МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЧЕРНІГІВСЬКА ПОЛІТЕХНІКА»

Кваліфікаційна наукова
праця на правах рукопису

БУРМАКА ІВАН АНАТОЛІЙОВИЧ

УДК 004.056.5:004.732(043.5)

ДИСЕРТАЦІЯ

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ВИЯВЛЕННЯ ТА АНАЛІЗУ АНОМАЛЬНИХ ПОДІЙ ДЛЯ ЗАХИСТУ КОМП'ЮТЕРНИХ МЕРЕЖ МАЛИХ ТА СЕРЕДНІХ ПІДПРИЄМСТВ НА ОСНОВІ BLOCKCHAIN

122 – Комп'ютерні науки

12 – Інформаційні технології

галузь знань

Подається на здобуття наукового ступеня доктора філософії

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

_____ Бурмака Іван Анатолійович

підпис

Науковий керівник Дорош Марія Сергіївна
доктор технічних наук, професор

Чернігів – 2024

АНОТАЦІЯ

Бурмака Іван Анатолійович. Інформаційна технологія виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 – «Комп'ютерні науки» (12 – «Інформаційні технології»). – Національний університет «Чернігівська політехніка», МОН України, Чернігів, 2024.

В роботі вирішено актуальне наукове завдання з розробки моделей та методів для інформаційної технології захисту комп'ютерних мереж, з врахуванням особливостей мереж малих та середніх підприємств, які базуються на методах зберігання та поширення інформації на основі blockchain технології. Крім того, важливим є завдання визначення архітектури інформаційної системи виявлення вторгнень для мереж малих та середніх підприємств, яка використовує blockchain компоненти.

Об'єктом дослідження є інформаційні процеси в системах забезпечення захисту від кіберзагроз та аномального трафіку комп'ютерних мереж.

Предметом дослідження було обрано методи, моделі та елементи інформаційної технології колаборативного захисту від кібератак та аномального трафіку для комп'ютерних мереж малих та середніх підприємств на основі blockchain технології.

Метою дисертаційного дослідження є підвищення ефективності захисту комп'ютерних мереж малих та середніх підприємств на основі блокчейн технології. Завдання дослідження полягає в побудові моделі розподіленої системи захисту комп'ютерних мереж на основі blockchain, спираючись на результати аналізу основних загроз для комп'ютерних мереж, зокрема, мереж малих та середніх підприємств.

В основу методології дослідження покладено імітаційне моделювання, UML проектування компонентів блокчейн технології, методи математичного моделювання для визначення оптимальних параметрів блокчейн підсистеми. Методи експертних оцінок використовувались для коректного вибору типових атак та навантажень на атаковані системи при побудові імітаційних моделей. Методи об'єктно-орієнтованого аналізу та функціонального моделювання, зокрема, SADT проектування, використані при концептуалізації бізнес-процесів у нотації IDEF0, які були взяті за основу при проектуванні інформаційної технології виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain.

У вступі обґрунтовано актуальність теми дослідження, сформульовані мета, задачі та методи дослідження та відображено зв'язок дослідження з науковими програмами кафедри, наведено наукову новизну і практичне значення результатів дисертаційної роботи.

У першому розділі виконано аналіз основних загроз інформаційній безпеці комп'ютерних мереж та факторів мережевої безпеки, що впливають на їх захищеність. Було виявлено, що мережі малих та середніх підприємств є більш вразливими до атак і потребують використання засобів захисту, що здатні оперативно виявляти нові, не відомі раніше, атаки та вчасно їх блокувати. Також було проведено аналіз основних методів та засобів захисту комп'ютерних мереж, як комерційних, так і рішень із відкритим вихідним кодом.

У другому розділі запропоновано модель розподіленої інформаційної системи виявлення та аналізу аномальних подій на основі blockchain технології та обґрунтовано необхідність її використання для побудови розподіленої системи виявлення вторгнень для захисту мереж малих та середніх підприємств. Також наведено основні методи класифікації аномальних подій, які легко можуть бути інтегровані у вигляді окремих

модулів у розподілену систему виявлення вторгнень та утворюють комплексний класифікатор для підвищення точності виявлення аномальних подій. Запропоновано метод вибору протоколу консенсусу для blockchain компонента розподіленої системи виявлення вторгнень та наведено адаптований для використання в системах виявлення вторгнень варіант протоколу консенсусу PoS.

У третьому розділі представлено загальну функціональну модель розподіленої системи захисту комп'ютерних мереж на основі blockchain технології, яка визначає основні вхідні, вихідні параметри, обмеження та ресурси з трьома рівнями деталізації. Також наведено архітектуру розподіленої системи захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain, представлену у вигляді UML діаграм.

У четвертому розділі розроблено імітаційну модель розподіленої системи захисту комп'ютерних мереж, призначену для тестування роботи компонентів системи за різних умов та оцінки її швидкодії. Експеримент проведено, як у віртуальному середовищі так і на реальному обладнанні з аналогічними результатами.

Основні результати дослідження та наукова новизна роботи полягають у розробці методів моделей та алгоритмів захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain. На основі аналізу актуальних загроз для комп'ютерних мереж малих та середніх підприємств визначено найбільш ефективні методи та засоби захисту таких мереж із врахуванням особливостей їх функціонування та експлуатації. В роботі запропоновано перелік основних класифікаторів для інформаційної технології захисту комп'ютерних мереж, що можуть бути об'єднані в комплексний класифікатор для підвищення точності виявлення не відомих аномальних подій розподіленою системою виявлення вторгнень. Розроблений метод підвищення ефективності використання ресурсів

blockchain підсистемою оцінює ймовірність успішного створення блоку та дозволяє знизити споживання ресурсів blockchain підсистемою.

Запропонована архітектура розподіленої системи захисту комп'ютерних мереж на основі blockchain дозволяє забезпечити надійний захист мереж малих та середніх підприємств від вторгнень, завдяки використанню даних зібраних багатьма вузлами розподіленої системи з різних мереж. Побудовані UML діаграми основних компонентів розподіленої системи захисту комп'ютерних мереж на основі blockchain для деталізації зв'язків між компонентами системи, та будови самих компонентів системи.

Кросплатформенна реалізація blockchain компонента та класифікатора забезпечує можливість тестування незалежно від апаратної та програмної платформи на широкому спектрі обладнання з підтримкою Unix систем.

Вперше розроблена концептуальна модель розподіленої інформаційної системи виявлення та аналізу аномальних подій в комп'ютерних мережах малих та середніх підприємств, яка на відміну від існуючих містить blockchain компонент для виявлення, накопичення, збереження та спільного використання інформації про аномальні події та блок мультикласифікатора для визначення наявності загрози, що дозволяє підвищити швидкість реагування на невідомі атаки;

Вперше запропоновано метод вибору протоколу консенсусу для розподіленої системи виявлення вторгнень на основі blockchain, який на відміну від існуючих враховує вимоги до обладнання, масштабування та керування учасниками систем виявлення вторгнень в комп'ютерні мережі, що забезпечує підтримку прийняття рішень при проектуванні систем захисту комп'ютерних мереж малих та середніх підприємств.

Удосконалено метод консенсусу PoS blockchain технології, який на відміну від існуючих, використовує в якості значення ставки час роботи вузла в розподіленій системі і дозволяє використовувати blockchain для

децентралізованого зберігання даних розподіленої системи виявлення вторгнень в комп’ютерні мережі малих та середніх підприємств.

Набула подальшого розвитку функціональна модель розподіленої системи захисту комп’ютерних мереж на основі blockchain технології для виявлення, накопичення, збереження та спільного використання інформації про аномальні події, яка визначає основні вхідні, вихідні параметри, обмеження та ресурси з трьома рівнями деталізації та є основою для проектування систем захисту комп’ютерних мереж малих та середніх підприємств.

Практичне значення отриманих результатів полягає в тому, що вони у своїй сукупності утворюють нову інформаційну технологію виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств на основі blockchain. Запропонована інформаційна технологія може бути використана як розробниками систем захисту комп’ютерних мереж, так і мережевими адміністраторами та ІБ спеціалістами малих та середніх підприємств. Розроблені бізнес процеси та архітектура є основою для розробки більш потужних та функціональних розподілених систем виявлення вторгнень.

Ключові слова: блокчейн, інформаційна безпека, система виявлення вторгнень, протокол консенсусу, аномальні події, інформаційна технологія, архітектура.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Burmaka, I., Stoianov, N., Lytvynov, V., Dorosh, M., & Lytvyn, S., "Proof of stake for blockchain based distributed intrusion detecting system," *Dorosh, M., & Lytvyn, S. (2020, August). Proof of Stake for Blockchain Based Distributed Intrusion Detecting System. In Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MOD,* vol. 1265, p. 237, 2020. https://doi.org/10.1007/978-3-030-58124-4_23 (SCOPUS) (0,7 ум. друк. арк.) (Особистий внесок здобувача: модифікація алгоритму Proof of Stake для використання в системі виявлення вторгнень). (0,2 ум. друк. арк.)

2. Burmaka, I., Dorosh, M., Skiter, I., & Lytvyn, S, "Architecture of Distributed Blockchain Based Intrusion Detecting System for SOHO Networks," *Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MODS, 2021 June 28–July 01, Chernihiv, Ukraine. Springer Nature,* pp. 313-326, 2021. https://doi.org/10.1007/978-3-030-89902-8_24 (SCOPUS) (0,85 ум. друк. арк.) (Особистий внесок здобувача: розробка архітектури розподіленої системи виявлення вторгнень на основі блокчейн). (0,36 ум. друк. арк.)
3. Burmaka, I., Zlobin, S., Lytvyn, S., & Nekhai, V., "Detecting flood attacks and abnormal system usage with artificial immune system," *Mathematical Modeling and Simulation of Systems: Selected Papers of 14th International Scientific-Practical Conference, MODS, 2019 June 24-26, Chernihiv, Ukraine,* pp. 131-143, 2019. https://doi.org/10.1007/978-3-030-25741-5_14 (SCOPUS) (0,52 ум. друк. арк.) (Особистий внесок здобувача:

розробка алгоритму виявлення аномальної поведінки комп'ютерної системи за допомогою штучної імуної системи). (0,25 ум. друк. арк.).

4. Skiter, I., Burmaka, I., & Sigayov, A., "Design of Technical Methods for Analysing Network Security Based on Identification of Network Traffic Anomalies," *Information & Security*, vol. 47, no. 3, pp. 306-316, 2020 https://doi.org/10.11610/isij.4722 (0,4 ум. друк. арк.) (Особистий внесок здобувача: проектування атакуючої підсистеми). (0,1 ум. друк. арк.).

5. Burmaka, I. A., Lytvynov, V. V., Skiter, I. S., & Lytvyn, S. V., "Evaluating a blockchain-based network performance for the intrusion detection system" *Математичні машини і системи*, vol. 1, pp. 99-109, 2020. DOI: 10.34121/1028-9763-2020-1-99-109 (0,85 ум. друк. арк.) (Особистий внесок здобувача: Створення моделі розподіленої системи виявлення вторгнень на основі блокчейн). (0,35 ум. друк. арк.).

6. V. Lytvynov, N. Stoianov, I. Stetsenko, I. Skiter, O. Trunova, A. Hrebennyk, V. Nekhai, I. Burmaka. Attacks defense of computer nets by tools using extended information about environment: monograph – Chernihiv: Chernihiv Politechnic National University, 2021. – 212 с. (10 ум. друк. арк.) (Особистий внесок здобувача розділ про побудову тренувального центра з кібербезпеки(. (0,87 ум. друк. арк.)

Наукові праці, які засвідчують апробацію матеріалів дисертації:

7. I. Burmaka, «CONSENSUS ALGORITHM COMPARISON FOR BLOCKCHAIN BASED INTRUSION DETECTING SYSTEM». Безпека ресурсів інформаційних систем: збірник тез I Міжнародної науково-практичної конференції(м. Чернігів 16-17 квітня 2020р.). –Чернігів: НУЧП, 2020. –с.6-14 (0,3 ум. друк. арк.).

8. Бурмака І.А., «КЛАСИФІКАЦІЯ СИСТЕМ ВИЯВЛЕНЯ ВТОРГНЕНЬ В РОЗПОДІЛЕНІ ІНФОРМАЦІЙНІ СИСТЕМИ». Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 17): збірник матеріалів II Міжнародної конференції (25–27 квітня 2017, м. Славутич). – Чернігів: ЧНТУ, 2017. – с. 59-63 (0,12 ум. друк. арк.).

9. Бурмака Іван Анатолійович, «Архітектура розподіленої системи виявлення вторгнень на основі blockchain технології». Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 2020) в режимі онлайн: збірник матеріалів V Міжнародної конференції (27–29 квітня 2020, м. Славутич). – Чернігів : ЧНТУ, 2020. с.54-59 (0,2 ум. друк. арк.).

10. І. А. Бурмака, М. С. Дорош «Оптимізація використання обчислювальних ресурсів розподіленою системою виявлення вторгнень на основі blockchain». Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 21) : збірник матеріалів VI Міжнародної конференції (27–29 квітня 2021,м. Славутич). – Чернігів : НУ «Чернігівська політехніка», 2021. – с. 47-50 (0,15 ум. друк. арк.) (Особистий внесок здобувача метод оптимізації використання обчислювальних ресурсів) (0,07 ум. друк. арк.).

ABSTRACT

Ivan Burmaka. Information technology for the detection and analysis of anomalous events for the protection of computer networks of small and medium-sized enterprises based on blockchain. – Qualifying scientific work on the rights of manuscript.

PhD thesis in Engineering Science under Specialty 122 – "Computer Science" – "Chernihiv Polytechnic" National University, Ministry of Education and Science of Ukraine, Chernihiv, 2024.

The thesis addresses the *current scientific task* of developing models and methods for information technology for protecting computer networks, taking into account the specifics of networks of small and medium-sized enterprises, which are based on methods of information storage and distribution using blockchain technology. Additionally, an important task is to determine the architecture of the intrusion detection information system for small and medium-sized enterprise networks that uses blockchain components.

The object of the research is the information processes in cybersecurity systems and the analysis of anomalous traffic in computer networks.

The subject of the research was selected methods, models, and elements of collaborative information technology for protection against cyber-attacks and anomalous traffic for computer networks of small and medium-sized enterprises based on blockchain technology.

The purpose of the research is to increase the efficiency of protecting computer networks of small and medium-sized enterprises based on blockchain technology. The research task is to build a model of a distributed computer network protection system based on blockchain, relying on the analysis results of the main threats to computer networks, especially networks of small and medium-sized enterprises.

The research methodology is based on simulation modeling, UML design of blockchain technology components, mathematical modeling methods for

determining optimal parameters of the blockchain subsystem. Expert assessment methods were used to select typical attacks and loads on attacked systems when building simulation models. Object-oriented analysis and functional modeling methods, including SADT design, were used in conceptualizing business processes in the IDEF0 notation, which served as the basis for designing information technology for detecting and analyzing anomalous events to protect computer networks of small and medium-sized enterprises based on blockchain.

The introduction justifies the relevance of the research topic, formulates the goal, objectives, and research methods, and reflects the connection of the research with the scientific programs of the department, providing the scientific novelty and practical significance of the thesis results.

The first chapter analyzes the main threats to information security of computer networks and factors of network security affecting their protection. It was found that networks of small and medium-sized enterprises are more vulnerable to attacks and require the use of protective measures capable of quick detecting of new, previously unknown attacks and blocking them. An analysis of the main methods and means of protecting computer networks, both commercial and open-source solutions, was also carried out.

The second chapter proposes a model of a distributed information system for detecting and analyzing anomalous events based on blockchain technology and justifies the need for its use in building a distributed intrusion detection system for the protection of small and medium-sized enterprise computer networks. The main methods of classification of anomalous events are also presented, which can be easily integrated as separate modules into a distributed intrusion detection system and form a comprehensive classifier to improve the accuracy of detecting anomalous events. A method for selecting a consensus protocol for the blockchain component of the distributed intrusion detection system is proposed, and an adapted version of the PoS consensus protocol for use in intrusion detection systems is presented.

The third chapter presents a general functional model of a distributed computer network protection system based on blockchain technology, which defines the main input, output parameters, constraints, and resources with three levels of detail. The architecture of a distributed computer network protection system for small and medium-sized enterprises based on blockchain is also provided, presented in the form of UML diagrams. In the fourth chapter, an emulation model of a distributed computer network protection system is developed, designed to test the operation of system components under different conditions and evaluate its performance. The experiment was made both in a virtual environment and on real equipment with similar results.

The main results of the research and the scientific novelty of the work lie in the development of methods, models, and algorithms for protecting computer networks of small and medium-sized enterprises based on blockchain. Based on the analysis of current threats to computer networks of small and medium-sized enterprises, the most effective methods and means of protecting such networks were determined, taking into account the specifics of their operation. The work proposes a list of main classifiers for information technology for protecting computer networks, which can be combined into a comprehensive classifier to improve the accuracy of detecting unknown anomalous events by the distributed intrusion detection system. The developed method for improving the efficiency of resource utilization by evaluating of the probability of successful block creation, and allows reducing the consumption of resources by the blockchain subsystem.

The proposed architecture of the distributed system for protecting computer networks based on blockchain allows ensuring reliable protection of networks of small and medium-sized enterprises from intrusions by using data collected by multiple nodes of the distributed system from different networks. UML diagrams of the main components of the distributed system for protecting computer networks based on blockchain are built to detail the relations between the components of the system and the construction of the components themselves.

A cross-platform implementation of the blockchain component and classifier provides the ability to test them independently of hardware and software platforms on a wide range of equipment supporting Unix systems.

For the first time, a conceptual model of a distributed information system for detecting and analyzing anomalous events in the computer networks of small and medium-sized enterprises has been developed. Unlike existing models, it includes a blockchain component for detecting, accumulating, storing, and sharing information about anomalous events, as well as a multi-classifier block for determining the presence of threats, which allows for increased response speed to unknown attacks.

A method for selecting a consensus protocol for a distributed intrusion detection system based on blockchain has been proposed for the first time. Unlike existing methods, it takes into account the requirements for equipment, scalability, and management of participants in intrusion detection systems in computer networks, providing support for decision-making when designing protection systems for the computer networks of small and medium-sized enterprises.

The PoS (Proof of Stake) consensus method of blockchain technology has been improved. Unlike existing methods, it uses node uptime in the distributed system as the stake value and allows blockchain to be used for decentralized data storage in the distributed intrusion detection system of small and medium-sized enterprises' computer networks.

The functional model of a distributed computer network protection system based on blockchain technology has been further developed for detecting, accumulating, storing, and sharing information about anomalous events. This model defines the main input and output parameters, constraints, and resources with three levels of detail and serves as the foundation for designing protection systems for the computer networks of small and medium-sized enterprises.

The practical significance of the obtained results is that they collectively form a new information technology for detecting and analyzing anomalous events

to protect the computer networks of small and medium-sized enterprises based on blockchain. The proposed information technology can be used by developers of computer network protection systems, as well as network administrators and IT security specialists of small and medium-sized enterprises. The developed business processes and architecture serve as the basis for developing more powerful and functional distributed intrusion detection systems.

Keywords: blockchain, information security, intrusion detection system, consensus protocol, anomalous events, information technology, architecture.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AAA – Authentication, Authorization, Accounting (Аутентифікація, авторизація та облік)

ASIC – Application-Specific Integrated Circuit (Інтегрована схема, спеціалізована для додатку)

DDOS – Distributed Denial of Service (Розподілена відмова в обслуговуванні)

DMZ – Demilitarized Zone (Демілітаризована зона)

DNS – Domain Name System (Система доменних імен)

DOS – Denial of Service (Відмова в обслуговуванні)

FTP – File Transfer Protocol (Протокол передачі файлів)

HIDS – Host-based Intrusion Detection System (Система виявлення вторгнень на основі хосту)

ICMP – Internet Control Message Protocol (Протокол керування повідомленнями Інтернету)

IDS – Intrusion Detection System (Система виявлення вторгнень)

IP – Internet Protocol (Протокол Інтернету)

IPS – Intrusion Prevention System (Система запобігання вторгненням)

NIDS – Network-based Intrusion Detection System (Система виявлення вторгнень на основі мережі)

PoW – Proof of Work (Доказ роботи)

PoS – Proof of Stake (Доказ власності)

PBFT – Practical Byzantine Fault Tolerance (Практична Візантійська відмовостійкість)

SOHO – Small Office/Home Office (Малий/домашній офіс)

TCP – Transmission Control Protocol (Протокол керування передачею)

SMB – Small/Medium Business (Мале/середнє підприємство)

SPoF – Single Point of Failure (Єдина точка відмови)

VoIP – Voice over Internet Protocol (Протокол інтернет-телефонії)

ВСТУП

Актуальність теми дослідження. Сучасні кібератаки зазвичай є добре підготовленими та спланованими і використовують великі об’єми ресурсів для обходу систем захисту. При цьому найбільш вразливими до кіберзагроз, в останній час, є невеликі комп’ютерні мережі малих та середніх підприємств (SOHO та SMB мережі), оскільки такі мережі, з одного боку, представляють інтерес для зловмисників, а, з іншого боку – не включають засоби безпеки такого рівня, як великі корпоративні мережі.

Виявлення та захист від кіберзагроз коп’ютерних мереж зазвичай забезпечується системами виявлення та попередження вторгнень, ефективність роботи яких сильно залежить від архітектури, а також їх місця розміщення в мережі. Дослідження відомих методів забезпечення кібербезпеки та виявлення вторгнень вказують, що реалізація нових моделей, методів та технологій забезпечення захисту комп’ютерних мереж та виявлення вторгнень, шляхом створення відповідних систем, потребує подальшого розвитку. А враховуючи сучасні реалії ведення гібридної війни, коли загрози можуть бути спрямовані на всі сфери життєдіяльності людини, завдання захисту інформаційної інфраструктури стає ще більш важливим. При цьому важливо, щоб захищеними залишались мережі всіх рівнів незалежно від розмірів та об’ємів трафіку. Отже, розробка інформаційної технології виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств на основі blockchain є **актуальною** задачею.

Технології та методи захисту мереж постійно вдосконалюються, враховуючи нові розробки в галузях обробки та класифікації великих об’ємів даних, а також захищеного зберігання та обміну даними. Зокрема, технології колаборативного виявлення вторгнень розглянуті у працях науковців Carol J. Fung, Olga Baysal, Trinh Anh Tuan, Vinod Yegneswaran, Paul Barford, Wenjuan Li, Weizhi Meng, Yu Wang, Lam For Kwok, Zhichun Li. Над вдосконаленням

механізмів виявлення вторгнень шляхом інтеграції blockchain технології працювали Nicholas Kolokotronis, Sotirios Brotsis, Georgios Germanos, Costas Vassilakis, Stavros Shiaeles, Abubakar, Aliyu Ahmed, Jinshuo Liu, Ezekia Gilliard, Gupta, Rajeev Kumar, Vedant Chawla, Rajesh Kumar Pateriya, Piyush Kumar Shukla, Saoucene Mahfoudh, Syed Bilal Hussain Shah, Kably, Salaheddine, Tajeddine Benbarrad, Nabih Alaoui, Mounir Arioua.

Проте роботи, присвячені використанню blockchain технології в задачах виявлення вторгнень, в основному розглядають загальну концептуальну модель системи виявлення вторгнень без врахування контексту її використання. Існує багато досліджень, які стосуються прикладів використання систем виявлення вторгнень на основі blockchain технології в мережах IoT, які достатньо сильно відрізняються, як за характером трафіку, так і за його об’ємами, від комп’ютерних мереж загального призначення і, зокрема, SOHO та SMB мереж, для яких гостро постає питання їх захисту.

Сучасні технології забезпечення інформаційної безпеки невеликих комп’ютерних мереж в основному базуються на локальних методах виявлення кіберзагроз, хоча деякі системи більш високого рівня використовують технології колаборативного виявлення вторгнень, коли локальні системи є частинами великої розподіленої системи, що об’єднує багато мереж.

Тому в роботі пропонується інформаційна технологія виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств на основі blockchain. В такому випадку залучення компонентів розподіленої системи дозволяє підвищити ефективність виявлення за рахунок використання накопиченої інформації в різних мережах, на які було здійснено кібератаки різного типу.

Отже, **актуальним науковим завданням** є розробка моделей та методів для інформаційної технології захисту комп’ютерних мереж, з

врахуванням особливостей мереж малих та середніх підприємств, які базуються на методах зберігання та поширення інформації на основі blockchain технології. Крім того, важливим є завдання визначення архітектури інформаційної системи виявлення вторгнень для комп'ютерних мереж малих та середніх підприємств, яка використовує blockchain компоненти.

Зв'язок роботи з науковими програмами, планами, темами. Представлена дисертаційна робота запланована та виконана в рамках міжнародного наукового проекту «Cyber Rapid Analysis for Defense Awareness of Real-time Situation – CyRADARS» за грантом NATO SPS, (grant agreement number: G5286)» та відповідно до плану науково – дослідної роботи Національного університету «Чернігівська політехніка» «Розробка моделей та методів захисту системи від зовнішніх атак з використанням технологій штучного інтелекту» (№0120U101931).

Мета й завдання дослідження. *Метою дисертаційного дослідження* є підвищення ефективності захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain технології.

Для досягнення мети дослідження в дисертації сформульовані та вирішені наступні завдання:

- виконати аналіз основних загроз інформаційній безпеці комп'ютерних мереж, а також методів та засобів забезпечення їх захисту;
- розробити модель розподіленої інформаційної системи виявлення та аналізу аномальних подій в комп'ютерних мережах малих та середніх підприємств на основі blockchain технології;
- запропонувати метод вибору протоколу консенсусу для розробленої моделі інформаційної системи;
- виконати адаптацію та підвищити ефективність роботи протоколу консенсусу PoS в розподілених системах виявлення вторгнень;

- виконати аналіз можливих методів класифікації аномальних подій та атак в комп'ютерних мережах малих та середніх підприємств і запропонувати комплексний класифікатор для захисту таких мереж;
- розробити функціональну модель та архітектуру розподіленої системи захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain технології;
- виконати UML проектування компонентів розподіленої системи захисту комп'ютерних мереж на основі blockchain технології;
- виконати тестування запропонованих моделей та методів системи захисту комп'ютерних мереж малих та середніх підприємств.

Об'єкт дослідження – інформаційні процеси в системах забезпечення захисту від кіберзагроз та аномального трафіку комп'ютерних мереж.

Предметом дослідження – є методи, моделі та елементи інформаційної технології колаборативного захисту від кібератак та аномального трафіку для комп'ютерних мереж малих та середніх підприємств на основі blockchain технології.

Методи дослідження. В основу методології дослідження покладено:

Імітаційне моделювання, UML проектування компонентів blockchain технології, методи математичного моделювання для визначення оптимальних параметрів blockchain підсистеми. Методи експертних оцінок використовувались для коректного вибору типових атак та навантажень на атаковані системи при побудові імітаційних моделей. Методи об'єктно-орієнтованого аналізу та функціонального моделювання, зокрема, SADT проектування, використані при концептуалізації бізнес-процесів у нотації IDEF0, які були взяті за основу при проектуванні інформаційної технології виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain.

Наукова новизна результатів досліджень полягає в наступному:

Вперше:

- розроблена концептуальна модель розподіленої інформаційної системи виявлення та аналізу аномальних подій в комп'ютерних мережах малих та середніх підприємств, яка на відміну від існуючих, містить blockchain компонент для виявлення, накопичення, збереження та спільного використання інформації про аномальні події та блок мультикласифікатора для визначення наявності загрози, що дозволяє підвищити швидкість реагування на невідомі атаки;
- запропоновано метод вибору протоколу консенсусу для розподіленої системи виявлення вторгнень на основі blockchain, який на відміну від існуючих, враховує вимоги до обладнання, масштабування та керування учасниками систем виявлення вторгнень в комп'ютерні мережі, що забезпечує підтримку прийняття рішень при проектуванні систем захисту комп'ютерних мереж малих та середніх підприємств.

Удосконалено:

- метод консенсусу PoS blockchain технології, який на відміну від існуючих, використовує в якості значення ставки час роботи вузла в розподіленій системі і дозволяє використовувати blockchain для децентралізованого зберігання даних розподіленої системи виявлення вторгнень в комп'ютерні мережі малих та середніх підприємств.

Набула подальшого розвитку:

- функціональна модель розподіленої системи захисту комп'ютерних мереж на основі blockchain технології для виявлення, накопичення, збереження та спільного використання інформації про аномальні події, яка визначає основні вхідні, вихідні параметри, обмеження та ресурси з трьома рівнями деталізації та є основою для проектування систем захисту комп'ютерних мереж малих та середніх підприємств.

Практичне значення отриманих результатів. Наведені вище наукові результати у своїй сукупності утворюють нову інформаційну технологію виявлення та аналізу аномальних подій для захисту комп'ютерних мереж на основі blockchain. Розроблено програмний модуль blockchain підсистеми, що слугує основою для побудови децентралізованої розподіленої системи виявлення вторгнень, а також програмний модуль мультикласифікатора, які є частиною інформаційної технології виявлення та аналізу аномальних подій. Запропонована інформаційна технологія, може бути використана як розробниками систем захисту комп'ютерних мереж, так і мережевими адміністраторами та ІБ спеціалістами малих та середніх підприємств. Розроблені бізнес процеси та архітектура є основою для розробки більш потужних та функціональних розподілених систем виявлення вторгнень.

Результати дисертаційного дослідження впроваджені:

- при виконанні міжнародного наукового проекту «Cyber Rapid Analysis for Defense Awareness of Real-time Situation – CyRADARS» за грантом NATO SPS, (grant agreement number: G5286)»;
- у навчальному процесі Національного університету «Чернігівська політехніка» при проведенні лекцій та лабораторних робіт з дисципліни «Системи захисту обчислювальних мереж» – в процесі навчання бакалаврів спеціальності 121 – «Інженерія програмного забезпечення» на кафедрі інформаційних технологій та програмної інженерії (довідка про впровадження №203/08-1128/ВС від 14.05.2024 Додаток В);
- в Чернігівському обласному філармонійному центрі фестивалів та концертних програм при виявленні аномальних подій зовнішнього та внутрішнього походження в комп'ютерній мережі підприємства з метою визначення основних векторів атак на мережу та подальшої розробки заходів з укріплення захисту мережевої інфраструктури

Чернігівського обласного філармонійного центру (довідка про впровадження №131/01-08 від 7.05.2024 Додаток В);

- в ТОВ "ІНФОРМАЦІЙНІ СИСТЕМИ ЗАХИСТУ" при визначенні основних векторів атак на інформаційну інфраструктуру підприємства та розробці комплексу заходів для підвищення рівня захищеності мережевої інфраструктури (акт про впровадження №54 від 13.05.2024 Додаток В).

Особистий внесок здобувача. Наукові результати, викладені в дисертаційній роботі, отримані автором особисто. В наукових роботах, опублікованих у співавторстві, в дисертації використані лише ті ідеї та положення, що є результатом особистої роботи.

Апробація результатів дисертації.

Основні положення дисертаційного дослідження доповідалися та обговорювалися на міжнародній конференції «Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища» INUDECO (м. Славутич, квітень 2017, квітень 2019, квітень 2020, квітень 2021), І Міжнародній науково-практичній конференції “Безпека ресурсів інформаційних систем” (м. Чернігів 16-17 квітня 2020р.), Чотирнадцятій міжнародній науково-практичній конференції “Математичне та імітаційне моделювання систем” (Чернігів, 24 - 26 червня 2019 р.),

Публікації. За темою дисертаційного дослідження з викладенням основних результатів опубліковано 10 наукових праць. Серед них 5 статей у фахових журналах, 3 з них включені до міжнародної наукометричної бази Scopus та опубліковані в закордонних виданнях [1–3]; 4 праці апробаційного характеру [7-10]. Результати роботи доповідалися на 8 міжнародних наукових конференціях.

Структура та обсяг роботи. Дисертаційна робота складається зі вступу, 4 розділів (глав), висновків, переліку умовних скорочень, переліку посилань зі 129 джерел та 3 додатків. Загальний обсяг роботи становить 231

сторінки, з яких зміст на 3 сторінках, вступ на 7 сторінках, перелік умовних скорочень на 1 сторінці, основний текст на 198 сторінках, список використаних джерел із 129 найменувань на 14 сторінках, 3 додатки на 12 сторінках. Робота містить 33 рисунки (з них 3 рисунки на 3 окремих сторінках) та 5 таблиць (з них одна таблиця на 8 окремих сторінках).

РОЗДІЛ 1
АНАЛІЗ ОСНОВНИХ ЗАГРОЗ ІНФОРМАЦІЙНІЙ БЕЗПЕЦІ КОМП'ЮТЕРНИХ МЕРЕЖ МАЛИХ ТА СЕРЕДНІХ ПІДПРИЄМСТВ ТА ІСНУЮЧИХ МЕТОДІВ ЇХ ЗАХИСТУ

1.1 Поняття кіберзагрози та їх класифікація

Інцидент кібербезпеки або кіберінцидент – це подія або ряд несприятливих подій ненавмисного характеру (природного, технічного, технологічного, помилкового, у тому числі внаслідок дії людського фактора) та/або таких, що мають ознаки можливої (потенційної) кібератаки, які становлять загрозу безпеці систем електронних комунікацій, систем управління технологічними процесами, створюють імовірність порушення штатного режиму функціонування таких систем (у тому числі зриву та/або блокування роботи системи, та/або несанкціонованого управління її ресурсами), ставлять під загрозу безпеку та захищеність електронних інформаційних ресурсів.

На відміну від кіберінцидента, кібератака - це спрямовані навмисні дії в кіберпросторі, які здійснюються за допомогою засобів електронних комунікацій, включаючи інформаційно-комунікаційні технології, програмні, програмно-апаратні засоби, інші технічні та технологічні засоби і обладнання. Згідно із Законом України «Про основні засади забезпечення кібербезпеки України» [1] кібератаки спрямовані на досягнення однієї або кількох перелічених цілей:

- порушення конфіденційності, цілісності, доступності електронних інформаційних ресурсів, що обробляються (передаються, зберігаються) в комунікаційних та/або технологічних системах, отримання несанкціонованого доступу до таких ресурсів;
- порушення безпеки, сталого, надійного та штатного режиму функціонування комунікаційних та/або технологічних систем;

- використання комунікаційної системи, її ресурсів та засобів електронних комунікацій для здійснення кібератак на інші об’єкти кіберзахисту;

У літературі терміни *загроза* та *атака* зазвичай використовуються для позначення одного і того ж поняття. Але W. Stallings [2]. зазначає, що при більш детальному розгляді визначень, взяті з RFC 4949 [3], які також відповідають рекомендації X.800 ITU-T [4], що наведені в Таблиці 1.1, можна побачити суттєву відмінність цих понять

Таблиця 1.1 – порівняння понять «загроза» та «атака».

Термін	Визначення
Загроза	Потенційна можливість порушення безпеки, яка існує за наявності обставин, умов, дій чи подій, які можуть порушити безпеку та завдати шкоди. Тобто загроза – це можлива небезпека, яка може використовувати вразливості системи.
Напад (атака)	Посягання на безпеку системи, яке походить від розумної загрози, тобто осмислений вчинок, який є навмисною спробою (особливо у сенсі методу чи технології) ухилитися від служб безпеки та порушити політику безпеки системи.

Таким чином термін атака із Інтернет глосарія RFC4949 змістовно відповідає визначенню кібератаки, яке прописане в Законі України “Про основні засади забезпечення кібербезпеки України”.

Одним із видів класифікації атак безпеки, який використовується як у X.800, так і в RFC 4949, є поділ на пасивні атаки та активні атаки. Пасивна атака спрямована на отримання або використання інформацію із системи, але не впливає на нормальну роботу системи. Активна атака це спроба змінити системні ресурси або вплинути на їх роботу.

Пасивні атаки мають характер прослуховування або моніторингу передач. Мета атакуючого – отримати інформацію, яка передається. Два типи пасивних атак – це перехоплення вмісту повідомлення та аналіз трафіку.

Перехоплення вмісту повідомлення найпростіший вид пасивних атак. Телефонна розмова, повідомлення електронної пошти або переданий файл можуть містити секретну або конфіденційну інформацію.

Другий тип пасивних атак, аналіз трафіку, є більш прогресивним. Припустимо, у нас був спосіб замаскувати вміст повідомлень або іншого інформаційного трафіку, щоб зловмисники, навіть якщо вони захопили повідомлення, не могли отримати їх вміст. Поширеною технікою маскування вмісту є шифрування. Навіть якщо наші повідомлення захищені шифруванням, зловмисник все одно може спостерігати шаблон цих повідомлень. Атакуючий може визначити місцезнаходження та ідентичність хостів, що спілкуються, і може спостерігати за частотою та довжиною повідомлень, якими обмінюються учасники. Ця інформація може бути корисною для здогадок про характер спілкування, що відбувається.

Пасивні атаки дуже важко виявити, оскільки вони не передбачають жодного втручання в дані, що передаються. Як правило, трафік повідомлень надсилається та отримується в нормальному вигляді, і ані відправник, ані одержувач не знають, що третя сторона читає повідомлення або спостерігає за шаблонами трафіку. Однак запобігти успішному проведенню цих атак можна за допомогою шифрування. Таким чином, акцент у захисті від пасивних атак робиться на попередженні таких атак, а не на їх виявленні [2].

Підслуховування — одна з найпоширеніших пасивних атак на думку Sumi [5]. І одна з причин цього полягає в тому, що, як правило, така атака відбувається у вигляді таємного прослуховування приватного обміну інформацією між хостами. Атакуючий намагається викрасти інформацію, яку користувач передає через мережу за допомогою комп’ютера, смартфона чи інших пристроїв. Зловмисник використовує незахищені мережеві

комунікації для доступу до даних, що надсилаються та отримуються користувачем. Цей тип атак важко виявити, оскільки вони не втручаються в роботу мережі. Головною метою цієї атаки є отримання конфіденційної інформації, яка можу бути використана в подальшому зловмисником.

Активні атаки передбачають деяку модифікацію потоку даних або створення помилкового потоку і можуть бути поділені на чотири категорії: маскування, повторне відтворення, модифікація повідомлень та відмова в обслуговуванні.

Маскування має місце, коли одна сутність видає себе за іншу сутність. Атака маскування зазвичай включає одну з інших форм активної атаки. Наприклад, послідовності автентифікації можуть бути захоплені та відтворені після того, як відбулася діюча послідовність автентифікації, що дозволяє авторизованому об'єкту з невеликими привілеями отримати додаткові привілеї, видаючи себе за об'єкт, який має ці привілеї.

Повторне відтворення передбачає пасивне захоплення блоку даних та його подальше повторне пересилання для отримання несанкціонованого доступу, привілеїв або несанкціонованого виконання певних дій.

Модифікація повідомлень означає, що якась частина легітимного повідомлення змінюється або повідомлення затримуються або переупорядковуються, щоб отримати несанкціонований ефект [2].

Одна з найбільш поширених активних атак - це атака відмови в обслуговування (DoS), а також розподілений її варіант DDoS. Атака розподіленої відмови в обслуговуванні виникає, коли кілька систем заповнюють пропускну здатність або ресурси цільової системи, як правило, одного або декількох веб-серверів. Це атака, призначена для перевантаження, а іноді і вимкнення машини або мережі, що робить її недоступною для користувачів. Ціль під час цієї атаки перевантажують трафіком або надсилають їй інформацію, яка викликає збій. Оскільки кожен сервер має обмежений пропускну здатність або можливість відповідати на вхідний

запит, коли трафік перевищує пропускну здатність сервера – сервер виходить з ладу. І коли відвідувач намагається отримати доступ до сайту, або іншого сервісу розміщеного на сервері, він отримує відмову у наданні послуги. Існує два типи DDOS-атак. Один з них направлений на виснаження пропускної здатності. Цей метод полягає в перевантаженні мережі – завантаження смуги пропускання великими об’ємами трафіку тоді призводить до виходу мережі з ладу. Інший тип – виснаження ресурсів. Зловмисник виснажує ключові ресурси, такі, як процесорний час, пам'ять тощо, чим і порушує роботу сервера. Атака зазвичай починається з численних джерел, спрямованих на одну ціль. Атаки спрямовані на кілька цілей зустрічаються рідше, однак, існує можливість для зловмисників розпочати такий тип атаки підробленою, зміненою або відтвореною інформацією про маршрутизацію.

З іншого боку, F. Sumi [5] виділяє два загальних методи DDOS-атак: «затоплення» сервісу або аварійне завершення сервісу, базуючись на причині відмови сервісу. Атаки «затоплення» трапляються, коли система отримує занадто багато трафіку, сервер при цьому вимушений буферизувати його, що призводить до уповільнення та збою в роботі служб при переповненні буфера. Аварійне ж завершення виникає, коли атакуючий надсилає трафік, що містить некоректні дані і викликає збій в роботі сервера. Така класифікація є більш актуальною при розгляді питання захисту мережевих сервісів від DDOS атак оскільки для побудови надійного захисту необхідна максимально деталізована інформація про вектор атаки.

Іншим популярним видом активних атак є фішинг – це атака, яка націлена на отримання грошей або конфіденційної інформації обманним шляхом. Фішинг – це пастка, яка обманним шляхом з певних причин просить людей ввести свою особисту інформацію для доступу до свого облікового запису [5]. Користувачі переходять на шахрайську копію веб-сайту початкової установи, наприклад переходячи за посиланням в електронному листі, і вводять свою інформацію, не здогадуючись, що стали жертвами

шахраїв. Потім шахрай отримує доступ до банківського рахунку клієнта в Інтернеті та коштів, що містяться на цьому рахунку, або інших чутливих даних, які можуть перепродаватися, або використовуватися з іншою вигодою для зловмисника.

Атаки прямого доступу також вважаються активними і трапляються, коли зловмисники отримують особисті дані користувача безпосередньо за допомогою фізичного пристрою цього користувача. Користувачі здебільшого не знають про атаку. При цьому прямий доступ може бути застосований не тільки до мобільних пристроїв, а і до недостатньо захищеної серверної інфраструктури.

Ще один вид активних атак, що широко застосовується це створення ботнетів. Коли користувачі несвідомо завантажують шкідливі програми, наприклад троянські коні, які можуть бути надіслані як вкладення електронної пошти, і заражають свій комп’ютер. Зловмисник при цьому отримує доступ до таких заражених комп’ютерів зомбі і може використовувати їх в своїх цілях. Згідно Thanh Vu [6], такі ботнети використовуються для проведення DDoS атак, видобування криптовалюти, або крадіжки цінної інформації, при цьому часто для користувача таке зараження непомітне.

В окрему групу також можна виділити атаки спрямовані на веб-ресурси. До таких атак належать міжсайтові сценарії та атаки клацання. Міжсайтовий сценарій (XSS) - це тип атаки, який використовує відповідну вразливість у веб-програмах, що дозволяє виводити зловмиснику шкідливий код на веб-сторінки, які переглядають інші користувачі. В своїй статті Kumar [7] Зазначає, що прикладами такого коду є HTML-код та сценарії на стороні клієнта із вбудованим зловмисним функціоналом. При цьому вразливість міжсайтових сценаріїв може використовуватися зловмисниками для обходу засобів контролю доступу. Атака міжсайтових сценаріїв - це тип ін’єкційної атаки, при якій шкідливі сценарії вводяться на довірені та надійні веб-сайти.

Клацання або clickjacking - це зловмисна техніка, при якій зловмисник приваблює людей за допомогою красивих зображень, що блимають на сторінці веб-сайту, але при цьому зловмисник хоче, щоб користувач натискав кнопку або посилання на іншій веб-сторінці, тоді як користувач мав намір натиснути на цій сторінці. Це робиться за допомогою декількох прозорих або непрозорих шарів [8]. Зловмисник намагається зламати особистий обліковий запис користувача, або виконати якусь дію від імені користувача, коли він натискає зображення, встановлене зловмисником.

Опис та порівняння основних сучасних типів кібератак наведено у додатку В.

Комп’ютерні мережі відрізняються за розмірами, за призначенням, за складом та технологіями, що використовуються. Такі відмінності впливають і на методи та технології захисту мереж від вторгнень. В першу чергу на методи захисту впливає кількість трафіку в мережі, яка зазвичай залежить від розміру мережі та наявних мережевих сервісів.

Одним із критеріїв класифікації мереж є їх масштаби. Відповідно до фізичного розміру мережі, мережі можуть бути:

- *Персональні мережі (PAN)* – призначені для однієї людини або невеликої групи людей, наприклад сім’ї. Типовим прикладом PAN є домашня мережа бездротового зв'язку. Ключовою можливістю PAN є надання можливості пристроям самостійно виявляти та підключатися один до одного.
- *Локальні мережі (LAN)* – це мережі невеликого розміру, які охоплюють лише кілька будівель або місцевість розмірами до декількох кілометрів. Локальні мережі широко використовуються для підключення персональних комп’ютерів та робочих станцій у офісах, на підприємствах та в університетах для обміну ресурсами та інформацією. Типовим прикладом локальної мережі є спільне використання принтерів у лабораторії чи відділі.

- *Мережі районів (MAN)* – можуть охоплювати місцевість, розмір якої менше 100 км. Найвідомішим прикладом MAN є мережі кабельного телебачення. Швидкісний бездротовий доступ до Інтернету - ще один приклад MAN.
- *Широкообласні мережі (WAN)* – охоплюють великі географічні райони, часто країни або континенти. Вони містять мільйони машин, які з'єднані підмережами зв'язку. Інтернет - це найбільша глобальна мережа, що була коли-небудь створена.

Окремо слід відзначити *корпоративні мережі*, особливістю яких є те, що вони надають послуги лише користувачам — співробітникам підприємства, яке користується мережею. Бондарєв у своїй книзі [9] вказує, шо на відміну від мереж операторів зв’язку, мережі підприємств, зазвичай, не надають послуг іншим організаціям або користувачам. А для корпоративних мереж, які користуються технологіями глобальної мережі Інтернет, часто використовується термін мережа Intranet. Залежно від масштабу підприємства, а також від складності й різноманіття розв’язуваних завдань розрізняють мережі відділу, мережі кампуса і мережі масштабу підприємства.

Якщо ж розглядати класифікацію у контексті корпоративних мереж, то за розміром мережі можна поділити на:

- *SOHO мережі* — домашні, або мережі дрібних офісів, (найближче до PAN, але може також включати невеликі LAN мережі). Також до цього класу можна віднести мережі робочих груп. Зазвичай включають до 15-20 користувачів, і не містять серверів налаштованих мережевих сервісів, зазвичай такі локальні мережі обмежуються використанням робочих груп Windows для обміну файлами, а більша частина сервісів, які використовуються користувачами, розташована в інтернеті або мережі вищого рівня. Такі мережі характеризуються простотою та однорідністю. Оскільки такі мережі не містять великої

кількості сервісів доступних ззовні, вони не вимагають складних систем захисту від вторгнень. Зазвичай достатньо правильно налаштованого міжмережевого екрану та за необхідності простої системи виявлення вторгнень.

- *Мережі відділів* — відрізняються від SOHO мереж більшою кількістю користувачів. Зазвичай такі мережі включають один або кілька серверів. Також характеризуються однорідністю оскільки використовуються в рамках одного відділу, для вирішення схожих задач. У більшості випадків такі мережі є частинами мережі підприємства, в такому випадку додаткового захисту вони не потребують. У випадку ж якщо така мережа має пряме з'єднання з глобальною мережею, потрібне використання міжмережевого екрану та простої системи виявлення вторгнень.
- *SMB мережі* — мережі підприємств малого та середнього бізнесу. Відповідно до українського [10] та європейського [11] законодавства, підприємства середнього бізнесу можуть мати до 250 працівників. В контексті масштабу мережі, можна вважати, що до цього класу належать мережі, що мають до 1000 користувачів. Такі мережі відповідають підприємствам малого та середнього бізнесу, такі мережі вже можуть включати домен Active directory, та можуть містити кілька серверів, зазвичай в якості сховища, та простих не високонавантажених сервісів. Оскільки такий тип мереж містить значно більше різноманітного обладнання та програмного забезпечення, такі мережі вимагають надійного захисту. Для таких мереж обов'язкове використання міжмережевого екрану, а також бажане використання системи виявлення вторгнень.
- *Enterprise мережі* – Мережі великих підприємств, зазвичай включають велику кількість сервісів, у тому числі відкритих до глобальної мережі. Такі мережі зазвичай включають в себе більш

дрібні мережі, що приводить до високого рівня неоднорідності мережі. Крім того, в таких мережах зазвичай одночасно працює велика кількість користувачів. Такі мережі вимагають використання спеціалізованого обладнання для захисту від вторгнень.

