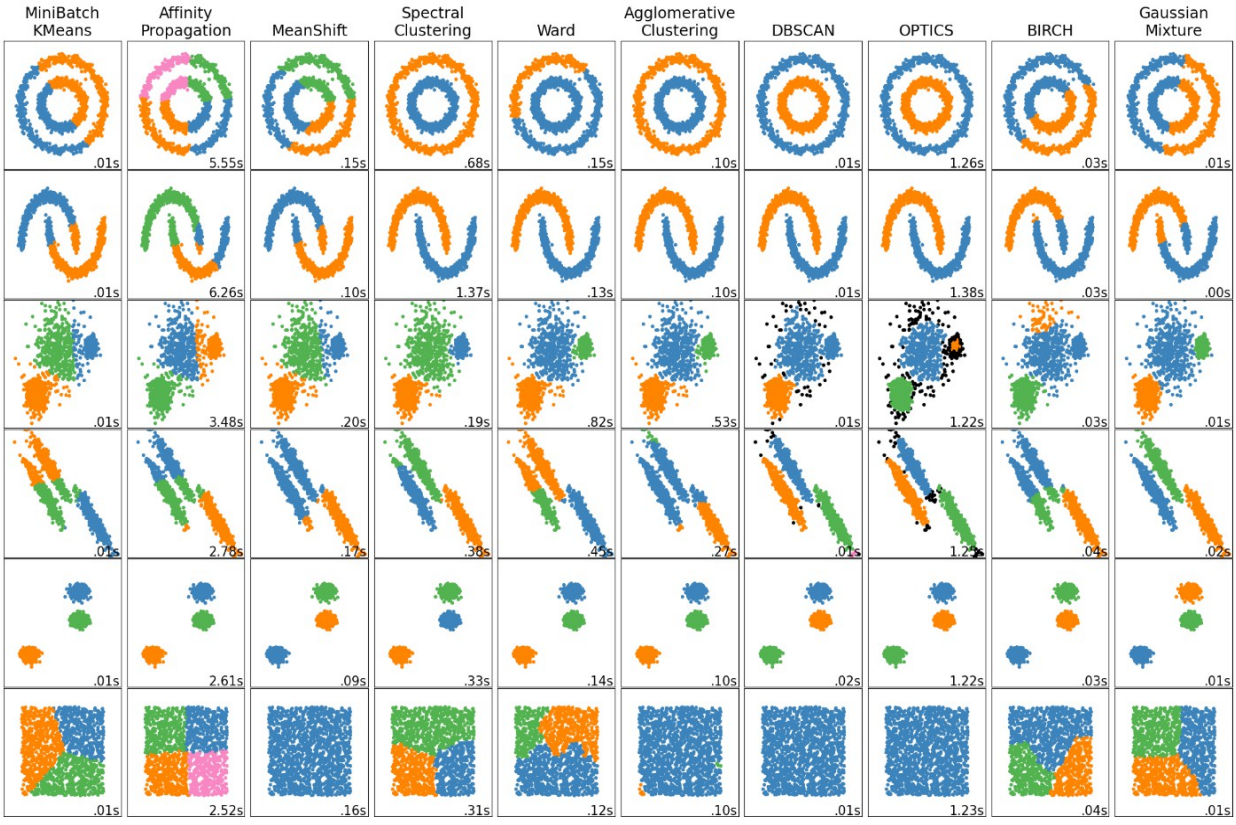

Рисунок 2.4 — Порівняння результатів роботи алгоритмів кластеризації, реалізованих в scikit-learn

Як можна побачити, початкові дані представлені у різних варіаціях, між ними існують різні залежності і в кожному випадку лише декілька алгоритмів добре справляються з поставленою задачею. Окрім того, один і той же алгоритм може відмінно працювати на одній вибірці даних і зовсім не впоратися з іншою.

Ще однією із проблем кластеризації, як методу машинного навчання без вчителя є те, що навіть в зовсім випадкових і ніяким чином не пов’язаних даних алгоритм буде формувати групи. Через це перш ніж застосовувати алгоритм кластеризації, необхідно зробити оцінку вибірки даних на придатність до кластеризації, наявність в вибірці кластерів, що мають сенс (clustering tendency assessment).

Для оцінки якості роботи алгоритму кластеризації існує ряд методів. Методи можна поділити на внутрішні (internal) та зовнішні (external). Іноді також виділяють відносні (relative).

Зовнішні методи оцінки якості кластеризації порівнюють результати кластеризації з деякою еталонною кластеризацією. Для деяких наборів даних вже відомо справжнє розбиття на кластери або можна провести його розрахунок, або вже є інформація про кількість кластерів. В такому випадку можна просто порівняти отриману кластеризацію з існуючим істинним розбиттям. Тобто оцінка роботи відносно інформації ззовні. Прикладами зовнішніх методів оцінки є Індекс Rand та Індекс Adjusted Rand, Ентропія, Чистота, F-міра і т. д.

Внутрішні методи оцінки якості кластеризації засновані на оцінці якості по відношенню до деяких наших уявлень про те, якою повинна бути хороша кластеризація. Оцінюється тільки сам кластер, об’єкти, які віднесені до нього. Наприклад, Індекс Данна та Узагальнений індекс Данна, Силует, Індекс Девіса-Болдуіна і т. ін.

Відносними називають методи, які порівнюють результати роботи одного й того ж алгоритму при зміні якихось вхідних параметрів. Вважається, що більшість кластерів все ж таки не здатні якісно визначити

справжню кількість кластерів в поданій вибірці даних. Частіше цей параметр або задається як вхідний, або ж визначається в результаті декількох запусків алгоритму кластеризації. Відносний метод оцінки якості роботи алгоритму кластеризації як раз полягає в тому, щоб запускати алгоритм декілька разів, трохи змінюючи параметри, а потім порівняти отримані результати кластеризації і на основі цих даних зробити висновок – при яких параметрах алгоритм кластеризував подану вибірку даних найкраще.

3 ОГЛЯД ІСНУЮЧОЇ РЕАЛІЗАЦІЇ

3.1 Засоби реалізації алгоритмів кластеризації

Детектування кіберзагроз, зокрема атак з використанням програм-вимагачів, може бути засновано на методі сигнатур або на методі аномалій. За допомогою машинного навчання можна навчити систему визначати аномальну поведінку користувача чи програми або аномальний трафік. Аналізуючи таку інформацію, можна побачити певні шаблони в поведінці зловмисних програм і в подальшому відрізняти їх від безпечних програм. Алгоритми кластеризації, обробляючи подані дані, виявляють закономірності, схожості ознак об’єктів вибірки даних, завдяки чому можуть виділити кластери аномальних об’єктів.

