Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кваліфікаційна наукова праця
на правах рукопису

АДАМОВ ОЛЕКСАНДР СЕМЕНОВИЧ

УДК 658:512.011: 681.326: 519.713

ДИСЕРТАЦІЯ

МОДЕЛІ І МЕТОДИ ЗАХИСТУ КІБЕРПРОСТОРУ НА ОСНОВІ АНАЛІЗУ ВЕЛИКИХ ДАНИХ З ВИКОРИСТАННЯМ МАШИННОГО НАВЧАННЯ

05.13.05 – комп'ютерні системи та компоненти

Технічні науки

Подається на здобуття наукового ступеня кандидата технічних наук

Дисертація містить результати власних досліджень.
Використання ідей, результатів і текстів інших авторів мають посилання
на відповідне джерело _____ О.С. Адамов

Науковий керівник:
Хаханов Володимир Іванович,
доктор технічних наук, професор

Цей примірник дисертації ідентичний за змістом
з іншими примірниками, що подані до спеціалізованої
вченої ради Д 64.052.01

Учений секретар спеціалізованої вченої ради Д 64.052.01 Є.І. Литвинова

Харків – 2019

АНОТАЦІЯ

Адамов О.С. Моделі і методи захисту кіберпростору на основі аналізу великих даних з використанням машинного навчання. – Кваліфікаційна наукова робота на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук (доктора філософії) за спеціальністю 05.13.05 «Комп'ютерні системи та компоненти». – Харківський національний університет радіоелектроніки, Міністерство освіти і науки України, Харків, 2019.

Мета дослідження – істотне зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування за рахунок введення обчислювальної надмірності в інфраструктуру кіберпростору.

Задачі дослідження:

1) Удосконалити структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів на основі використання дедуктивного аналізу обчислювальних систем.

2) Розробити сигнатурно-кубітні методи синтезу еталонних логічних схем malware-функціональностей і паралельного моделювання malware-driven великих даних для визначення приналежності поточного коду до існуючих деструктивних компонентів в malware бібліотеці.

3) Розробити сигнатурно-кубітну модель активного online cyber security комп'ютингу для моніторингу вхідних потоків malware-даних і управління процесом видалення деструктивних компонентів.

4) Удосконалити засоби захисту кіберпростору шляхом логічного тестування і діагностування атак і шкідливих компонентів на основі використання алгоритмів машинного навчання.

5) Розробити метод атрибутно-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів і метод перевірки поліморфних шкідливих програм на основі врахування контрольних сум Portable Executa-

ble секцій в виконувані файли і застосування апарату інтелектуального аналізу даних.

6) Виконати тестування і верифікацію розроблених програмних засобів тестування, перевірки та діагностування шкідливих програм шляхом емуляції атак на основі існуючих malware бібліотек.

Об'єкт дослідження – хмарні і edge-computing технології, засновані на високо-продуктивних дата-центрах, комп'ютерних гаджетах, системах і мережах, виконують сервісні функції зберігання та аналізу великих даних.

Предмет дослідження – структурно-логічні моделі, методи, засоби тестування деструктивних компонентів, захисту індивідуального та колективного сервіс-комп'ютингу від кіберзагроз.

Науково-практична задача – введення в інфраструктуру комп’ютингового простору програмної надмірності в формі моделей, методів і програмних додатків для істотного зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування.

Сутність дослідження – розробка моделей, методів і програмних додатків для істотного зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування за рахунок введення обчислювальної надмірності в інфраструктуру кіберпростору.

Наукова новизна результатів досліджень:

1) Удосконалено структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів, які відрізняються використанням методу дедуктивного паралельного аналізу обчислювальної системи для перевірки та діагностування malware.

2) Запропоновано нові методи синтезу еталонних логічних схем mal-ware-функціональностей, які характеризуються використанням сигнатурно-

кубітних структур, що дає можливість паралельно моделювати malware-driven великі дані для визначення приналежності поточного коду до існуючих деструктивних компонентів в malware бібліотеці.

3) Розроблено нову модель активного online cyber security комп'ютингу, яка характеризується сигнатурно-кубітним поданням інформації, що дає можливість підвищувати швидкодію процесів моніторингу вхідних потоків malware-даних і управління видаленням деструктивних компонентів.

4) Запропоновано новий метод атрибутно-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів і метод перевірки поліморфних шкідливих програм на основі врахування контрольних сум Portable Executable секцій в виконувані файли і застосування апарату інтелектуального аналізу даних.

5) Удосконалено засоби захисту кіберпростору, які відрізняються використанням моделей і методів сигнатурно-логічного тестування атак, пошуку криптопримітивів в троянських програмах-шифрувальниках на основі використання алгоритмів машинного навчання, що дає можливість істотно зменшити час відновлення працездатності обчислювальної структури.

Практичне значення одержаних результатів досліджень визначається:

5) Тестуванням, верифікацією і впровадженням розроблених програмних засобів перевірки, діагностування шкідливих програм і атак, що дає можливість виконувати їх моделювання із залученням існуючих додатків і malware бібліотек.

6) Програмною реалізацією методу атрибутного-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів, який відрізняється застосуванням апарату інтелектуального аналізу даних, що дає можливість визначати вірогідну оцінку небезпеки URL-адреси на основі його атрибутів.

7) Програмною реалізацією методу перевірки поліморфних шкідливих програм, який відрізняється інваріантністю до детермінізму сигнатур в коді і

урахуванням тільки контрольних сум Portable Executable (PE) секцій в виконуваних файлах, що дає можливість поліпшити продуктивність процедур діагностування деструктивних компонентів.

Обґрунтованість наукових положень. Отримані в процесі виконання досліджень наукові висновки і практичні результати є достовірними, що підтверджується точністю детектування і класифікацією прикладів загроз нульового дня, серед яких нові версії кріптолокерів і складних загроз (Advanced Persistent Threats – APT); позитивними відгуками вчених і фахівців на доповіді на міжнародних конференціях, присвячених кібербезпеці, таких як Virus Bulletin 2015 Празі, Чехія, 2018 у Монреалі, Канада, OpenStack Summit 2015 у Ванкувері, Канада, і OpenStack Summit 2015 в Остіні, штат Техас, США, IEEE East -West Design & Test Symposium (EWDTS'2017) 2017 в Novy Sad, Serbia.

Впровадження результатів дисертації. Результати дисертації у складі моделей, методів та інфраструктури впроваджені у навчальний процес Харківського національного університету радіоелектроніки (акти про впровадження від 20.05.2019, 21.05.2019); у науково-виробничу діяльність компанії Design & Test Lab (довідка від 18.05.2019), у навчальний процес Blekinge Institute of Technology (BTH), Karlskruna, Sweden (лист ‘Statment of Reseach Results Impact on University Education Program’ від 29.05.2019).

Публікації. Результати дисертаційної роботи відображені в 31 друкованій праці: 3 розділи у закордонних монографіях (з них 1 входить до наукометричної бази Scopus), 7 статей (з них 5 – у наукових журналах, включених до «Переліку наукових фахових видань України»; 2 статті в міжнародних наукових журналах за кордоном; 4 статті входять до міжнародних наукометричних баз), а також у 21 міжнародній науковій конференції (з них 13 за кордоном, 12 входять до наукометричної бази Scopus). Здобувач має 13 публікацій у наукометричній базі Scopus та індекс Хірша h=3.

СПИСОК ОПУБЛІКОВАНИХ РОБІТ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Список публікацій здобувача, в яких опубліковані основні наукові результати дисертації:

1. Hahanov V., Gharibi W., Litvinova E., *Adamov A*. Cyber Physical Computing for IoT-driven Services. – New York. USA. – Springer, 2018. 279p. [Chapter 2. Multiprocessor Architecture for Big Data Computing [Text] / V. Hahanov, W. Gharibi, E. Litvinova, *A*. *Adamov*. P. 21-41] (Springer, Scopus).

2. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: Education Challenges and Development of Advanced Malware Analysis Course [Text] / *A. Adamov*. P. 130-134].

3. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: The Cloud Threat Landscape [Text] / A. Carlsson, *A*. *Adamov*. P. 182-186].

4. *Adamov A.* Security risks and modern cyber security technologies for corporate networks [Text] / *A. Adamov*, V. Hahanov, W. Gharibi // Radioelektroniks and informatics. – 2010. – № 4. – P. 31–35. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

5. Хаханов В.И. Инфраструктура анализа и информационной безопасности киберпространства [Текст] / В.И. Хаханов, С.В. Чумаченко, Е.И. Литвинова, А.С. Мищенко, *А.С. Адамов* // Радиоэлектроника и информатика. – 2011. – №2 (53). – С. 40-60. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

6. *Адамов О.С.* Блокчейн інфраструктура для захисту кіберсистем [Текст] / *О.С. Адамов,* В.І. Хаханов, С.В. Чумаченко, В.Г. Абдуллаєв // Радиоэлектроника и информатика. – 2018. – №4 (83). – С. 64-85. (Журнал рефе-

рується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

7. *Адамов О.С.,* Хаханов В.І. Сигнатурно-кубітні методи розпізнавання деструктивних кодів [Текст] / *О.С. Адамов,* В.І. Хаханов // Радиоэлектроника и информатика. – 2019. – №1 (84). – С. 35-53. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

8. *Adamov A.* Analysis and Detection of Polymorphic Spyware [Text] / *A. Adamov*, A. Saprykin // Hakin9 Magazine. 2013. Vol. 8, № 01. Issue 01/2013 (61). Warsow: Software Press, 2013. – P. 6-11.

9. *Adamov A.S.*, Hahanov V.I. A Method for the Attribute-based Detection of URLs Using Frequency Patterns [Text] / Вестник Государственного Инженерного Университета Армении. Серия: Информационные Технологии, Электроника, Радиотехника. 2014. Вып. 17, No2. P. 59-66.

10. Сапрыкин А.С. Методика оценки убытков предприятия от вредоносных программ / Сапрыкин А.С., Бочарникова М.В., *Адамов А.С.* // Вестник национального технического университета "ХПИ" (Новое решение в современных технологиях). 2009. №8. С. 58-64.

Результати, які засвідчують апробацію матеріалів дисертації:

11. *Adamov A.* Electronic System Level Models for Functional Verification of System-on-Chip / *A. Adamov*, K. Mostovaya, Syzonenko, I et al. // Proc. of International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics” (CADSM). – Lviv-Polyana, Ukraine. – February 20-24, 2007. – P. 348–350. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

12. *Adamov A.* Transactional Data Analysis of Electronic System Level Models [Text] / *A. Adamov*, V. Hahanov, D. Melnyk et al. // Proc. of East-West

Design and Test Symposium. September 7–10, 2007. Yerevan, Armenia. 2007. – P. 745–748.

13. *Adamov A.* Model of Source Code Analyzer for Hardware Description Languages [Text] / Dmytro Melnyk, Sergei Zaychenko, *Aleksandr Adamov*, Vladimir Hahanov // Proc. of East-West Design and Test Symposium (EWDTS). September 7–10, 2007, Yerevan, Armenia. – Yerevan, 2007. – P. 470–474.

14. Hahanova I.V. Transaction level model of embedded processor for vector-logical analysis [Text] / Hahanova I.V., Obrizan V., *Adamov A.* et al // Proc. of East-West Design and Test Symposium. September 2013. Rostov-on-Don, Russia. – P. 1-4. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

15. *Adamov A.* Data Mining Techniques for Functional Verification of SoC [Text] / *A. Adamov,* R. Hwang, A. Gavrushenko // Proc. of International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008. – Lviv-Polyana, Ukraine. – February 20-24, 2008. – P. 557-559.

16. *Adamov A.* The Problem of Trojan Inclusions in Software and Hardware [Text] / *A. Adamov*, A. Saprykin // Proc. IEEE East-West Design and Test Symposium (EWDTS'2009). – Moscow, Russia. – 18-21 September 2009. – P. 449-451. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. *Adamov A.* The problem of Hardware Trojans detection in System-on-Chip [Text] / *A. Adamov*, A. Saprykin, D. Melnik // Proc. CAD Systems in Microelectronics (CADSM 2009), – Lviv-Polyana, Ukraine. – 24-28 Feb 2009. – P. 178-179. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. *Adamov A.* Security risks in hardware: Implementation and detection problem [Text] / *Adamov A.*, Hahanov V. // Proc. IEEE East-West Design and Test Symposium (EWDTS'2010), September 17–20, 2010, Saint Petersburg,

Russia. – Piscataway, NJ : IEEE, 2010. – P. 425–427. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. *Adamov A.* A security model of individual cyberspace [Text] / *A. Adamov*, V. Hahanov // Proc. IEEE East-West Design and Test Symposium, September 19–20, 2011. – Sevastopol, Ukraine. – 2011. – P. 169–172. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. *Adamov A.* Discovering New Indicators for Botnet Traffic Detection [Text] / *A. Adamov*, V. Hahanov, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov V. Structures for information retrieval in big data [Text] / V. Hahanov, S. Chumachenko, E. Litvinova, *A. Adamov* et al // Proc. of 13th International Conference Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2015), Lviv, Ukraine, 2015. – P. 70-75. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. *Adamov A.* A Sandboxing Method to Protect Cloud Cyberspace [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Test Symposium (EWDTS'2015). – Batumi, Georgia. – September 27-30, 2015. – P. 180–183. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

23. *Adamov A.* Android Ransomware: Turning CryptoLocker into CryptoUnlocker [Text] / *A. Adamov* // Proc. of the 25th Virus Bulletin International Conference, Prague, Czech Republic, 30 Sep-2 Oct 2015 – P. 220-223.

24. *Adamov A.* Detecting targeted attacks in the cloud [Text] / *A. Adamov* // Proc. of the OpenStack Summit, Vancouver, Canada, 18-22 May 2015. 1 p.

25. *Adamov A.* Using Open Source Security Architecture to Defend against Targeted Attacks / Alexander Adamov, Dan Lambright // Proc. of the OpenStack Summit, Austin, TX, US, April 25-29, 2016. 1 p.

26. *Adamov A.* Cloud incident response model [Text] / *Alexander Adamov*, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

27. *Adamov A.* The State of Ransomware. Trends and Mitigation Techniques [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – October 2, 2017, Novy Sad, Serbia. – P. 121–128. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

28. *Adamov A.* Battlefield Ukraine: finding patterns behind summer cyber attacks [Text] / *A. Adamov,* A. Carlsson // Proc. of the 27th Virus Bulletin International Conference, Madrid, Spain, 4-6 Oct 2017 – Appendix: Last-minute presentations. – P. 4-5.

29. *Adamov A.* Artificial Intelligence to Assist with Ransomware Cryptanalysis [Text] // Proc. of the 28th Virus Bulletin International Conference, Montreal, Canada, 3-5 Oct 2018. – P. 289-292.

30. *Adamov A.* Creating Ransomware Decryptors [Text] // Proc. of 14th Cyber Security Conference UISGCON14, Kyiv, 2018. P. 3.

31. *Адамов А.С.* Модель поиска вредоносного кода в программных продуктах [Текст] / А.С. Адамов, Д.А. Щербин, С.В. Чумаченко // Материалы 16-го Международного молодежного форума "Радиоэлектроника и молодежь в XXI веке", 17-19 апреля 2012. – P. 121–123.

Ключові слова: кіберпростір, великі дані, машинне навчання, логічний процесор, malware, кубітна модель, сінтез тестів, вразливість, Cyber Security Computing.

АННОТАЦИЯ

Адамов А.С. Модели и методы защиты киберпространства на основе анализа больших данных с использованием машинного обучения. – Квалификационная научная работа на правах рукописи.

Диссертация на соискание ученой степени кандидата технических наук (доктора философии) по специальности 05.13.05 «Компьютерные системы и компоненты». – Харьковский национальный университет радиоэлектроники, Министерство образования и науки Украины, Харьков, 2019.

Цель исследования – существенное уменьшение времени обнаружения и блокирования кибератак, направленных на киберпространство субъекта, путем использования разработанных матричных моделей и логических методов тестирования, проверки и диагностирования за счет введения вычислительной избыточности в инфраструктуру киберпространства.

Задачи исследования:

1) Усовершенствовать структурно-логические модели и методы проверки киберпространства для тестирования и диагностирования вредоносных компонентов на основе использования дедуктивного анализа вычислительных систем.

2) Разработать сигнатурно-кубитные методы синтеза эталонных логических схем malware-функциональностей и параллельного моделирования malware-driven больших данных для определения принадлежности текущего кода к существующим деструктивным компонентам в malware библиотеке.

3) Разработать сигнатурно-кубитную модель активного online cyber security компьютинга для мониторинга входных потоков malware-данных и управления процессом удаления деструктивных компонентов.

4) Усовершенствовать средства защиты киберпространства путем логического тестирования и диагностирования атак и вредоносных компонентов на основе использования алгоритмов машинного обучения.

5) Разработать метод атрибутно-ориентированного распознавания URL-адресов с использованием частотных паттернов и метод проверки полиморфных вредоносных программ на основе учета контрольных сумм Portable Executable секций в исполняемой файле и применения аппарата интеллектуального анализа данных.

6) Выполнить тестирование и верификацию разработанных программных средств тестирования, проверки и диагностирования вредоносных программ путем эмуляции атак на основе существующих mal-ware библиотек.

Об'єкт дослідження – облачные и edge-computing технологии, основанные на высоко-производительных дата-центрах, компьютерных гаджетах, системах и сетях, выполняющих сервисные функции хранения и анализа больших данных.

Предмет исследования – структурно-логические модели, методы, средства тестирования деструктивных компонентов, защиты индивидуального и коллективного сервис-компьютинга от киберугроз.

Научно-практическая задача – введение в инфраструктуру компьютингового пространства программной избыточности в форме моделей, методов и программных приложений для существенного уменьшения времени обнаружения и блокирования кибератак, направленных на киберпространство субъекта, путем использования разработанных матричных моделей и логических методов тестирования, проверки и диагностирования.

Сущность исследования – разработка моделей, методов и программных приложений для существенного уменьшения времени обнаружения и блокирования кибератак, направленных на киберпространство субъекта, путем использования разработанных матричных моделей и логических методов тестирования, проверки и диагностирования за счет введения вычислительной избыточности в инфраструктуру киберпространства.

Научная новизна результатов исследований:

1) Усовершенствованы структурно-логические модели и методы проверки киберпространства для тестирования и диагностирования вредоносных компонентов, которые отличаются использованием метода дедуктивного параллельного анализа вычислительной системы для проверки и диагностирования malware.

2) Предложены новые методы синтеза эталонных логических схем malware-функциональностей, которые характеризуются использованием сигнатурно-кубитных структур, что дает возможность параллельно моделировать malware-driven большие данные для определения принадлежности текущего кода к существующим деструктивным компонентам в malware библиотеке.

3) Разработана новая модель активного online cyber security компьютинга, которая характеризуется сигнатурно-кубитным представлением информации, что дает возможность повышать быстродействие процессов мониторинга входных потоков malware-данных и управления удалением деструктивных компонентов.

4) Предложен новый метод атрибутно-ориентированного распознавания URL-адресов с использованием частотных паттернов и метод проверки полиморфных вредоносных программ на основе учета контрольных сумм Portable Executable секций в исполняемые файлы с применением аппарата интеллектуального анализа данных.

5) Усовершенствованы средства защиты киберпространства, которые отличаются использованием моделей и методов сигнатурно-логического тестирования атак, поиска криптопримитивов в троянских программах-шифровальщиках на основе использования алгоритмов машинного обучения, что дает возможность существенно уменьшить время восстановления работоспособности вычислительной структуры.

Практичне значення одержаних результатів досліджень определяется:

6) Тестированием, верификацией и внедрением разработанных

программных средств проверки, диагностирования вредоносных программ и атак, что дает возможность выполнять их моделирование с привлечением существующих приложений и malware библиотек.

7) Программной реализацией метода атрибутно-ориентированного распознавания URL-адресов с использованием частотных паттернов, который отличается применением аппарата интеллектуального анализа данных, что дает возможность определять вероятностную оценку опасности URL-адреса на основе его атрибутов.

8) Программной реализацией метода проверки полиморфных вредоносных программ, который отличается инвариантностью к детерминизму сигнатур в коде и учетом только контрольных сумм Portable Executable (PE) секций в исполняемой файле, что дает возможность улучшить производительность процедур диагностирования деструктивных компонентов.

Достоверность и обоснованность научных результатов подтверждается: точностью детектирования и классификации примеров угроз нулевого дня, среди которых новые версии криптолокеров и сложных угроз (Advanced Persistent Threats – APT); позитивными отзывами ученых и специалистов на доклады на международных конференциях, посвященных кибербезопасности, таких как Virus Bulletin 2015 в Праге, Чехия, 2018 в Монреале, Канада, OpenStack Summit 2015 в Ванкувере, Канада и OpenStack Summit 2015 в Остине, штат Техас, США, IEEE East-West Design & Test Symposium (EWDTS'2017), 2017 в Novy Sad, Serbia.

Результаты диссертации в составе моделей, методов и инфраструктуры внедрены в учебный процесс Харьковского национального университета радиоэлектроники (акты о внедрении от 20.05.2019, 21.05.2019); в научно-производственную деятельность компанії Design & Test Lab (справка от 18.05.2019), учебный процесс Blekinge Institute of Technology (BTH), Karls-kruna, Sweden ('Statment of Reseach Results Impact on University Education Program' от 29.05.2019).

Результаты диссертации отражены в 31 печатной работе: 3 раздела в зарубежных монографиях (из них 1 входит в наукометрическую базу Scopus), 7 статей (из них 5 – в научных журналах, включенных в «Перечень научных специализированных изданий Украины»; 2 статьи в международных научных журналах за рубежом; 4 статьи входят в международные наукометрические базы), а также в 21 международной научной конференции (из них 13 за рубежом, 12 входят в наукометрическую базу Scopus). Диссертант имеет 13 публикаций в наукометрической базе Scopus и индекс Хирша h=3.

СПИСОК ОПУБЛИКОВАННЫХ РАБОТ ПО ТЕМЕ ДИССЕРТАЦИИ

Список публикаций соискателя, в которых опубликованы основные научные результаты диссертации:

1. Hahanov V., Gharibi W., Litvinova E., *Adamov A*. Cyber Physical Computing for IoT-driven Services. – New York. USA. – Springer, 2018. 279p. [Chapter 2. Multiprocessor Architecture for Big Data Computing [Text] / V. Hahanov, W. Gharibi, E. Litvinova, *A*. *Adamov*. P. 21-41] (Springer, Scopus).

2. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: Education Challenges and Development of Advanced Malware Analysis Course [Text] / *A. Adamov*. P. 130-134].

3. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: The Cloud Threat Landscape [Text] / A. Carlsson, *A. Adamov*. P. 182-186].

4. *Adamov A.* Security risks and modern cyber security technologies for corporate networks [Text] / *A. Adamov*, V. Hahanov, W. Gharibi // Radioelektroniks and informatics. – 2010. – № 4. – P. 31–35. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

5. Хаханов В.И. Инфраструктура анализа и информационной безопасности киберпространства [Текст] / В.И. Хаханов, С.В. Чумаченко, Е.И.

Литвинова, А.С. Мищенко, *А.С. Адамов* // Радиоэлектроника и информатика. – 2011. – №2 (53). – С. 40-60. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

6. *Адамов О.С.* Блокчейн інфраструктура для захисту кіберсистем [Текст] / *О.С. Адамов,* В.І. Хаханов, С.В. Чумаченко, В.Г. Абдуллаєв // Радиоэлектроника и информатика. – 2018. – №4 (83). – С. 64-85. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

7. *Адамов О.С.* Сигнатурно-кубітні методи розпізнавання деструктивних кодів [Текст] / *О.С. Адамов,* В.І. Хаханов // Радиоэлектроника и информатика. – 2019. – №1 (84). – С. 35-53. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

8. *Adamov A.* Analysis and Detection of Polymorphic Spyware [Text] / *A. Adamov*, A. Saprykin // Hakin9 Magazine. 2013. Vol. 8, № 01. Issue 01/2013 (61). Warsow: Software Press, 2013. – P. 6-11.

9. *Adamov A.S.*, Hahanov V.I. A Method for the Attribute-based Detection of URLs Using Frequency Patterns [Text] / Вестник Государственного Инженерного Университета Армении. Серия: Информационные Технологии, Электроника, Радиотехника. 2014. Вып. 17, No2. P. 59-66.

10. Сапрыкин А.С. Методика оценки убытков предприятия от вредоносных программ / Сапрыкин А.С., Бочарникова М.В., *Адамов А.С.* // Вестник национального технического университета "ХПИ" (Новое решение в современных технологиях). 2009. №8. С. 58-64.

Результаты, которые подтверждают апробацию материалов диссертации:

11. *Adamov A.* Electronic System Level Models for Functional Verification of System-on-Chip / *A. Adamov*, K. Mostovaya, Syzonenko, I et al. // Proc. of International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics” (CADSM). – Lviv-Polyana, Ukraine. – February 20-24, 2007. – P. 348–350. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

12. *Adamov A.* Transactional Data Analysis of Electronic System Level Models [Text] / *A. Adamov*, V. Hahanov, D. Melnyk et al. // Proc. of East-West Design and Test Symposium. September 7–10, 2007. Yerevan, Armenia. 2007. – P. 745–748.

13. *Adamov A.* Model of Source Code Analyzer for Hardware Description Languages [Text] / Dmytro Melnyk, Sergei Zaychenko, *Aleksandr Adamov*, Vladimir Hahanov // Proc. of East-West Design and Test Symposium (EWDTS). September 7–10, 2007, Yerevan, Armenia. – Yerevan, 2007. – P. 470–474.

14. Hahanova I.V. Transaction level model of embedded processor for vector-logical analysis [Text] / Hahanova I.V., Obrizan V., *Adamov A.* et al // Proc. of East-West Design and Test Symposium. September 2013. Rostov-on-Don, Russia. – P. 1-4. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

15. *Adamov A.* Data Mining Techniques for Functional Verification of SoC [Text] / *A. Adamov*, R. Hwang, A. Gavrushenko // Proc. of International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008. – Lviv-Polyana, Ukraine. – February 20-24, 2008. – P. 557-559.

16. *Adamov A.* The Problem of Trojan Inclusions in Software and Hardware [Text] / *A. Adamov*, A. Saprykin // Proc. IEEE East-West Design and Test Symposium (EWDTS'2009). – Moscow, Russia. – 18-21 September 2009. –

P. 449-451. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. *Adamov A.* The problem of Hardware Trojans detection in System-on-Chip [Text] / *A. Adamov*, A. Saprykin, D. Melnik // Proc. CAD Systems in Microelectronics (CADSM 2009), – Lviv-Polyana, Ukraine. – 24-28 Feb 2009. – P. 178-179. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. *Adamov A.* Security risks in hardware: Implementation and detection problem [Text] / *Adamov A.*, Hahanov V. // Proc. IEEE East-West Design and Test Symposium (EWDTS'2010), September 17–20, 2010, Saint Petersburg, Russia. – Piscataway, NJ : IEEE, 2010. – P. 425–427. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. *Adamov A.* A security model of individual cyberspace [Text] / *A. Adamov*, V. Hahanov // Proc. IEEE East-West Design and Test Symposium, September 19–20, 2011. – Sevastopol, Ukraine. – 2011. – P. 169–172. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. *Adamov A.* Discovering New Indicators for Botnet Traffic Detection [Text] / *A. Adamov*, V. Hahanov, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov V. Structures for information retrieval in big data [Text] / V. Hahanov, S. Chumachenko, E. Litvinova, *A. Adamov* et al // Proc. of 13th International Conference Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2015), Lviv, Ukraine, 2015. – P. 70-75. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. *Adamov A.* A Sandboxing Method to Protect Cloud Cyberspace [Text] / *A.Adamov*, A.Carlsson // Proc. of IEEE East-West Design & Test Test Symposium (EWDTS'2015). – Batumi, Georgia. – September 27-30, 2015. – P. 180–183. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

23. *Adamov A.* Android Ransomware: Turning CryptoLocker into CryptoUnlocker [Text] / *A.Adamov* // Proc. of the 25th Virus Bulletin International Conference, Prague, Czech Republic, 30 Sep-2 Oct 2015 – P. 220-223.

24. *Adamov A.* Detecting targeted attacks in the cloud [Text] / *A. Adamov* // Proc. of the OpenStack Summit, Vancouver, Canada, 18-22 May 2015. 1 p.

25. *Adamov A.* Using Open Source Security Architecture to Defend against Targeted Attacks / *Alexander Adamov*, Dan Lambright // Proc. of the OpenStack Summit, Austin, TX, US, April 25-29, 2016. 1 p.

26. *Adamov A.* Cloud incident response model [Text] / *Alexander Adamov*, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

27. *Adamov A.* The State of Ransomware. Trends and Mitigation Techniques [Text] / *A.Adamov*, A.Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'17), Sep 29 – Oct 2, 2017, Novy Sad, Serbia. – P. 121–128. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

28. *Adamov A.* Battlefield Ukraine: finding patterns behind summer cyber attacks [Text] / *A.Adamov,* A.Carlsson // Proc. of the 27th Virus Bulletin International Conference, Madrid, Spain, 4-6 Oct 2017. – Appendix: Last-minute presentations. – P. 4-5.

29. *Adamov A.* Artificial Intelligence to Assist with Ransomware Cryptanalysis, [Text] // Proc. of the 28th Virus Bulletin International Conference, Montreal, Canada, 3-5 Oct 2018. – P. 289-292.

30. *Adamov A.* Creating Ransomware Decryptors [Text] // Proc. of 14th Cyber Security Conference UISGCON14, Kyiv, 2018. P. 3

31. *Адамов А.С.* Модель поиска вредоносного кода в программных продуктах [Текст] / А.С. Адамов, Д.А. Щербин, С.В. Чумаченко // Материалы 16-го Международного молодежного форума "Радиоэлектроника и молодежь в XXI веке", 17-19 апреля 2012. – Р. 121–123.

Ключевые слова: киберпространство, большие данные, машинное обучение, логический процессор, malware, кубитная модель, синтез тестов, уязвимость, Cyber Security Computing.

ABSTRACT

Adamov Oleksandr Semenovych. Cyberspace protection models and methods based on big data analysis using machine learning. – Qualification scientific work as a manuscript.

Thesis for the degree of candidate of technical sciences (Ph.D.) in specialty 05.13.05 "Computer systems and components". – Kharkov National University of Radio Electronics, Ministry of Education and Science of Ukraine, Kharkov, 2019.

The purpose of the study is to significantly reduce the time of detection and blocking of cyber attacks aimed at the cyberspace of the subject, using the developed matrix models and logical methods of testing, testing and diagnosing by introducing computational redundancy into the cyberspace infrastructure.

Objectives of the study:

1) Improve the structural and logical models and methods of testing cyberspace for testing and diagnosing malicious components based on the use of deductive analysis of computing systems.

2) Develop signature-qubit methods for the synthesis of standard logic circuits of malware-functionalities and parallel modeling of malware-driven big data to determine the belonging of the current code to the existing destructive components in the malware library.

3) Develop a signature-qubit model of active online cybersecurity computing for monitoring the input streams of malware-data and managing the process of removing destructive components.

4) Improve cyberspace security by logical testing and diagnosing attacks and malicious components based on the use of machine learning algorithms.

5) Develop a method of attribute-based URL recognition using frequency patterns and a method for testing polymorphic malware based on the accounting of the Portable Executable checksums of sections in the executable file and using the data mining apparatus.

6) Perform testing and verification of the developed software for testing, checking and diagnosing malware by emulating attacks based on existing malware libraries.

The object of the study is cloud and edge-computing technologies based on high-performance data centers, computer gadgets, systems and networks that perform the service functions of storing and analyzing big data.

The subject of the research is structural-logical models, methods, tools for testing destructive components, protection of individual and collective service computing from cyber threats.

The scientific and practical task is to introduce software redundancy into the infrastructure of the computing space in the form of models, methods and software applications to significantly reduce the time to detect and block cyber attacks aimed at the subject's cyberspace by using the developed matrix models and logical methods of testing, testing and diagnosing.

The essence of the research is the development of models, methods and software applications to significantly reduce the time of detection and blocking cyberattacks aimed at the cyberspace of the subject, using the developed matrix models and logical methods of testing, checking and diagnosing by introducing computational redundancy into the cyberspace infrastructure.

The scientific novelty of research results:

1) Structural and logical models and methods for testing cyberspace for testing and diagnosing malicious components have been improved, which differ in using the method of deductive parallel analysis of a computing system to check and diagnose malware.

2) New methods for synthesizing reference logic circuits of malware-functionalities are proposed, which are characterized by the use of signature-qubit structures, which makes it possible to simulate malware-driven big data in parallel to determine whether the current code belongs to existing destructive components in the malware library.

3) A new model of active online cybersecurity computing has been developed, which is characterized by a signature-qubit representation of information, which makes it possible to increase the speed of monitoring the input malware-data streams and controlling the removal of destructive components.

4) Developing a method of attribute-based URL recognition using frequency patterns and a method for testing polymorphic malware based on the accounting of the Portable Executable checksums of sections in the executable file and using the data mining apparatus.

5) Cyberspace protection tools have been improved, which differ in the use of models and methods of signature-logic testing of attacks, the search for crypto-primitives in ransomware based on the use of machine learning algorithms, which makes it possible to significantly reduce the recovery time of the computing structure.

The practical significance of the results of research is determined by:

6) Testing, verification, and implementation of the developed software tools for checking and diagnosing malware and cyberattacks, which makes it possible to carry out their simulation with the assistance of existing applications and malware libraries.

7) Software implementation of the attribute-oriented URL recognition method using frequency patterns, which is distinguished by the use of the data mining apparatus, which makes it possible to determine the probabilistic risk assessment of the URL address based on its attributes.

8) Software implementation of the method for testing polymorphic malware, which is distinguished by the invariance to the determinism of signatures in the code and taking into account only the checksums of the Portable Executable (PE) sections in the executable file, which makes it possible to improve the performance of diagnosing destructive components.

The reliability and validity of scientific results is confirmed by the accuracy of detecting and classifying examples of zero-day threats, including new versions of ransomware and advanced threats (Advanced Persistent Threats - APT); the pos-

itive feedback from academics and experts at international cybersecurity conferences such as the Virus Bulletin 2015 in Prague, Czech Republic, 2018 in Montreal, Canada, the OpenStack Summit 2015 in Vancouver, Canada and the OpenStack Summit 2015 in Austin, Texas, USA, IEEE East-West Design & Test Symposium (EWDTS'2017), 2017 in Novy Sad, Serbia.

The results of the thesis as part of the models, methods, and infrastructure are introduced into the educational process of the Kharkov National University of Radio Electronics (implementation certificates dated from 20.05.2019, 21.05.2019); in the research and production activities of the Design & Test Lab company (reference 18.05.2019), educational process of the Blekinge Institute of Technology (BTH), Karlskruna, Sweden ('Statment of Reseach Results Impact on University Education Program' from 29.05.2019).

The results of the dissertation work are reflected in 31 publications: 3 sections in foreign monographs (1 of them are in the Scopus science database), 7 articles (5 of them in scientific journals included in the "List of specialized scientific publications of Ukraine"; 2 articles in international scientific journals abroad; 4 articles are included in the international scientometric databases), as well as 21 international scientific conferences (13 of them abroad, 12 are included in the Scopus scientometric database). The author has 13 publications in the Scopus scientometric database and the Hirsch index h = 3.

LIST OF PUBLICATIONS

Список публикаций соискателя, в которых опубликованы основные научные результаты диссертации:

1. Hahanov V., Gharibi W., Litvinova E., *Adamov A*. Cyber Physical Computing for IoT-driven Services. – New York. USA. – Springer, 2018. 279p. [Chapter 2. Multiprocessor Architecture for Big Data Computing [Text] / V. Hahanov, W. Gharibi, E. Litvinova, *A*. *Adamov*. P. 21-41] (Springer, Scopus).

2. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: Education

Challenges and Development of Advanced Malware Analysis Course [Text] / *A. Adamov*. P. 130-134].

3. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: The Cloud Threat Landscape [Text] / A. Carlsson, *A. Adamov*. P. 182-186].

4. *Adamov A.* Security risks and modern cyber security technologies for corporate networks [Text] / *A. Adamov*, V. Hahanov, W. Gharibi // Radioelektroniks and informatics. – 2010. – № 4. – P. 31–35. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

5. Хаханов В.И. Инфраструктура анализа и информационной безопасности киберпространства [Текст] / В.И. Хаханов, С.В. Чумаченко, Е.И. Литвинова, А.С. Мищенко, *А.С. Адамов* // Радиоэлектроника и информатика. – 2011. – №2 (53). – С. 40-60. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

6. *Адамов О.С.* Блокчейн інфраструктура для захисту кіберсистем [Текст] / *О.С. Адамов,* В.І. Хаханов, С.В. Чумаченко, В.Г. Абдуллаєв // Радиоэлектроника и информатика. – 2018. – №4 (83). – С. 64-85. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

7. *Адамов О.С.* Сигнатурно-кубітні методи розпізнавання деструктивних кодів [Текст] / *О.С. Адамов,* В.І. Хаханов // Радиоэлектроника и информатика. – 2019. – №1 (84). – С. 35-53. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

8. *Adamov A.* Analysis and Detection of Polymorphic Spyware [Text] / *A. Adamov*, A. Saprykin // Hakin9 Magazine. 2013. Vol. 8, № 01. Issue 01/2013 (61). Warsow: Software Press, 2013. – P. 6-11.

9. *Adamov A.S.*, Hahanov V.I. A Method for the Attribute-based Detection of URLs Using Frequency Patterns [Text] / Вестник Государственного Инженерного Университета Армении. Серия: Информационные Технологии, Электроника, Радиотехника. 2014. Вып. 17, No2. P. 59-66.

10. Сапрыкин А.С. Методика оценки убытков предприятия от вредоносных программ / Сапрыкин А.С., Бочарникова М.В., *Адамов А.С.* // Вестник национального технического университета "ХПИ" (Новое решение в современных технологиях). 2009. №8. С. 58-64.

Результаты, которые подтверждают апробацию материалов диссертации:

11. *Adamov A.* Electronic System Level Models for Functional Verification of System-on-Chip / *A. Adamov*, K. Mostovaya, Syzonenko, I et al. // Proc. of International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics” (CADSM). – Lviv-Polyana, Ukraine. – February 20-24, 2007. – P. 348–350. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

12. *Adamov A.* Transactional Data Analysis of Electronic System Level Models [Text] / *A. Adamov*, V. Hahanov, D. Melnyk et al. // Proc. of East-West Design and Test Symposium. September 7–10, 2007. Yerevan, Armenia. 2007. – P. 745–748.

13. *Adamov A.* Model of Source Code Analyzer for Hardware Description Languages [Text] / Dmytro Melnyk, Sergei Zaychenko, *Aleksandr Adamov*, Vladimir Hahanov // Proc. of East-West Design and Test Symposium (EWDTS). September 7–10, 2007, Yerevan, Armenia. – Yerevan, 2007. – P. 470–474.

14. Hahanova I.V. Transaction level model of embedded processor for vector-logical analysis [Text] / Hahanova I.V., Obrizan V., *Adamov A.* et al // Proc. of East-West Design and Test Symposium. September 2013. Rostov-on-Don,

Russia. – P. 1-4. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

15. *Adamov A.* Data Mining Techniques for Functional Verification of SoC [Text] / *A. Adamov*, R. Hwang, A. Gavrushenko // Proc. of International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008. – Lviv-Polyana, Ukraine. – February 20-24, 2008. – P. 557-559.

16. *Adamov A.* The Problem of Trojan Inclusions in Software and Hardware [Text] / *A. Adamov*, A. Saprykin // Proc. IEEE East-West Design and Test Symposium (EWDTS'2009). – Moscow, Russia. – 18-21 September 2009. – P. 449-451. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. *Adamov A.* The problem of Hardware Trojans detection in System-on-Chip [Text] / *A. Adamov*, A. Saprykin, D. Melnik // Proc. CAD Systems in Microelectronics (CADSM 2009), – Lviv-Polyana, Ukraine. – 24-28 Feb 2009. – P. 178-179. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. *Adamov A.* Security risks in hardware: Implementation and detection problem [Text] / *Adamov A.*, Hahanov V. // Proc. IEEE East-West Design and Test Symposium (EWDTS'2010), September 17–20, 2010, Saint Petersburg, Russia. – Piscataway, NJ : IEEE, 2010. – P. 425–427. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. *Adamov A.* A security model of individual cyberspace [Text] / *A. Adamov*, V. Hahanov // Proc. IEEE East-West Design and Test Symposium, September 19–20, 2011. – Sevastopol, Ukraine. – 2011. – P. 169–172. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. *Adamov A.* Discovering New Indicators for Botnet Traffic Detection [Text] / *A. Adamov*, V. Hahanov, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev,

2014. – P. 281–285. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov V. Structures for information retrieval in big data [Text] / V. Hahanov, S. Chumachenko, E. Litvinova, *A. Adamov* et al // Proc. of 13th International Conference Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2015), Lviv, Ukraine, 2015. – P. 70-75. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. *Adamov A.* A Sandboxing Method to Protect Cloud Cyberspace [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Test Symposium (EWDTS'2015). – Batumi, Georgia. – September 27-30, 2015. – P. 180–183. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

23. *Adamov A.* Android Ransomware: Turning CryptoLocker into CryptoUnlocker [Text] / *A. Adamov* // Proc. of the 25th Virus Bulletin International Conference, Prague, Czech Republic, 30 Sep-2 Oct 2015 – P. 220-223.

24. *Adamov A.* Detecting targeted attacks in the cloud [Text] / *A. Adamov* // Proc. of the OpenStack Summit, Vancouver, Canada, 18-22 May 2015. 1 p.

25. *Adamov A.* Using Open Source Security Architecture to Defend against Targeted Attacks / Alexander Adamov, Dan Lambright // Proc. of the OpenStack Summit, Austin, TX, US, April 25-29, 2016. 1 p.

26. *Adamov A.* Cloud incident response model [Text] / *Alexander Adamov*, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

27. *Adamov A.* The State of Ransomware. Trends and Mitigation Techniques [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – October 2, 2017, Novy Sad, Serbia. – P. 121–128. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

28. *Adamov A.* Battlefield Ukraine: finding patterns behind summer cyber attacks [Text] / *A. Adamov,* A. Carlsson // Proc. of the 27th Virus Bulletin International Conference, Madrid, Spain, 4-6 Oct 2017 – Appendix: Last-minute presentations. – P. 4-5.

29. *Adamov A.* Artificial Intelligence to Assist with Ransomware Cryptanalysis, [Text] // Proc. of the 28th Virus Bulletin International Conference, Montreal, Canada, 3-5 Oct 2018. – P. 289-292.

30. *Adamov A.* Creating Ransomware Decryptors [Text] // Proc. of 14th Cyber Security Conference UISGCON14, Kyiv, 2018. P. 3

31. *Адамов А.С.* Модель поиска вредоносного кода в программных продуктах [Текст] / А.С. Адамов, Д.А. Щербин, С.В. Чумаченко // Материалы 16-го Международного молодежного форума "Радиоэлектроника и молодежь в XXI веке", 17-19 апреля 2012. – Р. 121–123.

Keywords: cyberspace, big data, machine learning, logical processor, malware, qubit model, test synthesis, vulnerability, Cyber Security Computing.

ЗМІСТ

ВСТУП

В теперішній час спостерігається зростання кількості кібератак на державні і корпоративні (колективні) обчислювальні сервіси. Захищеність цифрової системи визначається захищеністю найбільш уразливого її компонента. Найбільш ефективним способом атак на колективні обчислювальні системи є пошук уразливого компонента (користувача, програмного або апаратного сервісу) шляхом збору інформації про нього (кіберрозвідку) і запуску атаки на обчислювальну систему за допомогою експлуатації виявленої технічної вразливості або за допомогою методів соціальної інженерії.

Під вразливістю цифрового сервісу, системи або людини слід розуміти несправність або слабкість, використовуючи яку можна навмисно порушити цілісність, доступність і конфіденційність інформації, з якою оперують зазначені суб'єкти.

Сукупність обчислювальних сервісів (C) і оброблюваної інформації (I) окремого суб'єкта (користувача, організації, країни) є кіберпростір (CS) даного суб'єкта:

$$CS = \{C, I\} = \cup_i \ (Ci, Ii), i = [0, N],$$

де N – кількість цифрових сервісів, які використовуються суб'єктом кіберпростору.

Таким чином, кажучи про захист кіберпростору, слід мати на увазі захист кожного з його сервісів, а також інформації, з якою дані сервіси оперують.

Методи захисту кіберпростору існують як правила захисту, що регламентують процеси доступу користувачів до інформації з обмеженим доступом, і інструментальні засоби: антивіруси, мережеві екрани, системи виявлення і запобігання вторгнень, що використовують інформацію про вже відомі загрози, показують свою неефективність при націлених атаках нульового дня.

Необхідні методи, які дозволять діагностувати атаку невідомого типу або нульового дня (0-day), наприклад, використовуючи евристичні алгоритми або моделювання в віртуальному (хмарному) середовищі з метою виявлення небезпечної поведінки на етапі тестування в демілітаризованій зоні.

Існуючі проблеми захисту кіберпростору:

1) Зростаюча кількість кібератак на кіберпростір.

2) Найбільш ефективними і руйнівними [Stuxnet 2010] є таргінг атаки нульового дня (невідомих на даний момент), які часом не в змозі відбити навіть самі антивірусні компанії.

3) Детерміністичні методи захисту кіберпростору засновані на використанні сигнатур вже відомих кібератак неефективні у разі атак нульового дня.

4) Евристичні методи не мають достатньої точності, що призводить до неприйнятного рівня помилкових спрацьовувань.

5) Поведінкові методи детектують кібератаку вже після її запуску і проникнення за периметр безпеки СК.

6) Методи на основі роботи емуляторів показали свою неефективність через невідповідність реального середовища емульованому, що дозволяє шкідливому коду легко виявляти факт підміни.

Найбільш вагомий внесок в наукові дослідження, що стосуються інформаційної безпеки та захисту від комп’ютерних загроз, внесли вчені: J. von Neumann, L. Penrose, F. Cohen, D.M. Chase, S.R. White, J.O. Kephart, F. Leitold, L. Adleman, B. Schneier, E. Chien, J. Bruce, J. Bates, R. Greenberg, R. Kusnierz, J. Hruska, F. Skulason, J. Norman, E. Spafford, E. Wilding, V. Bontchev, D. Ferbrache, S. Oxley, J. Norstad, S. Emery, M. Smith, K. van Wyk, S. Staniford, Liang-Jie Zhang, Jia Zhang, Hong Cai, Е. Касперский, С. Новиков, Горбенко І.Д, Харченко В.С.

Мета дослідження – істотне зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та

діагностування за рахунок введення обчислювальної надмірності в інфраструктуру кіберпростору.

Функція мети (Z) – мінімізація проміжку часу між моментом запуску атаки (A) на кіберпростір і моментом її діагностування (D), протягом якого обчислювальний сервіс залишається скомпрометованим (C), що одночасно дозволяє поліпшити якість сервісу шляхом забезпечення доступності, цілісності і конфіденіційності оброблюваної інформації на період атаки; мінімізації витрат на відновлення працездатності сервісу і фінансових втрат від його простою (TDT) за рахунок введення мінімально необхідної надмірності в інфраструктуру діагностування (I):

$$Z = F(TDT, TC, I) = \min[⅓(TDT + TC + I)],$$

де TC – час, на протязі якого обчислювальний сервіс залишається скомпрометованим з моменту запуску атаки зловмисником,

$$TC = t(D) – t(A),$$

де t (D) – момент детектування атаки, t (A) – момент запуску атаки.

Задачі дослідження:

1) Удосконалити структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів на основі використання дедуктивного аналізу обчислювальних систем.

2) Розробити сигнатурно-кубітні методи синтезу еталонних логічних схем malware-функціональностей і паралельного моделювання malware-driven великих даних для визначення приналежності поточного коду до існуючих деструктивних компонентів в malware бібліотеці.

3) Розробити сигнатурно-кубітну модель активного online cyber security комп'ютингу для моніторингу вхідних потоків malware-даних і управління процесом видалення деструктивних компонентів.

4) Удосконалити засоби захисту кіберпростору шляхом логічного тестування і діагностування атак і шкідливих компонентів на основі використання алгоритмів машинного навчання.

5) Розробити метод атрибутно-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів і метод перевірки поліморфних шкідливих програм на основі врахування контрольних сум Portable Executable секцій в виконувані файли і застосування апарату інтелектуального аналізу даних.

6) Виконати тестування і верифікацію розроблених програмних засобів тестування, перевірки та діагностування шкідливих програм шляхом емуляції атак на основі існуючих malware бібліотек.

Об'єкт дослідження – хмарні і edge-computing технології, засновані на високо-продуктивних дата-центрах, комп'ютерних гаджетах, системах і мережах, виконують сервісні функції зберігання та аналізу великих даних.

Предмет дослідження – структурно-логічні моделі, методи, засоби тестування деструктивних компонентів, захисту індивідуального та колективного сервіс-комп'ютингу від кіберзагроз.

Науково-практична задача – введення в інфраструктуру комп’ютингового простору програмної надмірності в формі моделей, методів і програмних додатків для істотного зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування.

Сутність дослідження – розробка моделей, методів і програмних додатків для істотного зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування за рахунок введення обчислювальної надмірності в інфраструктуру кіберпростору.

Методи досліджень подані апаратами: теорії множин і графів, бинарних матриць, інтеллектуального и статистического аналізу даних, мовами об’єктно-орієнтованого програмування – для розробки хмарного віртуального сервіса; прикладна теорія цифрових автоматів, булева алгебра, теорія множин, теорія графів, методи синтезу та аналізу структур даних – для побудови моделей Cyber Security Computing; векторно-логічний аналіз, теорія алгоритмів, мови опису, проектування та моделювання кіберфізичних систем – для синтезу та аналізу; методи і критерії якості створення комп’ютингових проектів – для тестування і верифікації програмно-апаратних компонентів інфраструктури хмарних сервісів.

Зв'язок роботи з науковими програмами, держбюджетними темами. Розробка теми дисертації здійснювалася відповідно до планів держбюджетних науково-дослідних робіт і міжнародних договорів, виконуваних на кафедрі Автоматизації проектування обчислювальної техніки ХНУРЕ в період з 2007 року, у тому числі: 1) Прикладна держбюджетна НДР № 216 «Енергозберігаючі інформаційні технології на основі паралельних обчислювальних процесів, безпровідних систем і мереж», 2007-2008, № ДР 0107U001540. 2) Договір про дружбу і співробітництво між ХНУРЕ та компанією «Aldec Inc.» (USA) № 04 від 01.11.2011. 3) Фундаментальна держбюджетна НДР № 232 «Теорія й проектування енергозберігаючих цифрових обчислювальних систем на кристалах, що моделюють і підсилюють функціональні можливості людини, 2009-2011, № ДР 0109U001646. 4) Фундаментальна держбюджетна НДР № 269 «Мультипроцесорна система пошуку, розпізнавання та прийняття рішень для інформаційної комп'ютерної екосистеми», 2011-2013, № ДР 0111U002956. 5) Фундаментальна держбюджетна НДР № 258 «Персональний віртуальний кіберкомп'ютер та інфраструктура аналізу кіберпростору», 2012-2014, № ДР 0112U000209. 6) Фундаментальна держбюджетна НДР № 297 “Кіберфізична система – «Розумне хмарне управління транспортом» (Cyber Physical System – Smart Cloud Traffic Control)”, 2015-2017, № ДР 0115U-000712 від

04.03.2015. 7) Фундаментальна держбюджетна НДР № 316 "Cyber Physical System – Smart Cyber University", 2017-2019, № ДР 0117U0002524. 8) Проект SEIDA BAITSE "Baltic Academic IT Security Exchange", Blekinge Institute of Technology, Sweden; 2011-2014. 9) Проект 530785-TEMPUS-1-2012-1-PL-TEMPUS-JPCR «Curricula Development for New Specialization: Master of Engineering in Microsystems Design (MastMEMS)» сумісно з університетом «Львівська політехніка», Київським національним університетом, Технічним університетом м. Лодзь (Польща), Ліонським університетом (Франція), Університетом м. Ільменау (Німеччина), Університетом м. Павія (Італія), 2012-2016. 10) Проект 544455-TEMPUS-1-2013-1-SE-TEMPUS-JPCR «Educating the Next Generation Experts in Cyber Security: the new EU-recognized Master's program (ENGENSEC)», 01 Dec 2013 – 30 Nov 2017.

Автор дисертаційної роботи брав участь при виконанні зазначених договорів і програм як розробник, менеджер і програміст кіберфізичної інфраструктури захисту кіберпростору у вігляді програмних засобів перевірки, діагностування шкідливих програм і атак, що дає можливість виконувати їх моделювання із залученням існуючих додатків і malware бібліотек.

Наукова новизна результатів досліджень:

1) Удосконалено структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів, які відрізняються використанням методу дедуктивного паралельного аналізу обчислювальної системи для перевірки та діагностування malware.

2) Запропоновано нові методи синтезу еталонних логічних схем mal-ware-функціональностей, які характеризуються використанням сигнатурно-кубітних структур, що дає можливість паралельно моделювати malware-driven великі дані для визначення приналежності поточного коду до існуючих деструктивних компонентів в malware бібліотеці.

3) Розроблено нову модель активного online cyber security комп'ютингу, яка характеризується сигнатурно-кубітним поданням інформації, що дає

можливість підвищувати швидкодію процесів моніторингу вхідних потоків malware-даних і управління видаленням деструктивних компонентів.

4) Удосконалено засоби захисту кіберпростору, які відрізняються використанням моделей і методів сигнатурно-логічного тестування атак, пошуку криптопримітивів в троянських програмах-шифрувальниках на основі використання алгоритмів машинного навчання, що дає можливість істотно зменшити час відновлення працездатності обчислювальної структури.

Практична значимість результатів досліджень полягає у наступному:

5) Тестуванням, верифікацією і впровадженням розроблених програмних засобів перевірки, діагностування шкідливих програм і атак, що дає можливість виконувати їх моделювання із залученням існуючих додатків і malware бібліотек.

6) Програмною реалізацією методу атрибутного-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів, який відрізняється застосуванням апарату інтелектуального аналізу даних, що дає можливість визначати вірогідну оцінку небезпеки URL-адреси на основі його атрибутів.

7) Програмною реалізацією методу перевірки поліморфних шкідливих програм, який відрізняється інваріантністю до детермінізму сигнатур в коді і урахуванням тільки контрольних сум Portable Executable (PE) секцій в виконуваних файлах, що дає можливість поліпшити продуктивність процедур діагностування деструктивних компонентів.

Отримані в процесі виконання досліджень наукові висновки і практичні результати є достовірними, що підтверджується точністю детектування і класифікацією прикладів загроз нульового дня, серед яких нові версії кріптолокеров і складних загроз (Advanced Persistent Threats - APT); позитивними відгуками вчених і фахівців на доповіді на міжнародних конференціях, присвячених кібербезпеки, таких як Virus Bulletin 2015 Празі, Чехія, 2018 у Монреалі, Канада, OpenStack Summit 2015 Ванкувері, Канада і

OpenStack Summit 2015 Остіні штат Техас, США, IEEE East -West Design & Test Symposium (EWDTS'2017) 2017 в Novy Sad, Serbia.

Результати дисертації у складі моделей, методів та інфраструктури впроваджені у навчальний процес Харківського національного університету радіоелектроніки (акти про впровадження від 20.05.2019, 21.05.2019); у науково-виробничу діяльність компанії Design & Test Lab (довідка від 18.05.2019), у навчальний процес Blekinge Institute of Technology (BTH), Karlskruna, Sweden (лист 'Statment of Reseach Results Impact on University Education Program' від 29.05.2019).

Особистий внесок здобувача. Всі наукові і практичні результати отримані автором особисто. У роботах, опублікованих зі співавторами, здобувачеві належать (аннотації, Додаток А, Б):

[1] – мультипроцесорна архітектура для обробки великих даних;

[2] – структурна модель malware-аналізу;

[3] – модель аналізу інформаційної безпеки кіберпростору;

[4] – технології кіберзахисту для корпоративних мереж;

[5] – інфраструктура кіберпростору для формування інформаційної безпеки;

[6] – удосконалені структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів, які відрізняються використанням методу дедуктивного паралельного аналізу обчислювальної системи для перевірки та діагностування malware;

[7] – методи синтезу еталонних логічних схем malware-функціональностей для паралельного моделювання malware-driven великих даних; модель активного online cyber security комп'ютингу;

[8] – метод визначення поліморфного шпигуна;

[9] – метод виявлення URL-адрес за допомогою частотних шаблонів;

[10] – аналітична модель оцінки збитків підприємства від шкідливих програм;

[11] – модель функціональної перевірки системи на кристалі;

[12] – опис програмних засобів для реалізації моделей системного рівня;

[13] – модель аналізатора вихідного коду для мов опису апаратури;

[14] – метод інтелектуального аналізу даних для функціональної перевірки системи на кристалі;

[15] – технологія інтелектуального аналізу даних для функціональної перевірки SoC;

[16] – опис та тестування програмних модулів для виявлення троянських включень у програмному забезпеченні;

[17] – метод детектування апаратних троянських включень;

[18] – методи діагностування і структури даних для інформаційного захисту;

[19] – модель захисту індивідуального кіберпростору;

[20] – аналіз і тестування програмних модулів;

[21] – структурна модель для отримання інформації у великих даних;

[22] – модель для захисту кіберпростору в хмарі;

[23] – структурна модель криптоперетворення та тестування програмних модулів для операційної системи Android;

[24] – метод виявлення таргетованих атак у хмарі;

[25] – архітектура безпеки з відкритим кодом для захисту від цільових атак;

[26] – опис програмних засобів для реалізації моделі реагування на хмарні інциденти;

[27] – тестування програмних модулів для реалізації методу пом'якшення;

[28] – аналіз кібератак;

[29] – застосування штучного інтелекту для криптоаналізу;

[30] – моделі дескрипторів;

[31] – модель пошуку шкідливого коду в програмних продуктах.