1.2 Основні актуальні загрози для мереж SOHO та SMB класу

На перший погляд здається, що основними цілями для хакерських атак є зазвичай великі корпоративні мережі, а загроз для SOHO та SMB настільки мало, що такі мережі не вимагають додаткових спеціалізованих методів та засобів захисту. Але проаналізувавши актуальні звіти по виявленим кібератакам, бачимо, що SOHO та SMB мережі все частіше стають ціллю для кібератак, більше того, за статистикою в останні роки більша частина кібератак направлені проти мереж такого класу.

На таку привабливість SOHO та SMB мереж для зловмисників впливає кілька основних факторів:

1. Невеликих мереж SOHO та SMB класу значно більше аніж великих корпоративних мереж, що дає зловмисникам простір для вибору, при чому мережі такого класу зустрічаються в більшості галузей людської діяльності;
2. Відсутність спеціалізованих апаратних або програмних засобів виявлення та захисту від вторгнень [12], а зловмисники частіше за все йдуть шляхом найменшого опору;
3. Відсутність спеціалістів потрібної кваліфікації.

Так згідно звітів The State of Cloud Security 2021 Report [13] та Cybersecurity INSIDERS Cloud Security Report у 2021 [14] році близько 36-50% компаній, які використовували хмарні засоби захисту мереж стикнулися з витоками інформації щодо безпеки або прогалинами в безпеці саме через неправильну конфігурацію хмарних засобів. Станом на 2023 рік ситуація змінюється в кращу сторону (згідно Cybersecurity INSIDERS 2023 Cloud

Security Report [15]), і витоки даних перестали бути основною загрозою, але тим не менш, близько 24% організацій стикнулись з кіберінцидентами пов’язаними з використанням хмарних ресурсів. При цьому вагомий відсоток інцидентів був пов’язаний із неправильним конфігуруванням або компроментацією облікових записів.

Як бачимо, через стрімке зростання кількості кібератак, практично в 6 разів, через пандемію COVID-19 і переведення великої частини бізнесу в онлайн, а із початком повномасштабного вторгнення в лютому 2022 року, кількість кібератак на мережі України значно зросла. При цьому мережі малих та середніх підприємств особливо потрапляють в групу ризику. Статистика говорить, що близько 70% відсотків таких мереж абсолютно не готові до таких атак. Більше того, близько 51% підприємств малого бізнесу взагалі не виділяють ніякого бюджету на питання кібербезпеки. Близько 20% малих та середніх підприємств взагалі не використовують ніяких засобів захисту мережі, а близько 33% покладаються лише на безкоштовні засоби. З необхідності фінансових витрат на питання безпеки випливає відсутність персоналу з необхідною кваліфікацією, а також відсутність апаратних та програмних засобів захисту. При чому тут слід звернути увагу, що безкоштовні програмні засоби для домашнього використання мало підходять для використання в мережах підприємств, оскільки вони в основному пристосовані для захисту локальних комп’ютерів від загроз, з якими частіше за все стикається домашній користувач. Загрози ж направлені на мережі підприємств зазвичай носять зовсім інший характер та часто спрямовані на зупинку роботи сервісів і в 34% випадків відновлення після таких кібератак займає більше одного тижня.

Іноді компанії проявляють недбале ставлення до питання інформаційної безпеки, оскільки вважають, що відсутність привабливих ресурсів знижує зацікавленість зловмисників. Але навіть якщо не приймати до уваги можливість втрати персональних даних або інтелектуальної

власності, компанія може представляти цінність як посередник для компрометації контрагентів [16].

Отже, мережі SOHO та SMB класів на даний момент перебувають в групі ризику та вимагають підвищеної уваги до питання безпеки в комплексному розумінні цього поняття. Але особливу увагу тут слід звернути на програмні та апаратні засоби мережевої безпеки, оскільки такі мережі вимагають простих, легких у розгортанні та ефективних засобів захисту.

1.3 Основні фактори мережевої безпеки

Часто вважається, що безпека мережі залежить в першу чергу від використовуваних програмних та апаратних засобах захисту, але навіть поверхневий огляд статистичних даних показує, що поняття безпеки є комплексним і жоден окремий компонент самостійно не гарантує надійного захисту. Отже, перш ніж захищати мережу слід з’ясувати основні фактори безпеки.

Згідно з CyBOK 1.1 [17] фактори безпеки можна поділити на дві категорії:

- Фактори, що стосуються захисту від атак та попередження вторгнень;
- Фактори, що стосуються людських ресурсів, організації та правових аспектів.

При цьому інформаційна безпека розглядається з точки зору трьох основних складових:

- Безпека інфраструктури;
- Системна безпека;
- Безпека програм і платформ.

Інфраструктурний рівень безпеки передбачає використання тих чи інших засобів захисту каналів зв’язку та організацію безпеки фізичного рівня

для інфраструктури. Хоча фізичні атаки є достатньо затратними і більш поширені для великих мереж і підприємств, погана захищеність SOHO та SMB мереж може також спровокувати зловмисника на фізичну атаку, тому базовим рівнем захисту на рівні інфраструктури для таких мереж повинен такий захист що передбачає використання криптографічно захищених мережевих з'єднань та обмеження фізичного доступу до важливих вузлів мережі, таких як сервери.

Безпека рівня системи передбачає використання захищених операційних систем, безпечних методів авторизації, аутентифікації та обліку як на рівні окремих вузлів, так і на рівні розподілених систем за їх наявності. Для невеликих мереж цей рівень передбачає використання актуальних операційних систем з встановленими оновленнями безпеки, шифрування цінних даних, що зберігаються в мережі, а також використання доступних засобів AAA (authentication, authorization, accounting – автентифікація, авторизація, облік).

Безпека рівня програм та платформ передбачає використання безпечного програмного забезпечення, тобто мережеве програмне забезпечення повинно мати актуальну версію та встановлені оновлення безпеки. Окрім того слід звертати увагу на життєвий цикл використовуваного програмного забезпечення та вчасно реагувати на виведення з експлуатації певних версій програмного забезпечення.

Фактори, що стосуються людських ресурсів, можна поділити на кілька груп, і основна група стосується поведінки користувачів мережі, оскільки поведінка напряму впливає на наявні вразливості мережі. Оскільки саме людський фактор стає причиною послаблення захисту мережі та витоків інформації. В таких випадках на передній план виходить поняття культури інформаційної безпеки [17]. Оскільки часто зловмисники використовують людино-орієнтовані загрози, наприклад підроблені листи, посилання, фальшиві профілі та ін., якими можуть скористатися користувачі в процесі

роботи і тим самим пустити зловмисника в мережу. Таким чином необачність користувача може призвести до серйозних збитків та втрати інформації. На думку Niekerk [18] це в першу чергу це пов’язане з тим, що зазвичай користувачі не мають достатнього рівня знань з інформаційної безпеки, оскільки це не пов’язане напряму з їх роботою. Тому навчання користувачів основам інформаційної безпеки є достатньо важливим фактором в забезпеченні інформаційної безпеки мереж підприємств, зокрема SOHO та SMB мереж.

Із вищесказаного слідує, що захист мережі має бути комплексним і враховувати всі доступні рівні та фактори. При цьому захист мережі повинен реалізовуватись як на рівні політик безпеки, так і за допомогою рішень безпеки. Рішення безпеки не еквівалентно політиці безпеки. Рішення безпеки підтримує політику безпеки, але не замінює її, і ця відмінність, хоча вона може здатися зрозумілою, має тенденцію розмиватися під час процесу проектування, якщо підприємство не має чітко визначеної політики.

Підприємства малого та середнього бізнесу, які не мають достатніх ресурсів для забезпечення внутрішньої мережевої безпеки персоналу, ймовірно, не мають політики безпеки і можуть покладати питання розробки політики безпеки на персонал, що розробляє рішення безпеки [19]. Хоча обидва завдання є необхідними для надійного захисту, розробка політики безпеки може мати інші юридичні наслідки, ніж розробка рішення безпеки для її реалізації, при цьому відсутність політики може негативно вплинути на якість розробленого рішення.

Також можна виділити групу персоналу відповідального за безпеку мережі, зазвичай це системні адміністратори або адміністратори безпеки. Від їх кваліфікації та сумлінної роботи залежить правильність налаштування та роботи як самої мережі, так і засобів її захисту.

Окрім людського фактору, достатньо сильний вплив на безпеку мережі має її архітектура. Великі мережі зазвичай мають деревовидну

структуру і поділені на домени, в рамках яких і налаштовуються політики безпеки в залежності від вимог до мережі та її функцій. Проте мережі SOHO та SMB сегменту зазвичай не включають такої великої кількості вузлів тому їх архітектура значно простіша. Такі мережі у випадку їх приєднання до глобальної мережі інтернет зазвичай мають мережевий екран або маршрутизатор з функцією мережевого екрану, який і відповідальний за відсіювання шкідливого трафіку, що приходить ззовні. Окрім того за наявності окремих зон, функція поділу мережі на зони також покладена на мережевий екран. Так, наприклад, веб сервер, що повинен бути доступний ззовні слід винести в демілітаризовану зону DMZ з якої відсутній доступ до основної мережі, тоді такий сервер зловмисники не зможуть використати в якості плацдарму для атаки на інші вузли мережі. Інші ж вузли в таких мережах зазвичай підключаються через серію комутаторів та бездротових точок доступу [20]. Структура типової SMB мережі наведена на рисунку 1.1. На ній окрім веб-сервера, що знаходиться в демілітаризованій зоні присутній файловий сервер та мережевий принтер, які підключені до основної зони мережі.

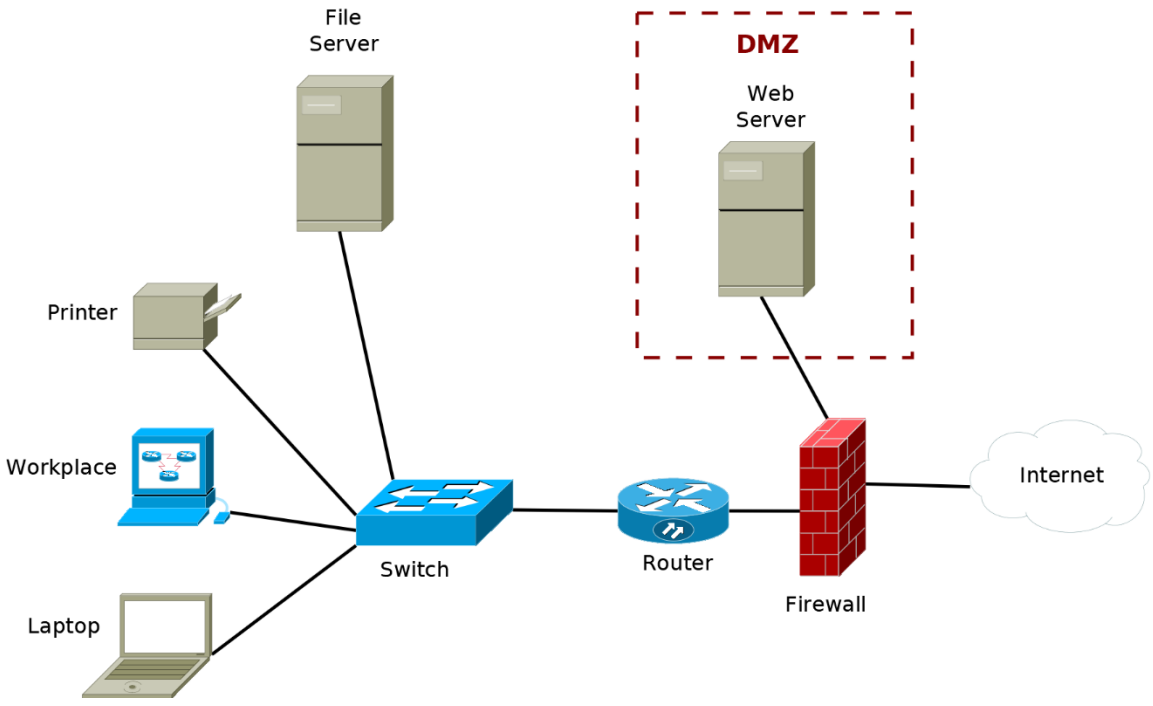


Рисунок 1.1 – Схема типової мережі SMB класу з демілітаризованою зоною

DMZ зона мережі (демілітаризована зона) – це зазвичай мережа периметра, яка захищає та додає додатковий рівень безпеки внутрішній локальної мережі організації від ненадійного трафіку. Це підмережа, яка знаходиться між загальнодоступним Інтернетом та приватними мережами. На думку Alvarez [21] кінцева мета DMZ — дозволити організації отримати доступ до ненадійних мереж, таких як Інтернет, при цьому забезпечити безпеку її приватної мережі або локальної мережі. Зазвичай організації зберігають зовнішні послуги та ресурси, а також сервери для системи доменних імен (DNS), протоколу передачі файлів (FTP), пошти, проксі-сервера, протоколу передачі голосу через Інтернет (VoIP) і веб-серверів у DMZ. Такі сервери та ресурси ізольовані та мають обмежений доступ до локальної мережі, щоб забезпечити доступ до них через Інтернет, без прямого доступу до внутрішньої мережі. У результаті підхід DMZ ускладнює хакеру отримання прямого доступу до даних організації та внутрішніх серверів через Інтернет.

DMZ ізольована шлюзом безпеки, таким як брандмауер, який фільтрує трафік між DMZ і локальною мережею. Сервер DMZ в більшості випадків захищений іншим шлюзом безпеки, який фільтрує трафік, що надходить із зовнішніх мереж [22]. В ідеальному випадку DMZ розміщується між двома брандмауерами. Якщо зловмиснику вдається проникнути через зовнішній брандмауер і зламати систему в DMZ, він також повинен пройти внутрішній брандмауер, перш ніж отримати доступ до конфіденційних корпоративних даних.

Основною перевагою DMZ є забезпечення внутрішньої мережі розширеного рівня безпеки шляхом обмеження доступу до конфіденційних даних і серверів. DMZ дозволяє відвідувачам веб-сайту отримувати певні послуги, забезпечуючи буфер між ними та приватною мережею організації. В результаті DMZ також пропонує додаткові переваги безпеки, такі як:

- *Увімкнення контролю доступу*: підприємства можуть надавати користувачам доступ до послуг за межами їхньої мережі через загальнодоступний Інтернет. DMZ надає доступ до цих послуг, реалізуючи сегментацію мережі, щоб ускладнити доступ неавторизованого користувача до приватної мережі [23]. DMZ також може включати проксі-сервер, який централізує внутрішній потік трафіку та спрощує моніторинг і запис цього трафіку.
- *Запобігання розвідці мережі*: забезпечуючи буфер між Інтернетом і приватною мережею, DMZ не дає можливості зловмисникам виконувати розвідувальну роботу, яку вони здійснюють для пошуку потенційних цілей та вразливостей. Сервери в DMZ відкриті для всіх, але брандмауер пропонує іншу політику безпеки, яка не дозволяє зловмисникам бачити внутрішню мережу. Навіть якщо система DMZ буде скомпрометована, внутрішній брандмауер відокремлює приватну мережу від DMZ, щоб забезпечити її захист та ускладнити зовнішню розвідку.
- *Блокування підробки Інтернет протоколу (IP)*: зловмисники намагаються знайти способи отримати доступ до систем, підробляючи IP-адресу та видаючи себе за затверджений пристрій, що увійшов у мережу. DMZ може виявляти та зупиняти такі спроби спуфінгу, оскільки інша служба перевіряє легітимність IP-адреси [24]. DMZ також забезпечує сегментацію мережі, щоб створити простір для організації трафіку та доступу до мережевих сервісів віддалено від внутрішньої приватної мережі.

1.4 Сучасні методи та засоби виявлення атак та захисту комп'ютерних мереж

Усі методи захисту комп'ютерних мереж можна розділити на превентивні методи, основна задача яких запобігати виникненню аномальних подій та відбиття атак на початковому їх етапі, та методи захисту від активних загроз, коли для виявлення чи блокування загрози необхідно спочатку проаналізувати саму загрозу та визначити правильний вектор її обробки. Зазвичай такі методи використовуються комплексно для отримання більш надійного мережевого захисту. При цьому після встановлення та налаштування усіх необхідних засобів важливо перевірити коректність їх комплексної роботи в конкретній мережі. Для цього можуть використовуватися апаратні та програмні рішення, що дозволяють моделювати деякі типи атак, і таким чином дозволяють перевірити надійність захисту. Один із варіантів такого технічного рішення був запропонований в роботі [25]. Запропоноване авторами рішення передбачає використання модуля імітації атак та моніторингового модуля, які дають можливість оцінити реакцію мережі на ті чи інші типи аномальних подій та вторгнень і таким чином виявити слабкі місця і оцінити ступінь ризику для кожного з векторів атак.

До основних технічних засобів захисту мереж можна віднести міжмережеві екрани, які в першу чергу слугують превентивними засобами захисту, блокуючи можливість потрапляння небажаного трафіку, та системи виявлення вторгнень, які на основі інформації про отриманий трафік здатні виявляти аномальний трафік та спроби атак.

1.4.1 Мережеві екрани (firewall)

Міжмережевий екран — це комплекс програмних або апаратно-програмних засобів, що виконує фільтрацію та контроль мережевих пакетів, що проходять через нього відповідно до заданих правил [26]. Іншими

словами, брандмауер відфільтровує з мережевого трафіку шкідливі повідомлення, експлойти, вторгнення та інші шкідливі дані [27].

Далеко не увесь мережевий трафік надходить із авторизованих джерел, такий трафік треба блокувати. Також частина трафіку може бути нецільовою або виходити за норми мережевої активності, що спричиняє проблеми в роботі мережі, тому проходження такого трафіку також треба обмежувати. Задачі блокування такого трафіку вирішуються за допомогою міжмережевого екрану [27].

У своїй книзі Nguyen [26] зазначає, що основна задача міжмережевого екрану захист комп'ютера або мережі від зовнішніх загроз (загроз з мережі Інтернет) та блокування витоку інформації із комп'ютера або захищеної мережі в зовнішню.

Класичний міжмережевий екран працює за принципом контролю всіх з'єднань та блокування недозволених портів та протоколів згідно із заздалегідь написаними правилами. При цьому міжмережевий екран може бути розташований як на шлюзі, через який проходить увесь трафік направлений в зовнішню мережу (мережевий брандмауер), так і на конкретному комп'ютері (персональний брандмауер) [26].

Персональні брандмауери призначені для захисту одного хоста від несанкціонованого доступу. В результаті багаторічного розвитку сучасні персональні брандмауери тепер включають в себе додаткові можливості, такі як антивірусний моніторинг програмного забезпечення, а в деяких випадках і поведінковий аналіз та виявлення вторгнень для захисту пристрою. Однією з особливостей персонального брандмауера є можливість блокувати трафік не тільки за портами і протоколами, а й за програмами, які його генерують, тобто в межах хоста ми можемо дозволяти або забороняти певним застосункам отримувати доступ до локальної чи глобальної мережі. Найбільш популярні комерційні брандмауери це BlackICE та Cisco Security Agent. В сегменті SOHO PC-cillin, ZoneAlarm та персональний брандмауер

Symantec є одними з найбільш популярних пропозицій на ринку. Брандмауер Windows також входить до числа персональних брандмауерів, є одним із базових компонентів операційних систем сімейства Windows, починаючи з Windows XP із пакетом оновлень 2 [28].

Персональні брандмауери, як видно з назви побудовані для захисту одного окремого персонального комп'ютера користувача, але при цьому вони можуть застосовуватися і в SOHO мережах через простоту їх встановлення та налаштування, при цьому брандмауер повинен бути встановлений на кожному з хостів [28]. Але при збільшенні кількості хостів, контроль дотримання політик безпеки та коректності налаштувань міжмережевого екрану сильно ускладнюється. Така проблема актуальна для більшості персональних брандмауерів, оскільки частіше за все вони не мають засобів централізованого керування. А ті рішення, які підтримують централізоване керування, знаходяться на межі персональних та корпоративних мережевих брандмауерів.

Мережеві брандмауери призначені для захисту від шкідливого трафіку цілих мереж. Такі брандмауери бувають у двох основних формах: спеціалізований пристрій або набір програмного забезпечення брандмауера, що встановлюється на операційну систему хоста (зазвичай мережевого шлюза). Приклади мережевих брандмауерів на базі пристроїв включають Cisco Secure Firewall [29], брандмауери NetScreen Juniper та Fortigate. Популярні брандмауери, засновані на програмному забезпеченні, включають брандмауер Check Point-1 NG або NGX, брандмауер Microsoft ISA Server, IPTables на базі Linux [30], а також фільтр пакетів BSD та PFSense [31]. У минулому операційна система Sun Solaris постачалася з корпоративним брандмауером SunScreen. З випуском Solaris 10, Sun почав поєднувати брандмауер IP-фільтра з відкритим кодом (IPF) як альтернативу SunScreen [28].

Програмні брандмауери встановлюються на універсальні операційні системи. Зазвичай пакет програмного забезпечення брандмауера включає модулі ядра або драйвери, патчі, а також зміни конфігурації, які необхідно застосувати, щоб захистити базову операційну систему від атак і забезпечити правильну роботу брандмауера. Основна перевага таких брандмауерів полягає в тому, що пристрій, на якому встановлено брандмауер, може мати багатоцільове використання. Наприклад, брандмауер також може бути встановлено на хості, який також є сервером доменних імен (DNS), або спам фільтром. Програмні брандмауери пристосовуються для багатоцільових ролей набагато простіше, ніж спеціалізовані апаратні брандмауери.

Вагомим недоліком програмних брандмауерів є необхідність врахувати потенційні вразливості базової операційної системи. Оскільки нові вразливості виявляються в різних аспектах операційної системи, адміністратору необхідно приймати рішення про встановлення патчів постачальника або відмовлятися від патчів через можливий негативний вплив на роботу брандмауера спираючись на інформаційні бюлетені постачальників [32]. Ще одним важливим недоліком програмних брандмауерів є можливість виникнення проблем через погану сумісність брандмауера та операційної системи, але зазвичай цього не виникає, якщо брандмауер вбудований розробниками в операційну систему та постачається разом із нею. Також до недоліків програмних брандмауерів можна віднести необхідність часто виконувати оновлення базової операційної системи та потенційно нижча продуктивність. Програмні брандмауери мають нижчу ефективність у порівнянні з апаратними, через відсутність оптимізації апаратної складової та базового програмного забезпечення операційної системи під роль брандмауера.

Основна ж перевага програмних брандмауерів - це можливість більш простої заміни апаратного забезпечення у випадку виходу його з ладу.

Програмний брандмауер може бути корисними в якості недорогого варіанта для користувачів SOHO та SMB мереж.

Апаратні брандмауери – це міжмережеві екрани, які інтегровані з вбудованим обладнанням. Зазвичай такі брандмауери використовують спеціальну операційну систему, яка оптимізована для використання з конкретною апаратною платформою.

У більшості випадків апаратні брандмауери мають кращі показники швидкодії та надійності порівняно з програмними брандмауерами завдяки наявності спеціалізованої базової операційної системи та використання спеціалізованих процесорів та інтегральних схем (ASIC) для обробки мережевих даних. Різниця у функціоналі між програмними та апаратними брандмауерами в основному мінімальна, але апаратні брандмауери можуть мати функціонал підкріплений апаратними можливостями, який потребував би багато обчислювальних ресурсів при використанні програмної реалізації.

Також однією з головних переваг апаратних брандмауерів є технічна підтримка від виробника [33]. І у випадку виникнення проблем у роботі системи може бути складно знайти першопричину проблеми та отримати адекватну технічну підтримку. У апаратного ж брандмауера за апаратні та програмні компоненти відповідає один постачальник, при цьому однакова апаратна та програмна платформа для усіх користувачів дозволяє значно швидше виявляти проблеми в роботі системи та прорахунки при її проектуванні. Іншими перевагами, характерними для апаратних брандмауерів, є загальна краща продуктивність, більш висока безпека операційної системи брандмауера [34].

До недоліків апаратних брандмауерів можна віднести достатньо короткий життєвий цикл обладнання – після закінчення терміну технічної підтримки платформа перестає отримувати оновлення та розширення функціоналу, крім того, навіть елементарне встановлення програмного

забезпечення може стати проблемою після закінчення терміну підтримки, що фактично означає необхідність заміни брандмауера на більш нову версію. При цьому альтернативні варіанти використання апаратної платформи зазвичай відсутні.

Також до недоліків апаратних брандмауерів можна віднести складність та обмеженість розширення функціоналу. Зазвичай виробники надають певний набір пакетів розширення, але можливості програмних брандмауерів щодо розширення функціоналу значно ширші, оскільки апаратна платформа та операційна система загального призначення не обмежують набір програмного забезпечення, що має бути встановлене.

Ще одна особливість, яка була впроваджена в мережевих брандмауерах – це можливість глибокої перевірки пакетів. Брандмауер може визначати вимоги трафіка не просто, переглядаючи інформацію рівня 3 та рівня 4, але й шляхом детального поглиблення в дані прикладного рівня, щоб брандмауер міг приймати рішення про те, як найкраще керувати потоком трафіку. Ця еволюція в дизайні та можливостях брандмауера призвела до розробки нового продукту брандмауера - інтегрованого брандмауера [28].

Також брандмауери можна класифікувати за принципом роботи:

- Фільтр пакетів
- Шлюз мережевого рівня
- Шлюз прикладного рівня.

Брандмауери не тільки захищають мережі від численних загроз з Інтернету, але і також захищають Інтернет від шкідливих дій користувачів або програм у приватних мережах. Брандмауери рівномірно розподіляють пропускну здатність інтернет підключення або приватної мережі, щоб авторизовані користувачі мали можливість працювати з мережевими ресурсами [27]. Тоді як без брандмауерів більша частина пропускної

можливості мережі споживатиметься марним або шкідливим трафіком з Інтернету.

Міжмережевий екран виконує аналіз мережевих пакетів за такими параметрами:

- IP-адреса джерела
- Порт джерела
- IP-адреса призначення
- Порт призначення
- IP-протокол

Інформація про заголовки пакетів (тобто порядкові номери, контрольні суми, прапорці даних, інформація про корисне навантаження тощо)

Для прийняття рішення про фільтрування брандмауер повинен перевіряти кожен пакет обох напрямках та на всіх інтерфейсах, а правила контролю доступу повинні існувати для всіх пакетів, які будуть перевірятися. Це може призвести до виникнення проблеми, коли рух пакетів в одному напрямку, дозволений, але відсутній дозвіл на проходження пакетів відповідей [28] [31].

Сучасні брандмауери поєднують перевірку пакетів та перевірку стану з’єднання між хостами, тобто окрім перевірки структури пакету, брандмауер перевіряє стан з’єднання (або етап його встановлення) та його наявність. Перевагою такої перевірки, є те, що після того, як з'єднання було встановлено та дозволено (після його відповідної перевірки), як правило, не потрібно визначати правило, яке дозволяє дозволити зворотню передачу, оскільки брандмауер знає, що повинна надійти відповідь [28]. Це значно спрощує створення правил для міжмережевого екрану (оскільки немає необхідності прописувати окремі правила для прийому пакетів-відповідей) і дає можливість базуючись на стані з’єднання динамічно дозволяти або блокувати трафік в залежності від стану з’єднання.

На думку Stewart [27] та Noonan [28],найбільш ефективними для обробки та популярними форматами звітів брандмауера є:

- Повідомлення в консолі: Це простий процес подання сповіщення на консоль. Недолік цього методу тривоги полягає в тому, що він вимагає, щоб хтось активно контролював консоль, щоб знати, що спрацював сигнал, але це достатньо простий спосіб передавати інформацію до програм, що виконують автоматичний моніторинг.
- SNMP-повідомлення: Простий протокол управління мережею (SNMP) може використовуватися для генерації пасток, які надсилаються до системи управління мережею (NMS), яка контролює брандмауер, може використовуватися для взаємодії з різним ПЗ.
- Повідомлення електронною поштою або месенджером: брандмауер просто надсилає електронне повідомлення на відповідну адресу електронної пошти або у месенджер.

Далі розглянемо найбільш популярні брандмауери з відкритим вихідним кодом для Unix подібних систем Iptables та Packet Filter.

Iptables

Брандмауер iptables був розроблений проектом Netfilter і став доступним у складі Linux з виходом ядра Linux 2.4 у січні 2001 року.

За ці роки iptables переріс у грандіозний брандмауер із функціоналом притаманним комерційним брандмауерам. Наприклад, iptables пропонує детальне відстеження стану з’єднання, перевірку пакетів прикладного рівня, обмеження швидкості з’єднання та потужний механізм для налаштування політики фільтрації. Усі основні дистрибутиви Linux включають iptables, і багато дистрибутивів пропонують користувачеві налаштувати політики iptables прямо з інсталятора [35].

Iptables має достатньо широкі можливості для конфігурування політик фільтрації, але єдиний вбудований спосіб обміну інформацією з іншими програмами це ведення журналів в файл. Тому взаємодія з іншими програмними модулями можлива тільки у випадку, якщо програма вміє обробляти файли журналу iptables.

Packet Filter

Розроблений в рамках проекта OpenBSD що має високу швидкість роботи та зручність в конфігуруванні, але використовується в багатьох BSD системах та навіть MacOS. Включає власне фільтр пакетів та утиліту для конфігурування. Працює фільтр в контексті ядра операційної системи. Керування відбувається в першу чергу за рахунок системних викликів, тому утиліта керування слугує лише додатковим інструментом. Архітектурно не передбачає багатопотокову обробку, але відсутність блокувань позитивно впливає на швидкодію. Packet Filter вміє пропускати непотрібні перевірки та оптимізувати порядок їх проходження під час проходження списку правил за рахунок чого підвищується швидкість обробки [31].

1.4.2 Системи виявлення вторгнень

Системи виявлення вторгнень IDS використовуються для виявлення шкідливих або підозрілих дій шляхом моніторингу мережевого трафіку та виявлення шкідливої поведінки. Це може варіюватися від моніторингу шкідливого коду, мережевих атак або будь-якого іншого типу шкідливої активності, яка може бути присутня в мережі.

Існує два основних типи систем виявлення вторгнень: мережеві та на базі хоста. Мережеві (NIDS) призначені для моніторингу мережі на предмет шкідливої активності шляхом аналізу трафіку в мережі. Вони використовується для виявлення потенційних загроз безпеці, таких як

шкідливий код, хробаки, віруси та DoS-атаки [36]. NIDS можуть використовуватися для виявлення таких подій, як сканування портів, запуск та завантаження шкідливого коду або будь-який інший тип шкідливої активності, який може бути присутнім у мережі.

Системи виявлення вторгнень на основі хоста (HIDS) призначені для встановлення на окремих комп'ютерах для виявлення будь-якої шкідливої активності, яка може бути присутня на хості. Це включає виявлення шкідливого коду, хробаків, вірусів та DoS-атак. HIDS також можна використовувати для моніторингу активності користувача на комп'ютері, наприклад, для реєстрації натискань клавіш або знімків екрану.

Обидва типи систем виявлення вторгнень мають свої переваги та недоліки. Перевага NIDS полягає в здатності виявляти шкідливу активність в мережі в режимі реального часу, що робить їх ефективним інструментом для використання адміністраторами безпеки. Однак Pradhan [37] вважає, що NIDS може бути важко налаштувати та підтримувати, і це може бути ресурсоємним. HIDS простіші у встановленні та підтримувати, але йому не вистачає можливостей виявлення NIDS у режимі реального часу.

Виявляючи шкідливу активність на ранній стадії, адміністратори безпеки можуть вжити заходів для запобігання поширенню шкідливого коду або обмеження шкоди, заподіяної мережевою атакою. Крім того, IDS може допомогти організаціям заздалегідь визначити потенційні загрози безпеці, дозволяючи їм вжити необхідних заходів для захисту своїх мереж.

Коротке порівняння популярних сучасних засобів захисту мереж наведено у таблиці 1.2.

Таблиця 1.2 – Порівняння засобів захисту комп’ютерних мереж

Критерій	Iptables	Packet Filter	Snort	Suricata	PFSense	Zeek	Windows Firewall	SolarWinds SEM	McAfee	Cisco Firepower	FortiGate
Апаратне рішення	-	-	-	-	+/-	-	-	-	-	+	+/-
OpenSource	+	+	+	+	+	+	-	-	-	-	-
Виявлення на основі сигнатур	-	-	+	+	+	+	-	+	+	+	+
Виявлення аномалій	-	-	*	*	*	-	-	-	+	+	+
Firewall	+	+	-	-	+	-	+	-	+	+	+
Глибоке дослідження пакетів	-	-	-	-	-	-	-	-	+	+	+
Комплексне рішення	-	-	-	-	+	-	-	-	+	+	+
Аналіз системних подій	-	-	-	-	-	-	-	+	+	-	-

Далі наведено більш детальний опис систем виявлення вторгнень представлених у таблиці.

SolarWinds Security Event Manager

SolarWinds Security Event Manager (SEM) - це система управління інформацією про безпеку та події (SIEM), розроблена SolarWinds. SEM-це комплексне рішення для забезпечення безпеки, покликане допомогти організаціям виявляти, аналізувати кіберзагрози і реагувати на них в режимі

реального часу. Об'єднує та співвідносить події безпеки з кількох джерел, включаючи мережеві пристрої безпеки, операційні системи, програми та бази даних, забезпечуючи моніторинг і оповіщення в режимі реального часу.

IDS, що працює на Windows, SolarWinds Event Manager може реєструвати повідомлення, створені не лише комп’ютерами Windows, а й комп’ютерами Mac-OS, Linux та Unix, Має як риси притаманні HIDS, так і NIDS. Сама програма виконує лише аналіз подій, отриманих від іншої системи, наприклад SNORT.

Розширені аналітичні можливості SEM дозволяють організаціям виявляти закономірності і тенденції в подіях безпеки, що дозволяє їм краще розуміти свою систему безпеки і активно усувати потенційні вразливості [38].

Система налаштована з понад 700 правилами для кореляції подій. Це дозволяє йому не тільки виявляти підозрілі дії, а й автоматично виконувати виправлення. SolarWinds Event Manager — це комплексний інструмент безпеки мережі.

Zeek

Zeek - це потужне рішення NIDS, яке надає організаціям можливість виявляти шкідливі дії у мережах. Призначене для виявлення та реагування на широкий спектр загроз, включаючи шкідливий мережевий трафік, зміни конфігурації системи та інші підозрілі дії. Zeek здатний виявляти як відомі, так і невідомі загрози, а також надавати детальну інформацію про джерело та характер загрози.

Zeek складається з декількох компонентів, включаючи систему захоплення мережевих пакетів, механізм аналізу журналів і засновану на правилах систему визначення загроз і реагування на них. Система захоплення пакетів відповідає за захоплення мережевого трафіку, тоді як механізм

аналізу журналів використовується для виявлення підозрілих дій. Система, заснована на правилах, визначає дії, які необхідно вжити при виявленні загрози [39].

На додаток до перерахованого вище, програмне забезпечення Zeek IDS використовує як відстеження подій, що відбуваються в мережі, так і політики безпеки. Це рішення з відкритим кодом, що означає, що його можна легко налаштувати відповідно до потреб будь-якої організації. Zeek також має високу масштабованість, що дозволяє використовувати його у великомасштабних мережах. Дана система виявлення вторгнень доступна для Unix подібних систем. Нарешті, Zeek відрізняється високою ефективністю і вимагає мінімальних ресурсів для роботи. [40]

Snort

Провідний інструмент NIDS, Snort, безкоштовний у використанні, і це одна з небагатьох систем виявлення вторгнень, яку можна встановити на Windows. Snort – це не тільки детектор вторгнення, але й реєстратор і аналізатор пакетів. Однак головною функцією цього інструменту все ж таки є виявлення мережевих вторгнень.

Snort має конфігурацію на основі правил. Основні правила можуть бути завантажені з веб-сайту snort, і далі налаштовані у відповідно до потреб мережі [41]. Snort виконує виявлення вторгнень за допомогою методів на основі аномалій і сигнатур.

Snort IDS-це ефективна система безпеки, що володіє безліччю функцій, які роблять її ефективним інструментом для виявлення шкідливих дій. Вона здатна аналізувати мережевий трафік для виявлення підозрілих дій, таких як сканування портів, сканування мережі, атаки відмови в обслуговуванні (DoS) та переповнення буфера [20]. Також є можливість виявляти шкідливі дії в зашифрованому трафіку, наприклад, віртуальні приватні мережі (VPN). Система Snort також здатна виявляти невідомі

шкідливі дії, аналізуючи трафік на предмет аномальних або підозрілих шаблонів [42].

До основних недоліків можна віднести використання в основному сигнатурних методів, що означає, що вона обмежена виявленням лише відомих шкідливих дій, представлених правилами, встановленими в системі Snort (інші методи виявлення не можна вважати достатньо надійними, щоб використовувати їх окремо від сигнатурних). Крім того, такі методи виявлення витрачають достатньо багато системних ресурсів, що може призвести до навантаження на систему, якщо вона не налаштована належним чином.

Suricata

Надійний засіб виявлення мережевих загроз Suricata є однією з основних альтернатив Snort. Suricata була розроблена як більш ефективна та економічна альтернатива існуючим комерційним рішенням IDS/IPS. Однак, основною відмінністю, що робить цей інструмент кращим за snort, є можливість збору даних на прикладному рівні. Suricata здатна виявляти широкий спектр атак, включаючи атаки прикладного рівня, зловмисне програмне забезпечення, вторгнення в мережу та загрози нульового дня. Система також здатна виявляти шкідливий трафік, який не покривається традиційними сигнатурами. Багатопотоковість в Suricata також реалізована краще та ефективніше ніж в SNORT.

Suricata легко налаштовується і може бути адаптована до будь-якого мережевого середовища. Підтримує безліч протоколів, включаючи TCP, UDP, ICMP, HTTP, FTP, SSH, TLS/SSL та багато інших, та може бути розгорнута як автономна система або як частина більшої інфраструктури безпеки. Також достатньо серйозною перевагою є вбудована підтримка синтаксису сигнатурних правил для Snort, що дає можливість використовувати спільні бази [43].

McAfee

McAfee Network Security Platform це потужний набір інструментів, призначених для захисту цілісності корпоративних мереж і збереження їх даних. Платформа забезпечує розширений захист у режимі реального часу від шкідливих атак, шкідливих програм та інших загроз. Він також пропонує розширену аналітику для виявлення підозрілих дій та реагування на них. Крім того, NSP дозволяє організаціям керувати мережевим трафіком і контролювати доступ до конфіденційних ресурсів [44].

Рішення від McAfee менеджер мережевої безпеки (NSM), який забезпечує централізоване представлення мережі та надає інформацію про інциденти безпеки в режимі реального часу. Крім того, платформа включає систему контролю доступу до мережі (NAC), яка може виявляти та блокувати несанкціоноване підключення пристроїв до мережі. NSP також включає систему захисту від вторгнень (IPS) та брандмауер для захисту від мережевих атак.

Платформа надає повний набір інструментів для забезпечення безпеки мереж, включаючи розширений захист в режимі реального часу від шкідливих атак, шкідливих програм та інших загроз. Крім того, платформа включає в себе розширену аналітику для виявлення підозрілої активності і реагування на неї, а також засоби управління мережевим трафіком і доступом до конфіденційних ресурсів. Як результат, організації можуть покладатися на McAfee NSP для забезпечення надійної та ефективної безпеки.

Ефективність McAfee NSP підтверджується великою кількістю досліджень. Наприклад, платформа була протестована та сертифікована Національним інститутом стандартів і технологій США (NIST) на предмет її безпеки та надійності. Крім того, McAfee вклала значні кошти в розробку

своїх технологій безпеки. В результаті платформа забезпечує високий рівень захисту від шкідливих атак та інших загроз [45].

PFsense

Програмний комплекс, що включає брандмауер та систему виявлення вторгнень, що призначений для розміщення на шлюзі та базується на операційній системі FreeBSD. Також даний продукт пропонується у вигляді апаратних рішень Netgate з достатньо широкою лінійкою починаючи від домашніх рішень і закінчуючи потужними серверними [46].

Продукт має широкі можливості для роботи з мережевим трафіком. Можна виконувати повне логування трафіку з подальшим аналізом. Елементи керування стандартним програмним забезпеченням доступні через веб-інтерфейс.

В якості брандмауера використовується Packet Filter який є стандартним брандмауером в BSD системах. Роль системи виявлення вторгнень же виконує Suricata достатньо популярна opensource система виявлення вторгнень. Окрім можливостей фільтрації трафіка даний програмний комплекс передбачає можливість встановлення великої кількості плагінів, що розширюють його функціонал і дають можливість налаштовувати VPN, проксі-сервери та сервери аутентифікації. Оскільки дане рішення є комплексним, то основні переваги і недоліки наслідуються із базових продуктів на основі яких система побудована [47].

Fortingate

Система виявлення вторгнень Fortinet (FIDS) - це комплексне рішення для забезпечення безпеки, яке поєднує в собі виявлення на основі сигнатур і виявлення аномалій, щоб забезпечити всебічний аналіз мережевого трафіку. Сімейство фізичних та віртуальних рішень FortiGate компанії Fortinet в

області уніфікованого управління загрозами включає такі функції забезпечення безпеки, як мережеві екрани, засоби запобігання вторгненням, веб-фільтри та захист від шкідливого програмного забезпечення або небажаної пошти. У сімейство входять рішення для підприємств і філій, а також рішення для підприємств, центрів обробки даних, та інтернет-провайдерів [48]. Компанія також реалізує мережеві екрани нового покоління (Next Generation Firewalls, NGFW), що поєднують у собі можливості міжмережевого екрану, VPN, системи запобігання вторгненням та інші функції безпеки [49].

Пристрої FortiGate, починаючи з 2013 року, включають функцію мережевого екрану, призначену для роботи у внутрішніх мережах та працює на мікропроцесорах спеціального призначення (ASIC) [50]. Також на платформі Amazon Web Services існують віртуальні засоби FortiGate, які можна використовувати для захисту хмарної інфраструктури [51]. Рішення на основі ASIC дозволяють робити продуктивні пристрої з високою енергоефективністю, що значно підвищує їх конкурентоспроможність у великих мережах у порівнянні з програмними рішеннями, оскільки програмні рішення на обладнанні загального призначення не зможуть забезпечити достатню швидкодію, що робить їх використання для мереж з великим навантаженням як мінімум нераціональним.

FortiGate NGFW забезпечує високоефективний захист як в локальній, так і в глобальній мережі, а також надійну аутентифікацію користувача. Він пропонує безліч можливостей контролю та примусового виконання, таких як управління додатками, IPS / IDS, веб-фільтрація та багато іншого. FortiGate NGFW також пропонує розширені можливості виявлення атак і реагування на них. Зокрема рішення від Fortinet здатні виявляти широкий спектр загроз, включаючи шкідливе програмне забезпечення, мережевих черв'яків та віруси. Шкідливі дії, такі як несанкціонований доступ, ексфільтрація даних

та віддалене керування системою також не є проблемою для систем виявлення вторгнень Fortinet і легко блокуються. І звісно наявні механізми для швидкого виявлення і успішного блокування DoS-атак, що дозволяє забезпечити високу надійність та відмовостійкість мережі. Крім того, FIDS може виявляти атаки на мережевому рівні, такі як сканування портів та прослуховування пакетів та несанкціоновані точки доступу (RAPs). Наявна можливість і виявлення атак прикладного рівня, таких як SQL ін’єкція, міжсайтові сценарії, атаки переповнення буфера а також атаки "людина посередині " (MITM) [52].

До основних переваг рішень Fortinet можна віднести комплексність таких рішень - в одному пристрої сконцентрована максимально доступний набір методів виявлення широкого спектру атак, включаючи сигнатурні методи та методи виявлення аномалій. Окрім того Fortinet розробляє свої пристрої так, щоб їх встановлення та базове налаштування було якомога простішим і не залежало від розміри чи масштабу мережі.

Не позбавлені пристрої FortiGate і недоліків - два основних недоліки для таких рішень це закритість вихідного коду та відсутність сумісності з іншими аналогічними рішеннями, що потребує його інтеграції як окремого компонента і налаштування мережі або її фрагмента саме під роботу із рішеннями Fortinet.

Cisco Secure Firewall

Cisco ASA – лінійка сучасних багатофункціональних пристроїв для захисту локальних мереж від зовнішніх атак та вторгнень. Основна функція мережевого екрану Cisco (firewall) – захист мережі від вторгнень, вірусів, спаму, шпигунських програм, фільтрація трафіку користувачів за контентом [53]. Програмне забезпечення Cisco Secure Firewall засноване на Linux, але включає достатньо широкий набір програмних модулів, при цьому система повністю перероблена під архітектуру пристрою та адаптовано для ролі

мережевого екрану.

Cisco Secure Firewall також може виявляти шкідливу активність, яку інші системи виявлення вторгнень не бачать, оскільки вона має здатність виявляти аномальні схеми руху та поведінку. Крім того, в офіційній документації Cisco [54], Cisco Secure Firewall може виявити введення шкідливого коду, атаки відмови в обслуговуванні та сканування портів, підозрілу активність у мережі, таку як несанкціонований доступ та порушення цілісності даних.

Функціонал попередніх лінійок Cisco ASA та Firepower також включав в себе потужний мережевий екран та систему виявлення вторгнень, але при цьому апаратна платформа була менш потужною, що сильно обмежувало можливість розкриття повного функціоналу програмного забезпечення [55].

1.5 Блокчейн технології в захищених розподілених системах

Блокчейни є цифровими книгами записів з можливістю явної перевірки, що захищені від фальсифікацій та реалізовані розподіленим способом (тобто без центрального сховища) і, як правило, без центрального управління. На своєму базовому рівні вони дозволяють спільноті користувачів реєструвати транзакції в спільній книзі записів всередині цієї спільноти, так, що за нормальної роботи мережі блокчейн жодна транзакція не може бути змінена після публікації.

У 2008 році ідея блокчейну була поєднана з кількома іншими технологіями та обчислювальними концепціями для створення сучасних криптовалют: електронні гроші, захищені за допомогою криптографічних механізмів, замість центрального сховища чи органу управління. Першою подібною криптовалютою на основі блокчейну був біткойн.

Основні ідеї технології блокчейн з’явилися наприкінці 1980-х - на початку 1990-х. У 1989 році Леслі Лампорт розробила протокол Paxos, а в 1990 році подала статтю Парламент за сумісництвом [56] до “ACM Transactions on Computer Systems”. Стаття була опублікована у випуску за 1998 рік. У статті описується модель консенсусу для досягнення згоди щодо результату в мережі комп'ютерів, де самі комп'ютери або мережа можуть бути ненадійними. У 1991 р. Підписаний ланцюжок інформації використовувався як електронна книга записів для цифрового підпису документів таким чином, що легко можна було показати, що жоден з підписаних документів у колекції не був змінений [57]. Ці концепції були поєднані та застосовані до електронних коштів у 2008 році та описані в роботі, Біткойн: Електронна система готівкових платежів однорангової мережі [58], яка була опублікована автором під псевдонімом Сатоші Накамото, який пізніше в 2009 році створив мережу блокчейн криптовалюти біткоїн. Документ Накамото містив концепт, якого дотримуються більшість сучасних систем криптовалют (хоча з варіаціями та модифікаціями). Біткойн був лише першим із багатьох додатків на основі блокчейну.