Для розробки програм, заснованих на алгоритмах машинного навчання використовується ряд мов, найпопулярнішими з яких є Python, R та C++. Перевагою мови Python є те, що вона доволі легка для розуміння і при цьому в ній є багато готових бібліотек для вирішення великої кількості різноманітних задач, наприклад, обробки даних та зображень, візуалізації, статистичних обчислень та ін.

Однією з найпопулярніших бібліотек мови Python для машинного навчання є sklearn. Проект scikit-learn надає прості та ефективні інструменти для аналізу даних, має відкритий вихідний код, активно розвивається і доступний усім користувачам абсолютно безкоштовно. Працює разом з бібліотеками NumPy, SciPy, а також matplotlib. За допомогою цієї бібліотеки можна з легкістю реалізовувати алгоритми класифікації та регресії, кластеризації тощо.

Для обробки та аналізу набору даних зазвичай використовують бібліотеки pandas і numpy. Бібліотека numpy надає можливість працювати з багатовимірними масивами та матрицями, а також різноманітними

математичними функціями. Бібліотека pandas призначена для обробки та аналізу даних. Вона побудована на основі DataFrame – двовимірної структури, що подібна таблиці і надає велику кількість методів для роботи з даними. Дані в цих структурах можуть бути різних типів. Бібліотека pandas також використовує Series – одновимірний індексований масив даних фіксованого типу. Окрім того, бібліотека pandas дозволяє працювати з файлами форматів csv, .tsv, .xlsx, sql тощо.

Для двовимірної та тривимірної візуалізації даних – побудови двовимірних та тривимірних графіків, лінійних, секторних, стовпчастих чи спектральних діаграм, гістограм, діаграм розсіювання і т.д. використовують бібліотеку matplotlib. Вона дозволяє створювати статичні, анімовані та інтерактивні візуалізації.

Ці бібліотеки є базовими для вирішення багатьох задач машинного навчання.

3.2 Інструменти та основні функції існуючої реалізації

В розглянутому прикладі існуючої реалізації алгоритму наведено варіант кластеризації методом k-means для аналізу нерозмічених даних, що використовується для створення алгоритму детектування аномалій в хмарних сервісах [15].

Алгоритм реалізовано мовою Python. Для обробки та аналізу набору даних використано бібліотеки pandas і numpy, серед бібліотек для машинного навчання обрано sklearn, а для візуалізації обрано бібліотеку matplotlib.

Відповідно до вимог обраного алгоритму кластеризації, перед його запуском необхідно визначити кількість кластерів, які очікується отримати на виході. Після цього випадковим чином обираються об’єкти вибірки даних, які вважаються центрами кластерів. Кожен з цих обраних центрів стає центром того кластеру, відстань до якого найкоротша. В утворених кластерах розраховується нові центри. Суть методу полягає в тому, що на кожній

ітерації переобчислюють центр мас для кожного кластера, отриманого на попередньому кроці, потім об'єкти знову розбиваються на кластери відповідно до того, який з нових центрів виявився ближчим згідно обраної метрики. Алгоритм завершується, коли на якійсь ітерації не відбувається зміни відстані всередині кластера [16].

У алгоритму k-means є модифікація k-means++, яка регламентує процес вибору початкових центів кластерів, що робить його більш стабільним порівняно зі звичайним алгоритмом.

Алгоритм кластеризації k-means доволі популярний через простоту реалізації і швидкість роботи. Але необхідність задавати кількість кластерів явно на початку роботи алгоритму дещо ускладнює роботу. Для визначення кількості кластерів можна використати скористатися простим перебором – почати з 2 чи 3 кластерів і після завершення роботи алгоритму кожного разу оцінювати якість роботи або скористатися формулою. Також, на коректність роботи алгоритму впливають центри кластерів, які довільно обираються на початку роботи алгоритму.

Перед початком роботи алгоритму необхідно підготувати вибірку даних для кластеризації. В наведеному прикладі реалізації за допомогою фреймворку NioCrytoSim було змодельовано атаки на хмарну інфраструктуру Google Cloud Platform (GCP), після чого з GCP було завантажено журнал подій.

Журнал подій для кожного об’єкта – події, містить одинадцять ознак, наприклад назву об’єкту, тип об’єкту, власника, назву події, опис події, дату, IP-адресу, з якої здійснювався доступ та ін. З усіх ознак було обрано шість ключових: назва об’єкту, користувач, дата, назва події, тип об’єкту, IP-адреса. Також було додано сьому ознаку – ентропія.

Отримані дані в форматі csv за допомогою методу read_csv з бібліотеки pandas було завантажено в роботу. В файлі формату csv дані розділяються комами та символом переходу на новий рядок. За допомогою функції LabelEncoder з бібліотеки sklearn усі нечислові дані було конвертовано в

числові.

Після завантаження і форматування даних починається безпосередньо робота алгоритму. Спочатку довільним чином обирається декілька точок, які представляють собою центри кластерів. Відбувається розподіл даних, потім робиться переобчислення центрів кожного кластеру та оновлення даних згідно розрахунків.

В представленій реалізації початкові значення - координати центрів кластерів обираються випадковим чином.

Лістинг 3.1 – Задання початкових центрів кластерів

```
centroids = {
    i + 1: [np.random.randint(0, 30), np.random.randint(0, 30)]
    for i in range(k)
}
```

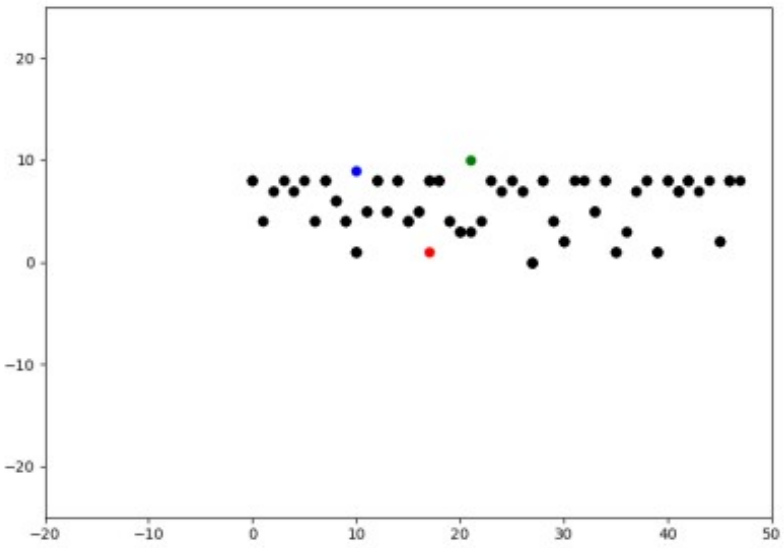

Рисунок 3.1 — Ініціалізація всіх об’єктів вибірки та початкових значень центрів кластерів