Апробація результатів дисертації. Результати роботи були представлені та обговорені на наступних конференціях: IEEE East-West Design and Test Symposium 2007, 2016 (Yerevan, Armenia), 2009 (Moscow, Russia), 2010 (Saint Petersburg, Russia), 2011 (Sevastopol, Ukraine), 2014 (Kyiv, Ukraine), 2015 (Batumi, Georgia), 2017 (Novi Sad, Serbia); International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008 (Lviv-Polyana, Ukraine); Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», 2012, (Харків, Україна); the 13th International Conference «The Experience of Designing and Application of CAD Systems in Microelectronics», CADSM 2007, 2009, 2015 (Lviv, Ukraine); the Virus Bulletin International Conference, 2015 (Prague, Czech Republic), 2017 (Madrid, Spain), 2018 (Montreal, Canada); OpenStack Summit, 2015 (Vancouver, Canada), 2016 (Austin, TX, USA); Cyber Security Conference UISGCON14, 2018 (Kyiv, Ukraine). Автор також брав участь у конкурсах інноваційних проектів та розробок, як запрошений експерт, спікер, член програмного комітету та журі, серед яких Міжнародна студентська конференція і конкурс наукових робіт з питань інформаційної безпеки «CyberSecurity for the Next Generation», 2011-2014, Kaspersky Lab.

Публікації. Результати дисертаційної роботи відображені в 31 друкованій праці: 3 розділи у закордонних монографіях (з них 1 входить до наукометричної бази Scopus), 7 статей (з них 5 – у наукових журналах, включених до «Переліку наукових фахових видань України»; 2 статті в міжнародних наукових журналах за кордоном; 4 статті входять до міжнародних наукометричних баз), а також у 21 міжнародній науковій конференції (з них 13 за кордоном, 12 входять до наукометричної бази Scopus). Здобувач має 13 публікацій у наукометричній базі Scopus та індекс Хірша h=3.

Дисертаційна робота має 241 сторінку (з них 218 представляють основний текст) і містить: 5 розділів, 48 рисунків, 14 таблиць, список джерел з 160 назв (на 16 с.), 4 додатки (на 16 с.), анотації на 27 с.

СПИСОК ОПУБЛІКОВАНИХ РОБІТ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Список публікацій здобувача, в яких опубліковані основні наукові результати дисертації:

1. Hahanov V., Gharibi W., Litvinova E., *Adamov A*. Cyber Physical Computing for IoT-driven Services. – New York. USA. – Springer, 2018. 279p. [Chapter 2. Multiprocessor Architecture for Big Data Computing [Text] / V. Hahanov, W. Gharibi, E. Litvinova, *A*. *Adamov*. P. 21-41] (Springer, Scopus).

2. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: Education Challenges and Development of Advanced Malware Analysis Course [Text] / *A. Adamov*. P. 130-134].

3. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: The Cloud Threat Landscape [Text] / A. Carlsson, *A*. *Adamov*. P. 182-186].

4. *Adamov A.* Security risks and modern cyber security technologies for corporate networks [Text] / *A. Adamov*, V. Hahanov, W. Gharibi // Radioelektroniks and informatics. – 2010. – № 4. – P. 31–35. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

5. Хаханов В.И. Инфраструктура анализа и информационной безопасности киберпространства [Текст] / В.И. Хаханов, С.В. Чумаченко, Е.И. Литвинова, А.С. Мищенко, *А.С. Адамов* // Радиоэлектроника и информатика. – 2011. – №2 (53). – С. 40-60. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

6. *Адамов О.С.* Блокчейн інфраструктура для захисту кіберсистем [Текст] / *О.С. Адамов,* В.І. Хаханов, С.В. Чумаченко, В.Г. Абдуллаєв // Радиоэлектроника и информатика. – 2018. – №4 (83). – С. 64-85. (Журнал рефе-

рується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

7. *Адамов О.С.* Сигнатурно-кубітні методи розпізнавання деструктивних кодів [Текст] / *О.С. Адамов,* В.І. Хаханов // Радиоэлектроника и информатика. – 2019. – №1 (84). – С. 35-53. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

8. *Adamov A.* Analysis and Detection of Polymorphic Spyware [Text] / *A. Adamov*, A. Saprykin // Hakin9 Magazine. 2013. Vol. 8, № 01. Issue 01/2013 (61). Warsow: Software Press, 2013. – P. 6-11.

9. *Adamov A.S.*, Hahanov V.I. A Method for the Attribute-based Detection of URLs Using Frequency Patterns [Text] / Вестник Государственного Инженерного Университета Армении. Серия: Информационные Технологии, Электроника, Радиотехника. 2014. Вып. 17, No2. P. 59-66.

10. Сапрыкин А.С. Методика оценки убытков предприятия от вредоносных программ / Сапрыкин А.С., Бочарникова М.В., *Адамов А.С.* // Вестник национального технического университета "ХПИ" (Новое решение в современных технологиях). 2009. №8. С. 58-64.

Результати, які засвідчують апробацію матеріалів дисертації:

11. *Adamov A.* Electronic System Level Models for Functional Verification of System-on-Chip / *A. Adamov*, K. Mostovaya, Syzonenko, I et al. // Proc. of International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics” (CADSM). – Lviv-Polyana, Ukraine. – February 20-24, 2007. – P. 348–350. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

12. *Adamov A.* Transactional Data Analysis of Electronic System Level Models [Text] / *A. Adamov*, V. Hahanov, D. Melnyk et al. // Proc. of East-West

Design and Test Symposium. September 7–10, 2007. Yerevan, Armenia. 2007. – P. 745–748.

13. *Adamov A.* Model of Source Code Analyzer for Hardware Description Languages [Text] / Dmytro Melnyk, Sergei Zaychenko, *Aleksandr Adamov*, Vladimir Hahanov // Proc. of East-West Design and Test Symposium (EWDTS). September 7–10, 2007, Yerevan, Armenia. – Yerevan, 2007. – P. 470–474.

14. Hahanova I.V. Transaction level model of embedded processor for vector-logical analysis [Text] / Hahanova I.V., Obrizan V., *Adamov A.* et al // Proc. of East-West Design and Test Symposium. September 2013. Rostov-on-Don, Russia. – P. 1-4. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

15. *Adamov A.* Data Mining Techniques for Functional Verification of SoC [Text] / *A. Adamov*, R. Hwang, A. Gavrushenko // Proc. of International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008. – Lviv-Polyana, Ukraine. – February 20-24, 2008. – P. 557-559.

16. *Adamov A.* The Problem of Trojan Inclusions in Software and Hardware [Text] / *A. Adamov*, A. Saprykin // Proc. IEEE East-West Design and Test Symposium (EWDTS'2009). – Moscow, Russia. – 18-21 September 2009. – P. 449-451. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. *Adamov A.* The problem of Hardware Trojans detection in System-on-Chip [Text] / *A. Adamov*, A. Saprykin, D. Melnik // Proc. CAD Systems in Microelectronics (CADSM 2009), – Lviv-Polyana, Ukraine. – 24-28 Feb 2009. – P. 178-179. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. *Adamov A.* Security risks in hardware: Implementation and detection problem [Text] / *Adamov A.*, Hahanov V. // Proc. IEEE East-West Design and Test Symposium (EWDTS'2010), September 17–20, 2010, Saint Petersburg,

Russia. – Piscataway, NJ : IEEE, 2010. – P. 425–427. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. *Adamov A.* A security model of individual cyberspace [Text] / *A. Adamov*, V. Hahanov // Proc. IEEE East-West Design and Test Symposium, September 19–20, 2011. – Sevastopol, Ukraine. – 2011. – P. 169–172. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. *Adamov A.* Discovering New Indicators for Botnet Traffic Detection [Text] / *A. Adamov*, V. Hahanov, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov V. Structures for information retrieval in big data [Text] / V. Hahanov, S. Chumachenko, E. Litvinova, *A. Adamov* et al // Proc. of 13th International Conference Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2015), Lviv, Ukraine, 2015. – P. 70-75. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. *Adamov A.* A Sandboxing Method to Protect Cloud Cyberspace [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Test Symposium (EWDTS'2015). – Batumi, Georgia. – September 27-30, 2015. – P. 180–183. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

23. *Adamov A.* Android Ransomware: Turning CryptoLocker into CryptoUnlocker [Text] / *A. Adamov* // Proc. of the 25th Virus Bulletin International Conference, Prague, Czech Republic, 30 Sep-2 Oct 2015 – P. 220-223.

24. *Adamov A.* Detecting targeted attacks in the cloud [Text] / *A. Adamov* // Proc. of the OpenStack Summit, Vancouver, Canada, 18-22 May 2015. 1 p.

25. *Adamov A.* Using Open Source Security Architecture to Defend against Targeted Attacks / Alexander Adamov, Dan Lambright // Proc. of the OpenStack Summit, Austin, TX, US, April 25-29, 2016. 1 p.

26. *Adamov A.* Cloud incident response model [Text] / *Alexander Adamov*, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

27. *Adamov A.* The State of Ransomware. Trends and Mitigation Techniques [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – October 2, 2017, Novy Sad, Serbia. – P. 121–128. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

28. *Adamov A.* Battlefield Ukraine: finding patterns behind summer cyber attacks [Text] / *A. Adamov,* A. Carlsson // Proc. of the 27th Virus Bulletin International Conference, Madrid, Spain, 4-6 Oct 2017 – Appendix: Last-minute presentations. – P. 4-5.

29. *Adamov A.* Artificial Intelligence to Assist with Ransomware Cryptanalysis, [Text] // Proc. of the 28th Virus Bulletin International Conference, Montreal, Canada, 3-5 Oct 2018. – P. 289-292.

30. *Adamov A.* Creating Ransomware Decryptors [Text] // Proc. of 14th Cyber Security Conference UISGCON14, Kyiv, 2018. P. 3

31. *Адамов А.С.* Модель поиска вредоносного кода в программных продуктах [Текст] / А.С. Адамов, Д.А. Щербин, С.В. Чумаченко // Материалы 16-го Международного молодежного форума "Радиоэлектроника и молодежь в XXI веке", 17-19 апреля 2012. – P. 121–123.

РОЗДІЛ 1
ОГЛЯД МОДЕЛЕЙ І МЕТОДІВ ЗАХИСТУ КІБЕРПРОСТОРУ

Наводиться аналітичний огляд існуючих моделей, методів і технологій захисту індивідуального сервіс-комп'ютингу. Визначаються переваги і недоліки найбільш затребуваних моделей і методів, опублікованих в спеціальній літературі: матеріалах конференцій і наукових журналах. На основі проведеного аналізу сформульовано мету і задачі дослідження, орієнтовані на усунення проблемних місць і недоліків існуючих моделей і методів у контексті їх реалізації в інфраструктурі захисту індивідуального сервіс-комп'ютингу.

1.1. Огляд моделей захисту кіберпростору

Розглянемо наступні моделі, що застосовані для захисту кіберпростору:

1. Модель стримування атак на кіберпростір [1];
2. Модель захисту персональних даних [2, 3];
3. Моделі управління доступом [4-8];
4. Моделі реагування на інциденти [9-13];
5. Моделі кіберзагроз [14-18].

Модель стримування кібератак на кіберпростір включає контроль над кіберзброєю і кіберрозвідку на національному рівні, захист приватних даних на індивідуальному і корпоративному рівнях [1].

Захист персональних даних представлена регуляторними моделями. Наприклад, General Data Protection Regulation (GDPR) для країн EU [2, 3]. GDPR положення являє собою юридичний інструмент для регулювання обробки персональних даних, що прийнятий в Європейському Союзі. Регламент схвалює стандартизацію. В цьому відношенні можна говорити про моделі захисту даних, з огляду на питання захисту персональних даних, з одного боку, а також етику і відповідальність в контексті кіберпростору – з

іншого. Недолік регуляторної моделі в тому, що приписи і стандарти мають на увазі самоконтроль за їх виконанням. Штрафні санкції вводяться тільки в разі, коли витік персональних даних став надбанням громадськості. У більшості випадків, подібні інциденти ховаються від користувачів, чиї персональні дані були скомпрометовані.

Існуючі моделі управління доступом Discretionary access control (DAC), Mandatory access control (MAC), Role-based access control (RBAC) описують права доступу для користувача тільки в рамках одного домена або організації [4, 5]. У мультидоменному середовищі можлива ситуація, коли в різних доменах визначені різні ролі і права доступу для різних акаунтів користувача, а також до різних об'єктів інформаційної інфраструктури, наприклад, файлів, процесів, мережевих ресурсів різних обчислювальних кластерів [6 - 8]. Таким чином, дані моделі не можуть бути використані в рамках кіберпростору, де у користувача існують різні уявлення, певні в різних доменах, і немає єдиної провайдер аутентифікації для всіх доменів і типів об'єктів.

Традиційні моделі реагування на інциденти, представлені в NIST 800-61 [9], ISO 27035 [10], SANS's Incident Handler's Handbook [11], модель загального реагування на інциденти [12], а також модель Agile [13] містять опис процесів, пов'язаних з виявленням, стримуванням, розслідуванням, аналізом, відновленням і запобіганням інцидентів безпеки в класичній інформаційній інфраструктурі. Однак всі вони не беруть до уваги особливості хмарної інфраструктури, такі як SDN та NFV, масштабованість, відмовостійкість, і віртуальний характер середовища.

Моделі кіберзагроз:

1. STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege). Найбільш популярна модель загроз від компанії Microsoft для розробників програмних додатків [14].

2. CAPEC (Common Attack Pattern Enumeration and Classification). Модель від MITRE для побудови периметра захисту організації на основі типових сценарієв атак [15].

3. Common vulnerability scoring system (CVSS). Модель оцінки загроз через наявність вразливості у системі [16].

4. ATT&CK (Adversary Tactics and Techniques and Common Knowledge) від MITRE. Глобально доступна база знань супротивної тактики та методик, заснована на спостереженнях у реальному світі. База знань ATT&CK використовується як основа для розробки конкретних моделей та методологій загроз у приватному секторі, уряді та у сфері продуктів і послуг кібербезпеки. [17]

5. OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation) є методологією для раціоналізації та оптимізації процесу оцінки ризиків інформаційної безпеки, щоб організація могла отримати достатні результати з невеликими інвестиціями у час, людей та інших обмежених ресурсів. Вона веде організацію до розгляду людей, технологій і засобів у контексті їхнього відношення до інформації та бізнес-процесів і послуг, які вони підтримують [18].

6. Модель розвиненої сталої загрози (Advanced Persistent Threat – APT) або кіберланцюг вбивства (Cyber Kill Chain). Вперше була запропонована компанією Mandiant, яка розкрила цілі та діяльність глобального актора APT1 [19]. Ці заходи включали в себе крадіжку сотень терабайт конфіденційних даних, у тому числі бізнес-планів, технологічних креслень і результатів тестування, щонайменше з 141 організацій у різних галузях. Вони оцінили середню тривалість збереження шкідливих програм у цільових організаціях в 1 рік. З того часу спостерігається зростаючий список документованих загроз APT із залученням на глобальну сцену потужних суб'єктів, включаючи суб'єктів національних держав. Розуміння мотивів та дій учасників APT відіграє важливу роль у вирішенні цих проблем. Для подальшого розуміння доповідь Мандіанта також запропонувала модель

життєвого циклу APT, також відома як кіберланцюг вбивства (cyber kill chain), що дозволяє отримати перспективу про те, які кроки вживає атакуюча сторона задля досягнення своїх цілей. Типова атака APT складається з таких етапів:

1. Кіберрозвідка. Збір даних про цілі.
2. Проникнення. Наприклад, завантаження шкідливого коду за допомогою попутного завантаження (drive-by-download) чи цільової фішингової (spear-phishing) атаки.
3. Фіксація у системі. Встановлення шкідливої програми.
4. Комунікації з центром командування і керування. Для цієї мети можуть використовуватися трояни віддаленого доступу (Remote Administration Tool – RAT).
5. Ескалацію привілеїв через експлуатацію вразливостей з метою отримання доступу до ресурсів організації.
6. Внутрішня кіберрозвідка, що робиться з метою знаходження серверів з цінною конфіденційною інформацією.
7. Скритне переміщення по внутрішній мережі.
8. Ексфільтрація чи вигрузка конфіденційної інформації.
9. Завершення атаки та самоліквідація або повернення до шагу 4 для отримання нових команд.

Таким чином, кіберланцюг вбивства забезпечує посилання для розуміння та відображення мотивів, цілей та дій учасників APT. Зараз існує більше 500 звітів APT [20].

1.2. Огляд методів захисту кіберпростору

Методи захисту, які використовуються в даний час в інфраструктурі захисту кіберпростору, представленої КСЗІ (Комплексна система захисту інформації) та КСАЗ (Комплексна система антивірусного захисту), можна умовно розділити на три групи по використовуваному підходу до детектування кіберзагроз:

– детерминистичний,

– імовірнісний.

1.2.1. Детерміністичний підхід

Детерміністичний підхід можна поділити на методи на основі зловживання, які виявляють поведінку, пов'язану з відомими атаками. Наприклад, такі методи описані в [21].

Методи на основі специфікації [22], [23], які виявляють атаки згідно політикам, визначеним експертами.

Методи на основі сигнатурних вердиктів використовують сигнатури, послудовності байт, що унікально ідентифікують атаку. Сигнатури зазвичай представлені у вигляді правил, що створюються експертами або спеціальними роботами і використовуються кібер спільнотою. У більшості випадків такі сигнатури можуть ефективно виявляти певну групу відомих атак. Однак ці підписи сприйнятливі до незначної модифікації коду, яка може бути викликана використуванням заплутування коду та шифруванням шкідливого коду.

Для швидкого генерування сигнатур на атаки за участі поліформних шкідливих программ використовуються такі методи:

1. Генерація підписів на основі мережі (NSG) [24]. Метод був запропонований як спосіб автоматичного і швидкого створення сигнатур для поліморфних хробаків на основі розробленої моделі NSG-PolyTree. Такі сигнатури і їхні варіанти схожі між собою, а деревоподібна структура може належним чином відображати їхню сімейну подібність.

2. Генерація сигнатури на основі навчання (Learning-based Signature Generation – LSG) включає методи, які шукають єдину велику інваріантну підрядку послідовностей байтів, а також методи, які шукають багато коротких інваріантних підрядків. Методи вилучення зразків є привабливими,

тому що сигнатури можуть генеруватися і ефективно узгоджуватися, і більш ранні роботи показали існування інваріантів в експлойтах. Але автори [25] показали фундаментальні обмеження на точність великого класу алгоритмів вилучення зразків у змагальній обстановці.

3. DeepSign. Метод глибокого навчання для автоматичного генерування та класифікації шкідливих програм. Цей метод використовує глибоку мережу переконань (Beep Belief Network – DBN), реалізовану з глибоким стеком автодекодерів, що генерує шуми, генеруючи інваріантне компактне представлення поведінки шкідливого програмного забезпечення.

Незважаючи на те, що для виявлення зловмисних програм звичайні підписи та методи, засновані на маркерах, не виявляють більшість нових варіантів існуючих шкідливих програм, результати, представлені в цій статті, показують, що сигнатури, створені DBN, дозволяють точно класифікувати нові варіанти шкідливих програм [25]. Проте результати DBN не можуть бути інтерпретовані, таким чином, порушуючи вимоги GDPR.

Запропонований в [27] підхід, націлений на раннє виявлення цільових атак з використанням логічних фільтрів, які в свою чергу приймають на вхід вердикти мережевих сигнатурних сканерів (Firewall L3,4,7 і Network IDS / IPS).

Запропонована методологія забезпечує виявлення індикаторів ранній стадії таргетованої атаки в мережевому трафіку. Проте, даний метод не був протестований на практиці. Автори обмежилися теоретичним моделюванням, що не дозволяє оцінити ефективність даного методу для захисту від реальних кібератак.

Популярним інструментом детектування кіберзагроз є Yara [28], який дозволяє створювати розширені сигнатури на основі регулярних виразів, використовуючи текстові і бінарні рядки в умовних виразах правил.

1.2.2. Ймовірнісний підхід та застосування машинного навчання

Імовірнісний підхід переважно використовує алгоритми машинного навчання, тому його слід розглядати в контексті методів і моделей, побудованих на основі алгоритмів машинного навчання.

Штучний інтелект (Artificial Intelligence – AI) – це наука, що дозволяє комп'ютеру автоматизувати те, що потрібно людині: інтелект, аналіз і прийняття рішень.

Машинне навчання (Machine Learning) – наука, що дозволяє комп'ютерам вчитися без явно запрограмованого на це. Машинне навчання застосовує статистику і алгоритми масштабування на великих обсягах даних. Одна з цілей машинного навчання – це досягнення штучного інтелекту.

Наука про дані (Data Science) – дисципліна, яка займається витягом інформації з даних. Наука про дані – це широке поле, що включає машинне навчання.

Серед алгоритмів машинного навчання, які застосовуються для вирішення завдань детектування кібератак, автори [29] виділяють наступні групи:

1. Supervised: Association Rule Classification (a); Artificial Neural Network (Deep Learning) (b); Support Vector Machines (c); Decision Trees (d); Bayesian Network (e); Hidden Markov Model (f); Kalman Filter (g); Bootstrap, Bagging, and AdaBoost (h); Random Forest (i).

2. Unsupervised: k-Means Clustering (a); Expectation Maximum (b); k-Nearest Neighbor (c); SOM ANN (d); Principal Components Analysis (e); Subspace Clustering (f).

3. Semi-supervised: Generative models (a); Low-density separation (b); Graph-based methods (c).

Алгоритми машинного навчання можуть працювати через навчання з вчителем або без. У навчанні з вчителем (supervised) алгоритм використовує додаткову інформацію та контекст, або дані для навчання надаються окремо,

в порядку, щоб машина стала більш розумною. У навчанні без вчителя (unsupervised), алгоритм має всю інформацію та контекст, з якого можно повністю зрозуміти надані навчальні дані до нього, щоб він міг сам вчитися. Існує також комбінований метод, де навчальні дані частково даються алгоритму машинного навчання.

Навчання з вчителем часто необхідно реалізовувати на наборі даних з доброякісними аномаліями, щоб передбачити майбутні аномалії, які не є доброякісними. Але в кібербезпеці високоякісні навчальні дані важко отримати через численні випадки помилкових спрацьовувань. Інженери в центрі операційної безпеки (SOC) повинні переглядати дані навчання і надавати коригування, такі як вказівка певних наборів подій, що представляють вагомі загрози безпеки, а інші - доброякісні. Тому очікується що фахівці завжди будуть необхідни для нагляду за навчанням для кібербезпеки.

Далі розглянемо приклади реалізацій методів на основі алгоритмів машинного навчання, що активно застосовуються у кібербезпеці.

Виявлення аномалій (викидів) є виявленням рідкісних подій, які викликають підозри, істотно відрізняючись від більшості даних [30].

Згідно з [31] алгоритми виявлення аномалій існують трьох типів:

1. Виявлення аномалій без вчителя – такі методи виявляють аномалії в неметованих наборах даних за умови, що більшість екземплярів у наборі даних є нормальними, шукаючи приклади, які найменш схожі на більшість даних.

2. Виявлення аномалій із вчителем – такі методи вимагають набір даних, які були б позначені як "нормальні" чи "ненормальні", і передбачає підготовку класифікатора (ключова відмінність від багатьох інших статистичних проблем класифікації полягає в незбалансованому характері виявлення викидів).

3. Виявлення аномалій із напівнаглядом – такі методи конструюють модель, що представляє звичайну поведінку з даного нормального набору

даних навчання, а потім перевіряє ймовірність того, що досліджувана модель буде згенерована досліджуваним екземпляром.

Методи, які найчастіше використовуються для виявлення аномалій:

1. Методики на основі щільності: a – k-найближчий сусід [32], [33], [34]; b – локальний фактор викиду [35]; c – ізоляційні ліси [36].

2. Виявлення викидів для високовимірних даних [37] на основі: a – підпростіру [38]; b – кореляції [39, 54]; c – тензора [40].

3. Однокласові машини опорних векторів (Support Vector Machines – SVM) [41].

4. Реплікатор нейронних мереж [42].

5. Байєсові мережі [42].

6. Приховані марковські моделі (HMM) [42].

7. Виявлення аномалій на основі кластерного аналізу [43], [44].

8. Відхилення від правил асоціації та частих наборів.

9. Виявлення аномалій на основі нечіткої логіки.

10. Ансамблі методів: а – беггінг [45, 46]; б – нормалізація балів [47, 48]; в – інші методи визначення різноманітності [49, 50].

Продуктивність різних методів багато в чому залежить від набору даних і параметрів, а методи мають незначні систематичні переваги перед іншими порівняно з багатьма наборами даних і параметрами [51].

Розглянемо на прикладах використання моделей машинного навчання для детектування аномалій.

У статті [52] подано алгоритм з висновком Байеса, який використовує переваги, засновані на сигнатурному методі і детектуванні аномалій. Запропонований підхід дозволяє витягувати патерни SQL-запитів у вигляді регулярного виразу, які можуть бути легко включені в будь-який механізм обробки правил (наприклад, NIDS Snort). Проте, даний підхід націлений на створення статичних сигнатур у вигляді правил, які дозволять детектувати лише певний в правилі спектр загроз.

Зіставлення зі зразком (pattern matching) – метод аналізу і обробки структур даних, заснований на виконанні певних умов в залежності від збігу досліджуваного значення з тим чи іншим зразком (шаблоном або патерном), в якості якого може використовуватися частина шкідливого коду чи мережевого трафіку.

Дінг, Фанг, та Чарленд з Університету Макгілла у своїй роботі [53] вирішують проблеми пошуку схожих частин асемблерного коду у шкідливих програмах захищених обфускацією та з використанням оптимізації коду під час компіляції. Пропонується спільно вивчити лексичні семантичні відносини та векторне представлення функцій складання на основі асемблерного коду. Розроблена модель представлення асемблерного коду Asm2Vec. Вона потребує лише кода збірки в якості вхідних даних і не вимагає попередніх знань, таких як правильне відображення між функціями складання. Метод може знайти і включити багаті семантичні відносини між токенами, що з'являються в коді збірки. Розроблений метод Asm2Vec вивчає векторне представлення функцій складання, розрізняючи його від інших. Asm2Vec не вимагає ніяких попередніх знань, таких як правильне відображення між функціями складання або використаним рівнем оптимізації компілятора. Модель вивчає лексичні семантичні відносини з'являються токенів у збірці коду, і представляє функцію складання як внутрішньо зваженої суміші прихованої семантики. Крім функцій складання, він може застосовуватися на різних гранулярностях послідовностей складання, таких як бінарні файли, фрагменти, основні блоки або функції. Авторами були проведені обширні експерименти з пошуку клонованого коду, у якому были використані різні опції оптимізації компілятора і методи обфускації. Результати показали, що Asm2Vec є точним і надійним проти сильної зміни в асемблерних інструкціях та графіку управління потоком.

Asm2Vec страждає від декількох обмежень. По-перше, він призначений для однієї мови Асемблер. Asm2Vec не застосовується безпосередньо до семантичних клонів через архітектури. Немає спільного лексично-

семантичного простору між двома різними мовами Асемблер. По-друге, поточний селективний механізм розширення не може визначити динамічні стрибки, такі як таблиця переходу. По-третє, це обмежена інтерпретативність. Asm2Vec не може пояснити або обґрунтувати свої результати, показуючи клоновані підграфи або доводячи символічну еквівалентність.

Дослідники з університетів Мічиган-Дірборн, Стоуні-Брук та Іллінойс в Чикаго [54] спробували вирішити проблему виявлення поточної кампанії APT (Advanced Persistent Threats) за допомогою *кореляційного аналізу*. APT атака складається з сукупності різнорідних кроків на багатьох хостах протягом тривалого періоду часу, в режимі реального часу та надання аналітику високорівневого пояснення сценарію нападу на основі хост-журналів та сповіщень IPS (Intrusion Prevention System) від підприємства. Існуючі системи IDS / IPS можуть виявляти та виробляти сповіщення про підозрілі події на хості. Проте, поєднання цих попереджень низького рівня з метою отримання картини високого рівня поточної кампанії APT залишається серйозною проблемою.

Недоліком розробленої системи HOLMES є аналіз лише логів системних викликів на вузлах мережі. У той час, коли індикатором атаки може стати події, інформація о котрих зберігається в логах файэрволів (L3-4, WAF), IDS/IPS, чи сервісу аутенфікації, наприклад під час DDoS чи bruteforce атак, які взагалі не є частиною APT моделі ланцюга вбивства.

Метод *асоціативних правил*, використаний у роботі [59], вирішує проблему великої кількості помилкових позитивних сигналів тривоги, які генеруються у великих інфраструктурах для виявлення вторгнень, ускладнює розділення помилкових оповіщень від реальних атак. Одним із засобів зменшення цієї проблеми є використання метасигналів або правил, які ідентифікують відомі шаблони атак у потоках сигналів тривоги. Очевидний ризик при такому підході полягає в тому, що база правил не може бути повною стосовно кожного профілю справжньої атаки, особливо тих, які є

новими. Зараз нові правила відкриваються вручну, процес, який є дорогим і схильним до помилок. Дослідники представляють новий підхід, що використовує видобуток правил асоціації, щоб скоротити час, що минув від появи нового профілю атаки в даних до його визначення, як правило, в інфраструктурі моніторингу органызації.

Недоліком методу є обмежений обсяг проведених експеріментів та необхідність апріорних знань про атаку задля виявлення асоціацій, побудови ланцюга атаки, та генерації правил. Автори також досліджують сигнали тривоги від мережевої системи виявлення вторгнень (Network IDS/IPS) для видобичи асоціативних правил.

У машинному навчанні та обробці природної мови, тематична модель є типом статистичної моделі для виявлення абстрактних «тем», які відбуваються в наборі документів. Тематичне моделювання є часто використовуваним інструментом видобування тексту для виявлення прихованих семантичних структур у текстовому тілі. Інтуїтивно, враховуючи те, що документ стосується певної теми, можна очікувати, що окремі слова з'являться в документі більш-менш часто [60].

У [61] автори аналізують повідомлення в блогах для різних категорій загроз кібербезпеки, пов'язаних з виявленням кібератак, кіберзлочинів і тероризму. Існуючі дослідження з аналізу інтелекту зосереджувалися на аналізі новин або форумів для інцидентів кібербезпеки, але лише деякі з них дивилися на веб-журнали чи Інтернет блоги. Автори використовують ймовірнісний латентний семантичний аналіз для виявлення ключових слів з веб-журналів кібербезпеки стосовно певних тем. В роботі продемонстровано, як цей метод може представити блогосферу з точки зору тематики з вимірними ключовими словами, таким чином відстежуючи популярні розмови та теми в блогосфері. Застосовуючи імовірнісний підхід, можливо поліпшити пошук інформації в мережі Інтернет і виявлення ключових слів, а також забезпечити аналітичну основу для майбутнього аналізу блогосфери. Недоліком роботи є її спрямованність лише на аналіз блогосферу.

Однак, тематичне модулювання можна використати не тільки задля захисту, але ж й для таргетованої атаки. Як це зробити, показали дослідники з SecureData Labs в рамках свого докладу на конференції RSA Conference 2019 [62], де вони, проаналізував тематику документів жертви, змогли використати цю інформацію для цільової фішингової атаки.

Таким чином, моделі машинного навчання, що побудовані з урахуванням моделей загроз, представляють ефективні інструменти для автоматизації виявлення кібератак на кіберпростір, підвищуючи обороносппроможність організації. При виборі моделі необхідно керуватися не тільки показниками її ефективності, але і типом навчання моделі: з учителем або без; інтерпретованістю результатів і прозорістю моделі – вимоги The EU General Data Protection Regulation (GDPR) [63].

1.2.3. Атакі на методи машинного навчання

У контексті методів машинного навчання слід згадати, що деякі методи самі по собі вразливі до атак. Так, у роботі [64] доводиться приклад такої атаки. Автори продемонстрували, що спільне машинне навчання та пов'язані з нею методи, такі як федеративне навчання можуть привести до ненавмисного витоку інформації про навчальні дані учасників через оновлення моделі. Таким чином, це дозволяє розвивати пасивні та активні атаки виводу для використання цього витоку.

Автори [65] стверджують, що моделі глибокого навчання (DL – Deep Learning) є також вразливими до змагальних прикладів. Тобто до зловмисно створених вхідних даних, що призводять до неправильної поведінки цільових моделей DL, що значно ускладнює застосування DL у доменах, що чутливі до безпеки. У своїй роботі автори представляють розробку, реалізацію та оцінку DEEPSEC, єдиної платформи, яка має на меті подолати цей пробіл. У своїй нинішній реалізації DEEPSEC об'єднує 16 найсучасніших атак з 10

показниками та 13 найсучасніших засобів захисту з 5 показниками оборонної корисності.

В роботі [66] було показано, що машинне навчання та глибокі нейронні мережі можуть бути обдурені атаками ухилення (також називаються прикладами змагальності), тобто малими змінами вхідних даних, які викликають помилкову класифікацію під час тестування. Ця робота вісвітлює уразливість методів виявлення шкідливих програм, які використовують глибокі мережі для вивчення з сирих байтів на прикладі атаки на основі градієнта, яка здатна уникнути нещодавно запропонованої глибокої мережі, пристосованої для цієї мети, лише змінюючи кілька конкретних байтів в кінці кожного зразка шкідливого програмного забезпечення, зберігаючи при цьому його шкідливу функцію. Таким чином змагальні зловмисні програмні файли ухиляються від детектування у цільовій мережі з великою ймовірністю, навіть якщо менше 1% їхніх байтів змінено.

Автори [67] представили першу надійну та узагальнюючу систему виявлення та пом'якшення атак на DNN на основі методів, які ідентифікують бекдори, описані в [68 - 72], і реконструюють можливі тригери атак. Вони ідентифікують кілька методів пом'якшення за допомогою вхідних фільтрів, обрізання нейронів і відривання від навчання. У роботі демонструється їх ефективність за допомогою великих експериментів на різних DNN, проти двох типів ін'єкцій, визначених попередньою роботою: атаку з повним доступом до навчальної моделі, і троянська атака, керована нейронами, без доступу до моделі навчання. Запропоновані методи також виявляють надійність у відношенні ряду варіантів бекдор атаки.

Недоліками метода виявлення атак на DNN є проблема узагальнення за межами поточного домену. Методи виявлення / пом'якшення можуть бути узагальнюючими: інтуїція для виявлення полягає в тому, що інфікована мітка є більш вразливою, ніж неінфіковані мітки, і це повинно не залежити від

домену. Проблема адаптації моделі до домену потребує сформулювати процес атаки бекдора і розробити метрику, що вимірює, наскільки вразлива конкретна мітка. Інша проблема запропонованого метода це великий простір потенційних зустрічних заходів зловмисника, які неможливо охопити в рамках одного дослідження.

Таким чином, моделі на основі нейронних мереж (ANN, DNN), володіючи низькою інтерпретованих результатів і стійкістю до атак на моделі машинного навчання, не можуть бути використані як надійний засіб виявлення кібератак реального світу.

1.2.4. Оцінка методів виявлення кібератак

Точність (Precision) та повнота (Recall, Sensetivity, True Positive Rate), які використовуються в бінарній класифікації, є найбільш адекватними критеріями оцінювання для проблем виявлення кібератак.

Для таких проблем, *точність* – це міра того, наскільки точним є класифікатор, що виявив атаку. Більше значення точності відповідає меншій кількості помилкових тривог (FP – False Positives). У той час як, *повнота* (Recall) показує, скільки атак класифікатор фактично виявив. Більш високе значення повноти відповідає меншій кількості пропущених атак (FN – False Negative). В ідеалі ми хочемо мати класифікатор з високою точністю і повнотою, оскільки це відповідає низьким значенням FP і FN [73].

Точність та повнота обчислюються згідно формул:

$$\text{Precision} = TP/(TP+FP), \quad \text{Recall} = TP/(TP+FN), \tag{1.2}$$

де TP – істинно-позитивне рішення, тобто кількість атак, коректно виявлених класифікатором; TN – істинно-негативне рішення, тобто кількість доброякісних подій, коректно виявлених класифікатором; FP – хибно-

позитивне рішення, тобто кількість доброякісних подій, помилково виявлених класифікатором, як кібератака; FN – хибно-негативне рішення, тобто кількість невиявлених класифікатором атак.

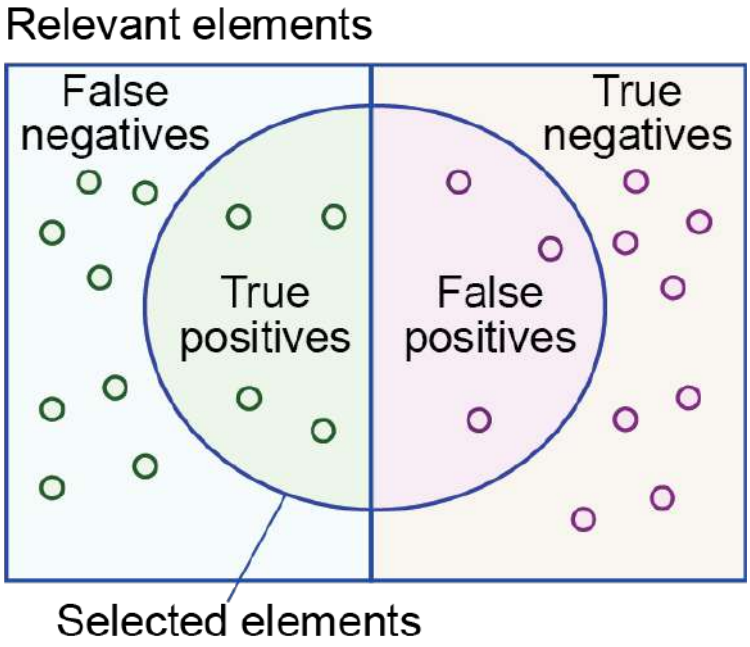


Рисунок 1.1 – Діаграма Ейлера, що показує відношення множин рішень класифікатора

У багатокласовому сценарії точність і повнота розраховуються окремо для кожного класу. Щоб розрахувати ці метрики для певного класу, інші класи розглядаються як один клас. Це також називається “одним проти всіх”. Нарешті, точність і повнота для всіх класів об'єднуються разом, використовуючи середньозважену величину [74].

Існують також інші показники, наприклад, F-міра чи міра Ван Ризбергена, але вони рідко застосовуються на практиці для оцінки методів виявлення кібератак, тому ми не будемо їх використовувати у роботі.

1.3. Аналіз великих даних

Генерація великих об’ємів даних (від 10 ТВ до 1 РВ на день) у комп'ютерних мережах та на вузлах великої організації створюють проблему для виявлення кіберзагроз класичними системами протидії кіберзагрозам та перетворює ці традиційні рішення на застарілі.

Хоча аналіз журналів, мережевих потоків і системних подій для криміналістики та виявлення вторгнень був проблемою в спільноті інформаційної безпеки протягом десятиліть, традиційні технології не завжди можуть зберігати аналітичні дані протягом тривалого часу та робити пошук у них.

По-перше, збереження великої кількості даних раніше не було економічно доцільним. Як наслідок, у традиційних інфраструктурах більшість журналів подій та інших записаних комп'ютерних операцій видалялось після фіксованого періоду зберігання (наприклад, 2-3 місяці). По-друге, виконання аналітичних і складних запитів на великих, неструктурованих наборах даних з неповними або зашумленими атрибутами було неефективним.

Для вирішення цієї проблеми застосовують методи Big Data Analytics (BDA). BDA може допомогти в реальному часі виявляти шкідливі та підозріли дії. Таким чином, ця технологія дозволяє посилити традиційні методи кібербезпеки [75].

Наприклад, BDA дозволяє виявляти банківські шахрайства та застосовувати системи запобігання вторгнень на основі виявлення аномалій (Intrusion Prevention System – IDS). Нові інструменти керування інформацією та подіями безпеки (SIEM) були розроблені для аналізу та управління неструктурованими даними, оскільки вони можуть ефективно чистити, готувати та запитувати дані в різнорідних, неповних та зашумлених форматах даних.

Виявлення шахрайства є одним з найбільш помітних способів використання аналітики великих даних: кредитні картки та телефонні компанії проводили широкомасштабне виявлення шахрайства протягом десятиліть. Однак, спеціально побудована інфраструктура, необхідна для розкриття великих даних для виявлення шахрайства, не була достатньо економічною для широкомасштабного прийняття.

Одним з основних наслідків технологій великих даних є те, що вони сприяють широкому колу галузей промисловості для створення доступних інфраструктур для моніторингу безпеки. Великі засоби передачі даних також особливо підходять для того, щоб стати фундаментальними для вдосконаленого виявлення просунутої сталої загрози (Advanced Persistent Threat – APT) та цифрової криміналістики [76].

APTs працюють в повільному режимі, тобто з малою кількістю подій та довгостроковим виконанням. Такі атаки можуть відбуватися протягом тривалого періоду часу, тоді як вторгнення залишається поза увагою жертви. Щоб виявити ці атаки, необхідно зібрати та зіставити велику кількість різноманітних даних, включаючи внутрішні джерела даних та зовнішні спільні дані розвідки, і виконати довгострокову історичну кореляцію для включення апостеріорної інформації про атаку в мережевій історії [77].

Проте недоліком всіх цих методів є відсутність способів отримання апріорної інформації про кібератаки або критеріїв виявлення аномалій, які можуть відрізнятися в залежності від джерела даних.

1.4. Інфраструктура захисту кіберпростору

Інфраструктура захисту кіберпростору може включати наступні компоненти:

1. Системи розмежування доступу до інформації;
2. Криптографічні системи;
3. Системи ідентифікації та автентифікації;
4. Системи аудиту та моніторингу;

5. Система виявлення та попередження вторгнень.

У цій роботі ми розглянемо системи активного захиста від кібератак, до яких можна віднести системи виявлення та попередження вторгнень. Система виявлення та попередження вторгнень може бути встановлена:

1. На кінцевій точці, наприклад Host Intrusion Prevention System (HIPS) або антивірус;

2. У мережі, наприклад Network Intrusion Prevention System (NIPS);

3. На контроллері безпеки (у хмарі), який збирає інформацію з хостів та мереж, наприклад Next-Gen SIEM або SOAR.

На сьогодняшній день системи активного захиста від кібератак поєднують з системами аудиту та моніторингу стану кібербезки кіберпростору організії з метою віиявлення аномалій в агрегованих даних та під час їх обробки, які можуть свідчити про наявність активної кіберзагрози. Найчастіше використовують системи безпеки та управління подіями (Security Information and Event Management -- SIEM), такі як: Splunk [55], LogRhythm [56], AlienVault OSSIM [57] та IBM QRadar [58], що допомагають робити кореляцію сигналів тривоги. Ці системи збирають журнал подій і сповіщень з різних джерел і корелюють їх. Таке співвідношення часто використовує доступні індикатори, наприклад, часові мітки.

Нова генерація SIEM рішень включає також оркестрацію інфраструктури безпеки та автоматизоване реагування на інциденти (Security Orchestration, Automation and Response – SOAR).

Ці методи кореляції є корисними, але часто не вистачає розуміння складних відносин, які існують між попередженнями і фактичними випадками вторгнення, і точності, необхідної для узгодження кроків атаки, які відбуваються на різних хостах протягом тривалих періодів часу (тижні, або в деяких випадках, місяці).

Тому у SIEM та SOAR системах впроваджується методи обробки великих даних на основі використання штучного інтелекту (AI) та машинного навчання (ML). Також, через потребу зберігати великі об’єми

даних впродовж 6-12 місяців, а також шукати в них інформацію про потенційну кібератаку, ці сервіси розгортають у хмарному середовищі.

1.4. Висновки до розділу 1

Мета дослідження – істотне зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування за рахунок введення обчислювальної надмірності в інфраструктуру кіберпростору.

Задачі дослідження:

1) Удосконалити структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів на основі використання дедуктивного аналізу обчислювальних систем.

2) Розробити сигнатурно-кубітні методи синтезу еталонних логічних схем malware-функціональностей і паралельного моделювання malware-driven великих даних для визначення приналежності поточного коду до існуючих деструктивних компонентів в malware бібліотеці.

3) Розробити сигнатурно-кубітну модель активного online cyber security комп'ютингу для моніторингу вхідних потоків malware-даних і управління процесом видалення деструктивних компонентів.

4) Удосконалити засоби захисту кіберпростору шляхом логічного тестування і діагностування атак і шкідливих компонентів на основі використання алгоритмів машинного навчання.

5) Розробити метод атрибутно-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів і метод перевірки поліморфних шкідливих програм на основі врахування контрольних сум Portable Executable секцій в виконувані файли і застосування апарату інтелектуального аналізу даних.

6) Виконати тестування і верифікацію розроблених програмних засобів тестування, перевірки та діагностування шкідливих програм шляхом емуляції атак на основі існуючих malware бібліотек.

1.5. Список використаних джерел до розділу 1

1. Cybersecurity Dilemmas: Technology, Policy, and Incentives: Summary of Discussions at the 2014 Raymond and Beverly Sackler U.S.-U.K. Scientific Forum, National Academy of Science, 2014.

2. P De Hert, V Papakonstantinou, The proposed data protection Regulation replacing Directive 95/46/EC: A sound system for the protection of individuals, Computer Law & Security Review, Elsevier, 2012.

3. E. Lachaud, The General Data Protection Regulation and the rise of certification as a regulatory instrument, Computer Law & Security Review, Volume 34, Issue 2, April 2018, Pages 244-256.

4. Bokefode J.D, Ubale S. A, Apte Sulabha S, Modani D. G, Analysis of DAC MAC RBAC Access Control based Models for Security, International Journal of Computer Applications, Volume 104–No. 5, October 2014.

5. Luo L., He H., Zhu J. (2016) Defect Analysis and Risk Assessment of Mainstream File Access Control Policies. In: Wang G., Ray I., Alcaraz Calero J., Thampi S. (eds) Security, Privacy, and Anonymity in Computation, Communication, and Storage. SpaCCS 2016. Lecture Notes in Computer Science, vol 10066. Springer.

6. Elsayed W., Gaber T., Zhang N., Ibrahim Moussa M. (2016) Access Control Models for Pervasive Environments: A Survey. In: Gaber T., Hassanien A., El-Bendary N., Dey N. (eds) The 1st International Conference on Advanced Intelligent System and Informatics (AISI2015), November 28-30, 2015, Beni Suef, Egypt. Advances in Intelligent Systems and Computing, vol 407. Springer.

7. Li, B., Tian, M., Zhang, Y., Lv, S.: Strategy of domain and cross-domain access control based on trust in cloud computing environment. In: Computer Engineering and Networking, pp. 791–798. Springer (2014)

8. Cha, B., Seo, J., Kim, J.: Design of attribute-based access control in cloud computing environment. In: Proceedings of the International Conference on IT Convergence and Security 2011, pp. 41–50 (2012)

9. Computer Security Incident Handling Guide, NIST 800-61, Sep 2016, http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf

10. Information security incident management (ISO/IEC 27035-1:2016), Sep 2016 https://www.iso.org/obp/ui/#iso:std:iso-iec:27035:-1:ed-1:v1:en

11. Incident Handler's Handbook, SANS Institute, Sep 2016, https://www.sans.org/reading-room/whitepapers/incident/incident-handlers-handbook-33901

12. Felix C. Freiling, Bastian Schwittay, A Common Process Model for Incident Response and Digital Forensics, IMF 2007, Stuttgart, September 2007, http://www.imf-conference.org/imf2007/2%20Freiling%20common_model.pdf

13. G. Grispos, W. B. Glisson, T. Storer, Rethinking Security Incident Response: The Integration of Agile Principles, Sep 2016, https://arxiv.org/ftp/arxiv/papers/1408/1408.2431.pdf

14. A. Shostack, Threat Modeling: Designing for Security, Wiley, 2014, p. 626

15. CAPEC: Common Attack Pattern Enumeration and Classification. https://capec.mitre.org/index.html, 2019

16. Common vulnerability scoring system (CVSS) v3.0: Specification document, https://www.first.org/cvss/specification-document, 2019

17. Adversary Tactics and Techniques and Common Knowledge, MITRE, https://attack.mitre.org/, 2019.

18. Richard A. Caralli, James F. Stevens, Lisa R. Young, William R. Wilson, Introducing OCTAVE Allegro: Improving the Information Security Risk Assessment Process, Software Engineering Institute, 2007.

19. MANDIANT: Exposing One of China’s Cyber Espionage Units. https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf, 2019

20. Колекція APT звітів, Github, https://github.com/aptnotes/data, 2019.

21. Phillip A Porras and Richard A Kemmerer. Penetration state transition analysis: A rule-based intrusion detection approach. In Computer Security Applications Conference, 1992. Proceedings., Eighth Annual, pages 220–229. IEEE, 1992.

22. Calvin Ko, Manfred Ruschitzka, and Karl Levitt. Execution monitoring of security-critical programs in distributed systems: A specification-based approach. In S&P. IEEE, 1997.

23. Prem Uppuluri and R Sekar. Experiences with specification-based intrusion detection. In RAID. Springer, 2001.

24. Yong Tang ; Bin Xiao ; Xicheng Lu, Signature Tree Generation for Polymorphic Worms, IEEE Transactions on Computers (Volume: 60 , Issue: 4 , April 2011, pp 565 - 579, DOI: 10.1109/TC.2010.130.

25. S. Venkataraman, A. Blum, D. Song, Limits of Learning-based Signature Generation with Adversaries. NDSS, The Internet Society, 2008.

26. E. David, N. S. Netanyahu, DeepSign: Deep Learning for Automatic Malware Signature Generation and Classification. International Joint Conference on Neural Networks (IJCNN), pages 1–8, Killarney, Ireland, July 2015.

27. S. Japertas, T. Baksys, Method of Early Staged Cyber Attacks Detection in IT and Telecommunication Networks, ELEKTRONIKA IR ELEKTROTECHNIKA, ISSN 1392-1215, VOL. 24, NO. 3, 2018.

28. Yara, https://virustotal.github.io/yara/, 2019.

29. S. Dua, X. Du, Data Mining and Machine Learning in Cybersecurity, CRC Press, 2011. - P. 23-157.

30. Zimek A., Schubert E. Outlier Detection, Encyclopedia of Database Systems, Springer New York, pp. 1–5, 2017.

31. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. ACM Computing Surveys. 41 (3): 1–58, 2009.

32. Knorr, E. M.; Ng, R. T.; Tucakov, V. (2000). "Distance-based outliers: Algorithms and applications". The VLDB Journal the International Journal on Very Large Data Bases. 8 (3–4): 237–253.

33. Ramaswamy, S.; Rastogi, R.; Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. Proceedings of the 2000 ACM SIGMOD international conference on Management of data – SIGMOD '00. p. 427.

34. Angiulli, F.; Pizzuti, C. (2002). Fast Outlier Detection in High Dimensional Spaces. Principles of Data Mining and Knowledge Discovery. Lecture Notes in Computer Science. 2431. p. 15.

35. Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; Sander, J. (2000). LOF: Identifying Density-based Local Outliers (PDF). Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. SIGMOD. pp. 93–104.

36. Liu, Fei Tony; Ting, Kai Ming; Zhou, Zhi-Hua (December 2008). Isolation Forest. 2008 Eighth IEEE International Conference on Data Mining. pp. 413–422.

37. Zimek, A.; Schubert, E.; Kriegel, H.-P. (2012). "A survey on unsupervised outlier detection in high-dimensional numerical data". Statistical Analysis and Data Mining. 5 (5): 363–387.

38. Kriegel, H. P.; Kröger, P.; Schubert, E.; Zimek, A. (2009). Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data. Advances in Knowledge Discovery and Data Mining. Lecture Notes in Computer Science. 5476. p. 831.

39. Kriegel, H. P.; Kroger, P.; Schubert, E.; Zimek, A. (2012). Outlier Detection in Arbitrarily Oriented Subspaces. 2012 IEEE 12th International Conference on Data Mining. p. 379.

40. Fanaee-T, H.; Gama, J. (2016). "Tensor-based anomaly detection: An interdisciplinary survey". Knowledge-Based Systems. 98: 130–147.

41. Schölkopf, B.; Platt, J. C.; Shawe-Taylor, J.; Smola, A. J.; Williamson, R. C. (2001). "Estimating the Support of a High-Dimensional Distribution". Neural Computation. 13 (7): 1443–71.

42. Hawkins, Simon; He, Hongxing; Williams, Graham; Baxter, Rohan (2002). "Outlier Detection Using Replicator Neural Networks". Data Warehousing and Knowledge Discovery. Lecture Notes in Computer Science. 2454. pp. 170–180.

43. He, Z.; Xu, X.; Deng, S. (2003). "Discovering cluster-based local outliers". Pattern Recognition Letters. 24 (9–10): 1641–1650.

44. Campello, R. J. G. B.; Moulavi, D.; Zimek, A.; Sander, J. (2015). "Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection". ACM Transactions on Knowledge Discovery from Data. 10 (1): 5:1–51.

45. Lazarevic, A.; Kumar, V. (2005). Feature bagging for outlier detection. Proc. 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. pp. 157–166.

46. Nguyen, H. V.; Ang, H. H.; Gopalkrishnan, V. (2010). Mining Outliers with Ensemble of Heterogeneous Detectors on Random Subspaces. Database Systems for Advanced Applications. Lecture Notes in Computer Science. 5981. p. 368.

47. Kriegel, H. P.; Kröger, P.; Schubert, E.; Zimek, A. (2011). Interpreting and Unifying Outlier Scores. Proceedings of the 2011 SIAM International Conference on Data Mining. pp. 13–24.

48. Schubert, E.; Wojdanowski, R.; Zimek, A.; Kriegel, H. P. (2012). On Evaluation of Outlier Rankings and Outlier Scores. Proceedings of the 2012 SIAM International Conference on Data Mining. pp. 1047–1058.

49. Zimek, A.; Campello, R. J. G. B.; Sander, J. R. (2014). "Ensembles for unsupervised outlier detection". ACM SIGKDD Explorations Newsletter. 15: 11–22.

50. Zimek, A.; Campello, R. J. G. B.; Sander, J. R. (2014). Data perturbation for outlier detection ensembles. Proceedings of the 26th International Conference on Scientific and Statistical Database Management – SSDBM '14. p. 1.

51. Campos, Guilherme O.; Zimek, Arthur; Sander, Jörg; Campello, Ricardo J. G. B.; Micenková, Barbora; Schubert, Erich; Assent, Ira; Houle, Michael E. (2016). "On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study". Data Mining and Knowledge Discovery. 30 (4): 891.

52. R. Kozik, M. Choraś, Machine Learning Techniques for Cyber Attacks Detection, Image Processing and Communications Challenges 5, 2014. - P. 391-398.

53. S. H. H. Ding, B. C. M. Fung, P. Charland, Asm2Vec: Boosting Static Representation Robustness for Binary Clone Search against Code Obfuscation and Compiler Optimization, Proc. of 40th IEEE Symposium on Security and Privacy, 2019.

54. S. M. Milajerdi, R. Gjomemo, B. Eshete , R. Sekar, V.N. Venkatakrishnan, HOLMES: Real-Time APT Detection through Correlation of Suspicious Information Flows, Proc. of 40th IEEE Symposium on Security and Privacy, 2019.

55. Splunk SIEM, https://www.splunk.com/, 2019.

56. LogRhythm SIEM, https://logrhythm.com/, 2019.

57. AlienVault® OSSIM™, Open Source Security Information and Event Management (SIEM), https://www.alienvault.com/products/ossim, 2019.

58. IBM QRadar SIEM, https://www.ibm.com/us-en/marketplace/ibm-qradar-siem, 2019.

59. Treinen J.J., Thurimella R. (2006) A Framework for the Application of Association Rule Mining in Large Intrusion Detection Infrastructures. In: Zamboni D., Kruegel C. (eds) Recent Advances in Intrusion Detection. RAID 2006. Lecture Notes in Computer Science, vol 4219. Springer, Berlin, Heidelberg.

60. Blei, David, Probabilistic Topic Models. Communications of the ACM. 55 (4), P. 77–84, 2012.

61. Flora S. Tsai, Kap Luk Chan, Detecting Cyber Security Threats in Weblogs Using Probabilistic Models, Proc. Intelligence and Security Informatics: Pacific Asia Workshop, PAISI 2007, Chengdu, China, April 11-12, 2007, P. 46-57.

62. E. Greeff, W. Ross, The Rise of the Machines, AI- and ML-Based Attacks Demonstrated, RSA Conference, 2019.

63. A. Burt, How will the GDPR impact machine learning? Answers to the three most commonly asked questions about maintaining GDPR-compliant machine learning programs, O'Reilly, 2018, https://www.oreilly.com/ideas/how-will-the-gdpr-impact-machine-learning.

64. L. Melis, C. Song, E. De Cristofaro, V. Shmatikov, Exploiting Unintended Feature Leakage in Collaborative Learning, Proc. of 40th IEEE Symposium on Security and Privacy, 2019.

65. X. Ling, S. Ji, J. Zou, J. Wang, C. Wu, Bo Li, and T. Wang, DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model, Proc. of 40th IEEE Symposium on Security and Privacy, 2019.

66. Kolosnjaji B., Demontis A., Biggio B., Maiorca D., Giacinto G., Eckert C., Roli F., Adversarial Malware Binaries: Evading Deep Learning for Malware Detection in Executables, 2018 26th European Signal Processing Conference (EUSIPCO), 2018, P. 533-537.

67. B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, Ben Y. Zhao, Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks, IEEE Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks, 2018.

68. X. Chen, C. Liu, B. Li, K. Lu, and D. Song, Targeted backdoor attacks on deep learning systems using data poisoning, arXiv preprint arXiv:1712.05526, 2017.

69. J. Clements and Y. Lao, Hardware trojan attacks on neural networks, arXiv preprint arXiv:1806.05768, 2018.

70. W. Li, J. Yu, X. Ning, P. Wang, Q. Wei, Y. Wang, and H. Yang, Hu-fu: Hardware and software collaborative attack framework against neural networks, in Proc. of ISVLSI, 2018.

71. T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," in Proc. of Machine Learning and Computer Security Workshop, 2017.

72. Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, Trojaning attack on neural networks, in Proc. of NDSS, 2018.