Використання блокчейну дозволило реалізувати біткойн розподіленим способом, таким чином, що жоден користувач не контролював електронні кошти і не існувало єдиної точки відмови, що сприяло його активному використанню. Основною перевагою такого рішення було забезпечення прямих транзакцій між користувачами без необхідності довіреної третьої сторони. Це також дозволило делегувати випуск нової криптовалюти за встановленим алгоритмом тим користувачам, яким вдається публікувати нові блоки та вести копії книги записів. Таких користувачів у Bitcoin мережі (а також мережіх інших криптовалют) називають майнерами. Автоматизована оплата майнерам дозволила запровадити розподілене адміністрування системи без необхідності організації. Використовуючи блокчейн та підтримку на основі консенсусу,

було створено механізм самоконтролю, який забезпечував додавання до блокчейну лише валідованих транзакцій та блоків.

У біткоїн мережі блокчейн дозволив користувачам бути псевдо-анонімними. Це означає, що користувачі анонімні, а ідентифікатори їхніх облікових записів - ні; крім того, усі транзакції є загальнодоступними. Це фактично дозволило біткойну пропонувати псевдо-анонімність, оскільки облікові записи можна створювати без будь-якого процесу реєстрації, ідентифікації або авторизації [59].

Оскільки біткойн був анонімним, дуже важливо було мати механізми для створення довіри в середовищі, де неможливо легко ідентифікувати користувачів. До використання технології блокчейн ця довіра, як правило, передавалася через посередників, яким довіряють обидві сторони. Без довірених посередників необхідна довіра всередині мережі блокчейн, що забезпечується чотирма ключовими характеристиками технології, описаними нижче:

- *Книга записів* – технологія використовує книгу записів, що працює лише на додавання, щоб забезпечити повну історію транзакцій. На відміну від традиційних баз даних, транзакції та значення збережені в блокчейні не замінюються.
- *Безпека* – блокчейн криптографічно захищений, гарантуючи, що дані, що містяться всередині книги, не піддаються фальсифікації, і при цьому дані в книзі є перевіреними.
- *Відкритість* – книга ділиться між кількома учасниками. Це забезпечує прозорість для всіх вузлів у мережі блокчейн.
- *Розподіленість* – блокчейн можна розподіляти, що дозволяє масштабувати кількість вузлів мережі, щоб зробити її більш стійкою до атак з боку користувачів. Збільшення кількість вузлів, зменшує здатність зловмисника впливати на консенсус-протокол, що використовується блокчейном.

Для мереж блокчейн, які дозволяють будь-кому анонімно створювати облікові записи та брати участь (так звані мережі блокчейн без контролю доступу), ці можливості забезпечують рівень довіри між сторонами без попереднього знання сторін одна про одну. Ця довіра може дозволити приватним особам та організаціям здійснювати прямі транзакції, що може пришвидшення проведення таких операцій, та зниження витрат на їх проведення. Для мережі блокчейн, яка більш жорстко контролює доступ (так звані мережі блокчейн з контролем доступу), де певна довіра може бути присутня серед користувачів, такі можливості допомагають зміцнити довіру.

Важливою складовою технології блокчейн є використання криптографічних хеш-функцій для багатьох операцій. Хешування – це метод застосування криптографічної хеш-функції до даних, яка обчислює відносно унікальний результат (називається дайджест повідомлення, або просто дайджест) для вводу майже будь-якого розміру (наприклад, файлу, тексту чи зображення). Це дозволяє підтверджувати те, що данні не змінювалися та цілісні. Навіть найменша зміна вхідного сигналу (наприклад, зміна одного біта) призведе до зовсім іншого дайджесту виведення.

Криптографічні хеш-функції мають такі важливі властивості захисту:

- Вони стійкі до знаходження прообразу. Це означає, що вони односторонні. Отже, неможливо обчислити правильне вхідне значення знаючи деяке вихідне значення (наприклад, для даного дайджесту знайти x таке, що хеш (x) = дайджест).
- Вони мають другу стійкість до прообразу. Це означає, що неможливо знайти вхід, який хешується до певного конкретного виходу. Тобто, криптографічні хеш-функції спроектовані таким чином, що з урахуванням конкретного входу, обчислювально неможливо знайти другий вхід, який видає той самий результат (наприклад, з урахуванням x , знайти y такий, що хеш (x) = хеш (y)). Єдиний

доступний підхід - це вичерпний пошук у вхідному просторі, але це обчислювально неможливо зробити за адекватний час з хоча б якимись шансами на успіх.

- Вони стійкі до колізій. Це означає, що неможливо знайти два входи, які мають хеш з одним і тим же виходом. Тобто, обчислювально неможливо знайти будь-які два набори вхідних даних, які видають однаковий дайджест (наприклад, знайти *x* та *y*, для яких *хеш* (*x) = хеш (y*)).

Конкретною криптографічною хеш-функцією, яка використовується у багатьох реалізаціях блокчейну, є алгоритм безпечного хешування (SHA) із вихідним розміром 256 біт (SHA-256). Багато комп'ютерів підтримують цей алгоритм в апаратному забезпеченні, що робить його швидким для обчислень. SHA-256 має вихід 32 байти (1 байт = 8 біт, 32 байти = 256 біт), як правило, відображається як шістнадцятковий рядок із 64 символів. Це означає, що існує $2^{256} \approx 10^{77}$ можливі значення дайджесту. Алгоритм для SHA-256, як і інших, визначений Федеральним стандартом обробки інформації (FIPS) 180-4 [60]. Веб-сайт безпечного хешування NIST [61] містить специфікації FIPS для всіх схвалених NIST алгоритмів хешування. Оскільки існує нескінченна кількість можливих вхідних значень і кінцева кількість можливих вихідних дайджест-значень, можливе, але дуже малоймовірно, зіткнення, коли *хеш* (*x*) = *хеш* (*y*) (тобто хеш двох різних входів видає однаковий дайджест). В такому випадку кажуть, що SHA-256 стійкий до колізій, оскільки, щоб знайти колізію в SHA-256, потрібно виконати алгоритм в середньому приблизно 2^{128} разів.

У мережі блокчейн криптографічні хеш-функції використовуються для багатьох задач, таких як:

- Генерація адреси;
- Створення унікальних ідентифікаторів;

- Захист даних блоку - вузол публікації буде хешувати дані блоку, створюючи дайджест, який буде зберігатися в заголовку блоку;
- Захист заголовка блоку - вузол публікації буде хешувати заголовок блоку.

Оскільки заголовок блоку включає хеш-представлення даних блоку, самі дані блоку також захищені, коли дайджест заголовка блоку зберігається в наступному блоці.

У технології блокчейн використовується багато сімейств криптографічних хеш-функцій (SHA-256 не єдина), таких як Keccak (який був обраний NIST переможцем у конкурсі на створення стандарту хешування SHA-3), а також RIPEMD-160 [62].

Транзакція в blockchain являє собою взаємодію між сторонами. Наприклад, у випадку з криптовалютами транзакція являє собою передачу криптовалюти між користувачами мережі блокчейн. Кожен блок у блокчейні може містити одну або більше транзакцій. Для деяких реалізацій блокчейну постійне надходження нових блоків (навіть при відсутності транзакцій) є критичним для підтримки безпеки мережі блокчейн; Завдяки постійному надходженню нових блоків, які публікуються, зловмисник не може наздогнати та створити новий довший, змінений ланцюжок блоків.

Дані, що містять транзакцію, можуть бути різними для кожної реалізації блокчейну, однак механізм транзакцій здебільшого однаковий. Інформація, що надсилається в мережу блокчейн, може включати адресу відправника (або інший відповідний ідентифікатор), відкритий ключ відправника, цифровий підпис, входи транзакцій та виходи транзакції, або дані, що записані в транзакціях.

Технологія блокчейн використовує криптографію асиметричного ключа. Криптографія асиметричного ключа використовує пару ключів: відкритий та закритий ключі, які математично пов’язані між собою. Відкритий ключ робиться загальнодоступним, не знижуючи безпеку

процесу, але приватний ключ повинен залишатись таємним, якщо дані мають зберігати свій криптографічний захист. Незважаючи на те, що між двома ключами існує взаємозв'язок, приватний ключ не може бути ефективно визначений на основі знання відкритого ключа. Можна зашифрувати за допомогою приватного ключа, а потім розшифрувати за допомогою відкритого ключа. Крім того, можна зашифрувати за допомогою відкритого ключа, а потім розшифрувати за допомогою закритого ключа.

Криптографія з асиметричним ключем забезпечує довірчі відносини між користувачами, які не знають або довіряють один одному, забезпечуючи механізм перевірки цілісності та достовірності транзакцій, одночасно дозволяючи транзакціям залишатися відкритими. Для цього транзакції мають «цифровий підпис». Це означає, що приватний ключ використовується для шифрування транзакції таким чином, що кожен, хто має відкритий ключ, може його розшифрувати. Оскільки відкритий ключ є у вільному доступі, шифрування транзакції із закритим ключем доводить, що той, хто підписав транзакцію має доступ до приватного ключа. Крім того, можна зашифрувати дані за допомогою відкритого ключа користувача, таким чином, що лише користувачі, що мають доступ до приватного ключа, можуть його розшифрувати. Недоліком є те, що криптографія асиметричного ключа часто обчислюється повільно.

Найбільш поширені варіанти використання криптографії асиметричних ключів у багатьох мережах блокчейн зазвичай включають:

- приватні ключі, що використовуються для цифрового підпису транзакцій;
- відкриті ключі, що використовуються для отримання адрес;
- відкриті ключі, що використовуються для перевірки підписів, згенерованих приватними ключами.

Криптографія асиметричного ключа надає можливість перевірити, що користувач, що передає значення іншому користувачеві, має приватний ключ, здатний підписати транзакцію.

Деякі мережі блокчейн можуть використовувати існуючу бізнес-інфраструктуру відкритих ключів для криптографії асиметричних ключів для надання облікових даних користувача - замість того, щоб кожен користувач мережі блокчейн керував власними асиметричними ключами. Це робиться за допомогою використання існуючих служб каталогів та використання цієї інформації в мережі блокчейн. Мережі блокчейн, які використовують існуючу службу каталогів, можуть отримати до неї доступ за допомогою існуючих протоколів, таких як «Легкий протокол доступу до каталогів» (LDAP) [63], і використовувати інформацію з каталогу вбудовано або імпортувати її в внутрішній центр сертифікації в мережі блокчейн.

Технологія Blockchain зазвичай забезпечує захист цілісності збережених даних, використовуючи як розподілене володіння, так і розподілену фізичну архітектуру. Розподілена фізична архітектура мереж блокчейн часто включає набагато більший набір комп'ютерів, ніж зазвичай використовується для розподіленої фізичної архітектури, що управляється централізовано. Зростаючий інтерес до розподіленого права власності на книги записів пояснюється можливою загрозою довіри, безпеки та надійності, пов’язаної з книгами записів з централізованою власністю. Переваги розподілених блочкейн систем перед системами з централізованим зберіганням даних наведені у таблиці 1.3.

Записи, що зберігаються централізовано, можуть бути втрачені або знищені. Користувач повинен довіряти, що власник правильно виконує резервне копіювання даних. Тоді як мережа блокчейн розподіляється, створюючи безліч резервних копій, які автоматично оновлюються та синхронізуються, таким чином важлива інформація завжди буде продубльована між основними вузлами. Ключовою перевагою технології

блокчейн є те, що кожен користувач може вести власну копію книги записів. Щоразу, коли нові повні вузли приєднуються до мережі блокчейн, вони звертаються, щоб виявити інші повні вузли та завантажують повну копію книги записів мережі блокчейн, ускладнюючи втрату або знищення записів.

Таблиця 1.3 – порівняння централізованого та розподіленого зберігання даних

Централізоване зберігання	**Розподілена блокчейн система**
Ризик втрати даних, уся відповідальність за резервне копіювання та цілісність на власнику обладнання	Безліч резервних копій розподілених між усіма учасниками блокчейн мережі значно підвищує надійність
Ризик однорідності архітектури, що спрощує проведення атак	Обладнання зазвичай різнорідне, що унеможливлює використання одних і тих же атак на всі вузли
Часто дані локалізовані в певному невеликому регіоні, ризик пошкодження усього обладнання одночасно.	Географічна різноманітність підвищує захищеність на випадок непередбачуваних подій
Можливість підробки транзакцій власником сховища, або не збереження деяких транзакцій	Криптографічна захищеність та цілісність даних гарантована алгоритмами блокчейн мережі

Книги записів, із централізованим зберіганням, можуть зберігатись в однорідній мережі, де все програмне та апаратне забезпечення, а також мережева інфраструктура можуть бути однаковими. Через цю характеристику загальна стійкість системи може бути знижена, оскільки успішна атака на одну частину мережі може бути застосована і до інших її частин. В той же час, мережа блокчейн – це різнорідна мережа, де програмне та апаратне забезпечення на пристроях і мережева інфраструктура

відрізняється. Через безліч відмінностей між вузлами в мережі блокчейн, атака на один вузол не гарантує успіху при повторній спробі на інший.

Книги записів, із централізованим зберіганням, можуть повністю розташовуватися на географічно обмежених територіях (наприклад, в межах однієї країни). Якщо в цьому місці відбулися перебої в роботі мережі, послуги, які залежать від такого сховища, можуть бути недоступними. На відміну ж від централізованого сховища мережа блокчейн може складатися з географічно віддалених вузлів, по всьому світу. Через це, і однорангова мережа блокчейн є стійкою до втрати будь-якого вузла, або навіть цілої групи вузлів.

Операції з централізованим сховищем не здійснюються прозоро і дійсність транзакцій для такого сховища легко може бути поставлена під сумнів, крім того користувач повинен довіряти тому, що власник перевіряє кожну отриману транзакцію. Блокчейн мережа ж автоматично перевіряє, чи всі транзакції дійсні, проводячи їх валідацію. Якщо зловмисний вузол передавав недійсні транзакції, інші виявляють його та ігнорують такі транзакції, перешкоджаючи поширенню недійсних транзакцій по всій мережі блокчейн.

Список транзакцій у журналі з централізованою власністю може бути не повним, і користувач повинен вірити, що власник включає всі дійсні транзакції, які були отримані. Тоді як мережа блокчейн зберігає всі прийняті транзакції в межах своєї розподіленої книги. Для побудови нового блоку необхідно зробити посилання на попередній блок. Якщо блок не містить посилання на коректний блок попередник – то інші вузли відхиляють такий блок .

Дані транзакцій у журналі з централізованою власністю можуть бути змінені – користувач повинен вірити, що власник не змінює минулі транзакції. Мережа блокчейн використовує криптографічні механізми, такі

як цифрові підписи та криптографічні хеш-функції, щоб забезпечити роботу сховища з очевидним механізмом захисту від фальсифікації записів.

Система, що знаходиться в центральній власності, може бути ненадійною або недостатньо захищеною і користувач повинен вірити, що пов'язані комп'ютерні системи та мережі отримують критичні виправлення безпеки. Мережа блокчейн, завдяки розподіленому характеру, не має централізованої точки атаки. Як правило, інформація про мережу блокчейн є загальнодоступною і не провокує крадіжки. Щоб атакувати користувачів мереж блокчейн, зловмисникові потрібно буде індивідуально націлюватись на них. Спроба атакувати сам блокчейн зустріне опір чесних вузлів, присутніх у системі. Якщо окремий вузол не буде виправлений, це вплине лише на цей вузол, а не на систему загалом.

Транзакції додаються до блокчейну, коли майнінговий вузол публікує блок. Блок містить заголовок блоку та дані блоку. Заголовок блоку містить метадані для цього блоку. Дані блоку містять перелік перевірених та достовірних транзакцій, які були подані до мережі блокчейн. Дійсність та достовірність забезпечується шляхом перевірки правильності форматування транзакції та завдяки тому, що постачальники цифрових активів у кожній транзакції підписали транзакцію. Це підтверджує, що постачальники цифрових активів для транзакції мали доступ до приватного ключа, який міг би підписати наявні цифрові активи або дані. Інші повні вузли перевірятимуть достовірність блоку та достовірність усіх транзакцій у опублікованому блоці і не прийматимуть блок, якщо він містить некоректні транзакції.

Блоки з'єднуються в ланцюжок оскільки кожен блок містить хеш-дайджест заголовка попереднього блоку, утворюючи таким чином блокчейн. Якщо раніше опублікований блок було змінено, він матиме інший хеш. Це, в свою чергу, призведе до того, що всі наступні блоки також матимуть різні

хеші, оскільки вони включають хеш попереднього блоку. Це дозволяє легко виявляти та відхиляти змінені блоки [64].

Коли користувач приєднується до мережі блокчейн, він погоджується з початковим станом системи. Це записано в єдиному попередньо налаштованому блоці, блоці генезису. Кожна мережа блокчейн має опублікований блок генезису, і кожен блок повинен бути доданий до блокчейну після нього, на основі узгодженої моделі консенсусу. Незалежно від моделі, однак, кожен блок повинен бути валідним і, отже, може бути перевірений незалежно кожним користувачем мережі блокчейну. Поєднуючи початковий стан та можливість перевіряти кожен блок, користувачі можуть самостійно узгодити поточний стан блокчейну. У випадку, якщо повний вузол отримує два дійсних ланцюжки, механізмом за замовчуванням у більшості мереж блокчейн запрограмовано так, що «довший» ланцюжок розглядається як правильний і буде прийнятий, тому, що для його отримання було виконано найбільше роботи.

Ключовою особливістю технології блокчейну є те, що немає необхідності мати довірену третю сторону, яка надає інформацію про стан системи – кожен користувач у системі може перевірити цілісність системи. Щоб додати новий блок до блокчейну, усі вузли повинні з часом прийти до спільної згоди, однак допускаються деякі тимчасові розбіжності.

Weizhi Meng у своїй роботі в основному обговорює застосовність блокчейну технології для пом'якшення проблем обміну даними і довірених обчислень у середовищі спільного виявлення вторгнень [65]. Також вони визначаили, що блокчейн має вплив на вдосконалення IDS, але при цьому не всі проблеми IDS можна вирішити за допомогою блокчейн технології. На їх думку, технологію блокчейн можна використовувати для вдосконалення продуктивності IDS, особливо CIDS в аспекті обміну даними та довіри обчислень. Технологія блокчейн – це новітнє рішення, яке все ще страждає від деяких притаманних проблем і обмежень:

- *Енергія та вартість.* Обчислювальна потужність є проблемою для використання блокчейну на думку Yli-Huumo [66]. Якщо розглянути в якості прикладу видобуток біткойнів, для цього потрібен високий рівень енергозатрат, щоб розраховувати та перевіряти транзакції. Ван і Лю [67] виявили, що обчислювальна потужність додавалась спочатку поодинокими майнерами, але з ростом мереж збільшувалась по експоненті.
- *Безпека та конфіденційність*. Багато існуючих програм на основі блокчейн використовують смарт-транзакції і контракти, які мають бути не анонімними, що піднімає проблеми конфіденційності та безпеки даних, що зберігаються в спільному журналі. Крім того, сама блокчейн технологія може бути привабливою мішенню для кіберзлочинців, а отже може постраждати від різноманітних атак, таких як розподілені атаки відмови в обслуговуванні.
- *Затримка та складність*. Через розподілений характер, транзакції на основі блокчейну може вимагати до кількох годин на завершення, доки всі сторони не оновлять свої книги записів. Ця затримка створює багато невизначеностей для учасників транзакції та відкриває шляхи для кіберзлочинців.

Nicholas Kolokotronis у своїй статті [68] запропонували розподілену архітектуру довірчого управління для CIDN. В такому випадку, кожна IDS ділиться з іншими вузлами IDS та зовнішніми хостами інформацією. При цьому дані, що стосуються безпеки, якими обмінюються учасники CIDN зберігаються в блокчейні, у вигляді довіреного лананцюга, щоб уникнути втручання з боку шкідливих вузлів. Поєднання PoW і PoS було запропоновано для використання в якості протокола консенсусу, згідно з цим протоколом вузол IDS, що заслуговує на довіру – це вузол з більшою обчислювальною потужністю і більшою ставкою – для нього підвищена ймовірність бути обраним для генерування наступного блоку. Відповідно до

філософії протоколів PoS (і відповідно PoW), вузол з найвищою ставкою має більшу ймовірність створення наступного блоку. Комбінація протоколів PoW і PoS в ланцюжку довіри дає можливість використання гібридного майнінгово-виборного метода досягнення консенсусу, де імовірність обрання лідером, що заслуговує довіри IDS вузла збільшується як з його обчислювальною потужністю, так і зі ставкою. Перевага комбінованого метода в тому, що він запобігає ситуаціям, в яких надійний вузол IDS з великою ставкою має можливість безперервно генерувати всі блоки.

Умотивовані популяризацією блокчейн технологій Wenjuan Li з командою розробили CBSigIDS, загальний фреймворк для колаборативного виявлення вторгнень на основі сигнатур з використанням блокчейн технологій для IoT [69]. Ця розробка пристосовує блокчейн для допомоги в розповсюдженні та побудові спільної довіреної бази сигнатур. Така архітектура показала підвищення надійності та ефективності виявлення вторгнень в IoT мережах в порівнянні з локальними IDS, і окрім того вона підвищує стійкість системи виявлення вторгнень до flood атак. На блокчейн компонент в ній покладено 3 головні ролі:

- *P2P зв'язок.* Цей компонент відповідає за встановлення з'єднання з іншими вузлами IDS в мережі та організацію роботи, управління а також фізичні комунікації.
- *Компонент співпраці*. Цей компонент використовується, щоб дозволяє вузлам IDS спільно збирати необхідну інформацію для оцінки надійності цільових вузлів і надіслати відповідний зворотний зв'язок, запитуваний іншими вузлами.
- *Компонент управління довірою*. Цей компонент відповідає за реалізацію довірених обчислень та репутацію вузлів. Як приклад, механізм довіри на основі викликів досліджує репутацію вузла шляхом порівняння отриманого зворотного зв'язку з очікуваною відповіддю [70] [71].

У цій роботі автори припускають, що зловмисник може контролювати один або кілька вузлів у CIDN, але не може успішно керувати великою кількістю вузлів IDS протягом короткого періоду часу. Крім того, оскільки кожен вузол CIDN має пару приватний і відкритий ключ, їхніми ідентифікаторами неможливо легко маніпулювати та продублювати. Такі обмеження зазвичай припускаються і для інших розподілених застосунків на основі блокчейн технології, оскільки у випадку масового контролю зловмисника над вузлами такої системи, стабільна робота блокчейн неможлива, або в розподіленій системі постійно виникатимуть конфлікти, або ланцюжок блоків розділиться на кілька паралельних, що приведе до фактичного поділу мережі.

1.6 Постановка задачі та логічна структура роботи

Розподілена система виявлення вторгнень в мережі малих та середніх підприємств на основі blockchain є частиною загальної системи управління кібербезпекою в організаціях.

З метою побудови розподіленої системи захисту комп’ютерних мереж на основі blockchain маємо вирішити наступні задачі.

1) Сформувати основні критерії, що впливають на захищеність комп’ютерних мережмалих та середніх підприємств.
2) Розробити модель інформаційних процесів виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств на основі blockchain.
3) Розробити імітаційну модель розподіленої системи захисту комп’ютерних мереж на основі blockchain.
4) Провести випробування роботи системи в штучно створеному наближеному до реального та реальному мережевому середовищі

для оцінки ефективності її системи

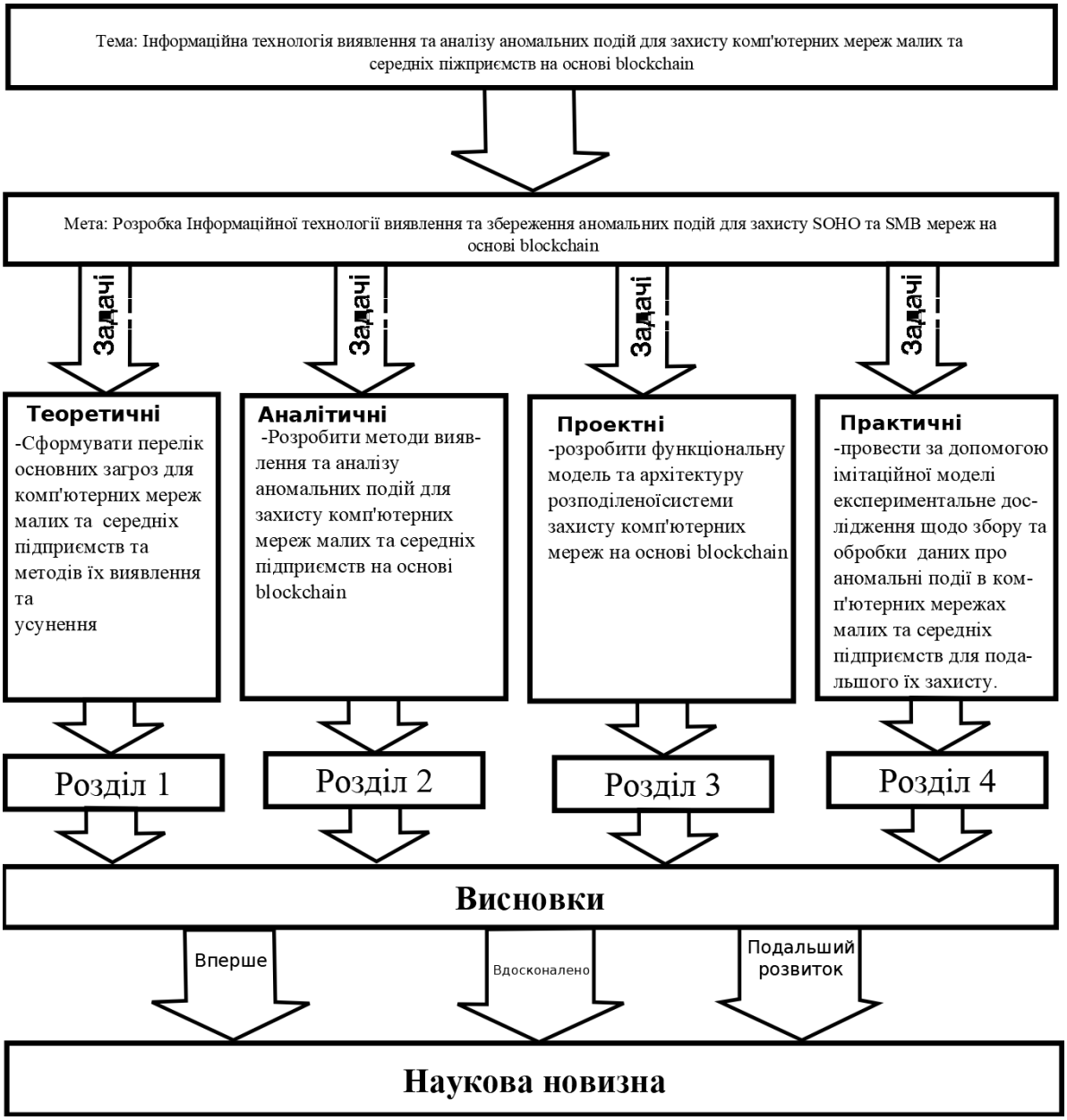


Рисунок 1.2 - Логічно-структурна модель дослідження

Для здійснення дослідження пропонуємо виокремити наступні підзадачі:

Проведення аналізу інформаційної безпеки комп’ютерних мереж малих та середніх підприємств:

– визначення типової структури мереж малих та середніх підприємств;

- визначення основних загроз для мереж даного класу;
- дослідження основних методів та засобів захисту мереж малих та середніх підприємств;

Визначити структуру, формат даних про аномальні події для подальшого зберігання в blockchain підсистемі.

Розробити спосіб розв'язання задачі – методи зберігання та розповсюдження даних про аномальні події за допомогою blockchain підсистеми для подальшого виявлення мережевих загроз і вторгнень.

Висновки до розділу 1

Аналіз основних загроз мережевій безпеці малих та середніх підприємств є підставою для наступних висновків:

1) Результати аналізу звітів за кіберінцидентами показали, що мережі малих та середніх підприємств значно частіше стають цілями для атак аніж великі мережі. В першу чергу це пов'язане із великою кількістю таких мереж та їх слабшою захищеністю. А причина слабшої захищеності мереж полягає в тому, що зазвичай бюджет на їх створення достатньо обмежений і не дає можливості використовувати потужні комерційні засоби для захисту та найняти кваліфікованих спеціалістів з кібербезпеки.
2) Велика кількість атак зазвичай мають масовий характер – тобто не направлені на конкретну мережу, а спрямовані на пошук вразливостей в достатньо широкому діапазоні мереж, що робить важливим завдання оперативного обміну актуальною інформацією між системами захисту мереж.
3) Існуючі засоби захисту мереж є або достатньо дорогими і потужними корпоративними рішеннями, що робить їх використання невигідним

для мереж невеликих підприємств. Або ж існують безкоштовні рішення з відкритим вихідним кодом, які не здатні забезпечити достатній рівень захисту без правильних налаштувань та постійного оперативного коригування баз правил, а тому їх використання вимагає наявності спеціаліста, який зміг би правильно налаштувати такі рішення.

4) Проблема оперативного обміну інформацією між системами захисту, для попередження і швидкої протидії масованим атакам, може бути вирішена шляхом використання розподілених децентралізованих систем, зокрема побудованих на основі технології блокчейн, яка дає можливість підвищити швидкість і надійність децентралізованого обміну даними без участі третьої довіреної сторони.

Результати досліджень приведених в розділі опубліковані в роботах [25, 36]

РОЗДІЛ 2
МОДЕЛІ І МЕТОДИ ВИЯВЛЕННЯ ТА АНАЛІЗУ АНОМАЛЬНИХ ПОДІЙ ДЛЯ ЗАХИСТУ КОМП'ЮТЕРНИХ МЕРЕЖ НА ОСНОВІ BLOCKCHAIN ТЕХНОЛОГІЇ

2.1 Концептуальна модель розподіленої інформаційної системи виявлення та аналізу аномальних подій на основі blockchain технології

Невеликі самостійні системи виявлення вторгнень, які зазвичай використовуються для захисту SOHO та SMB мереж, на жаль, не можуть забезпечити достатній рівень захисту оскільки спираються тільки на локальний набір правил і не враховують оточення. Тоді як в сучасних умовах важливо вчасно реагувати на нові загрози.

Для вирішення цієї проблеми потрібно змінити архітектуру системи виявлення вторгнень на більш глобальну, яка охоплюватиме значно більшу кількість локальних мереж та їх сегментів, і відповідно інформація між модулями системи виявлення вторгнень буде оперативно передаватися, що дасть можливість при перших ознаках атаки на один із сегментів мережі підготувати інші сегменти мережі для зменшення негативних наслідків атаки.

Однією з головних вимог у нашому випадку є вимога консенсусу. Класичний підхід до його реалізації – використання централізованих сторонніх надійних вузлів. Але використання централізованого обміну інформацією для розподіленого IDS не є гарною ідеєю, оскільки збільшення кількості розподілених вузлів збільшить навантаження на центральний надійний вузол, і цей вузол буде SPoF системи. Використання архітектури з наявністю центрального вузла для такої системи недоцільно, оскільки наявність центрального вузла робить систему більш вразливою та менш стабільною. А отже, архітектура повинна бути максимально

децентралізованою. Одним із шляхів реалізувати це безпечно є технологія Blockchain.

У розподіленій системі виявлення вторгнень технологія блокчейну може бути використана для безпечного та надійного обміну даними в одноранговій мережі між вузлами.

Blockchain – це фактично структура, що являє собою ланцюжок блоків, що поєднані між собою за допомогою криптографічних хешів, при цьому ланцюжок може тільки збільшуватися за рахунок додавання нових блоків. Кожен блок містить в собі хеш попереднього блоку, що дозволяє забезпечити цілісність даних та попередити зміну інформації в попередніх блоках. Кожен блок містить в собі структуру даних у вигляді списка, що дозволяє зберігати певний набір транзакцій [58]. Зазвичай blockchain використовується для побудови криптовалютних застосунків, але використання blockchain не обмежується тільки криптовалютами.

У системах виявлення вторгнень blockchain можна використовувати для створення безпечного, децентралізованого, розподіленого сховища для зберігання сигнатурних баз, налаштувань та інформації про виявлені вторгнення та аномалії [69]. Рішення на основі blockchain пропонувались авторами і для IoT середовища, враховуючи його особливості [72] [73], так і для мереж загального призначення [74] [75].

Слід мати на увазі, що Blockchain передбачає створення сховищ лише в форматі журналу, тобто такого сховища, до якого можна тільки додавати нові дані, і не можна змінювати або видаляти старі. Для системи виявлення вторгнень такий варіант сховища цілком придатний, оскільки усі дані з якими працює система в основному тільки додаються (це як сигнатурні бази та налаштування, так і журнали подій).

Отже, до переваг використання Blockchain для розподіленої системи виявлення вторгнень в SOHO та SMB мережах можна віднести :

- створення децентралізованого сховища, яке не залежить як від роботи жодного конкретного вузла, так і від доступності окремої мережі або її сегмента підключеного до розподіленої системи;
- контроль цілісності даних на усіх граничних вузлах SOHO та SMB мереж;
- довірений обмін інформацією між вузлами розподіленої системи виявлення вторгнень без використання центрального довіреного вузла;

До недоліків Blockchain для розподілених систем виявлення вторгнень можна віднести:

- значне зростання об’ємів внутрішнього трафіку при збільшенні кількості мереж приєднаних до розподіленої системи виявлення вторгнень;
- неможливість зміни даних, що були записані раніше, навіть якщо це було хибне спрацювання, що вимагає коректної обробки записаної інформації;
- постійне зростання розміру ланцюжка блоків.

Виходячи з цього можна зробити висновок, що Blockchain доцільно використовувати в розподілених системах виявлення вторгнень з великою кількість клієнтів, коли облік і контроль клієнтів виконати достатньо складно, що цілком відповідає розподіленій системі виявлення вторгнень для SOHO та SMB мереж.

Blockchain технологія в системі виявлення вторгнень фактично виконує функцію об’єднуючого компонета, що забезпечує захищений обмін інформацією та її зберігання. І при цьому збільшення кількості blockchain вузлів підвищує надійність та зламостійкість системи. Концептуальна модель розподіленої системи виявлення вторгнень на основі blockchain наведена на рисунку 2.1 і відображає зв’язки між основними компонентами системи.

Розподілена система виявлення вторгнень має містити кілька основних компонентів, включаючи компоненти моніторингу та захоплення трафіка, компоненти блокування та фільтрації трафіка, компоненти обробки та класифікації а також контролюючий компонент, які поєднані між собою в цілісну систему.

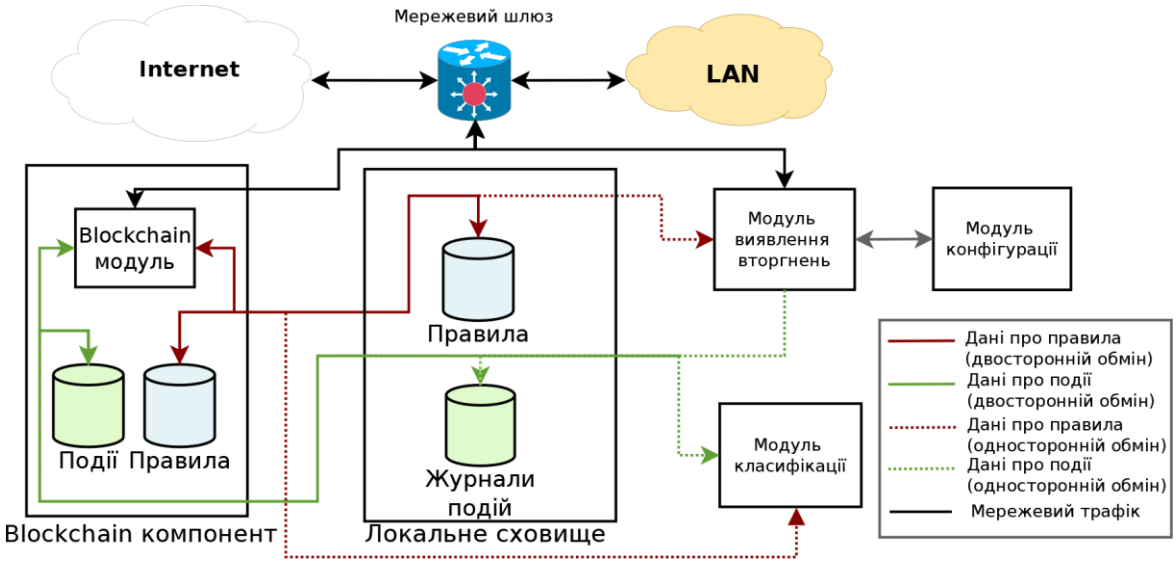


Рисунок 2.1 – Концептуальна модель розподіленої системи виявлення вторгнень на основі blockchain для комп'ютерних мереж малих та середніх підприємств

Окрім цього для системи виявлення вторгнень на основі Blockchain необхідні компоненти для роботи з Blockchain – повні blockchain вузли та тонкі blockchain клієнти. Повний вузол зберігає у себе повну копію ланцюжка блоків, тоді як тонкий клієнт звертається до повних вузлів для отримання ланцюжка блоків [76].

При цьому блоки будуть зберігати як інформацію про сигнатури атак так і поточну інформацію про стан усіх вузлів та підозрілу активність виявлену будь-яким із них, можлива також передача службової інформації, яка має бути записана в журнал. При цьому вузли системи виявлення вторгнень фактично утворюватимуть рівноправну peer-to-peer мережу, де

кожен вузол може працювати самостійно, але ділиться і отримує інформацію від інших вузлів про стан мережі, що знаходиться під захистом кожного вузла.

База даних правил із сигнатурами вторгнень є однією з найважливіших частин систем виявлення вторгнень, оскільки виявлення на основі сигнатур є найшвидшим методом виявлення вторгнень і не вимагає великої кількості обчислювальних ресурсів. Це означає, що виявлення на основі сигнатур можна використовувати навіть на апаратних засобах низького класу, але при цьому слід зазначити, що низька ресурсоємність такого методу виявлення вторгнень відносна, і зовсім слабке обладнання не зможе обробляти трафік навіть сигнатурними методами, або вимагатиме зменшення кількості правил. Методи на основі сигнатур також мають низький коефіцієнт хибно-позитивних помилок порівняно з виявленням на основі аномалій та статистичних методів. Але правила не можуть працювати самостійно, оскільки мережевий трафік потребує попередньої обробки, перш ніж його можна порівняти з сигнатурами. Різні протоколи мають свої специфічні алгоритми попередньої обробки. Наприклад, Snort IDS має для цієї мети модулі динамічної обробки, і дії, що виконуються цими модулями набагато складніші, ніж ті, що можна описати в простому наборі текстових правил.

Але для того, щоб система виявлення на основі сигнатур була ефективною, вона повинна бути належним чином налаштована. В даному випадку оптимальним буде використовувати спрощений метод конфігурації, щоб будь-який досвідчений користувач міг легко налаштувати систему самостійно. Перш за все слід згрупувати базу правил за протоколами, щоб користувач міг вибирати лише протоколи, які використовує його система. Це може допомогти зменшити розмір бази даних використовуваних правил. Це також допоможе відключити не використовувані динамічні процесори трафіку, а також зменшить навантаження на процесор [77].

Правила також можна розділити на правила високої чутливості для користувачів, яким потрібен максимальний рівень безпеки, але інша сторона цих правил полягає в тому, що такі правила викликають більш високий коефіцієнт хибно-позитивних помилок, позначаючи певну нормальну діяльність як підозрілу. Правила з меншим показником чутливості краще підійдуть для більшості користувачів, але такі правила можуть пропустити певну підозрілу активність.

Якщо просто вимкнути правила для невикористаних служб, система матиме вразливі місця. Отже, щоб зробити систему безпечною, потрібно закрити всі порти, які можуть бути пов’язані з відключеними службами. Це зменшить ймовірність того, що хтось буде використовувати деякі вразливості служб, які не були вимкнені користувачем. Ще одним кроком для підвищення безпеки системи є використання набору правил, які визначатимуть запити до відключених служб, навіть якщо їх порти закриті. Так, наприклад, якщо хтось намагається отримати доступ до служби HTTP на захищеному сервері, але протокол HTTP був вимкнений у конфігурації IDS, то порти будуть заблоковані, а IDS повідомить про підозрілу подію та про ймовірну атаку. Така поведінка IDS може допомогти виявити тих зловмисників, які шукають специфічні сервіси та їх уразливості. Наприклад, якщо розподілені IDS виявляють, що одна IP-адреса намагається отримати доступ до портів та протоколів однієї служби на багатьох захищених серверах, вона може принаймні занести таку адресу в чорний список і навіть повідомити відповідного провайдера про шкідливу активність.

Отже, виникає необхідність використанні підсистеми керування правилами. По-перше, це пов'язано з евристично створеними правилами на основі списку підозрілих подій. Звідки випливає, що список підозрілих подій, який поширюється за допомогою блокчейна, може бути оброблений усіма вузлами для загальнодоступних подій (або лише невеликим списком надійних вузлів для приватних подій), і вузол, який збирає достатньо

інформації для автоматичного створення правила виявлення, створює його та робить транзакцію в блокчейн з новим правилом. Такі автоматично створені правила позначаються як правила високої чутливості до перевірки вручну спеціалістами з кібербезпеки, щоб запобігти збільшенню високого коефіцієнта хибно-позитивних помилок.

2.2 Методи класифікації аномальних подій в розподіленій системі виявлення вторгнень

Для виявлення мережевих вторгнень та аномалій не записаних в сигнатурних базах, або таких, потрапляння яких в категорію шкідливої активності за сигнатурними правилами неоднозначне, використовується модуль класифікації подій (рисунок 2.1). Основна задача такого модуля класифікація подій на основі даних, отриманих різними вузлами розподіленої IDS. Для класифікації подій в цілому можна використовувати різні методи, але далеко не усі методи будуть ефективними в контексті виявлення вторгнень. Частково це пов’язане зі складністю процесу навчання для деяких класифікаторів. Основні типи класифікаторів, які можна виділити для розподіленої IDS це регресійний класифікатор, класифікатор на основі дерев рішень та класифікатор на основі штучної імунної системи.

Оскільки окремі класифікатори зазвичай дають недостатньо точний результат, для остаточного прийняття рішення щодо приналежності трафіка до категорії шкідливого використовується агрегація результатів роботи усіх наявних класифікаторів. Більше того, деякі з вказаних класифікаторів, наприклад класифікатор на основі штучної імунної системи, не підходить для самостійного використання оскільки може давати хибно позитивні результати при відхиленні від стандартного режиму роботи, а тому вимагає додаткового підтвердження від класифікаторів інших типів [78]. Остаточним результатом роботи модуля класифікації є значення , де 0 вказує на відсутність аномалій або шкідливого трафіку, а 1 на наявність аномального

або шкідливого трафіку. Для класифікаторів, які видають результат у діапазоні відмінному від *[0,1]*, в процесі нормалізації відбувається приведення значення до вказаного діапазону. Основні вхідні дані, які модуль класифікації отримуватиме від модуля виявлення включає такі поля: Часова мітка в форматі UNIX timestamp, тип події, IP адреси та порти відправника та приймача, протокол, та похідні від нього поля протокол рівня застосунків, тип ICMP протокола, у випадку наявності таких. Окрім того присутні поля, що відображають кількість та об'єми пакетів відправлених до сервера та до клієнта, полі що відповідають наявним TCP прапорцям у випадку якщо використовується TCP протокол. На основі полів кількість пакетів до сервера та кількість пакетів до клієнта створюємо також поле співвідношення, яке показує наявність перекосів по кількості пакетів в ту чи іншу сторону, що в деяких випадках може вказувати на наявність DDOS атак, але при цьому слід мати на увазі, що несиметричний трафік далеко не завжди свідчить про наявність аномалій, тому даний параметр можна використовувати для оцінки лише в комплексі з іншими параметрами трафіку.

2.2.1 Регресійні класифікатори

Особливість таких класифікаторів полягає в тому, що вони гарно підходять для ситуацій, де треба отримати обчислюване значення, оскільки результат отримується шляхом обчислень над числовими значеннями вхідних даних. На етапі навчання регресійної моделі встановлюється залежність результату від значень вхідних величин шляхом підбору параметрів регресії, після цього шляхом підстановки вхідних даних у отриману в результаті етапу навчання формулу обчислюється результат для того чи іншого набору вхідних даних. Кількість параметрів та складність процесу навчання залежить як від розмірності вектора вхідних даних, так і від типу регресії. В нашому випадку на вхід регресійного класифікатора

направляються усі доступні дані про трафік, які були перераховані раніше. Для випадку класифікації подій отриманих IDS найкращим випадком буде логістична регресія, яка в результаті дасть нам ймовірність того, що подія є шкідливою. В інших же випадках може знадобитися регресія на основі інших функцій, але точніше вид регресії можна підібрати лише після більш детального аналізу навчальної вибірки, а також шляхом тестування вже навчених класифікаторів на тестовій вибірці. Перевага логістичної регресії в даному випадку полягає в тому, що логістична регресія, на відміну від лінійної використовує для оцінки вірогідності настання події логістичну функцію наступного виду (1). Така функція здатна забезпечити чітку бінарну класифікацію на ширшому діапазоні вхідних даних.

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

В нашому випадку на вхід класифікатора подається тип виявленої події, протокол, протокол прикладного рівня, типи ICMP протокола та відповіді, якщо базовий протокол ICMP, а також ключові для події TCP прапорці, кількість пакетів направлених до сервера та до клієнта, кількість подій від одного адресата, частота виявлення підозрілих подій даного типу від одного адресата. Також можливе розширення вхідного набору даних для класифікатора іншими параметрами, які можна отримати від модуля виявлення вторгнень. На виході ж класифікатор видає логічне значення, яке вказує чи є подія шкідливою виходячи з відомих на даний момент даних. Тоді рівняння логістичної регресії можна відобразити наступним виразом:

$$y = \frac{e^{b_0+\sum_{i=1}^{n}(b_i*x_i)}}{1+e^{b_0+\sum_{i=1}^{n}(b_i*x_i)}}, \tag{2}$$

де x_i – описані вище вхідні параметри для класифікатора, b_i – відповідні їм коефіцієнти, що підбираються в процесі навчання класифікатора. Навчання класифікатора відбувається з використанням методу максимальної правдоподібності, який передбачає підбір таких параметрів b, які можуть

максимізувати значення функції правдоподібності на вибірці тренувального набору вхідних значень. А для максимізації такої функції в свою чергу найбільш популярним методом є метод градієнтного спуску. Оскільки в даному випадку задачею є пошук локального максимуму, то використовується алгоритм градієнтного підйому, основна відмінність якого полягає у напрямку кроків. Даний алгоритм є ітераційним алгоритмом оптимізації, який передбачає зміщення у напрямку градієнту в поточній точці. І хоча швидкість і результат роботи даного методу сильно залежать від налаштувань, даний метод дозволяє швидко визначити параметри для максимального значення функції.

2.2.2 Класифікатор на основі методу опорних векторів

Даний метод належить до сімейства лінійних класифікаторів, і основна його ідея полягає в тому, щоб перевести оригінальний простір векторів в більш багатомірний і знайти розділяючу гіперплощину з найбільшим зазором в цьому просторі. Даний метод передбачає наявність як мінімум двох класів, а отже також підходить для класифікації подій IDS. Метод опорних векторів в цілому може використовуватися як для класифікації так і для регресії, але оскільки в нашому випадку основна задача полягає в класифікації, розглянемо саме такий варіант.

В нашому випадку передбачено такий набір міток класів $Y = \{-1,+1\}$, де -1 це відсутність аномального трафіку, а +1 наявність аномалій або спроби вторгнення. На вхід класифікатора подається вектор, що містить N ознак $x = (x_1, x_2, ..., x_n)$ в просторі R^n. В результаті навчання класифікатора на основі навчальної вибірки формується функція $F(x) = y$, що приймає аргумент x з простору R^n і видає мітку класу y.