Кожен об’єкт вибірки даних відноситься до того кластеру, центр якого є найближчим для нього. Наближеність об’єкта до центра кластеру розраховується за допомогою метрик. Існують різні формули розрахунку

міри наближеності об’єктів один до одного: метрика Манхеттен, max-метрика, Евклідова метрика. Вибір залежить від вподобань того, хто реалізує алгоритм. В розглянутій реалізації алгоритму використовується саме Евклідова відстань. Процес розрахунку метрики та розподілення даних між кластерами наведено в лістингу 3.2.

Лістинг 3.2 – Функція призначення даних

```
def assignment(df, centroids):
    for i in centroids.keys():
        # sqrt((x1 - x2)^2 - (y1 - y2)^2)
        df['distance_from_{}'.format(i)] = (
            np.sqrt(
                (df['Item name'] - centroids[i][0]) ** 2
                + (df['Item Type'] - centroids[i][1]) ** 2
            )
        )

    centroid_distance_cols = ['distance_from_{}'.format(i) for i in
centroids.keys()]
    df['closest'] = df.loc[:, centroid_distance_cols].idxmin(axis=1)
    df['closest'] = df['closest'].map(lambda x:
int(x.lstrip('distance_from_')))
    df['color'] = df['closest'].map(lambda x: colmap[x])
    return df
```

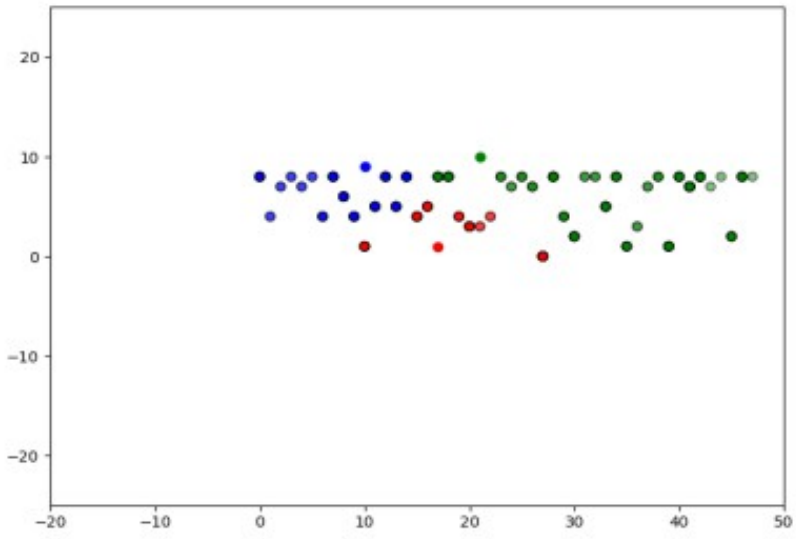

Рисунок 3.2 — Розподілення всіх об’єктів вибірки даних між заданими центрами кластерів

Після розподілу даних на кластери відбувається оновлення центрів кластерів.

Лістинг 3.3 – Функція оновлення даних після обчислень

```
def update(k):
    for i in centroids.keys():
        centroids[i][0] = np.mean(df[df['closest'] == i]['Item name'])
        centroids[i][1] = np.mean(df[df['closest'] == i]['Item Type'])
    return k
```

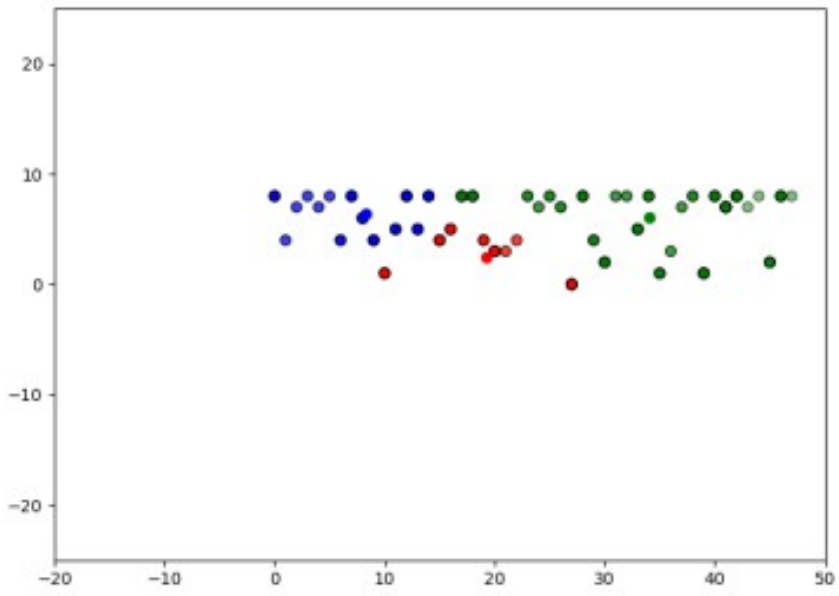

Рисунок 3.3 — Оновлення центрів кластерів

Після оновлення центрів кластерів повторюється перерозподіл усіх об’єктів вибірки між кластерами відповідно до нових значень центрів.