73. Powers, David M W. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. Journal of Machine Learning Technologies. 2 (1), P. 37–63, 2011.

74. K.N. Junejo, J. Goh, Behaviour-Based Attack Detection and Classification in Cyber Physical Systems Using Machine Learning, CPSS '16: Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security, May 2016.

75. A.A. Cárdenas, P.K. Manadhata, S.P. Rajan. Big Data Analytics for Security. IEEE Security & Privacy, Volume: 11, Issue: 6, 2013, pp. 74-76.

76. P. Giura and W. Wang, Using Large Scale Distributed Computing to Unveil Advanced Persistent Threats. Science J., vol. 1, no. 3, 2012, pp. 93–105.

77. T.-F. Yen et al., Beehive: LargeScale Log Analysis for Detecting Suspicious Activity in Enterprise Networks. Proc. Ann. Computer Security Applications Conference (ACSAC 13), ACM, Dec. 2013.

РОЗДІЛ 2
БЛОКЧЕЙН ІНФРАСТРУКТУРА ДЛЯ ЗАХИСТУ КІБЕРСИСТЕМ

Пропонується блокчейн технологія і математичний апарат створення інфраструктури програмно-апаратних телекомунікаційних інформаційних кібернетичних систем (КС), орієнтована на захист від несанкціонованого доступу до сервісів, визначених у специфікації системи, шляхом проникнення через легальні інтерфейси взаємодії компонентів, що мають уразливості. Інфраструктура захисних сервісів створюється разом з кіберсистемою і супроводжує останню протягом всього життєвого циклу, обслуговуючи всі наступні модифікації КС, і сама постійно підвищує свій інтелект шляхом поповнення історії та бібліотек конструктивних і деструктивних компонентів.

2.1. Практика використання топ-технологій

Високі витрати на дослідження і розробки від Amazon, Apple, Baidu, Google, IBM, Microsoft і Facebook стимулюють створення оригінальних патентованих рішень в області Deep Learning і Machine Learning, серед яких слід відзначити: Amazon Alexa, Apple Siri, Google Now, Microsoft Cortana. Компанія Gartner Inc. впевнена, що інструменти для глибокого навчання становитимуть 80% стандартних засобів для вчених. Сьогодні вже на сайтах компаній стають доступними технології і дані про наукові дослідження: Amazon Machine Learning, Apple Machine Learning Journal, Baidu Research, Google Research, IBM AI і Cognitive Computing, Facebook Research.

Впровадження 5G-технології телекомунікацій (рис. 2.1) в найближче десятиліття надасть ринку очікувані інноваційні рішення з безпеки, масштабованості і продуктивності глобальних мереж і з'єднань в транспорті, IoT, індустрії, охороні здоров'я.

Gartner Inc. прогнозує, що до 2020 року 3% мережевих провайдерів послуг мобільного зв'язку запустять комерційні мережі в 5G-форматі, що забезпечить якісно нові умови повсюдного впровадження телекомунікацій для масштабованої глобалізації сервісів: IoT, cloud-transport control, UHD-телебачення. Лідерами 5G-впровадження в 2017-2018 році виступають: AT & T, NTT Docomo, Sprint USA, Telstra, T-Mobile і Verizon. Технологія 5G є ультраширокополосний мобільний зв'язок в міліметровому діапазоні для Massive M2M транзакцій в реальному часі з допустимими для управління затримками (1мс), при одночасному підключенні близько 10 млн пристроїв на 1 км кв. 5G використовує технологію множинного доступу з поділом променя (Beam Division Multiple Access – BDMA) для взаємодії базової станції з мобільними пристроями. Бездротова стільникова архітектура 5G забезпечує пропускну здатність 10-50 Гбіт / с в міліметровому діапазоні частот 30-300 ГГц для додатків UHD відео і створення віртуальної реальності [4]. Інноваційна технологія 5G характеризується використанням: масиву приймально-передавальних антен Massive MIMO, мережі Cognitive Radio, організацією безпосереднього зв'язку D2D для IoT, створенням мережі радіодоступу, як хмарної послуги (radio access network as a service) і хмари віртуальних мережевих функцій (network function virtualization cloud – NFV).

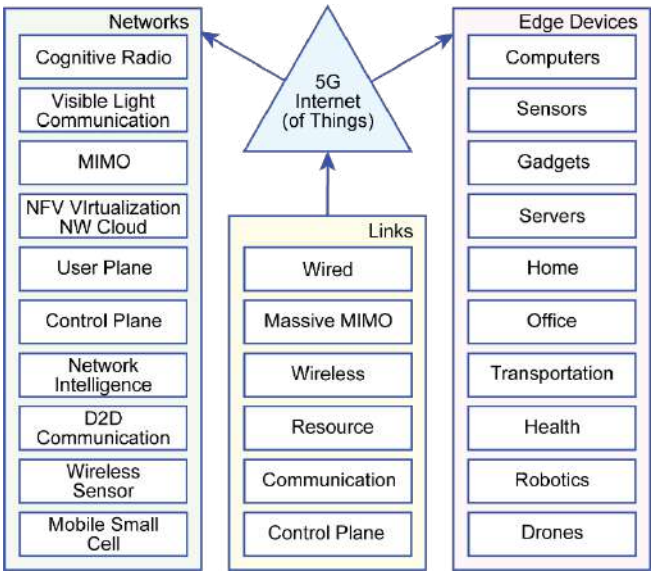


Рисунок 2.1 – Три складових технології 5G

Hype cycle, з незрозумілих причин, не розглядає дві істотних для людства абсолютно зелені і мультиміліардні технології, пов'язані з кіберсоціальним комп'ютингjv: Digital Humanity і Smart Cyber Digital State. Вони додані авторами публікації в Gartner's Table for Emerging Technologies, як запускаються інновації першої фази (innovation trigger), реалізація яких очікується через 5-10 років.

Digital Humanity – оцифроване людство передбачає точну цифрову ідентифікацію особистості з природничними біометричними параметрами (відбитки пальців, сканування обличчя, очі і ДНК), що виключають паперові носії інформації, пластикові карти, посвідчення, дипломи та паспорти в планетарному масштабі. Цифровий ідентифікатор дає можливість делегувати позиціонування кожної людини в часі і просторі хмарного сервісу, який знімає всі проблеми, пов'язані з кіберфізичним аналізом нелегітимних дій кожного індивідуума. Наслідком сталого розвитку цифрового людства в рамках зеленого інтернету речей для створення розумного світу [5] є багатомільярдна економія витрат на виробництво і використання паперових документів, збереження лісів і планетарної екології. Платою за отримання згаданих дивідендів є витрати на створення електронної інфраструктури для цифрової аутентифікації кожної особистості, процесу або явища в часі і просторі. Green IoT – кіберфізична культура людської діяльності, спрямована на забезпечення якості життя людей і збереження екології планети, енергії, ресурсів і часу. Компонентами IoT є: Identification, Sensing, Controlling, Communication, Cumputation, Sevices Intelligent, Digital Infrastructure. Розумний світ (smart world) надає кожній людині сервіси від: розумних пристроїв (watches, mobile phones, computers), розумного транспорту (aircrafts, cars, buses, trains), розумної інфраструктури (homes, hospitals, offices, factories, cities, states), розумної цифрової освіти (school, university).

Застосування кіберфізичних сервісів дозволяє реалізувати парадігму світа без посередників як інноваційного технологічного укладу прямих

телекомунікаційних контактів кожної особистості з будь-яким суб'єктом на планеті. Блокчейн-технологія [6] створює передумови для винищення корумпованих відносин у суспільстві, завдяки децентралізації, відкритості зберігання та обміну даними при виконані транзакцій, що забезпечує прямой зв’язок суб'єктів без посередників в рамках сервісів Інтернету Речей. Про це також йдеться у журналі IEEE Spectrum [7].

Технологія Blockchain (зокрема, bitcoin) являє собою кіберфізичний метричний хмарний криптозахищений комп'ютинг прозорого моніторингу та довірчого управління транзакціями в розподілених blockchain data (рис. 2.2).

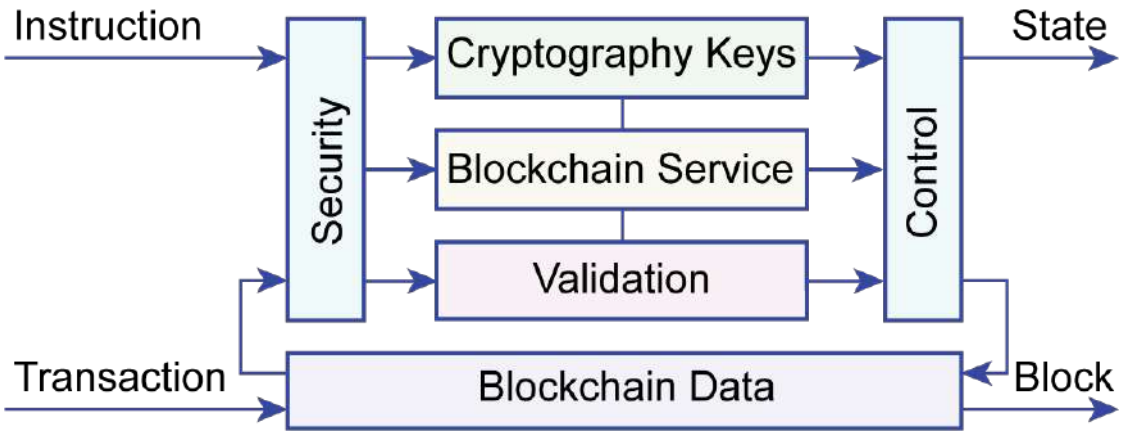


Рисунок 2.2 – Технологія Blockchain Computing

Ця технологія, що була запропонована Satoshi Nakamoto (2009), сьогодні стійко розвивається як розподілений в просторі і часі довірчий комп'ютинг з ненадійних компонентів: Ethereum Virtual Machine (2013, Vitalik Buterin); Microsoft blockchain додатки на хмарі Azure; IBM і Intel відкриті ресурси Hyperledger. За даними Blockchain.info в проект Bitcoin сьогодні вже залучено понад 375 000 чоловік (China, Eastern Europe, Iceland, Venezuela).

Криптовалюта стає вагомозначущим універсальним посередником між продавцем і покупцем для оцінювання соціальної значущості товарів і послуг. Ринкова капіталізація Bitcoin зросла до більш ніж 137 мільярдів

доларів, а вартість біткойнів у квітні 2018 року дорівнювала 8000 доларів. Приріст вартості криптовалют за рік склав 1000 доларів.

Отже, можна замінити грошові знаки на віртуальні, що розглядаються як цифровий код. При цьому слід замінити посередників, якими виступають чиновники, на комп'ютингові хмарні системи управління (рис. 2.3).

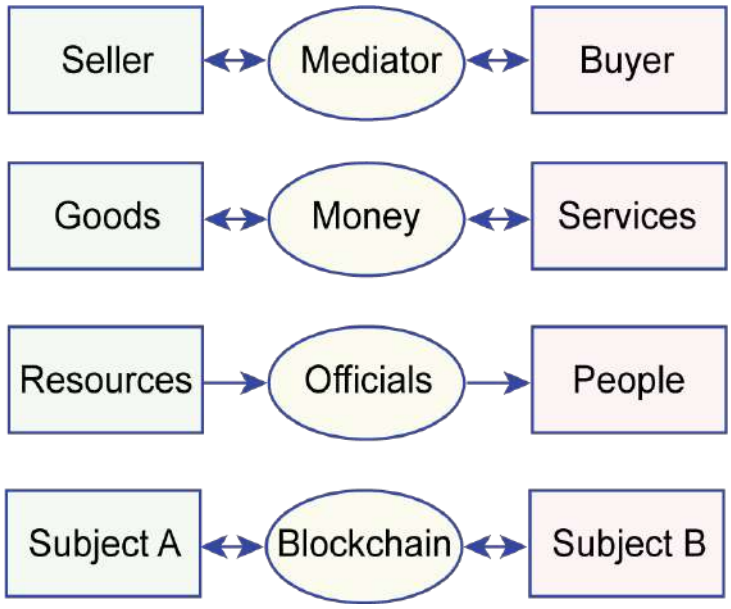


Рисунок 2.3 – Застосування Blockchain Computing

Область застосування технології Blockchain – кіберфізичні процеси і явища, що уражені корупцією, завдяки наявності посередників, які не можуть бути дійсно довірчими, зважаючи на людську слабкість – привласнити чуже, якщо відсутнє покарання. Фактично можна і потрібно будувати Blockchain довірчі системи для відкритого управління наукою, освітою, туризмом, транспортом, фінансами, соціумом, медициною і кадрами. Ethereum є blockchain (bitcoin) world-комп'ютинг, який може замінити Facebook, Twitter, Uber, Spotify, будучи невразливим для цензорів і прозорим щодо процесів, що відбуваються, а також довго працювати при відсутності людей, які його створили.

Недоліками Blockchain-комп'ютинга можна вважати наступні: 1) висока вартість створення Blockchain-інфраструктури; 2) відкритість більшості даних, що раніше становили інтелектуальну власність (як патенти і коди), що може привести до закриття частини компаній; 3) багатократне дублювання даних у мережі; 4) неготовність певних верств суспільства, низки державних та комерційних структур до впровадженню відкритої та незалежної системи розподілу ресурсів.

2.2. Blockchain Computing

Підвищення соціальної значущості безпаперової паралельної валюти постає як вагома конкуренція будь-яким посередницьким чи фінансовим інституціям, що так чи інакше повинні бути видалені з ринку [7]. Біткойн може розглядадтися як незалежна технологія моральної оцінки соціальної значущості результатів діяльності людей, спрямованої проти надмірності, несправедливості і корумпованості традиційної фінансової системи.

Біткойн замінює сервіси, що надаються сьогодні посередниками, на криптографію і виконуваний код. Біткойн і інші криптотранзакції замінюють договірні угоди з банком і з іншими людьми на розподілену і захищену базу даних, яка називається блок-ланцюжком. Процес верифікації, за допомогою якої володіння токеном біткойнів буде переходити від однієї людини до іншої, довіряється мережі комп'ютерів.

Майже через 10 років після створення першого блокчейна в формі криптовалюти люди досить успішно застосовують його до соціальних процесів і явищ, де можна замінити посередників на blockchain. Включивши мінімальну фантазію, можна запропонувати тисячі проектів для заміни на blockchain завжди дорогих посередників між продавцем і покупцем товарів і послуг: пропонувати безпосередньо розваги, подорожі, атракціони, телебачення, фільми, концерти, спортивні змагання, квитки на транспорт. При цьому не залишиться місця не тільки банкам, але і таким сервісам, як Uber, Netflix.

Згадані додатки є прикладами послуг, побудованих на Ethereum blockchain платформі, яка дистанційно виконує software на розподіленій комп'ютерній системі, названій Ethereum Virtual Machine. Ethereum blockchain кіберпростір, що має свою криптовалюта (ethers), є найбільш відкритим для проведення експериментів з боку численних груп дослідників, які хочуть змінити світ. Лідери IT-індустрії також грають на стороні blockchain. Microsoft пропонує своїм клієнтам інструменти для експериментів з blockchain в хмарі Azure. IBM, Intel та інші компанії підтримують blockchain з відкритим вихідним кодом в проекті Hyperledger, метою якого є створення архітектур для бізнес-орієнтованих blockchain. Тим часом багато хто з найбільших банків створили власну версію blockchain, намагаючись очолити успішний розвиток і просування технології. Проте проекти blockchain ще не революціонізували жодну галузь, як того б хотілося її творцям. Згідно з даними Blockchain.info, криптовалюта використовується в співтоваристві, що складається з 375000 чоловік з інтегральним рівнем cryptospace-капіталізації, рівному 400 мільярдів доларів [https://www.youtube.com/watch?v=UaCAHbOQQag&t=]. Але долари інвесторів незмінно дорожчають, а маси нових пропозицій виникають з невичерпного джерела blockchain, більшість з яких поки не сприймається бізнесом і науковим співтовариством як безальтернативне майбутнє.

Перша цифрова валюта працювала за принципом: гроші – це інструмент обліку і абстрагування вартості товарів і послуг при здійсненні операцій. Гроші є історичним метричним посередником для еквівалентного обміну товарами і послугами. Володіння фізичними знаками або монетами рівнозначно володінню матеріальними товарами або послугами. Якщо замість грошових знаків створити таблицю учасників закритого співтовариства, де будуть прописані рахунки кожної людини, то фізичні банкноти і монети стануть непотрібними. Банки вже частково трансформували фізичну валюту в цифрові записи обробки транзакцій в їх закритих системах. Біткойн завершив перетворення, створивши

універсальний цифровий ledger (бухгалтерську книгу), так званий blockchain, де зміни можуть бути зроблені тільки шляхом додавання нового запису в кінець блокового ланцюжка. Blockchain біткойнів, на відміну від бухгалтерських книг, підтримуваних традиційними фінансовими установами, реплікується на мережевих комп'ютерах і доступний для кожного учасника закритої або відкритою мережі. Клас учасників мережі, які називаються майнер, відповідає за виявлення запитів на транзакції від користувачів, їх перевірку і додавання нових блоків в blockchain.

Валідація виконує перевірку спроможності покупця, який володіє біткойнами в своїй транзакції, які він не витратив в іншому місці. Власність в blockchain ланцюжку біткойнів визначається парою криптографічних ключів. Перший, відкритий ключ, знаходиться в блокчейні для всіх учасників. Другий є закритим ключем, який власник зберігає в безпеці. Два ключі знаходяться в спеціальному математичному відношенні, яке робить їх корисними для цифрового підпису транзакцій. Ось як це відбувається: користувач формує повідомлення, об'єднує його зі своїм особистим ключем, виконує певні математичні обчислення для отримання довгого числа. Будь у кого є вихідне повідомлення і відповідний відкритий ключ може зробити деякі власні обчислення, щоб довести – довге число було створено за допомогою закритого ключа. У біткойнів транзакції підписуються закритими ключами, які відповідають відкритому ключу, пов'язаному з матеріалами, що витрачаються монетами. І коли транзакція обробляється, цим монетам присвоюється новий відкритий ключ. Головна роль blockchain-мінерів полягає в забезпеченні незворотності нових транзакцій, що робить їх остаточними і захищеними від несанкціонованого доступу з боку хакерів. Біткойн не має централізованої влади для дотримання правил. Майнери працюють анонімно в усьому світі, в просторі різноманітності культур, правових систем і нормативних зобов'язань. Тому немає простого фізичного способу привернути користувача-порушника до відповідальності. Щоб забезпечити легітимну поведінку людини, Bitcoin використовує схему,

названу доказом виконаної роботи, яка відповідає відкритому ключу, пов'язаному з матеріалами, що витрачаються монетами.

У відкритій часової мережі майнери запускають біткойн-код, отримують нові транзакції і збирають їх для створення нового блоку. Майнер конкурують один з одним. Перший, хто створить дійсний блок, отримує оплату в біткойні за цю послугу. Важливо, щоб всі мінери в мережі біткойн мали одну і ту ж копію блок-ланцюга, а всі зміни і транзакції незворотні. Щоб синхронізувати всіх мінерів, необхідно дороге програмне забезпечення, великі обчислювальні і енергетичні потужності для додавання дорогих за витратами змін в нових блоках.

Будь-який майнер, який намагається додати новий блок, повинен надати криптографічний доказ шляхом перетворення нового блоку за допомогою декількох обчислювально складних раундів визначення хеш-функції. Blockchain вимагає, щоб отриманий хеш починався з певної кількості нулів. Перший майнер, який знаходить задовільний хеш, оголошує новий блок іншим колегам, які перевіряють код і додають його до повної версії blockchain, яку вони записують на своїх комп'ютерах. За виконання даної роботи майнер отримує винагороду, а також гонорари за волонтерський видобуток «корисних копалин». Приклад: є замок, який потребує ключа для закриття, і є безліч ключів в розпорядженні користувача. Завдання полягає в пошуку правильного ключа за матеріальні стимули, який потім слід залишити в замку,

Біткойн-майнери інвестують ресурси в мережу, яку вони обслуговують, в частині електроенергії та комп'ютерного обладнання. Тому вони не схильні пошкодити валюту будь-якими діями або зовнішніми атаками, які можуть поставити під сумнів цілісність біткойнів і їх валідність. Успішність атак у міру зростання мережі зменшується, оскільки вартість зміни вмісту старих блоків збільшується з кожним новим блоком, який додається в ланцюжок. Наприклад, другий блок містить хеш тільки першого. Будь-які зміни в старих блоках приведуть до недійсним хеш для всіх наступних компонентів. Отже,

неможливо вставити фіктивні модифікації в попередній блок без повторення всієї роботи, яка була виконана після створення цього блоку. Конструкція блокування в кінці ланцюга залежить від усіх замків, які були зроблені перед нею. Тому зміна одного замка в середньому блоці ланцюжка означає необхідність пошуку нових ключів для кожного блоку після нього. Якщо є користувач, що володіє надпотужними обчислювальними ресурсами, то хакерська атака зі зміни записів можлива.

Сатоші створив першу життєздатну однорангову цифрову валюту. Але головне, він вирішив більше загальну проблему консенсусу, яка протягом десятиліть дратувала соціально-комп'ютерних вчених. Біткойн протягом останніх десяти років надійно стимулює до активності мережу потенційно нечесних анонімних учасників для чесної обробки транзакцій і забезпечення єдиної версії всіх подій. Результатом є постійно зростаючий ланцюжок даних, який будь-який користувач інтернету може перевіряти і доповнювати, а також той, що сьогодні є неймовірно захищеним від атаки.

Blockchain система може бути корисною набагато більше, ніж просто грошові відношення без посередників. Після успішного біткойн-дебюту дослідники розпочали генерувати інші ринковоорієнтовані додатки на блокчейн-платформі. Коли майнери перевіряють транзакції, вони запускають невеликі програми, які обробляють дані і надають «так-ні» експертний висновок за запитом транзакції. Вони можуть запускати більш складні програми, такі як соціальні мережі, онлайн-форуми, управління соціальними групами і державами.

Для поширення біткойну на інші сфери людської діяльності був запропонованій Ethereum, що використовує блок-ланцюг. На відміну від Bitcoin, Ethereum використовує транзакції, які є мініпрограмними або інтелектуальними контрактами з необмеженим ступенем складності. Користувачі можуть взаємодіяти з програмами, завантажуючи в них транзакції з інструкціями, які обробляють мінери. На практиці це означає, що будь-який користувач може вбудувати програму в транзакцію з упевненістю, що вона залишиться там

незмінною і доступною для всіх учасників. Теоретично Ethereum може замінити Facebook, Twitter, Uber, Spotify або будь-яку іншу цифрову службу новими версіями, які будуть недоступні для цензорів і прозорі для всіх учасників, працюючи нескінченно в часі при відсутності розробників. Дивно, що можна помістити комп'ютерну програму в Ethereum мережу, де всі учасники в системі можуть домовитися про те, що і коли буде відбуватися в мережі. Засновник Ethereum, Joseph Lubin, тепер запускає Consensys – Brooklyn-based інкубатор для децентралізованих додатків. Фактично можна створювати інфраструктури для накопичення та обміну будь-якими товарами і послугами, організовуючи спеціальні і універсальні мережі на основі blockchain-культури.

Обліковий запис або Permissioned Ledger. Паралельно з Ethereum-практикою використання технології blockchain для створення глобального комп'ютера, існує зворотна тенденція, пов'язана з реалізацією закритої і контрольованої мережі Сатоші. Група фінансових інститутів (Barclays, Goldman Sachs і JP Morgan) в 2014 році сформувала консорціум під назвою R3 для підвищення ефективності платежів між банками шляхом впровадження blockchain. Стало зрозуміло, що відкрита структура blockchain (біткойнів і ефіріум) суперечить їхнім потребам: анонімність користувачів, які на відкритих блокових ланцюгах представлені буквено-цифровими загальнодоступними адресами, без їх автентичності суперечить банківському законодавству в Сполучених Штатах і в інших країнах. Банкам важливо знати, хто є їх клієнтами і контрагентами. Фінансові установи юридично зобов'язані захищати дані про вкладників і контролювати їх транзакції за міждержавними і регіональними лініями. Публічні blockchain реплікують запис транзакції на кожному комп'ютері в мережі, що робить неможливим обмежити зберігання даних про транзакції при використанні технології блокового ланцюжка в банках.

В результаті з'явився підхід «дозволеної книги» в технології blockchain, де відомі ідентифікатори людей, що додають блоки, а дані в системі доступні

тільки для окремих призначених осіб. Оскільки право створювати нові блоки визначається людьми, які запускають код, а не випадково, то немає необхідності в перевірці валідності криптовалюти при оплаті. Така система (наприклад, Corda) використовується в ситуаціях, коли всі учасники блокового ланцюга мають певний ступінь довіри, але хочуть змоделювати послуги нейтральної третьої сторони (банків) при регулюванні міжнародних банківських переказів.

Підхід «дозволеної книги» поширюється за межі банків в інші галузі, які є охоронцями конфіденційних даних клієнтів. Багато з цих проектів побудовані за допомогою інструментів, що надаються Hyperledger проектом з відкритим вихідним кодом, організованим фондом Linux і підтримуваним великими технологічними фірмами. Hyperledger створює продукти для компаній, які хочуть працювати зі смарт-контрактами, але не наважуються використовувати відкриті blockchain (Ethereum, Bitcoin) мережі. «Користувачі повинні приймати нормативні вимоги, що пред'являються до таких організацій, як банки, страхові компанії і галузі охорони здоров'я. Останні не можуть дозволити собі ризик і невизначеність, які супроводжують відкриті системи» (Джонатан Леві, автор Hacera-системи управління доступом до блокових ланцюгів).

З'єднання смарт-контрактів і blockchain вимагає підтримки технологій для вирішення цілої низки проблем. Blockchain мережі не можуть зберігати структури великих даних, які вимагають реплікації. Наприклад, неможливо поширювати потокове відео за блочним ланцюжком, що містить мільйони вузлів. Ще одна проблема смарт-контрактів полягає в тому, що blockchain не може моніторити реальний світ, тільки віртуальний. Наприклад, якщо розумний контракт - система страхування польотних квитків, він повинен знати, коли літак злетить і приземлиться. Тут blockchain поки не має функцій запиту відповідних веб-сайтів про видачу польотної інформації, яку він повинен семантично розшифрувати. Для цього потрібні розумні доповнення до blockchain у вигляді хмарних сервісів або програмних додатків.

Розробники повинні створювати blockchain для зберігання і доступу до даних, захищених від уразливості, цензури і деструктивних проникнень. Проблема зберігання даних може бути вирішена за допомогою розподілених децентралізованих хмарних систем Labs Interplanetary Database або Storj Labs. Вони дозволяють користувачам здавати в оренду зайвий простір на своїх жорстких дисках. Такі розподілені мережі придатні для системи інтелектуальних контрактів на основі blockchain, де дані будуть надмірно зберігатися на безлічі комп'ютерів по всьому світу і завжди будуть доступні цензурі.

Імпорт даних в blockchain здійснюється в режимі реального часу «оракулами», які отримують оплату за надійний запит джерел даних і подачу їх на смарт-контракти в блок-ланцюжку. Наприклад, оракул Town Crier призначений для введення даних в блок-ланцюжок з надійного джерела на основі довірчого криптографічного програмного забезпечення на процесорах Intel. Отже, технологія blockchain повинна доповнюватися розумними програмними або хмарними сервісами, які будуть враховувати специфіку конкретної галузі людської діяльності.

Де взяти гроші для реалізації blockchain технології, щоб заплатити за техніку, сервіси та глибокі наукові дослідження? Створення нових функцій, алгоритмів і архітектур передбачає візуалізацію або знищення цінних даних, на основі яких виживають багато підприємств? Ethereum довіряє дані тим людям, які створили blockchain. Компанія не може вийти з рамок бізнес-моделі, яка збирає і продає товари і послуги, має історію покупок і дані про місцезнаходження партнерів. Компанія blockchain не може також покладатися на обмежене володіння своєю інтелектуальною власністю, оскільки програми на відкритому блокчейні доступні для загального огляду. Вже з'явився потенційний механізм фінансування підприємств з прив'язкою до blockchain, названий пропозицією монет Initial coin offering (ICO), який виявився прибутковим, хоча і юридично сумнівним. Групи, що фінансують проекти за допомогою ICO-залучення інвестицій у вигляді продажу

користувачам фіксованої кількості нових криптовалют, розробляють розумні контракти на основі використання монет для покупки додатків. Ці групи створюють монети до запуску проекту, які продаються на відкритому ринку. Наприклад, жетони для проїзду в метро випускаються і продаються до поїздки. Це дає можливість, замість пошуку інвестора, надрукувати масу монет (власну валюту) для продажу населенню, які потім можуть реалізовуватися за цінами, обумовленими вартістю поїздки на метро. Сьогодні понад півмільярда доларів обертаються в blockchain-компаніях шляхом продажу токенів, які в останні кілька місяців стали помітно зростати в ціні за рахунок появи нових інвестиційних пропозицій. Blockchain проект під назвою Tezos недавно встановив рекорд, зібравши більше 200 мільйонів доларів з ICO, ITO (продаж токенів). Підприємці blockchain насправді демонструють скупість і жадібність фінансових інститутів, що використовують страндартні валюти, підтримувані урядом. Коли гроші починають текти в інший бік, чиновники стають однаково незговірливими щодо громадськості, з якої вони вийшли.

Деякі експерти стверджують, що ICO, як новий клас інвестиційного інструменту, настільки ж руйнівна, як і фінансовані за даною схемою додатки. «Гроші не є коренем усього зла. Рівність є коренем усього зла», говорить Joel Monegro, засновник фонду Placeholder, орієнтованого на технології blockchain. Його аргумент полягає в тому, що надання засновникам і співробітникам акцій компанії спонукає їх накопичувати багатство, а не використовувати його для поліпшення своїх потреб. Монета, призначена для додатків, є не тільки фінансовим інструментом, а й засобом доступу до технологій. З цього випливає, що чим більше людей користуються послугою, тим більшим буде соціальний попит на токен, необхідний для доступу до цієї послуги. «Стимул компанії полягає не в отриманні більшого прибутку, а в отриманні більшого використання токена на основі масовості реалізації послуги серед населення». Сполучені Штати, ймовірно, будуть забороняти ICO-технологію, оскільки багато хто з

розглянутих токенів потрапляють в категорію цінних паперів і повинні підкорятися існуючим правилам. Необхідно, щоб Bitcoin і Ethereum функціонували в більших масштабах, а підприємствам слід децентралізовувати більше криптовалюти і забезпечувати конфіденційність даних. З огляду на величезні суми вкладених в blockchain грошей, необхідно залучати нових користувачів,

Децентралізація валют в глобальному масштабі на основі впровадження в практику bitcoin є моральною альтернативою розвитку людства, яка поки що не може промоделювати майбутнього. Однак з постулатів і визначень поступово складається картина blockchain культури, представлена на рис. 2.4.

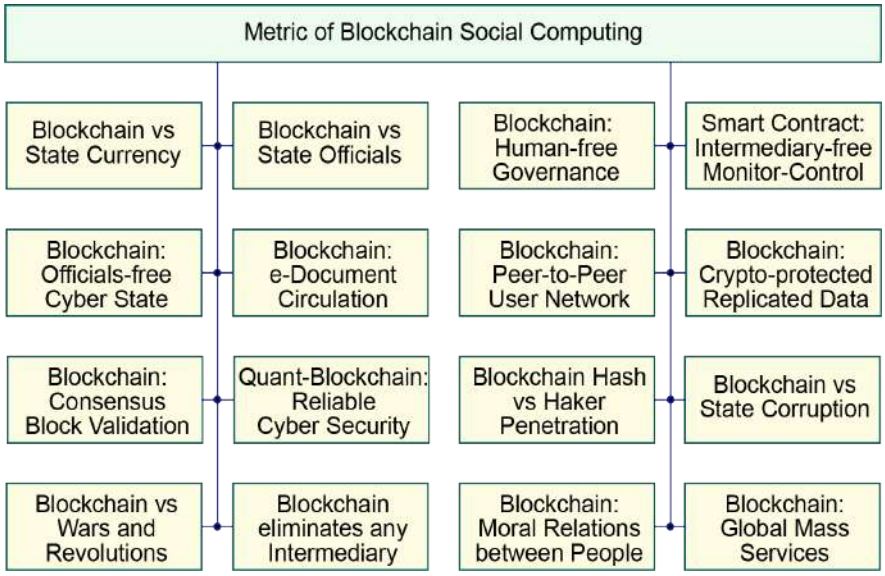


Рисунок 2.4 – Метричні параметри соціального blockchain-комп'ютинга

1) Криптовалюти здатні конкурувати з національними валютами і мирно захоплювати фінансову, політичну, економічну, законодавчу владу тотально або в окремих сегментах людської діяльності. 2) Кожна соціальна група, яка має спільні інтереси, може створювати власну криптовалюту для обміну певними товарами і послугами без використання зовнішніх валют. 3) Бартерні відносини всередині закритої соціальної групи: «ти – мені, я – тобі» є фізичний прихований від суспільства прообраз криптовалюти без

використання стандартних грошових знаків. 4) Будь-які корумповані, завжди закриті співтовариства людей, що довіряють один одному, використовують власну валюту для обміну цінностями: придбання посади, земельної ділянки, вступ до вишу, складання іспитів. 5) Випуск компанією цінних паперів, акцій рівнозначний емісії токенів, криптовалюти і bitcoin. 6) Приватизація частини державної власності через випуск акцій аналогічна створенню тимчасової мережі, де учасники-акціонери можуть купувати і продавати один одному свої частки власності. 7) Позитивізм створення часових blockchain мереж полягає у виключенні будь-яких посередників при укладенні розумних контрактів між учасниками закритого співтовариства. Посередниками виступають державні структури, приватні організації, банки і чиновники, які виробляють товари та послуги, що суттєво збільшують накладні витрати на ведення бізнесу. 8) Наявність метрики (криптовалюти) вимірювання соціальної значущості товарів та послуг в замкнутій мережі користувачів є умовою її існування. 9) Blockchain мережа, як система, має на меті підвищення якості життя та збереження екології планети шляхом морального human-free управління соціальними процесами без участі посередників для метричного розподілу матеріального і / або винагороди на основі вичерпного моніторингу соціальної значущості і верифікованих транзакцій, розумних контрактів. 10) Blockchain мережа масштабується в кібердержаву, де суб'єктами виступають компанії і організації, метрично розподіляють на основі консенсусу гроші платників податків без участі посередників. Кібердержава отримує частину прибутку від суб'єктів на розвиток і забезпечення життєдіяльності соціальної системи. 11) Смарт-контракт – метрично розподілені на основі консенсусу гроші платників податків без участі посередників. Кібердержава отримує частину прибутку від суб'єктів на розвиток і забезпечення життєдіяльності соціальної системи. 12) Смарт контракт – кіберсоціальна система електронних intermediary-human-free відносин між покупцем і продавцем товарів і послуг, реалізована у вигляді криптозахищеного програмного коду, що має на меті достовірне виконання

сторонами договірних умов на основі вичерпного моніторингу процесу виконання зобов'язань і вироблення адекватних актюаторних впливів на компоненти blockchain-інфраструктури. 13) Електронний документообіг – кіберсоціальна система електронних intermediary-human-free відносин між працівниками компанії на е-інфраструктурі, реалізована у вигляді криптозахищеного програмного коду, що має на меті достовірне виконання сторонами наказів і розпоряджень шляхом вироблення адекватних актюаторних впливів на основі вичерпного моніторингу процесу виконання документа. 14) Blockchain комп'ютинг – обчислювальний процес в замкнутій розподіленій кіберсоціальній часовій комп'ютерній мережі, призначеній для виконання смарт-контрактів і збереження реплікування криптозахищених ланцюжків записів про транзакції на основі human-free моніторингу і консенсусної валідації кожного нового блоку з метою створення толерантних метричних довірчих відносин, без посередників, між ненадійними учасниками мережі. 15) Квантовий комп'ютинг, що не тільки створює засоби для успішного проведення хакерських атак на сучасні blockchain-інфраструктури, але в більшій мірі надасть нові технології надійного кіберзахисту блокових ланцюжків шляхом впровадження логіки квантового змішування, що дозволяє стискати простір і час в структурах даних. 16) Хеш-функція являє собою цифрову сигнатуру фіксованої довжини для бажаної довгої кінцевої вхідної послідовності бітів, отриману в результаті послідовного застосування односпрямованого функціонального xor-шифрування цифрових блоків фіксованої довжини, при відомому початковому ключі. 17) Створюється моральна кібердержава, де функції посередника у відносинах між громадянами виконує blockchain мережа.

Інтегральне уявлення кіберфізичної системи blockchain-комп'ютинга зображено на рис. 2.5.

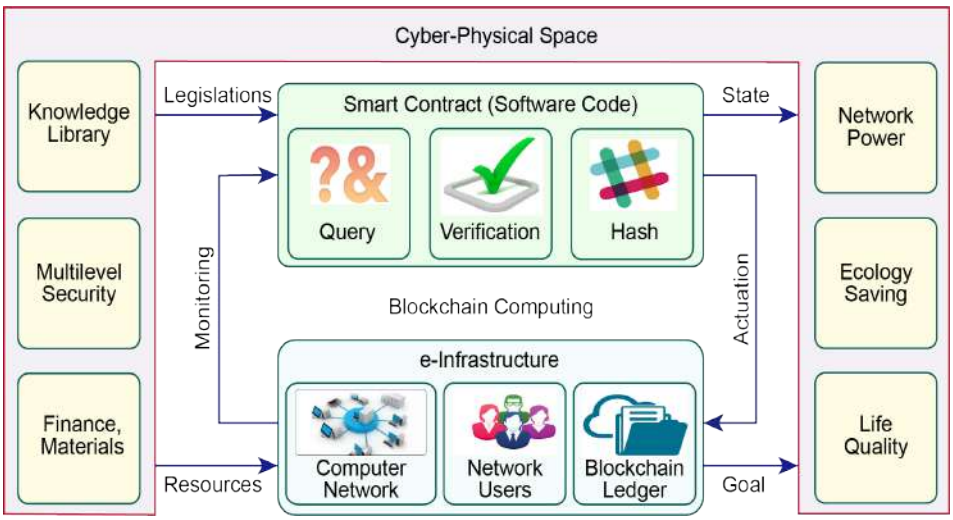


Рисунок 2.5 – Кіберфізична система blockchain-комп'ютинга

Система характеризується human-free управлінням процесами виконання розумного контракту на основі цифрового моніторингу сенсорів, пов'язаних з транзакціями товарів, послуг і фінансів, а також з формуванням захищених blockchain-структур даних на всіх комп'ютерах розподіленої мережі.

«Blockchain і цифрові активи, такі як Bitcoin і Ether, революційно оцифровують області людської діяльності, включаючи освіту, промисловість, фінансові послуги, охорону здоров'я, уряд, менеджмент, енергетику, нерухомість і суспільство" [12]. Динаміка вартості bitcoin почалася з цифри 936 доларів і досягла до середині грудня 2017 – 19 500 доларів. Однак за два тижні до кінця грудня, криптовалюта зменшилася на 30 % – 14 100 доларів. Тому при покупці криптовалюти слід користуватися правилом – вкладайте рівно стільки грошей, скільки вам не шкода втратити.

Квантові блокчейни можуть функціонувати як машини часу і протистояти атакам хакерів за допомогою квантових комп’ютерів. Quantum Blockchains Could Act Like Time Machines [13].

З'єднання двох технологій: квантового комп'ютинга і блокчейна може привести до створення Q-Blockchain Systems (QBS), непроникних для злому з боку квантових комп'ютерів. Важливо відзначити, що Q-Blockchain функціонує як машина часу, яка здатна впливати на власне минуле.

Blockchain являє собою базу даних, в якій зберігаються записи про минуле системи, наприклад, історія фінансових або інших транзакцій, які були узгоджені з кожним вузлом мережі без централізованого управління. Найбільш відомим застосуванням Blockchain є криптовалюта біткойн, проте існує безліч компаній і дослідників, які пропонують інші можливості для використання даної технології. За словами Del Rajan, фізика-теоретика з Новозеландського Університету в Веллінгтоні, очікується, що до 2027 року на основі технології Blockchain можна буде зберігати 10 відсотків світового ВВП. Кіберкультуру Blockchain очікує плідна зустріч з іншою сучасною технологією – квантовим комп'ютингом, що використовує квантові біти або кубіти, які через сюрреалістичну природу квантової фізики можуть перебувати в стані суперпозиції, коли X = {0,1}. Суперпозиція бітів в кубіті дає можливість одночасно виконувати два обчислення, а в загальному випадку – 2**n операцій на n кубітах. Теоретично квантовий комп'ютер з 300 кубітами здатний виконати більше обчислень в одну мить, ніж число атомів у видимій частині Всесвіту. Потужний квантовий комп'ютер здатний успішно перемогти класичну криптографію, в тому числі і сучасний захист Blockchain. Однак відбивання технологій Q-computing + Blockchain = Q-Blockchain може конструктивно протистояти спробам злому з боку квантових комп'ютерів.

Q-computing використовує квантове переплутування у просторі (Quantum Entanglement in Space), яке визначається взаємним впливом двох і більше часток один на одного, інваріантним до відстані – spooky action at a distance, за словами Альберта Ейнштейна. Переплутання в Q-computing означає згортку простору в одну точку.

Дізрапторна інновація Q-Blockchain визначається змішуванням або зв'язуванням квантових частинок в часі (Quantum Entanglement in Time), яке формується взаємним впливом двох і більше часток один на одного, інваріантним до будь-якої часової відстані між ними. Тут квантове переплутання означає згортку часу в одну точку.

Blockchain формує записи в блок-ланцюжки даних, які мають криптографічні посилання, задані в хронологічному порядку. Якщо хакер спробує змінити запис конкретного блоку, то криптографічний алгоритм анулює всі наступні блоки після зламаного.

Q-Blockchain формує записи в блок-ланцюжку фотонів або квантів, які переплутані або пов'язані між собою в хронологічному порядку, в часі. Тут записи формують квантовий блок-ланцюжок, де фотони, які кодують кожен блок, передаються всіма компонентами мережі квантових комп'ютерів. Q-Blockchain переплутання пов'язує всі фотони в часі. Інакше, записи в Q-Blockchain часового ланцюжку теперішньої і минулих транзакцій закодовані в єдиному квантовому стані, що означає згортку часу в одну точку.

Хакер не може втручатися в запис блок-ланцюжків минулого, оскільки відповідні фотони більше не існують в теперішньому часі – вони вже переплутані. У кращому випадку хакер може успішно змінити фотон останнього блоку (це зробить його недійсним, повідомляючи всім іншим, що він зламаний). Така атака є менш суттєвою, ніж стандартний випадок, коли хакер має можливість змінити будь-який блок в часовому ланцюжку.

Квантова заплутаність в часі означає, що вимір останнього фотона в блоці впливає на перший фотон цього блоку в минулому перед тим, як його виміряли. Measuring the last photon in a block influences the first photon of that block in the past before it got measured. По суті, поточні записи в квантовому блоковому ланцюжку не просто пов'язані із записом минулого, а швидше з записом в минулому, якого більше не існує. «Проект в певному сенсі може розглядатися як квантова машина часу», – говорить автор Метт Виссер (Matt Visser), фізик-теоретик з Веллінгтона.

2.3. Системні проблеми, пов'язані з «проникненням» і «вразливістю»

Поняття, що визначаються словами «проникнення» і «вразливість», є взаємодоповнюючими один одного. Якщо є вразливість, то в неї, як у дірку,

можливе проникнення деструктивності, яка вписується в функціональність кіберсистеми. Вірно і зворотне, якщо зафіксовано проникнення, то воно сталося внаслідок наявності в системі уразливості (дірки). Проблема захисту кіберсистеми від несанкціонованого доступу полягає в «неможливості» відрізнити деструктивність від «конструктивності» або валідного користувача. Проте існують методики, технології, програмні засоби і системи, здатні ефективно вирішувати питання захисту корпоративного або персонального кіберпростору з наперед заданою вірогідністю проникнення. Існуючі публікації по даному напрямку оперують такими термінами.

Тест проникнень – сукупність зовнішніх і внутрішніх деструктивних впливів, спрямованих на виявлення вразливостей доступу до сервісів КС шляхом моделювання або аналізу проникнень на моделі кіберсистеми.

Якість тесту визначається його повнотою, вираженою у відсотках, щодо перевірки всіх можливих типів вразливостей, що генеруються вручну або автоматично для кожної конкретної кіберсистеми.

Результат тестування реальної системи (System Under Penetration Test - SUPT) формує кількісну оцінку вразливості, а також список структурних вразливостей наперед заданих типів, виявлених в процесі тестового експерименту.

Якщо процес тестування зафіксував непорожній список деструктивних вразливостей, то необхідно виконувати діагностування на основі використання тестів з метою визначення місця, причини і виду вразливості з наперед заданою глибиною пошуку деструктивності.

Після точного визначення всіх вразливостей виконуються процедури їх усунення шляхом часткової або повної реконструкції кіберсистеми на основі використання перевірених бібліотечних структурних рішень.

Всі процедури, згадані вище, використовують три бібліотеки: 1) негативну, що описує всі можливі типи вразливостей; 2) позитивну, де кожній вразливості ставиться у відповідність вірне програмно-апаратне рішення, що усуває деструктивність; 3) неперевірені рішення, складові

потенціал «інтелекту» КС, який визначається в процесі експлуатації кіберсистеми. Всі три бібліотеки необхідно поповнювати як в процесі проектування КС, так і на стадії експлуатації в реальному часі.

Завдання інфраструктури захисного сервісу кіберсистеми:

1) Синтез (дедуктивної) моделі КС для тестування, діагностування та відновлення невразливості кіберсистеми.

2) Генерування тестів перевірки та діагностування вразливостей, близьких до 100% повноти.

3) Створення алгоритмів пошуку вразливостей з наперед заданою глибиною діагностування.

4) Створення генераторів тестів перевірки та діагностування вразливостей, близьких до 100% повноті.

5) Тестопридатності проектування (модифікації) невразливих кіберсистем, «вільних» від вразливостей на поточний момент розвитку технологічної та математичної культури.

6) Розробка вбудованої інфраструктури захисного сервісу для кіберсистем, орієнтованих на моніторинг, тестування, діагностування та відновлення невразливості в реальному масштабі часу в процесі експлуатації.

7) Розробка спеціалізованих маршрутів (алгоритмів, планів) моніторингу, тестування, діагностування та відновлення невразливості КС в реальному масштабі часу в процесі експлуатації.

8) Верифікація інфраструктурних тестопридатних рішень, розроблених для реальних КС.

Об'єкт тестування – кібернетична система взаємодіючих програмно-апаратних, телекомунікаційних, інформаційних компонентів, орієнтована на надання якісних сервісів через стандартні інтерфейси санкціонованому користувачеві в реальному масштабі часу. Всі типи вразливостей (проникнень) не виводять об'єкт тестування за кордон заданої функціональності кіберсистеми, представленої булевою функцією:

$$Y = f(X_1, X_2, ..., X_i, ..., X_n),\ X_i, Y \in \{0,1\}.$$

Тому модель вразливостей накладається на графову структуру функціональних модулів, що мають вхідні і вихідні транзакційні змінні. Транзакційний граф представлений дугами – функціональностями (сервісами) з моніторами (ассерція), а також вершинами, що формують стан кіберсистеми за допомогою змінних, пам'яті, інтерфейсних портів введення-виведення інформації, приймачів, терміналів, комп'ютерів: F = (A*B)xS, де S = {S1, S2, ..., Si, ..., Sm} – вершини або стани КС при моделюванні тестових сегментів. Кожен стан Si = {Si1, Si2, ..., Sij, ..., Sip} визначається значеннями істотних змінних КС (змінні, пам'ять, термінали, комп'ютери). Орієнтовані дуги графа є функціональні блоки:

$$B = (B_1, B_2, ..., B_i, ..., B_n), \bigcup_{i=1}^{n} B_i = B; \bigcap_{i=1}^{n} B_i = \varnothing,$$

де кожному з них може бути поставлена у відповідність ассерція Ai∈A = {A1, A2, ..., Ai, ..., An} для моніторингу функціональностей в часі і в просторі.

Існують базові технології тестування безпеки кіберсистем: OSSTMM – The Open Source Security Methodology Manual; NIST Guideline on Network Security Testing; ISACA Switzerland – Testing IT Systems Security With Tiger Teams; Draft Guideline on Network Security Testing; NIST Special Publication 800-26 Security Self-Assessment Guide for Information Technology Systems; Cybersecurity Vulnerability Assessment Methodologies (Cybersecurity VAMs); Information Systems Security Assessment Framework, OISSG.

Функція мети представлена підвищенням ефективності сервісного обслуговування на основі стандартів тестування, граничного сканування і спеціальних технологій діагностування та відновлення невразливості КС, яка визначається мінімальним значенням рівня вразливості, часу відновлення працездатності T і нефункціональної програмно-апаратної надмірності H:

$$E = F(L,T,H) = \min[\frac{1}{3}(L + T + H)],$$

$$Y = (1 - P)^n;$$

$$L = 1 - Y^{(1-k)} = 1 - (1 - P)^{n(1-k)};$$

$$T = \frac{(1-k) \times H^s}{H^s + H^a}; H = \frac{H^a}{H^s + H^a},$$

де L – доповнення до рівня невразливості Y, яке залежить від тестопридатності КС k, ймовірності P існування вразливостей і числа невиявлених деструктивних n. Час тестування і діагностування залежить від тестопридатності архітектури k, помноженої на число структурних компонентів інфраструктури, віднесені до загальної кількості елементів КС. Надмірність залежить від структурної складності тестопридатності надбудови, поділеної на програмно-апаратну складність КС. Надмірність інфраструктури забезпечує задану глибину діагностування вразливостей за час, що визначається замовником.

2.4. Математичний апарат інфраструктури захисного сервісу

Розглянемо метрику, алгебру, структури даних і моделі оцінювання якості взаємодії процесів, явищ, об'єктів і компонентів в кіберпросторі і кіберсистеми, необхідні при створенні ефективних двигунів для обчислювальних процедур аналізу даних в процесах тестування проникнень і відновлення невразливості.

Критерії взаємодії об'єктів тестування грунтуються на використанні бета-метрики вимірювання відстаней в кіберпросторі. Кіберпростір – дискретний векторно-логічний простір – сукупність взаємодіючих за відповідною метрикою інформаційних процесів і явищ, що описуються векторами логічних змінних і використовують як носій комп'ютерні системи та мережі. Метрика – спосіб вимірювання відстані в просторі між компонентами процесів або явищ, описаних векторами логічних змінних. Відстань (булева похідна, ступінь зміни, відмінності або близькості) в кіберпросторі визначається xor-ставленням векторів (матриць), які

позначають компоненти процесу або явища, що відрізняє його від кодової відстані по Хеммінгу. Процедури порівняння, вимірювання, оцінювання, розпізнавання, тестування, діагностування оперують xor-ставленням об'єктів або їх компонентів. Компонент простору представлений k-мірним вектором a={a1, a2, ..., aj, ..., ak}, ai ∈{0,1}, де кожна його координата визначена в двійковому алфавіті, 0 – «неправда», 1 – «істина». Нуль-вектор є k-мірний кортеж, всі координати якого дорівнюють нулю: aj = 0, j = 1, ..., k.

Метрика β кібернетичного простору визначається рівністю, яке формує нуль-вектор для xor-суми відстаней di між ненульовим і кінцевим числом об'єктів, замкнутих в цикл. Тут n – кількість відстаней між компонентами (векторами) простору, складовими циклу D = {d1, d2, ..., di, ..., dn}, di – є вектор відстані, відповідний ребру циклу, що з'єднує два компонента (вектора) a, b простору, який далі позначається без індексу як d (a, b). Відстань між двома об'єктами a і b є похідний вектор $d(a,b) = (aj \oplus bj)_1^k$. Векторному значенню відстані відповідає норма (скаляр), що визначається кодовою відстанню по Хеммінгу між двома векторами у вигляді числа одиниць вектора d (a, b). Метрика β векторного логічного двійкового простору є рівна нуль-вектору xor-сума відстаней між кінцевим числом вершин графа, що утворюють цикл. Тепер можна дати більш формальне визначення кіберпростору, як векторно-логічного, нормованого бета-метрикою, де xor-сума відстаней між кінцевим числом точок циклу дорівнює нуль-вектору. Визначення метрики через відносини дозволяє скоротити систему аксіом (рефлексивності, сімметрчності і транзитивності, трикутного замикання) з трьох до одного і поширити її дію на як завгодно складні структури n-мірного логічного простору. Класичне завдання метрики для визначення взаємодії однієї, двох і трьох точок у векторному логічному просторі є окремим випадком бета-метрики при i=1,2,3 відповідно. Визначення метрики через відносини дозволяє скоротити систему аксіом (рефлексивності, сімметрчності і транзитивності, трикутного замикання) з трьох до одного і поширити її дію на як завгодно складні структури n-

мірного логічного простору. Класичне завдання метрики для визначення взаємодії однієї, двох і трьох точок у векторному логічному просторі є окремим випадком бета-метрики при i = 1,2,3 відповідно $\beta=\bigoplus_{i=1}^{n} d_i = 0$:

$$M \subset \beta = \begin{cases} d_1 = 0 \leftrightarrow a = b; \\ d_1 \oplus d_2 = 0 \leftrightarrow d(a,b) = d(b,a); \\ d_1 \oplus d_2 \oplus d_3 = 0 \leftrightarrow d(a,b) \oplus d(b,c) = d(a,c). \end{cases}$$

Векторно-логічний транзитивний трикутник має повну аналогію з чисельним виміром відстані в метричному М-просторі, який задається системою аксіом, що визначає взаємодію однієї, двох і трьох точок в будь-якому просторі:

$$M = \begin{cases} d(a,b) = 0 \leftrightarrow a = b; \\ d(a,b) = d(b,a); \\ d(a,b) + d(b,c) \geq d(a,c). \end{cases}$$

Специфіка аксіоми трикутника (метричного) М-простору полягає в чисельному (скалярному) порівнянні відстаней трьох об'єктів. При цьому інтервальна невизначеність відповіді – дві сторони трикутника можуть бути більші або рівні третій – малопридатна для визначення точної довжини останньої сторони. Бета-метрика усуває даний недолік і виключає невизначеність бінарного відношення детермінованих процесів або явищ. Третя сторона трикутника у векторному логічному просторі визначається двійковим вектором-відстанню між двома вершинами шляхом обчислення xor-суми відстаней двох інших сторін трикутника:

$$d(a,b) \oplus d(b,c) = d(a,c) \rightarrow d(a,b) \oplus d(b,c) \oplus d(a,c) = 0.$$

Метрика β кібернетичного багатозначного векторно-логічного простору є вектор, що дорівнює значенню Ø по всіх координатах, отриманий шляхом застосування симетричної різниці відстаней між кінцевим числом точок, що утворюють цикл:

$$\beta = \overset{n}{\underset{i=1}{\Delta}} d_i = \varnothing .$$

Тут кожна координата вектора, відповідного об'єкту, визначена в алфавіті, що становить булеан на універсумі примітивів потужністю p:

$$a_j = \{\alpha_1, \alpha_2, ..., \alpha_r, ..., \alpha_m\}, m = 2^p .$$

На основі введеної метрики аналізу кіберпростору вводяться критерії оцінювання взаємодії кінцевого числа об'єктів між собою. Скалярний критерій взаємодії двох об'єктів (процесів) в дискретному булевому просторі, представлених k-мірними багатозначними векторами

$$m = (m_1, m_2, ..., m_j, ..., m_k),\ m_j \in \{0,1,x\};$$

$$A = (A_1, A_2, ..., A_j, ..., A_k),\ A_j \in \{0,1,x\},$$

необхідний для порівняння і подальшого вибору, кращого в деякому сенсі, рішення. Ступінь належності m-вектора до A-вектора $\mu(m \in A)$ позначається як неналежність $\bar{\mu}(m \in A)$. Існує 5 типів теоретико-множинної взаємодії двох векторів:

$$1)\ m = A;\ 2) m \subset A;\ 3) A \subset m;\ 4) m \cap A \neq \{m, A, \varnothing\};\ 5) m \cap A = \varnothing .$$

Мета скалярного критерію – оцінити будь-яке з зазначених взаємодій інтервальною оцінкою [0,1] шляхом спільного використання трьох параметрів: кодової відстані d (m, A) і двох функцій неналежності

$$\bar{\mu}(m \in A) = 1 - \mu(m \in A),\ \bar{\mu}(A \in m) = 1 - \mu(A \in m);$$

$$Q = \frac{1}{3}\left[\frac{1}{k}d(m,A) + [1-\mu(m\in A)] + [1-\mu(A\in m)]\right],$$

$$d(m,A) = \mathrm{card}\,(m_i \overset{k}{\underset{i=1}{\cap}} A_i = \varnothing);$$

$$\mu(m\in A) = 2^{c-a};$$

$$\mu(A\in m) = 2^{c-b};$$

$$a = \mathrm{card}\,(A_i = x), i = \overline{1,k};$$

$$b = \mathrm{card}\,(m_i = x), i = \overline{1,k};$$

$$c = \mathrm{card}\,(m_i \overset{k}{\underset{i=1}{\cap}} A_i = x).$$

Тут

$$d(m,A) = \mathrm{card}\,(m_i \overset{k}{\underset{i=1}{\cap}} A_i = \varnothing)$$

– потужність або кількість порожніх координатних перетинів двох взаємодіючих векторів, що складають відстань по Хеммінгу;

$$\mu(m\in A) = 2^{c-a}\ (\mu(A\in m) = 2^{c-b})$$

– відношення загального для m і A простору до простору вектора A (m), що формує зазначену функцію належності. Операції координатного перетину (and), симетричної різниці (xor) визначені для символів алфавіту Кантора A = {0,1, x = {0,1}, Ø}, кодованих векторами (01,10,11,00) відповідно:

∩	0	1	x	∅
0	0	∅	0	∅
1	∅	1	1	∅
x	0	1	x	∅
∅	∅	∅	∅	∅

∧	01	10	11	00
01	01	00	01	00
10	00	10	10	00
11	01	10	11	00
00	00	00	00	00

Δ	0	1	x	∅
0	∅	x	1	0
1	x	∅	0	1
x	1	0	∅	x
∅	0	1	x	∅

⊕	0	1	x	∅
0	00	11	10	01
1	11	00	01	10
x	10	01	00	11
∅	01	10	11	00

Нормування параметрів критерію (кодової відстані і функцій неналежності) дозволяє оцінити рівень взаємодії векторів в чисельному інтервалі [0,1]. З урахуванням ізоморфізму теоретико-множинних і логічних операцій критерій якості можна трансформувати до виду:

$$Q = \frac{1}{3}\left[\frac{1}{k}d(m,A) + [1-\mu(m\in A)] + [1-\mu(A\in m)]\right],$$

$$d(m,A) = \mathrm{card}\,(m_i \overset{k}{\underset{i=1}{\oplus}} A_i = U);$$

$$\mu(m\in A) = \mathrm{card}\,(A_i = U) - \mathrm{card}\,(m_i \overset{k}{\underset{i=1}{\wedge}} A_i = U);$$

$$\mu(A\in m) = \mathrm{card}\,(m_i = U) - \mathrm{card}\,(m_i \overset{k}{\underset{i=1}{\wedge}} A_i = U);$$

$$U = \begin{cases} 1 \leftarrow \{m_i, A_i\} \in \{0,1\}; \\ x \leftarrow \{m_i, A_i\} \in \{0,1,x\}. \end{cases}$$

Якщо вектори m і A – виконавчі по всіх координатах, то змінна U = 1 і обчислення проводяться за правилами двійкової операції. Якщо вектори m і A визначені в троїчному алфавіті, то змінна U = x ініціює обчислення на основі використання теоретико-множинної операції симетричної різниці. Перший компонент d (m, A) / n критерію формує ступінь розбіжності k-мірних векторів у вигляді кодової відстані по Хеммінгу, віднесеної до довжини вектора, шляхом виконання операції xor над усіма координатами; другий і третій компоненти

$$[1-\mu(m\in A)] + [1-\mu(A\in m)]$$

визначають ступень неналежності результату кон'юнкції до простору кожного з двох взаємодіючих векторів. Якщо такі міри дорівнюють нулю

$$\frac{1}{n}d(m,A) = 0,\ [1-\mu(m\in A)] = 0,\ \ [1-\mu(A\in m)] = 0,$$

то об'єкти ідентичні один одному. Поняття власності і неналежність є взаємодоповнюючими, але в даному випадку більш технологічно обчислювати неналежність, оскільки загальноприйнятим в літературі є поняття нульової розбіжності об'єктів, що свідчить про їх повну ідентичність. Цей критерій працює в інтервалі [0,1]. Повний збіг двох об'єктів

$$d(m, A) = 0,\ \mu(m \in A) = 1,\ \mu(A \in m) = 1$$

характеризується нульовою оцінкою критерію

$$Q = \frac{1}{3}\left[\frac{1}{k}0 + [1-1] + [1-1]\right] = 0.$$

Протилежним варіантом оцінювання є максимальна розбіжність двох об'єктів:

$$d(m, A) = k,\ \mu(m \in A) = 0,\ \mu(A \in m) = 0,$$

яка визначається оцінкою взаємодії:

$$Q = \frac{1}{3}\left[\frac{1}{k}k + [1-0] + [1-0]\right] = 1.$$

Якщо параметри взаємодії рівні

$$d(m, A) = 0,\ \mu(m \in A) = \frac{1}{2},\ \mu(A \in m) = \frac{1}{2},$$

то критерій буде мати таку оцінку:

$$Q = \frac{1}{3}\left[\frac{1}{k}0 + [1-\frac{1}{2}] + [1-\frac{1}{2}]\right] = \frac{1}{3}.$$

Взаємодія (перетин) двох векторів: A=(XXX1X) і m=(XX0X0) дає загальний простір, рівний (XX010)={00010, 01010, 10010, 11010}. Критерій якості взаємодії при параметрах

$$d(m, A) = 0,\ \mu(m \in A) = \frac{1}{2},\ \mu(A \in m) = \frac{1}{4}$$

матиме таку оцінку:

$$Q = \frac{1}{3}\left[\frac{1}{k}0 + [1 - \frac{1}{2}] + [1 - \frac{1}{4}]\right] = \frac{1}{4}.$$

Перевага введеного критерію (неналежність, відмінності) полягає в лінійності зміни його чисельного значення від 0 до 1 в міру збільшення «відстані» від повного збігу двох об'єктів до максимально можливого, коли кодова відстань одно d (m, A) = k.