В процесі навчання основна задача підібрати таке рівняння гіперплощини $w_1x_1 + w_2x_2 + ... + w_nx_n + w_0 = 0$ в просторі R^n, яка б оптимальним чином розділяла два класи деяким оптимальним чином.

$$F(x) = sign(w^T x - b) \quad (3)$$

Загальний варіант перетворення вхідних даних у мітку класу можна відобразити наступним чином (11). Гіперплощина підбирається таким чином, щоб максимізувати відстань від неї до об’єктів кожного класу, що забезпечує більш високу точність класифікації. Набір параметрів, що використовується для проведення класифікації аналогічний набору параметрів, що використовується для регресії.

2.2.3 Класифікатор на основі дерев рішень

Дерева рішень зазвичай являють собою класифікатори, що базуються на ланцюжку умовних перевірок, які приводять до того чи іншого результату. Особливістю цього класифікатора є те, що в багатьох випадках, коли кількість параметрів невелика, дерево можна побудувати вручну, спираючись на експертні знання. У випадку ж використання методів машинного навчання, одиноке дерево може не надавати достатньої точності, тоді у нагоді стають ансамблеві методи, які поєднують кілька схожих моделей для підвищення точності остаточного результату. Тобто замість одного дерева створюється набір дерев, які відрізняються за параметрами? які використовувались в процесі навчання, а при розрахунках остаточного результату відбувається агрегація результатів роботи кожного з дерев. Такі ансамблеві методи дають можливість зменшити вплив перенавчання та зробити модель більш точною на широких наборах значень вхідних даних.

На відміну від регресійного класифікатора та класифікатора на основі опорних векторів, класифікатор на основі дерев показує більш високу точність в деяких випадках, оскільки на відміну від класифікаторів, що будують функцію багатьох змінних, алгоритми дерев рішень є більш гнучкими і дають можливість описувати складніші залежності.

Методи побудови дерева часто базуються на розрахунку ентропії для навчального набору даних. Загальна формула ентропії виглядає наступним чином:

$$H = -\sum_{i=1}^{c} p_i \log_2 p_i, \quad (4)$$

де p_i – частотна вірогідність елемента і-го класу. Значення ентропії вказує, наскільки невпорядкованим є набір даних. При чому будь-який поділ набору даних на дві частини за певним параметром знижує ентропію кожної з частин. Таким чином, набір даних рекурсивно ділиться на частини за можливими критеріями, поки кожна з частин набору не матиме ентропію рівну 0 – що означає, що всі елементи такого набору належать до одного класу. Для вимірювання зміни ентропії використовується величина інформаційного виграшу, яку можна описати наступною формулою

$$IG = H_0 - \sum_{i=1}^{q} \frac{N_i}{N} H_i, \quad (5)$$

де q – кількість груп, на які відбувається поділ, N_i – кількість елементів в групі, N – загальна кількість елементів, H_i – ентропія для групи. Отже, даний параметр вказує на скільки знижується ентропія при обраному варіанті поділу, а значить і вказує, за яким параметром вигідніше поділяти набір на групи.

Також, на основі побудованого шляхом машинного навчання дерева, можна побудувати алгоритм, що відтворювався б вручну, що часто неможливе для класифікаторів, які вимагають складних математичних розрахунків.

Але однією з основних проблем даного класифікатора є схильність до перенавчання, оскільки алгоритми які визначають параметри при побудові дерева в процесі навчання будують занадто деталізоване дерево, а це окрім зниження точності роботи класифікатора знижує його швидкодію. Як одне з рішень цієї проблеми є зазвичай використовують відсікання гілок.

Також для зменшення розмірів дерева використовується метод зменшення помилки обмеження дерева. При використанні цього методу, починаючи з листя кожен вузол заміняється на найпопулярніший клас і виконується перевірка точності роботи класифікатора, якщо точність не змінюється, така заміна зберігається. Проте такі методи не гарантують побудови оптимального дерева, а тут слід мати на увазі, що збільшення розмірів дерева знижує швидкодію класифікатора, на відміну від регресійних класифікаторів, де швидкодія класифікатора не залежить від результатів процесу навчання.

2.2.4 Класифікатор на основі штучної імунної системи

Більшість типів DDoS атак спричиняють збільшення навантаження на процесор і пам’ять, та загальне використання мережевого трафіку оскільки сервер намагається обробити велику кількість пакетів, тому це може бути непрямою ознакою DDoS, на яку потрібно звернути увагу. В такому випадку може допомогти метод непрямого виявлення атак та зловживання системними ресурсами шляхом моніторингу навантаження процесора пам'яті, а також параметри трафіку, отриманого мережевими інтерфейсами вузла, аналогічні параметрам, що використовувалися іншими класифікаторами. Також даний класифікатор дозволяє використовувати в якості параметру загальну статистику по прийнятому та відправленому вузлом трафіку. Отриманий таким чином набір даних обробляється штучною імунною системою. Іншим варіантом використання даного класифікатора на основі штучної імунної системи може бути його використання для виявлення перевищення фонового рівня потенційно-шкідливих подій. Тоді в якості основних даних на класифікатор будуть направлятися події зафіксовані в процесі попередньої фільтрації та сигнатурного виявлення.

Цей підхід має кілька переваг. Перш за все, це низьке використання системних ресурсів, оскільки ми не перевіряємо велику кількість даних, і цей підхід забезпечує стабільне використання системних ресурсів і не залежить від завантаженості контрольованої системи. Друга перевага – портативність – такий підхід можна використовувати практично на будь-яких пристроях – серверах, маршрутизаторах, робочих станціях та вбудованих пристроях. Також цей підхід може допомогти виявити внутрішні загрози або системні проблеми, що робить його корисним для виявлення скриптів майнінгу, програм-вимагачів, шкідливого програмного забезпечення, що використовується для ботнетів, або просто проблеми з налаштуванням програмної або апаратної складової, що може спричиняти значне використання системних ресурсів [78].

Основним недоліком цього підходу є те, що він може виявляти лише проблеми, які впливають на використання процесора пам’яті та об’єми мережевого трафіка. Отже, це не заміна класичних IDS, але це може бути гарним доповненням до сигнатурних IDS. Але при цьому, для використання в розподіленій системі виявлення вторгнень такий модуль виявлення та класифікації повинен бути встановлений на найбільш вразливих до атак вузлах мережі, наприклад серверах підключених до глобальної мережі.

Використання інформації зібраної виключно в режимі реального часу явно недостатньо для повної оцінки стану системи та мережі даним типом класифікатора, тому в даному випадку важливо оцінити динаміку потрібних параметрів протягом певного періоду часу, наприклад, п’яти, п’ятнадцяти, тридцяти хвилин та однієї години. Тому важливим є створення наборів даних, які зберігають дані за обрані періоди часу. Наступним кроком є обчислення статистичних параметрів для цих наборів даних – середнє значення, дисперсія, максимальні та мінімальні значення.

Після цього можна сформувати вектор (6) з статистичних параметрів, і цей вектор буде використовуватися для оцінки поточного стану системи.

$$\vec{S}(M_1, V_1, L_1, H_1, M_5, V_5, \dots), \quad (6)$$

де M_n – середнє значення за n хвилин, V_n – значення дисперсії за n хвилин, L_n – найменше значення для n хвилин, H_n– найвище значення за n хвилин. Потрібно використовувати нормовані значення (7) для обчислення статистичних параметрів – оскільки параметри мають різні діапазони значень, і це спричинить проблеми з обчисленням дистанцій (більші значення матимуть більший вплив на відстань).

$$x' = (x - MIN[X]) / (MAX[X] - MIN[X]), \quad (7)$$

де x' – нормалізоване значення параметра, x – необроблене значення параметра, а X – набір вихідних значень. Також нормовані значення дозволять мати сигнатури станів для деяких відомих проблем і зроблять виявлення більш точним.

В якості алгоритма класифікації в даному випадку використовується алгоритм негативного відбору – один із найбільш популярних алгоритмів для реалізації штучних імунних систем. Алгоритм "Негативного відбору" наслідує процеси, що відбуваються в імунній системі ссавців [79], і в основному використовується для виявлення аномального стану або нестандартної поведінки.

Цей алгоритм включає два етапи – створення набору детекторів та виявлення нових екземплярів. На першому кроці ми створюємо багато різних «реагуючих клітин» – детекторів, які охоплюють поле можливих значень наших векторів стану. Потім кожен детектор проходить тест на відповідність векторам нормального стану. Якщо детектор збігається з будь-яким із нормальних векторів стану – він позначається як «нормальний» детектор, і не використовується для виявлення аномалій (У класичному «Негативному відборі» нам потрібно відмовитися від детекторів, що відповідають нормальному стану, але в нашому випадку вони будуть корисними для визначення нормального стану системи). Усі інші детектори позначені як "аномальні", і відповідність будь-якому з цих детекторів викликає сигнал

тривоги про аномальний стан. В результаті у нас є набір детекторів. Деякі з них представляють нормальний стан системи, а деякі – аномальний стан. Після цього система виявлення готова до роботи.

Аномальні детектори можна позначати мітками, які представляють найбільш популярні та помітні проблеми з використанням ресурсів системи, таких як flood атаки, або майнінг скрипти. Це дозволяє системі повідомляти не тільки про наявність аномального стану, але й називати можливий тип аномалії, якщо для цього достатньо даних.

В нашому випадку набір даних представлений векторами чисел, тому наш набір детекторів буде містити випадково сформовані та рівномірно розподілені вектори, у достатній кількості, щоб охопити більшу частину можливих значень векторів реального стану.

Замість того, щоб перевіряти зразки на пряме співпадіння з детекторами, ми вирішили шукати найближчі детектори з алгоритмом K-найближчих сусідів, який є одним з найпростіших алгоритмів машинного навчання для завдання класифікації. Цей алгоритм є хорошим вибором у нашому випадку, оскільки він добре працює з багатовимірними векторами і дозволяє нам використовувати менший набір детекторів, ніж нам потрібно, у порівнянні з методом прямого зіставлення. У нашому випадку буде краще використовувати його з K = 1, оскільки ми хочемо перевірити відповідність лише з одним детектором.

Для вимірювання відстані між векторами ми використовуємо просту Евклідову відстань (8), яка добре працює з багатовимірними векторами.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (8)$$

Де n - число розмірів вектора, p і q – вектори. Потім обираємо детектор з мінімальною відстанню до вектора як відповідний. Якщо це «аномальний» детектор – у нас проблема (якщо детектор також має заздалегідь визначену

мітку проблеми, ми навіть можемо здогадуватися, яка у нас проблема). Якщо детектор "нормальний" – система працює нормально.

Процес виявлення також простий. Кожен зразок перевіряється на відповідність усім детекторам, і якщо ми виявили, що вибірка відповідає “аномальному” детектору – у нас є аномалія.

Звичайний стан системи, як правило, має низьке завантаження процесора та використання пам'яті з невеликими піками – коли клієнти звертаються до сервера. Середні лінії майже прямі, як на рисунку 2.2 для завантаження процесора.

Іноді, коли багато клієнтів звертаються до сервера, навантаження на процесор може зростати, в більшості випадків використання пам’яті також зростає при збільшенні кількості клієнтів.

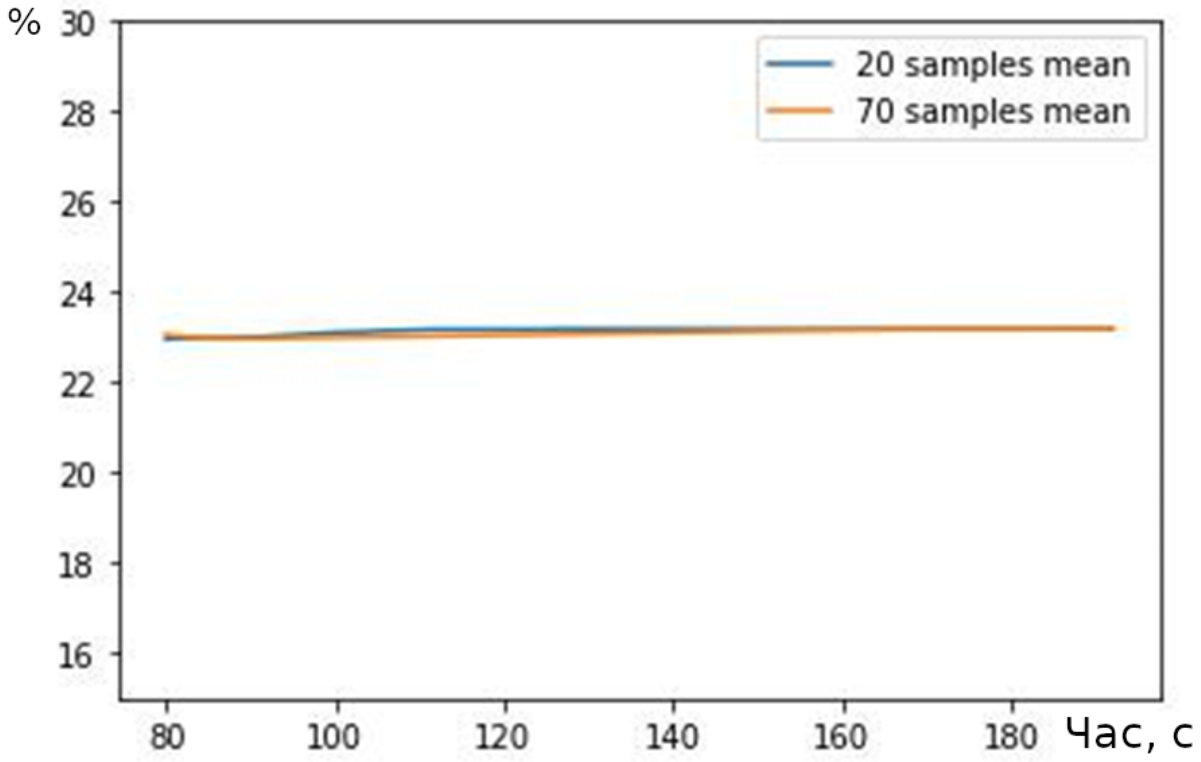


Рисунок 2.2 – Використання пам'яті на сервері з низькими запитами

Програмне забезпечення для майнінгу спричинить близьке до 100% завантаження процесора з невеликими падіннями під час перезапуску

сценарію або запиту на нову роботу (як на малюнку 2.3), коли використання пам'яті не значно збільшується (більшість сценаріїв майнінгу процесора не займає багато пам'яті), тому легко виявити такі типи зловживання ресурсами сервера. Деякі сценарії brutforce можуть виглядати однаково (тому ми також можемо це виявити) [78].

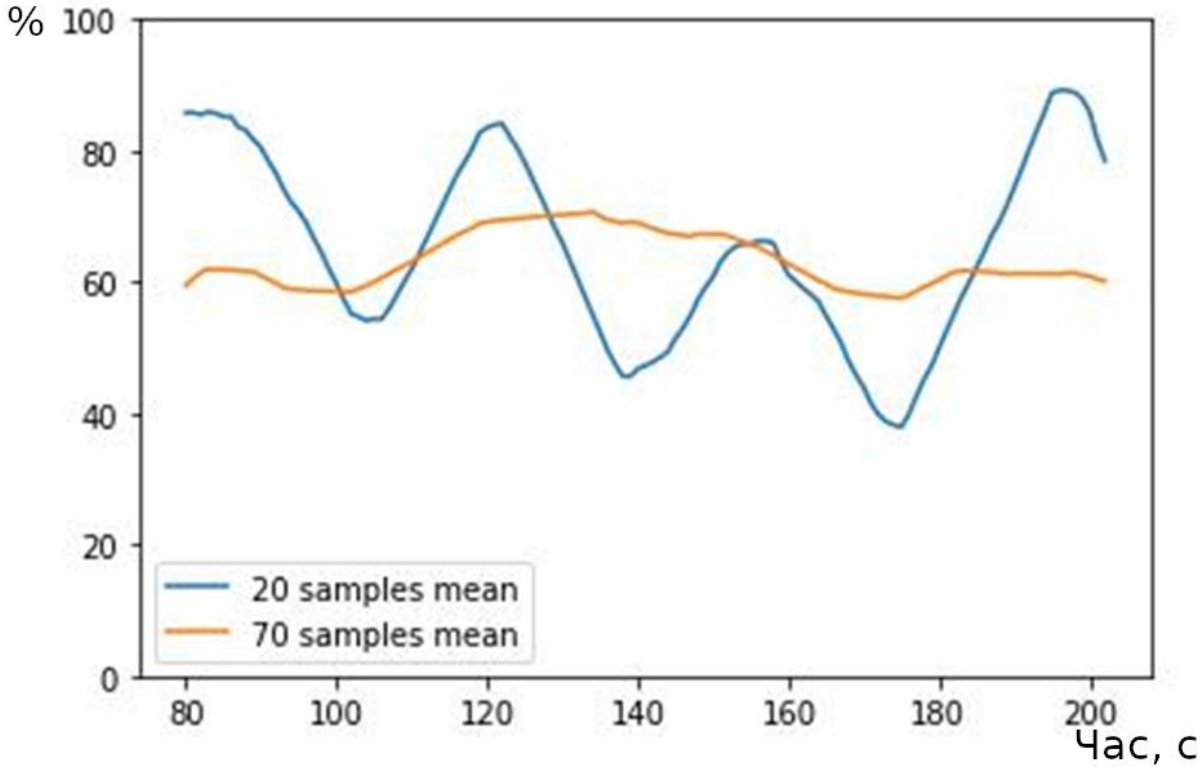


Рисунок 2.3 – Завантаження процесора на сервер із запущеним сценарієм майнінгу

Для детекторів можуть бути використані і параметри, що стосуються використання мережі та типів отриманого трафіку, тоді класифікатор зможе визначити аномальне використання мережі навіть при відсутності підозрілих пакетів, але слід зазначити, що даний тип класифікаторів скоріше виконує додаткову роль, оскільки в комбінації з іншими класифікаторам, даний класифікатор дозволяє підвищити точність визначення кіберзагроз та аномального трафіку, але самостійне використання класифікатора на основі

штучної імунної системи дає лише приблизний результат без детальної інформації про природу аномалії.

2.3 Модель блокчейн компонента розподіленої системи

Існує два типи блокчейну: блокчейн без контролю доступу (permissionless) та блокчейн з контролем доступу (permissioned). Блокчейн з контролем доступу також може бути поділений на загальнодоступний та приватний (блокчейн без контролю доступу буде загальнодоступний за замовчуванням). Також з точки зору контролю прав доступу, сучасні реалізації блокчейну поділяються на три категорії: загальнодоступні, консорціумні та приватні [80].

Загальнодоступний блокчейн використовується в більшості криптовалют, і головна особливість такого блокчейну полягає в тому, що будь-який пристрій, що підтримується може приєднатися до мережі або покинути її в будь-який час, і немає центрального органу, який може керувати участю вузлів. Також при використанні загальнодоступного блокчейна зазвичай немає можливості заблокувати певним вузлам можливість запису в блокчейн [81].

В приватному блокчейні набір читачів і авторів обмежується, для чого виникає необхідність створення центрального органа. Тут центральний орган вирішує та надає право окремим вузлам брати участь у операціях запису чи читання блокчейну. Щоб забезпечити інкапсуляцію та конфіденційність, читач і автор блоків також можуть працювати в окремих паралельних блокчейнах, які взаємопов’язані. У приватному блокчейні інформацію можуть читати лише уповноважені вузли, публічний блокчейн може читати та перевіряти будь-хто [81] [82].

Усі операції в блокчейн, включаючи створення та перевірку блоків, виконуються спираючись на консенсусний протокол, який гарантує, що всі

суб’єкти, що беруть участь в роботі блокчейн, узгоджують єдиний вигляд журналу записів. Протокол консенсусу може залежати від конкретної реалізації блокчейну та моделі загроз [83].

Однією із задач, що покладена на протокол консенсусу є запобігання підробки блоків, для цього блокчейн системи зазвичай використовуються алгоритми proof-of-work, proof-of-stake, proof-of-elapsed-time та інші. Такі алгоритми дозволяють обмежити неконтрольований випуск та підробку блоків (за умови, що більш ніж половина мережі утворена доброчесними вузлами), що значно ускладнює проведення атак на сховище такого типу [84].

У випадку приватного blockchain набір протоколів консенсусу відрізняється, в таких випадках можна застосовувати алгоритм practical byzantine fault tolerance – до переваг даного алгоритму можна віднести більш високу енергоефективність та нижче використання ресурсів, але цей алгоритм ускладнює масштабування системи [85].

Створення нових блоків у блокчейні регулюється консенсусним протоколом, який вказує, як вузли співпрацюють при створенні блоків. Цей протокол може сильно відрізнятися і залежить як від типу реалізації блокчейна, так і від моделі загрози. Щоб забезпечити гарантовані властивості безпеки, публічні блокчейни проектують консенсусну частину або як обчислювальну (Proof-of-Work), або на основі володіння дефіцитним ресурсом у системі (Proof-Of-Stake). З іншого боку, консорціум та приватні блокчейни застосовують толерантні до помилок алгоритми на основі візантійської угоди, такі як PBFT або SIEVE [86] для боротьби зі шкідливими вузлами.

Але використання блокчейну – не є універсальним рішенням, яке може вирішити усі проблеми побудови захищених застосунків, і в деяких випадках блокчейн може бути навіть проблемою.

У роботі [81] Вуст та Герве викладають дуже загальний процес прийняття рішень, який дозволяє визначити, чи є сенс використовувати

блокчейн технологію, і якщо є, то який тип блокчейну найбільше підходить для конкретного випадку.

Основними критеріями вибору типу блокчейну автори вважають, необхідність збереження стану додатку, кількість користувачів або вузлів, що будуть записувати дані та рівень довіри до них. Як правило, якщо нам не потрібно зберігати стан або у нас є не більше одного користувача, що виконує запис, нам не потрібен блокчейн, оскільки його використання призведе до значного сповільнення програми, у порівнянні з використанням простої бази даних, при цьому блокчейн принесе незручності пов'язані з неможливістю зміни вже записаних даних. З іншого боку, якщо у нас багато користувачів, які створюють записи, блокчейн може допомогти зберегти цілісність записів, замінивши довірену третю сторону, яка повинна бути завжди онлайн і контролювати рівень довіри та цілісність даних, в такому випадку блокчейн має перевагу у вигляді децентралізованого рішення, яке матиме значно вищу відмовостійкість. Але блокчейн все одно не потрібен, якщо всіх вузли або користувачі, що мають право на запис знають один одного і між ними встановлена довіра. Транспонувавши такий алгоритм вибору типу блокчейну на системи виявлення вторгнень отримаємо алгоритм зображений на рисунку 2.4.

Якщо всі вузли відомі, але достатній рівень довіри між вузлами не встановлений, ми можемо використовувати блокчейн з контролем доступу. Який вид блокчейну з контролем доступу слід використовувати (публічний чи приватний) буде залежати від того, чи потрібна нам публічна перевірка записів. Якщо так – тоді кожен може приєднатися до мережі як читач, що реалізується у публічному блокчейні. В іншому випадку нам потрібно використовувати приватний блокчейн з контролем доступу. Але ми повинні пам'ятати, що навіть загальнодоступний або блокчейн без контролю доступу потребує шифрування або хешування для захисту записів.

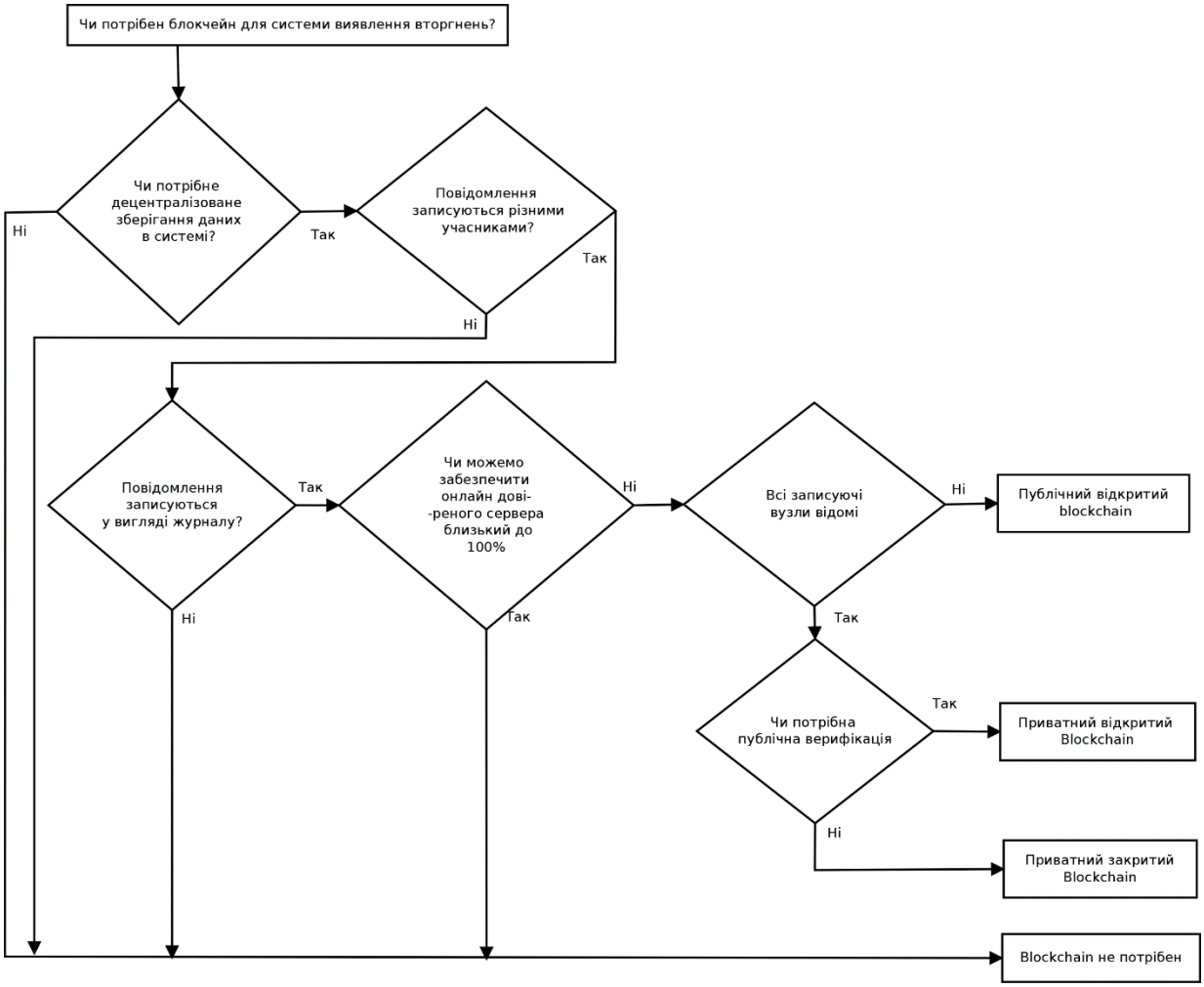


Рисунок 2.4 – Алгоритм вибору типу блокчейна для системи виявлення вторгнень в залежності від вимог до системи

Якщо набір авторів не відомий або може сильно коливатися – блокчейн без контролю доступу може бути хорошим рішенням.

Якщо ми відобразимо ці випадки на контекст виявлення вторгнень в мережі розподіленими засобами, то ми матимемо два випадки, коли ми можемо використовувати блокчейн. Перш за все, це система виявлення вторгнень в приватну мережу всередині якоїсь компанії або між кількома компаніями, які довіряють одна одній і можуть мати певну приватну інформацію. У цьому випадку ми будемо використовувати блокчейн з контролем доступу. Інший випадок – відкрита розподілена мережева система

виявлення вторгнень, до якої може приєднатися кожен. У цьому випадку найкращим рішенням буде блокчейн без контролю доступу.

2.3.1 Структура блокчейна

Як бачимо, IDS потребує багато інформації для роботи. Перш за все, це набір правил, необхідних модулю виявлення вторгнень для швидкої обробки трафіку. Цей набір даних зазвичай лише збільшується, тому використання блокчейна для збереження та розповсюдження набору правил цілком логічний варіант. У випадку ж якщо правило потрібно буде видалити, то його можна відмітити як застаріле.

Іншим видом інформації для спільних розподілених IDS є журнал подій. Журнал подій також постійно збільшується, тому його також можна поширювати через блокчейн. Але якщо правила IDS мають надзвичайно малий розмір (наприклад, набір правил спільноти SNORT займає менше 2 мегабайт), і вони можуть бути легко розміщеними в якості даних блоку в блокчейні, коли ми намагаємося зберегти журнал подій, ми стикаємося з проблемою, що розмір журналів подій дуже швидко збільшується, що не буде добре для сховища блокчейну і в більшості випадків інформація в журналі подій є тимчасовою, і для деяких вузлів постійне зберігання журналу буде неприйнятним. Наприклад, одноплатні (Singleboard) вузли, можливо, не матимуть такого об’єму сховища для зберігання. Це означає, що нам потрібно якимось чином зберігати тимчасову інформацію, щоб вузол міг вибрати, чи треба зберігати таку інформацію протягом тривалого часу. Найпростіший спосіб зробити – це перемістити інформацію за межі структури блокчейну та використовувати перевірене володіння даними (PDP) для перевірки цілісності даних [87]. Принцип роботи PDP наступний: спочатку дані M кодуються в M' так, що кожні дані з блоку m_i з M' містить

s сегментів даних, тобто $m_i = (m_{i,1}, m_{i,2}, \ldots, m_{i,s})$. Метадані σ_i розраховані для кожного блоку m_i мають наступний вигляд:

$$\sigma i = \left(H(name||i) \times \prod_{j=1}^{s} u_j^{m_{i,j}} \right)^{\alpha} \quad (9)$$

де α приватний ключ користувача і $u_j (1 \le j \le s)$ випадково обрані з білінійної групи *G*. Аналогічно до [88] фактор $\left(\prod_{j=1}^{s} u_j^{m_{i,j}}\right)^{\propto}$, що міститься в метаданих також підтримує операцію агрегації, а отже на етапі верифікації можна згенерувати відповідну часткову агрегацію.

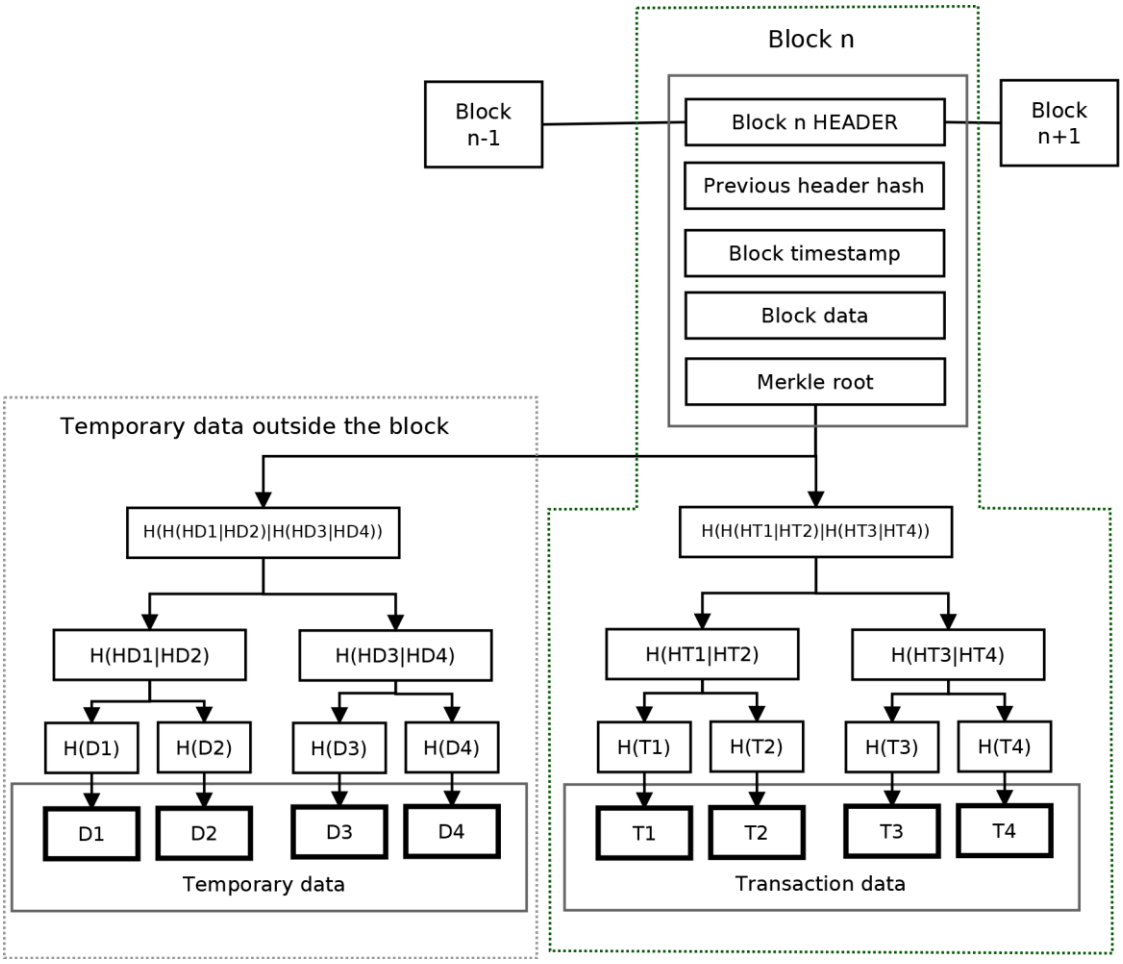


Рисунок 2.5 – Структура блока в блокчейн з пов'язаним деревом Меркла

Механізм PDP в основному використовується в мережах P2P для перевірки цілісності даних. Перший PDP ґрунтувався на ймовірнісній стратегії завершення перевірки цілісності, використовуючи гомоморфні

властивості механізму підпису RSA для агрегації доказів у невелику величину, що значно зменшує витрати на протокол. Але зараз найпоширеніші PDP використовують хеш-дерево [87].

Оскільки при використанні блокчейн технології в розподілений системі виявлення вторгнень доводиться мати справу з великими об'ємами тимчасових даних, які актуальні протягом певного обмеженого проміжку часу, то найкращим варіантом в даному випадку буде розділити дані на постійні, які зберігаються в транзакціях блоків та тимчасові зовнішні – які зберігаються окремо від блоків але криптографічно зв'язані з блоком. Для реалізації такого варіанту зберігання даних Хеш-дерево в нашому варіанті блокчейн розділене на два. Перше дерево відповідає за контроль цілісності зовнішніх даних, що зберігаються поза блоком, друге ж зберігає хеші транзакцій, що включені в блок. Загальний хеш, що включений до заголовку блока складається із кореневиxx хешів обох дерев. Рисунок 2.5 ілюструє структуру блоків в блокчейн із модифікованим деревом Меркла для зберігання тимчасових даних.

Хеш-дерево Merkle – це дерево, в якому кожен нелистковий вузол позначається як значення хешу його дочірніх вузлів, а кожен листковий вузол позначається як значення хешу блоку даних. На вершині хеш-дерева Меркла є кореневий вузол. Одним з найпоширеніших алгоритмів хешування для дерева Merkle є хеш Tiger, який добре оптимізований для 64-розрядних процесорів і не має таких уразливостей, як MD5 [89].

Заповнюється дерево Merkle зазвичай знизу догори, і на першому етапі для кожного блоку D_i даних розраховується хеш:

$$Hash_{00} = hash(D_0), \dots, Hash_{0i} = hash(D_i) \qquad (10)$$

Отримані значення записуються у вигляді листків дерева. Блоки що знаходяться рівнем вище заповнюються хешами від суми нащадків:

$$Hash_0 = hash(Hash_{00} + Hash_{01}), \quad (11)$$

В даному випадку під операцією + маємо на увазі конкатенацію. У випадку ж якщо кількість хешів непарна, останній хеш дублюється, не порушуючи логіку побудови дерева хешів. Рекурсивно повторюємо вказану процедуру, поки не отримаємо єдиний хеш *merkle_root* який і буде зберігатися в блоці як корінь дерева.

Виходячи з того, що блоки даних, які зберігає система виявлення вторгнень зазвичай не пов'язані один з одним, важливим для нас є момент перевірки наявності та цілісності конкретного блока, не перераховуючи хеші інших блоків. Для підтвердження наявності блока даних достатньо мати кореневий хеш, а також хеші верхнього рівня інших гілок. Так наприклад для дерева, що містить 4 блоки даних невиконання умови:

$$TopHash = hash\big(Hash_0 + hash(hash(D3) + Hash_{11})\big) \quad (12)$$

Свідчитиме про те, що елемент D*3* відсутній, або був змінений. В той же час верифікація блока не вимагатиме перерахунку хешів дерева Merkle взагалі, оскільки для перевірки блока зазвичай достатньо лише кореневого хешу дерева.

Для розподіленої системи виявлення вторгнень, використання дерева Merkle є більш вигідним, оскільки оскільки має більш просту реалізацію та дає можливість поступово вилучати непотрібні дані з блоків, залишаючи лише їх хеші. Окрім того, дерево Merkle дозволяє використовувати практично будь-який алгоритм хешування, що також достатньо вигідне для блокчейн підсистеми, і дозволяє уникнути перевантаження зайвими функціями і використовувати ту ж саму хеш функцію, яка використовується для хешування заголовків блоків. Використання дерева Merkle також

допоможе швидко шукати дані, що зберігаються на вузлах, оскільки будь-який вузол може швидко оголосити лише доступні корені, а інші вузли побачать, чи є у цього вузла дані, які їм потрібні. Звичайно, така ідея тимчасового зберігання журналу подій у майбутньому спричинить таку ситуацію, коли старі журнали подій будуть втрачені назавжди через деякий час, але в більшості випадків старий журнал подій стає менш корисним і практично не використовується для класифікації подій, оскільки аналіз великого об’єму журналів витрачатиме багато обчислювальних ресурсів, а інформація з такого журналу через деякий час стане неактуальною і навпаки знижуватиме точність класифікації.

Базову структуру блока, який задовольнятиме вимоги розподіленої системи виявлення вторгнень показано на рисунку 2.5. Як бачимо основним полем для блока є поле заголовок блоку, яке містить основну інформацію про блок, включаючи його розмір, номер блока в ланцюжку, хеш дерева з вбудованими в блок даними, хеш дерева із зовнішніми даними, хеш заголовка попереднього блоку, часову мітку створення блоку, а також версію протоколу консенсусу (збереження версії протоколу консенсусу потрібне на випадок його оновлення, для того, щоб для створених раніше блоків використовувались старі правила валідації).

Криптографічне поєднання блоків забезпечується збереженням хеша заголовка попереднього блоку, для кожного блоку окрім початкового, який є першим в ланцюжку і не має попереднього блоку. За таких умов зміна блоку, що знаходиться в середині ланцюжка потребуватиме зміни усіх наступних блоків, а отже чим довше проіснував блок тим більше після нього було створено інших блоків, а отже і тим складніше буде зловмиснику виконати підміну наступних блоків (а при достатній потужності мережі така задача стає неможливою). В блокчейн системах зазвичай кількість блоків, які були створені після блоку називають кількістю підтверджень блока, при цьому в залежності від швидкості створення блоків та протоколу консенсусу для

кожної блокчейн системи існує така кількість підтверджень, після якої можна вважати блок записаним назавжди.

Розрахунок хеша заголовка відбувається шляхом конкатенації усіх елементів заголовка блока у побайтовому вигляді, з подальшим розрахунком значення хеш функції. В якості хеш функції оптимальним варіантом буде використовувати подвійну функцію $Sha256$ як таку, яка має достатньо високий рівень безпеки, а отже підібрати дані які дадуть нам в результаті потрібне значення хешу практично неможливо в рамках актуального проміжку часу, оскільки єдиним способом є повний перебір. Отже, з врахуванням вищесказаного, хеш заголовка блоку матиме наступний вигляд:

$$HHash = Sha256\big(Sha256(Id + Ver + Ts + PHash + DRoot + EDRoot)\big) \qquad (13)$$

де $HHash$- хеш заголовка блоку, Id - номер блока в ланцюжку, Ver - версія протоколу консенсусу, Ts- часова мітка створення блоку в форматі Unix timestamp, $PHash$- хеш заголовка попереднього блоку, $DRoot$- корінь Merkle дерева для вбудованих в блок даних, $EDRoot$ - корінь Merkle дерева для даних, що зберігаються поза межами блоку.

2.3.2 Оцінка ефективності блокчейн мереж

Отже, блокчейн допомагає нам вирішити проблему довіри, але слід пам’ятати, що блокчейн має достатньо серйозний вплив на швидкість обміну інформацією. Тому оцінка ефективності використання апаратних ресурсів блокчейн підсистемою з різними параметрами мережі є важливим етапом проектування розподіленої системи виявлення вторгнень.

Перш ніж створювати модель блокчейн підсистеми слід розглянути, як блокчейн обробляє транзакції (тут ми маємо на увазі, що транзакція - це будь-яка атомарна частка інформації для обміну між вузлами, яка буде

записана в блокчейн). Зазвичай блоки в блокчейні генеруються з деякою стабільною частотою, яка визначена в правилах блокчейну, тому транзакції не можуть бути записані в блок негайно.

На першому кроці, коли транзакція створена, вузол зберігає її в пулі пам'яті та транслює до сусідніх вузлів. При отриманні транзакції кожен вузол також зберігає її в тимчасовому пулі пам'яті та транслює її іншим вузлам, з якими даний вузол має встановлені з’єднання. Таким способом транзакція буде розповсюджена по всій мережі.

Коли генерується новий блок, вузол, який його генерує, записує транзакції з тимчасового пулу пам'яті в блок. Потім вузол транслює створений блок для сусідніх вузлів. Після отримання нового блоку кожен вузол перевіряє, чи відповідає блок всім правилам протоколу консенсусу, і що всі транзакції в блоці відповідають правилам протоколу та програми. Після перевірки отриманого блоку вузол оновлює свій блокчейн - ланцюжок блоків, очищає включені в створений блок транзакції з тимчасового пулу пам'яті та передає блок далі [90].

Але одна з проблем блокчейну полягає в тому, що блоки в більшості випадків мають постійну кількість записів транзакцій. Ось чому є два фактори, які визначають, скільки транзакцій система може приймати та створювати. Перший – це максимальна кількість транзакцій у блоці. Другий – це період генерації блоків. Отже, середня кількість транзакцій за час створення блоку повинна бути меншою, ніж максимальне число транзакцій, що система може обробити, або ж черга тимчасового пулу пам’яті буде постійно зростати, і система аварійно завершить роботу через недостачу пам’яті, або деякі транзакції будуть безповоротно втрачені.

Для оцінки продуктивності блокчейну імітаційну модель засновану на агентах. Такий тип моделі був обраний виходячи з того, що блокчейн системи зазвичай складаються з багатьох автономних вузлів, які працюють самостійно, утворюючи цілісну розподілену систему. В такій системі кожний

вузол мережі представлений окремим агентом. Вузли підключені один до одного, утворюють однорангову мережу. У більшості блокчейн мереж кожен вузол підключений до деякої обмеженої кількості сусідів (для Bitcoin це 8 вихідних з'єднань з вузлами та до 125 з'єднань загалом [91]). Для створення імітаційної моделі було обрано систему anylogic, яка має широкі можливості для створення імітаційних моделей різних типів. Для моделювання стратегії підключення агентів, яка використовужться в блокчейн мережі, anylogic має стандартний тип мережі, який називається “Маленький світ”(“small world”). Цей тип мережі означає, що агенти зв’язані з сусідами і утворюють кільце, але деякі зв’язки встановлюються до далеких вузлів, так, що це може збільшити швидкість розповсюдження мережевих повідомлень [92].

Модель, що тут описується, буде дещо спрощена, оскільки основна її задача оцінити швидкодію блокчейн підсистеми. Робота блокчейну в даній моделі перевіряється в нормальному системному стані та в системі, яка зазнає атаки, що впливає на кількість транзакцій, які будуть створюватися за одиницю часу. У моделюванні звичайного стану ми визначаємо частоту оповіщення приблизно 1 оповіщення за 33 секунди, оскільки, як ми знаємо, більшість систем, що мають прямий доступ до Інтернету та загальнодоступну IP-адресу, постійно зазнають дрібних атак, від сканерів портів та випадкові брутфорс атаки, направлені на вгадування стандартних паролів. Звичайно, це трапляється не так часто, тому швидкість, яку ми обираємо для генерації попереджень, приблизно описуватиме справжню шкідливу діяльність, якої може зазнавати система виявлення вторгнень у нормальному стані.

Що ж стосується стану системи під час атаки, то тут ми обираємо швидкість створення транзакцій близько однієї транзакції за секунду. Звичайно це приблизне значення і може варіюватися в залежності від типу та масивності атаки, але в будь-якому випадку транзакції з інформацією про шкідливі події не будуть створюватися для кожного шкідливого пакета для уникнення перевантажень. У випадку надходження великої кількості

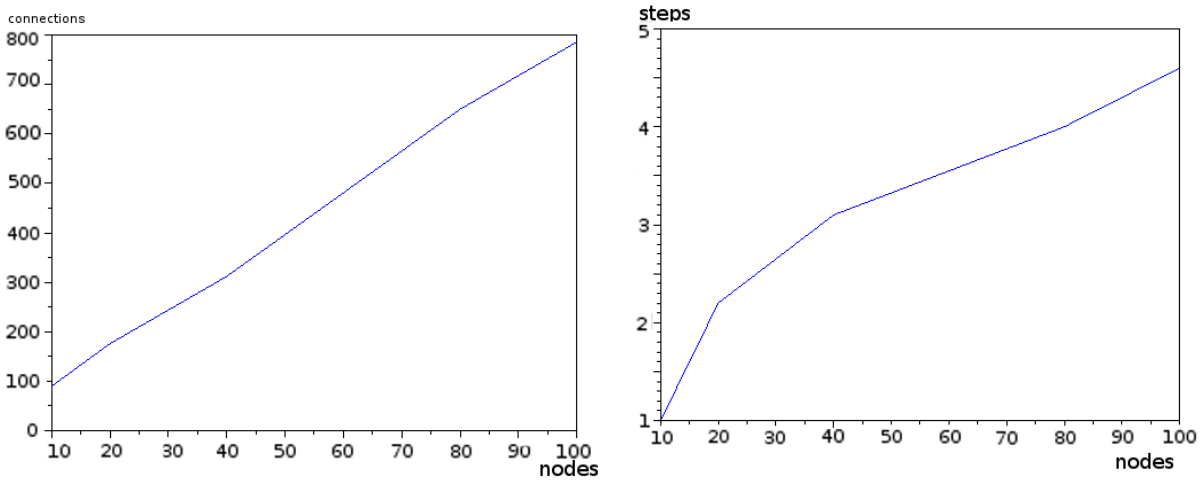
шкідливих подій, ми можемо згрупувати кілька подій, близьких за типом, які надійшли в певний невеликий часовий проміжок, в одне попередження. Отже, зараз ми використовуємо таку частоту генерування подій для представлення загальної поведінки системи під атакою.

Сповіщення в нашій моделі генеруються на мережевих вузлах випадковим чином з експоненціальним розподілом, що підходить для встановлення часу між випадковими вхідними дзвінками або повідомленнями, або, в нашому випадку сповіщеннями системи виявлення вторгнень. Створене сповіщення додається до черги тимчасового пулу пам'яті (у нашому випадку черга пулу пам'яті є окремою для кожного вузла, щоб ми могли дослідити розподіл повідомлень між вузлами) і розподіляється по всіх підключених вузлах, як у реальному блокчейні. Кожен вузол після отримання повідомлення також додає його до черги пулу пам'яті і перерозподіляє по сусідах.

Іншим завданням нашої моделі блокчейн є імітація процесу генерації блоків. Тут ми використовуємо стандартний для більшості схем блокчейн метод – блоки генеруються з практично постійним таймаутом (у нашому випадку постійним). Для спрощення моделі припустимо, що всі вузли мають однакові ймовірності генерувати новий блок (вибір вузла не має істотного впливу на загальну продуктивність блокчейн). Потім вузол, який був обраний для генерації блоку, створює новий блок і записує до нього деяку кількість транзакцій з пулу пам'яті, але не більше максимальної кількості транзакцій для блоку (розмір блоку).