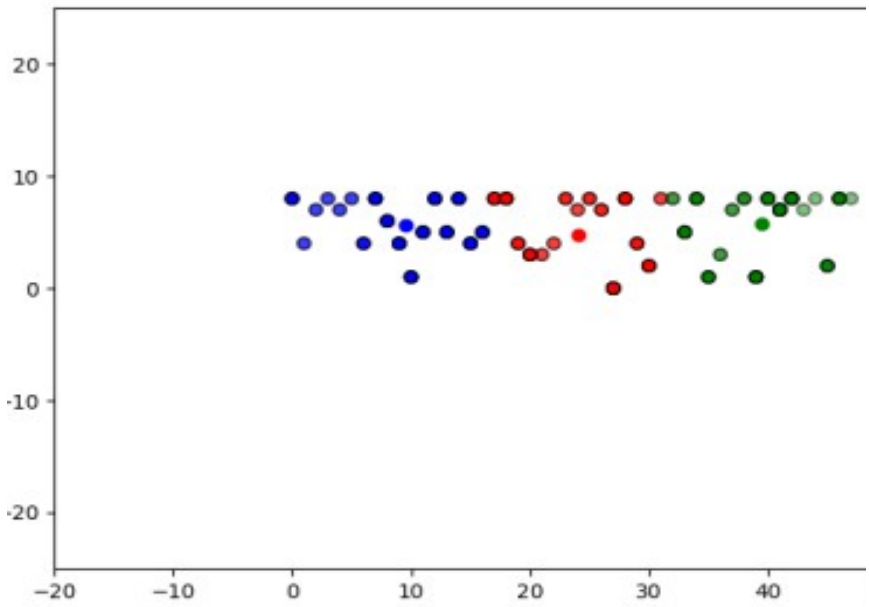

Рисунок 3.4 — Перерозподіл даних між кластерами після оновлення значень центрів

Повторюваність обчислень задається циклом while, вихід з якого здійснюється оператором break тоді, коли на ітерації більше не відбудеться змін відстані всередині кластера.

Лістинг 3.4 – Цикл, що контролює кількість обчислень

```
while True:
    closest_centroids = df['closest'].copy(deep=True)
    centroids = update(centroids)
    df = assignment(df, centroids)
    if closest_centroids.equals(df['closest']):
        break
```

Виведення координатної площини графіка, а також задання його параметрів та способу відображення даних – точкова діаграма (scatter), здійснюється як показано в лістингу 3.5.

Лістинг 3.5 – Створення графіку

```
fig = plt.figure(figsize=(10, 10))
```

```
      plt.scatter(df['Item name'], df['Item Type'], color=df['color'],
alpha=0.5, edgecolor='k')
      for i in centroids.keys():
          plt.scatter(*centroids[i], c=colmap[i])
      plt.xlim(-25, 50)
      plt.ylim(-20, 40)
      plt.show()
```

В результаті роботи методу можна побачити, як дані поступово, з кожною ітерацією алгоритму діляться на кластери, а також як центри кластерів оновилися відносно початково заданих.

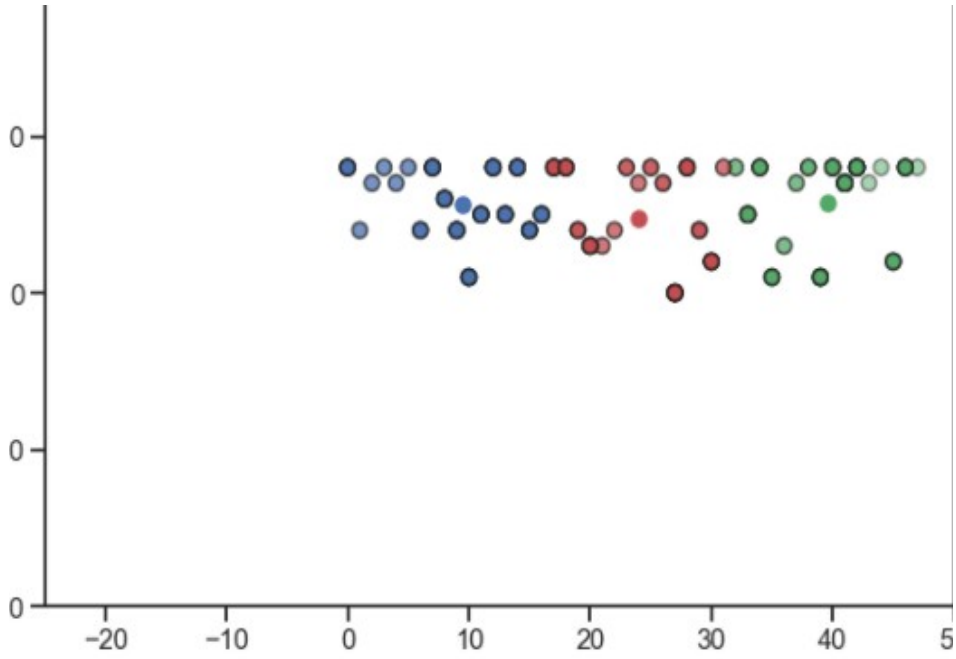

Рисунок 3.5 — Результат роботи алгоритму

Після закінчення роботи алгоритму центри кластерів опинилися в таких точках:

1: [24.0, 4.767123287671233],

2: [39.50632911392405, 5.784810126582278],

3: [9.47191011235955, 5.674157303370786]

3.3 Оцінка результатів роботи алгоритму

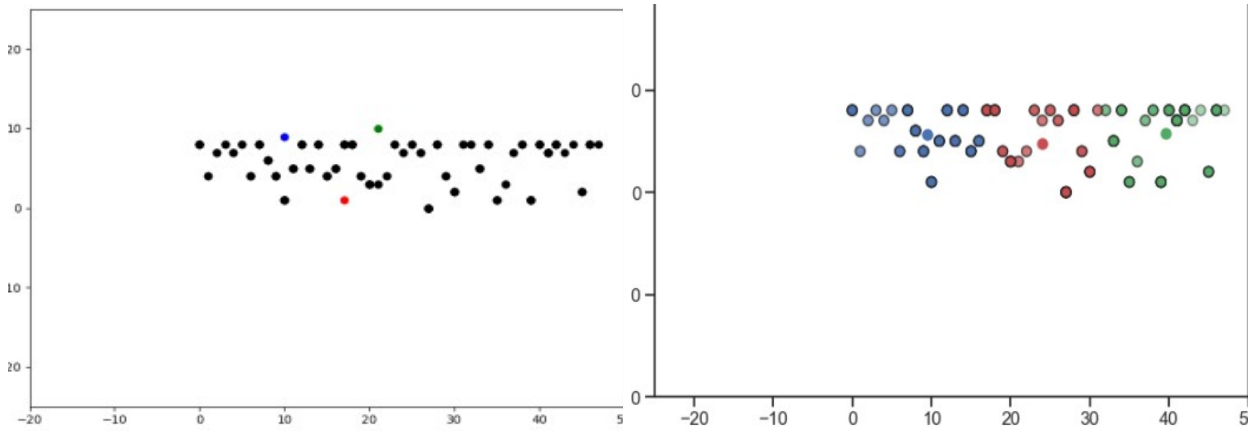

Рисунок 3.6 — Вибірка даних при ініціалізації та після закінчення останньої ітерації алгоритму

Хоча отриманий алгоритм здійснив кластеризацію вибірки даних за алгоритмом k-means, але виявилось, що трьох кластерів недостатньо для ідентифікації аномалій.