Критерій може бути використаний в задачах відстеження цілі, руху по заданому маршруту, тестування і діагностування функціональних порушень і вразливостей, пошуку, розпізнавання та прийняття рішень. Критерій якості Q, застосовуваний для виконання регуляторної функції при оцінюванні взаємодії об'єктів в реальному масштабі часу, необхідно мінімізувати.

Проте скалярна оцінка має лише інтегральні властивості взаємодії двох об'єктів, що дозволяє здійснювати порівняння декількох відстаней, частіше міри близькості одного об'єкта по відношенню до кінцевої множини інших. Недоліком інтегральної оцінки є неоднозначність її приведення до початкового векторного еквіваленту, як і будь-якого іншого функціонального відношення: пряма імплікація однозначна, зворотна – багатозначна. Тому повна картина аналізу взаємодії об'єктів повинна містити не тільки інтегральний скалярний критерій Q, але і результат їх векторного відношення Q (m, A) = m⊕A, який більш інформативний для подальшої корекції

напрямку вирішення завдань синтезу або аналізу процесів взаємодії в рамках існуючої системи. Як отримати векторний критерій якості взаємодії двох об'єктів? Формула скалярного критерію якості після проведення векторних операцій використовує процедури обчислення трьох компонентів: кодова відстань, яке визначається числом одиниць в координатах результуючого вектора, отриманого на основі xor-операції, d (m, A) = m⊕A і дві функції приналежності:

$$\mu = \mu(m \in A) \vee \mu(A \in m) = (A \wedge \overline{m \wedge A}) \vee (m \wedge \overline{m \wedge A}),$$

які в сукупності також визначаються xor-операцією, в загальному випадку на замкнутому теоретико-множинному алфавіті:

$$\begin{aligned}\mu &= (A \wedge \overline{m \wedge A}) \vee (m \wedge \overline{m \wedge A}) = [A \wedge (\overline{m} \vee \overline{A})] \vee [m \wedge (\overline{m} \vee \overline{A})] = \\ &= [(A \wedge \overline{m}) \vee (A \wedge \overline{A}) \vee (m \wedge \overline{m}) \vee (m \wedge \overline{A})] = \\ &= (A \wedge \overline{m}) \vee (m \wedge \overline{A}) = m \oplus A.\end{aligned}$$

Логічне об'єднання двох векторних функцій, які формують кодову відстань і взаємну належність один одному, дає, природно, шуканий результат:

$$Q = d(m, A) \vee [\mu(m \in A) \vee \mu(A \in m)] = (m \oplus A) \vee (m \oplus A) = m \oplus A.$$

Це означає, що по суті взаємодія будь-яких об'єктів в кіберпросторі визначається виконанням симетричної різниці в багатозначному алфавіті (xor-операції в двійковому):

Δ	0	1	x	∅
0	∅	x	1	0
1	x	∅	0	1
x	1	0	∅	x
∅	0	1	x	∅

0=01; 1=10
x=11; ∅=00 →

⊕	0	1	x	∅
0	00	11	10	01
1	11	00	01	10
x	10	01	00	11
∅	01	10	11	00

Але при кодуванні символів алфавіту двійковими векторами-примітивами операція симетричної різниці між символами в координатах векторів перетворюється в xor-операцію довічних векторів. Інші логічні операції при формуванні векторної оцінки взаємодії об'єктів в кіберпросторі, згідно з наведеними вище формулами, не використовуються. Як приклад, нижче запропоновані процедури виконання операції симетричної різниці і xor над двома формами об'єктів, представленими у вигляді символів алфавіту Кантора і двійкових кодів:

m =	x	x	x	x	1	0	1	0
A =	1	0	0	x	x	x	1	0
Δ =	0	1	1	∅	0	1	∅	∅
m =	11	11	11	11	10	01	10	01
A =	10	01	01	11	11	11	10	01
⊕ =	01	10	10	00	01	10	00	00

Другий приклад ілюструє обчислення взаємодії векторів в двотактному алфавіті опису автоматних змінних B2(Y) в форматах символьного і довічного опису координат:

m =	Y	A	B	S	P	L	E	Q
A =	H	S	J	L	E	L	F	C
Δ =	L	B	H	E	H	∅	Y	Y
m =	1111	1100	0011	1001	0110	1101	0100	1000
A =	0010	1001	0001	1101	0100	1101	1011	0111
⊕ =	1101	0101	0010	0100	0010	0000	1111	1111

Тут цікавий факт, що в кубітному форматі опису символьних змінних теоретико-множинні, в загальному випадку послідовно виконувані, операції над елементами множин замінюються паралельними операціями, що істотно підвищує швидкодію обчислювальних процесів аналізу моделей за рахунок

відповідного збільшення обсягу пам'яті. Для створення кубітних структур даних обчислювальних процесів необхідно визначити: 1) універсум примітивів (процесів або явищ) з подальшим їх унітарним кодуванням в межах кубіта; 2) компактну систему (структуру) відносин (функціональних), які задають поведінку об'єкта; 3) послідовність обробки компонентів структури на основі паралельного виконання векторних логічних операцій, які вигідно відрізняють теоретико-множинні, послідовні в часі, обчислювальні процедури.

Дві форми (скалярна і векторна) існування критерію якості q = {Q, Q (m, A)} спрямовані на вибір кращого рішення (для користувача) і деталізацію відмінностей між об'єктами (для комп'ютера) відповідно. Чисельний еквівалент зручний для людини, яка не здатна оперувати лінгвістичними (багатозначними) змінними при оцінці взаємодії об'єктів, представлених векторами. До того ж дві однакові чисельні оцінки не означають ідентичності двох відстаней при взаємодії трьох об'єктів у просторі.

Наприклад: d (a, b) = 0011 = 2, d (a, c) = 1100 = 2, при a = 0000, b = 1100, c = 0011. Тому до скалярної оцінки необхідно мати векторний еквівалент критерію якості взаємодії, який показує структуру подібності та відмінності за всіма параметрами (змінним) векторів.

Обчислити критерій – визначити ступінь належності чи неналежності даного процесу або явища, в тому числі до деякого класу об'єктів. Така класифікація шляхом порівняння аналізованого об'єкта з сімейством, але представленим у формі одного узагальненого вектора, дає можливість істотно підвищити швидкодію завдань аналізу структур даних. Для цього необхідно створювати ієрархічні формати структур даних, орієнтованих на компактне уявлення спеціальним чином закодованих об'єктів. При поданні об'єкта кіберпростору сукупністю теоретико-множинних або кубітних змінних структура вектора ділиться на сегменти, відповідні кубітам. Кубітна змінна (кубіт) – сукупність n двійкових розрядів, необхідних для унітарної кодування n примітивів і булеана породжених символів. Форми подання

вектора кубітних змінних: символьна і / або кубітно-двійкова орієнтовані на паралельне виконання теоретико-множинних операцій $(\cap,\cup,\widetilde{m})$ за допомогою алгебри векторної логіки $(\wedge,\vee,\overline{m})$. Приклади таких операцій в згаданих форматах мають вигляд:

m =	Y	A	B	S	P	L	E	Q
A =	H	S	J	L	E	L	F	C
∩ =	H	Q	J	S	E	L	∅	∅
m =	1111	1100	0011	1001	0110	1101	0100	1000
A =	0010	1001	0001	1101	0100	1101	1011	0111
∨ =	0010	1000	0001	1001	0100	1101	0000	0000
m =	Y	A	B	S	P	L	E	Q
$\widetilde{m}$ =	∅	B	A	P	S	H	F	C
m =	1111	1100	0011	1001	0110	1101	0100	1000
$\overline{m}$ =	0000	0011	1100	0110	1001	0010	1011	0111

При аналізі кубітно-двійкових форм представлення об'єктів в цілях визначення відстаней між ними необхідно враховувати: 1) Кодова відстань формується при наявності хоча б одного кубіта, рівного нулю по всіх його координатах. 2) В іншому випадку обчислюються функції належності на підставі підрахунку загального числа одиниць, отриманого при виконанні векторної операції кон'юнкції, віднесених до кількості одиниць кожного з векторів, що відповідають двом різним об'єктам кіберпростору. 3) Xor-сума відстаней об'єктів, що становлять цикл, дорівнює вектору, складеному з нульових кубітів. 4) Xor-сума всіх примітивів кубіта дорівнює вектору, який має всі одиничні координати. 5) Формування багатозначних сигнатур на основі кубітних структур даних може істотно розширити сферу застосування апарату xor-поліномів з нелінійними зворотними зв'язками. 6) Неструктурована множина примітивів, що самоорганізується в процесі моделювання або вирішення конкретного завдання, істотно зменшує обсяг моделей і час їх створення. 7) Реалізація дерева класифікації і процедур його аналізу значно скорочує обсяг структур даних, а також час вирішення відповідних завдань. Приклад такого дерева представлений на рис. 2.6, яке,

завдяки бінарної, виконує класифікацію (спуск по дереву) за мінімальне число кроків обчислювальної процедури.

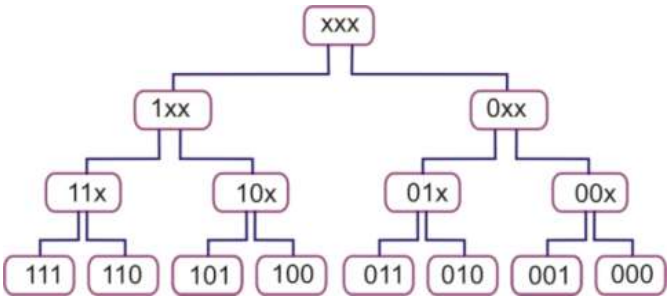


Рисунок 2.6 – Приклад класифікаційного бінарного дерева

Процедура класифікації: 1) Аналіз і-го розряду вхідного вектора m для вибору лівої чи правої гілки вершини дерева:

$$P = \begin{cases} P^0 \leftarrow m_i \oplus A_i = 0; \\ P^1 \leftarrow m_i \oplus A_i = 1. \end{cases}$$

Тут множина А визначає узагальнені коди-сигнатури, а також кінцеві вершини дерева. 2) Аналіз закінчується позитивно, якщо оброблені всі розряди вхідного вектора, який ідентифікований існуючим аналогом в бібліотеці. В іншому випадку об'єкт не може бути ідентифікований в рамках системи, яка повинна бути розширена. 3) Якщо результат аналізу має неоднозначність по відношенню до 0 і 1, то об'єкт ідентифікується вже не примітивом, а класом (підкласом). Час виконання процедури класифікації визначається виразом: $T=\log_2 N$, що є заслугою надлишкових вершин, які дозволяють систему з N відносин (нижній рівень кодів) представити у вигляді дерева.

Таким чином, запропонована модифікована модель критерію скалярної і векторної якості оцінювання бінарних відносин, яка відрізняється використанням функції неналежності та кодової відстані Хеммінга, що забезпечує лінійність зміни чисельного значення критерію від 0 до 1 в міру

збільшення «відстані» від повного збігу двох об'єктів до максимально можливого , коли кодова відстань одно d (m, A) = k. Критерій може бути використаний при оцінюванні взаємодії об'єктів в реальному масштабі часу в задачах тестування, діагностування функціональних порушень, вразливостей.

2.5. Апарат булевих похідних для синтезу тестів

Апарат призначений для перевірки суттєвості змінних і компонентів КС, включаючи аналіз суттєвості деструктивності (уразливості і проникнення) для стану кіберсистеми. Розглядаються методи взяття булевих похідних по таблиці істинності, диз'юнктивній формі або кубічному покриттю для створення умов активізації на вхідних змінних при синтезі тестів для перевірки вразливостей (проникнень). Дослідження методу пропонується виконати за допомогою трьох прикладів логічних функцій:

$$1)\ f(x) = x_1 \vee x_1\bar{x}_2;\ 2)\ f(x) = x_1x_2 \vee \bar{x}_1x_3\ ;\ 3)\ f(x) = \bar{x}_2\bar{x}_3 \vee x_1x_2x_3.$$

Питання, що підлягають вирішенню: 1) Визначення всіх похідних першого порядку по аналітичній, кубічній і табличній формі завдання логічної функції. 2) Верифікація отриманих умов активізації шляхом їх моделювання на одній з форм опису функціональності. 3) Синтез тестів активізації змінних логічної функції на основі обчислення похідних.

Приклад 2.1. Визначити всі похідні першого порядку по аналітичній формі логічної функції $f(x) = x_1 \vee x_1\bar{x}_2$. Застосування формули обчислення

$$f'(x_i) = df(x_1, x_2, \ldots, x_i, \ldots, x_n)/dx_i =$$
$$= f(x_1, x_2, \ldots, x_i=0, \ldots, x_n) \oplus f(x_1, x_2, \ldots, x_i=1, \ldots, x_n)$$

визначає булеву похідну першого порядку як суму по модулю для нульової і одиничної залишкових функцій.

Для даної функції виходить:

$$\frac{df(x_1, x_2)}{dx_1} = f(0, x_2) \oplus f(1, x_2) =$$

$$= (0 \vee 0\bar{x}_2) \oplus (1 \vee 1\bar{x}_2) = 0 \oplus 1 = 1;$$

$$\frac{df(x_1, x_2)}{dx_2} = f(x_1, 0) \oplus f(x_1, 1) =$$

$$=(x_1 \vee x_1 \cdot 1) \oplus (x_1 \vee x_1 \cdot 0) = x_1 \oplus x_1 = 0.$$

Нульове значення похідної означає відсутність умов активізації змінної x2, що дає підстави вважати її несуттєвою, а отже, прибрати з числа змінних, які формують функціональність.

Приклад 2.2. Визначити всі похідні першого порядку по аналітичній формі логічної функції $f(x) = x_1x_2 \vee \bar{x}_1x_3$. Для даної функції виконуються такі обчислення:

$$\frac{df(x_1, x_2, x_3)}{dx_1} = f(0, x_2, x_3) \oplus f(1, x_2, x_3) =$$

$$=(0 \vee 1 \cdot x3) \oplus (x2 \vee 0 \cdot x3) = x3 \oplus x2 = x_2\overline{x_3} \vee \overline{x_2}x_3;$$

$$\frac{df(x_1, x_2, x_3)}{dx_2} = f(x_1, 0, x_3) \oplus f(x_1, 1, x_3) = \bar{x}_1x_3 \oplus (x_1 \vee x_3) =$$

$$\overline{\bar{x}_1x_3}(x_1 \vee x_3) \vee \bar{x}_1x_3\overline{(x_1 \vee x_3)} = (x_1 \vee \bar{x}_3)(x_1 \vee x_3) \vee \bar{x}_1x_3\bar{x}_1\bar{x}_3 = x_1.$$

$$\frac{df(x_1, x_2, x_3)}{dx_3} = f(x_1, x_2, 0) \oplus f(x_1, x_2, 1) = x_1x_2 \oplus (\bar{x}_1 \vee x_2) =$$

$$\overline{x_1x_2}(\bar{x}_1 \vee x_2) \vee x_1x_2\overline{(\bar{x}_1 \vee x_2)} = (\bar{x}_1 \vee \bar{x}_2)(\bar{x}_1 \vee x_2) \vee x_1x_2x_1\bar{x}_2 = \bar{x}_1.$$

Для трьох змінних отримані 4 умови активізації, які відповідають чотирьом логічним шляхav в схемній структурі диз'юнктивної форми даної функції.

Приклад 2.3. Визначити всі похідні першого порядку по кубічній формі логічної функції:

$f(x) = \bar{x}_2\bar{x}_3 \vee x_1x_2x_3 =$

x_1	x_2	x_3	Y
X	0	0	1
1	1	1	1
X	0	1	0
X	1	0	0
0	1	X	0

=

x_2x_3 / x_1	00	01	11	10
0	1	0	0	0
1	1	0	1	0

=

x_1	x_2	x_3	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Процес-модель обчислення похідної по змінній для функції x_i, заданої табличною формою, має такі пункти: 1) Моделювання по таблиці істинності (кубічному покриттю) вхідних наборів для визначення стовпчика Y_i^0, де змінна x_i має тільки нульове значення для всіх рядків таблиці істинності. Число таких наборів завжди $q=2^{n-1}$, N – число змінних. 2) Обчислення координат стовпчика Y_i^1 з одиничним значенням змінної $Y_i^{\oplus} = Y_i^0 \oplus Y_i^1$ для всіх рядків таблиці. 3) Обчислення стовпчика з урахуванням правила $0 \oplus X \vee 1 \oplus X = X$. 4) Формування диз'юнктивної форми похідної функції за одиничними значеннями стовпчика $Y_i^{\oplus}$ без змінної хі, по якій береться похідна. Інакше, фіксуються рядки таблиці, відповідні одиничним значенням стовпця $Y_i^{\oplus}$, який визначає похідну функції. Аналітична модель процесу взяття похідної по функції, представленої таблицею, має такий вигляд:

$$df/dx_i = f(x_1, x_2, \ldots, x_i=0, \ldots, x_n) \oplus f(x_1, x_2, \ldots, x_i=1, \ldots, x_n);$$

$$Y_i\oplus = [Y_i^0 = f(x_1, x_2, \ldots, x_i=0, \ldots, x_n)] \oplus [Y_i^1 = f(x_1, x_2, \ldots, x_i=1, \ldots, x_n)].$$

Для двох різних табличних форм результат обчислення похідної за першою змінною представлений нижче:

$$\frac{df}{dx_1} =$$

x_1	x_2	x_3	Y	Y_1^0	Y_1^1	$Y_1^{\oplus}$
X	0	0	1	1	1	0
1	1	1	1	0	1	1
X	0	1	0	0	0	0
X	1	0	0	0	0	0
0	1	X	0	0	X	1

$= x_2 \vee x_2x_3 = x_2x_3;$

$$\frac{df}{dx_1} =$$

x_1	x_2	x_3	Y	Y_1^0	Y_1^1	$Y_1^{\oplus}$
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	1	1	1	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	1	0	1	1

$= x_2x_3.$

При неоднозначному значенні похідної для функції, заданої кубічним покриттям, вибирається терм, що має максимальне число змінних. Мінімізація похідної функції на основі тотожності $a \vee ab = a$ не зберігає умови активізації змінної, по якій береться похідна. Справді, значення функції від трьох змінних за умови $a\bar{b} \vee ab$ може дорівнювати нулю (одиниці), що означає можливість відсутності зміни функції при активізації змінної «c» (останній стовпець карти Карно):

ab / c	00	01	11	10
0	1	0	0	0
1	1	0	1	0

Лема неперетинання кубів. Можливість коректного взяття похідної для отримання тесту активізації по змінній x_i обмежується такою мінімальною структурою кубічного покриття або аналітичною диз'юнктивною (кон'юнктівною) нормальною формою, де перетин будь-яких кубів (рядків таблиці істинності) або термів ДНФ (КНФ) дає порожню множину:

$$df/dx_i = f(x_1, x_2, \ldots, x_i=0, \ldots, x_n) \oplus f(x_1, x_2, \ldots, x_i=1, \ldots, x_n) \in T$$
$$\leftrightarrow \forall i,j(Ci \cap Cj = \varnothing);\ i,j=1,\ldots,n;\ i \neq j.$$

Дійсно, якщо покриття, представлене вище, записати за правилами перетинання кубів, то всі похідні будуть валідними для синтезу тестів без додаткової перевірки:

$$f(x) = \bar{x}_2\bar{x}_3 \vee x_1x_2x_3 =$$

x_1	x_2	x_3	Y
X	0	0	1
1	1	1	1
X	0	1	0
X	1	0	0
0	1	X	0

$\rightarrow$

x_1	x_2	x_3	Y
X	0	0	1
1	1	1	1
X	0	1	0
X	1	0	0
0	1	1	0

Щоб отримати таке кубічне покриття, необхідно виконувати мінімізацію усіма існуючими методами (карти Карно, Квайна, істотних змінних, невизначених коефіцієнтів, бінарного графа) з урахуванням цього правила: покриття нульових і одиничних координат таблиці істинності в процесі мінімізації не повинні перетинатися. В даному випадку, коли функціональність переписана з урахуванням даного правила, навіть загальне число кубів не змінилося, в той час як покриття набуло якості неперетинання (як у таблиці істинності) для синтезу тестів активізації змінних:

$$\frac{df}{dx_1} =$$

x_1	x_2	x_3	Y	Y_2^0	Y_2^1	$Y_2^{\oplus}$
X	0	0	1	1	1	0
1	1	1	1	0	1	1
X	0	1	0	0	0	0
X	1	0	0	0	0	0
0	1	1	0	0	0	0

$= x_2x_3;$

$$\frac{df}{dx_1} =$$

x_1	x_2	x_3	Y	Y_2^0	Y_2^1	$Y_2^{\oplus}$
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	1	1	1	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	1	0	1	1

$= x_2x_3.$

Обчислення похідних по всіх вхідних змінних дає можливість побудувати тест активізації для функціональності, заданої вже не таблицею

істинності, а кубічним покриттям, що може істотно зменшити час синтезу тестів. Для другої змінної функції $f(x) = \bar{x}_2\bar{x}_3 \vee x_1x_2x_3$ процес обчислення похідних для трьох різних форм (кубічної, табличної і аналітичної) має наступний вигляд:

$$\frac{df}{dx_2} =$$

x_1	x_2	x_3	Y	Y_2^0	Y_2^1	$Y_2^{\oplus}$
X	0	0	1	1	0	1
1	1	1	1	0	1	1
X	0	1	0	0	X	X
X	1	0	0	1	0	1
0	1	1	0	0	0	0

$= \bar{x}_3 \vee x_1x_3 \vee \bar{x}_3 = \bar{x}_3 \vee x_1x_3;$

x_1	x_2	x_3	Y	Y_2^0	Y_2^1	$Y_2^{\oplus}$
0	0	0	1	1	0	1
0	0	1	0	0	0	0
0	1	0	0	1	0	1
0	1	1	0	0	0	0
1	0	0	1	1	0	1
1	0	1	0	0	1	1
1	1	0	0	1	0	1
1	1	1	1	0	1	1

$= \vee \bar{x}_1\bar{x}_3 \vee x_1\bar{x}_3 \vee x_1x_3 = \bar{x}_3 \vee x_1x_3;$

Аналогічний результат отриманий шляхом визначення похідної за диз'юнктивною нормальною формою логічної функції:

$$df(x_1,x_2,x_3)/dx_2 = f(x_1,0,x_3) \oplus f(x_1,1,x_3) = x_1x_3 \vee \bar{x}_3.$$

Для третьої змінної функції $f(x) = \bar{x}_2\bar{x}_3 \vee x_1x_2x_3$ похідні від трьох різних форм (кубічної, табличної і аналітичної) представлені у такому вигляді:

$$\frac{df}{dx_3} =$$

x_1	x_2	x_3	Y	Y_3^0	Y_3^1	$Y_3^{\oplus}$
X	0	0	1	1	0	1
1	1	1	1	0	1	1
X	0	1	0	0	0	0
X	1	0	0	0	X	X
0	1	1	0	0	0	0

$= \bar{x}_2 \vee x_1 x_2;$

x_1	x_2	x_3	Y	Y_3^0	Y_3^1	$Y_3^{\oplus}$
0	0	0	1	1	0	1
0	0	1	0	1	0	1
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	1	0	1
1	0	1	0	1	0	1
1	1	0	0	0	1	1
1	1	1	1	0	1	1

$= \vee \bar{x}_1 \bar{x}_2 \vee x_1 \bar{x}_2 \vee x_1 x_2 = \bar{x}_2 \vee x_1 x_2.$

$$df(x_1,x_2,x_3)/dx_3 = f(x_1,x_2,0) \oplus f(x_1,x_2,1) = x_1 x_2 \vee \bar{x}_2.$$

Таким чином, всі результати по обчисленню похідних від трьох форм завдання функції ідентичні. Найбільш технологічним є метод взяття похідної по таблиці істинності. Але використання кубічного покриття має меншу обчислювальну складність в силу компактного представлення функціональності за рахунок введення надмірності (символу X) в двійковий алфавіт. Використання аналітичної форми передбачає істотне підвищення складності алгоритмів, пов'язаної із застосуванням законів булевої алгебри і мінімізації функцій, що обмежує її застосування для вирішення практичних завдань.

Процес-модель отримання тесту

$$T = [T_{ij}],\ i = \overline{1,k};\ j = \overline{1,n}$$

комбінаційної функціональності:

1) $f'(x_i) = f(x_1, x_2, ..., x_i = 0, ..., x_n) \oplus f(x_1, x_{2,...,} x_i = 1, ..., x_n);$

2) $T = \bigcup_{i=1}^{n} [f'(x_i) * (x_i = 0) \vee (x_i = 1)];$

3) $T_{ij} = T_{i-1,j} \leftarrow T_{ij} = X;\ T_{1j} = 1 \leftarrow T_{1j} = X;$

4) $T = T \setminus T_i \leftarrow T_i = T_{i-r},\ r = \overline{1, i-1},\ i = \overline{2, n}.$

1) Обчислення похідних по всіх n змінних функціональності шляхом використання однієї з форм: аналітичної, табличної, кубічної. 2) Об'єднання всіх умов (векторів) активізації в таблицю, де кожному вектору шляхом конкатенації (*) ставиться у відповідність зміна адреси, за якою була взята похідна, що означає подвоєння числа тестових наборів по відношенню до загальної кількості (k) умов активізації. 3) Довизначення символу X = {0,1} в координаті шляхом присвоєння довічного значення однойменної координати в попередньому векторі для отримання тесту мінімальної довжини. 4) Мінімізація тестових векторів шляхом видалення повторюваних вхідних послідовностей.

Рис. 2.7 ілюструє таблиці ходу отримання тесту відповідно до пунктів 2-4 алгоритму для функціональності $f(x) = \bar{x}_2\bar{x}_3 \vee x_1x_2x_3$, представленої схемною структурою.

T =

x_1	x_2	x_3	Y
0	1	1	0
1	1	1	1
X	0	0	1
X	1	0	0
1	0	1	0
1	1	1	1
X	0	0	1
X	0	1	0
1	1	0	0
1	1	1	1

=

x_1	x_2	x_3	Y
0	1	1	0
1	1	1	1
1	0	0	1
1	1	0	0
1	0	1	0
1	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

=

x_1	x_2	x_3	Y
0	1	1	0
1	1	1	1
1	0	0	1
1	1	0	0
1	0	1	0

Рисунок 2.7 – Таблиці тестів і схемна структура булевої функції

Отриманий тест за кількістю і якістю ідентичний вхідним наборам, синтезованим раніше за допомогою F⊕L-методу. Отже, він має однакові властивості по покриттю несправностей і глибині пошуку дефектів.

Запропонована процес-модель синтезу тестів для тестування і діагностування вразливостей може бути використана як вбудований компонент інфраструктури сервісного обслуговування КС.

2.6. Дедуктивний метод пошуку вразливостей в КС

Основна ідея дедуктивного методу полягає в аналізі зіставлення вхідних і вихідних даних кіберсистеми з метою виявити деструктивні проникнення або уразливості шляхом виконання процедур порівняння між свідомо штатними (функціональними) режимами і ситуаціями, що викликають підозру. Для імплементації методу в інфраструктуру захисних сервісів необхідно мати графову модель логіки функціонування кіберсистеми, яка досить просто може бути трансформована до системи логічних рівнянь, придатної для дедуктивного аналізу. Далі пропонується модель дедуктивно-паралельного синхронного аналізу вразливостей (проникнень) кіберсистеми (об'єкта), яка дозволяє за одну ітерацію обробки структури обчислити всі деструктивні компоненти, що перевіряються на тест-векторі. Мета дедуктивного аналізу – визначити якість синтезованого тесту щодо повноти покриття їм вразливостей, а також побудувати таблицю перевірки тестовими наборами усіх виявлених вразливостей КС для виконання процедур діагностування. Така модель заснована на рішенні рівняння:

$$L = T \oplus F, \qquad (2.1)$$

де $F = (Fm+1, Fm+2, ..., Fi, ..., Fn)$, $i = m+1, ..., n$ – сукупність функцій справної (коректної) поведінки КС; m – число його входів; $Yi=Fi(Xi1, ..., Xij, ..., Xini)$ – ni-входовий i-й елемент схеми, що реалізує Fi для визначення стану лінії (виходу) Yi на тест-векторі Tt; тут Xij – j-й вхід i-го елемента; тест $T=(T1, T2, ..., Tt, ..., Tk)$ – упорядкована сукупність двійкових векторів, визначена в процесі справного моделювання на множині вхідних, внутрішніх і вихідних ліній, об'єднана в матрицю

$$T = [T_{ti}] = \begin{bmatrix} T_{11}, T_{12}, \ldots, T_{1i}, \ldots, T_{1n} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ T_{t1}, T_{t2}, \ldots, T_{ti}, \ldots, T_{tn} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ T_{k1}, T_{k2}, \ldots, T_{ki}, \ldots, T_{kn} \end{bmatrix}, \quad (2.2)$$

невхідна координата якої визначається моделюванням функції Tti = Yi = Fi (Xi1, ..., Xij, ..., Xini) на тест-векторі Tt;

$$L = (L_1, L_2, \ldots, L_t, \ldots, L_k)$$

– множина дедуктивних схем або моделей, які визначаються виразом (2.3), де

$$L_t = (L_{t1}, L_{t2}, \ldots, L_{ti}, \ldots, L_{tn});$$

$$L_{ti} = T_t \oplus F_i \quad (2.3)$$

– дедуктивна функція (ДФ) паралельного моделювання несправностей на тест-векторі T_t, відповідна справному елементу F_i, яка дає можливість обчислювати список вхідних проникнень, що транспортуються на вихід елемента F_i [17].

Поняття синхронності введеної моделі (2.1) визначається умовою:

$$\Delta t = (t_{j+1} - t_j) >> \tau >> \tau_i,$$

коли інтервал часу між зміною вхідних наборів (t_{j+1}–t_j), що подаються на КС, набагато більший від максимальної затримки системи τ і елемента τ_i. Це дозволяє виключити час як несуттєвий параметр [17], що використовується в технологіях моделювання та синтезу тестів.

У загальному випадку, коли функція КС представлена таблицею істинності, застосування формули (2.1) дозволяє отримати для заданого тест-вектора T_t таблицю транспортування вразливостей (проникнень), по якій можна записати ДФ моделювання деструктивностей. Приклади отримання таких функцій представлені в такому вигляді (перший доданок – тест-вектор, другий і результат – таблиці істинності і транспортування вразливостей):

X_1	X_2	Y_1
0	1	0

$\oplus$

X_1	X_2	Y_1
0	0	0
0	1	0
1	0	1
1	1	1

$=$

X_1	X_2	L_1
0	1	0
0	0	0
1	1	1
1	0	1

$L_1 = X_1X_2 \vee X_1\overline{X}_2;$

X_1	X_2	Y_2
1	1	1

$\oplus$

X_1	X_2	Y_2
0	0	0
0	1	0
1	0	0
1	1	1

$=$

X_1	X_2	L_2
1	1	1
1	0	1
0	1	1
0	0	0

$L_2 = X_1X_2 \vee X_1\overline{X}_2 \vee \overline{X}_1X_2.$

Тут дедуктивні функції L1, L2 записані у вигляді диз'юнктивної нормальної форми по конституентах одиниці таблиць транспортування деструктивностей.

З урахуванням розбиття тесту на складові вектори рівняння (2.1) отримання ДФ для Tt∈T приймає такий вигляд: Lt=Tt⊕F. Якщо функціональний опис КС представлено компонентами (примітивами), що формують стани всіх ліній (з'єднань) КС, то як формула перетворення справної моделі примітива Fi на тест-векторі Tt в дедуктивну функцію Lti виступає такий вираз:

$$L_{ti} = T_t \oplus F_i = f_{ti}[(X_{i1} \oplus T_{t1}),(X_{i2} \oplus T_{t2}),\ldots, (X_{ij} \oplus T_{tj}),\ldots,(X_{in_i} \oplus T_{tn_i})] \oplus T_{ti}, \quad (2.4)$$

який є основою дедуктивного аналізу деструктивних порушень КС [3, 6].

Приклад 2.4. Отримати дедуктивні функції паралельного моделювання вразливостей на вичерпному тесті для базису функціональних елементів And, Or, Not. З урахуванням виразу (2.4) виконуються такі очевидні перетворення для функції And:

$$L_{and}[T=(00,01,10,11), F=(X_1 \wedge X_2)] =$$
$$= L\{(\bar{x}_1\bar{x}_2 \vee \bar{x}_1x_2 \vee x_1\bar{x}_2 \vee x_1x_2) \wedge [(X_1 \oplus T_{t1} \wedge X_2 \oplus T_{t2}) \oplus T_{t3})]\} =$$
$$= (\bar{x}_1\bar{x}_2)\{[(X_1 \oplus 0) \wedge (X_2 \oplus 0)] \oplus 0\} \vee (\bar{x}_1x_2)\{[(X_1 \oplus 0) \wedge (X_2 \oplus 1)] \oplus 0\} \vee$$
$$\vee (x_1\bar{x}_2)\{[(X_1 \oplus 1) \wedge (X_2 \oplus 0)] \oplus 0\} \vee (x_1x_2)\{[(X_1 \oplus 1) \wedge (X_2 \oplus 1)] \oplus 1\} =$$
$$= (\bar{x}_1\bar{x}_2)(X_1 \wedge X_2) \vee (\bar{x}_1x_2)(X_1 \wedge \bar{X}_2) \vee (x_1\bar{x}_2)(\bar{X}_1 \wedge X_2) \vee (x_1x_2)(X_1 \vee X_2).$$

Аналогічно виконуються обчислення для функції Or:

$$L_{or}[T=(00,01,10,11), F=(X_1 \vee X_2)] =$$
$$= L\{(\bar{x}_1\bar{x}_2 \vee \bar{x}_1x_2 \vee x_1\bar{x}_2 \vee x_1x_2) \wedge [(X_1 \oplus T_{t1} \vee X_2 \oplus T_{t2}) \oplus T_{t3})]\} =$$
$$= (\bar{x}_1\bar{x}_2)\{[(X_1 \oplus 0) \vee (X_2 \oplus 0)] \oplus 0\} \vee (\bar{x}_1x_2)\{[(X_1 \oplus 0) \vee (X_2 \oplus 1)] \oplus 1\} \vee$$
$$\vee (x_1\bar{x}_2)\{[(X_1 \oplus 1) \vee (X_2 \oplus 0)] \oplus 1\} \vee (x_1x_2)\{[(X_1 \oplus 1) \vee (X_2 \oplus 1)] \oplus 1\} =$$
$$= (\bar{x}_1\bar{x}_2)(X_1 \vee X_2) \vee (\bar{x}_1x_2)(\bar{X}_1 \wedge X_2) \vee (x_1\bar{x}_2)(X_1 \wedge \bar{X}_2) \vee (x_1x_2)(X_1 \wedge X_2).$$

Тут $T_t=(T_{t1}, T_{t2}, T_{t3})$, t=1,2,3,4 – тест-вектор, який має 3 координати, де остання з них визначає стан виходу двохвходового елемента And (Or). У наступному перетворенні $T_t=(T_{t1}, T_{t2})$, t=1,2 – тест-вектор, який має 2 координати, де друга – стан виходу інвертора:

$$L_{not}[T=(0,1), F=\bar{X}_1] = L\{(\bar{x}_1 \vee x_1)[(\overline{X_1 \oplus T_{t1}}) \oplus T_{t2}]\} =$$
$$= \bar{x}_1[(\overline{X_1 \oplus 0}) \oplus 1] \vee x_1[(\overline{X_1 \oplus 1}) \oplus 0] = \bar{x}_1\bar{\bar{X}}_1 \vee x_1\bar{\bar{X}}_1 = \bar{x}_1X_1 \vee x_1X_1.$$

Останній вираз ілюструє інваріантність інверсії до вхідного набору для транспортування вразливостей. Вона трансформується в повторювач. Тому

дана функція не фігурує на виходах дедуктивних елементів. Спільна апаратурна реалізація ДФ для решти двохвходових елементів And, Or на вичерпному тесті представлена універсальним функціональним примітивом (рис. 2.8) дедуктивно-паралельного аналізу несправностей.

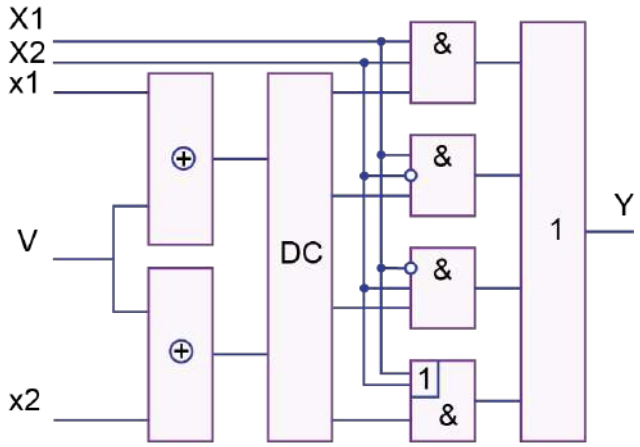


Рисунок 2.8 – Симулятор несправних примітивів

У симуляторі представлені булеві (x1, x2) і реєстрові (X1, X2) для кодування вразливостей входи, змінна вибору типу справної функції (AND, OR), вихідна реєстрова змінна Y. Стани двійкових входів x1, x2 і змінна вибору елемента визначають одну з чотирьох дедуктивних функцій для отримання вектора Y перевірки несправностей.

Для ілюстрації паралельного моделювання вхідних 8-розрядних векторів вразливостей в цілях отримання на виході Y множини перевірки деструктивниостей для логічних елементів 2And, 2Or використовується така таблиця:

(V, x1, x2) =	000	100	011	111	010	110
X1(RG)	01110001	01110001	10110110	00111011	00101010	10111001
X2(RG)	01111000	01111000	10110101	00110100	10111001	00101010
Y(RG)	01110000	01111001	10110111	00110000	10010001	10010001

Застосування такого симулятора дає можливість трансформувати функціональну модель F коректної поведінки КС в дедуктивну L, яка інваріантна в сенсі універсальності тестовим наборам і не передбачає в процесі моделювання використовувати модель F. Тому симулятор, як апаратна модель ДФ, є ефективним двигуном дедуктивно-паралельного моделювання КС, що підвищує швидкодію аналізу кіберсистем в 10 – 1000 разів у порівнянні з програмною реалізацією. Але при цьому співвідношення обсягів моделей коректного моделювання та аналізу вразливостей становить 1:10. Підхід апаратного аналізу деструктивний, спрямований на розширення функціональних можливостей вбудованих засобів моделювання, які можна зберігати на хмарі і постійно ними користуватися для верифікації інфраструктури захисту КС. Обчислювальна складність обробки проекту $Q=(2n^2r)/W$, де r – час виконання регістрової операції (And, Or, Not); W – розрядність регістра.

Для апаратної реалізації дедуктивно-паралельного моделювання на основі запропонованого симулятора може бути використана обчислювальна структура, представлена на рис. 2.9. Особливість схемної реалізації полягає в спільному виконанні двох операцій: однобітових – для емуляції функцій логічних елементів And, Or і паралельної – для обробки багаторозрядних векторів несправностей шляхом виконання операцій логічного множення, заперечення і складання.

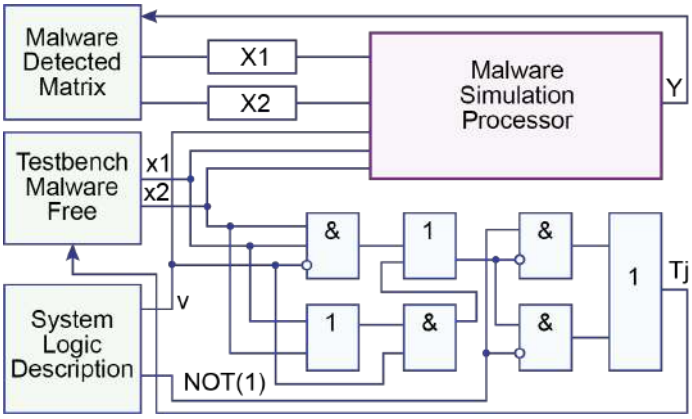


Рисунок 2.9 – HFS-структура апаратного моделювання

Функціональне призначення основних блоків (пам'ять і процесор): 1. M=[M_{ij}] – квадратична матриця моделювання деструктивних проникнень (ДП), де i, j = 1, q; q – загальне число ліній в оброблюванsq КС. 2. Вектори збереження станів коректного моделювання, визначені в моменти часу t-1 і t, необхідні для формування дедуктивних функцій примітивів. 3. Модуль пам'яті для зберігання опису КС у вигляді структури логічних елементів. 4. Буферні регістри, розмірністю q, для зберігання операндів і виконання регістрових паралельних операцій над векторами ДП, що зчитані з матриці M. 5. Блок коректного моделювання для визначення двійкового стану виходу чергового оброблюваного логічного елемента. 6. Дедуктивно-паралельний симулятор, що обробляє за один такт дві реєстрових змінних X1, X2 з метою визначення вектора ДП, що транспортуються на вихід логічного елемента Y.

Перевага запропонованої структури моделювання ДП. 1. Суттєве зменшення кількості модельованих ДП, які визначаються тільки числом збіжних розгалужень, що становить до 20% від загального числа ліній. 2. Зниження обсягу пам'яті, необхідного для зберігання матриці модельованих ДП. 3. Простота реалізації Hardware Vulnerability Simulator (HVS) в апаратному виконанні, що дозволяє на порядок збільшити швидкодію моделювання ДП. 4. Використання HVS в якості першої фази дедуктивно-топологічного методу, який грунтується на результаті обробки розгалужень, що сходяться, для швидкодіючого аналізу деревоподібних структур.

Маршрут моделювання КС з попереднім розбиттям моделі пристрою на дві структурні частини (розгалуження, що сходяться, і деревовидні підграфи) представлений на рис. 2.10.

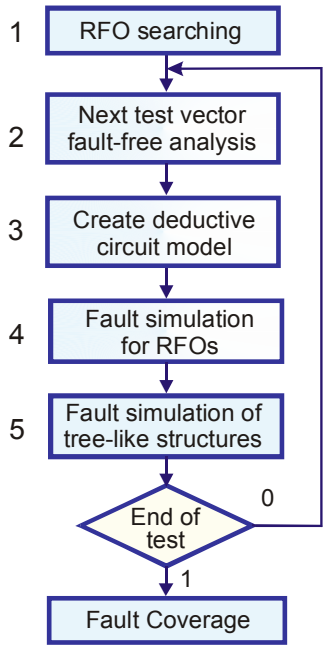


Рисунок 2.10 – Модель процесу дедуктивно-паралельного моделювання

Підсумки запропонованої технології моделювання з попередніми розбиттям КС на розгалуження, що сходяться, і деревовидні підграфи. Дедуктивно-паралельний аналіз ДП на основі їх зворотного простежування вимагає практично лінійних витрат пам'яті і часу, що залежать від числа ліній КС. Витрати часу для обробки розгалужень, що сходяться, мають квадратичну залежність від їх числа:

$$Q = (r^2 / W) + n_r + n_p + (n - r - r^0).$$

Тут (r^2 / W) – час моделювання ДП r розгалужень, що сходяться, число яких визначається як $r=0.2\times n$, $n_r=n$ – час реконфігурування примітивів схеми на вхідному наборі; $n_p=n$ – час пошуку підграфів ліній, відповідних неперевірюваним розгалуженням, що збігаються;

$$(n - r - r^0) = n - 0.2\times n - 0.4\times n = 0.4\times n$$

– час виконання суперпозиції рішень на множині ліній КС без сходяться розгалужень і попередників для непроверяемих сходяться розгалужень.

З огляду на значення параметрів функції від числа ліній, можна отримати оцінку швидкодії дедуктивно-паралельного методу [14 – 16]:

$$Q = [(0.2 \times n)^2 / W] + n + n + (n - 0.2 \times n - 0.4 \times n) = [(0.2 \times n)^2 / W] + 2.4 \times n).$$

Таким чином, виграш у швидкодії запропонованого методу тим більший, чим менший відсоток розгалужень, що сходяться, в КС.

Для порівняння: паралельний алгоритм має обчислювальну складність Ср, яка визначається функціональною залежністю від числа нееквівалентних ДП (b), довжини комп'ютерного слова (W), кількості еквівалентних вентилів (G):

$$C_p = (b^2 / W) \times G^3.$$

Дедуктивний алгоритм має відмінності у формулі оцінки швидкодії:

$$C_d = b^2 \times Q \times G^2 \big|_{Q=G} = b^2 G^3,$$

де Q – середнє число активізованих ДП вентилів. Дедуктивно-паралельний метод без розбивки схеми має швидкодію, що визначається виразом:

$$C_{dp} = G^2 + (b^2 / W) \times G^2.$$

Перший доданок задає час коректного моделювання, другий – час аналізу ДП, лінії якого не ранжовані. Для комбінаційної ранжованої структури швидкодія методу має оцінку

$$C_{dp}^r = G + (b^2 / W) \times G.$$

Швидкодія дедуктивно-паралельного методу вища паралельного і дедуктивного

$$(C_{dp}^{r} << \{C_p, C_d\}),$$

завдяки поділу фаз коректного і ДП моделювань.

Запропонована технологія програмно-апаратного дедуктивно-паралельного моделювання ДП орієнтована на створення моделей дедуктивних примітивів компонентів і зв'язків КС з метою тестування вразливостей (проникнень). Представлена структурна модель апаратного симулятора і пристрої моделювання в цілому, які орієнтовані на істотне підвищення швидкодії засобів моделювання КС великої розмірності шляхом поділу функцій коректного аналізу і обчислення списків перевірки вразливостей на тестових наборах.

Метод дедуктивно-паралельного моделювання дає можливість оцінювати якість (повноту) запропонованих тестів, а також визначати всі потенційно можливі місця існування вразливостей в цілях їх подальшого усунення.

2.7. Висновки до розділу 2

1. Визначено компоненти блокчейн технології, які використовуються для створення надійної інфраструктури захисту даних, складеної з ненадійних елементів.

2. Представлена структурна модель відносин на множині з чотирьох основних компонентів тестування і діагностики (функціональність, КС, тест, уразливості), яка характеризується повною xor-взаємодією всіх вершин графа і транзитивною оборотністю кожної тріади відносин, що дозволяє визначити і класифікувати шляхи вирішення практичних завдань, включаючи синтез тестів, моделювання та пошук вразливостей.

3. Запропоновано вдосконалені методи синтезу тестів для функціональностей, заданих матричними формами опису поведінки компонентів КС, які відрізняються паралелізмом векторних операцій над

таблицями, що дає можливість істотно (х2) підвищити швидкодію обчислювальних процедур.

4. Процес-моделі і методи синтезу тестів для функціональностей і діагностування ФН можуть бути використані як вбудовані компоненти інфраструктури сервісного обслуговування КС із застосуванням стандартів тестопридатності.

Результати розділу відображені у публікаціях [2, 3, 5, 6].

2.8 Список використаних джерел до розділу 2:

1. https://www.forbes.com/sites/louiscolumbus/2017/08/15/gartners-hype-cycle-for-emer-ging-technologies-2017-adds-5g-and-deep-learning-for-first-time/#646a4cf34be2

2. http://www.gartner.com/newsroom/id/3784363

3. http://www.wired.co.uk/article/ai-neuromorphic-chips-brains

4. *Gupta A. and Jha R. K.*, "A Survey of 5G Network: Architecture and Emerging Technologies," in IEEE Access, vol. 3, pp. 1206-1232, 2015.

5. *Zhu C., Leung V. C. M., Shu L. and Ngai E. C. H.,* "Green Internet of Things for Smart World," In *IEEE Access*, vol. 3, pp. 2151-2162, 2015.

6. *Christidis K. and Devetsikiotis M.,* "Blockchains and Smart Contracts for the Internet of Things," In *IEEE Access*, vol. 4, pp. 2292-2303, 2016.

7. *Blockchains*: How They Work and Why They'll Change the World IEEE Spectrum. October 2017.

[https://spectrum.ieee.org/computing/networks/blockchains-how-they-work-and-why-theyll-change-the-world]

8. *Zanella A A., Bui N., Castellani A., Vangelista L. and Zorzi M.*, "Internet of Things for Smart Cities," In IEEE IoT Journal, vol. 1, no. 1, pp. 22-32, Feb. 2014.

9. https://www.gartner.com/doc/3471559?srcId=1-7578984202&utm_campaign=RM_GB_ 2017_TRENDS_QC_E2_What&utm_medium=email&utm_source=Eloqua&cm_mmc=Eloqua-_-Email-_-LM_RM_GB_2017_TRENDS_QC_E2_What-_-0000

10. http://www.gartner.com/smarterwithgartner/three-digital-marketing-habits-to-break-2/

11. *Vladimir Hahanov,* Cyber Physical Computing for IoT-driven Services, New York, Springer, 2017. 259 p.

12. *Laura Shin*. Buying Bitcoin and Other Crypto Assets // Forbes Newsletters. 2018. 6 p.

[http://info.forbes.com/rs/790-SNV-353/images/crypto.pdf?mkt_tok=eyJpIjoiWVdOalltVTRNVEppWmpaaSIsInQiOiJkb2lLYWE1djgyV3laRkRseWh1MlNGc1N2S3hTVkFHejNDT0ZDZzV4RFA0MlV2cmRpMHNjNEp6MkU5S0VsSU1xUTRNd0RpeStQMGJ5SUdNWTMramdBTkV4V3FSY1JcL0VGYTRrcXNnTVFvMG1jdldEZ1RFcTkyWHhKQUZGZzVWb1oifQ%3D%3D]

13. https://spectrum.ieee.org/tech-talk/computing/networks/quantum-blockchains-could-act-like-time-machines?utm_source=computingtechnology&utm_campaign= computingtechnology-05-01-18&utm_medium=email

14. *Hahanov V., Wajeb Gharibi, Litvinova E., Chumachenko S.* Information analysis infrastructure for diagnosis // Information an international interdisciplinary journal. 2011. Japan. Vol.14. № 7. P. 2419-2433.

15. *Bau Jason, Bursztein Elie, Gupta Divij, Mitchell John.* State of the Art: Automated Black-Box Web Application Vulnerability Testing // 2010 IEEE Symposium on Security and Privacy. 2010. P. 332 – 345.

16. *Shahriar H., Zulkernine M.* Automatic Testing of Program Security Vulnerabilities // 33rd Annual IEEE International Computer Software and Applications Conference. 2009.Vol. 2. P. 550 – 555.

17. *Sedaghat S., Adibniya F., Sarram M.-A.* The investigation of vulnerability test in application software // International Conference on the Current Trends in Information Technology (CTIT). 2009. P.1 – 5.

18. *Хаханов В.И., Anders Carlsson, Чумаченко С.В.* Инфраструктура pentesting и управления уязвимостью // Автоматизированные системы управления и приборы автоматики. 2012. Вып. 160. С. 36-54.

РОЗДІЛ 3
СИГНАТУРНО-КУБІТНІ МЕТОДИ РОЗПІЗНАВАННЯ ДЕСТРУКТИВНИХ КОДІВ

Пропонуються унітарно кодовані кубітно-матричні моделі, структури даних, обчислювальні архітектури і методи паралельного логічного аналізу деструктивних кодів в кіберфізичному просторі. Вводяться кубітні векторні структури даних для опису параметрів змінних, що беруть участь у формуванні еталонних зразків (патернів) деструктивних вихідних кодів. Пропонується паралельний сигнатурно-кубітний метод моделювання malware даних для визначення їх приналежності до існуючих деструктивних компонентів malware library. Пропонується сигнатурно-кубітний метод синтезу еталонних логічних схем malware-функціональностей, який відрізняється від аналогів унітарним кодуванням сигнатур для кодів деструктивних компонентів і формуванням кубітних матриць. Вводиться сигнатурно-кубітний процесор активного online кіберфізичного cyber security комп'ютингу (CSC) на основі моніторингу вхідних malware-даних і їх моделювання на еталонних логічних схемах malware-функціональностей з метою подальшого актюаторного управління процесом видалення деструктивних компонентів.

3.1. Вступ. Визначення та правила CSC-комп'ютингу

Розвиток кіберпростору перетворює фізичний світ з домінуючого у підлеглий. Реальний світ все більш залежнить від віртуального, бо керується ним. Всі фізичні процеси і явища мають власні цифрові образи, які поступово трансформуються в прообрази. Кіберфізичний світ позитивно з'єднує все населення планети один з одним без посередників, завдяки соціальним мережам, хмарним сервісам і Edge Computing.

Тема Cyber Security Computing інтегрально має достатній індекс в бібліотеках IEEE Xplore – 10980 і Springer – 23345. Проте існують поодинокі

публікації в частині активного комп'ютингу, спрямованого на автоматичне виявлення і усунення malware без участі людини [1, 2]. Природно, що з'єднання двох ринково-орієнтованих наукових напрямків (security and computing) може дати істотний практичний результат для підвищення якості сервісів, життя і збереження екології планети. Згадані джерела побічно зачіпають питання активного кіберфізичного комп'ютингу, пов'язаного з актюаторним управлінням кібербезпеки. Позбавлення людини функції управління безпекою та передача її кіберфізичному CSC-комп'ютингу є найголовнішою організаційною проблемою креативного світу. Тому громадянин, соціальна група, компанія, держава і людство потребують створення масштабованого аватара у форматі Gartner-computing: «security assistent – digital twin – smart security robot», який позбавить людей невірних рішень, що призводять до небажаних наслідків на ринку кіберфізичних і соціальних технологій.

Комп'ютинг – галузь знань, яка займається розвитком теорії і практики надійного метричного управління віртуальними, фізичними (природними) і соціальними процесами і явищами на основі використання комп'ютерних дата центрів і мереж, великих даних і цифрового моніторингу кіберфізіческого простору за допомогою інтелектуальних пошуково-аналітичних сервісів, персональних гаджетів і розумних датчиків.