Тепер розглянемо основні моменти, які можуть вплинути на продуктивність розподіленої мережі на основі блокчейн. Перший – це кількість вузлів і напряму пов'язана з цим параметром кількість з'єднань та кількість кроків необхідних для поширення транзакцій. Як крок ми маємо на увазі ситуацію, коли вузол отримує повідомлення, а потім повторно відправляє його до інших вузлів.

Результат роботи імітаційної моделі показує, що при майже постійній кількості з'єднань на вузол ми маємо лінійну залежність між кількістю з'єднань і кількістю вузлів (рис. 2.6 (a)).



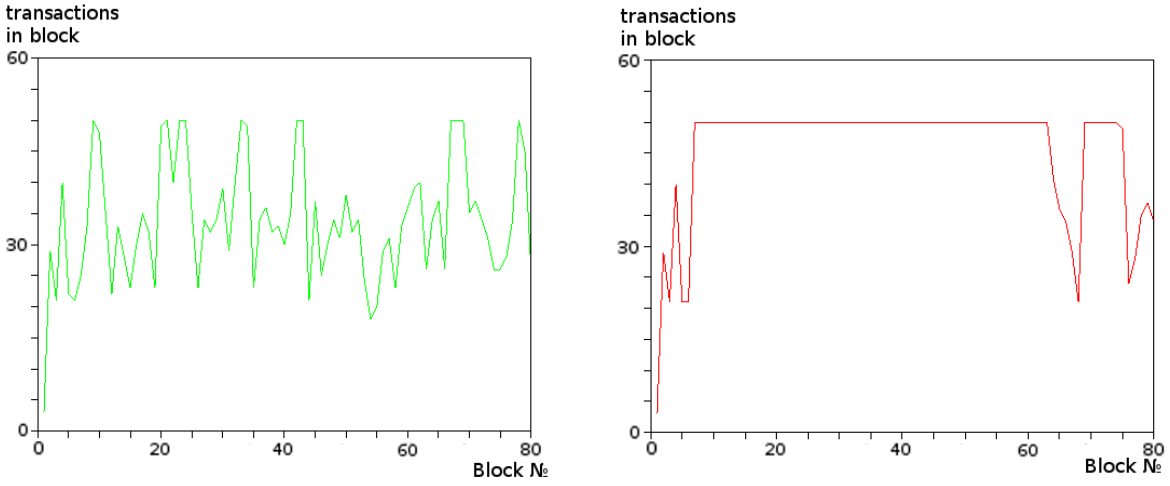
(a) (b)

Рисунок 2.6 – Залежність кількості з'єднань (a) та кроків для повного розповсюдження повідомлення (b) від кількості вузлів

Але кількість з'єднань має серйозний вплив на продуктивність лише на централізованій системі, оскільки центральні вузли повинні обробляти їх усі, але при децентралізованій мережі кожен вузол має обмежену кількість з'єднань. І замість того, щоб надсилати повідомлення безпосередньо на всі вузли мережі, повідомлення кілька разів надсилається та отримується мережевими вузлами, поки воно не досягне останнього вузла і буде повністю розповсюджене по мережі. У такому випадку кількість кроків буде важливішою, оскільки кількість кроків буде пропорційна часу розповсюдження повідомлення. Отже, давайте подивимося, як кількість кроків розповсюдження повідомлень залежить від кількості вузлів (Рисунок 2.6 (b)).

Як бачимо, кількість кроків має логарифмічну залежність від кількості вузлів (для 1000 вузлів це лише близько 6 кроків), тому цей тип децентралізованої мережі підходить для створення великих децентралізованих мереж без потужних центральних вузлів, при цьому обмін повідомленнями буде достатньо швидким.

Але слабким місцем блокчейн систем є запис транзакцій (у нашому випадку це дані з повідомленнями про підозрілі події) до блоків. І головне тут дотримуватися балансу між заповненням блоків та резервуванням ресурсів на випадок атаки або аномального стану мережі.



(а) (б)

Рисунок 2.7 – заповнення блоків для 20 вузлів в нормальному стані (а) та в стані атаки (б)

Але при фіксованому розмірі блока, збільшення розміру призведе до швидкого зростання розмірів блокчейну, а окрім цього великі блоки складніше передати цілісними без розривів з'єднання, а їх перевірка та обробка займає багато часу. І хоча дисковий простір не є великою проблемою в наші дні, коли жорсткі диски з обсягом більше 10 ТБ цілком доступні, поширення блоку та його перевірка можуть бути проблемою, оскільки всі вузли повинні прийняти новий блок в той час як наступний блок буде генеруватися. Тому більшість криптовалют вибирають менший розмір

блоку для свого блокчейну. Наприклад, на даний момент Bitcoin має теоретичний розмір блоку 4 мегабайти [93], а нові блоки генеруються кожні 10 хвилин. Такий блок може містити до 8000 транзакцій розміром 512 байт, це приблизно 13 транзакцій в секунду. І хоча блокчейн з такими характеристиками все ще підходить для використання в контексті криптовалют, для розподіленої системи виявлення вторгнень такий блокчейн не дасть можливості забезпечити достатню швидкодію.

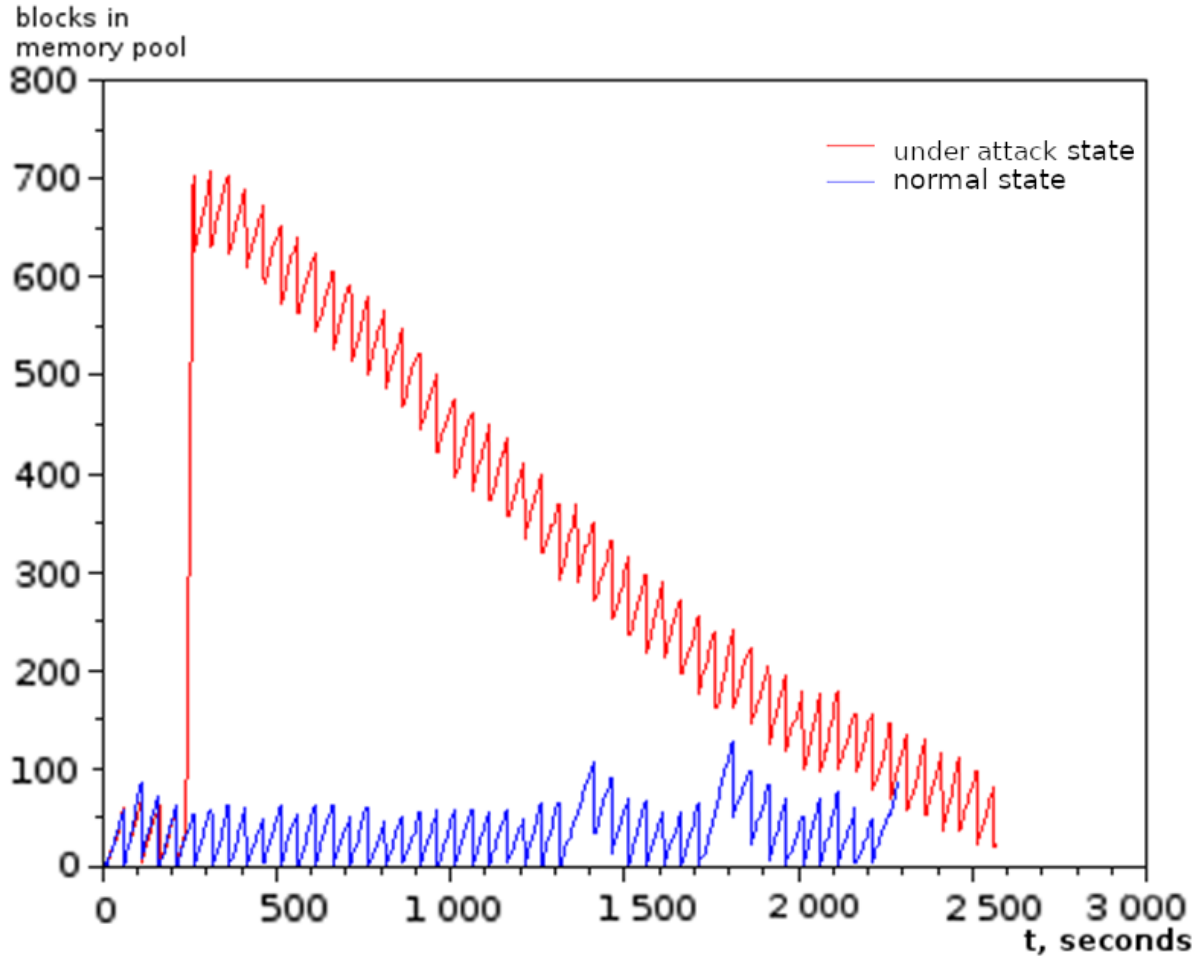


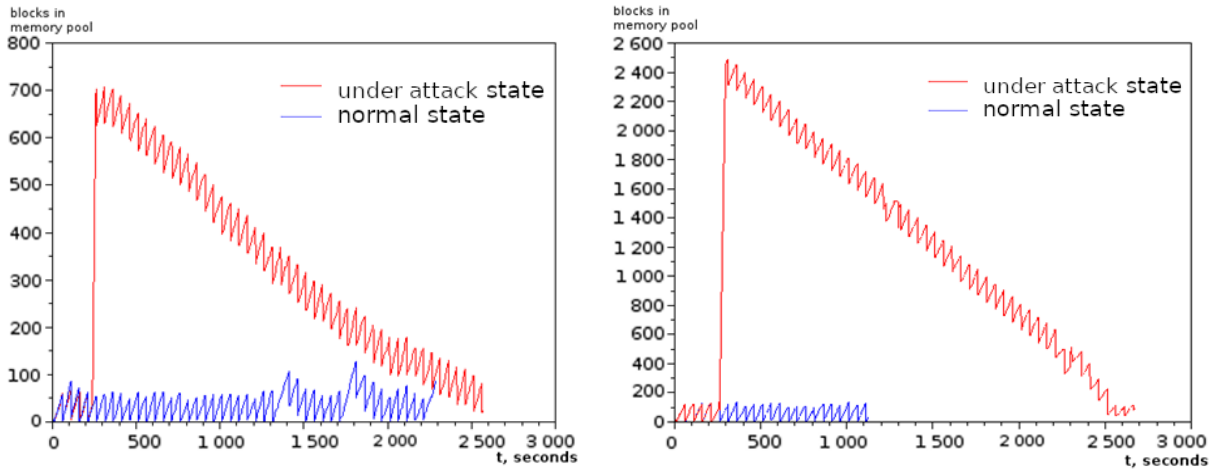
Рисунок 2.6 – Використання пулу пам'яті при розмірі мережі в 20 вузлів блокчейну (50 транзакцій на блок)

Особливістю криптовалют є те, що криптовалюта зазвичай в середньому має стабільний рівень транзакцій. Але у випадку системи виявлення вторгнень велика кількість блоків буде майже порожніми, але у

випадку, коли один або кілька вузлів системи знаходиться під атакою, блоки будуть заповнені великою кількістю транзакцій.

Для нашої моделі ми визначили блоки з постійною кількістю транзакцій, а не кількістю інформації, оскільки ми спростили модель, зробивши всі транзакції однаковими за розміром. А тепер ми будемо оцінювати заповнення блоків та використання пулу пам'яті для нашої моделі з різною кількістю вузлів. Отже, почнемо з 20 вузлів (рисунок 2.7).

Звичайно, розмір блоку, який ми обираємо (для 20 вузлів ми вибираємо 50 транзакцій в одному блоці і швидкість створення блоків рівну одному блоку за хвилину), є невеликим для стану атаки та близьким до граничного для нормального стану з деякими випадковими аномаліями. Але з цих графіків ми можемо побачити, що під час атаки блоки повністю заповнені транзакціями, тому давайте поглянемо на використання тимчасового пулу пам'яті (рис. 2.8).



(а) (б)

Рисунок 2.7: Використання пулу пам'яті, коли мережа містить (а): 40 вузлів (80 транзакцій на блок), (б): 80 вузлів (160 транзакцій на блок).

Тут ми бачимо, що атака викликає величезне збільшення пулу пам'яті, і відновлення розміру пулу до нормального стану займає приблизно в 30 разів більше часу, ніж тривалість атаки, і приблизно 64 блоки. Тож давайте

розглянемо подібні графіки для 40 вузлів (рис. 2.9 (а)) та 80 вузлів (рис. 2.9 (б)).

Тут ми можемо спостерігати ту саму ситуацію, навіть для короткочасних атак потрібно багато часу, щоб записати всі транзакції з пулу пам'яті (для цих двох випадків потрібно близько 45 блоків). І така ситуація є достатньо поганою для системи, що базується на блокчейні, оскільки через те, що пул пам'яті містить багато незаписаних транзакцій, система нестабільна. І така поведінка тимчасового пулу транзакцій, робить систему вразливою для тривалих атак, оскільки пул пам'яті може бути переповнений і вузол може аварійно завершити роботу.

2.4 Метод вибору протоколу консенсусу для розподіленої системи виявлення вторгнень на основі блокчейн

У розподілених системах, заснованих на блокчейні, зазвичай немає центрального вузла, якому можуть повністю довіряти інші вузли. Обмін інформацією між вузлами в системі, що базується на блокчейн технології, здійснюється шляхом трансляції інформації (транзакції) на всі доступні вузли. Потім кожен вузол ретранслює цю інформацію на інші доступні вузли. Таким чином інформація розподіляється по всій мережі блокчейн. Але інформація спочатку все одно зберігається лише в пулі пам'яті. Інформація записується в блокчейн лише тоді, коли один із вузлів запише її в блок, і цей блок буде прийнятий мережею. Отже, вузли, присутні в системі блокчейн, мають можливість генерувати наступний блок, тому системі потрібні протоколи для досягнення консенсусу між вузлами, щоб впевнитися, що всі вузли працюють за однаковими правилами з однаковою інформацією. Протокол консенсусу також допомагає вирішити конфлікти в ланцюжку блоків, які можуть іноді виникати, коли декілька вузлів намагаються створити блок одночасно. У разі такого роду конфліктів блокчейн може на деякий час стати роздвоєним. Тоді

алгоритм консенсусу буде використаний для вибору основної гілки розгалуженого блокчейну. Інша гілка буде позначена як недійсна та видалена.

Існує кілька популярних алгоритмів консенсусу, які використовуються в більшості базових систем блокчейнів (переважно криптовалют):

Доказ роботи(Proof of work, PoW) – був першим алгоритмом консенсусу, який був використаний у мережі Bitcoin. Цей алгоритм допомагає вибрати в децентралізованій мережі блокчейн без контролю доступу вузол для збереження записів у блок. Вибір випадкового вузла тут не підходить, оскільки такий варіант має багато вразливих місць. Наприклад, шкідливий вузол може спробувати створити блоки з неправильною інформацією. Цей метод дозволяє вузлу успішно прийняти блок, коли може бути доведено, що витрачений заздалегідь визначений обсяг обчислювальних ресурсів (відомий як «робота») [94]. У більшості випадків робота полягає в обчисленні криптографічних хешів, щоб знайти той, який відповідає деяким вимогам. Вузли, які створюють нові блоки та обчислюють їх криптографічні хеші, називаються майнерами, а процедура PoW – майнінгом [58]. В такому випадку заголовок блоку містить поле nonce, і майнери, часто змінюючи значення nonce та набір транзакцій включених до блоку, намагаються підібрати хеш-значення, що відповідатиме умовам протоколу. Консенсус вимагає, щоб обчислене значення мало бути рівним або меншим за певне задане порогове значення – складність. Складність роботи регулюється автоматично для досягнення стабільної швидкості генерації блоків шляхом періодичної корекції порогового значення спираючись на поточну частоту створення нових блоків. Коли один вузол досягає цільового значення, він передаватиме блок іншим вузлам, а інші вузли повинні взаємно підтверджувати правильність хеш-значення. Якщо блок перевірено, інші вузли додадуть цей новий блок до своїх власних блокчейнів. У децентралізованій мережі з PoW допустимі блоки можуть генеруватися одночасно, коли кілька вузлів знаходять відповідний nonce

майже одночасно. В результаті можуть створюватися паралельні гілки. Однак малоймовірно, що дві конкуруючі гілки одночасно генеруватимуть наступний блок і в подальшому. Тому у протоколі PoW ланцюжок, який в подальшому стає довшим, вважається автентичним, а інший видаляється, а блоки, які не були занесені в основний ланцюжок але були присутні в альтернативному повертаються в тимчасовий пул. Майнерам доводиться робити багато комп'ютерних розрахунків при використанні PoW протоколу, які витрачають багато обчислювальних ресурсів та енергії. Тому для направлення витрат в корисне русло були розроблені деякі протоколи PoW, в яких робота може мати деякі побічні результати, які можна використати в подальшому. Наприклад, Primecoin [95] шукає спеціальні ланцюжки простих чисел, які в подальшому можна використовувати для математичних досліджень. Zhang та ін. стверджують, що алгоритм Rem є найбільш підходящим для систем із апаратним забезпеченням з невеликою потужністю. Ідея Rem полягає в тому, щоб замінити енергоємний Proof of Work (PoW)" на Proof of Useful Work (PoUW), де майнери надають надійні звіти про цикли процесора, присвячені корисним робочим навантаженням [96]. Але головна проблема тут полягає в тому, що цей алгоритм базується на розширеннях Intel-Software-Guard SGX, що дозволяє виконувати надійний код в ізольованому середовищі, де не допускається фальсифікація, і SGX може віддалено довести результат таких виконань. Це означає, що цей алгоритм залежить від платформи і може працювати лише на процесорах Intel. Цей тип консенсусного протоколу не підходить для розпоідленої IDS, оскільки створення нових блоків вимагає великої кількості обчислювальних ресурсів, що означає великі витрати енергії, тому ця система буде дуже дорогою і не ефективною, особливо в контексті невеликих SMB мереж.

Доказ частки(Proof of stake, PoS) – енергоефективна альтернатива алгоритму PoW. Для підтвердження блоку майнерам потрібно довести право власності на певну кількість криптовалюти. Цей метод забезпечує досягнення

консенсусу шляхом врахування як випадкового відбору, так і впливу (відомого як "частка") суб'єктів-учасників. Передбачається, що вузли гарантують цілісність блоків, коли вони мають велику частку в ланцюговій [97] [98]. Вибір може базуватися на різних правилах, наприклад, на основі суми валюти, але такий вибір є досить несправедливим, оскільки вузли, які мають більше криптовалюти будуть домінвати в мережі. Для уникнення таких ситуацій створено багато алгоритмів які згладжують різницю вірогідностей, роблячи розподіл права на генерацію блока більш рівномірним. Зокрема, Blackcoin [99] використовує рандомізацію для вибору наступного вузла-генератора. Такий алгоритм використовує формулу, яка шукає найнижче хеш-значення в поєднанні з розміром ставки. Peercoin [100] надає перевагу вибору монет за віком. У Peercoin старші та більші набори монет мають більшу ймовірність видобутку наступного блоку. У порівнянні з PoW, PoS використовує значно менше обчислювальних ресурсів а відповідно і електроенергії, що робить його більш ефективним та дешевим. Але при цьому краща енергоефективність спрощує атаки на такий блокчейн. З іншого боку, цей тип консенсусного протоколу є одним з найбільш підходящих для IDS завдяки своїй енергоефективності та простоті реалізації, але вимагає деяких модифікацій для такого контексту використання.

Доказ минулого часу(Proof of Elapsed Time) – алгоритм на основі довіреної обчислювальної платформи Intel SXG. Цей метод забезпечує досягнення консенсусу, вимагаючи від кожного потенційного верифікатора спільного використання безпечного та випадкового часу очікування із надійного середовища виконання. Кожен користувач, генеруючи блок, також повинен сформувати підтвердження діяльності очікування за допомогою обладнання SGX, яке подається разом із блоком. Генерація випадкових чисел базується на певному розподілі, визначеному системою заздалегідь. Коли новий блок подається в систему, SGX допомагає вузлу, що створює блок, генерувати доказ часу очікування. Цей доказ можна легко перевірити іншими

вузлами за допомогою технології SGX. За допомогою статистичного тесту визначають, чи дійсно час очікування відповідає зазначеному розподілу [101] [102]. Доказ минулого часу також виглядає придатним для IDS, оскільки він базується на очікуванні, йому також не потрібна велика потужність обчислень. Але Lin Chen та ін. показав вразливості поточної конструкції протоколу [101], при цьому даний протокол є платформозалежним.

Доказ Важлиивості (Proof of Importance, PoI) – це алгоритм, який вперше представлений в системі NEM. Облікові записи з вищим показником важливості матимуть більшу ймовірність бути обраними для створення нового блоку. Розрахунок важливості відбувається на основі значення накопиченого ресурсу а також статистики здійснених акаунтом транзакцій з іншим при впливає не тільки кількість і розмір транзакцій, а і важливість учасників, з якими відбувається обмін даними. Енергоспоживання такого протоколу низьке [103]. Але безпека такого протоколу залишається під питанням, оскільки для шкідливих вузлів існує можливість підробки значення важливості за рахунок відправки транзакцій один одному, тому для безпечного використання такого протоколу слід використовувати складні алгоритми розрахунку важливості, які дозволять запобігти штучному відвищенню важливості за рахунок сміттєвих транзакцій.

DPOS (Делеговане підтвердження ставки). Основна різниця між PoS та DPOS полягає в тому, що PoS є безпосередньо демократичним, тоді як DPOS є представницько-демократичним. Зацікавлені сторони обирають своїх делегатів для створення та перевірки блоків. Маючи значно меншу кількість вузлів для перевірки блоку, блок можна швидко підтвердити, що дає можливість швидко підтверджувати транзакції. Такі параметри мережі, як розмір блоку та інтервали блоків, можуть бути налаштовані делегатами. Крім того, користувачам не потрібно турбуватися про нечесних делегатів, оскільки їх можна легко виключити шляхом голосування. DPOS є основою Bitshares [104].

PBFT (Практична візантійська відмовостійкість) – консенсусний алгоритм практичної візантійської терпимості до помилок, що відповідальний за роботу в асинхронних мережах [85]. Hyperledger Fabric використовує PBFT як алгоритм консенсусу, оскільки PBFT може стабільно працювати, поки кількість шкідливих вузлів не перевищить 1/3 від усіх наявних. Новий блок визначається в раунді. У кожному раунді первинний відбір обирається за деякими правилами. І він відповідає за замовлення операції. Весь процес можна розділити на три етапи: заздалегідь підготовлений, підготовлений та здійснений. У кожній фазі вузол вступає в наступну фазу, якщо отримав голоси понад 2/3 всіх вузлів. Отже, PBFT вимагає, щоб кожен вузол був відомий мережі. Як і PBFT, протокол зоряного консенсусу (SCP) [105] також є протоколом візантійської угоди. Існують також і інші протоколи консенсусу, які базуються на PBFT, такі як спрощена візантійська відмовостійкість та делегована візантійська відмовостійкість, але всі ці алгоритми можуть бути використані лише у блокчейні з контролем доступу.

Змішана Візантійська Відмовостійкість (MBFT, Mixed Byzantint Fault Tolerance) протокол базується на двох шарах BFT, і використовує технології сегментації та розшарування. Для підвищення захищеності та стійкості до помилок в даному протоколі впроваджено механізми випадкового вибору вузла та кредитування. Вибір вузла для верифікації транзакцій підтверджується іншими вузлами в мережі, а вузли для верифікації розподіляються на два шари. Перший шар – низькорівнева група консенсусу (LCG) та високорівнева група консенсусу (HCG). Група високорівневого консенсусу – отримує мініблок від низькорівневої групи, після того, як консенсус було встановлено, і перевіряє валідацію підписів, хешів та відсутність конфліктів перед подальшою обробкою [106]. Даний протокол забезпечує низьку затримку і велику швидкість обробки транзакцій.

Round Robin (Круговий консенсус) Консенсусна модель для приватних блокчейн-мереж, де вузли вибираються для створення блоків псевдовипадковим чином, але вузол повинен зачекати кілька циклів створення блоку, перш ніж він знову може бути обраним для додавання нового блоку. Ця модель гарантує, що жоден учасник не створює більшість блоків. Аналогічно до PoET протокол використовує довірену обчислювальну платформу Intel SXG, що позитивно впливає на безпеку протоколу, але ставить вимоги до апратної платформи. Позитивна сторона цього протоколу – відсутність криптографічних головоломок і низьке енергоспоживання. Протокол консенсусу Round Robin, використовується в таких блокчейн проектах як Multichain і Tendermint [107] [108].

Таблиця 2.1 - Порівняння алгоритмів консенсусу

Властивість / Протокол консенсусу	Тип Блокчейн	Витрати Енергії	Відмово-стійкість	Масшта-бування	Підходить для IDS	Особливості
PoW	Відкритий	Висока	50%	Добре	Не підходить	Високе енерго-споживання
PoS	Відкритий	Низька	50%	Добре	Підходить	Вимагає адаптації для використання в IDS
DPoS	Закритий, Відкритий	Низька	50%	Добре	Підходить	Аналогічно до PoS
PoET	Закритий	Низька	Наявність шкідливих вузлів становить загрозу	Погане	Підходить	Вимагає специфічне обладнання
PoI	Закритий, Відкритий	Низька	Наявність шкідливих вузлів становить загрозу	Добре	Підходить	Складний розрахунок важливості, вузли не рівнозначні
PBFT	Закритий, Відкритий	Дуже низька	33%	Погане	Підходить	Для невеликих груп консенсусу
MBFT	Закритий, Відкритий	Дуже низька	33%	Погане	Підходить	Для невеликих груп консенсусу
Round Robin	Закритий	Дуже низька	Наявність шкідливих вузлів становить загрозу	Погане	Підходить	Вимагає специфічне обладнання

Тепер нам потрібно порівняти переваги та недоліки різних алгоритмів консенсусу для системи виявлення вторгнень на основі блокчейну. Порівняння алгоритмів консенсусу показано в таблиці 2.1.

Перш за все, нам потрібно поглянути на вимоги до обладнання для використання консенсусних алгоритмів, оскільки більшість PoW алгоритмів вимагають специфічного обладнання, такого як графічні процесори або чіпи ASIC, щоб створити стабільну та безпечну систему. PoET також вимагає специфічної апаратної платформи, яка підтримує технологію Intel SGX, що вносить додаткові обмеження для системи виявлення вторгнень.

Отже, алгоритм PoW не підходить для використання в системі виявлення вторгнень на основі блокчейну через низьку енергоефективність та специфічні вимоги до обладнання. Підтвердження PoW майнінг використовуватиме багато обчислювальних ресурсів (і звичайно електроенергії), що зробить IDS занадто дорогою для утримання [109].

PoS та DPoS – це два алгоритми, які базуються на накопиченні ставок, тому вони не потребують великих обчислювальних потужностей і можуть бути застосовані для випадку IDS. Незважаючи на те, що PoS представляється найбільш перспективним механізмом заміни PoW, PoS все ще стикається з декількома вразливими місцями.

1) Безкоштовне моделювання: Безкоштовне моделювання є основною вразливістю схем PoS, не заснованих на BFT, особливо PoS на основі ланцюга, в яких PoS використовується для імітації процесу PoW. Безкоштовне моделювання буквально означає, що будь-який учасник може імітувати будь-який сегмент історії блокчейну не виконуючи ніякої реальної роботи, окрім спекуляції, оскільки PoS не вимагає інтенсивних обчислень, тоді як блокчейн фіксує всю історію ставок. Це може дати зловмисникам короткі шляхи для створення альтернативного блокчейну. Інші атаки зазвичай базуються на безкоштовному моделюванні [110].

2) Ніщо на кону: Ніщо на кону – це перша виявлена проблема безкоштовного моделювання, яка впливає на PoS на основі ланцюга. Він також відомий як проблема "мультиставки" або "раціонального розгалуження". На відміну від PoW-майнера, PoS-майнеру потрібні додаткові ресурси, для перевірки транзакції та генерування блоків на декількох конкуруючих ланцюгах одночасно. Отже, якщо значна частина вузлів виконує стратегію мультиставки, зловмисник, що має набагато менше 50% токенів, може здійснити успішну атаку подвійного витрачання [111] [112].
3) Підкуп постфактум: Ключовим фактором, що сприяє задній атаці, є публічна доступність історії ставок на блокчейні, яка включає адреси зацікавлених сторін та суми ставок. Зловмисник може спробувати підкупити зацікавлені сторони, котрі колись мали значні ставки, і якщо зловмиснику вдасться це зробити, можлива атака подвійної витрати [113].
4) Атака на дальню дистанцію: придумана засновником Ethereum Віталіком Бутеріном [114]. Передбачає, що невелика група зловмисників, що змовляються, може створити довший ланцюг, який починається з першого блоку. Оскільки, у випадку, коли зацікавлених сторін небагато, зловмисники можуть дуже швидко розробити шкідливий ланцюг і переробити всі блоки PoS. Протоколи BFT використовують механізм контрольних точок, щоб впевнитись, що домовленості остаточні та старі записи можуть бути безпечно відкинуті.
5) Ризик централізації: PoS стикається з ризиком централізації багатства, подібним до PoW. У PoS майнери можуть законно реінвестувати свій прибуток у ставки, що дозволяє тому, хто має велику суму невикористаних токенів, стати ще багатшим і врешті-решт досягти монопольного статусу [110].

Вказані недоліки дають нам зрозуміти, що PoS протокол можна застосовувати, але з урахуванням його особливостей, інакше це може призвести до стороннього втручання в роботу блокчейн.

Доказ минулого часу також може бути хорошим рішенням, але наразі цей алгоритм має вразливість та вимагає специфічне обладнання для своєї роботи, тому це рішення для нашого випадку не оптимальне.

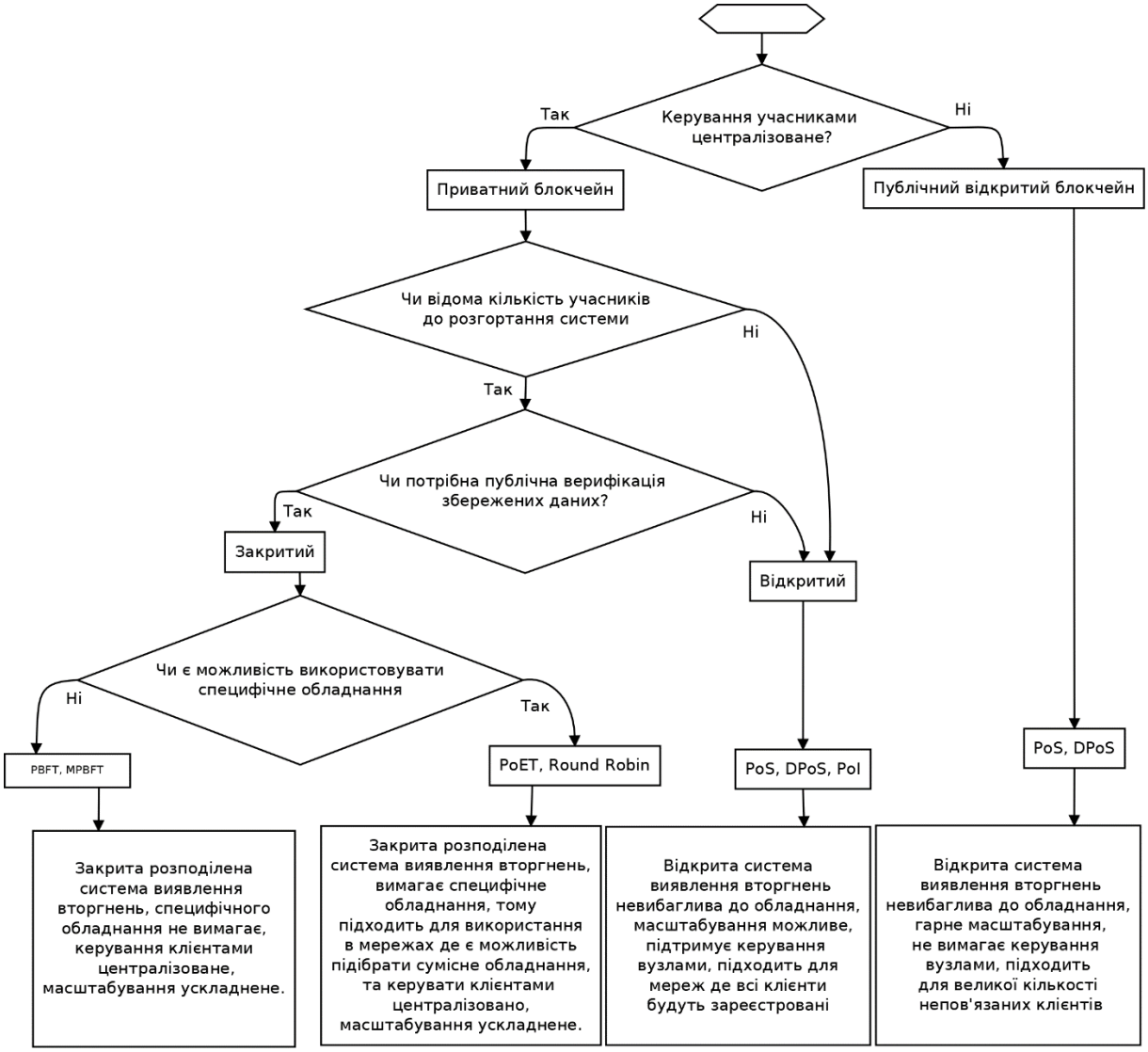


Рисунок 2.10 – Схема методу вибору протоколу консенсусу для розподіленої IDS

Система виявлення вторгнень може бути побудована як на закритому, так і на відкритому блокчейні, тому алгоритми PBFT також підходять. Але ці алгоритми ускладнять масштабування системи блокчейн. Спираючись на вище вказані особливості протоколів консенсусу можна виділити наступну схему (рисунок 2.10) метода вибору протокола для розподіленої системи виявлення вторгнень.

Отже, виходячи з вище перерахованих переваг та недоліків алгоритмів консенсусу, найкращим алгоритмом консенсусу для розподіленої системи виявлення вторгнень в SOHO та SMB мережах буде PoS-подібний алгоритм. Щоб більш детально визначитися з типом PoS протоколу слід спочатку розглянути його основні різновиди.

PoS на основі ланцюжка – це рання схема PoS, запропонована як альтернативний механізм генерації блоків до PoW. Працює в рамках консенсусу Накамото зберігається розповсюдження повідомлень, правила перевірки блоків та правило найдовшого ланцюга. Майнер може розгадати хешувальну головоломку лише один раз за певний період часу. Оскільки складність хешування зменшується зі значенням ставки майнера, очікувана кількість спроб хешування для майнера, щоб вирішити задачу, може бути значно зменшена, якщо величина його ставки висока. прикладами PoS на основі ланцюжка є криптовалюти Peercoin [100] та Nxt [115].

PoS на основі ланцюжка витримує до 50% зловмисних ставок. Якщо зловмисники контролюють більше 50% ставок, вони можуть створювати свій ланцюжок блоків швидше за інших і здійснити атаку подвійної витрати. Оскільки ставка реєструється в транзакціях, до зловмисних вузлів можуть бути застосовані санкції, наприклад у вигляді анулювання їх ставок.

PoS на основі комітету – є альтернативою попередньому варіанту PoS, PoS на основі комітету пропонує створити комітет зацікавлених сторін і дозволити учасникам комітету генерувати блоки по черзі. Для отримання такого комітету в розподілений мережі часто використовується схема

безпечних багатосторонніх обчислень (MPC). MPC – це варіант розподілених обчислень, де кілька сторін мають індивідуальні вхідні дані, отримають однаковий результат [116]. Процес MPC в PoS, що базується на комітетах, по суті реалізує метод, який приймає на вході поточний стан блокчейну, що включає значення ставок від усіх зацікавлених сторін, і дає на виході псевдовипадкову чергу зацікавлених сторін, на основі, якої і формується комітет, який створює блоки. Ця послідовність лідерів повинна бути однаковою для всіх зацікавлених сторін. Учасники які мають вищі значення ставок, можуть зайняти більше місць у такій черзі.

Незважаючи на впорядковану схему пропонування блоків, PoS на основі комітетів, як і раніше дотримується правила найдовшого ланцюга. Поки зловмисна сторона має менше 50% ставок, блокчейн буде підтримуватися чесними учасниками. Але розширення комітету може призвести до значного зниження продуктивності протоколу. Процес виборів до комітетів, що базується на раундах, із заздалегідь визначеною тривалістю туру стикається з проблемами масштабування, що можна виправити шляхом обмеження розміру комітету шляхом введення вимоги до мінімальної ставки [110].

PoS на основі BFT – на відміну від двох попередніх варіантів не використовує протокол Накамото, при цьому PoS на основі BFT (або гібридний PoS-BFT) включає додатковий рівень консенсусу BFT, який забезпечує швидку та детерміновану фіналізацію блоків. Правило найдовшого ланцюга, в даному випадку, може бути безпечно замінено правилом стабільної контрольної точки для визначення стабільного головного ланцюга. Популярні протоколи блокчейнів PoS на основі BFT включають Tendermint [117], Algorand [118] та Casper FFG [119].

Делегований PoS (DPoS) – можна розглядати як демократичну форму PoS на базі комітету, оскільки комітет (консенсусна група) обирається шляхом делегування публічних ставок. В даний час його використовують

EOSIO [120] та Cosmos [121]. DPoS був розроблений для управління розміром групи консенсусу, щоб зменшити накладні витрати на обмін повідомленнями протоколу консенсусу. Вибори делегатів називаються процесом делегування. Фактично, процес делегування та отримання голосів може включати сторонні стимули. Як правило, делегату, що претендує на участь, потрібно залучити достатньо голосів від звичайних власників ставок. Це часто досягається шляхом пропонування бонусів та підвищення репутації за допомогою пропагандистських кампаній. Віддавши голос делегату через транзакцію в блокчейн, користувач криптовалюти довіряє делегатові власну частку [120].

2.5 Адаптація та підвищення ефективності роботи протоколу консенсусу PoS для розподіленої системи виявлення вторгнень на основі блокчейн

Простий варіант доказу ставки, на основі ланцюжка, являє собою відкриту модель IDS, яка допускає приєднання будь-якого сумісного вузла в мережу без будь-яких обмежень.

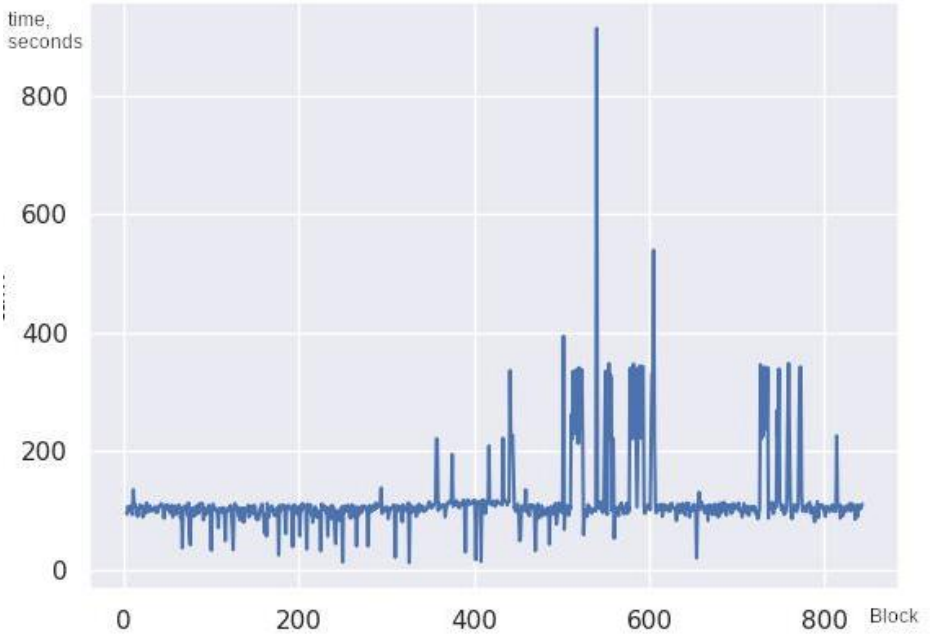


Рисунок 2.11 – Час генерування блоків для системи.

Консенсус на основі ланцюжка PoS використовується лише в відкритому блокчейні, тому IDS, заснований на цьому типі консенсусного протоколу, буде повністю децентралізована. Це зробить систему більш стійкою до зовнішнього впливу та надійною.

Класичний алгоритм PoS на основі ланцюга не повністю відповідає нашим цілям. Як ми бачимо у більшості криптовалют, протокол консенсусу PoS працює на основі ставки деякої кількості криптовалюти. Але цей спосіб нам не підходить, оскільки в нашому випадку транзакція не пов’язана з жодною криптовалютою. Отже, найкращим альтернативним значенням для ставки у нашому випадку буде час, оскільки його легко виміряти на основі міток часу блоків.

Структура блоків для нашого алгоритму консенсусу залишається стандартною. Блок має заголовок з кількома загальними полями. Перш за все, це хеш попереднього блоку, який допомагає зв’язати окремі блоки в ланцюжок. Іншим важливим полем є мітка часу, яка використовується для перевірки правильності хешу блоку. У нашому випадку мітка часу також використовується вузлами для вимірювання значення накопиченого часу. Для спрощення обчислення нам не потрібно зберігати накопичений час у блокчейні, тому що ми можемо просто виміряти його для кожного вузла з транзакції про приєднання до мережі та блоків, створених цим вузлом. Ще одне поле – це складність блоку, яка буде використана для створення наступного блоку. Це значення обчислюється на основі поточної швидкості зростання блокчейна, а в нашому випадку також базується на черзі поточних блоків у пулі пам'яті. І найважливіше поле – це поточний підпис блоку.

Перш за все валідатор перевіряє можливість створення блоку (доказ відповідності), використовуючи хеш-функцію і порівнюючи її з цільовим значенням для поточного валідатора. Тож вхідними значеннями для нашої хеш-функції будуть PublicKey та поточний час блоку. Умова генерування

нового блоку виражена формулою. 6, який ми отримуємо, модифікуючи стандартне рівняння PoS [122]:

$$Hash(PubKey_V||t_{block})<T_b*d_{current}*t \quad (14)$$

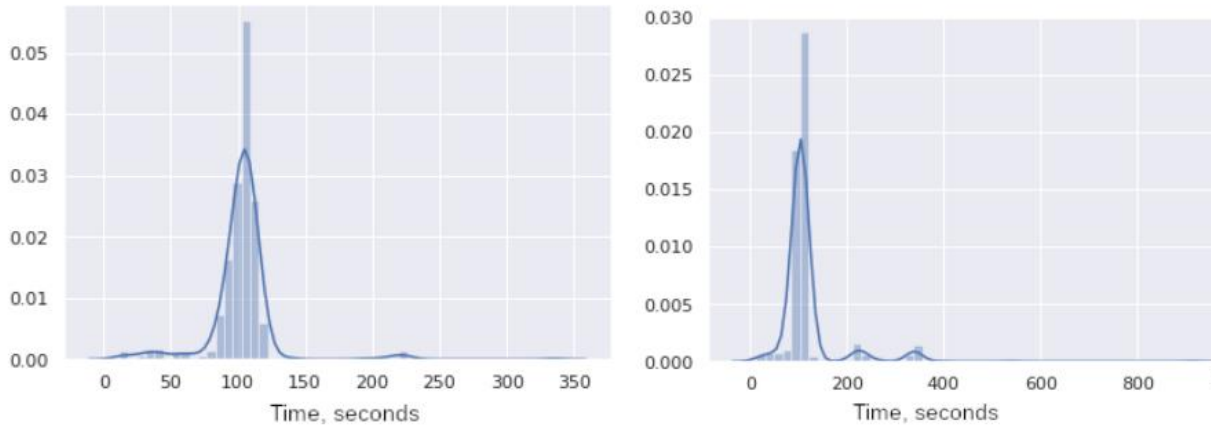
де $PubKey_v$ - це поточний відкритий ключ вузла перевірки, а t_{block} - поточний час блоку, T_b - основна ціль, яка є постійною, але може бути змінена за допомогою параметра складності блоку $d_{current}$ і множиться на поточну валідаційну ставку t_{stake}. Отже, чим вище значення ставки - тим більша ймовірність, що вираз (14) буде правдою.

Розгалуження в ланцюжку для нашого алгоритму будуть вирішені стандартним способом – вузли вважають, що правильний ланцюжок при розгалуженні – це ланцюжок, де сума ставок для всіх блоків є найвищою (відповідає більшій кількості роботи в консенсусі PoW) і при цьому мінімальна різниця значень ставок перевищує певне заздалегідь визначене число. Така умова дозволяє уникнути постійного перемикання між альтернативними ланцюжками і обирати головним той ланцюжок, який значно випереджає інші.

Але, як ми можемо бачити вище, просте використання значення часу знижує безпеку протоколу, оскільки деякі шкідливі вузли можуть ігноруючи створення блоків накопичити велику ставку, а потім використовувати її для створення альтернативного ланцюжка. Отже, нам потрібен певний спосіб обмежити значення ставки часу. Найпростішим рішенням тут є зробити значення ставки часу обмеженим після деякого значення, як у Peercoin, але це обмежує значення ставки, і шкідливі вузли все ще мають можливість отримати максимально можливе значення. Іншим рішенням є використання модульної арифметики - тоді встановлене значення буде обмежене модулем і буде періодично обнулятись протягом часу. Це буде несправедливо у випадку криптовалют, де вузол-переможець отримує винагороду за створення нового блоку, оскільки обмеження ставки на основі модульної арифметики може призвести до пропуску винагороди. Але у випадку, якщо вузли IDS не

отримують винагороди за створення блоків, то це лише мотивуватиме вузли залишатися в мережі якомога довше, щоб використовувати свою ставку більш ефективно. Також обмеження максимального значення ставки таким чином допоможе зменшити кількість зацікавлених сторін з високим значенням ставки та уникнути великої кількості роздвоєних ланцюжків, хоча при використанні PoS повністю уникнути розгалужень неможливо.

Таке обмеження ставки допоможе запобігти ситуації, коли шкідливий вузол буде ігнорувати спроби створення блоку, щоб накопичити велику ставку і матиме майже 100% ймовірність генерування блоку. Імітаційна модель показує, що цей консенсусний протокол дозволяє досягти стабільного створення блоків для мереж із невеликою кількістю вузлів [84].



(a) (b)

Рисунок 2.12 – Розподіл часу генерації блоку для (a) - двох вузлів, (b) - трьох вузлів.

Але слід мати на увазі, що даний протокол консенсусу не захищає від появи шкідливих вузлів (якщо у нас публічний відкритий блокчейн), але є кілька методів, які дозволяють блокувати шкідливі вузли, щоб вони не могли брати участь у створенні блоку і втрачали всю частку . Це зробить спроби атаки незручними для зловмисника [123].

Тестова реалізація такого типу протоколу консенсусу показує, що генерація блоків в процесі роботи стабільна і достатньо рівномірна для

забезпечення прийнятної швидкодії системи виявлення вторгнень. Конфігурація для двох і трьох вузлів перевірялася з часовим модулем 120 секунд. Таке тестування показує, що час, який використовується для створення одного блоку, у більшості випадків становить менше 120 секунд. Результати вимірювання часу показані на рисунку 2.11. Розподіл при цьому показує, що збільшення кількості вузлів може збільшити час, який використовується для створення блоку через виникнення конфліктів в ланцюжку (рисунок 2.12).

Виходячи зі стабільної рівномірної генерації блоків та оптимального використання процесорних ресурсів, даний протокол консенсусу буде зручним для побудови децентралізованої розподіленої системи виявлення вторгнень.