4 КЛАСТЕРИЗАЦІЯ ЗА ДОПОМОГОЮ АЛГОРИТМУ AFFINITY PROPAGATION

4.1 Огляд використаних інструментів

Для реалізації обраного алгоритму кластеризації було використано мову Python версії 3.9.

В якості середовища роботи використано Jupyter Notebook – інтерактивне середовище розробки, дуже зручне для роботи з кодом, наукових розрахунків, роботи зі схемами, графіками. Jupyter Notebook підтримує багато мов програмування та може працювати в браузері без будь-яких додаткових інсталяцій. Але зручніше ним користуватися через Anaconda Navigator – графічний інтерфейс, який є частиною дистрибутиву Anaconda. Кросплатформенний дистрибутив Anaconda об’єднує в собі різні програмні продукти, бібліотеки та пакети, а також хмарне середовище для роботи з даними, вирішення статистичних та аналітичних задач, застосуванні методів машинного навчання тощо.

Для наочності при реалізації алгоритму було обрано ту ж саму вибірку даних, яка використовується в наведеному прикладі алгоритму кластеризації k-means – файл у фоматі csv, який представляє собою витяг журналу подій з хмарної інфраструктури GCP після симуляції атак на неї і містить 7 ознак об’єктів.

Серед бібліотек для роботи обрано sklearn, numpy, pandas, matplotlib.

4.2 Опис алгоритму

Для реалізації було обрано алгоритм кластеризації поширення схожості – Affinity Propagation. Він заснований на концепції попарного «обміну

(поширення) повідомленнями» між точками – об’єктами вибірки, що кластеризується.

Цей метод кластеризації був описаний в 2007 році Бренданом Фрейєм та Делбертом Дуеком в роботі «Clustering by passing messages between data points».

Для алгоритму Affinity Propagation не потрібно задавати кількість кластерів як вхідний параметр. Алгоритм самостійно визначає зразкові елементи серед усіх об’єктів вибірки даних – точок даних і формує кластери з даних навколо них. Припускаючи, що кожен з об’єктів вибірки даних може бути центром кластеру, Affinity Propagation уникає проблем інших алгоритмів, пов’язаних з невдалою ініціалізацією [21]. Алгоритм має вхідні параметри, але на кількість кластерів впливають і результати обміну повідомленнями між об’єктами вибірки даних.

Основними поняттями, якими оперує алгоритм є:

- схожість (similarity);
- відповідальність (responsibility);
- доступність (availability).

Схожість кількісно визначає міру схожості між будь-якими двома об’єктами вибірки. Зазвичай береться від’ємна квадратична відстань, Евклідова відстань. Подається у вигляді матриці.

Алгоритм виконується шляхом чергування двох етапів передачі повідомлень, які оновлюють дві матриці: відповідальності і доступності. Матриця відповідальності R кількісно визначають, наскільки певний об’єкт вибірки підходить для того, щоб служити зразком в порівнянні з іншими об’єктами-кандидатами. Матриця доступності A містить значення, які відображають наскільки для об’єкта доречно обрати одного кандидата як зразок відносно інших кандидатів.

Отже, кожен з об’єктів вибірки відправляє повідомлення, яким інформує інші об’єкти про свою доступність і про готовність брати відповідальність за інші об’єкти, а також приймає такі повідомлення від

інших об’єктів. При Affinity Propagation одна ітерація включає обчислення всіх повідомлень про відповідальність на основі поточних повідомлень про доступність, подібності вхідних даних і переваг введення, а потім обчислення всіх повідомлень про доступність на основі повідомлень про відповідальність, які були тільки що оновлені. Процес триває до тих пір, поки межі кластерів не перестануть змінюватись.

Алгоритм Affinity Propagation доволі простий у реалізації, гнучкий, а також захищений від можливих помилок через те, що вхідні параметри впливають на результат кластеризації опосередковано.

Недоліком алгоритму Affinity Propagation є те, що він більше підходить для роботи з невеликими чи середніми об’ємами даних або з розрідженими даними. Це зумовлено об’ємом роботи, яку йому доводиться виконувати, а також великим об’ємом пам’яті, необхідної йому для роботи.

У цього алгоритму є модифікація Fast Affinity Propagation, при якій найвіддаленіші один від одного об’єкти не обмінюються повідомленнями, тим самим скорочуючи кількість обмінів повідомленнями і пришвидшуючи процес кластеризації.

4.3 Реалізація алгоритму

Вже підготовлена вибірка даних для кластеризації в форматі csv за допомогою методів бібліотеки pandas була завантажена в роботу, а усі ознаки об’єктів було нормалізовано - конвертовано в числові значення за допомогою LabelEncoder з бібліотеки sklearn для зручності роботи.

При ініціалізації моделі задаються параметри, які напряму та опосередковано впливають на модель. Основними параметрами є «preference» – контролює кількість використовуваних зразків (або прототипів), а також «damping» – знижує відповідальність і доступність повідомлень, щоб уникнути числових коливань при оновленні цих повідомлень. Обчислювальні обов'язки та доступність відповідно до простих

правил оновлення часто призводять до коливань, викликаних «перевищенням» рішення, тому повідомлення про відповідальність та доступність «затухають». Високі значення «preference» призведуть до того, що Affinity Propagation призведе до знаходження безлічі екземплярів (кластерів), в той час як низькі значення приведуть до невеликого числа екземплярів (кластерів). Хорошим початковим вибором для переваги є мінімальна схожість або середня схожість. Значення «damping» знаходяться в діапазоні [0,5-1).

Лістинг 4.1 – Ініціалізація та тренуваня моделі

```
af = AffinityPropagation(preference=-95,random_state=15, damping = 0.7, max_iter = 500, verbose = True)

clustering = af.fit(df[['Item name', 'Item Type']])

cluster_centers_indices = af.cluster_centers_indices_
```

Візуалізація графіків до та після застосування алгоритму кластеризації, задання його параметрів та способу відображення даних (точкова діаграма – scatter), здійснюється як показано в лістингу 4.1.