Комп'ютинг (рис. 3.1) – процес моніторингу (5) і актюації (6) метричних відношень (2) в інфраструктурі управління (3) і виконання (4) для досягнення і візуалізації (8) мети – продукції (1) при заданих ресурсах (7).

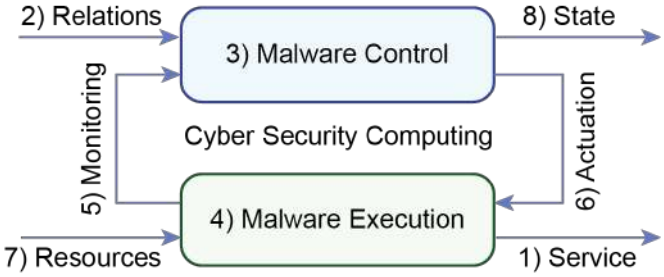


Рисунок 3.1 – Комп'ютинг

Cyber Security Computing (CSC) – галузь знань, що займається розвитком теорії і практики надійного метричного online управління кіберзахисту великих даних, віртуальних, фізичних (природних) і соціальних процесів і явищ в комп'ютерних дата центрах і мережах на основі точного цифрового моніторингу деструктивних компонентів кіберфізичного простору за допомогою інтелектуальних пошуково-аналітичних сервісів і розумних сенсорів.

Метричне визначення комп'ютингу за допомогою 8 взаємопов'язаних компонентів надає теоретичну фундаментальну основу для формального і фактичного створення будь-якого процесу в заданій сфері людської або природної діяльності. Види комп'ютингу за введеною метрикою охоплюють всі сфери людської діяльності: космологічний, біологічний, флористичний, фізичний, віртуальний, кібербезпечний, квантовий, соціальний, державний, медичний, транспортний, інфраструктурний, науковий, освітній, виробничий, спортивний, відпочинку, подорожей, розваг.

Процес – матеріально-енергетична взаємодія системних компонентів в часі і просторі для досягнення мети. Глобально: процес – матеріально-енергетична зміна в просторово-часовому континуумі. Локально: процес є розвиток просторового відношення компонентів (явищ) в часі.

Явище – компонент (системи) або фрагмент процесу в фіксований момент часу, що сприймається рецепторами, почуттями, вірою або розумом.

Cyber Security Computing: процес – спостережувана взаємодія механізмів управління та виконання в часі і просторі на основі моніторингу та актюації метричних відносин між програмними додатками і деструктивним кодом для досягнення мети у вигляді тестування, діагностування та усунення різного виду malware.

Відношення – структура взаємопов'язаних компонентів, що визначає властивості процесу або явища. Структура визначає властивості компонентів, але ніяк не навпаки. Первинно відношення-сигнатура, вдруге носії-компоненти. Алфавіт є носієм відношення, що визначається за допомогою

операцій (сигнатури) над символами. Просто символи алфавіту не мають сенсу. Відношення є визначальними при створенні ефективних: математичних теорій, структур даних, алгоритмів, архітектур, моделей, методів, технологій, програмно-апаратних додатків, кіберфізичних і соціальних систем, включаючи економіку, охорону здоров'я, транспорт, юриспруденцію, охорону правопорядку, кібербезпеку, екологію і державність. Потужність відношення, як інтегральна сукупність і якість взаємних зв'язків між компонентами, формує метрику, яка дає можливість ідентифікувати ефективність структури.

Визначення комп'ютингу, корисні для розуміння і практичного використання: 1) Комп'ютинг є процес цілеспрямованого розвитку компонентів, що беруть участь в ньому. 2) Все є комп'ютинг і нічого крім нього. 3) Найпростіші види комп'ютингу, доступні для розуміння і реалізації: reading-writing, speaking-listening, monitoring-control. 4) Всі процеси в природі детерміновані і цілеспрямовані. 5) Первинним є відношення, яке породжує елементи, що беруть участь в ньому. Елементів без відношення не існує. Жоден з похідних компонентів процесу не може існувати самостійно. 6) Первинним є процес, який породжує явища або компоненти, що взаємодіють у ньому. 7) Явища курки і яйця є похідними від комп'ютингу або еволюційного процесу. 8) Еволюція за Дарвіном є комп'ютинг природних явищ в часі і просторі. 9) Соціальний комп'ютинг є процес розвитку суспільних відносин між політичною елітою і громадянами в часі і просторі для досягнення поставлених цілей.

Метрика відношень. Елементарна основа світобудови є відношення між двома компонентами: процесами або явищами. Як правило, в процесі – це відношення нерівності пари компонентів, яке вимірюється ставленням рівності (xor, not-xor), що становить сутність метрики.

Не існує компонента без відношення, оскільки дана процедура є відношенням між двома компонентами. Це вірно і для випадку, коли сам

компонент знаходиться в відношенні рефлексивності з собою. Тому елемент визначається і розглядається як частина відношення.

Метрика первинності відносини нерівних компонентів поширюється глобально на всі явища і процеси, пояснює їх і породжує їх: 1) Одиниця – Нуль. 2) Чорне – Біле. 3) Багатий – Бідний. 4) Добро і Зло. 5) Розумний – Дурень. 6) Студент – Учитель. 7) Керівник – Виконавець. 8) Елемент – Множина. 9) Чоловік – Жінка. 10) Моніторинг – Управління. 11) Курка – Яйце. 12) Простір – Час. 13) Матерія – Енергія. 14) Reading – Writing. 15) Listening – Speaking. 16) Процес – Явище. 17) Хаос і Порядок. 18) Еліта і народ. 19) Живе – Неживе 20) Software – Malware.

Взаємодія протилежних асиметричних явищ у часі створює стійку структуру або процес еволюції. Взаємодія синонімічних явищ в часі створює нестійку структуру або процес деградації. Похідна по антонімічним або протилежним явищ або процесів дорівнює їх об'єднанню. Похідна за всіма 20 згаданим парам відношень дорівнює їх об'єднанню. Симетрія або еквівалентність компонентів відношення не здатна до еволюції. Тому відношення рівності компонентів в системі означає кінець розвитку. Критерієм даного факту є нульове значення похідної між взаємодіючими компонентами системи. Компоненти відношень в процесі еволюції системи перетворюються один в одного. Відношення породжує елементи, але не навпаки. Мета процесу – еволюційний перехід з одного явища в інше.

CSC – процес тестування, моніторингу, діагностування та активації сигналів деструкції шкідливих компонентів на основі метричних відносин між malware і software в кіберфізичному просторі.

CSC-процес – спостережувана взаємодія механізмів malware і software в часі і просторі на основі моніторингу та актюації метричних відносин для досягнення мети у вигляді усунення malware при виділених ресурсах.

Malware-функціональність (MF) являє собою структуру взаємопов'язаних логічних елементів, яка забезпечує цифрову реалізацію деструктивної поведінки об'єкта в заданому просторі software змінних.

Malware-змінна (MV) визначається упорядкованим універсумом примітивних значень, який формує проекцію поведінки об'єкта на векторі змінних, що створює malware-функціональність.

Логічний malware-елемент (ML) являє собою еталонне відображення значень багатозначної змінної в двійковий кубітний вектор, заданий на упорядкованому універсумі примітивних значень.

Значення (LV) змінної – унікальна примітивна властивість об'єкта, що має пустий перетин з іншими примітивами, яке в суперпозиції з ними складає універсум.

Таким чином, проглядається структурована ієрархія введених понять, рис. 3.2:

(CSC – MF – MV – ML – LV),

яка формує можливі архітектурні рішення malware-комп'ютингу.

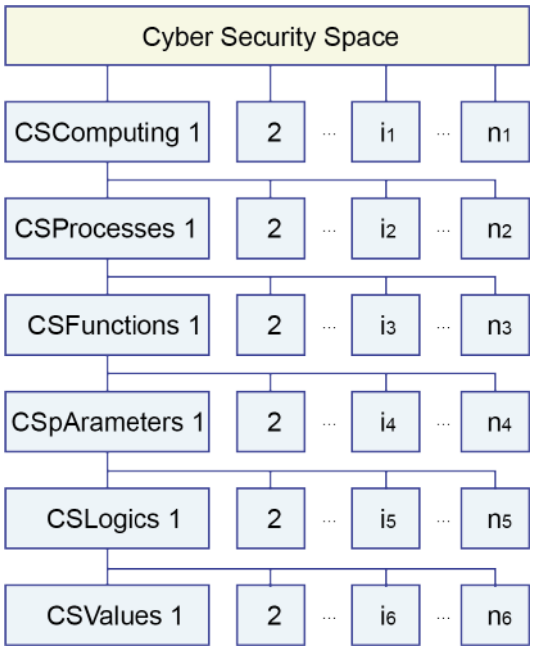


Рисунок 3.2 – Ієрархія malware-комп'ютингу

ML-рівень архітектури характеризується синтезом логічної схеми, де кожен елемент має одну багатозначну регістрову змінну, яка фактично представлена кубітним вектором, де число одиничних координат може бути більше одиниці. Дана властивість кубіта дає можливість створювати

компактні структури даних malware-функціональностей з метою їх паралельної обробки. Для виконання квантового методу моделювання на кубітних структурах даних необхідно унітарно закодувати вхідні символьні дані за допомогою таблиць-універсумів значень, що відповідають кожної змінної.

Змінна ототожнюється з ключовим словом-поняттям (keywords), яке найбільш часто зустрічається в потоці вхідних даних. Набір таких keywords створює непересічну множину змінних в malware-процесі, де їх значення представлені синонімами ключових слів, які формують багатозначність змінної, як клас еквівалентності.

Сукупність змінних створює простір malware-процесу, в якому визначаються еталонні, практично орієнтовані функціональності malware-комп'ютингу у вигляді логічних кубітних схем для моделювання вхідних потоків даних, взятих із хмари, мережі, комп'ютерів або гаджетів.

Мета – створення логічного кубітного процесора для паралельного моделювання та розпізнавання malware-функціональностей у вхідних потоках великих даних, отриманих шляхом метричного моніторингу інфраструктурних компонентів, для подальшого цифрового управління актюаторними сигналами по деструкції шкідливих компонентів.

Завдання пов'язані зі створенням моделі, методу і процесора malware-комп'ютингу, спрямованого на автоматичний синтез і аналіз кубітних логічних схем, орієнтованих на моніторинг, моделювання, розпізнавання і деструкцію шкідливих кодів.

1) Процесор malware-комп'ютингу на основі синтезу кубітних логічних схем для моніторингу, моделювання, розпізнавання і деструкції шкідливих кодів.

2) Кубітно-векторна модель опису універсуму значень багатозначної змінної для синтезу логічного секвенсора, орієнтованого на аналіз malware-компонентів.

3) Кубітний метод синтезу логічної схеми для моделювання та розпізнавання malware-функціональностей на основі унітарного кодування значень багатозначної змінної.

4) Кубітний метод аналізу malware-компонентів на основі використання еталонних логічних елементів malware-функціональностей з унітарним кодуванням багатозначних змінних.

5) Тестування і верифікація кубітних моделей і методів malware-комп'ютингу на прикладах аналізу і розпізнавання malware-процесів у вхідних потоках великих даних, пов'язаних з кіберфізичним відображенням обчислювальних процесів.

3.2. State of the art. Квантові процеси і явища

Gartner тенденції світової кіберкультури [3-14] (рис. 3.3) формують технологічну культуру для створення глобального кіберфізичного CSC-комп'ютингу в рамках укладу Internet of Things.

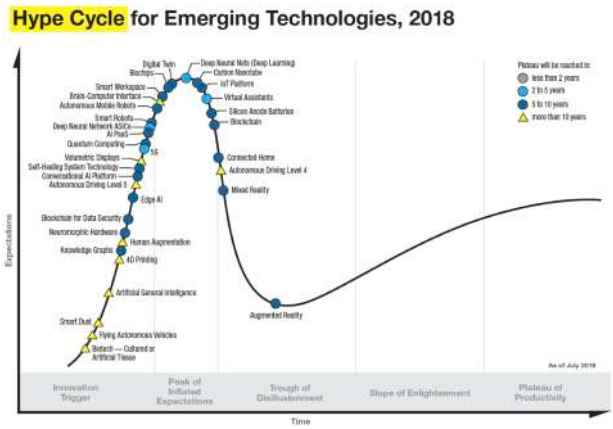


Рисунок 3.3 – Gartner топ-технології 2018

При цьому квантовий комьпютинг розглядається як енергозберігаюче майбутнє цифрового світу, створюваного для підвищення якості життя і збереження екології планети. Зокрема, квантовий паралельний комп'ютинг і кубітні структури даних дозволяють спростити алгоритми в області cyber

security computing і підвищити швидкодію програмних продуктів на класичних комп'ютерах.

Стратегічні тенденції в області цифрових технологій [3 – 7] приведуть у 2020 році до суттєвих дізрапцій, що надасть нові можливості розробникам корпоративної архітектури і конструктивних інновацій з метою створення конкурентних переваг при використанні нових трендів кіберкультури: 1) Автономні фізичні та віртуальні інтелектуальні та координовані речі. 2) Розширена, доповнена кіберпростором, соціальна CSC-аналітика для вироблення актюаторних впливів. 3) AI-кероване проектування, розширений і автоматичний розробник. 4) Цифрові близнюки; цифровий образ організації або компанії. 5) Взаємодоповнюючі один одного Edge + Cloud Computing. 6) Досвід занурення в цифрову дійсність і своєчасне сприйняття деструктивних змін в цифровому світі. 7) Використання Blockchain в соціальній схемі. 8) Smart Spaces. 9) Цифрова етика і конфіденційність особистого життя. 10) Квантові обчислення, квантова кібербезпека, розвиток і становлення квантового комп'ютингу.

Визначення «квантовий» характерно не тільки для субатомних структур, взаємодіючих за моделлю (комплексних) гільбертового простору квантової механіки. Дане визначення у теперішній час носить глобальний характер і імплементується в усі сфери науки, освіти і людської діяльності.

Вчені і практики планети створюють сьогодні новий квантовий технологічний уклад, який формується на основі класичних наук, що представлені квантовою електродинамікою, механікою, фізикою та математикою, і на якому успішно приростають і розвиваються нові дисципліни і галузі знань: квантовий комп'ютинг, квантові структури даних, квантові схеми, квантові алгоритми, квантова логіка, квантове моделювання схем, квантові групи, квантове моделювання (quantum computer modeling – IEEE Xplore 3436), квантові розподілені системи, квантова інформация, квантові телекомунікації, квантові системи, квантові точки, квантова метрологія, квантова радіотехніка, квантові комп'ютерні науки, квантова

криптографія і кібербезпека, квантова телепортація, квантова компресія даних, квантова пам'ять, квантова фотоніка,

Quantum computer modeling & quantum-driven algorithm everywhere, як інженерні рішення, використовує технологічну ідею квантового змішування і суперпозиції для синтезу продуктивних паралельних алгоритмів memory-driven комп'ютингу в усіх сферах людської діяльності.

Ефективність квантових обчислень на класичному комп'ютері визначається метрикою паралельного рішення комбінаторних задач на кубітних структурах даних за рахунок ускладнення алгоритмів і збільшення пам'яті для зберігання унітарних кодів.

Квантовий комп'ютинг оперує двома базовими технологічними операціями: суперпозиція і переплутування, яким ставляться у відповідність логічні операції функціонально повного базису: диз'юнкція і інверсія в класичному обчислювачі.

Основа квантової технології визначається квантовою невизначеністю електронів, суперпозиція яких дає можливість отримати в одній точці простору будь-який універсум кінцевого числа примітивних сигналів, включаючи нуль і одиницю. Ізоморфним аналогом виступає дискретна невизначеність Кантора X={0,1}, яка використовується для отримання компактних форм структур даних, а також для паралельного виконання теоретико-множинних операцій при унітарному кодуванні елементів універсуму.

Приклад quantum-driven алгоритму: знайти в множині елементів Q число X або визначити належність X∈Q. Рішення зводиться до перебору порівнянь числа X з усіма числами масиву Q, кількість яких дорівнює n. Обчислювальна складність такої процедури дорівнює C=n. Квантове рішення даної задачі зводиться до унітарного кодування всіх чисел з метою синтезу структури даних в формі одного вектора в форматі універсуму-множини чисел, де кожному з них відповідає середнє арифметичне значення координати, адреса якої дорівнює числу, присутньому в множині. Після

цього виконується операція: if Q(X)=1 then Y=1, або простіше Y=Q(X), де і є число, яке підлягає визначенню приналежності X∈Q. Значення Y=1 є індикатором належності числа до заданої множини: X∈Q. Обчислювальна складність процедури пошуку числа в множині пов'язана з побудовою кубітного вектора, на що витрачається n операцій, і з виконанням логічної перевірки вмісту комірки за адресою-числом, що надійшов на вхід квантового секвенсора: C=n+1. Дійсно, разове використання такого обчислювача не дає збільшення швидкодії. Однак багато разів (m) повторена процедура пошуку числа в неврегульованому масиві при великих значеннях m, призводить до одиниці обчислювальну складність алгоритму, включаючи первинне ініціювання вектора унітарних кодів чисел. Даний факт дає підстави вважати обчислювальну складність процедури по багаторазовому пошуку числа в масиві, рівну одиниці: C=(n+m)/m.

Приклад визначення приналежності деякої підмножини елементів X множини Q. Рішення: після унітарного кодування елементів обох множин виходять два вектора однакової довжини, до них застосовуються лише три векторно-регістрових паралельних операції (not, and, or):

$$Y = \vee[(X \wedge Q) \oplus X] = \vee[(X \wedge \neg Q)],$$

які разом з препроцесінгом унітарного кодування мають обчислювальну складність C=2n+3. Класичний варіант вирішення даної задачі має обчислювальну складність, яка дорівнює

$$C = (1/2) * (n-1) ** 2.$$

Вже при n=7 квантовий або кубітний метод починає вигравати в продуктивності рішення даного завдання перед класичним.

3.3. Кубітні моделі кіберфізичного соціального CSC-комп'ютингу

Пропонуються архітектури і класичні структури, пов'язані з кіберфізичним комп'ютингом (метричний моніторинг і цифрове управління), спрямованим на прийняття рішень, пошук і ідентифікацію malware, визначення функцій належності вхідних даних до заданого деструктивного

зразку на основі введеної метрики визначення відстаней. Всі моделі орієнтовані на схемотехнічну реалізацію методів і алгоритмів online моделювання з метою вироблення human-free адекватних автоматичних актюаторних впливів, спрямованих на знищення деструктивних кодів.

Логічні кубітні структури здатні розпізнавати деструктивні коди, надходять на вхід спеціалізованого комп'ютера, і усувати їх з програмних додатків.

Кожен malware-параметр може мати свою альтернативу значень в позитиві та негативі (мультиверсність також допускається), тоді їх число подвоюється. Але можна використовувати тільки позитивні зразки, засновані на конструктивних параметрах або атрибутах. Такі образи – логічні процесори – формують еталонні якості malware-процесів і явищ.

Квантові технології паралельних обчислень ефективно використовуються для вирішення комбінаторних проблем, емулюючи обчислення на класичних комп'ютерах [14-16]. З іншого боку, таблиці істинності або кубічні покриття для опису логічних елементів є ефективними структурами даних для вирішення проблем CSC-комп'ютингу та пошуку необхідних даних [17, 18]. Автоматичний синтез кубітних покриттів функціональностей є одним з основних важко формалізованих завдань, без якої неможливо виконувати аналітику великих даних [19-22]. Для цих цілей далі вводиться аналітична модель W кубітно-логічного процесора CSC-комп'ютингу, яка оперує двома матрицями: універсумів U примітивів і кубітних функціональностей Q, а також логічними примітивами L, що интегрують функціональності в комбінаційну схему CSC-процесора:

$$W = (U,Q,L),$$
$$U = (U_1, U_2, ..., U_i, ..., U_n);$$
$$\bigcup_{i-1}^{n} U_i = U;\ U_i \bigcap_{i,k=\overline{1,n}}^{i \neq k} U_k = \varnothing;$$
$$Q = (Q_1, Q_2, ..., Q_i, ..., Q_n);$$
$$\bigcup_{i-1}^{n} Q_i = Q;\ Q_i \bigcap_{i,k=\overline{1,n}}^{i \neq k} Q_k = \varnothing;$$
$$Q_i = (Q_{i1}, Q_{i2}, ..., Q_{ij}, ..., Q_{im});\ Q = [Q_{ij}];$$
$$U_i = (U_{i1}, U_{i2}, ..., U_{ij}, ..., U_{im});\ U = [U_{ij}];$$
$$L = f[Q] = (Q_1 \circ Q_2 \circ, ..., \circ Q_i \circ, ..., \circ Q_n)$$
$$\circ = \{\wedge, \vee, \oplus\};$$
$$U_{ij} \in U_i \in U;\ Q_{ij} \in Q_i \in Q;\ Q_i \in U_i;\ Q \in U;$$
$$Q_{ij} = 1 \leftarrow \max \mu(R, U_{ij}).$$

Метрика-універсум U тут виконує роль еталонного зразка для порівняльного аналізу еталона з вхідним потоком деструктивних кодів і даних R = X, що реалізується за допомогою аналізатора-компаратора, що видає максимальне значення функції приналежності, яке трансформується в одиницю на відповідній координаті одного з кубітів $Q_{ij} = 1 \leftarrow \max \mu(R, U_{ij})$.

Архітектура метричної взаємодії U-матриці універсумів з потоком даних R для обчислення функцій належності μ(R, U), з метою отримання Q-матриці значень і подальшого L-об'єднання кубітів в комбінаційну схему кіберсоціального процесора, представлена на рис. 3.5.

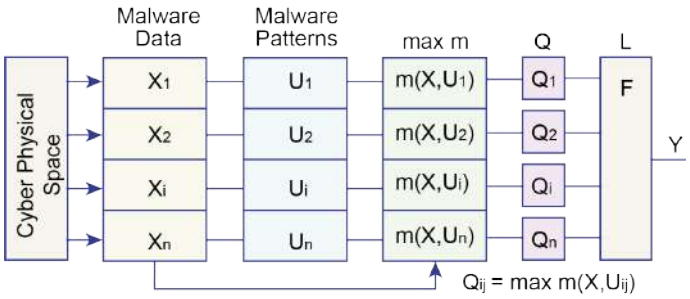


Рисунок 3.5 – Архітектура для синтезу CSC-процесора

Тут вхідний потік модельованих великих деструктивних даних R має такий же формат, як U-, Q-матриці і комбінаційна схема процесора. Алгоритм синтезу Q-матриці полягає у визначенні максимального значення функції приналежності вхідного фрейму розглянутої змінної до одного з

значень відповідного рядка матриці універсумів. В результаті такого порівняння по всіх координатах U-матриці формуються поодинокі координати кубітної матриці, де кожен рядок являє собою примітивну функціональність по розглянутій змінної. Разом всі рядки Q-матриці створюють комбінаційну схему логічного CSC-процесора для моделювання будь-якого вхідного впливу з метою визначення його приналежності до даного еталону деструктивного процесу або явища.

Процедура синтезу кубітів необхідна для аналізу malware-потоків даних шляхом використання логічних CSC-еталонів. Фрагменти даних надходять на входи одного або декількох логічних елементів, що формують метрику CSC-процесу:

$$U_{ij} \in U_i \in U;\ Q_{ij} \in Q_i \in Q;\ Q_i \in U_i;\ Q \in U;$$

В результаті моделювання вхідного потоку деструктивних даних формуються бінарні значення переваг еталона в кубітному векторі кожного логічного елемента, відповідного одному параметру. Для цього використовується метричний вимір функції приналежності вербальних даних до кожного значення наперед заданого універсуму примітивів логічної змінної. Так автоматично створюються кубітні зразки CSC-функціональності.

Формування повної множини параметрів-змінних CSC-процесу або явища також пов'язано з аналітикою великих даних, спрямованою на отримання ключових понять-слів, максимально віддалених один від одного по метриці (кодової відстані) класів еквівалентності, які покривають всі CSC-змінні. Слід зауважити, що універсум примітивів ототожнюється з класом непересічних еквівалентностей, що створюють всі можливі значення даної змінної-класу в той час, як множина еквівалентних класів відповідає універсуму змінних вищого рівня ієрархії. Дані властивості використовуються при аналітичному синтезі універсуму змінних, що покривають цифровий образ CSC-процесу гранями, які формують еталон-функціональності процесу або явища.

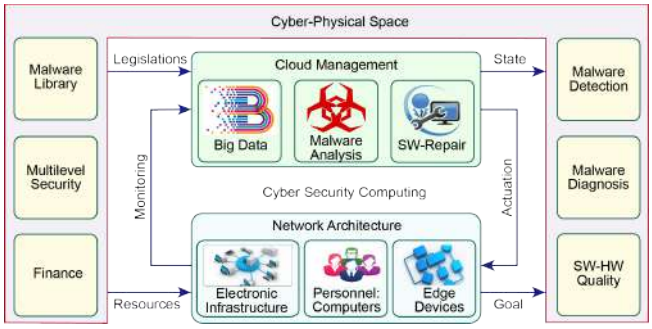
Наприклад, необхідно синтезувати віртуального CSC-асистента (virtual assistent) або цифрового двійника (digital twin), або розумного робота (smart robot) який буде реагувати на зовнішні вхідні дані. Алгоритм для створення аватара містить наступні кроки: 1) Синтез універсуму змінних-примітивів, що покривають все функціональності CSC-процесу. 2) Синтез U-матриці універсумів значень-примітивів, що покривають всі можливі варіанти кожної змінної, в рамках CSC-процесу або явища. 3) Синтез Q-матриці конкретних значень-примітивів в форматі кубіт-вектора кожної змінної в рамках CSC-процесу або явища. 4) Перевірка отриманої U-матриці універсумів CSC-процесу або явища на повноту і примітивізм змінних і значень.

Після синтезу кубітних векторів за всіма параметрами в Q-матриці всі значення виходів кубітних елементів надходять на входи інтегратора L, що працює по функції and (може бути і інша функція, наприклад, not-and), який видає два значення {1,0}: позитивний або негативний результат моделювання, який можна інтерпретувати як бінарну функцію приналежності до ідеалу malware-функціональності, формує, наприклад, властивості CSC-процесу.

Таким чином, logic-процесор, синтезований на основі використання Q-матриць квантових структур даних, здатний online моделювати будь-які CSC-процеси і явища, недоступні сьогодні для класичного комп'ютингу в базисі традиційних логічних елементів, зважаючи на складність формалізації поведінки malware для синтезу цифрових моделей-схем.

Формалізм створення еталон-схеми для CSC-процесу або явища полягає у визначенні числа істотних параметрів, де всередині кожного з них генерується сукупність значень, унітарно кодованих для синтезу кубітного вектора логічного елемента. Логічні примітиви, що відповідають істотним параметрам, об'єднуються за функціями (and, or, not, xor), які регулюють взаємні відносини між параметрами для формування кінцевого результату про валідність вхідного процесу або явища по відношенню до одного або декількох стандартів.

CSC-комп'ютинг може бути представлений як кіберфізична система інтелектуального хмарного управління CSC-процесами на основі точного цифрового моніторингу: розумної електронної інфраструктури; співробітників компанії, оснащених комп'ютерами та персональними гаджетами; транзакцій і процесів, заданих в часі і просторі. Структура системи CSC-комп'ютингу представлена трьома взаємодіючими макрокомпонентами: хмарне інтелектуальне управління, електронна CSC-архітектура, кіберфізичний простір (рис. 3.6).



Рисунко 3.6 – CSC-комп'ютинг моніторингу та управління процесами

Хмарні компоненти-сервіси моніторингу та усунення malware працюють за схемою: факт – оцінка – дія. Тут виконується знімання великих даних з різних розумних сенсорів і комп'ютерів, інтелектуальний аналіз даних на основі CNN, DNN, ML. Останнім компонентом хмарного сервісу є формування цифрових актюаторних впливів, орієнтованих на управління інфраструктурними компонентами, для досягнення мети (Goal) у вигляді видалення malware і забезпечення високої якості кіберпослуг. Вся система CSC-комп'ютингу безпосередньо взаємодіє з кіберпростором або інтернетом, який обов'язково є входом і виходом для створюваної структури. Крім того, входами є Legislations, які формують відносини в компанії, а також Resourses у вигляді фінансів і матеріалів, необхідних для захисту процесів створення продукції і/або сервісів. Важливим виходом системи є State, який ідентифікує стан розвитку комп'ютерної системи.

CSC-комп'ютинг є технологією ефективного хмарного управління для істотного зниження накладних непродуктивних витрат і підвищення прибутку, яка характеризується оперативним online моніторингом процесів на основі використання сучасної кібер-культури, що включає: IoT, Cyber Physical Systems, Cloud Computing, e-Infrastructure, Big Data Analytics , Artificial Intelligence, Blockchain smart contract, e-Dicument Circulation and Internet.

Принципи реалізації: 1) Моніторинг програмних додатків за допомогою впровадженого агента в умовах інваріантності геопозіції. 2) Моніторинг всіх пристроїв для інтелектуального аналізу і подальшого управління структурними компонентами, дає можливість оперативно приймати рішення по реконфігурації CSC-процесів в реальному часі; 3) Моніторинг, замкнутий на online управління, без активної участі людини. У цьому сьогодні головне і ще не вирішене завдання CSC-ринку.

Моніторинг malware без актюаторних впливів на усунення деструктивних компонентів, що виробляються кіберфізичною human-free CSC-системою, не представляє ринкового інтересу з позиції сучасної кіберкультури. Рішення проблеми цілком очевидне – створення CSC-системи моніторингу, але головне – online управління CSC-процесами на основі створення розумних алгоритмів або смарт-контрактів, що програмують легітимні відносини в кіберпросторі. Програмний код реалізує тріаду соціальних подій, без участі чиновника: факт – оцінка – дія, яка модельно зводиться до кодування алгоритму обробки вхідних даних для отримання вихідних актюаторних впливів, спрямованих на компоненти CSC-інфраструктури, яка виконується в рамках технологічного укладу IoT. Компонентами CSC-системи є: 1) Відносини, прийняті в компанії (державі) на основі існуючого законодавства, статуту (конституції), наказів, традицій, історії, культури. 2) Мета та/або напрямок руху, зрозумілі для ринку, що мобілізують співробітників для якісного виконання завдань. 3) Цифровий менеджмент або управління – секретний ключ ринкового успіху, - обов'язково використовує хмарні сервіси, максимально виключають участь

людини в моніторингу виробничих процесів і подальшому прийнятті рішень. 4) Інфраструктура підприємства, що забезпечує комфортні умови для конструктивної роботи, якісного харчування та активного відпочинку в форматі 24/7: onsite & remote online. 5) Кадри, що створюють ринкову продукцію та послуги, - головне надбання або інтелект будь-якої компанії, який оцінюється симетричної різницею компетенцій співробітників [17]:

$$I = \bigoplus_{i=1}^{n} P_i = \bigcup_{i=1}^{n} P_i \leftarrow P_i \cap P_j = \varnothing;$$
$$I = \bigwedge_{i=1}^{n} P_i = \bigcap_{i=1}^{n} P_i \leftarrow P_i = P_j.$$
$$\forall i,j$$

3.4. Унітарно-кодовані структури даних. Сигнатурний аналіз malware

Інтерес становить проблема аналізу великих даних з метою встановлення нових malware-функціональностей на природному тлі вже визначених. Аналогічна задача була вирішена в Лабораторії Касперського (ЛК) і вирішується досі засобами роботів і експертів в області malware and virus аналітики. Тут використовується сигнатурний аналіз, адаптований до вірусної аналітиці, який дозволяє мати досить компактний код-сигнатуру деструктивності з метою високопродуктивної ідентифікації в потоці великих даних старих вірусів. Це дає можливість акцентувати увагу робота-експерта на детальному аналізі нових деструктивів з метою їх подальшого блокування. Перекладаючи згадану сигнатурну технологію на рішення проблем malware-аналітики, слід зазначити важливість отримання компактної таблиці malware-функціональності, інваріантної до часу. Першим кроком в цьому напрямку є мінімізація таблиці унітарного кодування malware-функціональності по дозволеним логічним правилам (суперпозиція), які дають можливість отримати один стовпець, що ідентифікує malware-функціональність. Структурні протиріччя при об'єднанні координат стовпців унітарно-кодованої матриці відсутні. Природно, що в результуючому стовпці-сигнатурі буде втрачена структурна інформація про порядок виконання

сервісу, що є платою за компактність і швидкодію по ідентифікації стовпців malware-функціональності. Однак структурна інформація не стирається і може бути затребувана в разі необхідності. Теоретичним підтвердженням і обгрунтуванням запропонованої суперпозіціонної інновації зі стиснення стовпців в один є той факт, що кодування будь-якої таблиці істинності двома і навіть одним кубітним вектором, отриманим за допомогою суперпозиції унітарних кодів вхідних впливів будь-якого, як завгодно складного цифрового пристрою (рис. 3.7). Обмеження: всі атрибути в матриці унітарного кодування, що підлягають суперпозиції по конкретних даних, повинні бути незалежними один від одного. Суперпозиція стовпців унітарної матриці дає можливість отримати покриття всіх атрибутів-змінних одиничними значеннями різноманітності даних. Якщо одиницями покриті в повному обсязі значення атрибутів, то існує некоректність в аналізі та кодування даних по конкретному атрибуту. В масштабах метрики значень интегральний стовпчик **malware**-функціональності завжди буде уявляти собою підмножину з нульових та одиничних координат на фоні повністю одиничних значень інтегрального стовпця універсуму P∈R if P∩R=P.

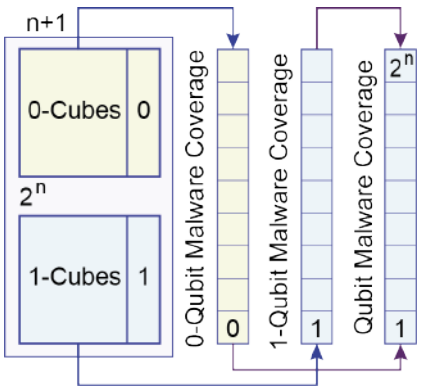


Рисунок 3.7 – Унітарне кодування універсуму примітивів по кожній змінній

Суперпозіційна модель уявлення malware-функціональності інваріантна до часу. Ідея класифікації полягає в порівнянні великих даних з

інтегральним вектором, який виходить шляхом суперпозиції або об'єднання всіх стовпців malware-функціональності

$$P = \bigcup_{i=1}^{n} P_i.$$

Процедура ідентифікації зводиться до операції перетину між стовпцем вхідних даних і інтегральним стовпцем malware-функціональності: $S \in P \leftrightarrow S \cap P = S,$ яка повинна дорівнювати вектору вхідних даних. Природно, виникнуть ситуації, коли не буде виконуватися наведена вище умова за всіма порівняннями за допомогою стовпців malware-функціональності. Тоді слід керуватися правилом домінування мінімальної кодової відстані за Хемінгом:

$$S_i \in P_j \leftrightarrow \min(S_i \bigcap_{j=1}^{m} P_j = \varnothing), i = \overline{1,n};$$

$$S_i \in P_j \leftrightarrow \min(S_i \bigwedge_{j=1}^{m} P_j = 1), i = \overline{1,n}.$$

Для аналізу детермінованої двійкової моделі malware-функціональності існує ефективний апарат булевих похідних, який визначає істотність і неістотність змінних щодо формування вихідного значення функціональності. Якщо зміна стану змінної-атрибута не призводить до зміни функціональності, то така змінна є несуттєвою і її можна виключити з моделі CSC-функціональності.

При вербальному завданні моделі malware-функціональності розробники використовують свій досвід і інтуїцію для формування екстра-функціональностей, дублюючих деякі істотні атрибути на моделі malware-функціональності. Дана, в межі 100%-ва, надмірність може бути використана також для асерціонной верифікації моделі CSC-процесу. Сенс такої верифікації полягає в незалежному створенні і подальшому використанні-порівнянні двох моделей, де перша – максимально точна за всіма параметрами, друга – створює картину станів істотних змінних. Асерції, що не несуть нової інформації про модель, але дають можливість уточнити наявність стану даних для істотних атрибутів біз malware-функціональності в конкретному часовому фреймі.

Модельна надмірність, як правило, є корисною для прискорення обчислювальних процесів за рахунок диверсифікації структур даних. Наприклад, просторово-часова модель malware-функціональності за рахунок конволюції часу в точку може бути компактно представлена одним інтегральним стовпцем даних.

Багатозначність параметрів malware-функціональності укладається в наступну матричну модель:

$$P = [P_{ij}], i = \overline{1,n}; j = \overline{1,m};$$
$$P = (P_1, P_2, ..., P_i, ..., P_n);$$
$$P_i = (P_{i1}, P_{i2}, ..., P_{ij}, ..., P_{im}).$$

Тут n – число рядків матриці malware-функціональності, m – кількість значень параметра Pi при її кодуванні.

Для оптимізації malware-функціональності необхідно і достатньо використовувати відомі аксіоми алгебри логіки:

$$1)\ a \vee a = a \leftrightarrow 1 \vee 1 = 1;$$
$$2)\ a \vee ab = a(1 \vee b) = a \leftrightarrow 1x \vee 11 = 1;$$
$$3)\ ab \vee a\overline{b} = a(b \vee \overline{b}) = a \leftrightarrow 11 \vee 10 = 1x = 1;$$
$$4)\ abc \vee b = b.$$

Логічні аксіоми трансформуються в закони теорії множин, де фігурують елементи в форматі унітарних кодів значень вхідних змінних:

$$1)\ a \cup a = a;\ a \cap a = a;$$
$$2)\ a \cup ab = a(1 \cup b) = a \leftrightarrow 1x \cup 11 = 1x = 1;$$
$$3)\ ab \cup a\overline{b} = a(b \cup \tilde{b}) = a \leftrightarrow 11 \cup 10 = 1x = 1;$$
$$4)\ abc \cap b = b \leftrightarrow 111 \cap 010 = 010.$$

Всі вербальні значення або частини істотних (додаткових) параметрів повинні бути унітарно і єдино-метрично закодовані з метою представлення координат матриці malware-функціональності двійковими векторами, які дають можливість в паралельному режимі визначати приналежність вхідного вектора одному або кількох стовпців malware-функціональності шляхом застосування логічної процедури :

$$a \in ab \leftrightarrow a \cap ab = a \rightarrow 10 \cap 11 = 10 \wedge 11 = 10.$$

У загальному випадку метрична взаємодія двох компонентів однієї розмірності може мати тільки п'ять випадків, рис. 3.8 [19]:

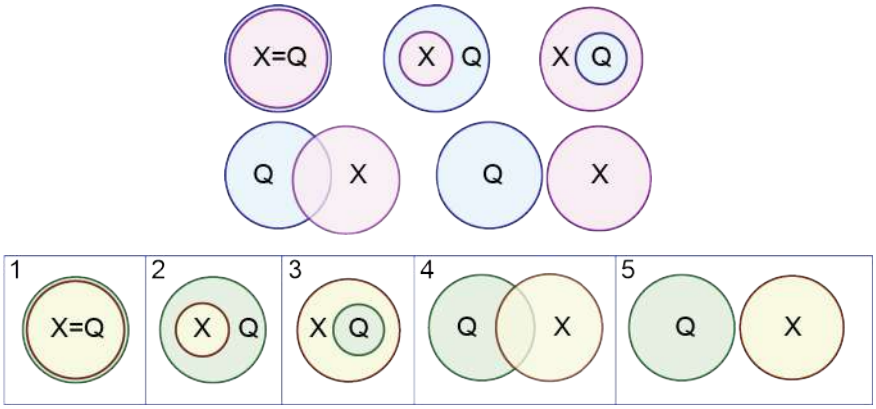


Рисунок 3.8 – Взаємодія даних по and-операції

1) Належність чи рівність об'єктів один одному, якщо виконується умова:

$$a = b \leftrightarrow a \cap b = \{a,b\} \rightarrow 10 \cap 10 = 10 \wedge 10 = 10.$$

2) Належність першого A другому m, якщо виконується умова:

$$a \in b \leftrightarrow a \cap b = a \rightarrow 10 \cap 11 = 10 \wedge 11 = 10.$$

3) Приналежність другого m першому A, якщо виконується умова:

$$b \in a \leftrightarrow a \cap b = b \rightarrow 11 \cap 10 = 11 \wedge 10 = 10.$$

4) Часткова приналежність об'єктів один одному, якщо виконується умова:

$$a \neq b \leftrightarrow a \cap b \neq \{\varnothing,a,b\} \rightarrow 110 \cap 011 = 110 \wedge 011 = 010.$$

5) Неналежність об'єктів один одному, якщо виконується умова:

$$a \neq b \leftrightarrow a \cap b = \varnothing \rightarrow 01 \cap 10 = 01 \wedge 10 = 00.$$

Структурна карта модулів комп'ютингу для аналізу CSC-процесів (рис. 3.9): синтез матриці істотних змінних; побудова унітарної матриці даних; декомпозиція унітарної матриці даних; синтез M-RPA (Malware Robotic Process Automation) на основі застосування ML-технології до матриць malware-функціональностей.

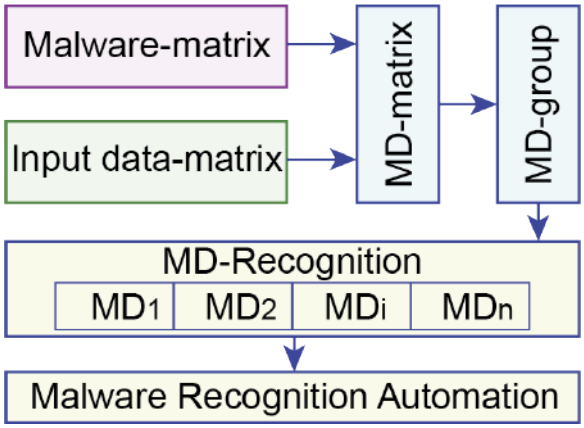


Рисунок 3.9 – Структура для аналізу malware-процесів

Визначення унітарної матриці істотних змінних malware-процесів і кодування всіх значень.

Рішення. Організовується цикл по n змінним без malware-функціональності, де всередині створюється цикл за значеннями змінних, де є ще один вкладений цикл, що перераховує всі існуючі malware-функціональності, які обробляються на предмет їх оригінальності (рис. 3.10). Таким чином, програмний модуль P-matrix, що містить три вкладених цикли, створює таблицю відповідності текстових значень істотних змінних їх десятковим номерам або унітарним кодам для подальшого аналізу CSC-процесів.

У загальному випадку метрична взаємодія двох стовпців-malware однієї розмірності може мати тільки п'ять випадків:

1) Належність чи рівність malware один одному, якщо виконується умова:

$$a = b \leftrightarrow a \cap b = \{a, b\} \rightarrow 10 \cap 10 = 10 \wedge 10 = 10.$$

2) Належність першого malware A другого m, якщо виконується умова:

$$a \in b \leftrightarrow a \cap b = a \rightarrow 10 \cap 11 = 10 \wedge 11 = 10.$$

3) Приналежність другого malware m першому A, якщо виконується умова:

$$b \in a \leftrightarrow a \cap b = b \rightarrow 11 \cap 10 = 11 \wedge 10 = 10.$$

4) Часткова приналежність malware один одному, якщо виконується умова:

$$a \neq b \leftrightarrow a \cap b \neq \{\varnothing, a, b\} \rightarrow 110 \cap 011 = 110 \wedge 011 = 010.$$

5) Неналежність malware один одному, якщо виконується умова:

$$a \neq b \leftrightarrow a \cap b = \varnothing \rightarrow 01 \cap 10 = 01 \wedge 10 = 00.$$

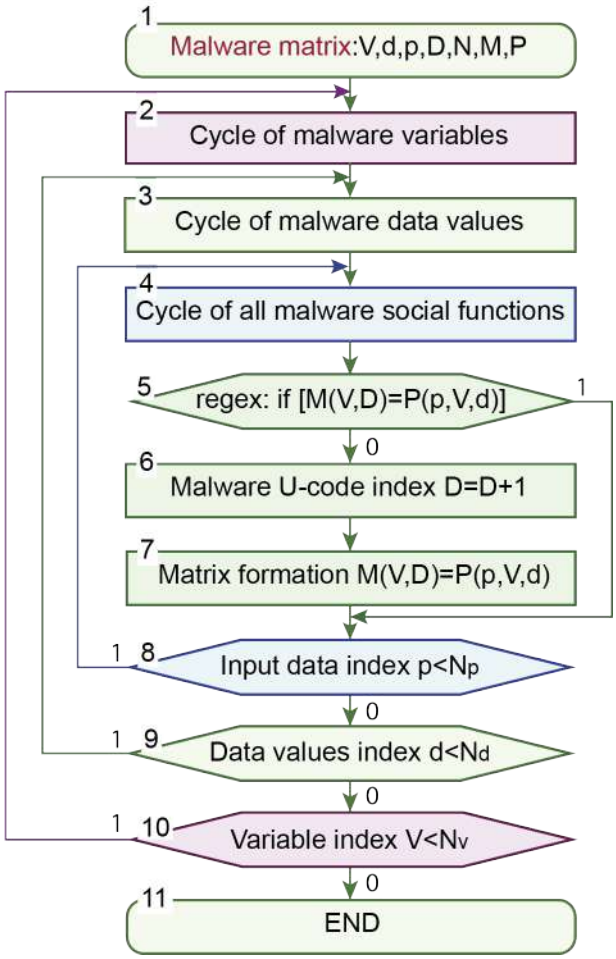


Рисунок 3.10 – Алгоритм формування матриці значень параметрів

3.5. Метод кубітно-сигнатурного аналізу malware

Розглядається паралельний метод кубітного моделювання malware великих даних, який характеризується використанням сигнатурного аналізу і кубітними структурами даних, що дає можливість в паралельному режимі визначати приналежність поточного коду до існуючих деструктивних компонентів в malware library.

Мета сигнатурного аналізу полягає в суттєвому зменшенні (від квадратичної до лінійної) обчислювальної складності процедури унітарного кодування текстових (мови опису malware: assembler, C++, Java) значень атрибутів при обробці великих даних вхідного потоку, що містить тисячі malware.

Сутність сигнатурного аналізу полягає в швидкому перетворенні значення текстової змінної в код-сигнатуру, чисельне значення якої є унікальним з ймовірністю 0,9998, що дає підстави ідентифікувати сигнатурою текстові фрагменти malware. Далі вирішуються завдання:

1) Визначення приналежності значення текстової (строкової) змінної поточного malware до одного з уже існуючих закодованих значень шляхом порівняння сигнатур, коли створення нового унітарного коду не потрібно.

2) Підтвердження оригінальності значення строкової змінної, коли встановлений факт, що в результаті виконаного перегляду її сигнатура не збіглася з уже існуючими, отже, необхідно створення нового унітарного коду.

3) Створення таблиці за форматом <текст-сигнатура>, яку можна розглядати як <текст-число>, дає можливість звести задачу унітарного кодування значень текстових змінних до обчислювальної складності 2n (замість 0,5n**2) і отримати таблицю <malware-сигнатура-код>. Тут синтез унітарних кодів пов'язаний з переглядом стовпчика всіх отриманих сигнатур, як десяткових чисел, представлених 16 двійковими розрядами, з метою заповнення одиничними значеннями координат апріорно нульового вектора

покриття, розмірністю 2**16, за адресами, який формується сигнатурами текстів. Потім рахункова процедура (if U(i)=1 then j=j+1 and A(j)=i) перегляду всіх координат вектора покриття на предмет пошуку одиничних значень сформує вже компактний вектор координат, який буде представляти не що інше, як упорядкований вектор адрес А для унітарного кодування значень malware змінних. Наприклад, є множина десяткових цифр {5,2,1,9,7,4}, які необхідно впорядкувати на основі застосування процедури лінійної обчислювальної складності. Завдання вирішується шляхом розгляду множини цифр, як адрес, з метою подальшого занесення одиничних значень в координати вектора покриття за адресами-елементам: U=(110110101). Обчислювальна складність даної процедури складає n елементів множини. Далі послідовно виконується читання значень координат, де в разі фіксації одиниці в поточну комірку вектора значень записується адреса одиничної координати: if U(i)=1 then j=j+1 and A(j)=i. Результат обробки вектора U по останній процедурі має наступний вигляд: A=(1,2,4,5,7,9). Обчислювальна складність процедури запису упорядкованих елементів практично дорівнює n елементів безлічі, якщо розкид значень чисел незначний, в межах одного порядку. Інтегрально дві процедури мають витрати, рівні Q=2n.

Слід зауважити, що двійковий вектор покриття може бути ефективно використаний для істотного підвищення продуктивності процедури кодування значень malware змінних.

Стан проблеми: для присвоєння коду-числа фрагменту тексту необхідно переконатися, що він є оригінальним у масштабах всього CSC-процесу. Це досягається шляхом порівняння фрагмента з усією бібліотекою вже закодованих текстів, що належать деякому атрибуту malware. Обчислювальна складність процедури тільки одного порівняння дорівнює Q=m*n, де m – довжина фрагмента, n – число рядків в бібліотеці закодованих malware.

Метод кубітно-сигнатурного аналізу malware на основі унітарного кодування пропонує: 1) Ідентифікувати текстові фрагменти malware кодами-

сигнатурами. 2) Занести одиничні значення в координати вектора покриття V, адреси яких дорівнюють сигнатурам оригінальних malware U(S). 3) При аналізі вхідного потоку великої розмірності оригінальність malware по одному атрибуту перевіряється шляхом формування сигнатури S з подальшою перевіркою в координаті U(S)=1 значення одиниці. Якщо одиниця є, то текстовий фрагмент не є новим і виконується перехід до наступного malware. В іншому випадку, якщо U(S)=0, фрагменту присвоюється унітарний код-ідентифікатор, а сигнатура виступає адресою для занесення значення U(S)=1. Обчислювальна складність такої бесперебороной процедури дорівнює Q=s+2, s – складність отримання сигнатури для рядка символів.

Приклад використання сигнатурного кодування текстових значень атрибутів представлений в таблиці:

Malware	S(M)	U[S(M)]=1 simulation
A	**73A1**	U(S1=0)=**1**000 0000
B	**FC67**	U(S2=0)=1000 **1**000
C	**2C67**	U(S2=0)=101**0** 1000
D	**ACF0**	U(S3=0)=1010 11**0**0
E	**E143**	U(S4=0)=1001 110**1**
F	**FC67**	U(S2=1)=1001 1101
G	**ACF0**	U(S3=1)=1001 1101
H	**EF43**	U(S4=0)=11**0**1 1101

Результуючий U-вектор унітарного кодування практично без обчислювальних витрат дозволяє вирішувати наступні завдання: 1) Визначати число одиничних координат, що дає можливість генерувати унітарні коди текстових значень змінних і довжину кожного коду. 2) Декодувати сигнатури і текстові фрагменти за координатами одиничних значень U-вектора. 3) Без переборних обчислень визначати статус текстового фрагмента: є він новим або вже існуючим в бібліотеці значень параметра.

Сигнатура виконує роль адреси-посередника між довгим текстовим malware фрагментом і вектором покриття, який моделює процес ідентифікації оригінальності значень текстових змінних: U[S(M)]={0,1}.

Заповнення вектора покриття за правилом: U[S(M)]=1 дає можливість змоделювати кількість і якість оригінальних текстових фрагментів. У загальному випадку технологія моделювання невпорядкованих чисел для їх упорядкування за зростанням укладається в схему: Числа – Як адреси – Вектор одиниць – Запис адрес одиничних координат в порядку їх зростання. Технологія моделювання текстових фрагментів для їх упорядкування за зростанням сигнатур, що виконують роль адрес, укладається в схему: Тексти – Як адреси-сигнатури – Вектор одиниць – Запис адрес-сигнатур одиничних координат в порядку їх зростання.

Таким чином, інтерпретація чисел D=A, як адрес 1-координат для початкового 0-вектора, є ефективною технологією рішення комбінаторних завдань: U(D)=1. Слід врахувати, що malware вихідні тексти, в загальному випадку, неможливо уявити у вигляді адрес вектора. Тому необхідний буферний компонент, який, з одного боку, може компактно ідентифікувати malware змінну, а з іншого – бути її адресою. Таким компонентом може виступати 16-розрядна сигнатура (або hash-функція), як ідентифікатор-адреса для тексту і вектора: U[S(M)]=1.

M	S(M)	U[S(M)]		D	U(D)	D*
A	**73A1**	1000 0000		3	0010 00000	1
B	**FC67**	1000 1000		6	0010 01000	2
C	**2C67**	1010 1000		8	0010 01010	3
D	**ACF0**	1010 1100		2	0110 01010	4
E	**E143**	1001 1101		4	0111 01010	6
F	**FC67**	1001 1101		9	0111 01011	7
G	**ACF0**	1001 1101		1	1111 01011	8
H	**EF43**	1101 1101		7	1111 01111	9

Big Data malware коди мають два суттєвих недоліки: 1) Для них необхідна велика пам'ять. 2) Для їх аналізу необхідно виконувати порівняння вхідного фрагмента даних з вже існуючими malware в бібліотеці або базі даних. Обчислювальна складність такого переборного порівняння дорівнює Q=L(Input)xN(Records)xL(Record). Чим більше записів і чим більше їх довжина, тим кубічний більше часу потрібно на визначення приналежності

вхідного фрагмента до вже існуючого запису в бібліотеці. Якщо такого запису немає, то виконується поповнення бібліотеки новим записом після фіксації негативного результату порівняння вхідного фрагмента з бібліотечними атрибутами.

Як уникнути істотних витрат часу, що мають кубічну функціональну залежність обчислювальної складності? За рахунок надмірності пам'яті і додаткових обчислювальних процедур, що дають можливість зменшити обсяг прямих обчислювальних витрат, пов'язаних з перебором при порівнянні malware значення вхідної змінної з таблицею бібліотечних рішень. Для цього існує технологія хеширования або сигнатурного стиснення двійковій інформації в 16-розрядний код сигнатуру-адресу, яка з імовірністю 0,9998 відображає як завгодно довгу двійкову послідовність. Платою за таку компактність є додаткові часові витрати для отримання сигнатури Q=L(Input), які можуть бути неприйнятними при сигнатурному стисненні невеликих обсягів даних, порядку сотень біт. Інтегрально обчислювальна складність сигнатурного методу аналізу вхідних потоків даних визначається виразом: Q = L(Input) + L(Sign) x N(Sign), де перший доданок формує час сигнатурного стиснення вхідної послідовності, другий – задає час порівняння вхідної сигнатури з бібліотечними аналогами. В результаті обчислювальна складність стає квадратичною, але вже по відношенню до сигнатурних кодів, які за розмірністю істотно менше, ніж вхідні потоки даних. Якщо порівнювати сигнатурні коди паралельно, що можна здійснити на низькорівневих мовах програмування або опису апаратури, то обчислювальна складність аналізу CSC-процесів стає лінійної і залежною від часу отримання сигнатури і числа бібліотечних аналогів, з якими виконується порівняння: Q=L(Input) + N(Sign).

Порівняння сигнатур здійснюється з метою привласнення кожному унікальному 16-розрядному коду десяткового номера, який далі може служити адресою для генерування унітарного коду сигнатури або вхідного

потоку даних. Якщо число сигнатур в базі даних дорівнює 10, то це означає, що розмірність унітарного коду для ідентифікації сигнатур дорівнює 10.

Перетворення: <input – **Sign** – Digit – Ucode> має дізрапторну практичну спрямованість, яка пов'язана з відсутністю перебору на рівні даних при взаємодії вхідного потоку з бібліотекою або таблицею рішень. Однак такий перебір тут існує на рівні сигнатур, який дає можливість формувати компактну базу даних шляхом присвоєння кожній сигнатурі десяткового номера або унітарного коду для виконання паралельних логічних операцій:

M	S(M)	D[S(M)]	U{D[S(M)]}
A	**73A1**	1	1000 00
B	**FC67**	2	1100 00
C	**2C67**	3	1110 00
D	**ACF0**	4	1111 00
E	**E143**	5	1111 10
F	**FC67**	2	1111 10
G	**ACF0**	4	1111 10
H	**EF43**	6	1111 11

Альтернативне рішення може бути отримано шляхом використання сигнатури як адреси для формування унітарного коду в масштабі 2**16=65536. Розмірність адресного простору дає можливість абсолютно без перебору і порівняння вирішувати завдання по визначенню приналежності вхідного потоку даних до значень атрибутів бібліотеки. Завдання вирішується шляхом запису одиничного значення за адресою-сигнатурою вхідного потоку. Це дає можливість звести процедуру ідентифікації вхідного потоку шляхом запису одиниці в нульову координату інтегрального вектора унітарних кодів. Якщо за даною адресою вже існує одиниця, то вхідний потік не є оригінальним і робиться висновок про його приналежність до значення конкретного атрибута:

M	S(M)	U[S(M)]	D[S(M)]
A	**73A1**	…01000 000…	1
B	**FC67**	…01100 000…	2
C	**2C67**	…01110 000…	3
D	**ACF0**	…01111 000…	4
E	**E143**	…01111 100…	5
F	**FC67**	…01111 100…	2
G	**ACF0**	…01111 100…	4
H	**EF43**	…01111 110…	6

Рішення даного завдання дозволяє: 1) Ідентифікувати великі потоки даних і вигляді компактної сигнатури (16-ковий алфавіт: 0-9, A, C, F, H, P, U), яка може являти собою адресу одиничного значення координати в унітарному векторному просторі, розмірністю 2**16; 2) Уникнути кубічної складності переборного завдання з пошуку в базі даних вхідного потоку і привести її рішення до лінійної обчислювальної складності Q=2n. 3) Впорядковувати вхідні потоки по чисельним значенням сигнатур з метою зменшення обчислювальної складності при вирішенні завдань аналізу CSC-процесів. 4) Визначати актуальне число розрядів для унітарного кодування множини вхідних потоків даних після отримання множини оригінальних сигнатур.

Можна не використовувати буферний компонент у вигляді сигнатури, яку досить важко обчислювати через неможливість її генерувати паралельно. У цьому випадку пропонується кожному новому текстовому фрагменту mal-ware привласнювати номер-ідентифікатор, який може виступати і адресою для формування унітарного коду фрагмента шляхом занесення одиниці за адресою у відповідному полі вектора покриття. Однак в даному випадку кожен новий текстовий фрагмент, що надходить на вхід <malware-address> кодера, потребує порівняно з вже існуючими текстами.