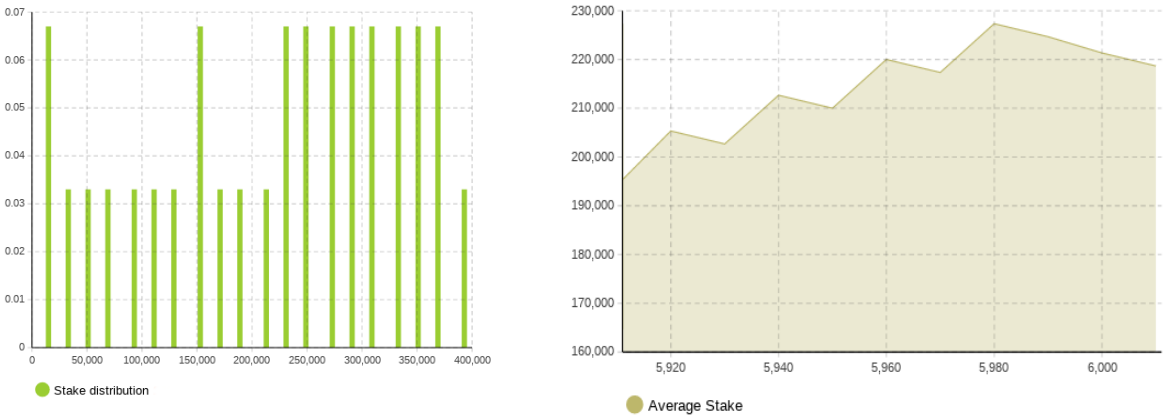
Для оцінки стану блокчейну з такою модифікацією протоколу було створено мультиагентну модель, яка імітує роботу вузлів блокчейну, які використовують час в якості ставки. Така модель є дещо спрощеною і ігнорує наявність транзакцій, оскільки вони не так важливі в даному випадку. Основна мета цієї моделі - визначити, чи буде мережа, заснована на блокчейні, з таким видом обмеження частки, забезпечувати достатню продуктивність, щоб заповнювати всі блоки вчасно.

Модель складається з агентів, частину з яких були додано до мережі під час її створення, інші ж агенти приєдналися до мережі після створення кількох блоків. При кожному створенні блоку один агент (валідатор) повністю втрачає свою частку після успішного створення блоку. У випадку розгалуження – нечисленні агенти можуть втратити ставку (шляхом скидання часу ставки), але коли розгалуження вирішено, а інший альтернативний ланцюжок відкинутий, агенти які працювали на альтернативний ланцюжок отримають свої ставки назад. Агент також може втратити ставку, якщо вона стає більшою, ніж модуль, що обмежує максимальне значення.

Перший результат моделювання наведено на рисунку 2.13. Тут ми вибрали обмежений час у 500 секунд та часом блоку у 20 секунд для швидшого моделювання (значення в реальній системі залежатимуть від кількості даних та кількості вузлів). У цьому випадку граничне значення достатньо низьке, тому деякі вузли не мають можливості використовувати свою ставку до досягнення обмежувального модуля.

Як ми бачимо на графіку, низька межа ставки може спричинити вищий діапазон середніх коливань ставки, і розподіл показує, що значення розподіляються не рівномірно, але на всіх діапазонах значень ставки наявні, що свідчить про достатньо рівномірний процес генерування. Це означає, що система все ще буде придатною для використання, навіть якщо велика кількість вузлів пропускатиме свої ставки через досягнення межі модуля.

При наступному запуску моделі ліміт модуля було збільшено до 1000 секунд, всі інші параметри залишились незмінними. Результат другого моделювання наведено на рисунку 2.14.



(a) (b)

Рисунок 2.13 – Симуляція стейкінгу з низькими обмеженнями *a* - Розподіл накопичених значень, *b* - середнє накопичене значення

В даному випадку видно, що коливання середнього значення ставки набагато нижчі. Розподіл у цьому експерименті показує, що значення ставки в мережі розподіляються майже рівномірно, а це означає, що в будь-який

період часу є вузли, у яких є достатня ставка для створення нового блоку. Також можна побачити, що вузли не досягають межі ставки, оскільки вся їхня ставка використовується при створенні блоків.

(a) (b)

Рисунок 2.14 – Симуляція стейкінгу з високими значеннями обмеження *a* - Розподіл накопичених значень, *b* - середнє накопичене значення

Отже, обмеження максимального значення ставки модулем дає нам хороший розподіл ставок, який підходить для використання в IDS на основі блокчейну. Навіть якщо ліміт низький і багато вузлів скидають свої ставки, досягнувши ліміту, система все одно матиме достатньо ресурсів для створення та перевірки блоків.

Запропонований для поставленої мети протокол консенсусу, – модифікація протоколу PoS, де замість накопичення певного об’єму криптовалюти використовується “накопичення” часу. Такий протокол консенсусу є значно ефективніший в плані використання обчислювальних ресурсів [84]. Він дозволяє підтримувати роботу розподіленої мережевої системи виявлення вторгнень, не створюючи при цьому значних навантажень на обчислювальні ресурси системи, у порівнянні з POW протоколом [124].

Але не зважаючи на те, що протокол консенсусу достатньо невибагливий до обчислювальних ресурсів, в описаному алгоритмі створення блоків все ж можна помітити неоптимальне використання обчислювальних ресурсів (так само як і в класичному варіанті протоколу Proof-of-Stake [125]). Ця не оптимальність полягає у тому, що вузли, які претендують на створення нових блоків виконують спроби створити новий блок увесь час, не зважаючи на те, що ймовірність створення нового блоку при певних умовах для конкретного вузла може бути достатньо малою. А, отже, така витрата обчислювальних ресурсів у нашому випадку не має сенсу (оскільки в нашому випадку вузли не конкурують за нагороду за створення блоку, тоді як у контексті криптовалют за протоколом Proof-of-Stake, такі спроби мають певний сенс, оскільки рано чи пізно спроби призведуть до створення нового блоку, а отже і отримання нагороди за блок [122]). Оскільки, в нашому випадку, вузли не отримують нагороди за створення блоків, а оптимальне використання ресурсів для системи виявлення вторгнень є важливим, можна зменшити навантаження на центральний процесор за рахунок пропуску спроб створення нового блоку, у випадку, коли ймовірність його створення дуже низька.

Для оцінки вірогідності вдалого створення блоку, спочатку впевнимося, що ланцюжок хеш функцій дає нам рівномірний розподіл, для цього для 8000 випадкових наборів даних порахуємо значення *sha256(sha256(data))* та побудуємо графік розподілу, щоб впевнитися, що результат близький до рівномірного розподілу. Як бачимо на графіку (рисунок 2.15), результати подвійної функції sha256 на випадкових даних розподілені достатньо рівномірно, отже, для оцінки вірогідності створення блоку будемо вважати. що розподіл рівномірний. У випадку рівномірного розподілу, математичне сподівання складає (7). При цьому імовірність появи будь-якого із значень однакова і дорівнює (8),

$$M(X)= (n+1)/2 \qquad (15)$$

$$P_k = P(_{X=k})=1/n \quad (16)$$

де n кількість значень у множині X.

Виходячи з умови рівномірності розподілу можемо оцінювати ймовірність створення блоку при поточній складності як відношення складності до максимального значення результату хеш-функції. Sha256 видає результат розміром 256 біт, а отже максимальне числове представлення результату 2^{256}.

Спираючись на запропонований алгоритм розрахунку складності здобування блоку (залежно від того, скільки часу вузол працював і не видобував нових блоків, та сумарної швидкості видобування блоків в мережі) проведемо оцінку вірогідності створення нового блоку усього діапазону можливих значень складності (рисунок 2.16).

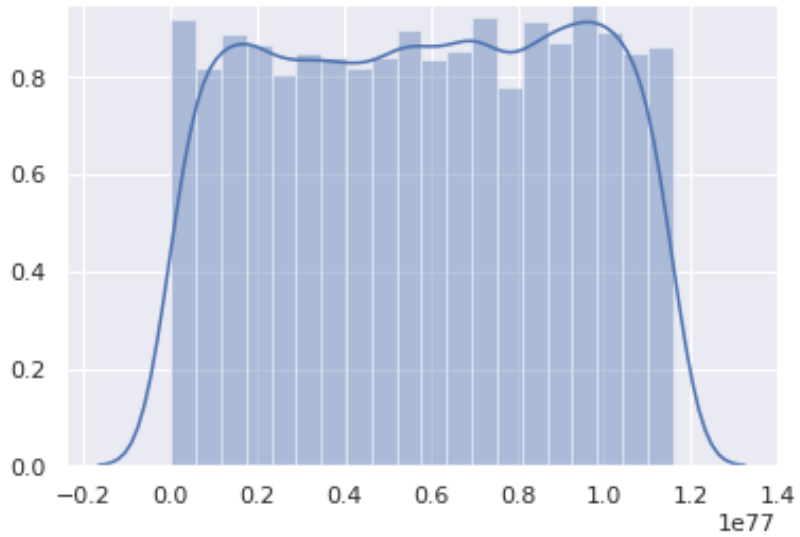


Рисунок 2.15 – Розподіл подвійної sha256 на випадкових даних

Проаналізувавши графік бачимо, що при пороговому значенні складності меншому за 2^{245}, ймовірність створення нового блоку близька до нуля, це означає, що навіть дуже велика кількість спроб, скоріш за все, не призведе до створення нового блоку. А отже, спроби створити блок при таких

значення складності це частіше за все марна витрата обчислювальних ресурсів та енергії.

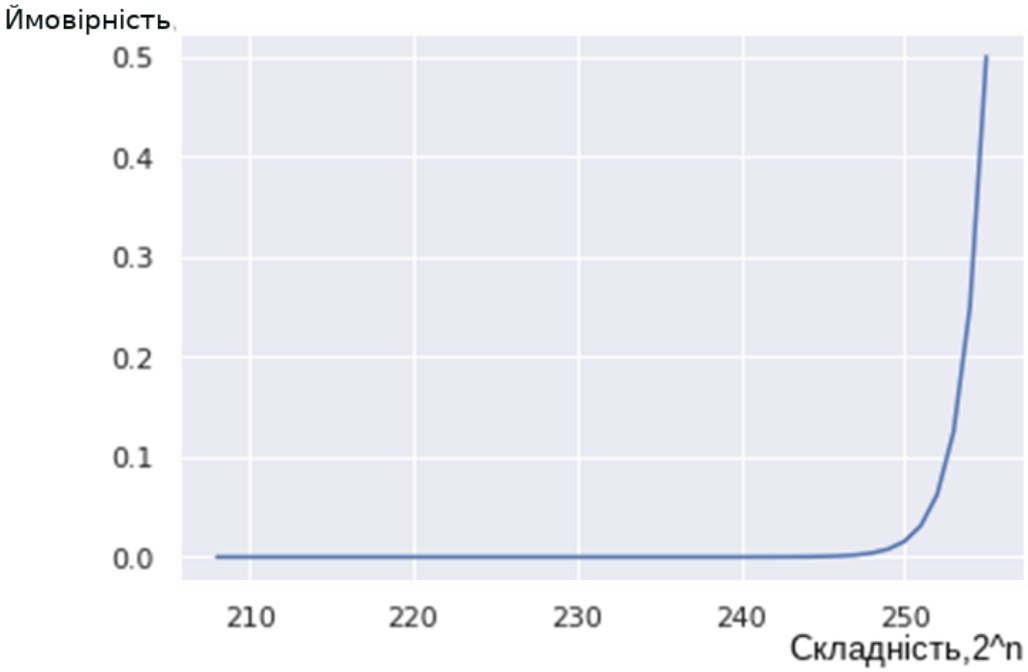


Рисунок 2.16 – Залежність вірогідності створення нового блоку від поточного значення складності (складність задана у вигляді степеня числа 2).

Виходячи з того, що діапазон ефективного створення нових блоків займає близько п’ятої частини загального діапазону значень складності, можна вважати, що така оптимізація дозволить зекономити близько 80% процесорного часу, що особливо важливо для малопотужних вузлів, тоді як для більш потужних вузлів цей діапазон в теорії можна розширити, що дасть можливість частіше створювати нові блоки за рахунок менш ефективного використання обчислювальних ресурсів.

Висновки до розділу 2

В результаті моделювання системи виявлення та аналізу аномальних подій можна зробити наступні висновки:

1) Запропонована концептуальна модель розподіленої інформаційної системи виявлення та аналізу аномальних подій на основі blockchain технології, відповідає сформованим раніше вимогам і дозволяє спроектувати систему для захисту комп'ютерних мереж малих та середніх підприємств.
2) Запропонований метод вибору протоколу консенсусу для розподіленої інформаційної системи виявлення та аналізу аномальних подій на основі blockchain технології, що враховує вимоги до обладнання, масштабування та керування учасниками систем та забезпечує підтримку прийняття рішень при проектуванні розподілених систем виявлення вторгнень.
3) Адаптований для використання в розподіленій системі виявлення вторгнень на основі blockchain протокол консенсусу на основі PoS, забезпечує достатню швидкодію blockchain компонента для оперативного обміну даними між вузами розподіленої системи.
4) Запропонований метод підвищення ефективності використання ресурсів блокчейн підсистемою, заснований на розрахунку вірогідності успішного створення блоку при заданій складності дозволяє знизити енерговитрати на підтримку роботи blockchain, а також звільнити додаткові ресурси для модуля виявлення вторгнень та підвищити швидкодію blockchain підсистеми.
5) Запропонований комплексний класифікатор, для колаборативного виявлення кіберзагроз та аномального трафіку, що комбінує результати окремих класифікаторів, де в якості класифікаторів пропонується використовувати як класичні регресійні класифікатори, так і класифікатор на основі штучної імунної системи, дозволяє підвищити точність виявлення невідомих аномальних подій для розподіленої системи виявлення вторгнень.

Результати досліджень приведених в розділі опубліковані в роботах [78, 84, 109, 123, 124]

РОЗДІЛ 3
ФУНКЦІОНАЛЬНА МОДЕЛЬ ТА АРХІТЕКТУРА РОЗПОДІЛЕНОЇ СИСТЕМИ ЗАХИСТУ КОМП'ЮТЕРНИХ МЕРЕЖ НА ОСНОВІ BLOCKCHAIN ТЕХНОЛОГІЇ

3.1 Загальна функціональна модель

Функціональна модель процесу «Захист мережі від вторгнень» верхнього рівня наведена на рисунку 3.1.

Рисунок 3.1 – Функціональна модель процесу верхнього рівня «Захист мережі від вторгнень»

Вхідною інформацією для системи є необроблений трафік, базовий набір правил та налаштування локальної мережі організації. Цей набір даних є обов’язковим для коректної роботи розподіленої системи Захисту комп’ютерних мереж. Керуючими елементами є нормативна документація, міжнародні стандарти в ІБ галузі, потреби організації, мережеві протоколи, технічні характеристики вузла захисту та час. В ролі механізмів керування виступає системний адміністратор мережі відповідальний за інформаційну

безпеку мережі (що характерно для невеликих мереж), дисковий простір, який використовується вузлом системи захисту комп’ютерних мереж для зберігання правил та тимчасової інформації, мережа в рамках якої вузол забезпечує захист, та яка приєднана до глобальної мережі або мережі вищого рівня, мережевий шлюз, база відкритих правил для виявлення вторгнень (наприклад, правила спільноти SNORT), протокол консенсусу для блокчейн підсистеми, а також класифікатор призначений для виявлення шкідливих подій в мережі. На виході з системи отримуємо відфільтрований трафік, який направляється в мережу захищену вузлом системи, дані про аномальні та шкідливі події, створені правила для виявлення нових атак та вторгнень, блокчейн базу, яка містить перераховану вище інформацію та дозволяє забезпечити її цілісність, також на виході ми отримуємо звіти щодо усіх етапів роботи вузла системи захисту комп’ютерних мереж.

3.1.1 Вхідна інформація

Оскільки система виявлення та попередження вторгнень працює за принципом мережевого фільтра, основними вхідними даними для системи є *необроблений інтернет трафік*, який приходить в мережу, що знаходиться під захистом модуля виявленні попередження вторгнень. В процесі роботи він використовується для оцінки подій, а також проходить процес фільтрації.

Базовий набір правил слугує основою для виявлення вторгнень в мережу, і покриває основні відомі атаки та шкідливі події, основна задача цього набору правил забезпечити швидке виявлення популярних атак. Задається системним адміністратором в процесі налаштування системи захисту комп’ютерних мереж і окрім правил з існуючих баз може включати правила написані адміністратором власноруч. Окрім того стандартний набір правил зазвичай повинен бути скоригованим у відповідності до того, який трафік буде актуальним для конкретної мережі.

Налаштування мережі мають прямий вплив на процес виявлення вторгнень та шкідливих подій, оскільки правила можна застосовувати для фільтрації трафіка лише у випадку правильного налаштування як самих правил так і мережі.

3.1.2 Керуючі елементи

Нормативна документація законодавча база щодо питань інформаційної безпеки України та стандарти побудови блокчейн систем.

В основу покладені ЗУ «Про інформацію» від 02.10.1992 №2657-XII [126], «Про основні засади забезпечення кібербезпеки України» [1], «Про захист інформації в інформаційно-телекомунікаційних системах» [127], а також нормативні документи, постанови Кабінету Міністрів України, та ін.

Міжнародні стандарти ISO 22739:2020, ISO/TR 3242:2022, ISO 23257:2022, ISO/TS 23635:2022 слугують джерелом рекомендацій щодо побудови захищених розподілених застосунків на основі блокчейн технології.

Міжнародні стандарти в ІБ галузі – включають стандарти побудови безпечної архітектури мережі, виявлення вторгнень в мережах та забезпечення захищеного обміну даними між вузлами розподіленої системи захисту комп’ютерних мереж.

Потреби організації – включають особливості налаштувань та роботи мережі конкретної організації, які впливають на налаштування системи виявлення та необхідний рівень безпеки.

Час є основою для роботи блокчейн підсистеми, оскільки всі елементи ланцюжка блоків мають містити часову мітку, окрім того протокол консенсусу та механізм виявлення вторгнень спираються на часові обмеження для верифікації блоків, фільтрації трафіка та створення правил для виявлення вторгнень.

Мережеві протоколи в першу чергу визначають вимоги до фільтрації трафіка, оскільки при відсутності обмежень на дозволені протоколи забезпечити ефективну фільтрацію трафіка достатньо складно. Також мережеві протоколи використовуються блокчейн підсистемою для обміну інформацією з іншими вузлами розподіленої блокчейн системи.

Технічні характеристики вузла системи захисту та підконтрольної мережі впливають на доступні ресурси для виявлення вторгнень, аналізу і класифікації даних для створення нових правил та створення нових блоків для Blockchain. У випадку дефіциту ресурсів пріоритетною задачею повинна бути задача виявлення вторгнень та шкідливих подій в мережі.

3.1.3 Елементи і механізми виконання

Системний адміністратор – має глибокий рівень знань щодо структури та роботи мережі а також практичний досвід її налаштування, що дає змогу адміністратору правильно налаштувати вузол системи захисту комп’ютерних мереж, спираючись на потреби організації, власні знання та досвід, включаючи вибір базового набору правил, створення власних правил для обраного набору мережевих протоколів, та враховуючи архітектуру конкретного сегмента мережі та технічні характеристики доступного обладнання.

Дисковий простір – використовується вузлами системи захисту комп’ютерних мереж для зберігання актуального набору правил, переліку шкідливих та підозрілих подій, ланцюжка блоків блокчейн бази, а також звітів щодо роботи вузлів системи. Для роботи повноцінного вузла блокчейн дисковий простір повинен дозволяти розмістити цілісний ланцюжок блоків.

Мережа з одного боку є середовищем для передачі трафіку, який піддається подальшій фільтрації, а отже пропускна здатність мережі напряму пов’язана із потужність вузла, що виконуватиме фільтрацію. Крім того архітектура мережі впливає на маршрутизацію відфільтрованого трафіка.

Також мережа використовується розподіленою системою захисту від вторгнень для обміну даними між вузлами, в даному випадку блокчейн підсистема передає через мережу транзакції та блоки таким чином забезпечуючи синхронізацію даних між вузлами.

Мережевий шлюз є апаратним засобом, який відділяє локальну мережу організації від глобальної мережі, вузол розподіленої системи захисту комп'ютерних мереж розміщується на мережевому шлюзі, завдяки чому полегшується задача фільтрації трафіка (так як увесь трафік, що приходить ззовні або направлений назовні буде проходити через шлюз.

Відкриті правила – набір правил, наприклад правила спільноти, які включаються в базовий набір правил при налаштування вузла системи захисту від вторгнень. Зазвичай відкриті правила включають в себе набір для виявлення найбільш поширених атак та шкідливих подій.

Протокол консенсусу прописує алгоритми створення та верифікації блоків у блокчейні та забезпечує його стабільну роботу. Одна із задач, що покладається на протокол консенсусу це забезпечення розподіленого генерування блоків усіма вузлами блокчейн підсистеми, що робить систему стабільною та захищеною від підробки блоків.

*Класифікато*р виконує задачу розпізнавання шкідливих подій, на основі поточної інформації про трафік а також спираючись на попередньо-виявлені підозрілі події, результати класифікації можуть використовуватись в подальшому для створення нових правил. Для класифікації можуть використовуватися різноманітні алгоритми, починаючи від регресій, закінчуючи штучними нейронними мережами.

3.1.4 Вихідна інформація

Основною вихідною інформацією *є відфільтрований трафік*, який потрапляє в сегмент мережі, який знаходиться під захистом вузла розподіленої системи захисту комп'ютерних мереж, що пройшов процес

фільтрації, спираючись на базовий набір правил, а також правила створені в процесі роботи системи захисту комп’ютерних мереж.

На виході вузол розподіленої системи захисту від вторгнень надає *блокчейн базу*, яка містить усю необхідну, підготовлену для передачі іншим вузлам інформацію, збережену у вигляді ланцюжка блоків, які в свою чергу містять набір транзакцій, а також за необхідності сторонні дані, які зберігаються за межами блоків, але криптографічно з ними пов’язані.

Створені в процесі роботи системи захисту комп’ютерних мереж *правила* також є вихідною інформацією системи. Набір створених правил направляється як в блокчейн базу так і в окремий набір файлів доступний для перегляду адміністратором, після перегляду адміністратор може прийняти рішення щодо подальшого використання тимчасово створених правил в якості правил базового набору.

Дані про аномальні події також достатньо важливі для роботи розподіленої системи захисту комп’ютерних мереж, оскільки спираючись на них, адміністратор може вручну створити необхідні правила. Процес автоматичного створення правил класифікатором також передбачає використання інформації про підозрілі події, при чому використання інформації з різних вузлів дозволяє виявити вторгнення раніше, аніж окремий вузол міг би зробити це самостійно.

Звіти, що створюються вузлом розподіленої системи захисту комп’ютерних мереж містять узагальнену інформацію щодо процесу роботи як кожного окремого вузла, так і короткий звіт щодо роботи системи в цілому. Спираючись на звіти адміністратор може прийняти рішення щодо правильності налаштувань та коректності роботи системи, і за необхідності надіслати розробникам відгук щодо виявлених проблем.

3.1.5 Декомпозиція функціональної моделі

Виконавши декомпозицію функціональної моделі бізнес процесу захист мережі від вторгнень отримаємо модель процесу другого рівня зображену на рисунку 3.2.

В результаті декомпозиції основної задачі захист від вторгнень отримуємо наступний перелік робіт:

1. Налаштування системи;
2. Фільтрація трафіка;
3. Виявлення аномальних подій;
4. Поширення через блокчейн;
5. Генерація правил;
6. Генерація звітів.

Перш ніж почати використання розподіленої системи захисту комп’ютерних мереж на кожному окремому сегменті мережі підприємства, системний адміністратор проводить попереднє налаштування кожного окремого вузла розподіленої системи захисту комп’ютерних мереж. За необхідності, спираючись на результати роботи системи в подальшому налаштування можуть бути виконані повторно, або скориговані вже існуючі.

Спираючись на потреби організації, набір використовуваних мережевих протоколів та нормативну документацію системний адміністратор підбирає базовий набір правил (можна залишити і набір правил за замовчуванням, які будуть відповідати найбільш поширеним сценаріям використання мережі організацією), які будуть використовуватися для попередньої фільтрації трафіка в процесі роботи системи захисту комп’ютерних мереж.

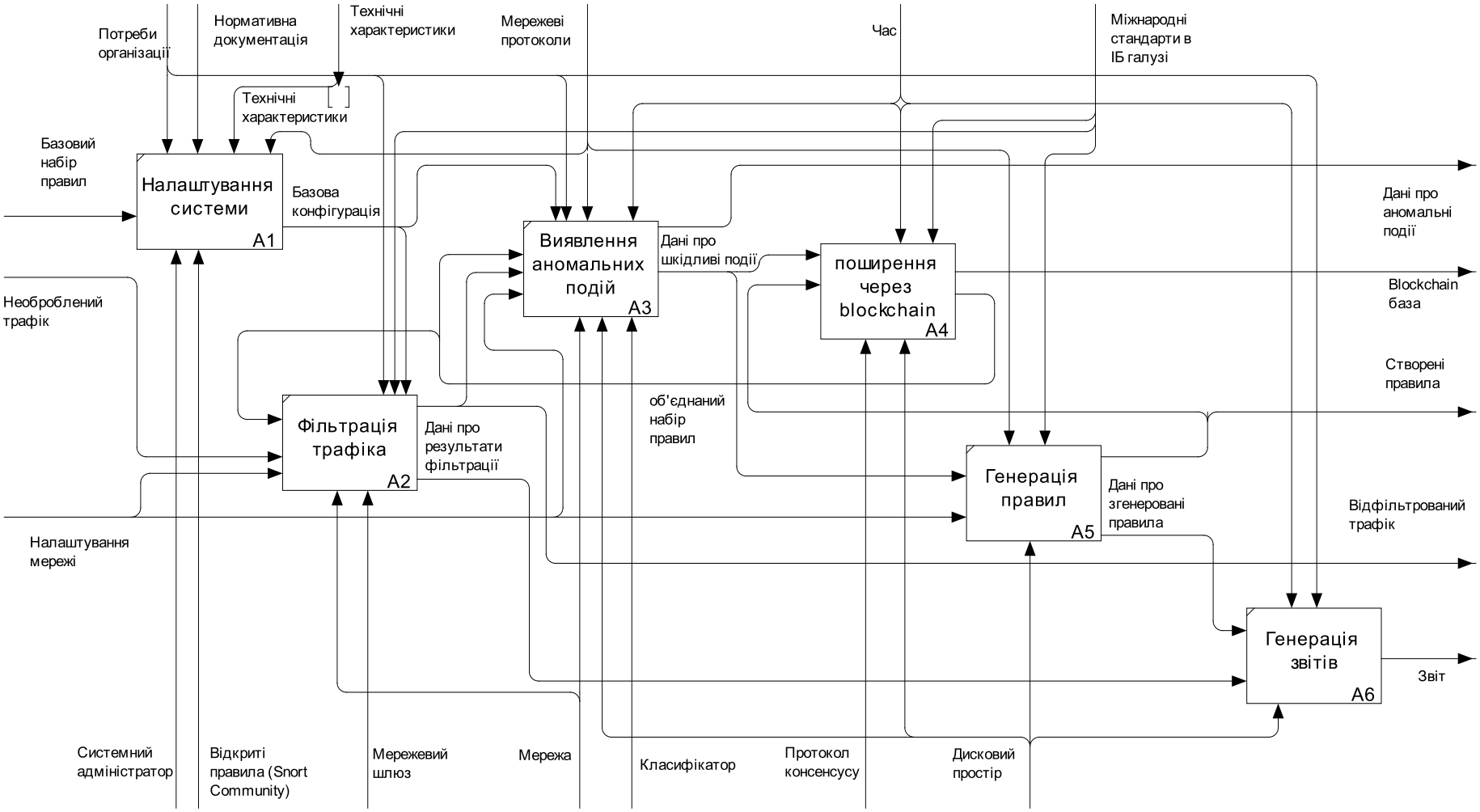


Рисунок 3.2 – Функціональна модель процесу другого рівня “Захист мережі від вторгнень”

Також виходячи з технічних характеристик обладнання, яке використовується в мережі для розміщення вузла системи захисту комп’ютерних мереж, адміністратор може налаштувати режим роботи вузла, обравши коректні пріоритети для процесів та обмеження на використанні ресурсів, забезпечивши оптимальну для організації роботу вузла розподіленої системи захисту комп’ютерних мереж.

Відповідно до отриманих налаштувань вузол системи виконує *фільтрацію трафіка*, використовуючи сигнатурні та статистичні методи, основа для яких прописана в правилах заданих при налаштуванні та створених в процесі роботи системи. Відфільтрований трафік відправляється в мережу з призначенням, також, відфільтрований трафік та інформація про відфільтровані пакети, отримана на етапі фільтрації направляється на виявлення підозрілих подій.

Виявлення аномальних подій відбувається на основі відфільтрованого трафіка, а також інформації про фільтрацію за допомогою різноманітних класифікаторів. В першу чергу це регресійний класифікатор, який завдяки простому алгоритму не вимагає багато ресурсів для виявлення підозрілих та шкідливих подій. Також можливе використання класифікаторів на основі нейронних мереж, які можуть значно підвищити точність виявлення, за рахунок збільшення використання ресурсів.

Дані про шкідливі події використовуються для *генерації правил* за шаблонами, завдяки чому система зможе блокувати такі події, не залучаючи до їх виявлення класифікатори. Автоматично створені правила застосовуються не перманентно, а з часовими обмеженнями, оскільки такі блокування можуть в подальшому істотно обмежити доступ користувачів до сервісів.

Дані про шкідливі події а також згенеровані правила направляються на блокчейн підсистему і завдяки блокчейн базі поширюються між вузлами розподіленої системи захисту комп’ютерних мереж. Також за допомогою

блокчейн підсистеми вузол може отримувати дані від інших вузлів, які так само поширюють отримані ними правила та результати виявлення підозрілих подій.

Дані про відфільтрований трафік та дані про створення правил фільтрації використовуються для генерації звітів, що дає змогу системному адміністратору в подальшому оцінити ефективність роботи вузла розподіленої системи захисту комп'ютерних мереж для свого сегменту мережі. Також користуючись звітами можна перевірити автоматично створені правила фільтрації і перетворити їх із тимчасових в постійні.

Розглянемо детальніше етап поширення інформації за допомогою блокчейн. Функціональна модель цього процесу зображена на рисунку 3.3.

Першим етапом при записі будь-якої інформації в блокчейн в першу чергу необхідно створити транзакцію, яка містить дані які треба записати та час створення транзакції.

Транзакції є складовими для створення блоків, але перш ніж передати транзакцію на етап створення блока, блокчейн підсистема відправляє транзакцію в тимчасовий пул, який синхронізується з усіма вузлами блокчейн підсистеми, завдяки чому транзакція поширюється по блокчейн мережі значно раніше аніж буде створено блок, який містить таку транзакцію. Процес поширення транзакцій в блокчейн мережі релізовується роботами передача транзакцій та приймання транзакцій, які включають в себе роботу з тимчасовим пулом.

Наступним етапом іде створення блоків, в які записуються транзакції взяті з тимчасового пулу, тобто кожен вузол блокчейн мережі може використовувати при створенні блоків транзакції інших вузлів. Кожен вузол намагається створювати блоки самостійно, через це, в певні моменти часу в блокчейн мережі може утворитися кілька конкуруючих ланцюжків, але, кожен вузол періодично сканує активні ланцюжки, і обирає для додавання своїх блоків той, що швидше росте.

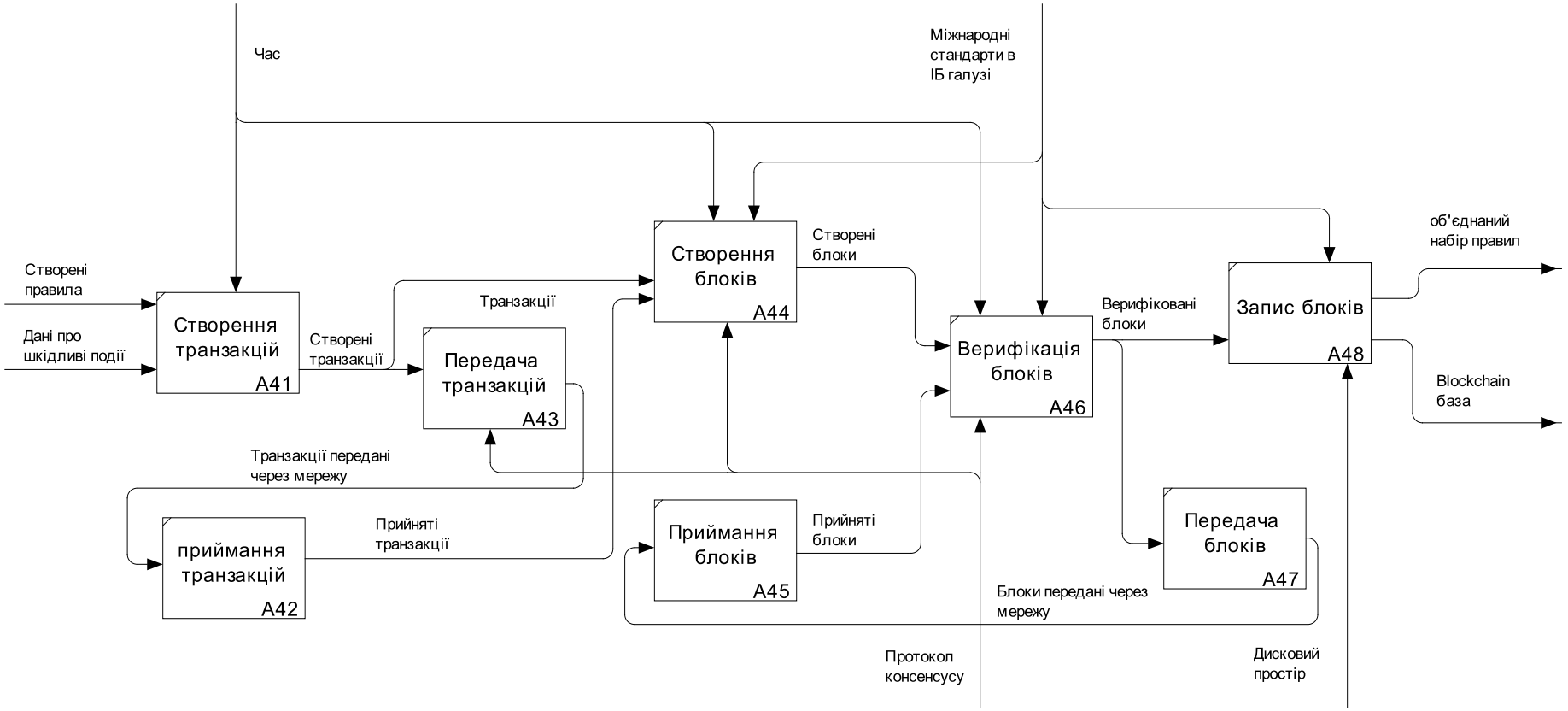


Рисунок 3.3 – Функціональна модель процесу третього рівня “Поширення через blockchain”

Після створення блок проходить стадію верифікації, на якій перевіряється відповідність блока умовам протокола консенсусу. Зазвичай для створення блоку, який відповідатиме умовам протоколу консенсусу і пройде верифікацію вузол витрачає достатньо багато часу і спроб (особливо при використанні протоколу PoW). Циклічний процес створення блока з його наступною верифікацією називається майнінгом. Але оскільки у нашому випадку використовується протокол консенсусу на основі PoS використання ресурсів значно менше у порівнянні з PoW. Для верифікації використовуються стандартні алгоритми хешування, в нашому випадку Sha256.

Створений і верифікований блок записується в локальний ланцюжок блоків, а також поширюється серед усіх вузлів блокчейн мережі (передача та приймання блоків), таким чином кожен вузол, який працював над ланцюжком, до якого був доданий новий блок, переходить до створення наступного блоку. Розгалуження ж в ланцюжку зазвичай виникають, коли два або більше вузлів створили блок практично одночасно, тоді швидкість росту кожної з гілок ланцюжка залежатиме від того, скільки вузлів підтримають кожну з гілок.

Локальна копія ланцюжка блоків записується на диск, і оскільки кожний блок криптографічно поєднаний з попереднім, для можливості повної верифікації ланцюжка необхідно зберігати його від самого початку, тому з часом ланцюжок вимагатиме достатньо багато дискового простору.

3.2 Варіанти використання системи

Перш ніж починати розробку системи слід виділити основні її функціональні можливості.

Основними функціональними вимогами до системи є:

- збір зберігання обробка та поширення інформації щодо аномальних подій та кібератак для формування інформації про оточення мереж, яка може бути використана в подальшому для генерування нових правил;
- створення, зберігання, оновлення та поширення набору основних та тимчасових правил для вузлів розподіленої системи захисту комп’ютерних мереж;
- формування та виведення звітів щодо виявлених аномальних подій та створених правил;
- синхронізація роботи з сусідніми вузлами розподіленої системи, підтримка цілісності розподіленої системи.

Спираючись на бізнес-логіку процесу, детально представлену в попередньому підрозділі, Operation Capabilities Breakdown діаграма, зображена на рисунку 3.4, відображає основні варіанти використання системи акторами. Основним актором для системи є системний адміністратор, відповідальний за роботу локальної мережі.

Як бачимо основними варіантами використання системи адміністратором є:

- налаштування системи;
- перегляд актуальних наборів правил;
- перегляд звітів по роботі системи.

Для кінцевих користувачів мережі система працює прозоро і не вимагає додаткових взаємодій.

Детально розглянувши варіант використання «Налаштування системи» ми бачимо, що усі налаштування діляться на дві категорії - налаштування при встановленні та налаштування в процесі роботи.

До налаштувань при встановленні належать такі налаштування, які зазвичай виконуються один раз, і не вимагають частої зміни або оновлення, до них зокрема належать базові налаштування параметрів мережі,

налаштування основних сервісів використовуваних в мережі, а також налаштування базового набору правил.

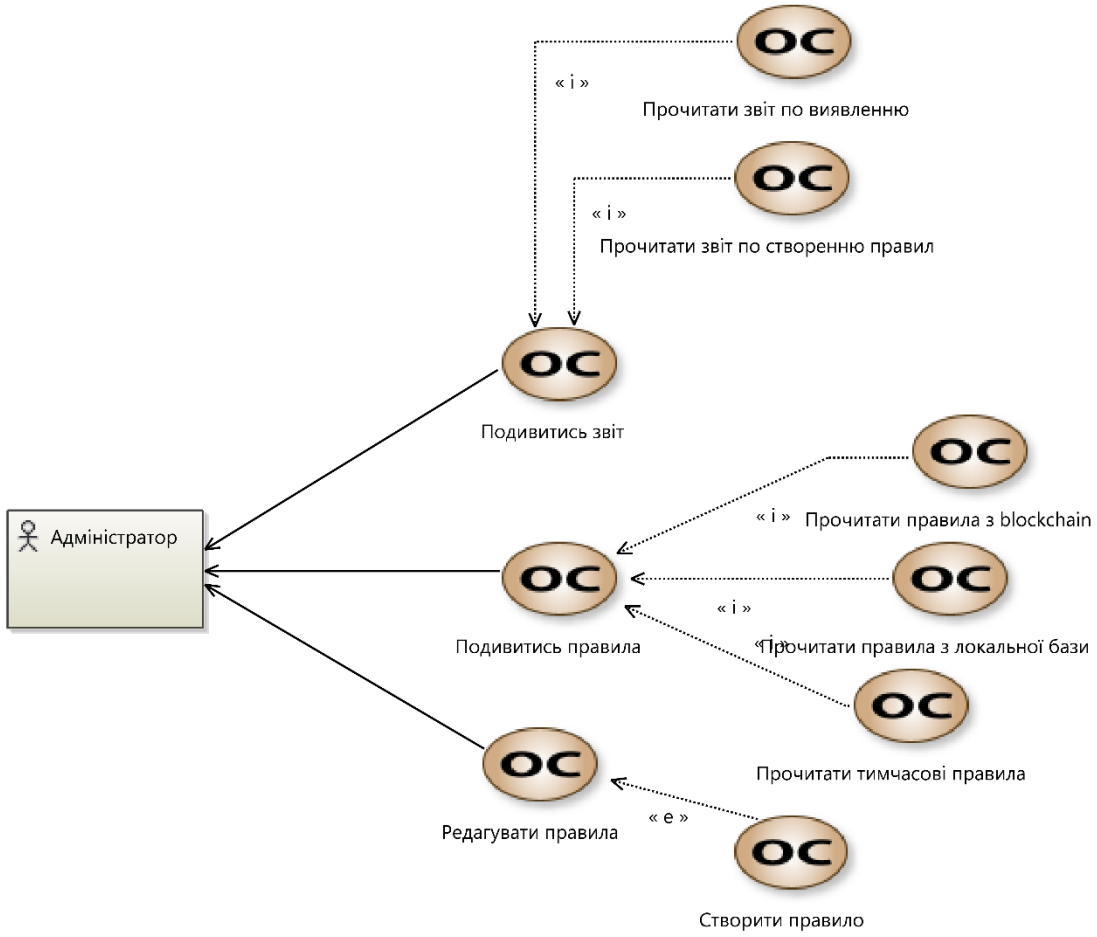


Рисунок 3.4 – Діаграма основних варіантів використання розподіленої системи захисту комп’ютерних мереж

Налаштування ж в процесі роботи – це такі налаштування, які зазвичай коригуються адміністратором в процесі роботи системи в залежності від особливостей роботи мережі та даних щодо виявлених вторгнень. До таких налаштувань можна віднести налаштування кількості активних комп’ютерів, використовувані мережеві протоколи, а також чутливість алгоритмів виявлення вторгнень.

Перегляд актуальних наборів правил включає в себе перегляд правил з початкового набору, перегляд правил з blockchain – це правила, які придатні до постійного використання, записані в ланцюжку blockchain. Правила з тимчасової бази, на відміну від правил записаних в ланцюжку розраховані на зберігання та використання протягом обмеженого проміжку часу (періоду актуальності правила), і призначені для зберігання інформації про тимчасові блокування.

3.3 Взаємодія модулів інформаційної системи

Основні активності виконувані кожним окремим вузлом розподіленої системи захисту комп'ютерних мереж відображені на OAIB діаграмі(рисунок 3.5).

Першою активністю в ланцюжку є «Отримання трафіку» і реалізовується модулем захоплення трафіку на мережевому шлюзі. В якості альтернативного варіанту трафік з основного інтерфейсу може дублюватися на модуль захоплення, у випадку, якщо мережевий шлюз не має можливості виконувати повноцінну обробку трафіка, але в такому випадку повноцінне блокування вторгнень буде також неможливим.

Наступні активності – «попередня фільтрація» та «фільтрація» виконуються мережевим екраном та модулем виявлення вторгнень на основі наявного набору правил і на виході передають відфільтрований трафік у захищений сегмент локальної мережі. Окрім цього, результатами фільтрації є підозрілі події, які в процесі роботи поширюються за допомогою блокчейна між усіма вузлами розподіленої системи захисту комп'ютерних мереж.

Отримавши аномальну подію від модуля виявлення вторгнень або через блокчейн, вузол виконує класифікацію події, користуючись зібраними даними про події, які вдалося відловити. Спираючись на результати класифікації, відбувається створення правил, що дозволяють швидко

відловлювати подібні шкідливі події. Правила застосовуються до модуля виявлення вторгнень та мережевого екрану.

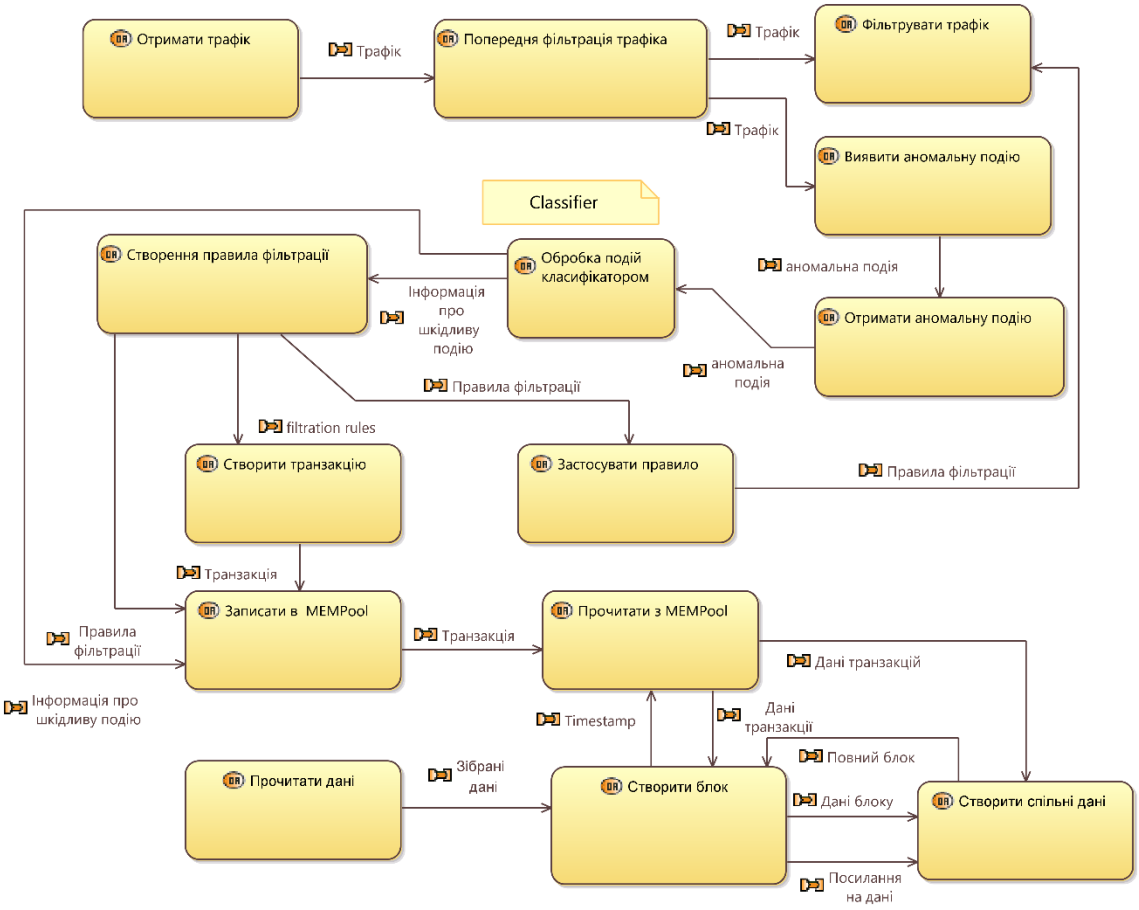


Рисунок 3.5 – Operation Activities Interaction Breakdown діаграма для розподіленої системи захисту комп’ютерних мереж на основі blockchain

Для поширення ж правил та інформації про підозрілі події створюються блокчейн транзакції, що направляються до тимчасового пулу, з якого будь-який вузол може їх прочитати, і використовувати для створення блоків.

Блоки на відміну від тимчасового пулу зберігають транзакції на постійній основі, тоді як тимчасовий пул зберігає транзакції лише до їх остаточного запису в блок.

Дані ж прочитані вузлом розподіленої системи захисту комп’ютерних мереж використовуються в залежності від їх типу в процесі роботи. Дані транзакцій з тимчасового пулу також можуть застосовуватися, але транзакція з тимчасового пулу не має гарантованого підтвердження незмінності.

Взаємодія основних компонентів системи при генерації правил для виявлення вторгнень наведено на діаграмі послідовності, рисунок 3.6.

Як бачимо, події отримані з блокчейн або компонента виявлення вторгнень збираються та обробляються аналітичним компонентом, який робить висновок щодо того, була дана подія шкідливою чи ні. У випадку шкідливої події компонент генерації правил створює узагальнене правило для блокування подібних спроб втручання, після цього правило у вигляді транзакції, через блокчейн, поширюється на інші вузли розподіленої системи виявлення вторгнень.