Лістинг 4.1 – Створення графіків

```
plt.scatter(df['Item name'], df['Item Type'], cmap='viridis')
plt.xlim(-25, 50)
plt.ylim(-20, 30)
plt.show()


print('Estimated number of clusters: %d' % n_clusters_)
plt.scatter(df['Item name'], df['Item Type'],  c=clustering.labels_, cmap='rainbow',  alpha=0.9, edgecolors='b')
plt.title(f'Estimated number of clusters = {n_clusters_}')
plt.xlim(-25, 50)
plt.ylim(-20, 30)
plt.show()
```

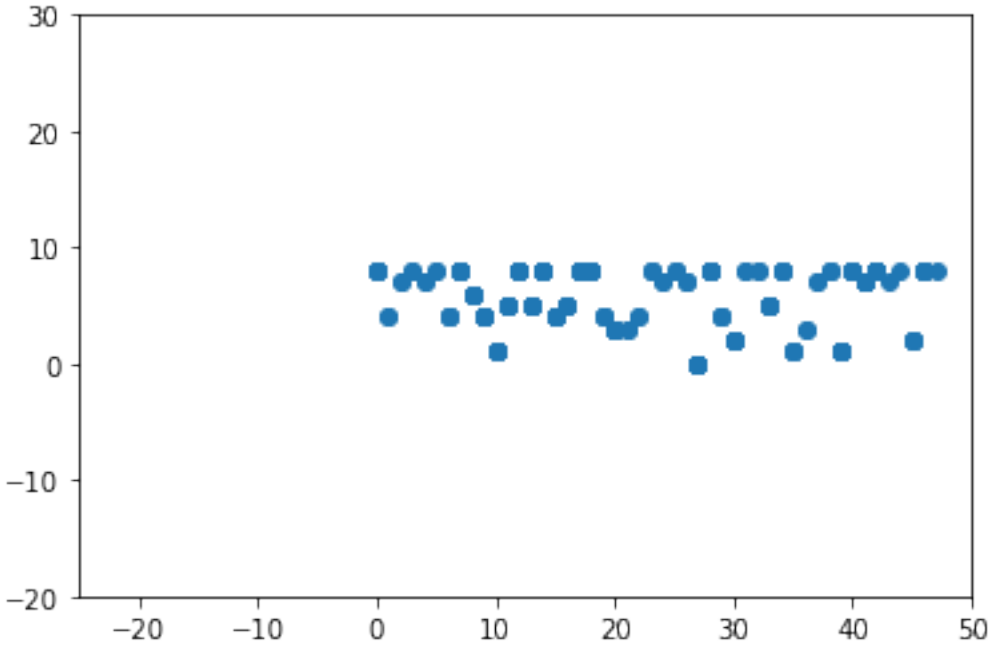

Рисунок 4.1 — Вибірка даних на початку кластеризації

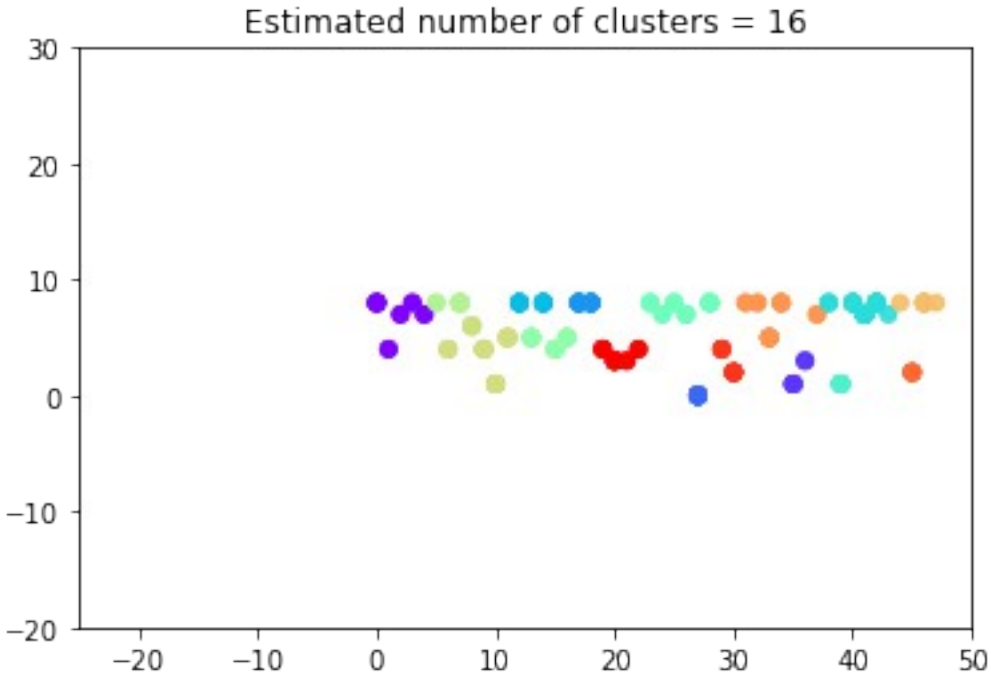

Рисунок 4.2 — Вибірка даних після останньої ітерації алгоритму

Після закінчення роботи алгоритму можна побачити, як дані

розподілилися між 14 кластерами. Візуально кластери виглядають згрупованими.

Центри кластерів мають наступні координати:

1: [2 7]

2: [35 1]

3: [27 0]

4: [17 8]

5: [14 8]

6: [41 7]

7: [39 1]

8: [26 7]

9: [15 4]

10: [7 8]

11: [9 4]

12: [46 8]

13: [34 8]

14: [45 2]

15: [30 2]

16: [20 3]

4.4 Оцінка якості роботи алгоритму

Через те, що не існує еталонної вибірки, інформації про справжнє розбиття на кластери або щодо кількості кластерів, для оцінки якості роботи алгоритму були використані відносні та внутрішні методи оцінки якості кластеризації.

Відносний метод полягає у порівнянні результатів роботи алгоритму при запуску з різними параметрами. Найбільший вплив на результат роботи алгоритму мають вхідні параметри «preference», а також «damping». Зміна

значень цих параметрів найбільше впливає на кількість кластерів, отриманих в результаті.

Для того, щоб скористатися внутрішніми методами оцінки якості роботи алгоритму кластеризації, можна підключити модуль metrics з з бібліотеки sklearn.

Для оцінки якості алгоритму Affinity Propagation було обрано ряд методів оцінки.

Оцінка якості кластеризації за допомогою Силуету. Цей аналіз відноситься до методу інтерпретації та перевірки узгодженості всередині кластерів даних. Значення силуету - це показник того, наскільки об'єкт схожий на свій власний кластер в порівнянні з іншими кластерами. Його можна використовувати для вивчення відстані розподілу між отриманими кластерами. Графік силуету відображає міру близькості кожної точки в одному кластері до точок в сусідніх кластерах і, таким чином, забезпечує спосіб візуальної оцінки таких параметрів, як кількість кластерів.