Формальна постановка задачі: отримати мінімальну множину символів з вхідного набору букв (тексту), що має повторення.

Example 1 (minimal set number should be ordered).

Input = {4,5,2,8,4,3,7,1,5,4}.

Buffer = (000 000 000). Writing 1-unit on addresses, pointed in Input-set:

4(000 100 000), 5(000 110 000), 2(010 110 000), 8(010 110 010), 4(010 110 010), 3(011 110 010), 7(011 110 110), 1(111 110 110), 5(111 110 110), 4(111 110 110).

Output = 1-unit address decoding (111 110 110) = {1,2,3,4,5,7,8}.

Computing Complexity Q=2n instead of Q=1/2[n**2].

Example 2 (minimal set number).

Input = {we are the world, we are the children}.

Buffer = (w-1, e-2, a-3, r-4, t-5, h-6, o-7, l-8, d-9, c-10, h-11, I-12 , n-13).

Initial U-vector = (000 000 000 00) – Resulting U-vector = (111 111 111 11)

Output = {w, e, a, r, t, h, o, l, d, c, n}

Example 3 (minimal set number).

1) Input = {we are the world, we are the children}.

2) Buffer (coder) = (a-1 b-2 c-3 d-4 e-5 f-6 g-7 h-8 i-9 j-10 k-11 l-12 m-13 n-14 o-15 p-16 q-17 r-18 s-19 t-20 u-21 v-22 w-23 x-24 y-25 z-26)

3) Initial U-vector = (000 000 000 000 000 000 000 000 00)

4) Resulting U-vector = (101 110 010 001 011 001 010 010 00)

5) Output (pattern) = {a, c, d, e, h, l, n, o, r, t, w}

3.6. Модель кубітно-матричного процесора для CSC-компьютінга

Рішення задач malware аналіза пов'язано зі створенням моделі, методу та інтерпретативного гнучкого процесора CSC-комп'ютингу, спрямованого на автоматичний синтез і аналіз кубітних логічних схем, орієнтованих на моніторинг, моделювання, розпізнавання і управління CSC-процесами і явищами:

1) Процесор CSC-комп'ютингу на основі синтезу кубітних логічних схем для моніторингу, моделювання, розпізнавання і управління CSC-процесами.

2) Кубітно-матрична модель універсуму змінних і універсуму значень кожної багатозначної змінної для синтезу логічних схем malware-патернів, орієнтованих на аналіз і управління CSC-процесами.

3) Кубітно-матричний метод синтезу логічної схеми для моделювання та розпізнавання malware-функціональностей на основі унітарного кодування значень багатозначних змінних з метою оптимального управління CSC-процесом.

4) Кубітно-матричний інтерпретативний метод моделювання вхідних потоків malware-даних CSC-процесів на основі використання еталонних логічних елементів malware-функціональностей з унітарним кодуванням багатозначних змінних. Метод відрізняється від розумних блокчейн-контрактів, що представляють собою виконуваний код, інтерпретативним аналізом гнучких структур даних, що дає можливість змінювати модель CSC-процесу в online режимі реального часу.

5) Тестування і верифікація кубітно-матричної архітектури CSC-комп'ютингу на прикладах аналізу і розпізнавання процесів у вхідних потоках великих даних, пов'язаних з кіберфізичним відображенням і управлінням діяльністю організації.

Кубітно-матрична інтерпретативна модель malware-патернів і її унітарно-кодований екземпляр для моделювання і актюаторного управління CSC-процесом представлені на рис. 3.11.

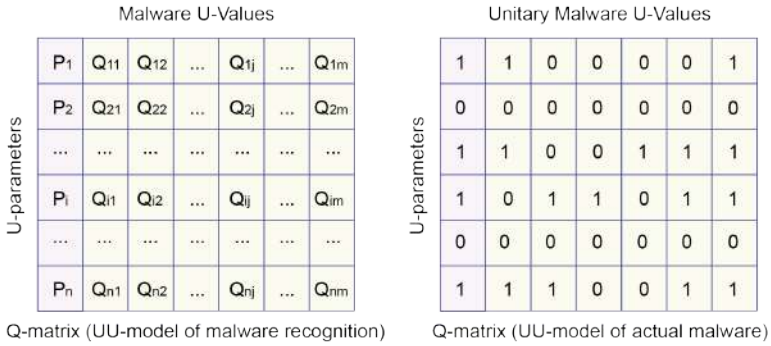


Рисунок 3.11 – Кубітна UU-матриця CSC-процесу

Фактично, Q-матриця являє собою формальний запис протоколу, анкети або ідеального блокчейн-контракту, який додатково виробляє актюаторні сигнали, які спонукають систему виконувати певні дії для

досягнення мети – ліквідації malware. Таким чином, прямі і безпосередні інтеракції між CSC-роботом-автоматом і користувачем створюють CSC-комп'ютинг, який використовує Q-матрицю і логічні схеми, спрямовані на отримання CSC-сервісів за мінімальний час без участі людини.

3.6. Сігнатурно-кубітний метод синтезу еталонних логічних схем

Комбінаційна модель для сигнатурно-кубітного методу синтезу еталонних логічних схем malware-функціональностей використовує унітарне кодування сигнатур для кодів деструктивних компонентів і формування кубітних матриць, що дає можливість в паралельному режимі визначати наявність malware в обчислювальній системі або мережі. Щоб показати схему для формування структур даних malware-функціональності (U, Q, X), необхідно з'єднати вхідний потік даних S для кожної malware-змінної. Схема для формування структур даних CSC-процесу (V, P, Y) з'єднує вхідні потоки даних для змінних F щодо кожної malware-функціональності, рис. 3.12.

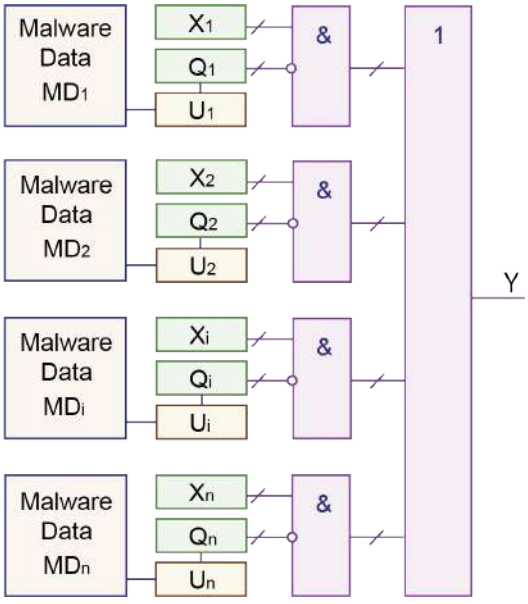


Рисунок 3.12 – Сигнатурно-матрична модель комбінаційної схеми для malware-аналізу

Універсум універсумів (UU) являє собою дворівневу модель процесу або явища, призначену для активного цифрового моделювання з метою точного розпізнавання великих потоків вхідних даних шляхом метричного порівняння з заданими еталонами. Кожен з параметрів представлений значеннями, які в сукупності складають універсум другого рівня. В результаті виходить UU-метрика функціональності еталонного процесу або явища, щодо якого можна вимірювати-моделювати потік вхідних даних, які претендують на дану роль. Для проектування CSC-комп'ютингу використовуються два види моделей логічних процесорів. Кубітно-регістрова модель (КРМ) оперує двома входами логічного and-елемента, на які подаються: 1) кубітний вектор-еталон багатозначної змінної malware-функціональності; 2) вхідний вектор даних CSC-процесу, що підлягає розпізнаванню. Обидва вектора паралельно порівнюються між собою за процедурою $Y=\vee[(X\wedge Q)\oplus X]$, яка включає три паралельних операції, що призводить до двійкового результату: Y = 0, якщо X належить вектору Q, в іншому випадку Y=1. Кубітно-логічна модель (КЛМ) оперує одним входом логічного and-елемента, на який подається значення-адреса з даних CSC-процесу, що відповідає даній змінній. За адресою, отриманою в результаті кодування текстового фрагмента, визначається координата кубітного вектор-еталона багатозначної змінної malware-функціональності, значення якої формує стан виходу Y для розпізнавання значення вхідних даних щодо заданого кубітного вектора, що формує логічний еталон. Іншими словами, одна ітерація моделювання на кубітному елементі КЛМ-моделі malware-функціональності визначає приналежність вхідного значення до кубітного вектору логічного еталона. КРМ модель змінної malware-функціональності дає можливість паралельно, за один автоматний цикл, визначати приналежність вхідного вектора до кубітного вектору логічного еталона. У цьому їх відмінність. Природно, що вхідні впливу для обох моделей формуються з текстових фрагментів вхідних даних шляхом їх унітарного

кодування на універсальній множині значень кожної змінної malware-функціональності.

3.7. Процесорна архітектура CSC-комп'ютингу. Проблема і рішення

Проблема CSC комп'ютингу полягає в складності формалізації malware-логіки, яку необхідно привести до цифрового детермінізму функціональних відповідностей, яке виключає ймовірність і невизначеність. Піти від евристики у бік автоматизації синтезу та аналізу логічних схем CSC-процесингу для моделювання та передбачення malware-колізій – завдання, актуальна для ринку. Для її вирішення пропонується трансформувати функціонально закінчений потік великих даних до структурованої матричної двійкової форми, яка унітарно кодує сукупність malware-змінних і всі можливі їх значення, складові універсум примітивів. Метод для вирішення проблеми представлений синтезом класів еквівалентних відношень на сукупності змінних P:

$$P = \{P_1, P_2, \dots, P_i, \dots, P_j, \dots, P_n\}, P_i \cap P_j = \emptyset,$$

де кожна з них $P_i = \{P_{i1}, P_{i2}, \dots, P_{ik}, \dots, P_{ir}, \dots, P_{in}\}$ приймає універсальну множину malware-значень, що створюють між собою еквівалентні відношення $P_{ik} \sim P_{ir}$ і утворюють при цьому порожні перетини $P_{ik} \cap P_{ir} = \emptyset$.

Метрика: визначення, аксіоми і рівняння CSC-комп'ютингу.

1) Метрика є спосіб вимірювання відстаней $d_i \in D$ між процесами і явищами в просторі заданих параметрів з виконанням аксіоми циклічного конволюціонного замикання:

$$D = \sum_{i=1}^{n} d_i = 0$$

2) Вимірювання – процедура визначення відстані між кінцевою множиною процесів або явищ, відмінних від нуля.

3) Параметрами виступають логічні змінні $P = \{P_1, P_2, \dots, P_i, \dots, P_n\}$, які із заданим ступенем адекватності описують процес або явище.

4) Змінні визначаються за допомогою їх значень $P_i = \{P_{i1}, P_{i2}, \dots, P_{ij}, \dots, P_{in}\}$, кількість яких не може бути менше двох.

5) Матриця універсумів є упорядкована сукупність змінних і їх значень $U = [P_{ij}] = [U_{ij}]$ для метричного вимірювання процесу або явища. Унітарно кодована матриця U має одиничні значення по всіх координатах. Матриця універсумів U може бути представлена одним вектором шляхом конкатенації її рядків

$$P = (P_1 * P_2 * \dots * P_i * \dots * P_n).$$

6) Матриця malware $Q = [Q_{ij}]$, є підмножина значень змінних універсальної матриці $Q \in U$, яке формує зразок конкретного процесу або явища. Найбільш зручною формою завдання malware є матриця унітарного двійкового кодування значень змінних.

7) Матриця вхідних даних $X = [X_{ij}]$ є підмножина (двійкових) значень змінних універсальної матриці $X \in U$, яке формує фрагмент реального процесу або явища.

8) Матриця вимірювання $Y = [Y_{ij}] = [Q_{ij}] \oplus [X_{ij}]$ є підмножина $Y \in U$ (двійкових) значень змінних універсальної матриці, яке формує відмінності в однойменних координатах матриць Q і X шляхом виконання паралельної xor-операції між malware і фрагментом обчислювального процесу або явища.

9) Відстань є скалярна оцінка кількості відмінностей, зазначених одиницями, в однойменних координатах матриць вимірюваних процесів або явищ. Відстань визначається шляхом підрахунку одиничних координат в матриці вимірювання:

$$d(Q, X) = \sum_{j=1,m}^{i=1,n} Y_{ij}.$$

10) Функція відмінності є відношення відстані (кількості відмінностей в однойменних координатах) до загальної кількості координат матриці (вимірювання):

$$\mu = \frac{d(Q,X)}{n \times m} = \frac{\sum_{j=1,m}^{i=1,n} Y_{ij}}{n \times m}.$$

Далі розглядається процесорна архітектура активного online кіберфізичного cyber security комп'ютингу, яка характеризується моніторингом вхідних потоків malware-даних, їх подальшим моделюванням на еталонних логічних схемах malware-функціональностей, що дає можливість в online режимі актюаторно управляти процесом видалення деструктивних компонентів.

Логічна структура, представлена на рис. 3.13, характеризується комп’ютинговою архітектурою malware-аналітики, яка має всі вісім компонентів моделі універсального обчислювача, взаємодіючих між собою за формулою: вхід R ініціює команди для виконання CSC-процесу, який починається з отримання D-ресурсів: фінансових, кадрових, інформаційних, що надходять на виконавчий механізм E, який активує сенсори, що передають інформацію по шині моніторингу M для подальшого формування потоку S-даних, призначеного для синтезу U-матриці універсуму змінних для опису CSC-процесу за допомогою універсумів вербальних значень кожної змінної, яка слугує базовим форматом для синтезу X-матриці векторів вхідних даних по кожній змінній для виконання ітерації моделювання відносно до Q-матриці, що задає сукупність кубітних двійкових векторів для опису еталонного malware-патерна за всіма змінними, що дає можливість на основі &-операції обчислювати функцію приналежності μ, що має вихід візуалізації стану V CSC-процесу, і відповідну Y, у вигляді чисельного значення кількості різних двійкових однойменних координат, отриманого при порівнянні матриць X і Q, який визначається за допомогою паралельної операції $(X \wedge notQ) = Y$, яка дає можливість синтезувати актюаторні вербальні сигнали W, відповідні одиничним значенням координат вихідної матриці Y в форматі U-матриці, які по шині A ініціюють обчислювальні процедури, спрямовані на усунення розходжень між X-матрицею і еталонною Q-матрицею malware-патерну шляхом корекції координат X-матриці за допомогою інфраструктури E, що забезпечує виконання CSC-процесу компанії для отримання сервісів або продукції P.

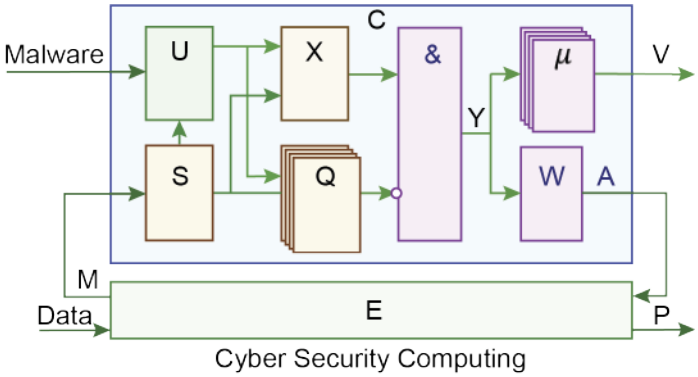


Рисунок 3.13 – Логічна матрична паралельна схема CSC-комп'ютингу

Архітектура CSC-комп'ютингу характеризується кінцевою множиною Q-матриць для паралельної обробки вхідних X-матриці з метою отримання μ-матриць, також обчислюються паралельно, що представляє собою універсум функцій приладдя чергового вхідного впливу malware-паттернам, серед яких вибирається мінімальна оцінка відмінностей і скалярні оцінки для усунення malware.

Оригінальність моделі кубітно-матричного процесора, що відповідає архітектурі, представленої на рис. 3.13, характеризується використанням кубітних матриць даних для виконання операцій CSC-комп'ютингу, структура якого містить механізми управління C і виконання E, з'єднані шинами моніторингу M і актюації A, де блок управління має зовнішній вхід команд R і вихід візуалізації стану CSC-процесу V, а блок виконання містить зовнішній вхід даних D і вихід сервісів або продукції P, що в сукупності становить вісім компонентів, що характеризують CSC-комп'ютинг, який починається з визначення вхідного потоку даних S, що надходить від сенсорів виконавчого механізму, обслуговуючого CSC-процес, який призначений для синтезу U-матриці універсуму змінних і опису CSC-процесу за допомогою універсумів вербальних значень кожної змінної, яка є базовим шаблоном для сінтезу X-матриці вхідних даних і наступного моделювання сумісно з вже визначеними Q-матрицями, що задають множину еталонних

malware-патернів, що дає можливість на основі &-операції паралельно обчислювати n матриць Yi = (X ∧ notQi), (i=1,n) моделювання, аналіз яких на мінімальне число одиничних координат min(Yi=1), (i=1,n), дає можливість визначити номер і матриці, що має мінімальне значення з n функцій приналежностей

$$\mu = \min_i \mu_i \leftarrow \mu_i = \sum_{j=1,k}^{r=1,m} Y_{ijr},$$

формують чисельні значення відмінностей двійкових однойменних координат, отриманих при порівнянні матриці X і n матриць Q, що дає можливість синтезувати матрицю актюаторних вербальних сигналів W, відповідну одиничним значенням координат вихідний матриці Y, що має minμ, в форматі вербальних значень U-матриці, які ініціюють обчислювальні процедури у виконавчому механізмі E, спрямовані на усунення розходжень між X-матрицею і Q-матрицею malware-патерну с min μ, шляхом корекції координат X-матриці, що забезпечує оптимальне виконання мети P CSC-процесу.

Алгоритм роботи кубітно-матричного процесора, представленого на рис. 3.14, визначає послідовність процедур комп'ютингу в часі і просторі для розпізнавання malware-патернів у вхідних потоках даних і вироблення актюаторних сигналів-впливів з метою усунення розбіжностей між ними або деструкції malware, що починається з блоку 1) ініціювання алгоритму з наступним переходом до 2) блоку смислового аналізу вхідних даних S, що надходять від входу D і сенсорів M цифровий інфраструктури, при цьому виконується 3) перевірка наявності U-matrix з метою переходу на блок 6, а в разі її відсутності алгоритм активує процедура розбиття текстових фрагментів на непересічні класи еквівалентності для 4) вилучення універсуму malware або змінних-примітивів, де для кожної з них визначається універсум значень, що в сукупності формує U-матрицю універсуму універсумів, яка служить шаблоном для наступного 5) сінтезу

кубітних Q-матриць malware-патернів і 6) кубітної X-матриці вхідних даних, які у подальшому використовуються для виконання 7) паралельної &-операції та отримання n матриць Yi=(X∧notQi), (i=1,n) моделювання, в яких підраховується кількість одиничних координат min (Yi = 1), (i = 1, n), дає можливість 8) визначити номер і матриці, що має мінімальне значення з n функцій приналежностей $\mu = \min_i \mu_i \leftarrow \mu_i = \sum_{j=1,k}^{r=1,m} Y_{ijr}$, формують чисельні значення відмінностей двійкових однойменних координат, отриманих при порівнянні матриці X і n матриць Q, що дає можливість 9) синтезувати матрицю актюаторних вербальних сигналів W, відповідну одиничним значенням координат вихідний матриці Y, що має minμ, В форматі вербальних значень U-матриці, які, 10) при μ≠0, ініціюють 11) обчислювальні процедури у виконавчому механізмі E, спрямовані на усунення розходжень між X-матрицею і Q-матрицею malware-патерну с minμ, шляхом корекції координат X-матриці, що забезпечує оптимальне виконання мети P CSC-процесу, а при існуванні μ≠0 виконується завершення роботи алгоритму на заданій матриці вхідних впливів, який забезпечує реалізацію CSC-комп'ютингу, орієнтованого на деструкцію malware за рахунок цифровізації, автоматизації та просторово-часової оптимізації CSC-процесу при створенні сервісу або продукту.

Таким чином, матрична структура даних для паралельного виконання операцій CSC-комп'ютингу дає можливість істотно підвищити швидкодію інтерпретативного паралельного моделювання вхідних потоків malware-даних для розпізнавання в них malware-патернів з метою генерування актюаторних сигналів, що усувають malware або відмінності в X-матриці по відношенню до одного з еталонних CSC-патернів.

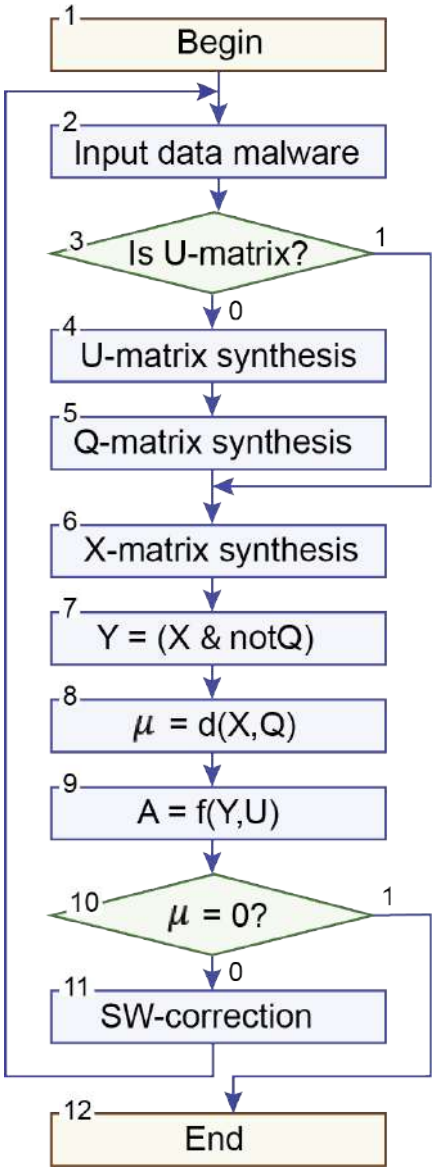


Рисунок 3.14 – Алгоритм CSC-комп'ютингу

3.7. Висновки до розділу 3

Наукова новизна полягає в створенні логічного процесора для паралельного моделювання та розпізнавання malware-патернів в потоках великих даних на основі створення інтерпретативних кубітних матричних моделей, методів і архітектур CSC-комп'ютингу, спрямованого на автоматичний синтез і аналіз логічних схем, орієнтованих на моніторинг і управління CSC-процесами і явищами для усунення деструктивних компонентів у кіберпросторі:

1) Запропоновано паралельний сигнатурно-кубітний метод моделювання malware-driven великих даних, який характеризується використанням сигнатурного аналізу і кубітними структурами даних, що дає можливість в паралельному режимі визначати приналежність поточного коду до існуючих деструктивних компонентів в malware library.

2) Розроблено сигнатурно-кубітний метод синтезу еталонних логічних схем malware-функціональностей, який відрізняється від аналогів унітарним кодуванням сигнатур для кодів деструктивних компонентів і формуванням кубітних матриць для моделювання, що дає можливість в паралельному режимі визначати наявність malware в обчислювальній системі або мережі.

3) Розроблено процесорну сигнатурно-кубітну модель-архітектуру активного online cyber security комп'ютингу, яка характеризується моніторингом вхідних потоків malware-даних, їх подальшим моделюванням на еталонних логічних схемах malware-функціональностей, що дає можливість в online режимі актюаторно управляти процесом видалення деструктивних компонентів.

4) Напрями майбутніх досліджень. Так само як біологічні віруси деструктурують людину, кібер віруси вражають організм world-комп'ютингу в масштабах планети, завдаючи людству багатомільярдну матеріальну та моральну шкоду. Одним з можливих варіантів може бути кібер-імунітет, як кіберфізичний моральний комп'ютинг метричного вичерпного моніторингу всіх процесів і явищ для цифрового human-free управління на основі моделювання, передбачення наслідків від malware, діагностування та знищення некорисних програмних додатків.

Результати розділу опубліковані у роботах [1, 7, 11, 12, 14, 17, 18, 26].

3.8. Список використаних джерел до розділу 3:

1. Lehto, Martti, Neittaanmäki, Pekka Cyber Security: Analytics, Technology and Automation, Springer, 2015. 269 p.]

2. Orojloo Hamed, Mohammad Abdollahi Azgomi. Modelling and evaluation of the security of cyber-physical systems using stochastic Petri nets. IET Cyber-Physical Systems: Theory & Applications (2019), 4 (1), P. 50-57.]

3. https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/

4. https://www.gartner.com/doc/3891569?src Id=1-7251599992&cm_sp=swg-_-gi-_-dynamic

5. A. Gupta and RK Jha, "A Survey of 5G Network: Architecture and Emerging Technologies," in IEEE Access, vol. 3, pp. 1206-1232, 2015.

6. C. Zhu, VCM Leung, L. Shu and ECH Ngai, "Green Internet of Things for Smart World," in IEEE Access, vol. 3, pp. 2151-2162, 2015.

7. K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," in IEEE Access, vol. 4, pp. 2292-2303, 2016.

8. Blockchains: How They Work and Why They'll Change the World IEEE Spectrum. October 2017.

9. A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities," in IEEE IoT Journal, vol. 1, no. 1, pp. 22-32, Feb. 2014.

10. Frahim J. Securing the Internet of Things: A Proposed Framework / J. Frahim // Cisco White Paper.- 2015.

11. Kharchenko V. Green IT Engineering: Concepts, Models, Complex Systems Architectures / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In the book series "Studies in Systems, Decision and Control" (SSDC) .- vol. 1.- Berlin, Heidelberg: Springer International Publishing.- 2017.

12. Kharchenko V. Green IT Engineering: Components, Networks and Systems Implementation / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In the book series "Studies in Systems, Decision and Control" (SSDC). - vol. 2.- Berlin, Heidelberg: Springer International Publishing.- 2017.

13. Memory-Driven Computing. [Online]. Available: https://www.labs.hpe.com/next-next/mdc

14. Benenti G., Casati G., Strini G. Principles of Quantum Computation and Information. Volume 1: Basic Concepts.-World Scientific.- 2004.- 256 p.

15. Imai Hiroshi, Hayashi Masahito. Quantum Computation and Information. From Theory to Experiment.- Springer.-2006.- 234 p.

16. Nielsen MA, Chuang IL Quantum Computation and Quantum Information.- Cambridge University Press.- 2010.- 710 p.

17. Abramovici M. Digital System Testing and Testable Design / M. Abramovici, MA Breuer and AD Friedman.- Comp. Sc. Press.- 1998.- 652 p.

18. Benso A. Control-flow checking via regular expressions / A. Benso, S. Di Carlo, G. Di Natale, P. Prinetto, L. Tagliaferri // Proceedings 10th Asian Int. Test Symposium.- Kyoto.- 2001.- P . 299-303. [Online]. Available: http://dl.acm.org/citation.cfm?id= 872025.872649

19. Vladimir Hahanov. Cyber Physical Computing for IoT-driven Services. New York. Springer. 2018. 279p.

20. Hahanov V.I. Qubit technologies for analysis and diagnosis of digital devices / V.I. Hahanov, T. Bani Amer, S.V. Chumachenko, E.I. Litvinova // Electronic Modeling.- vol. 37, no. 3.- 2015.- P. 17-40.

21. Хаханов В.И. Кубитные структуры данных вычислительных устройств / В.И. Хаханов, В. Гариби, Е.И. Литвинова, А.С. Шкиль // Электронное моделирование. - 2015. - Т. 37, № 1. - С. 76-99.

22. Hahanov V. Cloud-driven Cyber Managing Resources / V. Hahanov, S. Chumachenko, E. Litvinova, O. Mishchenko, I. Yemelyanov, Bani Amer Tamer // Australian Journal of Scientific Reseach. – № 1(5). – 2014. – С. 202-215.

23. Hahanov I. QuaSim – Cloud Service for Digital Circuits Simulation / I. Hahanov, W. Gharibi, I. Iemelianov, T. Bani Amer // Proceedings of IEEE East-West Design & Test Symposium. – 2016.– Yerevan, Armenia.– P. 363- 370.

РОЗДІЛ 4
МЕТОДИ ДІАГНОСТУВАННЯ КІБЕРАТАК З ВИКОРИСТАННЯМ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ

Розглядається модель загроз кіберпростору, а також методи діагностування кібератак на кіберпростір з використанням алгоритмів машинного навчання на основі великих даних, що дозволяють виявити загрози, запропоновані у моделі загроз кіберпростору: 1) метод атрибутно-орієнтованого впізнання Інтернет посилань з використанням частотних шаблонів; 2) метод детектування поліморфних шкідливих програм; 3) метод пошуку криптопримітивів в троянських програмах-шифрувальниках.

4.1 Модель загроз кіберпростору

Попередні дослідження в області безпеки кіберпростору базувалися або на аналізі стану комп'ютерної мережі та виявлення в ній вразливостей [1], або використовуючи як критерій безпеки багаторакурсні параметри для оцінки і прогнозування безпеки мережевої системи [2].

У більш пізніх дослідженнях даний підхід визнається неспроможним, оскільки не враховує поведінку самого користувача при виникненні системних аномалій. Згідно [3] термін кіберпростору визначається як "a massive socio technical system of systems, with a significant component being the humans involved". Таким чином, автори пов'язують кібератаки з соціальними, політичними, економічними і культурними явищами.

На сьогоднішній день віртуальний простір користувача розширюється за рахунок повсюдної експансії соціальних мереж та Інтернет сервісів, які дозволяють за рахунок сучасних Інтернет технологій перевести обробку і зберігання даних в хмару. Таким чином, користувач стає менш прив'язаним до свого персонального комп'ютера, який служить лише для доступу до онлайн сервісів з метою отримання необхідних даних і здійснення операцій над ними.

Даний підхід дозволяє абстрагуватися від апаратних характеристик пристрою доступу в Інтернет і використовувати будь-які мобільні апаратно-програмні платформи для вирішення широкого спектра завдань у "хмарі" [4]. Як приклади можна привести сервіси для роботи з офісними документами (Google Documents, Microsoft Office Live), сховища файлів і зображень, картографічні сервіси, перекладачі, календарі та, нарешті, соціальні мережі, де кожен учасник мережі може зберігати інформацію про себе і отримувати доступ до мультимедійного вмісту інших користувачів. Все це є свідченням переходу людства до загального використання хмарних технологій для вирішення персональних завдань.

Як основа для існування кіберпростору виступає множина онлайн сервісів в рамках концепції хмарних обчислень (Cloud Computing), що дозволяють отримати доступ до необхідної інформації в хмарі, не прив'язуючись до конкретної апаратно-програмній платформі. Так як інформація, яку публікує в хмарі, а також віртуальні особистості в соціальних мережах можуть викликати інтерес у третіх осіб, то виникає необхідність захисту кіберпростору від кібератак, а також забезпечення конфіденційності даних в хмарі на стороні провайдера.

Завдання захисту є актуальною, так як дані технології повсюдно починають використовуватися організаціями для створення сервісної бізнес і державної інфраструктури. Відповідно, необхідно гарантувати безпеку корпоративних даних в хмарі, що є важко здійсненним завданням. Для її вирішення компанії укладають з сервіс провайдерів Service Level Agreement (SLA), де описуються питання безпеки на різних рівнях уявлення [5].

З цією метою була створена організація Cloud Security Alliance (CSA) [6], в яку входять провідні компанії в галузі Cloud Computing, такі як: Google, Microsoft, IBM, Intel, Symantec, PGP, Sun, HP, Dell, VeriSign, TrendMicro, McAfee, Cisco, VMWare та інші. Основна мета організації – просування передових практик щодо захисту в рамках Cloud Computing.

Найбільш розвиненою інфраструктурою захисту користувача в хмарі володіє компанія Intel, яка розробила комплекс рішень для забезпечення безпечного доступу і зберігання даних в хмарі. Технології компанії Intel підтримуються лідерами антивірусної індустрії компаніями Symantec і McAfee. Концепція користувальницької безпеки в хмарі компанії Intel ґрунтується на ряді інноваційних технологій [7].

Запропоновані технології захисту кіберпростору включають:

1. Ідентифікацію користувача за допомогою вбудованого апаратного токена (Identity Protection Technology);
2. Управління і моніторинг доступу в хмару (Expressway Cloud Access 360);
3. Використання токенізації в хмарі для доступу до електронного банкінгу (Expressway Tokenization Broker);
4. Деактивация комп'ютера користувача через Інтернет (Anti-Theft Technology);
5. Шифрування даних (AES-NI);
6. Захист гипервизора і віртуального середовища хмари (Trusted Execution Technology).

З огляду на існуючі розробки в цій галузі, в рамках даної роботи пропонується модель захисту кіберпростору, що передбачає створення безпечного середовища для зберігання і обробки даних з використанням технології хмарних обчислень (cloud computing).

4.1.1. Проблеми захисту персональних даних і доступу до них

Хмарні технології дозволяють абстрагуватися від клієнтської комп'ютерної системи і перенести всю інформацію в глобальну мережу за допомогою хмарних сервісів. Наприклад, в рамках соціальних мереж вже давно можливе використання даних сервісів для зберігання персональної інформації та доступу до мультимедійного контенту.

Таким чином, немає необхідності зберігати цю інформацію локально на комп'ютері користувача, при цьому сам користувач має можливість отримувати доступ до своєї персональної інформації через різні канали доступу в Інтернет і з будь-якої точки географічного простору, наприклад, використовуючи для цього мобільні пристрої.

Проблеми захисту персональних даних в хмарі:

– захищений доступ до хмарного сервісу;

– зберігання паролів;

– конфіденційність і цілісність інформації в хмарі.

Перша проблема може бути вирішена за допомогою SSL шифрування в рамках HTTPS протоколу, або засобами VPN з'єднання.

Зберігання паролів може бути реалізовано за допомогою спеціального ПО – менеджера паролів, або безпосередньо в Інтернет браузері, в якості альтернативи cookies.

Конфіденційність і цілісність інформації в хмарі може гарантувати тільки сам користувач наступними діями.

Налаштування відповідних політик безпеки і прав доступу в кожному конкретному сервісі. Однак варто врахувати, що жоден безкоштовний сервіс не гарантує цілісність, доступність і конфіденційність персональної інформації користувача в силу суб'єктивних (доступ з боку адміністраторів і модераторів) і об'єктивних (отримання доступу до даних за допомогою експлуатації уразливості в сервісі) факторів.

Фільтрація інформації, що публікується інформації. Публікація тільки тієї інформації, яка не зможе нашкодити її власнику.

Шифрування розміщуються в хмарі даних. Можливе використання як симетричного шифрування для швидкого доступу до великих обсягів інформації, так і асиметричного для особливо важливої документації. Також можливе використання методів стеганографії для приховування інформації, наприклад, в медіа контейнерах (зображення, відео, звук).

Для забезпечення доступу до різних соціальних сервісів на даний момент існує єдина система доступу OpenID [8]. OpenID являє собою відкриту децентралізовану систему аутентифікації, яка дає можливість використовуючи один OpenID-акаунт авторизуватися на багатьох абсолютно не пов'язаних між собою сайтах. Найбільш відомих: LiveJournal, Google, MySpace, Yahoo, за винятком тільки соціальної мережі Facebook.

Ідея одного облікового запису істотно спрощує користувачеві доступ в його кіберпростір, без необхідності зберігати десятки паролів для різних Інтернет сервісів.

Прикладом є використання зовнішньої аутентифікації в форматі Identity Federation на базі OpenStack хмари в дослідницькій лабораторії CERN [9]. Центр обробки даних CERN обробляє понад 5 петабайт інформації в день. Використання зовнішнього провайдера для авторизації, дозволяє вченим з 21 країни отримувати доступ до даних, що надходять від Великого адронного коллайдера (БАК) без створення і управління внутрішньої обліковим записом [10].

4.1.2. Основні положення кіберпростору

Створення дружнього і безпечного кіберпростору є головним завданням в роботі над комплексною системою захисту. Вхід в кіберпростір повинен бути інваріантним по відношенню до засобу спілкування (комп'ютер, мобільний телефон, смартфон, планшет). Це означає, що індивідуальний простір в майбутньому не повинно бути пов'язане з персональним комп'ютером і залежно від нього. Кіберпростір позиціонується як зовнішній компонент по відношенню до засобу управління обчислювальними процесами і відображення інформації.

При цьому обчислювальні процеси реалізуються на потужних серверах, а користувач має технологію хмарних обчислень для всіх сфер людської діяльності. Таке спрощення функцій персонального комп'ютера і необхідність створення зовнішнього індивідуального простору, підключеного до

обчислювальних ресурсів Інтернет, ставить нові проблеми розробки засобів захисту і сервісного обслуговування кіберпростору шляхом проектування фільтрів, структур даних, бібліотек і технологій вимірювання взаємодії процесів і явищ в кіберпросторі. Зокрема, засоби захисту інформації слід створювати не тільки для програмних і апаратних продуктів, комп'ютерів і мереж, а й для кіберпростору.

Виходячи з наведених фактів можна говорити про поняття персонального кіберпростору, в якій людина представлений у вигляді сукупності своїх віртуальних уявлень. Дана концепція описана в роботі [11], де поняття кіберпростору (cyberspace) визначено як «метафорична абстракція, яка використовується в філософії і в комп'ютерах, по суті, є віртуальною реальністю.

В основі запропонованої моделі лежать наступні положення кіберпростору.

1) Віртуальний світ прагне до повторення Фізичного - клонів не буде, але буде доступність кожного продукту, як компонента кіберпростору для всіх суб'єктів обох світів.

2) Кожен суб'єкт Фізичного світу матиме в ідеалі один образ (сайт) в Віртуальному світі.

3) Автентичність — бажаний ідеал для всіх конструктивних (законослухняних) суб'єктів обох світів.

4) Сайт в Віртуальному світі не уособлює сутність образу, який повинен бути для індивідуума (прообразу) з Фізичного світу більше ніж відображення в дзеркалі. Це історія, дійсність, метричні і медичні дані, звички, бізнес, побут, відпочинок, майбутнє, контакти, пристрасті.

5) Такий віртуальний образ або цифрова особистість в рамках індивідуальне кіберпростір — має бути надійно захищене від несанкціонованого доступу.

6) Захищати від деструктивних компонентів і дій потрібно буде не комп'ютер або вузол мережі, а суб'єкт кіберпростору (цифрову особистість)

на основі запропонованої технології впровадження надлишкової інфраструктури.

7) Відстань — бінарне відношення об'єктів в кіберпросторі є ступінь зміни, відмінності в процесах чи явищах.

4.1.3. Формальна модель загроз в кіберпросторі

Розглянемо формальну модель відображення об'єктів реального світу в кіберпросторі, а також визначимо властивості, яким має відповідати дане відображення.

Нехай дано R - множина об'єктів реального світу, а С – сукупність об'єктів кіберпростору. При це об'єкти безлічі З є відображенням R, тобто кожному об'єкту r ∈ R відповідає об'єкт або підмножина об'єктів с ∈ С. Якщо об'єкту r відповідає с, то с називається чином об'єкта r в кіберпросторі, а r - прообразом об'єкта с:

$$R \to C . \qquad (4.1)$$

Відображення (4.1) має такі властивості:

Кожному елементу r ∈ R може відповідати кілька елементів с ∈ С.

Двом різним елементам множини R завжди відповідають два різних елемента множини С.

Всякий елемент множини С відповідає хоча б одному елементу безлічі R.

Завдання захисту кіберпростору зводиться до виконання наступних властивостей:

Кожному об'єкту кіберпростору с ∈ С може відповідати тільки один об'єкт з реального світу r ∈ R. Наприклад, один власник облікового запису в соціальній мережі. У разі якщо у віртуального образу кіберпростору в якості прообразу виступає група об'єктів реального світу (наприклад, адміністрація веб або форуму, що складається з декількох учасників), то така група розглядається як єдиний об'єкт реального світу.

Якщо одному прообразу в кіберпросторі $c_1 \in C$ відповідають два і більше образів реального світу $r_1, r_2 \in R$, то в такому випадку об'єкт с є скомпрометованим і не входить більш в індивідуальне кіберпростір об'єкта $r_1 \in R$.

Основним завданням захисту кіберпростору є забезпечення наступного відповідності між об'єктами реального світу R і об'єктами кіберпростору С:

$$r_1 \rightarrow \{c_1^1, c_1^2, ..., c_1^n\}. \quad (4.2)$$

Вираз (4.2) показує відображення, при якому одному об'єкту реального світу відповідає кілька образів в кіберпросторі. При цьому у кожного з образів кіберпростору може бути тільки один оригінальний прообраз. В іншому випадку, якщо у образу кіберпростору є два прообразу, то це говорить про несанкціонований доступ та порушення цілісності даних з боку одного з об'єктів реального світу (наприклад, порушення авторського права або злом облікового запису користувача).

Визначимо поняття композиції бінарних відносин [Haggarty R., Discrete mathematics for computing, Pearson Education Limited, 2002 pp. 91-113].

Нехай R - бінарне відношення між множинами X і Y, а S - бінарне відношення між Y і третім безліччю Z. композицією R і S називається бінарне відношення між X і Z, яке позначається S про R і визначається формулою:

$$S\ o\ R = \{(x, z) :\ x \in X, z \in Z \text{ и } x \rightarrow y, y \rightarrow z \text{ для деякого } y \in Y\}. \quad (4.3)$$

Нове ставлення встановлює зв'язок між елементами множин X і Z, використовуючи елементи з Y в якості посередників.

Розглянемо приклад і визначимо композицію об'єктів кіберпростору. Припустимо об'єкт реального світу John$\in$ R має образи (облікові записи) в таких сервісах кіберпростору, як Facebook, Google $\in$ C, а об'єкт Dave $\in$ R - образ Twitter $\in$ C. При цьому використовуючи дані образи, об'єкт John і Dave можуть мати доступ до сервісів MySpace і LiveJournal за допомогою облікових записів пов'язаного типу з множини C*.

Припустимо відношення P і Q задані наступними графами.

Рисунок 4.1 – Відображення користувачів John і Dave в кіберпростір C і C*.

Уявимо відношення у вигляді матриць M і N. Відповідно кожен елемент матриці M для відносини P буде вирахувано відповідно до формули:

$$\begin{aligned} &M(i, j) = 1, \text{ якщо } (x_i, y_i) \in P, \\ &M(i, j) = 0, \text{ якщо } (x_i, y_i) \notin P. \end{aligned} \tag{4.4}$$

Аналогічно матриця N для відносини Q:

$$\begin{aligned} &N(i, j) = 1, \text{ якщо } (y_i, z_i) \in Q, \\ &N(i, j) = 0, \text{ якщо } (y_i, z_i) \notin Q. \end{aligned} \tag{4.5}$$

Тоді матриці будуть мати такий вигляд:

$$M = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad N = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}. \tag{4.6}$$

Обчислимо композицію P o Q, перемноживши відповідні матриці:

$$\text{P o Q} = M \cdot N = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{4.7}$$

Або у вигляді графа відносини:

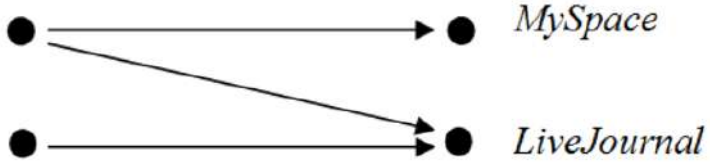


Рисунок 4.2 – Граф композиції відносин P і Q.

Таким чином, за допомогою композиції відносин можливо визначити безліч додаткових об'єктів, які можуть входити до складу кіберпростору за допомогою акаунтів пов'язаного типу.

Композиція відносин об'єктів в кіберпросторі P o Q дозволяє отримати список довірених об'єктів в рамках кібер. На його основі можливо побудувати механізм захисту від фішинг і фармінг атак, метою яких є імітація доступу до легальних сервісів і викрадення даних аутентифікації. У той час як, завдання захищеного зберігання паролів до облікових записів користувача може бути вирішена тільки в рамках відносини P.

У разі якщо, об'єкт Dave ∈ R за допомогою атаки на кіберпростір об'єкта John отримав доступ до облікового запису Facebook ∈ C об'єкта John ∈ R, то відповідно Dave буде також мати доступ до облікових записів об'єкта John в MySpace і LiveJournal через композицію відносин. Матриця відносин P, а також композиція відносини P o Q буде виглядати наступним чином:

$$M^{A} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \qquad \mathrm{P\,o\,Q}^{\mathrm{A}} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}. \tag{4.8}$$

Для виявлення порушення доступу до об'єктів кіберпростору, необхідно провести порівняння матриць відносин до і після атаки за допомогою операції xor.

$$M \oplus M^{A} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \oplus \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$
$$\mathrm{P\,o\,Q} \oplus \mathrm{P\,o\,Q}^{\mathrm{A}} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \oplus \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}. \tag{4.9}$$

З порівняння матриць M випливає, що зміни відбулися з доступом до об'єкту Facebook з боку об'єктів John і Dave. При порівнянні композиційних матриць зміни відбулися щодо об'єкта John і об'єктів MySpace і LiveJournal, при цьому також змінилося ставлення об'єкта Dave до об'єкта MySpace.

Таким чином, верифікація відносин об'єктів в рамках кіберпростору може бути зведені до такої формули:

$$W \oplus W^A = V,$$

$$W \oplus W^A \oplus V = 0, \qquad (4.10)$$

де W - вихідна матриця бінарних відносин; WA - матриця бінарних відносин, отриманих в результаті атаки на кіберпростір об'єкта; V - результат бінарного порівняння матриць.

4.2. Аналіз великих даних з використанням алгоритмів машинного навчання.

Аналіз великих даних (Big Data Analysis) - науковий підхід, який дозволяє витягати знання і виявляти приховані залежності в великих обсягах необроблених даних.

Машинне навчання (Machine Learning) - науковий підхід, який дозволяє створювати моделі, які можуть самостійно навчатися на основі вхідних даних.

Основним завданням методів інтелектуального аналізу великих даних є знаходження цікавлять залежностей відносин між змінними, щоб показати різні властивості аналізованого проекту.

Так як обсяг даних великий, досить складно виконати аналіз і отримати необхідну інформацію або знання, використовуючи стандартні методи.

Інтелектуальний аналіз даних може застосовуватися для всіх типів сховищ даних, таких як реляційні бази даних, операційні дані, сховища даних, або навіть звичайні файли.

Сховище даних - це предметно-орієнтована, інтегрована, залежна від часу, неруйнівного сукупність даних. Воно використовується для консолідації великої кількості даних, зібраних з численних джерел та/або протягом тривалого періоду часу, в організовану і узагальнену модель даних. Створення сховища даних включає: 1) очистку даних; 2) перетворення даних;

3) інтеграцію даних. Що само по собі є важливим етапом попередньої обробки для інтелектуального аналізу даних.

У даній роботі буде розглянуто метод інтелектуального аналізу даних, який використовується для пошуку асоціацій між значеннями в транзакционной базі даних загального призначення. Методи асоціацій часто використовуються при аналізі ринку в цільовому маркетингу, визначаючи купівельні шаблони, при крос-аналізі ринку, в Інтернет банкінгу тощо.

4.3. Метод атрибутно-орієнтованого впізнання Інтернет посилань з використанням частотних шаблонів

Основним завданням методів інтелектуального аналізу даних є знаходження цікавлять залежностей відносин між змінними, щоб показати різні властивості аналізованого проекту.

Так як обсяг даних великий, досить складно виконати аналіз і отримати необхідну інформацію або знання, використовуючи стандартні методи.

Інтелектуальний аналіз даних може застосовуватися для всіх типів сховищ даних, таких як реляційні бази даних, операційні дані, сховища даних, або навіть звичайні файли.

Асоціативний аналіз (Link Analysis) - один із напрямів інтелектуального аналізу даних, призначене для виявлення взаємозв'язків між значеннями в общецелевого базі даних об'єктів [12]. Існують два підходи до аналізу асоціацій:

— виявлення асоціацій (association discovery),

— виявлення послідовності (sequence discovery).

Розглянемо підхід виявлення асоціацій. Позначимо асоціацію як A => B, де A називається попередником, а B називається нащадком. Кількість поява конкретної асоціації в базі даних є число транзакцій, які містять дану асоціацію. Дане поняття також відомо, як частота (frequency) або число підтримки (support count) асоціації:

$$Sup(A=>B) = Freq(A \& B). \qquad (4.11)$$

Низький рівень підтримки може означати, що конкретні асоціацію не дуже важливі або це може вказувати на наявність неякісних даних або їх невеликої кількості.

Правила, отримані асоціативними алгоритмами при сортуванні даних і підрахунку частоти народження. Таким чином, може бути розрахована підтримка асоціації. Можливість зробити це найбільш ефективним способом і є відмінною рисою алгоритмів асоціації, через експоненційної залежності від числа правил. Більшість алгоритмів представляють отриманий результат у вигляді бази даних правил і підтримок.

Виявлення асоціативних правил для даного набору даних D, включає в себе два основних етапи [13]:

1. знаходження кожного набору елементів з частотою їх спільної зустрічальності, яка повинна бути вище певного мінімального рівня підтримки;
2. знаходження асоціативних правил з найбільш часто зустрічаються наборів. Для кожного часто встречаемого набору елементів f, шукаються всі непусті підмножини f. Після чого для кожного такого підмножини *a*, правило виду *a => (f - a)* створюється для кожного підмножини *a* з *f*, де ставлення підтримки *Sup(f - a)* і *Sup(a)* більше або дорівнює мінімальному рівню підтримки.

Першим кроком знаходження правила асоціації є вибір часто зустрічаються наборів, що є досить ресурсномістким завданням. Саме тому даний напрямок є однією з найбільш популярних тем досліджень в області інтелектуального аналізу даних.

Ще однією корисною функцією асоціативних алгоритмів є можливість вказати ієрархію елементів, для цього важливо правильно вибрати рівень агрегування. Ієрархія елементів забезпечує ефективний спосіб контролювати рівень агрегування і можливість експериментувати на різних його рівнях.

Асоціація або правило порядку описує відносини в конкретній базі даних. Спосіб, яким отримані правила можуть бути використані, не настільки

очевидний. В даному випадку останнє слово залишається за аналітиком, як правильно застосувати і інтерпретувати виявлені шаблони. Таким чином, подальший аналіз і експеримент є необхідними для отримання будь-яких вигод від асоціативних правил.

З причини вищесказаного використання графічних методів також може бути дуже корисним для візуалізації структури даних. З метою зробити подання ще більш зрозумілим, асоціації можуть бути відображені на проблемні області, щоб підкреслити більш важливі відносини при аналізі атрибутів.

Методи аналізу даних не можуть дати однозначної відповіді про те, чи є дана Інтернет посилання шкідливої чи ні, тобто носить імовірнісний характер, проте це дозволить повідомити користувача про потенційну загрозу переходу за посиланням.

4.3.1. Формальна модель

Ключовими складовими інтелектуального аналізу даних є: дані, модель, системна логіка генерації моделі, повноцінність створеної моделі.

Метою опису формальної моделі є визначення базових принципів роботи алгоритмів інтелектуального аналізу даних з метою знаходження асоціативних правил.

Основні положення в теорії асоціативних правил.

Таблиця відношень – уявлення знання K: V → C, яке представлено атрибутами домену множини об'єктів, де V – набір об'єктів, і C – глобальний атрибут домену. Запишемо відносну таблицю як:

$$K = (V, A), \qquad (4.1)2$$

де K – таблиця, V - безліч об'єктів, і A = {A1, A2 ... An} – набір атрибутів.

В апараті асоціативних правил існують два основних критерії: підтримка і довіра, причому підтримка є основною. Іншими словами, висока частота народження атрибутів даних важливіша, ніж імплікація. Критерій

підтримки визначає частотні шаблони, асоціативні правила або просто асоціації.

Визначимося з поняттями, які будемо використовувати в описі формальної моделі:

— елемент *e* – значення атрибута;

— набор елементів *q* – кортеж довжини q або q-кортеж;

— *q-кортеж* є поширеним або q-асоціацією, якщо його поява перевищує або дорівнює граничному значенню підтримки;

— q-асоціація – частотний шаблон, який може мати і інші форми;

— всі атрибути повинні бути відмінними один від одного (неізоморфних) [14].

Асоціації (часто зустрічаються набори) і узагальнені асоціації (асоціації в таблиці з узагальненнями на основі атрибутів (Attribute Oriented Generalization - AOG)) часто використовувані поняття для визначення частотних шаблонів. Іншими словами, асоціація є кон'юнкція значень атрибутів, які мають високу підтримку. Відзначимо, що деякі диз'юнкції значень атрибутів дають нові атрибути. Тому узагальнені асоціації є поєднанням цих нових атрибутів, які мають високу підтримку [15, 16, 17, 18].

4.3.2. Алгоритмічна модель

Метод заснований на наступних алгоритмах аналізу частотних шаблонів (Frequent Patterns – FP): FP-growth [19] і FP-trees [20].

FP-growth алгоритм використовується для рекурсивного аналізу і виявлення умовних залежностей по частотним деревам (FP-trees).

Частотне дерево є деревовидної структурою даних, що представляє інформацію про зв'язки елементів в базі даних. FP-trees – спеціальний алгоритм побудови таких дерев. Обидва алгоритми реалізують ідею асоціативного аналізу даних.

Спочатку є база даних Інтернет посилань і їх атрибутів, отримана за допомогою WhoIs і GeoIP онлайн сервісів.

FP-growth алгоритм [19] є одним з найшвидших і найбільш популярних алгоритмів для визначення часто зустрічаються наборів елементів. Він працює з деревовидним поданням бази даних транзакцій (FP-tree). Даний формат представлення даних дозволяє скоротити необхідний обсяг пам'яті для зберігання транзакційних даних і підкреслити важливість ієрархічних відносин між елементами наборів (атрибутами).

Основною ідеєю методу FP-growth є рекурсивний пошук часто зустрічаються шаблонів і поділ бази даних. Алгоритм складається з наступних кроків:

1. Для кожного часто встречаемого елемента, будується база імплікативних шаблонів, а потім його частотне дерево;
2. Даний процес повторюється для всіх новостворюваних частотних дерев.

До тих пір, поки кінцеве FP-дерево не буде порожнім, або містити тільки один шлях – по заданому шляху генеруються всі можливі комбінації вхідних в нього шляхів, кожен з яких є частотним патерном.

Метою методу є виявлення частотних шаблонів для шкідливих, фішинг і чистих наборів Інтернет посилань, які в подальшому будуть використані для евристичного детектування невідомих Інтернет посилань.

FP-growth алгоритм, який використовується в даному методі для генерації частотних шаблонів забезпечує відмінну продуктивність, в порівнянні з існуючими аналогами, такими як: Apriori і TreeProjection [20], що позначається на загальній продуктивності методу, враховуючи великий розмір наборів шкідливих, фішингових і шкідливих Інтернет посилань. Для пошуку всіх частотних шаблонів необхідні всього два сканування таблиці: перше сканування необхідно для визначення частоти кожного елемента множини, друге - для побудови частогоних дерева (FP-tree). Далі за допомогою алгоритму FP-Growth рекурсивно скануємо дерево і отримуємо шукані частотні патерни.

Інтернет посилання є шкідливою тоді і тільки тоді, коли по ній завантажуються дані, які фіксуються сканерами як шкідливі.

Інтернет посилання є фішинг-посиланням тоді і тільки тоді, коли по ній завантажуються сторінки, що імітують популярні Інтернет сервіси. Дані посилання публікуються і детектируются репутаційної системою PhishTank [21]. Інтернет посилання є чистою тоді і тільки тоді, коли по ній завантажуються сторінки і дані, що не детектируются як вредносние або фішинг.

Для даного дослідження чисті Інтернет посилання були отримані з репутаційного Інтернет сервісу Alexa.com [22], де надається статистика відвідувань Інтернет сторінок, на підставі якої вебсайт отримує рейтингову оцінку. У вибірці чистих посилань використовувалися тільки вебсайти, які мають високі рейтингові оцінки, тобто з високою кількістю відвідувань.

Для виявлення false negatives, пропущених шкідливих та фішингових посиланнь, використовувалися IDS Suricata [23], Google Safe Browsing [24], та Virustotal [25] сервіси.

Набори Інтернет посилань для виявлення частотних шаблонів і побудови частотних дерев представлені в таблиці 4.1.

Таблиця 4.1 – Набори Інтернет посилань використовуються для тестування

Набір	Кількість посилань в наборі	Джерело, звідки був завантажений набір
Доброякісні	9991	Alexa.com
Фішинг	98054	PhishTank.com
Шкідливі	16294	Lavasoft MAS

Для кожної Інтернет посилання було отримано таку набір атрибутів:

– країна, де знаходиться сервер зі сторінкою або проксі-сервер (за даними GeoIP);

– тип вмісту сторінки (content-type);

– назва реєстратора домену, якщо використовується доменне ім'я сервера, а не IP адресу;

– час життя домену, якщо використовується доменне ім'я сервера (Lifetime - LT).

Рівень підтримки був встановлений 0.005, що означає відсіювання атрибутів, якщо їх частота народження в наборі менше 0.5% від загальної кількості елементів в наборі.

4.4. Метод детектування поліморфних шкідливих програм

Поліморфні шпигунські програми стають все більш поширеними в даний час як метод, щоб перемогти антивірусні сканери. У цій статті ми розглянемо, як поліморфна мутація допомагає запобігти виявленню шкідливого програмного забезпечення шляхом вивчення нещодавно виявленого поліморфного хробака NrgBot / DorkBot. Потім ми розглянемо, як знайти створений поліморфний шпигун.

Творці шкідливих програм постійно шукають нові технології, щоб залишатися на крок попереду дослідників антивірусних програм, щоб уникнути виявлення антивірусними програмами. Метод, який ми будемо обговорювати тут, є часто використовуваним трюком, який широко використовується веб-експлуататами і відомими ботнетами – поліморфізмом на стороні сервера.

Приклади цього методу включають Shiz, Carperb і Nrgbot / Dorkbot. Основною метою цих бекдорів є крадіжка облікових даних для інтернет-банкінгу, торгових платформ і RBS (віддалених банківських послуг).