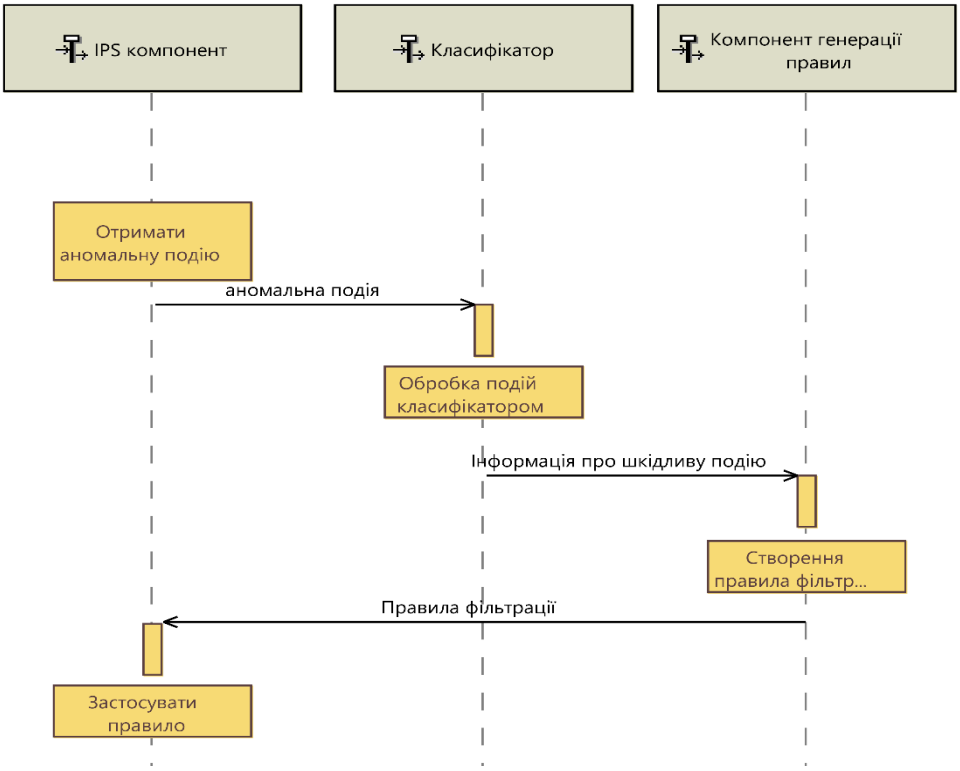


Рисунок 3.6 – Діаграма послідовностей для процесу автоматичної генерації правил

Процес обробки підозрілих подій з врахуванням створення blockchain транзакцій відображено на рисунку 3.7.

Як бачимо, інформація про події в будь-якому випадку додається до blockchain оскільки в подальшому вона може знадобитися для обробки подій аналітичним компонентом, при цьому транзакції про події зазвичай зберігаються на в самому ланцюжку блоків, а в тимчасових даних, оскільки актуальність інформації про виявлені події достатньо обмежена в часі, а оскільки обробка великої кількості інформації займає багато часу і використання застарілої інформації не підвищує ймовірність виявлення актуальних вторгнень розподіленою системою, такі дані можна видалити для зменшення ресурсного навантаження на вузли системи виявлення.

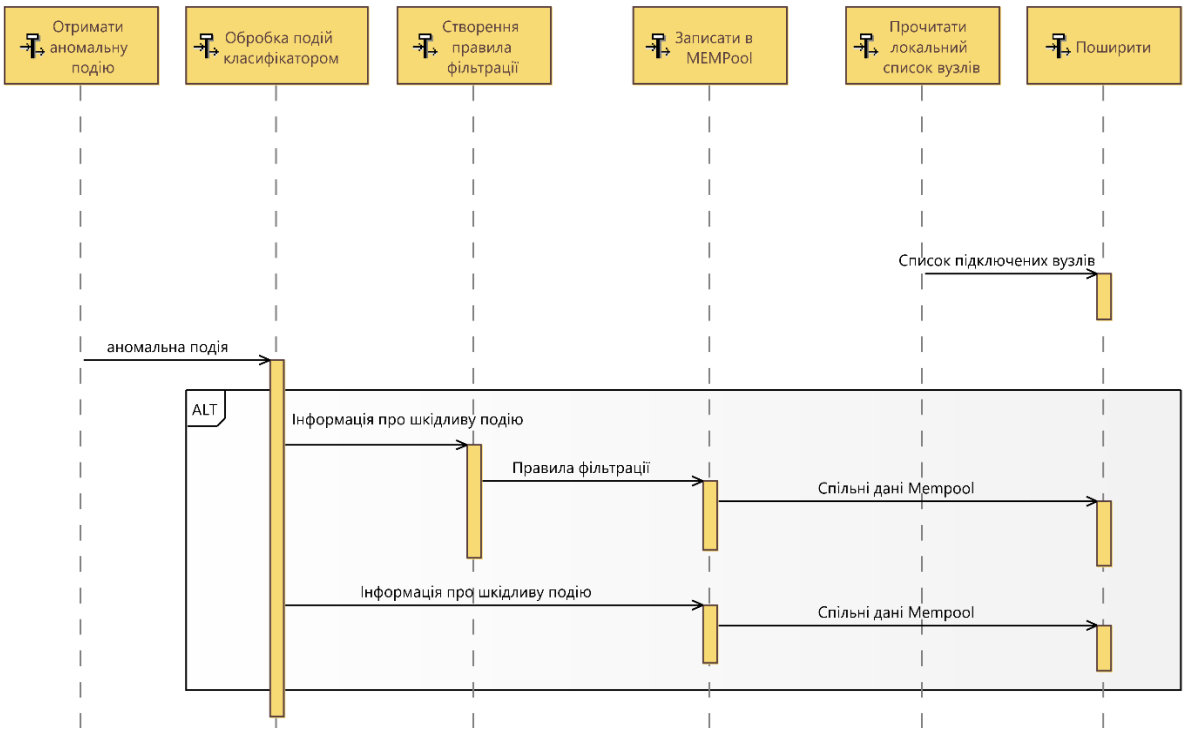


Рисунок 3.7 – Діаграма послідовності поширення події через блокчейн

Отже, після створення об’єктів подій та правил фільтрації вони додаються до тимчасового пулу у вигляді транзакцій і далі використовуються для створення блоків і розповсюджуються на вузли системи захисту комп’ютерних мереж із локального списку вузлів, який містить список вузлів, з якими поточний вузол має встановлені з’єднання. Кожен вузол може обмінюватися з іншими локальним списком вузлів, тому кожен вузол має можливість встановити максимальну кількістю оптимальних з’єднань, завдяки чому усі блокчейн вузли об’єднуються в мережу з достатньо складною топологією.

Наступним етапом після створення транзакцій є їх запис у блоки. Процес створення та запису блоків відображений на наступній діаграмі послідовності (Рисунок 3.8).

Спочатку транзакції зчитуються з тимчасового пулу, і дані з транзакцій записуються або в тіло блоку, або в додаткові тимчасові розповсюджувані дані, які будуть поширюватися разом з блоками між вузлами системи захисту комп’ютерних мереж. Далі відбувається перевірка блока на відповідність протоколу консенсусу та верифікація, після цього блок готовий до остаточного запису в ланцюжок та розповсюдження на інші вузли.

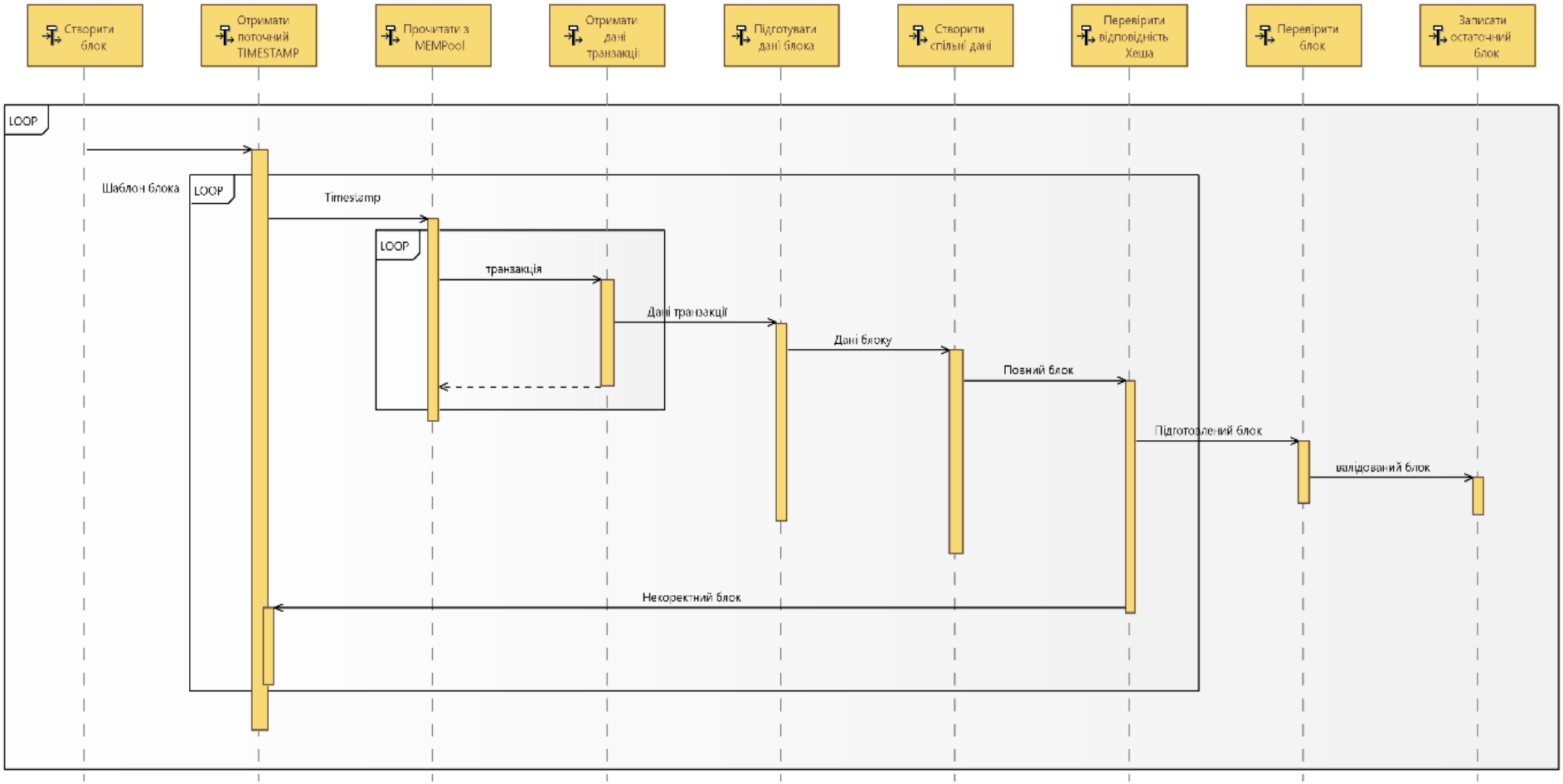


Рисунок 3.8 – Діаграма послідовності роботи блокчейн підсистеми

Процес встановлення з’єднання між блокчейн вузлами розподіленої децентралізованої системи захисту комп’ютерних мереж відображено на діаграмі послідовності зображеній на рисунку 3.9.

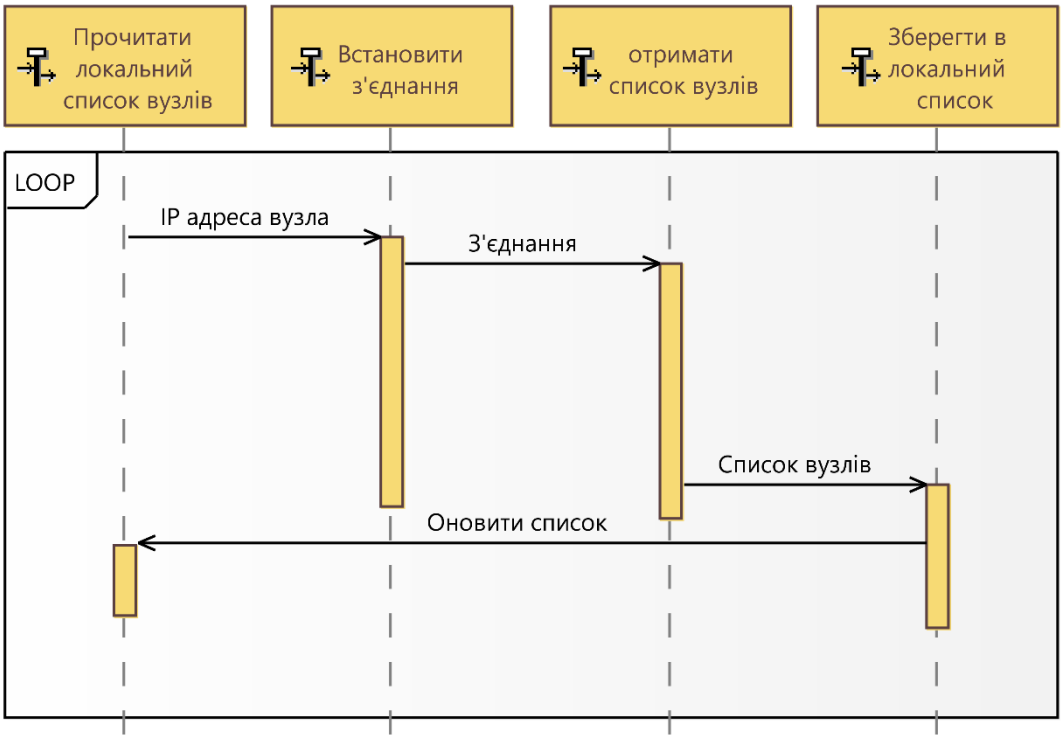


Рисунок 3.9 – Діаграма послідовності оновлення списка вузлів

Структура програмної реалізації blockchain підсистеми системи захисту комп’ютерних мереж представлена у вигляді діаграми класів наведеної на рисунку 3.10. Основна blockchain структура представлена тут трьома класами Chain, Block, Transaction відповідальними за ланцюжок, блок та транзакцію відповідно. При цьому як бачимо ланцюжок містить зв’язаний список блоків, а блок зв’язаний список транзакцій, при цьому, оскільки в деякі моменти часу можуть бути одночасно присутні кілька активних розгалужень списку, ланцюжок містить також посилання на альтернативні

голови списку (кожен блокчейн вузол обирає для себе головну гілку, роботу з якою і продовжує).

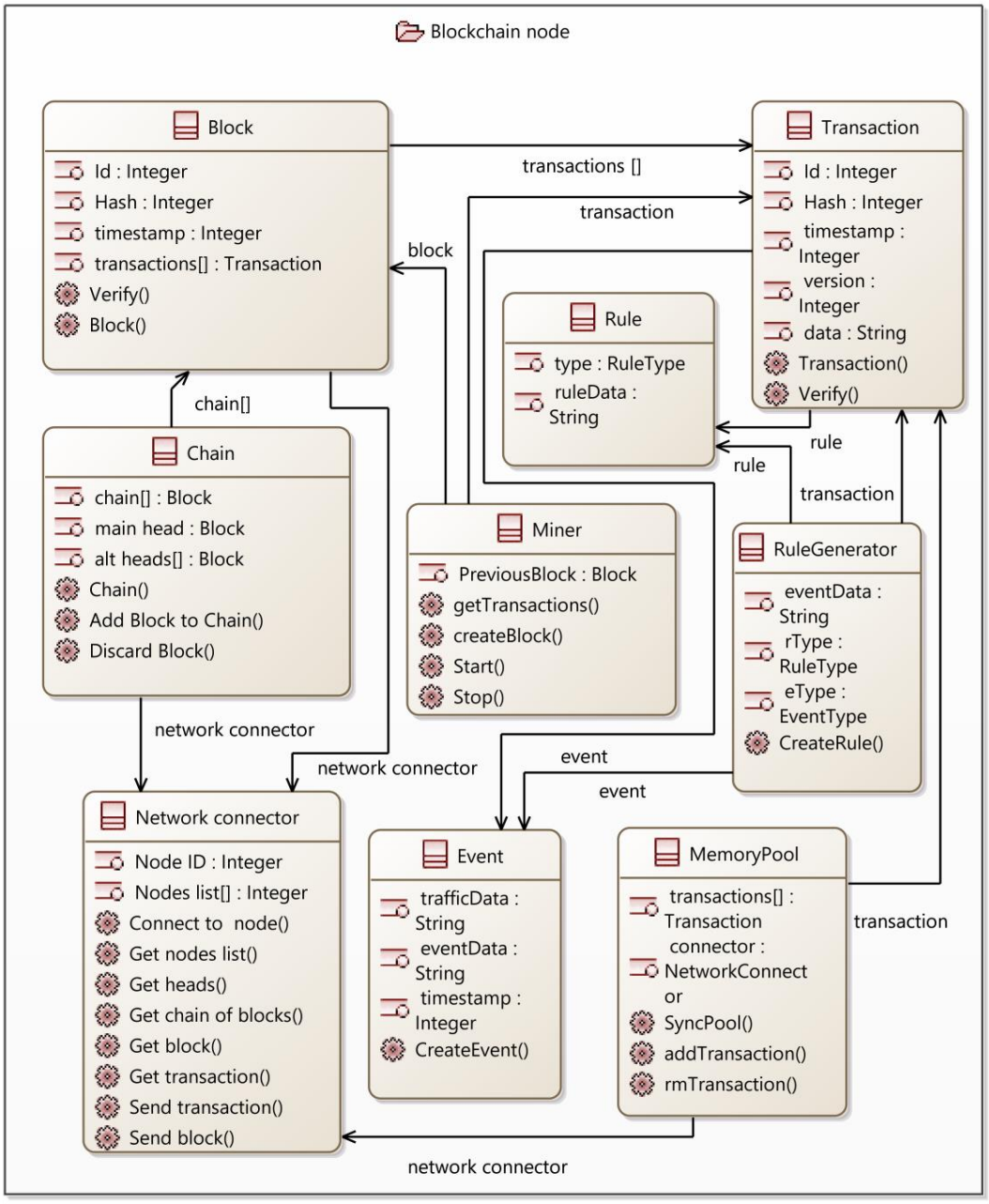


Рисунок 3.10 - Діаграма класів блокчейн підсистеми розподіленої системи захисту комп’ютерних мереж

Транзакції можуть створюватися на основі отриманих підсистемою подій, або згенерованих правил (класи Event та Rule). Класи Event та Rule містять тільки тип події або правила та інформацію про подію або правило, з методів доступні лише стандартні конструктори.

Після створення транзакції додаються до тимчасового пулу – MemoryPool, який синхронізується з іншими вузлами за допомогою NetworkConnector. Також NetworkConnector відповідальний за взаємодію blockchain підсистеми на різних вузлах і виконує відправку та прийом блоків та транзакцій між вузлами, забезпечуючи стабільну роботу розподіленої системи.

Клас Miner відповідальний за створення нових блоків спираючись на заданий протокол консенсусу. Методи цього класу спрямовані на вибір транзакцій з тимчасового пулу та заповнення нового блоку транзакціями таким чином, щоб блок проходив верифікацію відповідно до протоколу консенсусу.

3.4 Архітектура розподіленої системи захисту комп'ютерних мереж на основі blockchain

Для реалізації поставлених задач запропоновано наступну архітектуру вузла розподіленої системи виявлення вторгнень на основі blockchain, що представлена у вигляді діаграми розгортання(Рисунок 3.11).

Як бачимо, кожен вузол включає в себе п'ять основних компонентів:

- Модуль мережевого екрану – використовується для попередньої фільтрації трафіка та відсікання протоколів, які в даному сегменті мережі не використовуються, і таким чином зменшує навантаження на модуль виявлення вторгнень;
- модуль виявлення вторгнень – на відміну від мережевого екрану, даний модуль використовує більш складну систему правил, завдяки чому дозволяє виявляти значно більший набір атак, разом з модулем мережевого екрану, приймає участь у фільтрації трафіку для локальної мережі, захищеної системою виявлення вторгнень;
- модуль класифікації підозрілих подій, який використовує регресійні класифікатори та класифікатори, що використовують нейронні

мережі для аналізу підозрілих подій отриманих від модуля виявлення вторгнень та мережевого екрану, а також аналогічних даних отриманих від інших вузлів розподіленої системи захисту комп'ютерних мереж;

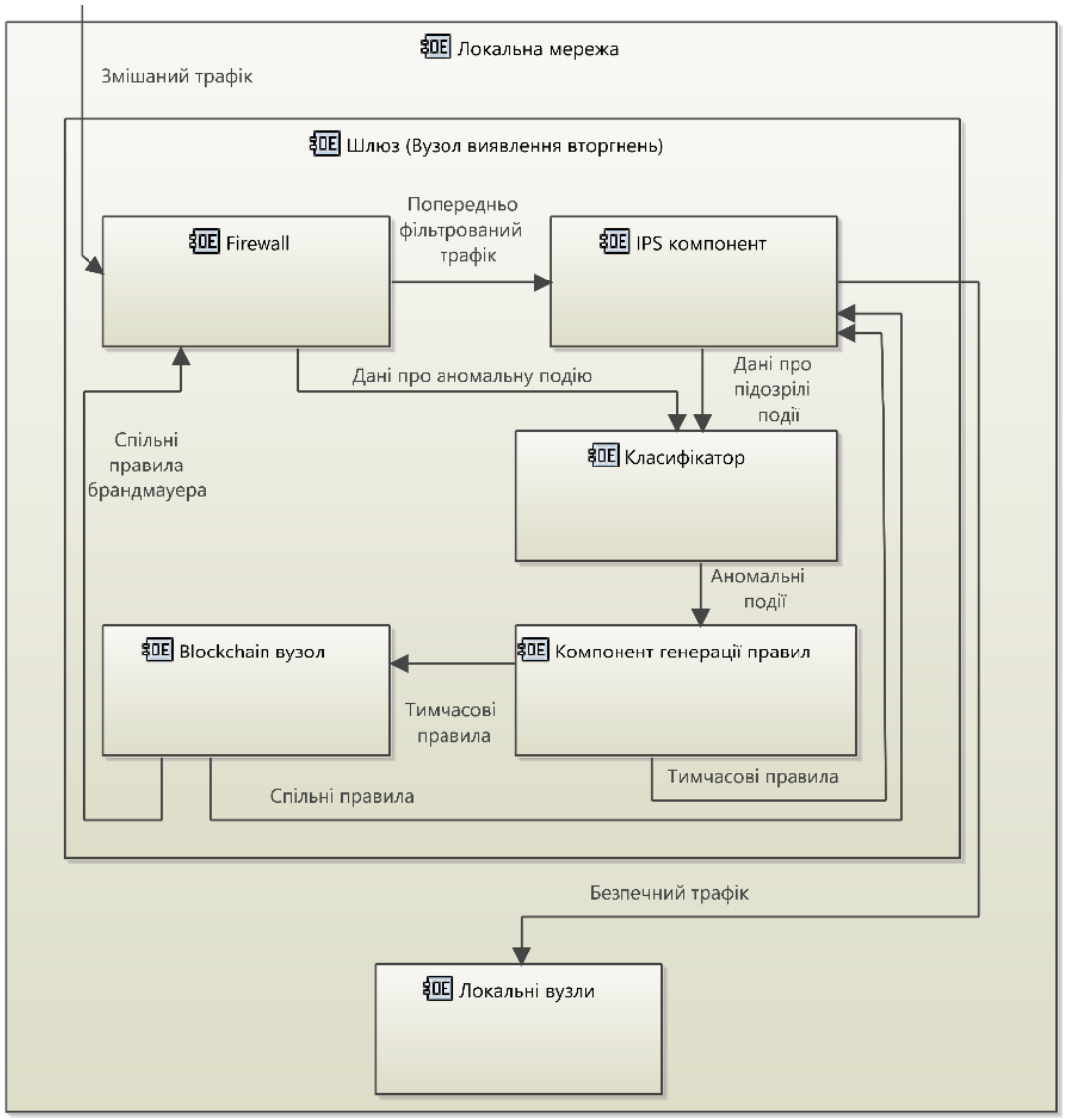


Рисунок 3.11 – Діаграма розгортання

– генератор правил – на основі результатів класифікації підозрілих подій даний модуль створює нові тимчасові правила, які дозволяють блокувати подібні атаки без етапу класифікації, створені правила

направляються на мережевий екран та модуль виявлення вторгнень, а також поширюються через блокчейн на інші вузли системи захисту комп’ютерних мереж;

– модуль блокчейн підсистеми – слугує для поширення між вузлами створених правил а також інформації про виявлені підозрілі події, включає в себе структуру блокчейн, а також необхідні для його функціонування компоненти, включаючи генератор блоків, верифікатор, а також мережевий компонент призначений для об’єднання блокчейн модулів в розподілену мережу і слугує основним сполучним компонентом для вузлів розподіленої системи захисту комп’ютерних мереж.

Більш детальний розгляд блокчейн модуля показує, що повноцінний блокчейн модуль має в свою чергу містити чотири основних компоненти, але в цілому для базової роботи з блокчейном достатньо лише двох, якщо ми відмовляємося від створення нових блоків і працюємо в режимі тільки читання. На перший погляд, це не має ніякого сенсу, але з врахування того, що створення нових блоків вимагає певних витрат системних ресурсів, для вузлів, що мають сильно обмежені процесорні ресурси вигідніше буде відмовитися від створення блоків, забезпечивши оптимальну роботу шлюза з системою захисту комп’ютерних мереж.

Повний набір блокчейн компонентів представлений на рисунку 3.12:

Базовий блокчейн компонент, відповідальний за взаємодію з іншими блокчейн вузлами за допомогою мережі, а також дозволяє приймати та передавати блоки та транзакції, є обов’язковим для функціонування блокчейн підсистеми;

Валідатор – компонент який забезпечує перевірку блоків, обов’язковий для використання, оскільки приймання неперевірених блоків небезпечне, оскільки невалідний блок може містити підмінені дані, а створення нових блоків без компонента валідації неможливе;

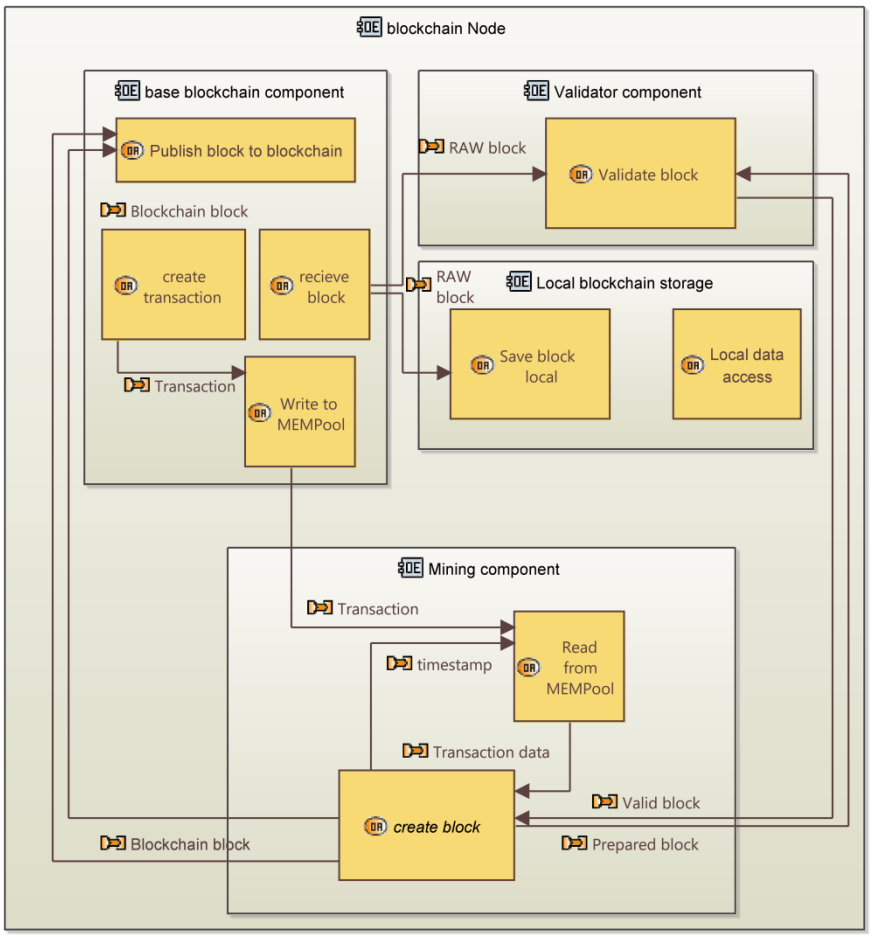


Рисунок 3.12 – Діаграма компонентів блокчейн підсистеми розподіленої системи захисту комп’ютерних мереж

Локальне блокчейн сховище – ланцюжок блоків, необов’язковий блокчейн компонент, відповідальний за зберігання локальної копії даних блокчейн, завдяки чому підвищує надійність зберігання даних для блокчейн мережі, але для вузлів, які не мають достатнього об’єму сховища від даного компоненту можна відмовитися на користь збереження системних ресурсів;

Майнінговий компонент – забезпечує створення нових блоків відповідно до протокола консенсусу, необов’язковий компонент, але в контексті розподіленої системи захисту комп’ютерних мереж бажано, щоб

кожен вузол мав такий компонент, що дозволить підвищити надійність системи та захистити її від підміни блоків.

3.5 Структура даних блокчейн компонента

Для розподіленої системи захисту комп'ютерних мереж невід'ємною частиною є блокчейн, що виконує роль сховища для основних даних які мають зберігатися і поширюватися між вузлами розподіленої системи виявлення вторгнень.

Логічну структуру блокчейн компонента показано на рисунку 3.13. Де увесь блокчейн являє собою ланцюжок блоків, що починається з genesis блока – тобто такого блока, який заздалегідь прописаний в програмному забезпеченні і завжди вважається верифікованим, від цього блока вузли починають будувати ланцюжок блоків заповнених необхідною інформацією.

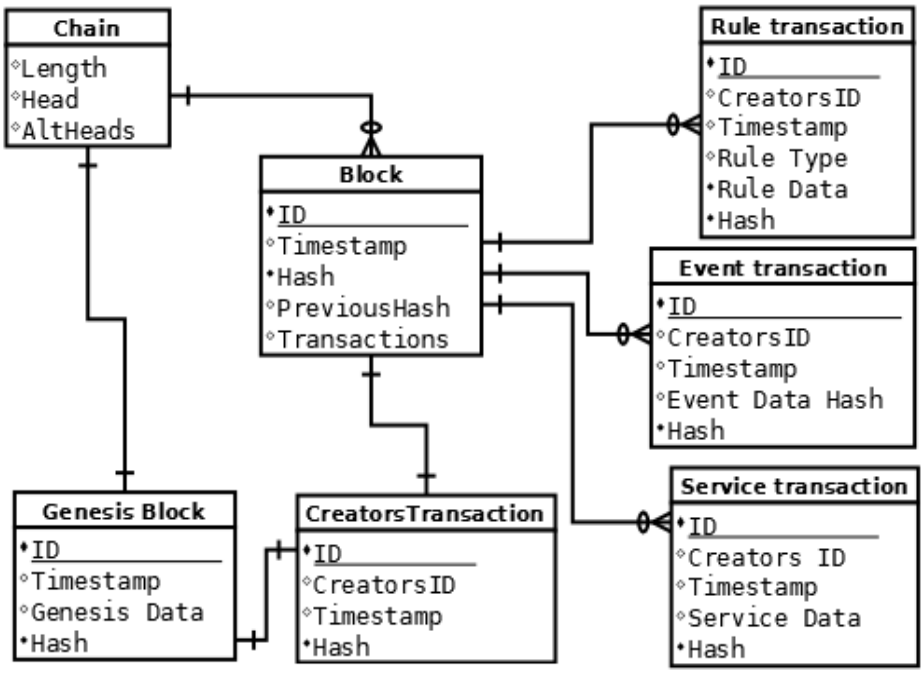


Рисунок 3.13 – Структура даних блокчейн компонента

Кожен блок починається з транзакції автора (Creators Transaction) яка записує, яким вузлом був створений даний блок, також ця транзакція записує час створення блока. За цією транзакцією в нашому випадку визначається,

скільки часу пройшло з моменту створення останнього блоку для кожного з вузлів. Завдяки цьому з’являється можливість зробити розподіл блоків між вузлами максимально рівномірним, при цьому враховуються лише транзакції з блоків основного ланцюжка блоків, до якого планується додавати новий блок, а розгалуження не враховуються.

Далі область даних блока заповнюється транзакціями взятими з тимчасового пулу, але при цьому кількість транзакцій, які записуються в блок обмежена максимальним числом транзакцій. Таке обмеження дозволяє обмежити максимальний розмір блока і завдяки цьому забезпечити стабільну передачу блоків з мінімальною вірогідністю зривів з’єднання. Основних типів транзакцій, якими заповнюється блок, окрім транзакцій автора три. Перший тип – це транзакція правило – така транзакція містить опис правила створеного системою захисту комп’ютерних мереж, яке записується в блокчейн. Наступний тип транзакцій це транзакції подій, які містять інформацію про виявлені та підозрілі події, але оскільки така інформація швидко втрачає актуальність і займає достатньо багато дискового простору, сенсу зберігати дані таких транзакцій в блокчейн немає, тому основні дані по таким транзакціям зберігаються окремо від блокчейн, а в самій транзакції записаний лише хеш MerkleTree даних, які зберігаються в окремих файлах. Ще одним типом транзакцій є службові транзакції, основна задача яких – обмін між вузлами службовими даними, які необхідні для коректної роботи розподіленої системи захисту комп’ютерних мереж.

Висновки до розділу 3

Запропонована архітектура розподіленої системи захисту комп’ютерних мереж на основі блокчейн дозволяє забезпечити надійний захист локальних мереж невеликих підприємств та домашніх мереж від вторгнень з глобальної мережі. Це досягається за рахунок використання

даних зібраних багатьма вузлами розподіленої системи з різних мереж. Архітектура передбачає п’ять основних компонентів, які розміщуються на мережевому шлюзі – мережевого екрану, модуля виявлення вторгнень, класифікатора, генератора правил, а також блокчейн компонента.

До основних переваг такої архітектури можна віднести:

1) Модульна архітектура дозволяє легко оновлювати та навіть заміняти на альтернативні окремі компоненти системи, а також створювати додаткові плагіни для неї. Це відкриває широкі можливості для подальшого розвитку системи і дає можливість в подальшому використовувати більш ефективні компоненти для класифікації подій та виявлення вторгнень.
2) Розподілена архітектура на основі блокчейн дозволяє значно підвищити надійність зберігання даних та захисту комп’ютерних мереж, при цьому завдяки децентралізації жоден з вузлів не залежить від інших, тому вихід з ладу конкретних вузлів ніяк не впливає на працездатність інших.
3) Спільне використання інформації отриманої різними вузлами, як одна з реалізацій колаборативного виявлення вторгнень, дозволяє більш ефективно виявляти вторгнення та шкідливі події та готувати вузли, які ще не були атаковані до атак такого типу, і виявляти потенційно шкідливі зони у глобальній мережі, з яких зафіксована велика кількість шкідливого трафіку.
4) Захищений обмін даними з використанням блокчейн технології дає можливість оперативно обмінюватися інформацією необхідною для роботи розподіленої системи, гарантуючи її коректність та незмінність, що дозволяє безпечно поширювати оновлення сигнатурних баз та іншої інформації без центрального довіреного вузла.

Результати досліджень приведених в розділі опубліковані в роботах [76, 77]

РОЗДІЛ 4
ПРАКТИЧНА РЕАЛІЗАЦІЯ МОДЕЛІ РОЗПОДІЛЕНОЇ СИСТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ В КОМП'ЮТЕРНІ МЕРЕЖІ МАЛИХ ТА СЕРЕДНІХ ПІДПРИЄМСТВ НА ОСНОВІ BLOCKCHAIN

4.1 Формування вимог до розподіленої системи виявлення вторгнень в комп'ютерних мережах малих та середніх підприємств

Для забезпечення придатності розроблюваної системи для використання в SOHO та SMB мережах вона повинна мати наступні характеристики.

Система повинна бути повністю розподіленою, оскільки SOHO та SMB мережі здебільшого пропонують дешеві апаратні рішення із середньою, а іноді й низькою, надійністю, звідки випливає, що будь-який вузол може припинити роботу, але при цьому система повинна продовжувати працювати. Наявність центральних вузлів знижує надійність і водночас підвищує вимоги до їх апаратної складової.

Ефективне виявлення вторгнень також є важливим параметром, оскільки відсутність захисту від більшості відомих атак робить систему непрацездатною. Отже хороша IDS потребує періодичного оновлення сигнатурних баз. Також IDS повинна мати евристичні модулі, які допомагають виявляти нові, не прописані в базах атаки та створювати правила для їх швидкого пошуку.

Колаборативний підхід до виявлення вторгнень, означає, що кожен вузол використовує інформацію про виявлені події та наявні оновлення від сусідніх вузлів для виявлення та запобігання вторгненням. В першу чергу це допомагає швидше виявляти невідомі для цього вузла вторгнення.

Легке конфігурування дуже важливе для SOHO та SMB мереж, де користувачам потрібна система, яку буде легко налаштувати. Тому

найкращим рішенням для такого типу мереж буде IDS, встановлена з оптимальними налаштуваннями за замовчуванням, і адміністратор може вибрати, які послуги використовуються в цій мережі та рівень чутливості виявлення. Звичайно, система повинна мати можливість розширеної конфігурації для тих, хто може і знає як налаштувати налаштування вручну.

Платформонезалежність – система, повинна працювати на апаратному забезпеченні загального призначення та бути сумісною з якомога більшою кількістю апаратних засобів, оскільки конкретне обладнання може бути занадто дорогим для SOHO та SMB сегменту.

4.2 Реалізація моделі

4.2.1 Вибір платформи (програмної та апаратної)

Першим етапом розробки розподіленої системи виявлення вторгнень в SOHO та SMB мережах є вибір апаратної та програмної платформи зручної для розробки розгортання та використання такої системи. Оскільки робота системи пов’язана з перехопленням та фільтрацією мережевих пакетів, операційна система повинна мати пристосований для цього мережевий стек. Оскільки фільтрація пакетів зазвичай вимагає достатньої обчислювальної потужності, в якості апаратної платформи було обрано X86_64 сумісну платформу. До переваг такої апаратної платформи можна віднести широкий вибір операційних систем та варіантів апаратної складової, гарну обчислювальну потужність та розповсюдженість платформи. Альтернативним варіантом може бути ARM платформа, але на даний момент підібрати таке обладнання на ARM платформі, яке б відповідало вимогам по обчислювальній потужності та наявності кількох мережевих адаптерів, достатньо складно. Отже, остаточним варіантом апаратної платформи обираємо X86_64, що дасть можливість використовувати стандартне серверне і користувацьке обладнання в якості мережевого фільтра.

В якості базової операційної системи найкращим варіантом буде використання Unix подібних систем, зокрема FreeBSD, завдяки непоганому мережевому стеку, а також наявності усіх необхідних компонентів та додаткового програмного забезпечення. Непоганим варіантом операційної системи на основі FreeBSD є OpnSense. Окрім наявності необхідних базових компонентів, ця система має зручний вебінтерфейс, що дозволяє керувати вбудованими компонентами, і використовувати систему на апаратних платформах, не підключених до монітора.

Якщо ж розглядати платформу для вузлів, основна задача яких це підтримка роботи блокчейн компонента, то будь-яка Unix-сумісна система буде непоганим вибором. Але такий сервер вже не буде виконувати фільтрацію пакектів та виявлення вторгнень, що знизить навантаження на такий сервер і дасть можливість більш продуктивно обробляти блокчейн транзакції.

Для забезпечення сумісності з максимальною кількістю операційних систем та апаратних платформ, в якості мови програмування для розробки блокчейн підсистеми було обрано мову Python 3, завдяки непоганій кросплатформенності та наявності великої кількості бібліотек під усі необхідні для розробки задачі.

4.2.2 Компоненти фільтрації та виявлення вторгнень

В якості компонентів для фільтрації мережевого трафіка та виявлення вторгнень, найкращим варіантом в нашому випадку буде використовувати вбудовані в систему OpnSense компоненти PacketFilter та Suricata. Перевагою використання вбудованих компонентів в нашому випадку є краща їх інтеграція в систему та відсутність необхідності їх додаткового встановлення, що спрощує розгортання системи вцілому. Окрім того вбудовані компоненти мають зручний вебінтерфейс для конфігурації, завдяки чому спрощується процес їхнього налаштування.

PacketFilter – мережевий екран, що використовується в системі FreeBSD та системах побудованих на її основі. Має достатньо широкий функціонал мережевого фільтра, включаючи звичайну фільтрацію, прозору фільтрацію другого рівня, NAT, балансування з'єднань, гнучкі налаштування правил, включаючи макроси та мітки трафіка.

Складається PacketFilter з двох компонентів - самого фільтра та утиліти керування, відповідальної за запуск, зупинку оновлення конфігурації та ін. Фільтр працює на рівні ядра, використовуючи системні виклики ioctl [128], тому утиліта керування для роботи фільтра не обов'язкова, але без неї керування роботою фільтра сильно ускладнюється.

Достатньо цікавою особливістю PacketFilter є наявність таблиць адрес, що надає PacketFilter переваги в швидкодії та гнучкості налаштувань у порівнянні з конкуруючими продуктами. Таблиці можуть містити IP адреси (як 4 так і 6 версії), при цьому можливість відмічати адреси як виключення дозволяє описувати таким чином достатньо складні топології. При цьому використання таблиць дозволяє сильно пришвидшити пошук адрес, в порівнянні з використанням звичайних правил, що дає можливість використовувати більше записів для фільтрації без втрати швидкодії. Окрім того таблиці дозволяють вести статистику для кожної з записаних адрес, а також використовувати умовні записи – коли в таблицю заноситься адреса, для якої був перевищений певний кількісний параметр трафіка. Також таблиці дозволяють автоматично видаляти записи, для яких термін дії перевищено. За допомогою системних викликів керувати таблицями мережевого екрана можуть і деякі сторонні утиліти, робота яких непряму пов'язана з мережевим екраном, наприклад DHCP сервер, який записує в таблиці активні, звільнені та заборонені адреси.

Основна задача мережевого екрану для розподіленої системи виявлення вторгнень полягає в попередній фільтрації трафіка, та трансляції

адрес для підключення локальної мережі, що знаходиться під захистом розподіленої системи, до глобальної мережі.

В якості базового компонента для виявлення вторгнень будемо використовувати вбудований компонент Suricata, оскільки це одне із популярних opensource рішень для виявлення вторгнень, і на відміну від Snort, дане рішення значно краще працює з багатопотоковими системами, і окрім цього дозволяє використовувати GPU для прискорення обробки, що дає непогану продуктивність на швидкостях до 10ГБіт/с. Однією з основних відмінностей IDS від мережевого екрана полягає в тому, що мережевий екран не має доступу до вмісту пакетів, і усі фільтрації відбуваються лише на рівні обробки заголовків пакетів, тоді як IDS мають доступ до глибокого аналізу пакетів, і окрім того система правил IDS дає значно ширші можливості для налаштування фільтрації.

Процес обробки мережевих пакетів в Suricata можна розділити на такі кроки:

- Захоплення пакета - на цьому етапі відбувається перехоплення пакетів на усіх інтерфейсах, для яких задано відповідні налаштування.
- Декодування пакета - на цьому етапі система намагається отримати доступ до внутрішнього вмісту пакета для подальшої обробки.
- Виявлення основі правил – дозволяє визначати пакети, що відповідають правилам на основі усіх доступних параметрів.
- Реагування – на цьому етапі відповідно до правил система реагує на ту чи іншу подію.

Набори правил для Suricata стандартні і сумісні з Snort, завдяки чому в якості базових наборів правил можна використовувати відкриті набори правил спільноти Snort, що дасть нам непоганий базовий набір правил, для попереднього виявлення відомих атак.

Окрім того, в OpnSense вбудований вебінтерфейс керування Suricata, який покриває основні функції системи виявлення вторгнень та надає гнучкі

можливості керування наборами правил, в тому числі автоматичне завантаження та оновлення правил з репозиторіїв.

Також серйозною перевагою Suricata є можливість збереження логів у форматі JSON що значно спрощує подальшу обробку даних про зафіксовані події. В іншому ж випадку довелось би спочатку створювати обробник, який би в першу чергу виконував парсинг файлів журналу подій, що за відсутності стандартизованого формату значно ускладнило б таку обробку. Обробник файла з логами подій побудовано таким чином, щоб забезпечити циклічну перевірку файла на наявність нових подій, і після додавання нової події ініціюється механізм створення нової транзакції. Тобто блокчейн компонент відслідковує події, які згідно поточної політики безпеки повинні бути занесені до блокчейну, і при виявленні такої події створюється нова транзакція, що дає можливість поширити інформацію про подію між усіма вузлами-учасниками розподіленої системи виявлення вторгнень.

Лістинг 4.1 – Код функції transactionCreateFromLog

```
    def transactionCreateFromLog():
        if os.path.isfile("/var/log/suricata/eve.json"):
            file = open("/var/log/suricata/eve.json", "r")
            Block.threads_stopped[1]=1
            file.seek(0,2)
            while 1:
                where = file.tell()
                line = file.readline()
                if Block.app_stopped:
                    print("Transaction thread stopped")
                    Block.threads_stopped[1]=1
                    sys.exit(0)
                if not line:
                    time.sleep(1)
                    file.seek(where)
                else:
                    obj = json.loads(line)
                    t=Transaction(2, 222, int(time.time()))
                    t.gen(obj)
```

```
                    chain.addTransactionToPool(t)
                    for node in nodes:
                        try:
                            x=requests.post(node[1]+'transaction/',
data={'transaction':t.serialize()[1]}, timeout=2.5)
                        except:
                    transactions_self_to_csv(t)
        pass
```

4.2.3 Класифікатори

В рамках реалізації та випробування роботи системи були реалізовані наступні класифікатори – логістична регресія, класифікатор на основі опорних векторів, та ансамблевий класифікатор AdaBoostClassifier. Класифікатор на основі штучної імунної системи в рамках даної роботи не реалізовувався, у зв’язку з значно складнішим процесом його тренування і необхідністю налаштування під кожну конкретну систему та режим її використання.

Процес створення класифікаторів можна розділити на кілька етапів:

1) Підготовка навчального та тестового набору даних та розробка модуля підготовки та попередньої обробки даних
2) Навчання класифікатора
3) Тестування класифікатора
4) Створення остаточного модуля класифікації

На першому етапі підготовки та попередньої обробки даних необхідно для початку провести їх детальний аналіз, використовуючи доступні засоби. Зручним варіантом для обробки даних від модуля виявлення вторгнень є Python з Jupyter Notebook – така комбінація мови програмування та середовища виконання дозволяє користуватися потужним набором бібліотек Python для аналізу даних, при цьому зберігаючи можливість інтерактивного виконання коду з графічним інтерфейсом та використовувати

потім отриманий код для створення остаточного модуля класифікації без кардинальних змін, що є серйозною перевагою в порівнянні з вузькоспеціалізованими математичними середовищами виконання типу Matlab.

Оскільки далеко не усі поля що містяться в звітах системи виявлення мають числовий формат, необхідно виконати їх перетворення в числовий вигляд, наприклад для даних які мають текстовий формат, можна застосувати маркування числовими мітками, так як усі класифікатори працюють саме з числовими даними. Першим кроком побудуємо матрицю кореляції параметрів, таким чином можемо виявити вплив кожного з параметрів на інші, а також видалити параметри, які негативно впливають на генералізацію. До таких параметрів належать час відправки, IP адреса та порт, оскільки в контексті одного запису такі параметри ніяк не допомагають в виявленні шкідливої активності. Для конвертації ж нечислових даних в числові, для випадку, коли можливий набір значень для поля обмежений можна методом get_dummies, який для кожного із значень, які зустрічаються в наборі створює поле, що містить 0 або 1 і вказує на наявність або відсутність такого значення для поля.