Індекс Силуету знаходиться в межах [-1, 1].

Якщо значення силуету близьке до 1, зразок добре кластеризований і вже віднесений до дуже відповідного кластеру. Якщо значення силуету дорівнює 0, вибірка може бути призначена іншому найближчому до неї кластеру, і вибірка знаходиться однаково далеко від обох кластерів. Це означає, що він вказує на те, що кластери перекриваються. Якщо значення силуету близько до -1, зразок класифікується неправильно і просто поміщається десь між кластерами.

Лістинг 4.1 – Розрахунок і виведення індексу Силуету

```
      print("Silhouette Coefficient: %0.3f"
      % metrics.silhouette_score(X[['Item name', 'Item Type']], labels,
metric='sqeuclidean'))
```

Індекс Девіса-Болдуіна визначається як середня міра подібності кожного кластера з найбільш схожим кластером, де подібність являє собою

відношення відстаней всередині кластера до відстаней між кластерами. Таким чином, кластери, розташовані далі один від одного і менш розосереджені, приведуть до кращого результату.

Мінімальний бал дорівнює нулю, а нижчі значення вказують на кращу кластеризацію.

Лістинг 4.1 – Розрахунок і виведення індексу Девіса-Болдуіна

```
print("Davies-Bouldin Index: %0.3f"
% metrics.davies_bouldin_score(X[['Item name', 'Item Type']], labels))
```

Ще одним зі способів оцінки якості кластеризації є Індекс Calinski-Harabasz. Чим вище значення цього індексу, тим краще в створеній моделі виражені кластери.

Лістинг 4.1 – Розрахунок і виведення індексу Calinski-Harabasz

```
print("Calinski-Hasabasz Index:
%0.3f" % metrics.calinski_harabasz_score(X[['Item name', 'Item Type']],
labels))
```

Саме ці три метрики було використано для оцінки якості роботи створеного алгоритму.

Алгоритм кластеризації Affinity Propagation запускався декілька разів з різними вхідними параметрами, кожного разу аналізувалась кількість кластерів, коректність кластеризації оцінювалась візуально, а також за допомогою вищезазначених методів, тому оцінка якості роботи алгоритму була проведена ще й відносним методом.

Алгоритм виконав кластеризацію всієї вибірки даних за 103 ітерації і розбив її на 16 кластерів.

Таблиця 4.1 – Результати оцінки якості алгоритму кластеризації Affinity Propagation

Оптимальні параметри:	preference=-95 random_state=15 damping = 0.7
Кількість ітерацій	103
Кількість кластерів	16
Silhouette Coefficient	0.762
Davies-Bouldin Index	0.479
Calinski-Hasabasz Index	1238.884

Отримані результати показників якості роботи алгоритму кластеризації доволі непогані. Отже, можна зробити висновок, що алгоритм Affinity Propagation впорався з задачею кластеризації, хоча і не ідеально.

5 АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

5.1 Порівняння двох алгоритмів кластеризації

Вибір алгоритму кластеризації залежить лише від поставленої задачі та обсягу даних. Єдиного, універсального алгоритму не існує. Тому для вирішення поставленої задачі варто спробувати декілька варіантів рішення, а потім оцінити якість їх роботи і обрати той алгоритм, який показав кращий результат.

Алгоритм k-means та алгоритм Affinity Propagation – алгоритми кластеризації. Обидва ці алгоритми доволі прості в реалізації. Але вони відрізняються

Недоліком алгоритму k-means є необхість розрахуватии правильно і задати на початку роботи алгоритму кількість кластерів, а також вибір ініціалізуючих значень центроїдів. Обидва ці параметри алгоритму напряму впливають на результат. Від дуже чутливий до початкових конфігурацій.

Affinity propagation не потребує задання таких важливих для результату параметрів. Але при його ініціалізації все ж задаються параметри, які впливають на те, як відбуватиметься «обмін повідомленнями» об’єктів вибірки – це показники подібності між парами даних. Окрім того, алгоритм розглядає абсолютно усі об’єкти в якості центрів кластеру.

Але цей алгоритм на великих вибірках даних вимагає багато обчислювальних потужностей і об’ємів пам’яті, стає дуже складним.

5.2 Якість отриманих результатів для двох реалізацій

Маючи результати роботи обох алгоритмів, можна зробити висновки про якість їх роботи та чи підходять вони для вирішення поставленої задачі – виявлення аномалій.

В розглянутій реалізації кластеризації вибірки даних за алгоритмом k-means було встановлено, що трьох кластерів не достатньо для ідентифікації аномальних подій у вибірці даних, тому кількість кластерів було збільшено до десяти.