Після випуску, дуже часто, що нова копія поліморфного шпигунського програмного забезпечення не виявляється більшістю сканерів AV-файлів (рис. 4.3).

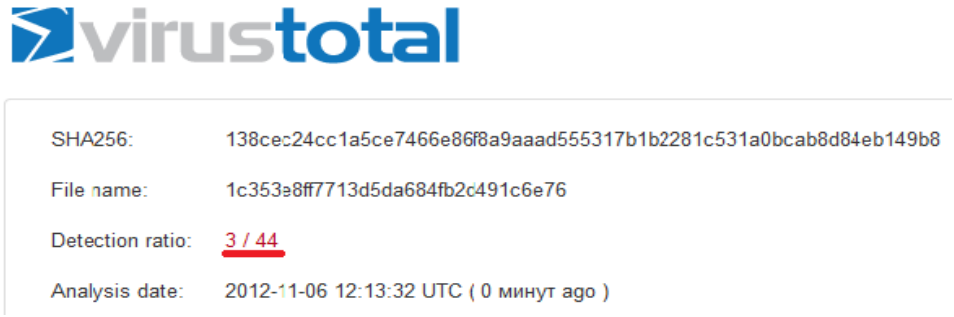


Рисунок 4.3 – Результат сканування VirusTotal нового зразка Nrgbot майже порожній (DrWeb: BackDoor.IRC.NgrBot.146, Fortinet: W32/EncPk.CWP!tr, TrendMicro-HouseCall: TROJ_GEN.RC9H1K6)

Таким чином, він робить виявлення шкідливих програм, створених з використанням поліморфізму на стороні сервера, більш складним для традиційного підходу на основі підпису.

Ідея поліморфного шифрування не є новою і полягає в повторному шифруванні шкідливого файлу на серверах зловмисників кожного разу, коли його запитує інфікована машина бота. Розглянемо схему поліморфної інфекції (рис. 4.4).

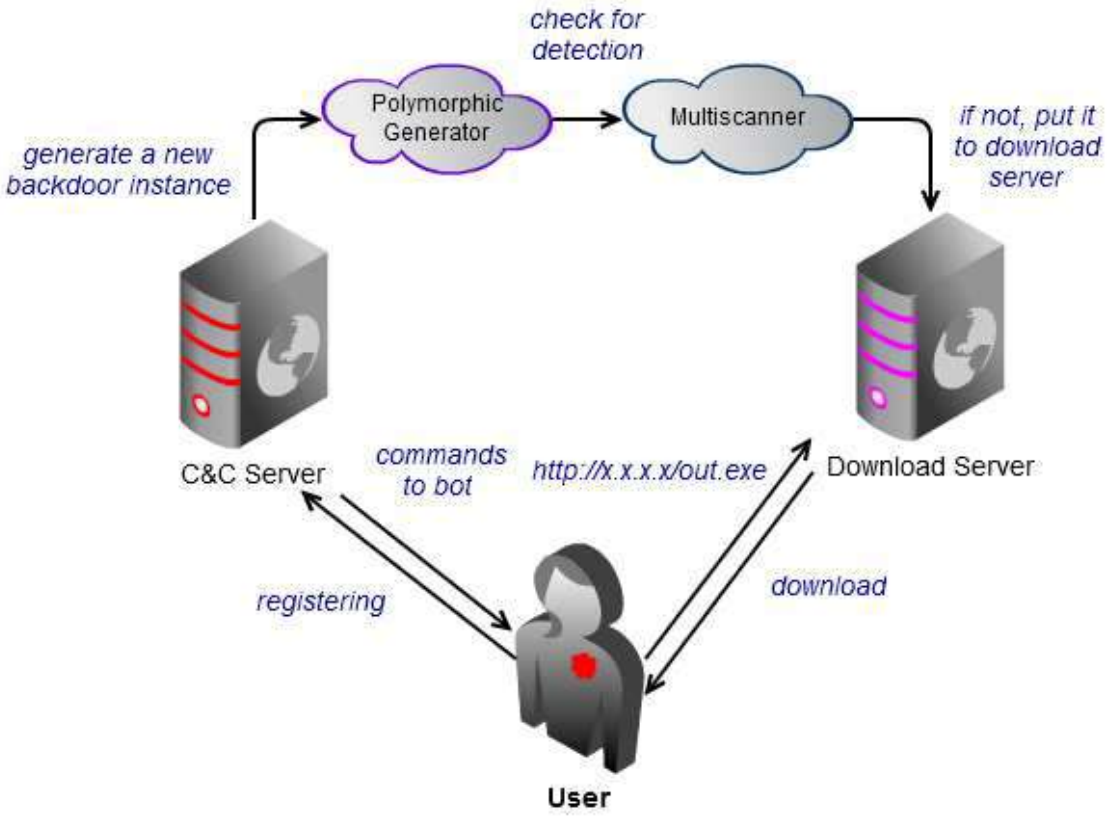


Рисунок 4.4 – Модель поліморфізму на стороні сервера

Після зараження комп'ютер користувача надсилає реєстраційну інформацію на сервер C&C. Потім сервер C&C відповідає набором команд для виконання на комп'ютері жертви.

Новий шматок шкідливого програмного забезпечення генерується "поліморфним генератором", який повторно пакує або повторно шифрує його за допомогою випадково згенерованого ключа. Цей метод гарантує, що зловмисне програмне забезпечення є унікальним, надаючи йому значну перевагу - він ніколи не буде спійманий та проаналізований дослідниками шкідливих

програм. Це значно збільшує ймовірність того, що вона не буде виявлена. Зловмисник може скористатися популярним антивірусним програмним забезпеченням для сканування новоствореної копії, щоб перевірити, що не відбувається виявлення. Незважаючи на те, що файл може бути сканований за допомогою онлайнових служб, таких як VirusTotal, автори шкідливих програм, як правило, не беруть цього маршруту, оскільки зразок буде розподілений серед AV спільноти, що веде до аналізу файлу та додавання до баз даних виявлення. Після того, як копія генерується і перевіряється як не виявлена, вона зберігається на сервері "Завантаження", а посилання передається потерпілому.

Давайте поглянемо на реальний приклад. Після встановлення Nrgbot [26] отримує від C&C URL, щоб оновити себе.

```
PASS smart
KCIK N|UA|XPa|liwoiaq
SSRR liwoiaq 0 0 :liwoiaq

           001| N|UA|XPa|liwoiaq :us, N|UA|XPa|liwoiaq!liwoiaq@[redacted]59.131

           005 N|UA|XPa|liwoiaq

           332 N|UA|XPa|liwoiaq #dpi :!up http://146.185.246.27/out.exe
B379EB791038E522EFDA14A29C7D2BCD -r
           332 N|UA|XPa|liwoiaq #dpi :!j #}
           353 N|UA|XPa|liwoiaq @ #dpi :N|UA|XPa|liwoiaq
.....
SEND #mod smart
SEND #}

           353 N|UA|XPa|liwoiaq @ #mod :N|UA|XPa|liwoiaq
           .....

           353 N|UA|XPa|liwoiaq @ #} :N|UA|XPa|liwoiaq
.....
QUIT :rebooting
```

Рисунок 4.5 – Зв'язок Nrgbot з ботом-сервером

Після цього бот завантажує новий примірник бекдору. Після «оновлення» бекдор стає невидимим для сканерів на основі підписів AV. Більш того, такі бекдори часто блокують доступ до веб-сайтів AV, зупиняючи додаток безпеки користувача від завантаження нових оновлень бази даних виявлення.

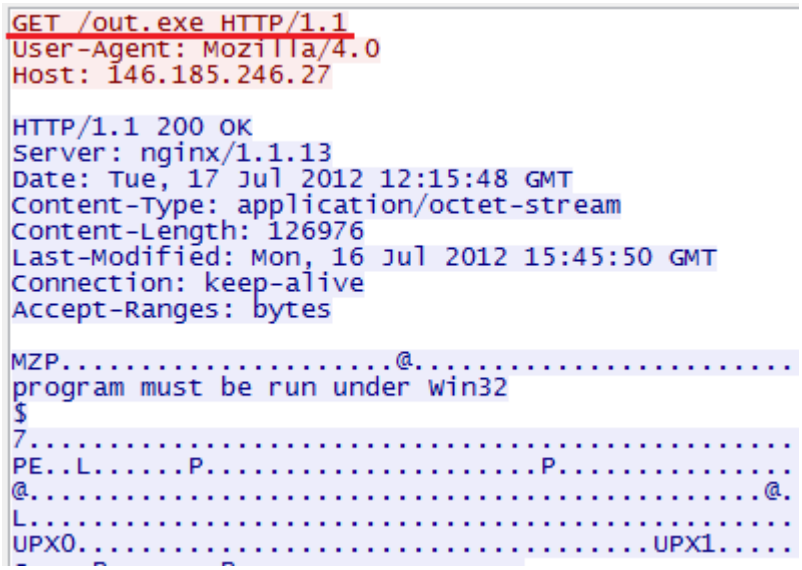


Рисунок 4.6 – Оновлення Nrgbot

Якщо порівняти два поліморфні екземпляри одного і того ж бекдора, то ми побачимо наступне зображення:

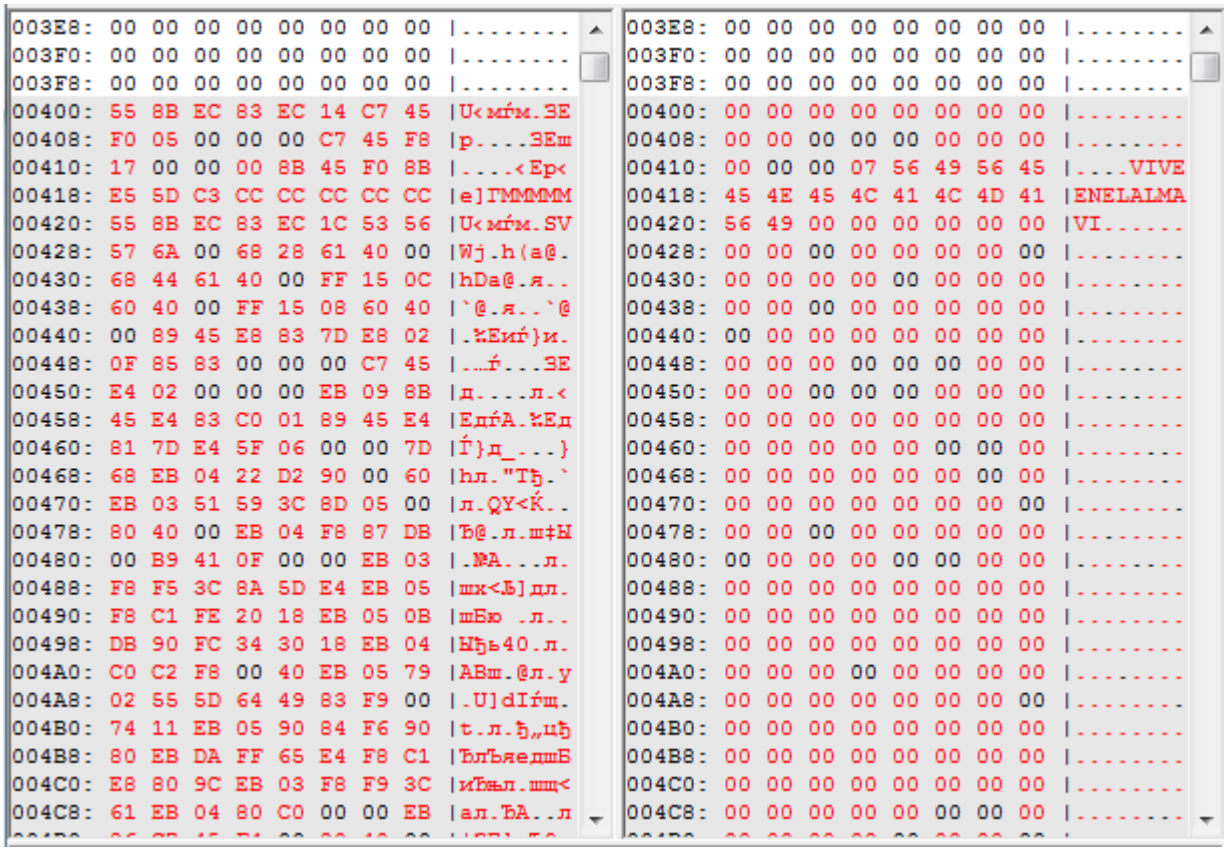


Рисунок 4.7 – Порівняння копій NrgBot

Код і розмір файлу абсолютно різні. Ця різниця може бути досягнута за допомогою поліморфного мутатора. На рисунку показано, що структуру та розмір коду можна змінювати, додаючи нулі та повторно шифруючи дані. Як наслідок, ми бачимо суттєві відмінності у файловій структурі (рис. 4.8).

PE Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	Sectioin MD5
.text	4096	59776	61440	4,08314	5027fb97a60db04070ddc607ab6141f5
.data	65536	9772	4096	0,0	620f0b67a91f7f74151bc5be745b7110
.rsrc	77824	100660	102400	5,07721	5ae8dc0c72763d83c1c2d7cf75422a40

PE Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	Sectioin MD5
.text	4096	17659	17920	4,47736	18402d3b1eff468b3ff381ba732df8c7
.rdata	24576	8130	8192	3,28484	7407710f75d3232683ba8fe33de827ae
.data	32768	10240	7168	2,55455	59c6173eec2b21c5e6064f7160c12524
.rsrc	45056	57952	58368	5,54193	1757eaa1c0ad64e602ca1d659ecac60b
.reloc	106496	94208	20992	5,48767	63bbba5a7ca3e2a8d356c86b322baa3e

Рисунок 4.8 – PE структури двох копій NrgBot (MD5: ee66a7139bce6a4f9cab1e8d368cd287, MD5: fe6364de90e740b2db420940866204f8)

Однак, якщо ми запустимо обидва зразка в пісочниці і подивимося на код, введений в системні процеси, ми побачимо майже ідентичні дані (рис. 4.9).

Незважаючи на значні відмінності у вмісті файлів, обидва зразки мають однакову функціональність і корисне навантаження, що відображається в шкідливих ін'єкціях (рис. 6). Якщо антивірусні сканери були здатні запускати зразок у пісочниці або емуляторі під час сканування, вони не будуть обмануті поліморфним шифруванням і негайно зловлять новостворені копії з точним сімейним вердиктом.

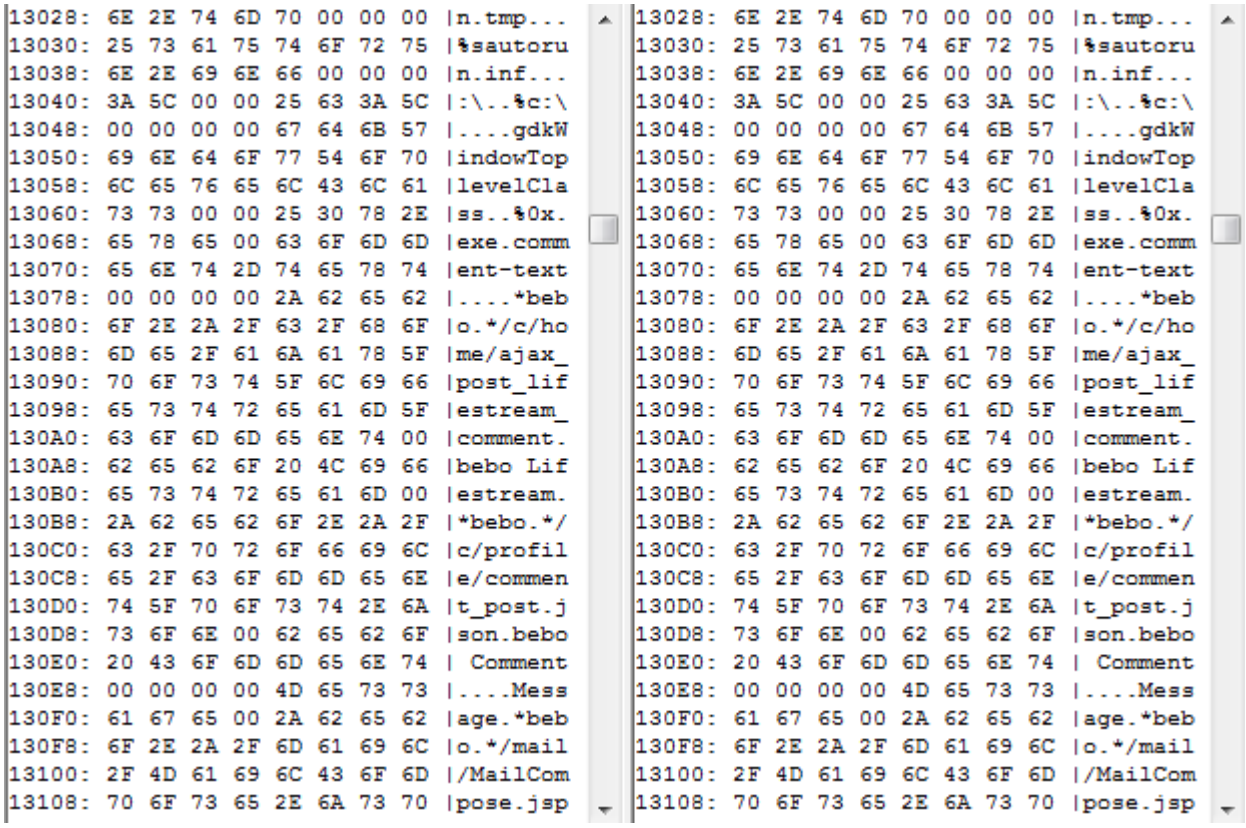


Рисунок 4.9 – Порівняння дампів двох різних ін'єкцій Nrgbot: alg.exe_248_rwx_00A90000_0004E000.dmp та alg.exe_640_rwx_00A90000_0004E000.dmp (319 488 байт)

Опишемо правила Yara, які допоможуть дослідникам шкідливого програмного забезпечення ідентифікувати зразки шкідливого програмного забезпечення Nrgbot / Dorkbot на інфікованій машині.

Щоб знайти унікальні рядки, які використовуються для ідентифікації інфекції, потрібно зробити дамп коду Nrgbot. Дамп вводиться в адресний простір всіх запущених процесів, за винятком системи, smss.exe і lsass.exe.

Нижче наведено приклад пошуку ін'єкції, аналізуючи дескриптори віртуальних адрес (VAD) Explorer.exe [27].

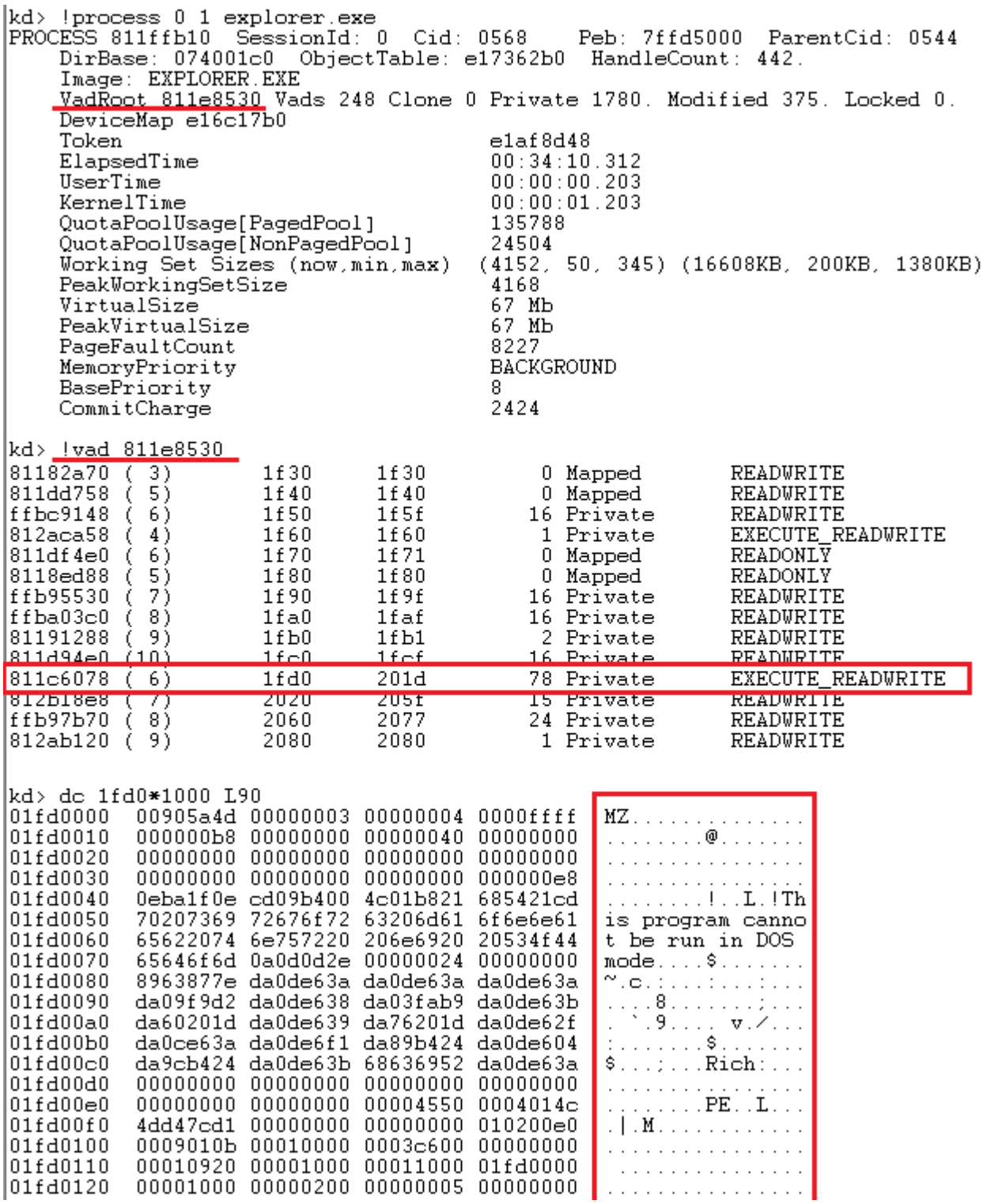


Рисунок 4.10 – Пошук ін'єкцій Nrgbot у Windbg

Крім того, скидання зловмисного коду, вкладеного в процес Explorer.exe, може бути виконане за допомогою PETools.

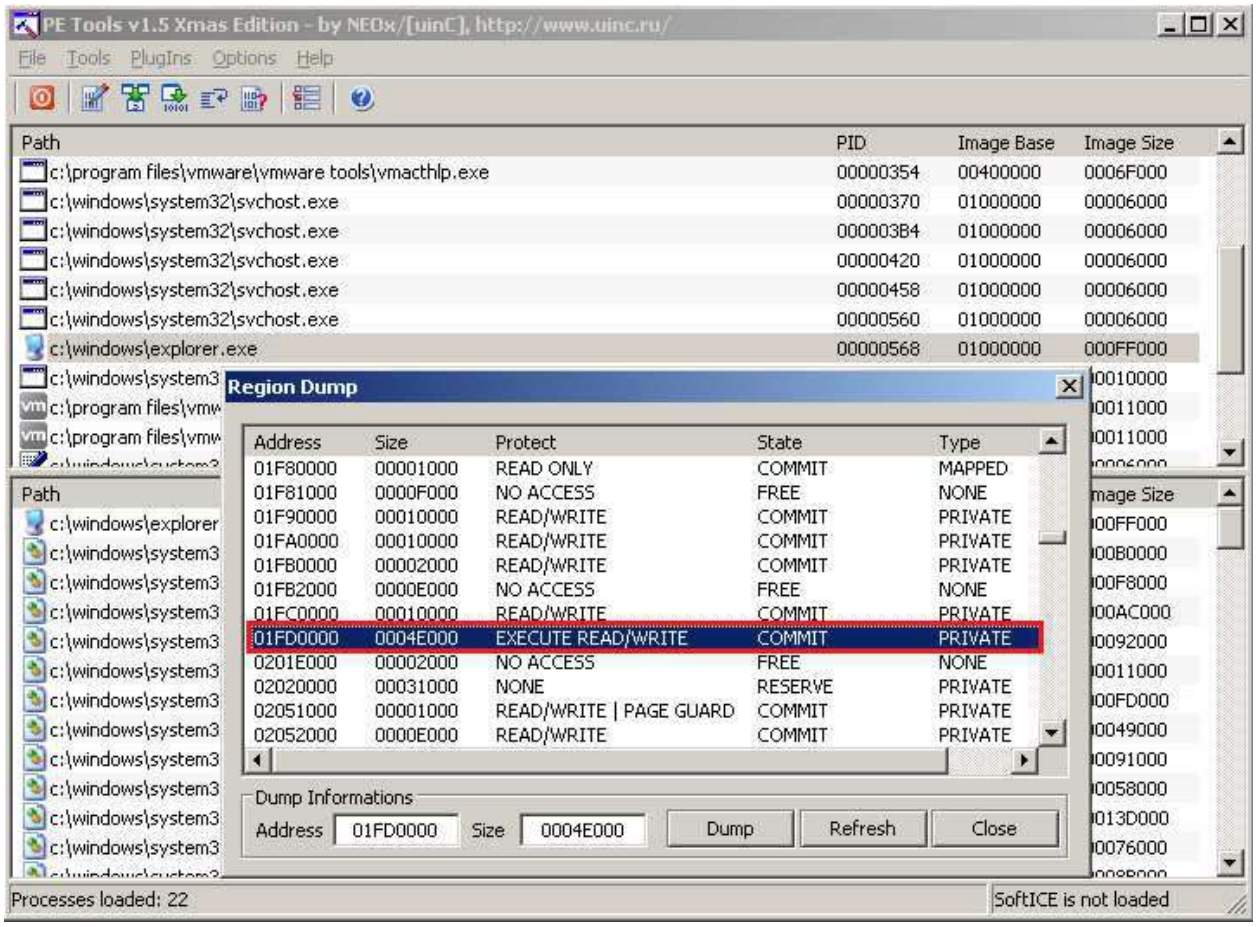


Рисунок 4.11 – Створення дампа Nrgbot з інструментами PE.

Приклад звалища шкідливого коду показано на рис. 4.12.

```
(%s != %s)   dlds    http:// rebooting   [Login]: %s [DNS]: Blocked %d doma
in(s) - Redirected %d domain(s)   [Speed]: Estimated upload speed %d KB/s S
oftware\Microsoft\Windows\CurrentVersion\Run   % s : Z o n e . I d e n t i
f i e r     nqrBot  running IPC_Check   w i n i n e t . d l l   s e c u r 3
 2 . d l l   w s 2 _ 3 2 . d l l     shell\open\command= shell\explore\comm
and=  icon=shell32.dll,7♫◙    useautoplay=1♫◙ action=Open folder to view fi
les♫◙  shellexecute=   [autorun]♫◙ .lnk    %windir%\system32\cmd.exe   &&%%
windir%%\explorer.exe %%cd%%%s  /c "start %%cd%%RECYCLER\%s RECYCLER    ..
```

Рисунок 4.12 – Dump_01FD0000_0004E000.dmp фрагмент ін'єкції

Дамп сканується безкоштовним онлайн-сканером VirusTotal.

Рисунок 4.13 – Результати сканування Nrgbot (Microsoft: Worm:Win32/Dorkbot.A, Norman:W32/Dorkbot.U, Sophos: W32/Dorkbot-L)

Результати сканування показують, що більшість антивірусних програм не можуть виявити Nrgbot у пам'яті. Для спрощення процесу пошуку унікальних рядків дампа, які будуть використовуватися для створення правил Yara, ми будемо використовувати утиліту String. Нижче наведено фрагмент рядків "Dump_013E0000_0004E000.dmp":

virusbuster.nprotect.	athanisqueer.com
gdatasoftware.	ngrBot
virus.	hotshows101.com
precisesecurity.	ngrBot
lavasoft.	77.79.7.246
heck.tc	ngrBot
emsisoft.	#ngr
onlinemalwarescanner.	ngrbot
onecare.live.	b0ss.edu
f-secure.	"I"Z
bullguard.	n1.1.0.0
clamav.	3698d30a
pandasecurity.	CnrBqXhcGileOrwW

sophos.	die
malwarebytes.	msn.set
sunbeltsoftware.	msn.int
norton.	http.set
norman.	http.int
mcafee.	http.inj
symantec	mod
comodo.	mdns
avast.	stats
avira.	speed
avg.	logins
bitdefender.	rs1
eset.	ipconfig.exe
kaspersky.	verclsid.exe
trendmicro.	regedit.exe
iseclab.	rundll32.exe
virscan.	cmd.exe
garyshood.	regsvr32.exe
viruschief.	pidgin.exe

Фрагмент представляє назви інтернет-ресурсів, заблокованих антивірусною програмою, а також унікальний маркер "ngrBot". Використовуючи рядки "Dump_013E0000_0004E000.dmp", створімо правило Yara (рис. 4.14).

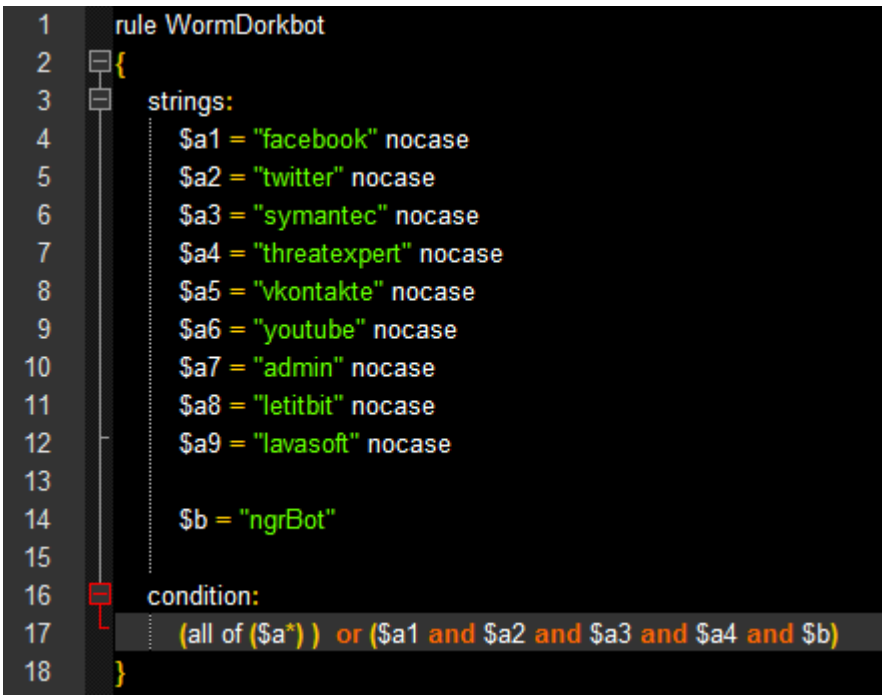


Рисунок 4.14 – Yara правило для Nrgbot

У правилі вище ми перевіряємо всі рядки в $a1-$a9 або тільки "facebook", "twitter", "symantec", "threatexpert" зі стандартним маркером "ngrBot". Як ви можете бачити, рядок "ngrBot" виключається з першої частини нашого стану. Це пов'язано із зразками без підпису "ngrBot" в дампі.

Використовуючи створене правило, скануємо процес Explore.exe PID. Команда для сканування процесу Explore.exe така:

Yara.exe Yara.txt 1544> YaraResult

4.5. Метод пошуку криптопримітивів в троянських програмах-шифрувальниках

Незважаючи на те, що ми спостерішали перехід від крипто-здирництва до криптодобування - нової перспективної області, щоб заробити мільйони доларів США [28] - обганяючи, наприклад, вимогами WannaCry за кількістю інфекцій [29], ще рано говорити про вимирання шифрувальників. CyberSecurity Ventures передбачали, що до 2019 року глобальні втрати від шифрувальників перевищує $11,5 млрд. [30]. Навпаки, криптовалютний ажіотаж допомагає злочинцям ще більше збагачуватись [31].

У січні 2018 року нове сімейство шифрувальників, які працюють по моделі Шифрувальник-як-Сервіс (Ransomware-as-a-Service – RaaS) GandCrab (декриптор для першої версії доступний [32]) показав несподіваний підйом на початку 2018 рока [33] і затьмарив відомих гравців RaaS з 2016 і 2017: Cerber, Locky і Spora [29]. Другу [34] і третю [35] версію шифрувальників GandCrab вже було випущено в травні 2018 року для продовження гри в кіт-мишу з дослідниками безпеки.

У зв'язку з цим, запитання більшості жертв шифрувальників, як правило, це: «Чи можна розшифрувати мої файли без сплати викупу?» Щоб відповісти на нього, по-перше, необхідно з'ясувати, як шифруються файли користувача чи організації. Зокрема:

1. Який шифр використовувався під час атаки?
2. Як виграшник створює ключ (и) шифрування і де зберігає їх для майбутнього розшифрування?
3. І, нарешті, чи можна отримати або створити ключ дешифрування або створити інструмент дешифрування?

Саме тут вступає в дію криптоаналіз шифрувальника. На жаль, такий аналіз вимагає ручної роботи фахівця, що володіє специфічними навичками зворотного проектування (reverse engineering), і може тривати невизначений час [36]. Для того, щоб допомогти дослідникам криптографії, машинне навчання може стати в нагоді.

Загальноприйнятий сигнатурний підхід не може виявити жорстко закодовані шифри, які не використовують Crypto API або криптоконстанти. Наприклад, RC4 не використовує (крипто) константи і, отже, широко використовується авторами шкідливих програм.

В експерименті було використано Krypto ANALyzer (KANAL) для інструменту PEiD [37] та загальнодоступні правила Yara [38, 39, 40] для виявлення криптопримітивів.

Таблиця 4.2 – Виявлення крипто-примітивів у файлах PE та дампів пам'яті викупників з використанням правил PEiD та Yara

Шифрувальник	Сіметричний шифр	Джерело даних	Сигнатурний детект (Yara, KANAL PEiD)
GlobeImposter	AES-256-CBC; RC4, 16-byte key	PE file	List of primes, Big numbers, CryptGenKey import
		Memory dump	List of primes, Big numbers, CryptGenKey import, Rijndael_AES_CHAR, Rijndael_AES_LONG
TeslaCrypt	AES-256-CBC	PE file	N/A
		Memory dump	CryptGenKey import, Big numbers
MoneroPay	Salsa20, 32-byte key	PE file	N/A
		Memory dump	N/A

Таблиця 4.2 містить результати виявлення криптопримітивів у PE файлах шифрувальника і дампах пам'яті. Отримані детекти показують, що тільки в одному випадку KANAL правильно виявив AES у шифрувальнику GlobeImposter. Іншими словами, виявлення на основі підпису не може розглядатися як надійний спосіб ідентифікації жорстко закодованих шифрів у шифрувальниках.

Запропонований спосіб передбачає створення моделі класифікації, яка споживає приклади коду шифру, а потім намагається знайти відомі фрагменти коду в шифрувальниках.

Метод виявлення криптопримітивів включає наступне:

1. Підготовка кодових шаблонів шифрів, представлених в Assembler - бібліотеці зразків.

2. Вихідний код C / C ++ завантажується і компілюється з використанням різних параметрів компіляції, таких як умова виклику: cdecl, stdcall, fastcall; і оптимізація генерації коду: за розміром, продуктивністю, або без них, щоб отримати більшу різноманітність шаблонів коду Assembler для розпізнавання даного шифра. В додаток. Параметр Security Check (/ GS) вимкнено для усунення надлишкового коду, доданого компілятором VC.

3. Нормалізація коду.

4. Заміна імен регістрів процесору та адрес пам’яті, які залежать від конкретного компілятора та структури PE-файлу шаблонами для уніфікації коду Assembler. Наприклад, той же алгоритм, складений двома різними компіляторами з однаковими параметрами генерації коду, може використовувати різні регістри ЦП і адреси для викликів функцій і збереження змінних. Тому, перш ніж порівнювати два асемблерних списки, необхідно замінити всі імена реєстру та адреси, а також видалити коментарі та відповідні роздільники. В іншому випадку значення відстані Левенштейна буде більшим.

5. Локалізація шаблону криптокоду в нормалізованому коді, який відповідає використанню бібліотеки Google Match.

6. Отримання векторів diffs з використанням алгоритму розкладання Myer [41] між кодом шаблону та фрагментами відповідного коду в вимогах до випробування.

7. Розрахунок відстані Левенштейна [42] для векторів diffs, що відображає мінімальний кількість вставлених, видалених або замінених символів, щоб змінити шаблон у відповідний код у вимогах.

Формально для двох рядків S1 і S2 довжини n і m відповідно відстань Левенштейна можна розрахувати, використовуючи наступну формулу повторення:

$D(n, m) = 0$, коли $i = 0, j = 0$;

$D(n, m) = i$, коли $i > 0, j = 0$;

$D(n, m) = j$, коли $i = 0, j > 0$;

$D(n, m) = \min(D(i,j\text{-}1), D(i\text{-}1, j), D(i\text{-}1, j\text{-}1) + 1S_1[i] \neq S_2[j])$, коли $i > 0, j > 0$,

де $i \in \{0,1,..,n\}, j \in \{0,1,\dots,m\}$.

Якщо значення відстані Левенштейна нижче заданого порогу, два фрагменти коду можна розглядати як пов'язані з тим же криптопримітивом. У цьому випадку можна зробити припущення, що даний шифрувальник ви-

користовує шифр, до якого належить шаблон. Поріг може бути отриманий експериментальним шляхом або вказаний аналітиком зловмисних програм.

8. Розпізнаний криптопримітив додається до бібліотеки зразків, яка буде використана пізніше для ідентифікації того ж самого шифру.

4.6. Висновки до розділу 4

Наукова новизна полягає у створенні методів виявлення кіберзагроз, які здатні навчатися на великих даних, з метою виявлення кібератак, які можуть бути реалізовані у вигляді Інтернет посиланнях, поліморфних шкідливих програмах (polymorphic malware), та троянських програмах-шифрувальниках, що займаються здирництвом:

1) Метод атрибутно-орієнтованого впізнання Інтернет посилань з використанням частотних шаблонів, що може провести оцінку атрибутів та відрізнити доброякісні, фішинг, та шкідливі посилання.

2) Метод детектування поліморфних шкідливих програм, що дозволяє виявляти поліморфні шкідливі програми за допомогою аналізу хешів PE секцій та пошуку схожих секцій у бібліотеці шкідливого коду. Після підтвердження детектування PE файла, нові хеші секцій додаються до бібліотеки.

3) Метод пошуку криптопримітивів в троянських програмах-шифрувальниках. Завдяки цьому методу, можливо відповісти які алгоритми шифрування використовував здирник, та чи можливо дешифрувати зашифровані файли користувача чи організації.

Результати розділу подані у роботах [8, 9, 29].

4.7. Список використаних джерел до розділу 4:

1. Wei Hu Jianhua Li Jianjun Shi, A Novel Approach to Cyberspace Security Situation Based on the Vulnerabilities Analysis, Intelligent Control and Automation, WCICA, 2006.
2. Xiaobin, Tan Yong, Zhang Hongsheng, Xi, Multi-Perspective Quantization Model for Cyberspace Security Situation Awareness, Computational Intelligence and Security, 2007.
3. Gandhi, R. Sharma, A. Mahoney and others, Dimensions of Cyber-Attacks: Cultural, Social, Economic, and Political, Technology and Society Magazine, IEEE, Spring 2011.
4. Carl Hewitt, ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing, IEEE Internet Computing, Issue No. 5, pp. 96-99, 2008.
5. Kandukuri, B.R. Paturi, V.R. Rakshit, A., Cloud Security Issues, Services Computing, SCC '09 2009.
6. Cloud Security Alliance (CSA) http://www.cloudsecurityalliance.org/
7. Building Trust from Client to Cloud webinar, Alan Priestley, Intel, June 2011.
8. OpenID, 2011, http://openid.net/
9. Європейська організація з ядерних ДОСЛІДЖЕНЬ (ЦЕРН), http://home.cern/
10. Tim Bell, Smashing Particles, Revolutionizing Medicine and Exploring Origins of the Galaxy, OpenStack Summit 2016, Barcelona
11. Tiecoura Yves, С.В. Чумаченко, В.І. Хаханов, Радіоелектроніка та інформатика, 2011.
12. Introduction to Data Mining and Knowledge Discovery. – Two Crows Corporation. – 1999. – pp. 25-67.
13. Lin T. Y., Ohsuga S., Liau C. J., Hu X., Tsumoto S. Foundations of Data Mining and Knowledge Discovery. – Springer-Verlag Berlin Heidelberg. – 2005. – pp 25-35.

14. T.Y. Lin. Attribute (Feature) Completion – The Theory of Attributes from Data Mining Prospect //Proc. of IEEE International Conference on Data Mining, Maebashi, Japan, Dec 9–12.-2002.-pp. 282–289.
15. T.T. Lee. Algebraic Theory of Relational Databases.-The Bell System Technical Journal Vol 62, No 10.-1983.-pp. 3159–3204.
16. E. Louie, T.Y. Lin. Finding Association Rules using Fast Bit Computation: Machine-Oriented Modeling //Foundations of Intelligent Systems, Z. Ras and S. Ohsuga.-Springer-Verlag, 2000.-pp. 486–494.
17. Y.D. Cai, N. Cercone, and J. Han. Attribute-oriented induction in relational databases," in Knowledge Discovery in Databases, pp. 213–228. AAAI/MIT Press, Cambridge.-MA.-1991.
18. T.Y. Lin, S. Oshuga, C.J. Liau, X. Hu, S. Tsumoto. Foundations of Data Mining and Knowledge Discovery.-Springer.-2005.-pp.34-53.
19. Borgelt C., An Implementation of the FP-growth Algorithm. – OSDM'05, 2005, 135 p.
20. Mao R., Adaptive-FP: An Efficient and Effective Method for Milti-Level Multi-Dimensional Frequent Pattern Mining. – Simon Fraser University. – 2001. – 56 p.
21. PhishTank онлайн сервис. Доступен: http://www.phishtank.com/, январь 2014.
22. Alexa.com онлайн сервис. Доступен: http://www.alexa.com /, январь 2014.
23. IDS Suricata, открытый IDS/IPS/NSM движок. Доступен: http://suricata-ids.org/, январь 2014.
24. Google Safe Browsing, онлайн сервис Google для проверки URL. Доступен: https://developers.google.com/safe-browsing/, январь 2014.
25. VirusTotal, онлайн мультисканнер. Доступен: https://www.virustotal.com/, январь 2014.

26. A. Adamov, Analysis of Nrgbot/Dorkbot worm. Lavasoft/Adaware, https://www.adaware.com/myadaware/malware-descriptions/blog/nrgbot, 2013.
27. Russinovich ME, Solomon DA. Microsoft Windows internals, Microsoft Windows Server(TM) 2003, Windows XP, and Windows 2000 (Pro-Developer). 4th ed. Redmond, WA, USA: Microsoft Press, ISBN 0735619174; 2004.
28. Smominru Monero mining botnet making millions for operators, Proofpoint, January 31, 2018, available at https://www.proofpoint.com/us/threat-insight/post/smominru-monero-mining-botnet-making-millions-operators
29. 2017 Annual Security Roundup: The Paradox of Cyberthreats, TrendMicro, 2017, available at https://documents.trendmicro.com/assets/rpt/rpt-2017-Annual-Security-Roundup-The-Paradox-of-Cyberthreats.pdf
30. Global Ransomware Damage Costs Predicted To Hit $11.5 Billion By 2019, Cybersecurity Ventures, 2018, available at https://cybersecurityventures.com/ransomware-damage-report-2017-part-2/
31. SpriteCoin: Another New CryptoCurrency…or NOT!, Fortinet, January 22, 2018, available at https://www.fortinet.com/blog/threat-research/spritecoin-another-new-cryptocurrency-or-not.html
32. 'No More Project' website, https://www.nomoreransom.org/en/index.html
33. https://twitter.com/WDSecurity/status/968270740549193730
34. Lawrence Abrams, GandCrab Ransomware Version 2 Released With New .Crab Extension & Other Changes, BleepingComputer.com, March 6, 2018.
35. Lawrence Abrams, GandCrab Version 3 Released With Autorun Feature and Desktop Background, BleepingComputer.com, May 4, 2018, available at https://www.bleepingcomputer.com/news/security/gandcrab-version-3-released-with-autorun-feature-and-desktop-background/

36. B. Herzog, Y. Balmas, Great Crypto Failures, Check Point Software Technologies, May 23, 2016, available at https://blog.checkpoint.com/wp-content/uploads/2016/10/GreatCryptoFailuresWhitepaper_Draft2.pdf
37. PEiD Tool website, http://peid.has.it
38. The Yara rules to detect crypto primitives, https://github.com/Yara-Rules/rules/tree/ae82fb6e1e3145a85f52c4856985f7743796aae6/Crypto
39. The Yara rules to detect crypto primitives, https://github.com/x64dbg/yarasigs
40. The Yara rules to detect crypto primitives, https://github.com/polymorf/findcrypt-yara
41. E.W. Myers, An O(ND) Difference Algorithm and Its Variations, Department of Computer Science, University of Arizona, Tucson, US., available at https://neil.fraser.name/writing/diff/myers.pdf

РОЗДІЛ 5
ХМАРНИЙ СЕРВІС ML-АНАЛІЗУ ВЕЛИКИХ ДАНИХ

Пропонується хмарний сервіс для виявлення кібератак на основі аналізу великих даних, який включає три основних компоненти: 1) Сервер, керуючий потоком вхідних і вихідних даних про кібереатаки. 2) Мультисканер з препроцесором для статичного аналізу. 3) Пісочниця (Sandbox), яка предназанчена для автоматизованого запуску шкідливого коду з метою проведення динамічного аналізу.

5.1. Архітектура хмарного сервісу

Пісочниця – ядро системи аналізу, призначене для динамічного аналізу зразків в ізольованому середовищі.

Мета – запуск невиявлених Multiscanner зразків і збір інформації про їх активності.

Пісочниця побудована на технології VMWare для запуску декількох екземплярів Windows XP SP2. Кожен екземпляр включає в себе наступні модулі аналізу: Dumper – зберігає дампи нових процесів в системі і дампи процесів в процесах (рис. 5.1, 5.2).

Для реалізації сервісу була створена інфраструктура на основі гипервизора VMWare ESXi (рис. 5.3).

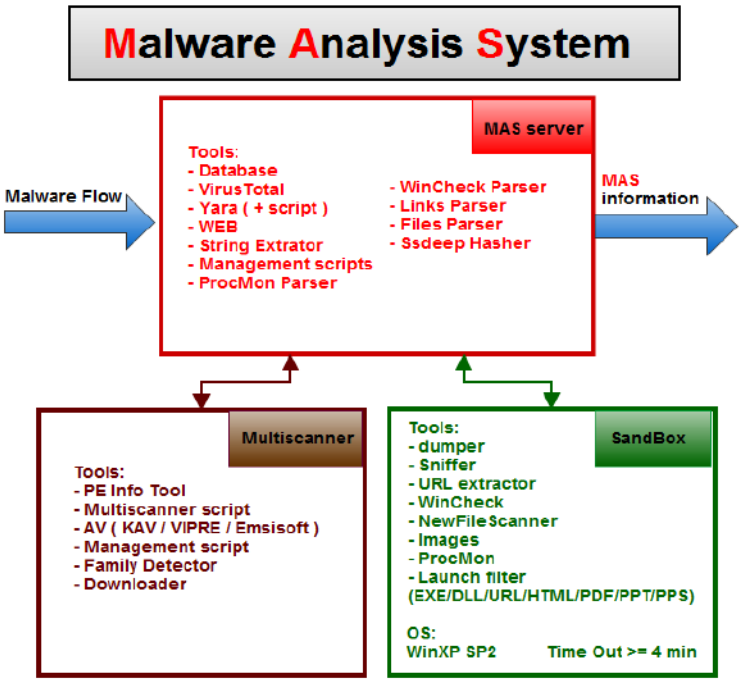


Рисунок 5.1 – Основні компоненти хмарного сервісу

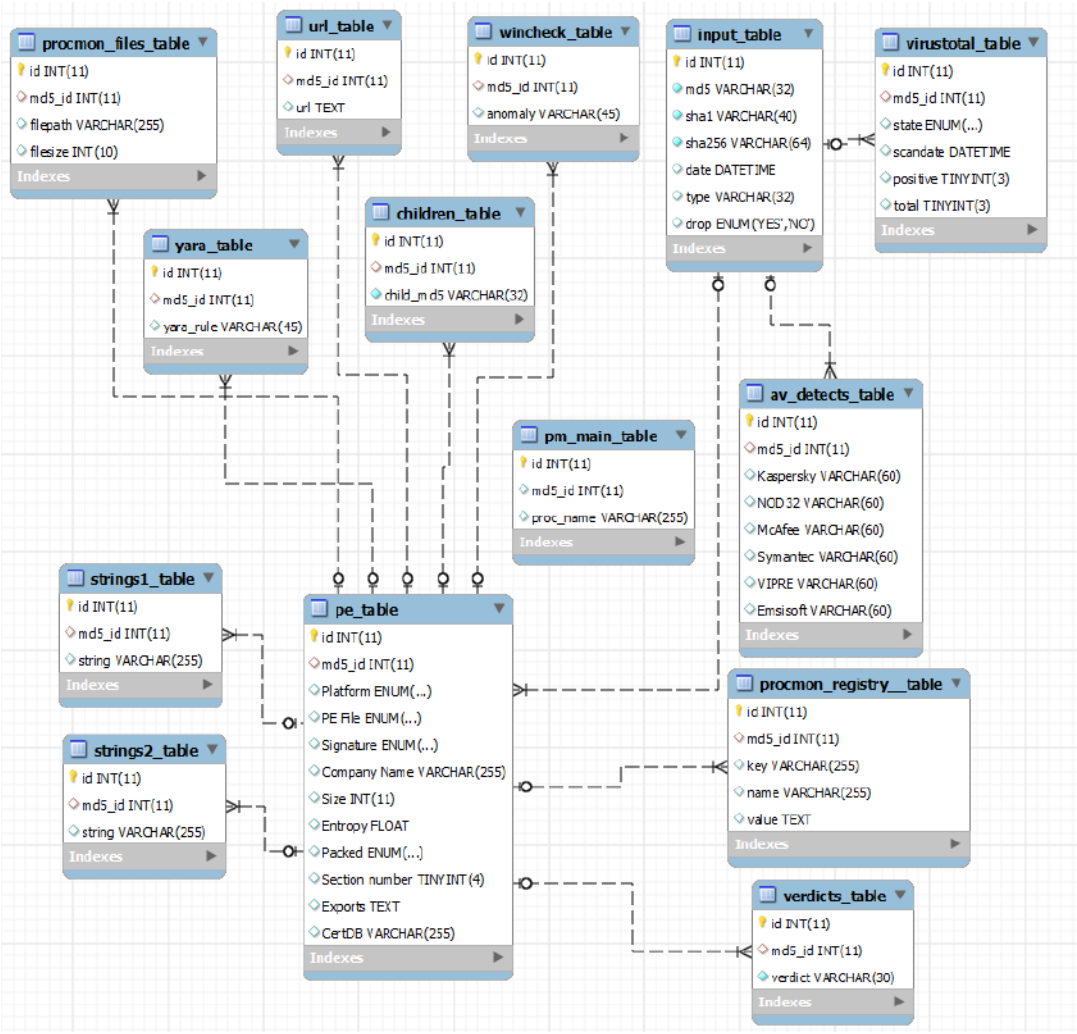


Рисунок 5.2 – Database Architecture

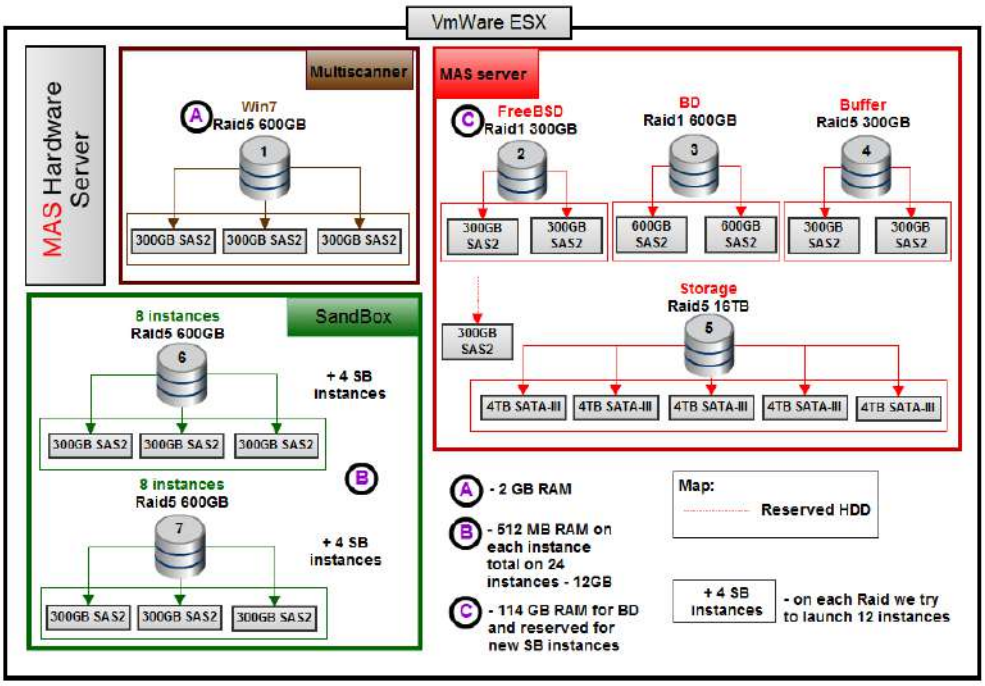


Рисунок 5.3 – Інфраструктура VMWare ESXi

5.2. Функціональна модель сервісу

Сервер вирішує наступні завдання:

– завантаження вхідного потоку в загальну папку GlobalInput;

– підтримка бази даних зразків, завантажених в GlobalInput, і отфільтровка дублікатів;

– зберігання зразків в GlobalOutput, завантажених після попередньої обробки;

– завантаження зразків з папок:

GlobalOutput \ win32 \ {dll, exe, drv, pe32} \ {Suspicious, Unknown}

into Input of Sandbox.

Розібрати Output Sandbox і оновити базу даних по обробленим зразкам:

– збір інформації з Links.txt;

– збір інформації з File.txt;

– збір MD5 відкинутих / завантажених зразків;

– збір інформації з wincheck.txt;

– запуск StringExtractor.exe і збереження рядка, витягнутої з дампа, в сховище;

– збереження виявлень, отриманих після сканування на VirusTotal, в БД;

– зберігати спрацювали правила Yara в БД.

Зберігати скріншоти, якщо вони доступні для зберігання. Аналізовані зразки зберігаються при необхідності.

Розбір пісочниці. Збір всіх файлів з каталогу «files» і відправка в MultiScanner або попередня обробка. Установка бази даних для повторного відправлення зразків на обробку.

Запуск веб-сервера.

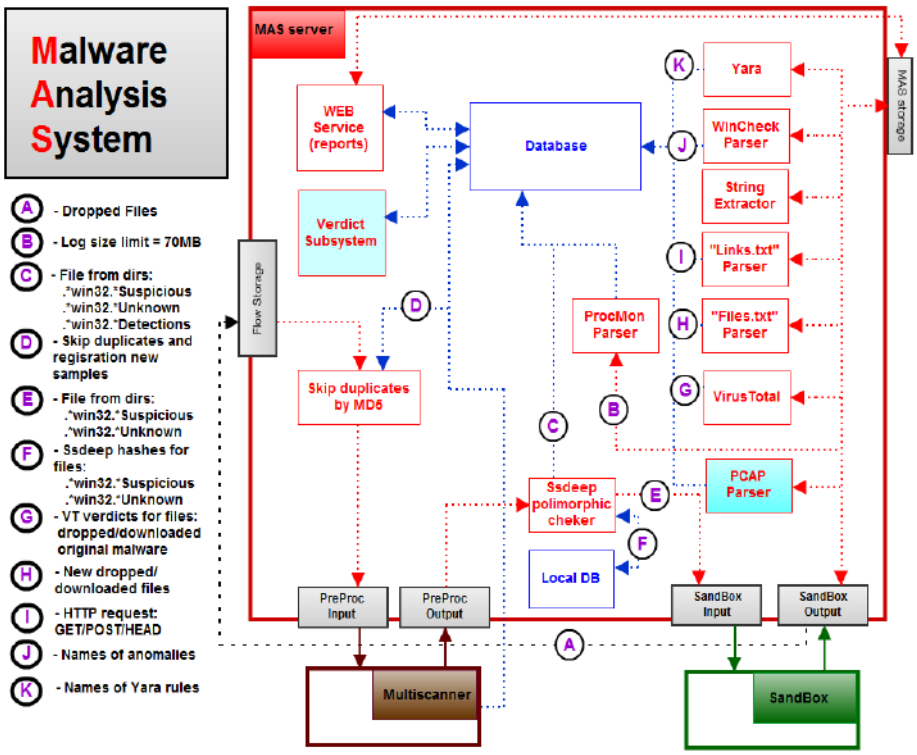


Рисунок 5.4 – Data flow model of MAS server

Мультисканер вирішує наступні завдання.

Препроцесинг. Отримання з папки GlobalInput на файловому сервері зразків, обробка їх і розміщення результату в локальному каталозі «Output».

Перевірка локального «Output» KAV, створення папки «Detected» в одній папці і переміщення туди всіх виявлених файлів, наприклад:

Output \ win32 \ exe \ detected

Output \ win32 \ dll \ detected

Переміщення в локальну папку OUTPUT в каталог файлового сервера GlobalOutput.

Збір інформації про семплах, обробка і запис в базу даних на Fileserver.