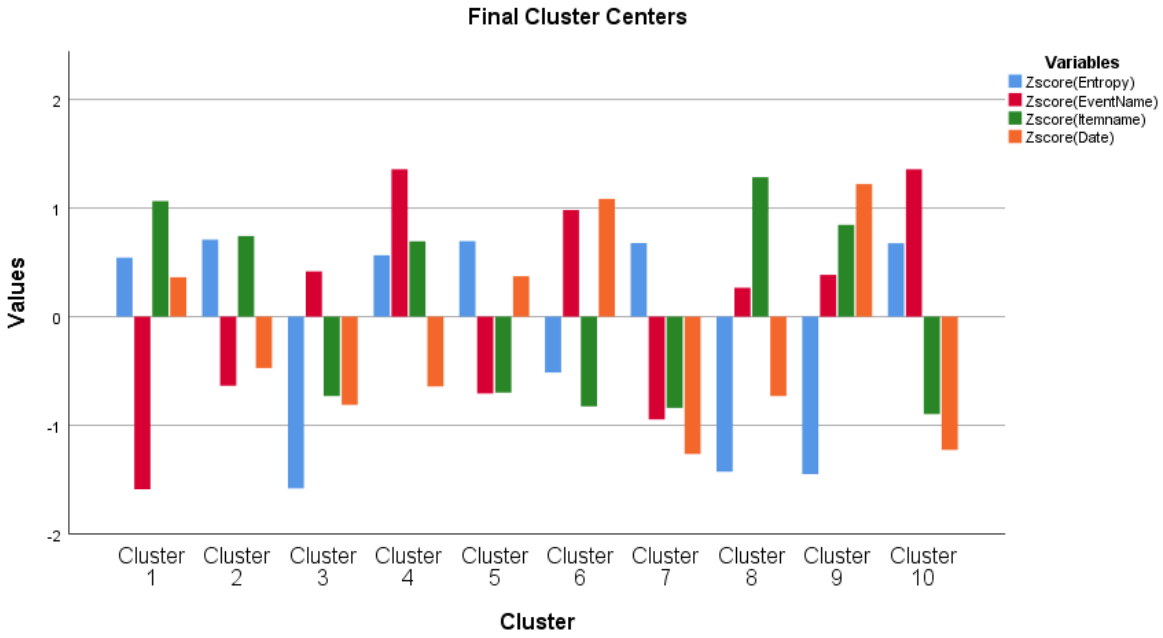

Рисунок 5.1— Результати розрахунків

Таблиця 5.1 — Дані декількох кластерів

Кластери	TP	FP	FN	Точність	Відгук	F1-міра
Кластер 2	21	4	10	0.84	0.68	0.75
Кластер 5	43	13	10	0.77	0.81	0.79
Кластер 7	10	10	10	0.5	0.5	0.5

При проведенні розрахунків оптимуму 10 кластерів з використанням алгоритму k-means було встановлено, що отриманим кластерам дещо бракує точності в детектуванні аномалій.

Алгоритм affinity propagation показав трохи кращі результати, а також розбив дані на більшу кількість кластерів. Але все одно обом алгоритмам кластеризації не вистачає точності. Можливо подана на вхід вибірка даних недостатня для цих алгоритмів.

Я вважаю, що ці базові алгоритми можуть бути складовою більшої системи детектування програм-вимагачів або використовуватись в якихось комплексних алгоритмах машинного навчання, наприклад мета-навчання.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було реалізовано алгоритм кластеризації affinity propagation, який можна використовувати для детектування аномальних подій у хмарному середовищі.

Окрім цього в роботі було проведено аналіз програм-вимагачів, визначення основ та напрямків застосування машинного навчання, дослідження існуючих методів виявлення програм-вимагачів з використанням машинного навчання, а також існуюча реалізація алгоритму кластеризації.

Машинне навчання має великий потенціал і вже активно використовується великою кількістю компаній, які створюють рішення та різні продукти у сфері кібербезпеки. Моделі машинного навчання застосовуються для аналізу існуючих даних, прогнозування, захисту систем в режимі реального часу, управління ризиками, оцінки загроз та менеджменту реагування на загрози, перевірки систем на наявність вразливостей, які можна використати для здійснення атак.

Розроблений алгоритм було протестовано на вибірці даних – журналі подій з хмарного середовища, на яке було здійснено симуляцію атаки. В результаті алгоритм показав непогані результати, але йому, як і алгоритму з прикладу реалізації, все одно бракує точності, самостійно цей алгоритм не . Можливе його використання як частини більшої і складнішої системи детектування програм-вимагачів у хмарному середовищі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Ransomware [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Ransomware

2. A History of Ransomware Attacks: The Biggest and Worst Ransomware Attacks of All Time [Електронний ресурс] – Режим доступу: https://digitalguardian.com/blog/history-ransomware-attacks-biggest-and-worst-ransomware-attacks-all-time

3. Timeline of computer viruses and worms [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Timeline_of_computer_viruses_and_worms

4. RANSOMWARE AS A SERVICE (RAAS) EXPLAINED [Електронний ресурс] – Режим доступу: https://www.crowdstrike.com/cybersecurity-101/ransomware/ransomware-as-a-service-raas/

5. Ransomware world in 2021: who, how and why [Електронний ресурс] – Режим доступу https://securelist.com/ransomware-world-in-2021/102169/

6. Over half of ransomware victims pay the ransom, but only a quarter see their full data returned [Електронний ресурс] – Режим доступу: https://www.kaspersky.com/about/press-releases/2021_over-half-of-ransomware-victims-pay-the-ransom-but-only-a-quarter-see-their-full-data-returned

7. ATTACK LANDSCAPE UPDATE [Електронний ресурс] – Режим доступу: https://blog-assets.f-secure.com/wp-content/uploads/2021/03/30120359/attack-landscape-update-h1-2021.pdf

8. Машинне навчання [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Машинне_навчання

9. Machine learning [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Machine_learning

10. Artificial Intelligence vs. Machine Learning in Cybersecurity

[Електронний ресурс] – Режим доступу: https://www.varonis.com/blog/ai-vs-ml-in-cybersecurity/

11. Use cases for AI and ML in cyber security [Електронний ресурс] – Режим доступу: https://www.information-age.com/use-cases-for-ai-ml-cyber-security-123491392/

12. 30 COMPANIES MERGING AI AND CYBERSECURITY TO KEEP US SAFE AND SOUND [Електронний ресурс] – Режим доступу: https://builtin.com/artificial-intelligence/artificial-intelligence-cybersecurity

13. MACHINE LEARNING CYBERSECURITY: HOW IT WORKS AND COMPANIES TO KNOW [Електронний ресурс] – Режим доступу: https://builtin.com/artificial-intelligence/machine-learning-cybersecurity

14. The State of Ransomware in 2021 [Електронний ресурс] – Режим доступу: https://www.blackfog.com/the-state-of-ransomware-in-2021/

15. Anomaly detection approach to detect ransomware attack in the cloud. Akintunde Adedamola Emmanuel, 2020.

16. Кластеризация [Електронний ресурс] – Режим доступу: http://neerc.ifmo.ru/wiki/index.php?title=%D0%9A%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F.

17. Кластерний аналіз [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/ Кластерний_аналіз

18. Clustering [Електронний ресурс] – Режим доступу: https://scikit-learn.org/stable/modules/clustering.html

19. How Ransomware Attacks [Електронний ресурс] – Режим доступу: https://www.sophos.com/en-us/medialibrary/pdfs/technical-papers/sophoslabs-ransomware-behavior-report.pdf

20. Affinity propagation [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Affinity_propagation

21. Brendan J. Frey, Delbert Dueck. Clustering by passing messages between data points propagation [Електронний ресурс] – Режим доступу:

http://utstat.toronto.edu/reid/sta414/frey-affinity.pdf

22. Оценка качества в задаче кластеризации [Електронний ресурс] – Режим доступу: https://neerc.ifmo.ru/wiki/index.php?title=%D0%9E%D1%86%D0%B5%D0%BD%D0%BA%D0%B0_%D0%BA%D0%B0%D1%87%D0%B5%D1%81%D1%82%D0%B2%D0%B0_%D0%B2_%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B5_%D0%BA%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8