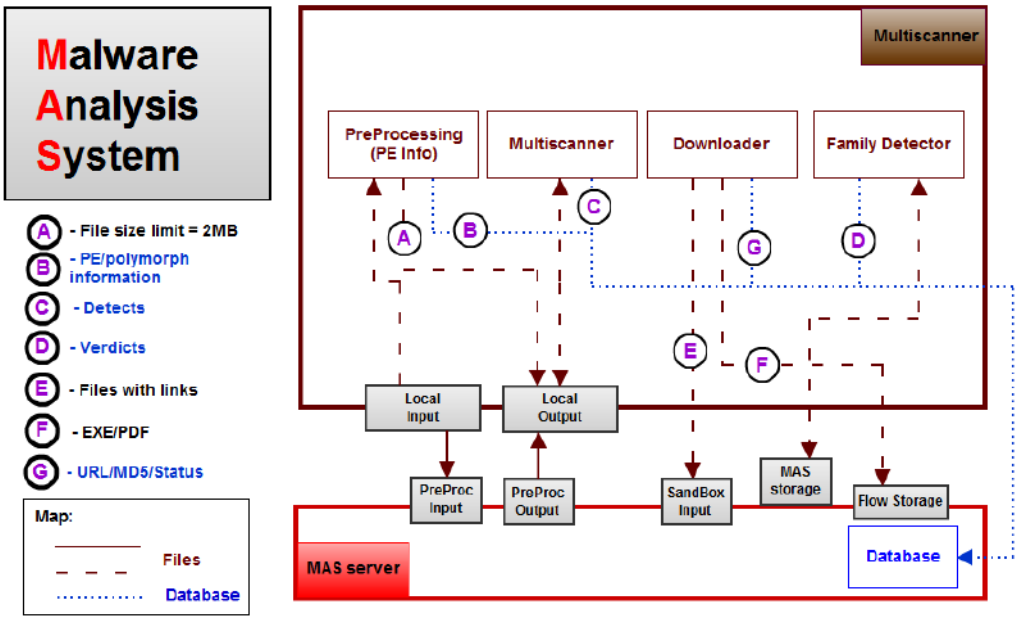
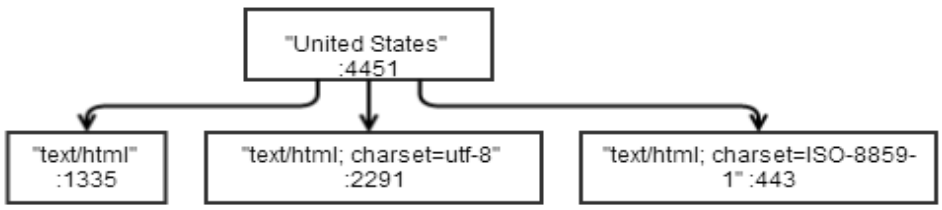


Рисунок 5.5 – Data flow model of multiscanner

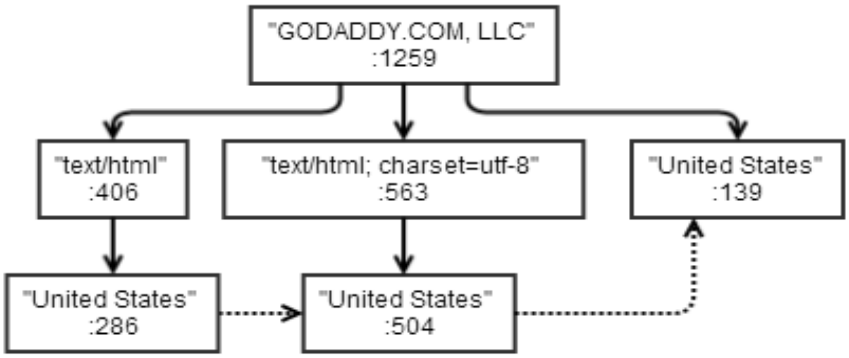
5.3. Результати детектування URL-адрес з використанням частотних шаблонів

В результаті роботи алгоритмів FP-Tree і FP-Growth були отримані частотні дерева і що випливають з них частотні шаблони. Розглянемо кілька прикладів на різних наборах Інтернет посилань.

1) Частотні дерева і шаблони для безлічі чистих посилань, що містять більше одного елемента:

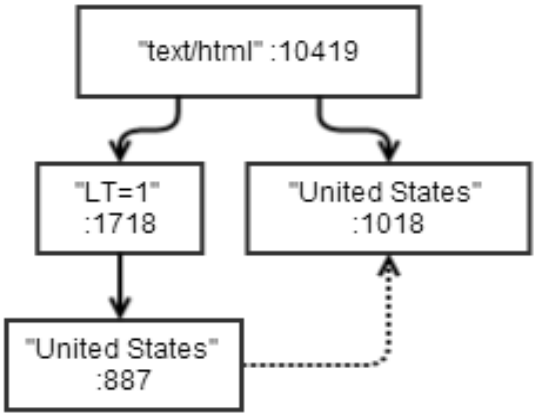


("Text / html", "United States": 1335), ("text / html; charset = utf-8", "United States": 2291), ("text / html; charset = ISO-8859-1" , "United States": 443).

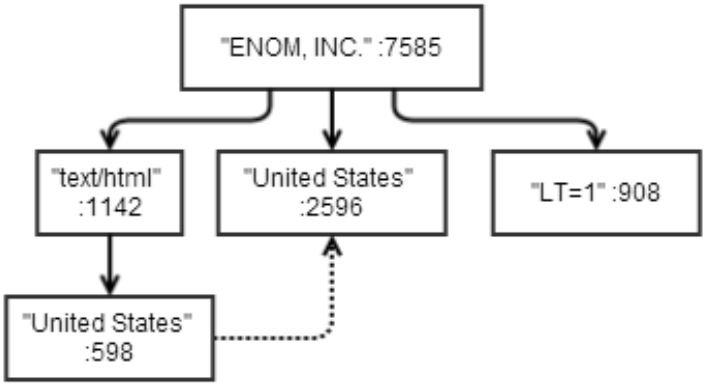


("['GODADDY.COM, LLC']", "text / html; charset = utf-8": 563), ("['GODADDY.COM, LLC']", "text / html": 406), ("['GODADDY.COM, LLC']", "text / html; charset = utf-8", "United States": 504), ("['GODADDY.COM, LLC']", "text / html "," United States ": 286), (" ['GODADDY.COM, LLC'] "," United States ": 929)

2) Частотні дерева і шаблони для безлічі фішинг посилань:

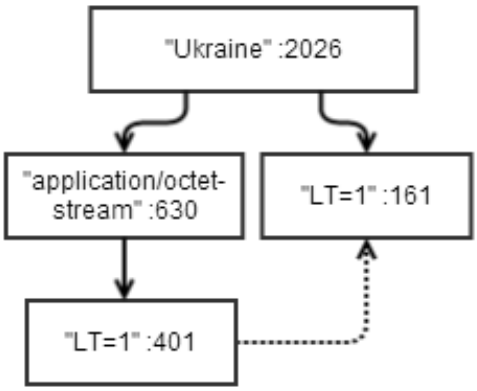


("Text / html", "United States": 4970), ("text / html", "LT = 1", "United States": 887), ("text / html", "LT = 1": 1718)

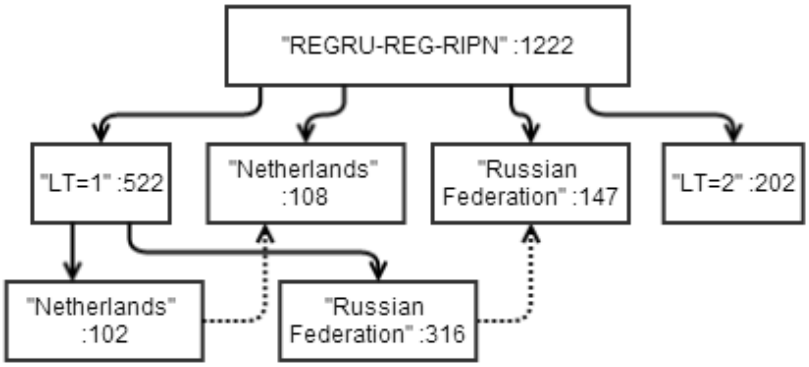


("['ENOM, INC.']", "Text / html", "United States": 598), ("['ENOM, INC.']", "United States": 3194), ("[' ENOM, INC. '] "," LT = 1 ": 908), (" [' ENOM, INC. '] "," text / html ": 1142 Отримати)

3) Частотні дерева і шаблони для безлічі шкідливих посилань:



("Ukraine", "application / octet-stream": 630), ("Ukraine", "LT = 1": 562), ("Ukraine", "application / octet-stream", "LT = 1": 401)



("['REGRU-REG-RIPN']", "LT = 1": 522), ("['REGRU-REG-RIPN']", "Russian Federation", "LT = 1": 316), ("['REGRU-REG-RIPN']", "Netherlands", "LT = 1": 102), ("['REGRU-REG-RIPN']", "LT = 2": 202), ("['REGRU-REG-RIPN'] "," Russian Federation ": 463), (" ['REGRU-REG-RIPN'] "," Netherlands ": 210)

В результаті роботи реалізованого методу Frequent Pattern Analysis (FPA) в рамках середовища Lavasoft MAS для детектування шкідливих і фішинг посилань "in-the-wild" (з «дикої» середовища мережі Інтернет) були отримані наступні результати.

У таблицях 5.1 – 5.3 наведені результати за наступними категоріями соответсвенно:

– кількість заблокованих посилань, які були визначені як шкідливі даним методом (Malicious);

– кількість помилкових спрацьовувань (FP – False Positives), тобто чиста посилання була визначена як шкідлива;

– кількість незадетектірованних шкідливих посилань даним методом (FN – False Negatives), але задетектірованние GSB і IDS.

Були використані сторонні (безкоштовні) сканери IDS Suricata (надалі IDS) [1] і Google Safe Browsing (надалі GSB) [2]. Для верифікації помилкових спрацьовувань використовувався онлайн мультисканер VirusTotal [3]. Використання тристоронніх URL сканерів, дозволило порівняти результати детектування з використанням розробленого методу з іншими сканерами, а також провести оцінку кожного з сканерів в порівнянні з двома іншими (табл. 5.1).

Таблиця 5.1 – Статистика детектування шкідливих посилань

Detection rates	07.2013	08.2013	09.2013	10.2013	Av.,%
FPA method	1112	520	1145	+1247	4,93
IDS Suricata	31	40	34	32	0,16
GSB	89	29	13	202	0,43
Total URLs	19263	22407	22266	19215	

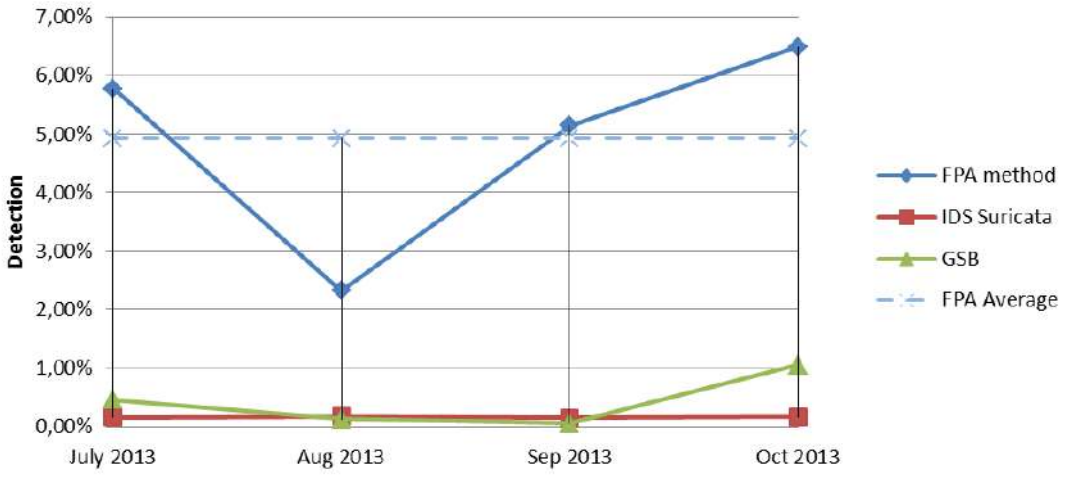


Рисунок 5.6 – Статистика детектування шкідливих посилань з використанням розробленого методу FPA, IDS системи і Google Safe Browsing онлайн сервісу

Таблиця 5.2 – Статистика помилкових спрацьовувань

FP rates	07.2013	08.2013	09.2013	10.2013	Av.,%
FPA method	3	5	15	12	0,04
IDS Suricata	0	0	0	0	0,00
GSB	0	0	0	0	0,00
Total URLs	19263	22407	22266	19215	

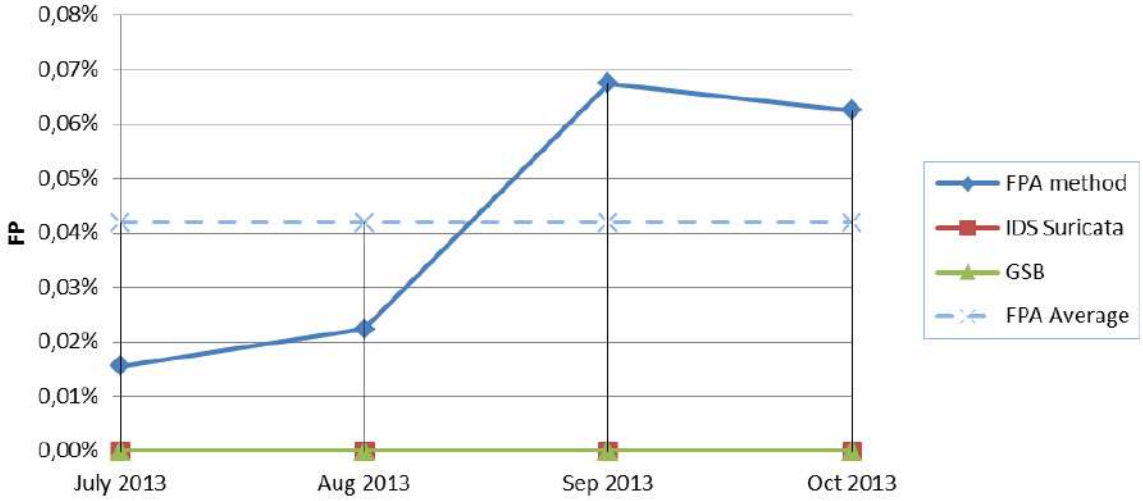


Рисунок 5.7 – Статистика помилкових спрацьовувань з використанням розробленого методу FPA, IDS системи і Google Safe Browsing онлайн сервісу

Таблиця 5.3 – Статистика пропущених шкідливих посилань

FN rates	07.2013	08.2013	09.2013	10.2013	Av.,%
FPA method	120	69	47	234	0,59
IDS Suricata	1 198	544	1143	1437	5,32
GSB	1140	555	1164	1267	5,06
Total URLs	19263	22407	22266	19215	

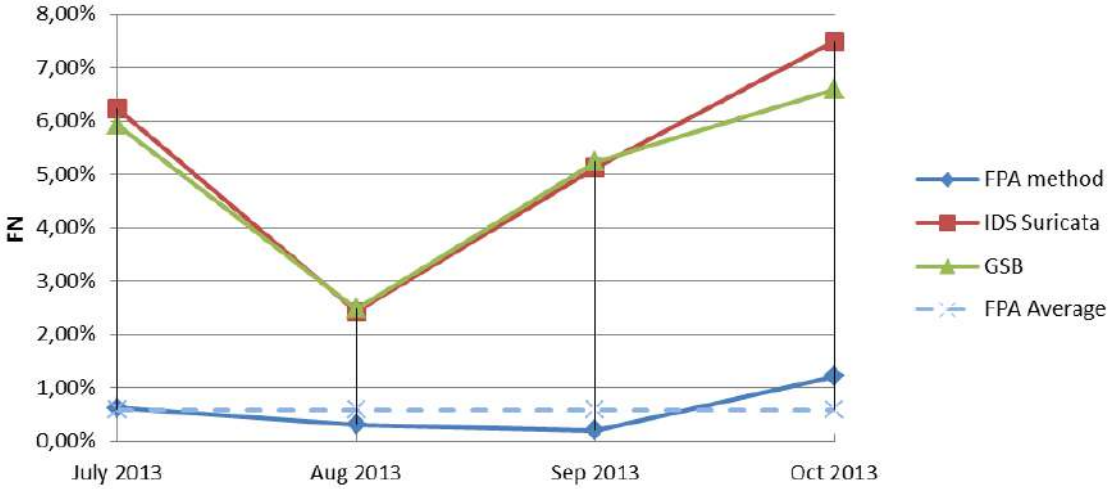


Рисунок 5.8 – Статистика пропущених шкідливих посилань з використанням розробленого методу FPA, IDS системи і Google Safe Browsing онлайн сервісу

5.4. Результати детектування поліморфних шкідливих програм

Використовуючи створене правило, можна просканувати процес Explore.exe по PID. Команда для сканування процесу Explore.exe виглядає наступним чином:

Yara.exe Yara.txt 1544> YaraResult

Результати представлені на рис. 5.9.

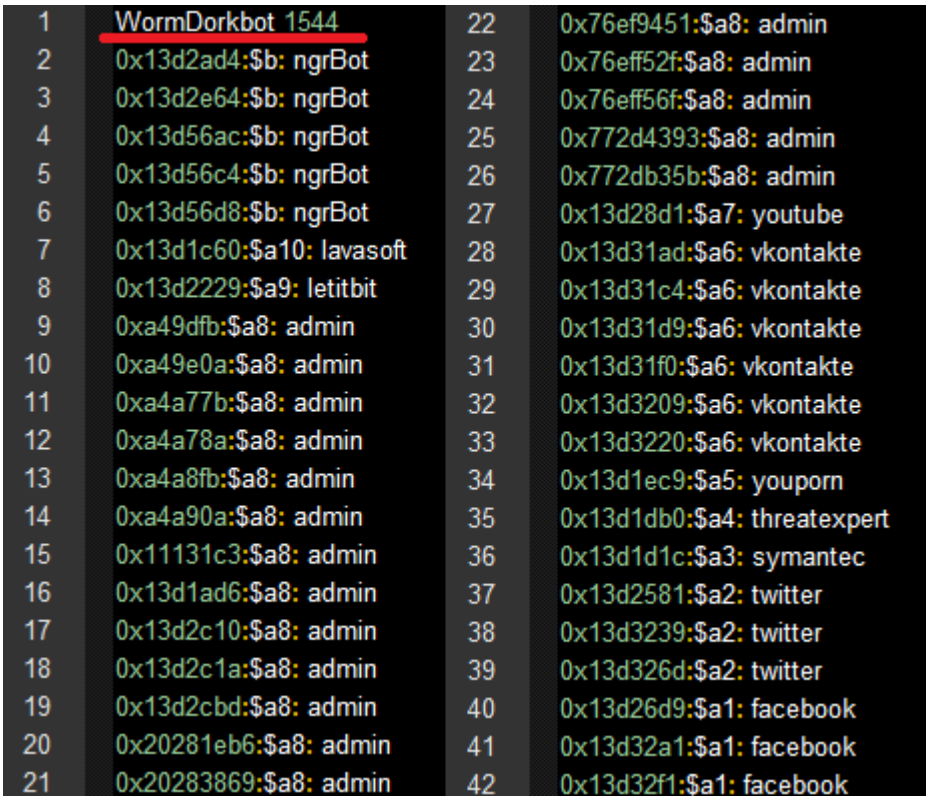


Рисунок 5.9 – Результати сканування процесу Explorer.exe з використанням Yara

Yara успішно виявила шкідливу програму Nrgbot. Програму можна видалити вручну, дотримуючись її опису.

Аналогічне правило можна створити для сімейства шкідливих програм Shiz (рис. 5.10).

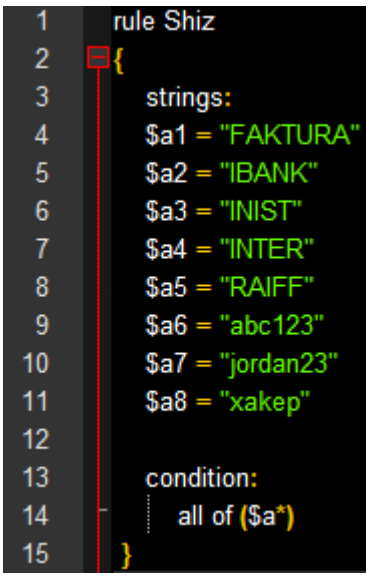


Рисунок 5.10 – Правило Yara для Shiz

Підписи Yara, що застосовуються до унікальних рядків або байтовим послідовностям, узятим з дампів або ін'єкцій шкідливих програм, дозволяють ідентифікувати поліморфні шпигунські програми в системі.

5.5. Результати пошуку кріптопримітивів в троянські програми-шифрувальники

Експерименти проводилися за допомогою бібліотек Google Diff, Match і Patch [4]. Бібліотеки забезпечують інтелектуальне зіставлення фрагментів коду, отримання різницевих векторів з використанням алгоритму відмінностей Майера і обчислення відстані Левенштейна.

Під час експериментів використовувався поріг відповідності за умовчанням, рівний 0,5.

Таблиця 5.4 – Зразки здирників, використані в експериментах

Ransomware	SHA256
TeslaCrypt	9e3827dffc24d1da72cb3d423bddf4cd535fa636062e4ea63421ef327fec56ad
GlobeImposter	a0e5bced56025f875721043df981c400fc28e4efc68ffe42ac665633de085ab1
MoneroPay [17]	ababb37a65af7c8bde0167df101812ca96275c8bc367ee194c61ef3715228ddc

Результати експериментів показують відповідність коду розширення ключа AES в здирників TeslaCrypt і GlobeImposter, коду PRGA RC4 в здирників GlobeImposter і коду кватерфунда Salsa20 в здирників MoneroPay.

Таблиця 5.5 – Розпізнавання AES (розширення ключа) в здирників TeslaCrypt

Iteration No	1	2	3	4	5
Expected location	0	1500	3000	10000	20000
Matched location	115	1473	2986	10006	19953
Levenshtein distance	95	**60**	93	76	75
Correct match in ransomware	FALSE	**TRUE**	FALSE	FALSE	FALSE

Таблиця 5.6 – Розпізнавання AES (розширення ключа) в здирників GlobeImposter

Iteration No	1	2	3	4	5
Expected location	100	1000	4400	10000	20000
Matched location	399	999	4425	9968	19991
Levenshtein distance	61	113	**50**	132	91
Correct match in ransomware	FALSE	FALSE	**TRUE**	FALSE	FALSE

Таблиця 5.7 – Розпізнавання RC4 (PRGA) в здирників GlobeImposter

Iteration No	1	2	3	4	5
Expected location	0	500	800	1000	1500
Matched location	340	340	828	1 063	1553
Levenshtein distance	**20**	**20**	76	75	83
Correct match in ransomware	**TRUE**	**TRUE**	FALSE	FALSE	FALSE

Таблиця 5.8 – Розпізнавання RC4 (PRGA) в здирників GlobeImposter

Iteration No	1	2	3	4	5
Expected location	0	100	1000	1500	3000
Matched location	2	100	1000	1500	3094
Levenshtein distance	**118**	146	177	619	389
Correct match in ransomware	**TRUE**	FALSE	FALSE	FALSE	FALSE

Приклади шаблонів шифрування, сценарію Python, використовуваного для нормалізації коду ASM, і отримані вектори Diffs можна знайти в загальнодоступному репозиторії [5].

Щоб підвищити продуктивність і точність описаного методу розпізнавання образів, рекомендується вказати приблизне місце розташування, де кріптопримітив може з'явитися в цільовому файлі.

Виявлення на основі сигнатур константами може допомогти методу знайти кріптопрімітиви в здирників. Наприклад, для пошуку місця розташування коду, який належить симетричним шифрам AES, Salsa20 і TEA, які мають ці константи.

5.6. Висновки до розділу 5

Висновок за результатами детектування Інтернет посилань на основі аналізу атрибутів з використанням частотних паттернів. Як випливає з наведених вище даних, метод дозволяє визначити 4,93% (таблиця 5.1) посилань як шкідливі на підставі аналізу їх атрибутів із загального числа посилань "in-the-wild". IDS система і GSB сервіс зміг визначити 0,16% і 0,43% посилань як шкідливі відповідно в тому ж наборі.

Середнє число помилкових спрацьовувань у методу FPA вище (таблиця 5.2), ніж IDS і GSB сервісів і состовляет 0,04% (9 посилань) від загального числа посилань. В контексті детектування підозрілих Інтернет посилань даний показник не є критичним, як, наприклад, для додатків, оскільки користувач може зайти на заблоковану сторінку, проігнорувавши попередження системи безпеки.

При цьому середня кількість пропущених шкідливих і фішинг посилань вище у систем IDS і GSB і становить 5,32% і 5,06% відповідно при тому, що FPA метод пропустив всього 0,59% небезпечних посилань (таблиця 5.3). Даний параметр є найбільш істотним в системах блокування шкідливих

і фішинг сайтів, так як дозволяє попередити користувача про потенційну небезпеку при відвідуванні зараженого веб ресурсу.

В результаті порівняльного аналізу методу сигнатурного детектування IDS системи Suricata і методу детектування з використанням чорних списків Google Safe Browsing з магічними методом FPA на основі аналізу атрибутів Інтернет посилань ми отримали значну перевагу методу за кількістю детектіруемих шкідливих і фішинг посилань. При цьому розроблений метод FPA має прийнятний відсоток помилкових спрацьовувань (0,04%), що є незначним недоліком методу на даний момент.

Висновок за результатами детектування поліморфних шкідливих програм. Як було показано вище, технологія поліморфізму може значно захистити нові частини програм-шпигунів від виявлення антивірусом 0 дня, роблячи себе практично невидимим в комп'ютерній системі.

Більш того, після установки поліморфні бекдори можуть запускати процедуру поновлення для завантаження нової версії шпигунського ПЗ, що збільшує термін його служби на зараженому комп'ютері. Подано також спосіб виявлення поліморфних шпигунських програм і то, як цей підхід в основному заснований на динамічному аналізі зразків. Після запуску поліморфна програма-шпигун виявляє свою шкідливу корисне навантаження безпосередньо в пам'яті процесу. Цей метод може бути успішно виявлений за допомогою правил Yara, спеціально створених для родин Nrgbot і Shiz.

Приймаючи цю інформацію до уваги, можна запропонувати описаний метод виявлення численних інфекцій, наприклад, в корпоративної мережі. Використовуючи описані методи, адміністратор або інженер з безпеки може створити правило Yara для певного сімейства шпигунських програм і почати виявляти активне зараження в мережі. Як тільки вірус виявлений, керівництво з видалення може допомогти вилікувати систему.

Висновок за результатами пошуку криптопримітивів. Результати експериментів підтвердили первинну гіпотезу про можливість ідентифікації крипто примітивів в упакованому і деобфускірованному коді вимагачів.

Випадки, представлені в таблицях 5.4 – 5.8, мають правильно розпізнаний фрагмент коду з мінімальним відстанню Левенштейна.

Таким чином, показана можливість ефективного розпізнавання кріптопрімітівов в коді здирників на основі коду асемблера ethalon з симетричних шифрів, які розглядаються як шаблони пошуку.

Запропонований метод успішно виявив криптографічний код симетричних шифрів AES, RC4, Salsa20 в здирників TeslaCrypt, GlobeImposter і MoneroPay. Проте, заснований на сигнатурах підхід, з використанням правил Yara та відкритим вихідним кодом і KANAL в PEiD, показав низькі можливості виявлення.

Описаний спосіб може бути додатково поліпшений, щоб автоматично генерувати правило Yara для виявлення фрагмента коду, що співпав, вимагача, відповідального за симетричне шифрування даних.

Результати розділу відображені у роботах [4, 10, 15, 19, 20, 22, 25, 27, 30].

5.7. Список використаних джерел до розділу 5

1. Система виявлення вторгнень IDS Suricata [https://suricata-ids.org/]

2. Google Safe Browsing [https://safebrowsing.google.com/]

3. Веб-русурс мультісканнера VirusTotal [https://www.virustotal.com/]

4. Сховище бібліотек Google Diff, Match і Patch [https://github.com/google/diff-match-patch]

5. Diffs вектори [https://github.com/AlexanderAda/NioGuardSecurityLab/tree/master/RansomwareAnalysis/DiffMatchPatterns]

ВИСНОВКИ

Проведене дослідження вирішує науково-практичну задачу – введення в інфраструктуру комп’ютингового простору програмної надмірності в формі моделей, методів і програмних додатків для істотного зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування.

Для досягнення поставленої мети – істотного зменшення часу виявлення і блокування кібератак, спрямованих на кіберпростір суб'єкта, шляхом використання розроблених матричних моделей і логічних методів тестування, перевірки та діагностування за рахунок введення обчислювальної надмірності в інфраструктуру кіберпростору, – в роботі були вирішені завдання, які дозволили отримти результати, що мають наукову новизну:

1) *Удосконалені* структурно-логічні моделі і методи перевірки кіберпростору для тестування і діагностування шкідливих компонентів, які відрізняються використанням методу дедуктивного паралельного аналізу обчислювальної системи для перевірки та діагностування malware.

2) *Нові* методи синтезу еталонних логічних схем malware-функціональностей, які характеризуються використанням сигнатурно-кубітних структур, що дає можливість паралельно моделювати malware-driven великі дані для визначення приналежності поточного коду до існуючих деструктивних компонентів в malware бібліотеці.

3) *Нова* модель активного online cyber security комп'ютингу, яка характеризується сигнатурно-кубітним поданням інформації, що дає можливість підвищувати швидкодію процесів моніторингу вхідних потоків malware-даних і управління видаленням деструктивних компонентів.

4) *Нові* методи виявлення кіберзагроз, які здатні навчатися на великих даних, з метою виявлення кібератак, що можуть бути реалізовані у вигляді Інтернет посилань, поліморфних шкідливих програмах (polymorphic mal-ware), та троянських програмах-шифрувальниках, що займаються

здирництвом: метод атрибутно-орієнтованого впізнання Інтернет посилань з використанням частотних шаблонів, що може провести оцінку атрибутів та відрізнити доброякісні, фішинг, та шкідливі посилання; метод детектування поліморфних шкідливих програм, що дозволяє виявляти поліморфні шкідливі програми за допомогою аналізу хешів PE секцій та пошуку схожих секцій у бібліотеці шкідливого коду (після підтвердження детектування PE файла, нові хеші секцій додаються до бібліотеки); метод пошуку криптопримітивів в троянських програмах-шифрувальниках, що дозволяє виявити алгоритми шифрування, використовувані здирником, та можливість дешифрувати зашифровані файли користувача чи організації.

5) *Удосконалені* засоби захисту кіберпростору, які відрізняються використанням моделей і методів сигнатурно-логічного тестування атак, пошуку криптопримітивів в троянських програмах-шифрувальниках на основі використання алгоритмів машинного навчання, що дає можливість істотно зменшити час відновлення працездатності обчислювальної структури.

Практичне значення одержаних результатів досліджень визначається:

6) Тестуванням, верифікацією і впровадженням розроблених програмних засобів перевірки, діагностування шкідливих програм і атак, що дає можливість виконувати їх моделювання із залученням існуючих додатків і malware бібліотек.

7) Програмною реалізацією методу атрибутного-орієнтованого розпізнавання URL-адрес з використанням частотних паттернів, який відрізняється застосуванням апарату інтелектуального аналізу даних, що дає можливість визначати вірогідну оцінку небезпеки URL-адреси на основі його атрибутів.

8) Програмною реалізацією методу перевірки поліморфних шкідливих програм, який відрізняється інваріантністю до детермінізму сигнатур в коді і урахуванням тільки контрольних сум Portable Executable (PE) секцій в виконуваних файлах, що дає можливість поліпшити продуктивність процедур діагностування деструктивних компонентів.

ДОДАТОК А

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Список публікацій здобувача, в яких опубліковані основні наукові результати дисертації:

1. Hahanov V., Gharibi W., Litvinova E., *Adamov A*. Cyber Physical Computing for IoT-driven Services. – New York. USA. – Springer, 2018. 279p. [Chapter 2. Multiprocessor Architecture for Big Data Computing [Text] / V. Hahanov, W. Gharibi, E. Litvinova, *A*. *Adamov*. P. 21-41] (Springer, Scopus).

2. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: Education Challenges and Development of Advanced Malware Analysis Course [Text] / *A. Adamov*. P. 130-134].

3. Carlsson A., Sokolianska I., *Adamov A.* Educating the Next Generation MSc in Cyber Security. – Sweden. – BTH, 2018. – 205 p. [Chapter: The Cloud Threat Landscape [Text] / A. Carlsson, *A. Adamov*. P. 182-186].

4. *Adamov A.* Security risks and modern cyber security technologies for corporate networks [Text] / *A. Adamov*, V. Hahanov, W. Gharibi // Radioelektroniks and informatics. – 2010. – № 4. – P. 31–35. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

5. Хаханов В.И. Инфраструктура анализа и информационной безопасности киберпространства [Текст] / В.И. Хаханов, С.В. Чумаченко, Е.И. Литвинова, А.С. Мищенко, *А.С. Адамов* // Радиоэлектроника и информатика. – 2011. – №2 (53). – С. 40-60. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

6. *Адамов О.С.* Блокчейн інфраструктура для захисту кіберсистем [Текст] / *О.С. Адамов,* В.І. Хаханов, С.В. Чумаченко, В.Г. Абдуллаєв // Радиоэлектроника и информатика. – 2018. – №4 (83). – С. 64-85. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

7. *Адамов О.С.,* Хаханов В.І. Сигнатурно-кубітні методи розпізнавання деструктивних кодів [Текст] / *О.С. Адамов,* В.І. Хаханов // Радиоэлектроника и информатика. – 2019. – №1 (84). – С. 21-39. (Журнал реферується або індексується міжнародними базами Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR, Національною бібліотекою України ім. В. І. Вернадського).

8. *Adamov A.* Analysis and Detection of Polymorphic Spyware [Text] / *A. Adamov*, A. Saprykin // Hakin9 Magazine. 2013. Vol. 8, № 01. Issue 01/2013 (61). Warsow: Software Press, 2013. – P. 6-11.

9. *Adamov A.S.*, Hahanov V.I. A Method for the Attribute-based Detection of URLs Using Frequency Patterns [Text] / Вестник Государственного Инженерного Университета Армении. Серия: Информационные Технологии, Электроника, Радиотехника. 2014. Вып. 17, No2. P. 59-66.

10. Сапрыкин А.С. Методика оценки убытков предприятия от вредоносных программ / Сапрыкин А.С., Бочарникова М.В., *Адамов А.С.* // Вестник национального технического университета "ХПИ" (Новое решение в современных технологиях). 2009. №8. С. 58-64.

Результати, які засвідчують апробацію матеріалів дисертації:

11. *Adamov A.* Electronic System Level Models for Functional Verification of System-on-Chip / *A. Adamov*, K. Mostovaya, Syzonenko, I et al. // Proc. of International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics” (CADSM). – Lviv-Polyana, Ukraine. – February 20-24, 2007. – P. 348–350. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

12. *Adamov A.* Transactional Data Analysis of Electronic System Level Models [Text] / *A. Adamov*, V. Hahanov, D. Melnyk et al. // Proc. of East-West Design and Test Symposium. September 7–10, 2007. Yerevan, Armenia. 2007. – P. 745–748.

13. *Adamov A.* Model of Source Code Analyzer for Hardware Description Languages [Text] / Dmytro Melnyk, Sergei Zaychenko, *Aleksandr Adamov*, Vladimir Hahanov // Proc. of East-West Design and Test Symposium (EWDTS). September 7–10, 2007, Yerevan, Armenia. – Yerevan, 2007. – P. 470–474.

14. Hahanova I.V. Transaction level model of embedded processor for vector-logical analysis [Text] / Hahanova I.V., Obrizan V., *Adamov A.* et al // Proc. of East-West Design and Test Symposium. September 2013. Rostov-on-Don, Russia. – P. 1-4. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

15. *Adamov A.* Data Mining Techniques for Functional Verification of SoC [Text] / *A. Adamov*, R. Hwang, A. Gavrushenko // Proc. of International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008. – Lviv-Polyana, Ukraine. – February 20-24, 2008. – P. 557-559.

16. *Adamov A.* The Problem of Trojan Inclusions in Software and Hardware [Text] / *A. Adamov*, A. Saprykin // Proc. IEEE East-West Design and Test Symposium (EWDTS'2009). – Moscow, Russia. – 18-21 September 2009. – P. 449-451. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. *Adamov A.* The problem of Hardware Trojans detection in System-on-Chip [Text] / *A. Adamov*, A. Saprykin, D. Melnik // Proc. CAD Systems in Microelectronics (CADSM 2009), – Lviv-Polyana, Ukraine. – 24-28 Feb 2009. – P. 178-179. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. *Adamov A.* Security risks in hardware: Implementation and detection problem [Text] / *Adamov A.*, Hahanov V. // Proc. IEEE East-West Design and

Test Symposium (EWDTS'2010), September 17–20, 2010, Saint Petersburg, Russia. – Piscataway, NJ : IEEE, 2010. – P. 425–427. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. *Adamov A.* A security model of individual cyberspace [Text] / *A. Adamov*, V. Hahanov // Proc. IEEE East-West Design and Test Symposium, September 19–20, 2011. – Sevastopol, Ukraine. – 2011. – P. 169–172. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. *Adamov A.* Discovering New Indicators for Botnet Traffic Detection [Text] / *A. Adamov*, V. Hahanov, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov V. Structures for information retrieval in big data [Text] / V. Hahanov, S. Chumachenko, E. Litvinova, *A. Adamov* et al // Proc. of 13th International Conference Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2015), Lviv, Ukraine, 2015. – P. 70-75. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. *Adamov A.* A Sandboxing Method to Protect Cloud Cyberspace [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Test Symposium (EWDTS'2015). – Batumi, Georgia. – September 27-30, 2015. – P. 180–183. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

23. *Adamov A.* Android Ransomware: Turning CryptoLocker into CryptoUnlocker [Text] / *A. Adamov* // Proc. of the 25th Virus Bulletin International Conference, Prague, Czech Republic, 30 Sep-2 Oct 2015 – P. 220-223.

24. *Adamov A.* Detecting targeted attacks in the cloud [Text] / *A. Adamov* // Proc. of the OpenStack Summit, Vancouver, Canada, 18-22 May 2015. 1 p.

25. *Adamov A.* Using Open Source Security Architecture to Defend against Targeted Attacks / Alexander Adamov, Dan Lambright // Proc. of the OpenStack Summit, Austin, TX, US, April 25-29, 2016. 1 p.

26. *Adamov A.* Cloud incident response model [Text] / *Alexander Adamov*, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

27. *Adamov A.* The State of Ransomware. Trends and Mitigation Techniques [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – October 2, 2017, Novy Sad, Serbia. – P. 121–128. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

28. *Adamov A.* Battlefield Ukraine: finding patterns behind summer cyber attacks [Text] / *A. Adamov,* A. Carlsson // Proc. of the 27th Virus Bulletin International Conference, Madrid, Spain, 4-6 Oct 2017 – Appendix: Last-minute presentations. – P. 4-5.

29. *Adamov A.* Artificial Intelligence to Assist with Ransomware Cryptanalysis, [Text] // Proc. of the 28th Virus Bulletin International Conference, Montreal, Canada, 3-5 Oct 2018. – P. 289-292.

30. *Adamov A.* Creating Ransomware Decryptors [Text] // Proc. of 14th Cyber Security Conference UISGCON14, Kyiv, 2018. P. 3.

31. *Адамов А.С.* Модель поиска вредоносного кода в программных продуктах [Текст] / А.С. Адамов, Д.А. Щербин, С.В. Чумаченко // Материалы 16-го Международного молодежного форума "Радиоэлектроника и молодежь в XXI веке", 17-19 апреля 2012. – P. 121–123.

ДОДАТОК Б
АПРОБАЦІЯ РЕЗУЛЬТАТІВ

Результати, які засвідчують апробацію матеріалів дисертації:

1. *Adamov A.* Electronic System Level Models for Functional Verification of System-on-Chip / *A. Adamov*, K. Mostovaya, Syzonenko, I et al. // Proc. of International Conference "The Experience of Designing and Application of CAD Systems in Microelectronics" (CADSM). – Lviv-Polyana, Ukraine. – February 20-24, 2007. – P. 348–350. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

2. *Adamov A.* Transactional Data Analysis of Electronic System Level Models [Text] / *A. Adamov*, V. Hahanov, D. Melnyk et al. // Proc. of East-West Design and Test Symposium. September 7–10, 2007. Yerevan, Armenia. 2007. – P. 745–748. – *очна участь з доповіддю*

3. *Adamov A.* Model of Source Code Analyzer for Hardware Description Languages [Text] / Dmytro Melnyk, Sergei Zaychenko, *Aleksandr Adamov*, Vladimir Hahanov // Proc. of East-West Design and Test Symposium (EWDTS). September 7–10, 2007, Yerevan, Armenia. – Yerevan, 2007. – P. 470–474. – *очна участь з доповіддю*

4. Hahanova I.V. Transaction level model of embedded processor for vector-logical analysis [Text] / Hahanova I.V., Obrizan V., *Adamov A.* et al // Proc. of East-West Design and Test Symposium. September 2013. Rostov-on-Don, Russia. – P. 1-4. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

5. *Adamov A.* Data Mining Techniques for Functional Verification of SoC [Text] / *A. Adamov*, R. Hwang, A. Gavrushenko // Proc. of International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), 2008. – Lviv-Polyana, Ukraine. – February 20-24, 2008. – P. 557-559. – *очна участь з доповіддю*

6. *Adamov A.* The Problem of Trojan Inclusions in Software and Hardware [Text] / *A. Adamov*, A. Saprykin // Proc. IEEE East-West Design and Test Symposium (EWDTS'2009). – Moscow, Russia. – 18-21 September 2009. – P. 449-451. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

7. *Adamov A.* The problem of Hardware Trojans detection in System-on-Chip [Text] / *A. Adamov*, A. Saprykin, D. Melnik // Proc. CAD Systems in Microelectronics (CADSM 2009), – Lviv-Polyana, Ukraine. – 24-28 Feb 2009. – P. 178-179. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

8. *Adamov A.* Security risks in hardware: Implementation and detection problem [Text] / *Adamov A.*, Hahanov V. // Proc. IEEE East-West Design and Test Symposium (EWDTS'2010), September 17–20, 2010, Saint Petersburg, Russia. – Piscataway, NJ : IEEE, 2010. – P. 425–427. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

9. *Adamov A.* A security model of individual cyberspace [Text] / *A. Adamov*, V. Hahanov // Proc. IEEE East-West Design and Test Symposium, September 19–20, 2011. – Sevastopol, Ukraine. – 2011. – P. 169–172. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

10. *Adamov A.* Discovering New Indicators for Botnet Traffic Detection [Text] / *A. Adamov*, V. Hahanov, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

11. Hahanov V. Structures for information retrieval in big data [Text] / V. Hahanov, S. Chumachenko, E. Litvinova, *A. Adamov* et al // Proc. of 13th International Conference Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2015), Lviv, Ukraine, 2015. – P. 70-75.

(Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

12. *Adamov A.* A Sandboxing Method to Protect Cloud Cyberspace [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Test Symposium (EWDTS'2015). – Batumi, Georgia. – September 27-30, 2015. – P. 180–183. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

13. *Adamov A.* Android Ransomware: Turning CryptoLocker into CryptoUnlocker [Text] / *A. Adamov* // Proc. of the 25th Virus Bulletin International Conference, Prague, Czech Republic, 30 Sep-2 Oct 2015 – P. 220-223. – *очна участь з доповіддю*

14. *Adamov A.* Detecting targeted attacks in the cloud [Text] / *A. Adamov* // Proc. of the OpenStack Summit, Vancouver, Canada, 18-22 May 2015. 1 p. – *очна участь з доповіддю*

15. *Adamov A.* Using Open Source Security Architecture to Defend against Targeted Attacks / Alexander Adamov, Dan Lambright // Proc. of the OpenStack Summit, Austin, TX, US, April 25-29, 2016. 1 p. – *очна участь з доповіддю*

16. *Adamov A.* Cloud incident response model [Text] / *Alexander Adamov*, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *заочна участь з доповіддю*

17. *Adamov A.* The State of Ransomware. Trends and Mitigation Techniques [Text] / *A. Adamov*, A. Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – October 2, 2017, Novy Sad, Serbia. – P. 121–128. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore). – *очна участь з доповіддю*

18. *Adamov A.* Battlefield Ukraine: finding patterns behind summer cyber attacks [Text] / *A. Adamov,* A. Carlsson // Proc. of the 27th Virus Bulletin

International Conference, Madrid, Spain, 4-6 Oct 2017 – Appendix: Last-minute presentations. – P. 4-5. – *очна участь з доповіддю*

19. *Adamov A.* Artificial Intelligence to Assist with Ransomware Cryptanalysis, [Text] // Proc. of the 28th Virus Bulletin International Conference, Montreal, Canada, 3-5 Oct 2018. – P. 289-292. – *очна участь з доповіддю*

20. *Adamov A.* Creating Ransomware Decryptors [Text] // Proc. of 14th Cyber Security Conference UISGCON14, Kyiv, 2018. P. 3. – *очна участь з доповіддю*

21. *Адамов А.С.* Модель поиска вредоносного кода в программных продуктах [Текст] / А.С. Адамов, Д.А. Щербин, С.В. Чумаченко // Материалы 16-го Международного молодежного форума "Радиоэлектроника и молодежь в XXI веке", 17-19 апреля 2012. – Р. 121–123. – *очна участь з доповіддю*

ДОДАТОК В
ДОКУМЕНТИ, ЩО ПІДТВЕРДЖУЮТЬ ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЇ

2019-05-29
Karlskrona, Sweden

Statement of research results impact on university education program

Results from the research work "*Cyberspace protection models and methods based on the analysis of big data using machine learning*" by Alexander Adamov, submitted for the degree of candidate of technical sciences in specialty 05.13.05 - computer systems and components in Kharkiv National University of Radio Electronics have been included into the education program MSc. in Computer Security at Blekinge Institute of Technology (BTH), Sweden.

More specifically, Alexander Adamov contributed to development and teaching of the "Malware Analysis" course within the ENGENSEC project 544455-TEMPUS-1-2013-1-SE-TEMPUS-JPCR in the period of 01 Dec 2013 – 30 Nov 2017. This course has been successfully adopted as the official BTH course 'DV2567 Malware Analysis', which is part of the MSc. program in Computer Security. This course is also offered as optional course for Erasmus students visiting BTH.

Alexander Adamov has taught the 'DV2567 Malware Analysis' course several times at the BTH campus in Karlskrona, Sweden. The course has received positive reviews from the students and the other staff involved.

Dr. Dragos Ilie
Assistant Professor in Telecommunication Systems
Blekinge Institute of Technology (BTH)
Dept. of Computer Science
Tel: +46 455 38 58 71
E-mail: dragos.ilie@bth.se

Д О В І Д К А

про можливість впровадження методів та моделей
інфраструктури захисту кіберпростору

Розроблена на кафедрі Автоматизації проектування обчислювальної техніки Харківського національного університету радіоелектроніки за участю (30%) здобувача Адамова Олександра Семеновича архітектура логічного процесора для паралельного моделювання та розпізнавання malware-патернів в потоках великих даних на основі створення інтерпретативних кубітних матричних моделей, методів і архітектур Cyber Security Computing (CSC-комп'ютингу), спрямованого на автоматичний синтез і аналіз логічних схем, орієнтованих на моніторинг і управління CSC-процесами і явищами для усунення деструктивних компонентів в кіберпросторі, що дає можливість на 20% прискорити розпізнавання та усунення деструктивних компонентів malware.

Запропоновано паралельний сігнатурно-кубітний метод моделювання malware-driven великих даних, який характеризується використанням сигнатурного аналізу і кубітними структурами даних, що дає можливість в паралельному режимі визначати приналежність поточного коду до існуючих деструктивних компонентів в malware library.

Достовірність результатів підтверджується проведеними експериментами, тестуванням і верифікацією запропонованих моделей і методів.

Директор Design & Test Lab ОБРІЗАН В.І.

Підпис

Дата

«ЗАТВЕРДЖУЮ»
Перший проректор ХНУРЕ
_____ Рубан І.В.
"___" _____ 2019 р.

АКТ

про впровадження у навчальний процес ХНУРЕ результатів дисертаційної роботи Адамова Олександра Семеновича «Моделі і методи захисту кіберпростору на основі аналізу великих даних з використанням машинного навчання», представленої на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.05 – комп’ютерні системи та компоненти

Комісія у складі зав. каф. АПОТ проф. Чумаченко С.В., доц. каф. АПОТ Шкиля О.С., проф. каф. АПОТ Немченко В.П. розглянула матеріали дисертаційної роботи ст. викладача Адамова О.С., які використовуються в навчальному процесі кафедри АПОТ ХНУРЕ у 2018/2019 навчальному році, і прийшла до наступного висновку.

Розроблені у дисертаційній роботі методи і моделі кіберфізичної інфраструктури захисних сервісів, а саме:

– структурна модель відношень на множині з чотирьох основних компонентів тестування і діагностики (функціональність, кіберсистема, тест, уразливості), яка характеризується повною xor-взаємодією всіх вершин графа і транзитивною оборотністю кожної тріади відношень, що дозволяє визначити і класифікувати шляхи вирішення практичних завдань, включаючи синтез тестів, моделювання та пошук вразливостей;

– вдосконалені методи синтезу тестів для функціональностей, заданих матричними формами опису поведінки компонентів КС, які відрізняються паралелізмом векторних операцій над таблицями, що дає можливість істотно (х2) підвищити швидкодію обчислювальних процедур;

впроваджені у навчальний процес кафедри АПОТ ХНУРЕ та використовуються у таких навчальних дисциплінах:

1. У навчальній дисципліні «Комп'ютерні віруси і засоби боротьби з ними» для бакалаврів напрямку «Комп’ютерна інженерія» у лекційному матеріалі за темою «Захист серверів та робочих станцій» та у лабораторних заняттях за темами «Дослідження вразливостей браузерів та їх реалізації в скрипт технологіях», «Основні ознаки присутності шкідливих програм і методи по усунення наслідків вірусних заражень».

2. У навчальній дисципліні «Комп'ютерні загрози: методи детектування та аналізу» для магістрантів спеціальності 123 «Комп’ютерна інженерія» освітньо-професійної програми «Спеціалізовані комп’ютерні системи» у лекційному матеріалі за темами «Сучасні методи детектування загроз», «Аналіз та детектування хмарних загроз».

Зав. каф. АПОТ проф. Чумаченко С.В.,
Доц. каф. АПОТ Шкиль О.С.,
Проф. каф. АПОТ Немченко В.П.

«ЗАТВЕРДЖУЮ»
Перший проректор ХНУРЕ
_____ Рубан І.В.
"___" _____ 2019 р.

АКТ

про впровадження у навчальний процес ХНУРЕ результатів дисертаційної роботи Адамова Олександра Семеновича «Моделі і методи захисту кіберпростору на основі аналізу великих даних з використанням машинного навчання», представленої на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.05 – комп'ютерні системи та компоненти

Комісія у складі директора Центру навчання студентів іноземною мовою проф. Горбачов В.О., доц. каф. ЕОМ Янковський О.А., доц. каф. ЕОМ Волк М.О. розглянула матеріали дисертаційної роботи ст. викладача Адамова О.С., які використовуються в навчальному процесі кафедри АПОТ ХНУРЕ у 2018/2019 навчальному році, і прийшла до наступного висновку.

Розроблені у дисертаційній роботі методи і моделі кіберфізичної інфраструктури захисних сервісів, а саме:

– паралельний сигнатурно-кубітний метод моделювання malware-driven великих даних, який характеризується використанням сигнатурного аналізу і кубітними структурами даних, що дає можливість в паралельному режимі визначати приналежність поточного коду до існуючих деструктивних компонентів в malware library;

– розроблено сигнатурно-кубітний метод синтезу еталонних логічних схем malware-функціональностей, який відрізняється від аналогів унітарним кодуванням сигнатур для кодів деструктивних компонентів і формуванням кубітних матриць для моделювання, що дає можливість в паралельному режимі визначати наявність malware в обчислювальній системі або мережі;

впроваджені у навчальний процес Центру навчання студентів іноземною мовою ХНУРЕ та використовуються у таких навчальних дисциплінах, що викладаються англійською мовою:

1. У навчальній дисципліні «Information Security in Computer Systems and Networks» для бакалаврів напряму «Комп'ютерна інженерія».

2. У навчальній дисципліні «Technologies for Detecting Threats in Computer Networks» для магістрантів спеціальності 123 «Комп'ютерна інженерія» освітньо-професійної програми «Комп'ютерні системи та мережі».

Директор ЦНСІМ, проф. Горбачов В.О.
Доц. каф. ЕОМ Янковський О.А.
Доц. каф. ЕОМ Волк М.О.

ДОДАТОК Г

ПРОГРАМНИЙ ДОДАТОК

Фрагмент коду інтелектуального модуля детектування загроз хмарного сервісу

```
	// FamiltDetector.cpp : Defines the entry point for the console
application.
	//
	#ifndef _DICTIONARY_H_
	#include "Dictionary.h"
	#endif
	#ifndef _LOADER_H_
	#include "Loader.h"
	#endif
	#include <iostream>
	#include <windows.h>
	#include "TimeGuard/TestTimer.h"
	int main(int argc, char* argv[])
	{
	const char* path = NULL;
	const char* family_folder = NULL;
	//check time
	FileScanner::Means::TestTimer timer;
	timer.start();
	//Retrieve Analyzer object
	Loader* theLoader = Loader::Provide();
	theLoader->runMode = Loader::MODE_DETECT_MAS;
	if (argc > 1 && argc <= 4)
	{
		theLoader->runMode = Loader::MODE_DETECT_LOCAL;
		if(!strcmp(argv[1], "-gen"))
		{
			theLoader->runMode = Loader::MODE_FAMGEN;
		}
		else if(!strcmp(argv[1], "-cluster"))
		{
			theLoader->runMode = Loader::MODE_CLUSTER;
			path = argv[2];
		}
		else if(!strcmp(argv[1], "-storedb"))
		{
			theLoader->runMode = Loader::MODE_DETECT_DB;
			path = argv[2];
		}
		else if(!strcmp(argv[1], "-url"))
		{
			theLoader->runMode = Loader::MODE_URL_REP;
		}
		else if(!strcmp(argv[1], "-desgen"))
		{
			theLoader->runMode = Loader::MODE_DESGEN;
		}
		else
			path = argv[1]; //get file for detection
		if (argc>2)
		{
			family_folder = strstr(argv[argc-1],"-family=");
			if(family_folder)
				family_folder+=8;
		}
	}
	else
	{
		family_folder = theLoader->options_.GetFamilyFolder();
```

```
		path = theLoader->options_.GetInputFolder();

		if(!strlen(family_folder) || !strlen(path) )
		{
			std::cout << "\n1) Using for detection: FamilyDetector.exe <path to input folder> -family=<path to family folder>" << std::endl;
			std::cout << "2) Using for template generation: FamilyDetector.exe -gen -family=<path to family folder>" << std::endl;
			std::cout << "3) Using for clustering: FamilyDetector.exe -cluster <path to input folder>" << std::endl;
			std::cout << "4) Using for storing detects into DB: FamilyDetector.exe -storedb <path to input folder> -family=<path to family folder>" << std::endl;
			std::cout << "5) Using for Description Generator: FamilyDetector.exe -desgen" << std::endl;
			return 0;
		}
	}
	//Try to connect DB
	if((theLoader->runMode == Loader::MODE_DETECT_MAS) || (theLoader->runMode == Loader::MODE_DETECT_DB) || (theLoader->runMode == Loader::MODE_URL_REP) || (theLoader->runMode == Loader::MODE_DESGEN))
	{
		if(!theLoader->ConnectDB())
			return 0;
	}
	//Load all families
	if((theLoader->runMode != Loader::MODE_CLUSTER) && (theLoader->runMode != Loader::MODE_URL_REP) && (theLoader->runMode != Loader::MODE_DESGEN))
	{
		//std::cout << "==========Families loading...==========" << std::endl;
		try
		{
			theLoader->LoadAllFamilies(family_folder);
		}
		catch(...)
		{
			std::cout << "ERROR: Can't load families from " << family_folder << std::endl;
			return 1;
		}
	}
	if(theLoader->runMode == Loader::MODE_DETECT_MAS)
	{
		size_t loop_counter = theLoader->options_.GetLoops(); //launch every
		size_t delay_ms = theLoader->options_.GetDelay()*1000;

		theLoader->InitURLVerdictor();

		while(loop_counter--)
		{
			theLoader->RunDetectionFromDB();
			theLoader->RunURLVerdictor();
			theLoader->RunDesGen();//only for MAS2
			Sleep(delay_ms);
		}
		//Restart application
		try
		{
			HINSTANCE nResult = ShellExecuteA(NULL, NULL, argv[0], NULL, NULL, SW_MINIMIZE);
		}
		catch(...)
		{
			std::cout << "ERROR: Cannot execute new instance of FamilyDetector: " << argv[0] << std::endl;
```

```
		}
	}
	if((theLoader->runMode == Loader::MODE_DETECT_LOCAL) || (theLoader-
>runMode == Loader::MODE_DETECT_DB))
	{
		theLoader->RunDetectionInFolder(path);
	}

	if(theLoader->runMode == Loader::MODE_FAMGEN)
	{
		//Generate template for family
		std::cout << "================Template is
generating...==================" << std::endl;
		try
		{
			theLoader->GenerateFamily(family_folder);
		}
		catch(...)
		{
			std::cout << "ERROR: Error when generating family template"
<< family_folder << std::endl;
			return 1;
		}
	}
	if(theLoader->runMode == Loader::MODE_CLUSTER)
	{
		theLoader->RunClustering(path);
	}
	if (theLoader->runMode == Loader::MODE_URL_REP)
	{
		theLoader->InitURLVerdictor();
		theLoader->RunURLVerdictor();
	}

	if (theLoader->runMode == Loader::MODE_DESGEN)
	{
		theLoader->RunDesGen();
	}
	try
	{
		//clean memory
		theLoader->Clean();
	}
	catch(...)
	{
		std::cout << "ERROR: Error when freeing memory" << std::endl;
		return 1;
	}
	//print time
	timer.stop();
	std::cout<< "Processing time: " <<timer << " ms"<< std::endl;
	return 0;
	}
```