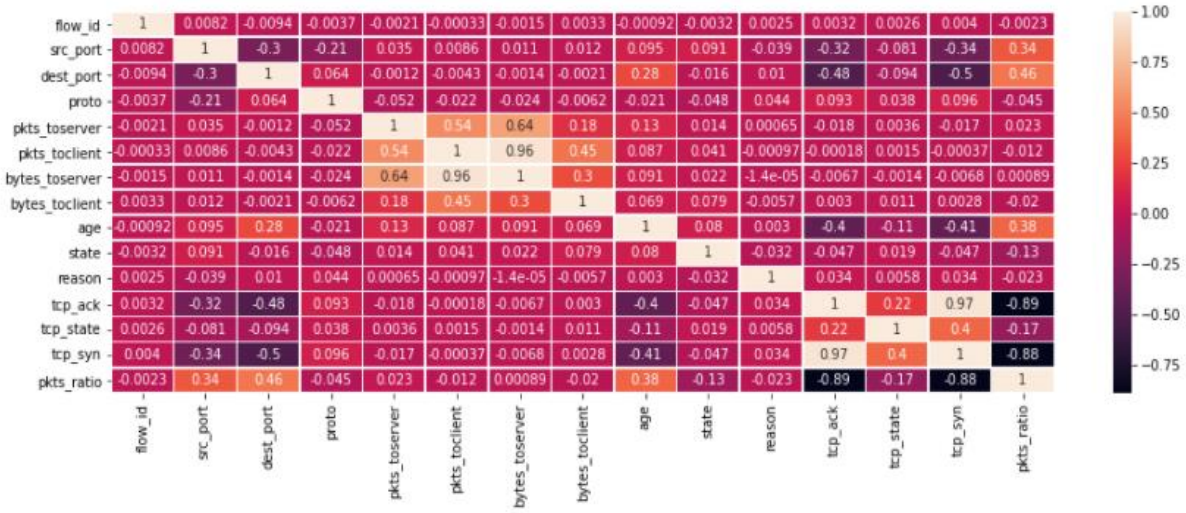


Рисунок 4.1 - Матриця кореляції параметрів трафіку

Для реалізації класифікатора мовою Python найкращим варіантом є використання бібліотеки Scikit-learn – це одна з найбільш часто використовуваних бібліотек для обробки даних та машинного навчання. Дана бібліотека містить велику кількість реалізованих алгоритмів регресій та класифікаторів. Заснована бібліотека Scikit-learn на бібліотеках NumPy та SciPy .

Для реалізації логістичної регресії скористаємося класом LogisticRegression з бібліотеки Scikit-learn. Основні параметри класу *multi_class* - параметр відповідальний за вибір типу класифікації *«ovr»* один проти інших або *«multinomial»* багато проти багатьох, *solver* алгоритм вирішення задачі оптимізації, *random_state* – ініціалізація псевдовипадкового генератора для повторюваності результатів та можливості їх оцінки в процесі підгонки моделі, *n_jobs* задає кількість потоків, а *verbose* задає рівень докладності виведення інформації. Після створення класу необхідно виконати тренування з використанням тренувального набору записів, і оцінити точність роботи моделі використовуючи тестовий набір. Такий метод навчання регресійного класифікатора дозволяє оцінити якість моделі на відмінному від тренувального наборі і уникнути таким чином перенавчання, коли модель гарно працює на тренувальному наборі, але на відмінних наборах точність різко знижується.

Лістинг 4.2 – Код регресійного класифікатора

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(multi_class='multinomial', solver='newton-cg',
random_state=42, n_jobs=-1, verbose=10)
clf.fit(X_train, y_train)
prediction = clf.predict(X_train)
accuracy_score(y_train, prediction)
```

На тестовому наборі логістична регресія дає нам точність близько 88.7%, що є достатньо непоганим результатом.

Для реалізації класифікатора на основі опорних векторів скористаємося класом SVC. Деякі параметри для цього класу відрізняються, тут *tol* - параметр що відповідальний за толерантність для зупинки процесу тренування, а *C* параметр регуляризації. Для даного класу навчання відбувається подібно до логістичної регресії. На тестовому наборі отримуємо точність 91,87%, що краще ніж у випадку з логістичною регресією.

Лістинг 4.3 – Код класифікатора SVC

```
from sklearn.svm import SVC
rf = SVC(random_state=42, tol=1e-4, verbose=10, C=1.5)
rf.fit(X_train, y_train)
prediction = rf.predict(X_train)
accuracy_score(y_train, prediction)
```

В якості ж ансамблевого класифікатора використаємо AdaBoostClassifier, єдиним параметром для нього вказуємо *base_estimator* який вказує на основі якої моделі будується ансамбль. В нашому ж випадку обираємо DecisionTreeClassifier який є звичайним деревом рішень. Для підбору параметрів ансамблевої моделі потрібно також скористатись класом GridSearchCV, параметр *cv* задає розподіл набору для кросвалідації, що дозволяє підвищити точність класифікації, а параметр *scoring* задає стратегію оцінки продуктивності кросвалідації. Використання AdaBoostClassifier дозволило підняти точність оцінки на тестовому наборі до 92.27%, що перевищує значення отримані для попередніх класифікаторів.

Лістинг 4.4 – Код класифікатора AdaBoostClassifier

```
abc = AdaBoostClassifier(base_estimator=DecisionTreeClassifier())
parameters = {'base_estimator__max_depth':[i for i in range(12,25,3)],
              'base_estimator__criterion':['entropy'],
              'n_estimators':[25, 50, 100, 250],
              'learning_rate':[0.01, 0.1],
              }
CV_clf = GridSearchCV(abc, parameters, verbose=10, cv=3, scoring='f1_macro',
n_jobs=-1)
CV_clf.fit(X_train, y_train)
best_estimator = CV_clf.best_estimator_
```

```
prediction = best_estimator.predict(X_train)
accuracy_score(y_train, prediction)
```

Комбінуючи дані отримані трьома моделями отримуємо простий модуль класифікації подій зафіксованих IDS, і таким чином отримуємо можливість відсіяти випадкові події, а також передати явно шкідливі події на модуль генерації правил, для того, щоб зменшити обчислювальне навантаження на систему для їх виявлення.

4.2.4 Блокчейн компонент

Розробка блокчейн компонента починається зі сховища для блоків, оскільки основна задача блокчейн компонента полягає в зберіганні та поширенні даних. Для зберігання основних даних збережених в ланцюжку блоків скористаємося бінарним файлом, який міститиме в дані в сирому вигляді, а для індексації блоків в ланцюжку скористаємося базою levelDB, що дозволяє пришвидшити пошук блоків в бінарному файлі, оскільки бінарний файл з часом збільшується в розмірі і задача пошуку без індексації ускладнюється. Для блоків база містить записи формату ключ-значення, де ключем є хеш блока, а значенням адреса його зсуву відносно початку файлу. У випадку ж якщо у нас є необхідність обмежити максимальний розмір файлу, наприклад у зв’язку з використанням файлової системи FAT32 можна використовувати максимальний розмір файлу як модуль, і тоді значення зсуву взяте по модулю вкаже зсув відносно файлу, в якому записаний конкретний блок, а значення зсуву поділене на максимальний розмір файлу вкаже на номер файлу, в якому даний блок записаний. Такий принцип роботи зі сховищем є стандартним для більшості блокчейн застосунків, і показує непогану продуктивність. При розповсюдженні ланцюжка блоків передавати базу даних немає необхідності, оскільки база генерується на вузлі в процесі верифікації блоків. Обов’язкова верифікація блоків при їх отриманні сильно сповільнює початкове завантаження ланцюжка блоків, але в цілях безпеки такий підхід застосовується для практично усіх блокчейн застосунків, а

оскільки повний ланцюжок блоків завантажується лише при першому запуску, заважати роботі програми це не буде. Робота з базою в нашій системі реалізовується класом DBConnector, який містить основні методи для підключення до бази та роботи з її записами.

Лістинг 4.5 – Код функції makeBlockRecord

```
    def makeBlockRecord(self, headerHash, header, blockid, trcount, offset):
           self.blockDB.put(b'b'+headerHash,  header+int(blockid).to_bytes(4,
'little')+int(trcount).to_bytes(4,          'little')+int(offset).to_bytes(4,
'little'), sync=True)
    def readBlockRecord(self, headerHash):
            rec = self.blockDB.get(b'b'+headerHash)
            if rec is not None:
                return (rec[0:71], rec[71:75], rec[75:79], rec[79:83])
            else:
                return (b'', b'', b'', b'')
```

Для представлення об’єкта блока створюємо клас Block який описує всі основні поля блока та методи для створення, запису та зчитування блоків їх майнінгу, верифікації та серіалізації для передачі в мережі. Для запису в файл, як вже було сказано раніше, дані записуються в бінарному форматі спираючись на фіксовані зсуви, та вказані в самому файлі розміри блоків. Але передачі ж через мережу такий формат не є зручним, оскільки його верифікація може бути ускладнена, так як бінарний формат без надлишковості не передбачає перевірки на цілісність, тому для зручної передачі та подальшого відтворення переданих блоків на інших вузлах вирішено було обрати формат Yaml як такий, що дозволяє зручно кодувати та передавати об’єкти в текстовому вигляді, окрім того відлагоджування такої передачі даних також достатньо зручне.

Конструктор приймає в якості параметра тільки індекс блока, оскільки інші параметри, які встановлюються конструктором прописані в коді, і залежать або від версії протоколу або від попередніх блоків або від вузла, що створює блок.

Метод writeToFile дозволяє створити новий запис у бінарному файлі блоків у відповідному форматі. Спочатку виконується перевірка на наявність відповідного файлу, далі виконується запис усіх необхідних полів, і на останньому етапі виконується створення запису в базі даних.

ReadFromFile метод призначений для зчитування блока з бінарного файлу. Даний метод приймає один параметр з індексом блока, який необхідно зчитати. Далі створюється новий об’єкт блока, в який записуються зчитані з бінарного файлу поля, при цьому також створюються усі транзакції, що були записані в блок.

Метод calcMerkleHash підраховує хеш дерева доданих в блок транзакцій, отриманий в результаті хеш дозволяє перевірити цілісність вмісту блоку.

Лістинг 4.6 – Код функції makeBlockRecord

```
    def calcMerkleHash(self):
            transactionhashes=[]
            for t in self.transactions:
                transactionhashes.append(t.getHash())
            while len(transactionhashes)>1:
                transactionhashesnew=[]
                if (len(transactionhashes) % 2 != 0) :
                    transactionhashes.append(transactionhashes[-1])
                transactionhashes.reverse()
                while len(transactionhashes)>0:

transactionhashesnew.append(hashlib.sha256(transactionhashes.pop()+transacti
onhashes.pop()).digest())
                transactionhashes = transactionhashesnew
            return transactionhashes[0]
```

Методи blockToYaml та blockFromYaml дозволяють серіалізувати блоки для їх подальшої передачі мережею та відновлювати серіалізовані блоки після їх отримання відповідно.

Метод verify виконує верифікацію блока на відповідність протоколу консенсусу. На першому етапі іде розрахунок цільового значення для конкретного блоку, виходячи з того, який вузол створював блок і скільки часу він накопичив. Далі іде порівняння хеша заголовка з цільовим значенням, якщо значення хеша менше, блок вважається верифікованим.

Метод mine реалізовує POS майнінг, принцип якого ґрунтується на спробах створити створити блок, в сприятливі для цього проміжки часу, коли розмір ставки сприятливий для створення блока, що пройде верифікацію [77]. Для збільшення вірогідності створення блока одним із вузлів список транзакцій, які записуються в блок сортується в випадковому порядку, що впливає на хеш дерева Меркла, який впливає на хеш заголовка.

Клас Transaction реалізує об’єкти транзакцій і має достатньо просту реалізацію. Він містить методи для створення та серіалізації транзакцій в YAML формат, а також їх хешування. В якості параметрів конструктор транзакції приймає індекс транзакції, тип транзакції, ідентифікатор вузла, та час створення транзакції. Основні типи транзакцій це транзакція автора блоку, яка записується в блок першою і в ній вказано автора блоку та час його створення, що відіграє важливу роль в роботі протоколу консенсусу. Наступний тип транзакції – порожня транзакція, ніякого змістовного навантаження такі транзакції не несуть, але можуть використовуватися при тестуванні роботи системи або для заповнення блоків і більш швидкого їх підтвердження при дефіциті реальних транзакцій, що дозволить уникнути непередбачуваних затримок при створенні нових блоків. Основний тип транзакцій – це транзакції, що містять інформацію про шкідливі та підозрілі події. І останній, але не за важливістю, тип транзакцій – це транзакції, що містять оновлення правил для підсистеми виявлення вторгнень.

Клас Chain містить методи необхідні для роботи з ланцюжком блоків, зокрема chooseMainHead – метод що дозволяє обрати головний ланцюжок у

випадку розгалуження, даний метод викликається періодично і сканує усі доступні ланцюжки, вимірюючи їх довжину.

Лістинг 4.7 – Код функції chooseMainHead

```
       def chooseMainHead(self):
            if len(self.altHeads)>1:
                fl = self.calculateForkLength()
                print(("Fork length is: ", fl))
                headLength=[]
                for head in self.altHeads:
                    headLength.append(self.getBlocknoVerify(head).time      -
fl['timestamp'])
                hlc = headLength.copy()
                if fl['depth']<3:
                    return
                if len(hlc)>1:
                    hlc.sort(reverse=True)
                    diff=hlc[0]-hlc[1]
                    if (diff>120) or (fl['depth']>7):
                        try:
                            mainHead=self.altHeads[headLength.index(hlc[0])]
                            self.altHeads = [mainHead]
                            self.lastHash = mainHead
                            Block.lasthash = self.lastHash
```

Метод addBlock дозволяє додавати нові блоки у ланцюжок, оскільки не дивлячись на те, що блоки мають криптографічний зв’язок один з одним, об’єкт ланцюжка необхідний нам для зручного оперування блоками. В якості параметра цей метод приймає новий блок, що повинен додатися в ланцюжок, і на основі даних цього блока коригуються поля об’єкта поточного ланцюжка.

Також даний клас містить методи для роботи з тимчасовим пулом addTransactionToPool, та removeFromPool, які відповідно додають та видаляють транзакції з тимчасового пулу. В якості параметрів обидва методи приймають посилання на об’єкт транзакції.

Для забезпечення обміну даними між блокчейн вузлами в мережі застосовано фреймворк Flask, який дозволяє створювати зручні API

інтерфейси, що працюють на основі HTTP протоколу. Для створення такого API в першу чергу необхідно запустити мережевий сервіс, що буде прослуховувати вказаний порт і встановлювати з'єднання з іншими клієнтами. Також необхідно створити обробники маршрутів, які будуть викликати методи, що відповідають тому чи іншому запиту з відповідною адресою. В якості прикладу наведемо обробник запиту ланцюжка блоків.

Лістинг 4.8 – Код функції get_chainc

```
    @app.route('/chain/<head>', methods=['GET'])
    def get_chainc(head):
        try:
            blockIndex=chain.getBlockIndex(unhexlify(head))
            return yaml.dump(blockIndex)
        except:
            return 'None'
```

Такий обробник приймає запит на отримання ланцюжка з блоків, який закінчується блоком з вказаним хешем. Хеш передається з HTTP запитів у відповідний метод обробник. Обробник перевіряє наявність запитаного ланцюжка або блока і при його наявності відправляє відповідні дані, а у випадку відсутності повертається повідомлення про помилку. Значення що повертається методом обробником надсилається у відповідь на запит, таким чином, вузол що виконував запит отримує необхідні дані або повідомлення про неможливість їх отримання.

Блокчейн підсистема розподіленої системи виявлення вторгнень має багатопотокову реалізацію, що дає можливість більш оптимально використовувати ресурси багатопроцесорних систем. Основні задачі розподілені між потоками наступним чином: перший потік відповідальний за підтримання ланцюжка блоків в актуальному стані, включаючи обробку та додавання нових блоків, а також синхронізацію стану ланцюжка з іншими блокчейн вузлами, другий потік відповідальний за створення транзакцій, на основі даних отриманих від модуля виявлення вторгнень, третій потік реалізовує POS майнер і відповідальний за створення нових блоків [124].

Інші задачі виконуються основним потоком, включаючи обробку API запитів та керування потоками.

У випадку необхідності завершення роботи одного з вузлів блокчейн підсистеми, завершення роботи повинне бути виконано таким чином, щоб забезпечити збереження цілісності записаних даних. Для цього основний потік надсилає іншим потокам сигнал про необхідність завершення роботи і перемикається у режим очікування відповіді. Інші ж потоки, отримавши сигнал про необхідність завершення роботи доводять поточний етап виконуваного ними завдання до логічного завершення, якщо потік в цей момент працював, або просто завершують роботу, якщо потік в цей момент простоював. Головний потік, отримавши повідомлення про завершення інших потоків також виконує завершення роботи. Таке завершення роботи дозволяє уникнути виникнення помилок в бінарних файлах через переривання запису та зменшує ймовірність зриву передачі даних мережею іншим вузлам.

На випадок же аварійного завершення роботи блокчейн підсистеми, яке може бути спричинено як перебоями в живленні так і певними програмними збоями операційної системи, передбачається контроль цілісності бінарних файлів. І хоча такі події є скоріше виключеннями аніж нормою, можливість їх виникнення слід передбачити, окрім того пошкодження бінарних файлів можуть бути спричинені поганим станом носія, на якому зберігається блокчейн база. А спроба використання пошкоджених файлів може призвести до аварійного завершення роботи системи або виникнення помилок в роботі, а отже для попередження таких ситуацій потрібно вчасно виконувати перевірку та відновлення цілісності блокчейна. Для виконання такої перевірки на цілісність блокчейн бази, в залежності від налаштувань, при запуску відбувається сканування останнього або усіх файлів блокчейна. Сканування передбачає пошук усіх доступних блоків у файлі та розрахунок хешів для контролю їх цілісності. У

випадку, коли знаходиться некоректний блок, увесь ланцюжок створений після цього блока вважається некоректним і такі данні видаляються з файла і відбувається повторне завантаження блоків з потрібними порядковими номерами.

4.3 Проведення експерименту

Оскільки система є достатньо складною та багатокомпонентною, її випробування вимагає наявності достатньої кількості апаратних ресурсів, та розгортання програмних компонентів на кількох апаратних платформах. Виходячи з цього, було вирішено розділити експеримент на два етапи. Перший передбачає етап передбачає попереднє тестування системи на рівні імітаційної моделі, що дозволяє вчасно виправити помилки, які виникають в процесі роботи системи, і позбавить необхідності зайвий раз розгортати систему на реальній апаратній платформі. Окрім того моделювання є більш зручним для збору даних аніж запуск на реальному обладнанні. Другий етап передбачає запуск на реальній апаратній платформі, що дає можливість перевірити роботу системи в реальних умовах, а також впевнитися в достовірності результатів, що були отримані на етапі моделювання.

Перший етап випробування роботи системи вирішено було провести в віртуальному середовищі за допомогою програми GNS3, яка дозволяє створювати віртуальні мережі на основі віртуальних машин з достатньо складною топологією мереж, і окрім звичайних віртуальних машин, що представляють комп’ютери та сервери, підтримуються віртуальні машини для специфічних операційних систем маршрутизаторів, комутаторів, мережевих екранів та іншого мережевого обладнання. Окрім використання віртуалізації дана програма дозволяє використовувати контейнери Docker в якості елементів віртуальної мережі, що дозволяє знизити навантаження на хост систему, оскільки контейнеризація витрачає значно менше ресурсів при використанні Linux систем, оскільки контейнер не емулює роботу

комп’ютера, а використовує ядро хост системи для запуску процесів в ізольованому середовищі. Зазвичай можливостей контейнерів цілком достатньо, якщо немає необхідності використовувати графічний інтерфейс, і створювати більш громіздкі віртуальні машини для Linux систем немає сенсу.

Наша модель мережі відображає чотири локальні мережі, що відповідають SOHO та SMB сегменту, що підключені до глобальної мережі Інтернет. Кожна мережа використовує для підключення свій маршрутизатор. Для зменшення навантаження на систему віртуалізації мережі були змодельовані в спрощеному вигляді і містять лише кілька вузлів, оскільки кількість вузлів в локальних мережах мало впливає на роботу системи виявлення вторгнень.

Для уникнення конфлікту з реально існуючими IP адресами усі адреси для моделі були обрані з діапазонів адрес, які не маршрутизуються в Internet це адреси з діапазонів для локальних мереж (стандарт RFC1918) а також адреси для документації (стандарт RFC5737).

GlobalBorderRouter в даному випадку це обладнання інтернет провайдера через яке мережі підключаються до Internet, кінцеві користувачі (власники локальних мереж) впливу на нього не мають, і в даному випадку його особливості та конфігурація суттєвої ролі не грають. Cloud1 представляє собою глобальну мережу, і може бути джерелом шкідливого трафіка. До GlobalBorderRouter підключене комутаційне обладнання, до якого підключаються і локальні мережі, і інші вузли, які приймають участь в даному експерименті.

GlobalServer в даному випадку є вузлом системи виявлення вторгнень який використовується для підтримки роботи блокчейн підсистеми і не приймає участі у виявленні вторгнень, тому він базується на Linux системі і має можливість швидко надсилати відповіді великій кількості клієнтів.

NetX_BorderRouter – маршрутизатори через які локальні мережі підключаються до глобальної мережі. На них встановлена остання на даний момент версія операційної системи OpnSense версії 22. Маршрутизатор через NAT підключає локальну мережу до глобальної, і одночасно виступає в якості DHCP, Firewall та IDS. На всіх маршрутизаторах NetX_BorderRouter встановлено розподілену систему виявлення вторгнень на основі блокчейн. При запуску кожен з цих вузлів підключається до сервера GlobalServer, як до блокчейн вузла з відомою IP адресою, оскільки при першому запуску перелік доступних вузлів не відомий, і в наявності є лише невеликий список постійних вузлів, і завантажує з нього блоки та список доступних вузлів, до яких також намагається підключитись для завантаження блоків.

Для генерації шкідливого трафіку в нашій моделі мережі передбачена машина з встановленою операційною системою KaliLinux, яка містить великий набір утиліт для тестування на вразливості і генерування атакуючого трафіку.

Модуль виявлення вторгнень Suricata за замовчуванням має достатньо обмежену базу правил, які не реагують на більшу частину подій, тому базовий набір правил було розширено набором правил з відкритої бази спільноти Snort. Така база містить значно більше правил і дозволяє виявляти атаки сканування та DoS атаки.

Атака сканування виконувалася за допомогою утиліти nmap з налаштуванням на періодичне сканування усього діапазону портів. Така атака часто зустрічається в реальному середовищі оскільки за допомогою сканування зловмисники намагаються знайти відкриті порти на доступних через інтернет комп’ютерах, а потім шукають вразливі сервіси на відкритих портах, за допомогою яких можна отримати доступ до комп’ютера. Окрім того, такі атаки сканування можуть створювати додаткове навантаження на програмне забезпечення, що прослуховує ці порти.

Для виконання ж DoS атаки вирішено було використовувати фреймворк metasploit, який являє собою open source набір утиліт та експлоїтів, які використовуються для тестування систем на вразливості. Фреймворк побудований за модульним принципом і містить стандартизовані інтерфейси для налаштування експлоїтів, що дозволяє комбінувати експлоїти з пейлоадами, які будуть викликатися на віддаленій машині, якщо завантаження пейлоада підтримується експлоїтом. Серед великого набору експлоїтів в даному фреймворку є експлоїти спрямовані на tcp протокол, зокрема експлоїт для виконання TCP SYN Flood атаки, яка є одним із різновидів DoS атак. Основні налаштування для такого експлоїта – це кількість пакетів, що будуть відправлені – параметр NUM, а також адреса віддаленого хоста – параметр RHOST. Для автоматизованого виконання експлоїтів фреймворком можна використовувати конфігураційні файли, які містять опис експлоїтів, які необхідно виконати, їх параметри та команди запуску, це дає можливість автоматизувати певні сценарії тестування.

Оскільки нам потрібно було випробувати роботу системи протягом достатньо великого проміжку часу, ручне створення шкідливого трафіку програмами достатньо незручне, тому для таких цілей було створено скрипт, що дозволяє з певними проміжками часу виконувати надсилання шкідливих пакетів на ті чи інші вузли змодельованої мережі. Скрипт поділено на дві частини, перша написана мовою Python генерує конфігурацію для атаки набору хостів, друга є звичайним shell скриптом і відповідальна за періодичний запуск генерування конфігурації та запуск утиліти msfconsole.

Лістинг 4.9 – Вихідний код генератора конфігурації для metasploit

```
import random
targets =['203.0.113.250', '203.0.113.200']
print ("use auxiliary/dos/tcp/synflood")
print ("set NUM 500")
random.shuffle(targets)
```

```
    for i in targets:
        print("set RHOST "+ i)
      print("run")
    for i in targets:
        print("set RHOST "+ i)
      print("run")
    for i in targets:
        print("set RHOST "+ i)
      print("run")
    for i in targets:
        print("nmap -p- "+ i)
    print("exit")
```

Лістинг 4.10 – Вихідний код скрипта запуску генератора шкідливого трафіку

```
    #!/bin/bash
    for ((;;))
    do
    python gen.py >target.rc
    msfconsole -r target.rc
    sleep 10
    done
```

Для розгортання блокчейн підсистеми на OPNSense довелося виконати налаштування системи, які дають можливість встановити усе необхідне для запуску програмне забезпечення та бібліотеки, оскільки в базовому варіанті системи OPNSense репозиторії обмежені базовим набором пакетів, необхідних для роботи мережевого екрану а також включений спеціальний репозиторій, що містить усі інтерфейси налаштування для даного мережевого екрану. Код скрипта наведений в лістингу 4.11.

Лістинг 4.11 – Код скрипта встановлення додаткових пакетів

```
    #!/bin/sh
    cat /usr/local/etc/pkg/repos/FreeBSD.conf|sed -E 's/FreeBSD: \{ enabled:
no \}/FreeBSD: \{ enabled: yes \}/g' > /usr/local/etc/pkg/repos/FreeBSD.conf
    env ASSUME_ALWAYS_YES=YES pkg upgrade
    env ASSUME_ALWAYS_YES=YES pkg install git python leveldb py38-pip py38-
leveldb py38-yaml
    ln -s /usr/local/include/leveldb /usr/include/
    python3 -m pip install flask
```

Вузли, що забезпечують підтримку роботи блокчейн підсистеми мають можливість взяти на себе більше навантаження, якщо на них не покладена задача виявлення вторгнень, тоді кількість запитів на хвилину зростає. Вцілому, кількість запитів за хвилину зазвичай пропорційна до кількості активних з'єднань у блокчейн вузла, а також залежить від наявності чи відсутності шкідливого трафіка на підключених вузлах. Більш детальний розгляд захопленого під час випробування системи трафіку показує, що час обробки запита між вузлами залежить від його типу. Найшвидше виконуються запити на отримання коротких наборів даних, таких як наприклад список підключених вузлів, їх виконання займає близько двох мілісекунд, найбільш часті запити - запити на відправку, коли вузол пересилає транзакцію також виконуються достатньо швидко - виконання такого запиту займає в середньому 5мс, це пов'язане з тим, прийом транзакцій не вимагає виконання тривалих операцій читання або запису на носій. Найдовше виконуються запити повного ланцюжка, оскільки вони вимагають як звернення до бази, так і можливої переіндексації блоків. Виконання таких запитів може зайняти до однієї секунди в нашому випадку, але в реальних системах швидкість виконання буде сильно залежати від швидкодії апаратної платформи, на якій працює блокчейн підсистема.

Слід також зауважити, що такі дані були отримані у віртуальному середовищі, де швидкість мережевого обміну даними достатньо висока, а ping близький до 0. В реальних умовах такий результат отримати практично неможливо, особливо якщо вузли будуть підключені до глобальної мережі, а отже час виконання запитів буде сильно залежати від віддаленості конкретних вузлів один від одного.

Для того, щоб впевнитись, що система буде працювати нормально і в реальній мережі було вирішено створити тестову мережу зі схожою архітектурою в навчально тренувальній лабораторії Моделювання кіберзагроз, експериментальна мережа містила 5 вузлів об'єднаних в один

сегмент. Кожен з вузлів представляє сегмент мережі нижчого рівня, складові якої були створені шляхом віртуалізації, оскільки вузли розташовані за маршрутизатором практично не впливають на хід експерименту. KaliLinux для створення шкідливого трафіку також була встановлена на окремий комп'ютер. Така модель мережі цілком відповідає об'єднанню окремих мереж SMB та SOHO класів. Швидкість з'єднання по локальній мережі зазвичай виходить більша аніж при взаємодії через інтернет, а час проходження запиту менший, але такі умови більш наближені до реального середовища аніж мережа на основі віртуальних машин.

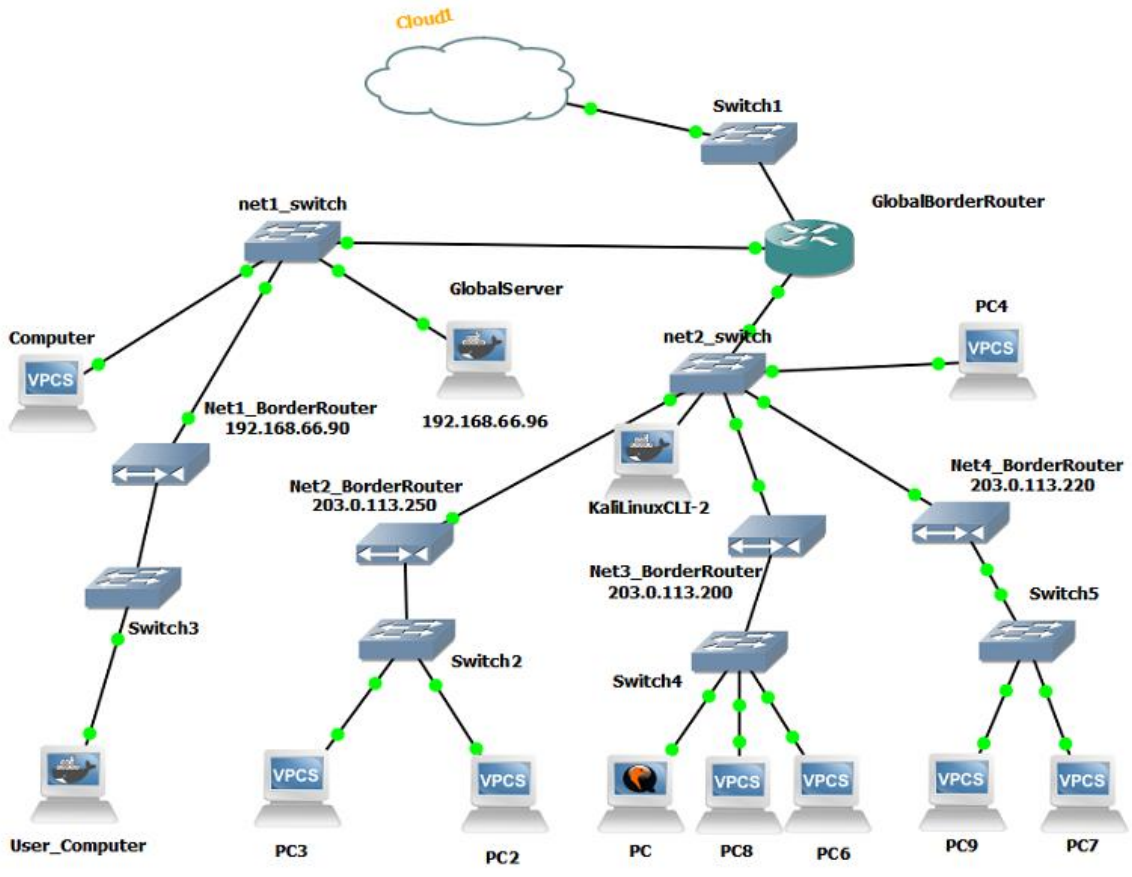


Рисунок 4.2 – Схема віртуальної мережі для тестування розподіленої системи виявлення вторгнень

Експеримент було поставлено аналогічним чином до віртуального середовища, але оскільки реалізувати захоплення трафіка на реальній мережі значно складніше, для оцінки поведінки в блокчейн підсистемі було увімкнене логування відправлених та прийнятих запитів.

Оскільки результати запуску системи на реальному обладнанні схожі на результати запуску в віртуальній моделі мережі, з врахуванням збільшення часу проходження мережевих пакунків, а захоплення трафіку значно простіше при використання віртуальної моделі мережі, подальше дослідження пакетів буде виконуватися на основі захопленого в віртуальній мережі трафіка.

4.4 Обробка результатів експерименту

Оскільки основна мета експерименту полягає в підтвердженні достатньої швидкодії блокчейн підсистеми розподіленої системи виявлення вторгнень для SOHO та SMB мереж. Також метою експерименту є оцінка швидкості обміну повідомленнями про можливі джерела шкідливого трафіку та вторгнень в цілях підготовки вузлів до блокування джерел шкідливого трафіку.

Отже, основними даними-результатами експерименту, що нас цікавлять є файли захоплених пакетів, на основі яких можна отримати інформацію про таймінги передачі повідомлень та блоків, на основі чого ми зможемо оцінити ефективність використання такої розподіленої системи виявлення вторгнень для застосування в SOHO та SMB мережах.

Першим етапом обробки набору захоплених пакетів є відсіювання шкідливого трафіку, який використовувався для тестування роботи системи, оскільки такий трафік ускладнює аналіз пакетів і сповільнює роботу програми. Разом із шкідливим трафіком можемо також відсіяти і інший трафік, який не стосується роботи нашої системи. Найбільш оптимальний

шлях в цьому випадку це встановити фільтр на пропускання тільки HTTP трафіку, оскільки основний обмін даними в розподіленій системі виявлення вторгнень на основі блокчейн відбувається на основі HTTP протоколу.

Далі слід скористатися більш точними фільтрами для виділення конкретних типів повідомлень, що передавалися між вузлами системи виявлення вторгнень.

Першим типом повідомлень, що нас цікавить, є повідомлення, які передають транзакції. Саме цей тип повідомлень відповідальний за оперативну передачу даних щодо виявлених джерел шкідливого або підозрілого трафіку. Для транзакцій, що містять події згенеровані модулем виявлення вторгнень у нас є кілька доступних часових міток, які допоможуть нам визначити середній час доставки транзакції і системі.

SolarWind Response time viewer дає нам наступну статистику по використання мережевих ресурсів системою під час роботи:

- Середній час відгуку мережі: 1.4мс, максимальний 194мс
- Середній час відгуку програми 43мс, максимальний 830мс
- Середній потік даних 45 КБайт/с, загальний об’єм даних за годину роботи 20мб
- Середній об’єм запитів 15 запитів за хвилину.

Перша часова мітка міститься в даних події створеної підсистемою виявлення вторгнень, і вказує така мітка на час виявлення тої чи іншої події. Звичайно така часова мітка не буде абсолютно відповідати часу надходження шкідливого трафіка, оскільки виявлення також може відбуватися з певною затримкою, але фактично це найбільш рання часова мітка, яку ми можемо зафіксувати, оскільки такі засоби як мережевий екран хоча і дадуть можливість зафіксувати час з меншою затримкою, не дають можливості настільки детально фільтрувати та класифікувати трафік, виділяючи шкідливі та підозрілі події.

Наступна часова мітка відповідає часу створення транзакції блокчейн підсистемою. На цьому етапі транзакція вже повністю сформована, записана в локальний тимчасовий пул і передається сусіднім вузлам розподіленої системи виявлення вторгнень, з якими встановлено з'єднання.

При створенні HTTP пакета йому також присвоюється часова мітка. Вона має бути максимально наближеною до часової мітки транзакції, але часто через розсинхронізацію годинників вузла, що виконує захоплення трафіка та вузла, на якому запущено блокчейн підсистему, час може відрізнятися на певне фіксоване значення.

$$t_{заг} = t_{tran} - t_{IDS} + \sum_{i=1}^{n} (t_{res_i} - t_{req_i}) / 2 \qquad (17)$$

де $t_{заг}$ – загальний час проходження транзакції, *с;* t_{tran} - час створення транзакції,; t_{IDS} -*час виявлення події системою виявлення;* t_{res} - час надходження відповіді на запит на кроці *i*, t_{req} -час надсилання запиту на кроці *i*, *n* - кількість кроків необхідних для передачі транзакції.

Остання часова мітка це час надходження відповіді на HTTP пакет – така мітка вказує на те, що транзакція була прийнята і надійшла на обробку до іншого вузла. Сумарні часові проміжки виділені такими часовими мітками складають повний час проходження пакета до сусіднього вузла.

Сформувавши перелік потрібних нам часових міток перенесемо їх до CSV файла для подальшої статистичної обробки, оскільки збереження даних в такому форматі дає можливість імпортувати їх в більшість доступних засобів статистичної обробки.

Для обробки скористаємося системою інтерактивних блокнотів Jupyter та мовою Python. Попередній аналіз мережевих пакетів показує, що середній час відгуку мережі: 3.5мс, максимальний 206мс, а середній час відгуку програми 51мс, максимальний 847мс.

Отже, виходячи з результатів обробки експериментальних даних, отримуємо, що транзакція, яка містить інформацію про шкідливу подію,

створюється в середньому через 0.8 секунди після виявлення такої події, враховуючи додаткове навантаження на систему, що виникало під час експерименту через DoS атаку. Час доставки ж транзакції сильно залежатиме від віддаленості вузлів один від одного, швидкості роботи та затримок в мережі, але в середньому для мережі Інтернет така затримка складатиме кілька десятків мілісекунд для передачі між двома вузлами. Відсутність же прямого з'єднання між двома вузлами збільшить цей час пропорційно до кількості проміжних вузлів тоді цей час може збільшитись до кількох секунд. При цьому передачі транзакції не записаної в блок достатньо для того щоб повідомити інші вузли мережі про можливі джерела шкідливого трафіку.

Отже, в найкращому випадку повідомлення сусідніх вузлів учасників розподіленої системи виявлення вторгнень на основі блокчейн займе не більше секунди. В ситуаціях ж з повільним або нестабільним мережевим з'єднанням та великою віддаленістю вузлів час передачі рідко перевищуватиме 5 секунд. Отримати оновлення актуальних баз для системи виявлення вторгнень із стандартних джерел за такий час не можливо, оскільки зазвичай баз для систем виявлення вторгнень не мають такої частоти оновлень, а отже і не дадуть оперативно підготуватися до можливої загрози. Ручне внесення правил в бази за такий короткий час також неможливе, а отже розподілена система виявлення вторгнень дозволяє оперативно підготувати мережу, що знаходиться під її захистом до можливої атаки. Практика показує, що враховуючи час реакції людини, аналіз ситуації та час прийняття рішення, повідомлення про можливе вторгнення в ручному режимі займе в більшості випадків не менш ніж пів хвилини, і окрім того вимагатиме скоординованих дій системних адміністраторів відповідальних за захист кожної з мереж. Більш детальне порівняння різних етапів реагування на кіберзагрози при виконанні їх системним адміністратором ручному режимі та при використанні розподіленої системи на основі блокчейн наведено в таблиці.

Таблиця 4.1 – Порівняння швидкості реагування для локальної та розподіленої системи захисту комп’ютерних мереж малих та середніх підприємств

Етап	Час реагування для локальної системи		Час реагування для розподіленої системи	
	Min	Max	Min	Max
Виявлення зловмисних дій в мережі	~1с	~10хв	~1с	~10хв
Аналіз події	~5хв	~30хв	~100мс	<10с
Відправка інформації про подію іншим учасникам	~1хв	30хв	~5мс	~30с
Застосування змін іншими учасниками	2хв	30хв	~51мс	~0,7с
Всього	8хв	~1.5 год	<4с	<11хв

Але тут слід звернути увагу на те, що частина таблиці, яка відображає значення часу для ручного режиму побудована на основі експертних оцінок, і в залежності від ситуації значення можуть відрізнятися. Окрім того, для обох випадків етап виявлення передбачає як використання технічних засобів для виявлення підозрілих подій, та вторгнень в мережі, так і ручне виявлення підозрілої активності, спираючись на аномальні режими роботи апаратного та програмного забезпечення.

При цьому дані передані в транзакціях є тимчасовими і не будуть передані вузлам які не були підключені до мережі на момент передачі транзакції. Постійно записаними даними є лише транзакції, що записані в блоки. Проте формування блоку займає певний час – повинна накопичитися потрібна кількість транзакцій, а після цього вузли повинні сформувати повноцінний блок який буде відповідати протоколу консенсусу. Виходячи з

отриманих в результаті експерименту даних середній час створення блоку не перевищує 10 хвилин, але при цьому слід мати на увазі, що вузли, які приймали участь в експерименті, генерували невелику кількість транзакцій, що також могло призвести до затримок в генеруванні нових блоків. В рамках даного експерименту для створення блоку були задані наступні параметри: мінімальна кількість транзакцій починаючи з якої ідуть спроби створення блоку – 5 транзакцій, оскільки при меншій кількості створення блоку не має сенсу, окрім того мала кількість транзакцій сповільнює пошук потрібного хеша, максимальна ж кількість прийнята за 240 транзакцій, щоб не перевантажувати блоки та зробити їх зручними для відправки. В реальній же системи параметри можна уточнювати в залежності від потреб та завантаженості системи. В даному випадку встановлений за допомогою параметра максимального значення ставки період сприятливий для генерації нового блока наступає один раз на дві хвилини, і збільшення кількості вузлів, які приймають участь у генеруванні блоків наближують час створення блока до двох хвилин і менше.

$$t_{blk} = \frac{n}{\sum_{i=0}^{n}\left(\frac{N_{t\arg}(i)}{N_{\max}}\right)\times k} \qquad (18)$$

де t - середній час створення блоку, n – максимальне значення ставки часу, N_{targ} – кількість цільових варіантів хеша в момент часу i, N_{max} – загальна кількість варіантів хеша, k – кількість вузлів. І хоча теоретичні розрахунки показують зростання швидкості створення нових блоків, це не означає, що швидкість нарощування ланцюжка зростатиме так стрімко, оскільки збільшення кількості вузлів призводить до збільшення кількості конфліктів і паралельних ланцюжків, звідки слідує що більша частина блоків при цьому буде в подальшому відкинута. Проміжок часу порядку кількох хвилин достатньо значний в даному контексті і таке повідомлення не можна назвати оперативним з одного боку, але з іншого боку повідомленнями записаними в

блоки користуватимуться лише ті учасники розподіленої системи виявлення вторгнень, які з тої чи іншої причини пропустили повідомлення з транзакцією і скоріше за все на момент його створення були оффлайн, а тому затримка на кілька хвилин для них буде не настільки принциповою і все одно дасть можливість підготуватися до можливого вторгнення.

Висновки до розділу 4

За результатами розробки окремих модулів інформаційної технології забезпечення кібербезпеки SOHO та SMB мереж на основі blockchain та проведення експерименту можна зробити наступні висновки.

1. Визначені виходячи з особливостей комп'ютерних мереж малих та середніх підприємств параметри для розподіленої системи виявлення вторгнень на основі блокчейна дають можливість адаптувати її роботу для мережах такого класу з метою зниження споживання системних ресурсів та досягнення оптимального рівня виявлення кіберзагроз.
2. Розроблена з використанням мови програмування Python імітаційна модель розподіленої системи захисту комп'ютерних мереж на основі блокчейн технології, що відображає роботу основних компонентів системи, зокрема блокчейн компонента та класифікатора дає можливість тестувати роботу розподіленої системи захисту за різних умов та при використанні різного мережевого обладнання.
3. Тестування роботи компонентів розподіленої системи захисту комп'ютерних мереж в віртуальному середовищі GNS3 та на реальному обладнанні показало, що компоненти відповідають базовим вимогам для використання в розподіленій системі виявлення вторгнень для комп'ютерних мереж малих та середніх підприємств на основі блокчейн.

4. Оцінка швидкодії системи показала, що розподілена система захисту комп’ютерних мереж на основі блокчейн дозволяє значно пришвидшити поширення та застосування актуальних оновлень правил та вчасно розповсюджувати між усіма своїми вузлами інформацію про шкідливий або підозрілий трафік, що в свою чергу дозволяє посилити захист мереж малих та середніх підприємств.

Результати досліджень приведених в розділі опубліковані в роботах [77, 124]

ВИСНОВКИ

В результаті виконання дисертаційного дослідження сформульовано та вирішено актуальне наукове завдання щодо створення інформаційної технології виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain.

Актуальність дослідження підтверджується ретельним аналізом публікацій та інформаційних джерел, який виявив необхідність формування методів та моделей інформаційної технології виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain з врахуванням типів використовуваного в них обладнання, а також основних загроз, які становлять найбільшу небезпеку, для мереж такого класу.

За результатами дослідження можна зробити наступні висновки:

1) Результати аналізу основних загроз інформаційній безпеці комп'ютерних мереж малих та середніх підприємств дозволили визначити існуючі методи та засоби їх захисту з врахуванням особливостей функціонування та експлуатації мереж такого класу, що дає можливість створювати нові та покращувати існуючі методи захисту.
2) Розроблена модель розподіленої інформаційної системи виявлення та аналізу аномальних подій на основі блокчейн технології є основою для побудови різноманітних розподілених систем захисту комп'ютерних мереж малих та середніх підприємств.
3) Запропонований метод вибору протоколу консенсусу дозволив обрати для розподіленої інформаційної системи виявлення та аналізу аномальних подій протокол, який базується на методі PoS, що дозволяє знизити навантаження на обчислювальне обладнання в процесі роботи системи, а також зменшити споживання електроенергії. При цьому вимоги до обчислювальної потужності

обладнання знижуються, що робить систему придатною для використання в комп'ютерних мережах малих та середніх підприємств.

4) Адаптований для використання в розподілених системах виявлення вторгнень протокол консенсусу PoS забезпечує надійну та стабільну роботу blockchain компонента та витрачає менше обчислювальних ресурсів аніж базовий варіант протоколу на 80%.
5) На основі результатів аналізу методів класифікації аномальних подій сформований перелік основних класифікаторів, які об'єднуються в комплексний класифікатор для підвищення точності виявлення невідомих аномальних подій розподіленою системою виявлення вторгнень.
6) Розроблена функціональна модель та архітектура розподіленої системи захисту комп'ютерних мереж на основі blockchain забезпечує високий ступінь децентралізації та можливість масштабування в умовах обмежених обчислювальних ресурсів.
7) Побудовані UML діаграми основних компонентів розподіленої системи захисту комп'ютерних мереж на основі blockchain дали можливість деталізувати як зв'язки між компонентами системи, так і будову самих компонентів та варіанти їх розгортання в комп'ютерних мережах.
8) Реалізація запропонованих моделей та методів в розподіленій системі захисту комп'ютерних мереж малих та середніх підприємств дозволила отримати збільшення швидкості реагування розподіленої системи захисту комп'ютерних мереж на невідомі загрози у порівнянні з локальною системою захисту майже в 10 разів.

Практичне значення отриманих результатів полягає в тому, що наведені вище наукові результати у своїй сукупності утворюють нову

інформаційну технологію виявлення та аналізу аномальних подій для захисту комп’ютерних мереж на основі blockchain. Запропонована інформаційна технологія може бути використана як розробниками систем захисту комп’ютерних мереж, так і мережевими адміністраторами та ІБ спеціалістами малих та середніх підприємств. Розроблені бізнес процеси та архітектура є основою для розробки більш потужних та функціональних розподілених систем виявлення вторгнень.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] "Закон України «Про основні засади забезпечення кібербезпеки України» від 5 жовтня 2017 року № 2163-VIII," [Online]. Available: https://zakon.rada.gov.ua/laws/show/2163-19#Text.

[2] W. Stallings, Cryptography and Network Security: Principles and Practice, 8th edition ed., Pearson; , 2019, p. 832.

[3] R. Shirey, " Internet Security Glossary, Version 2," Network Working Group, August 2007. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc4949. [Accessed 20 10 2023].

[4] "Security architecture for Open Systems Interconnection for CCITT applications," ITU Telecommunication Standardization Sector (ITU-T), 22 03 1996. [Online]. Available: https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=3102&lang=en. [Accessed 2023].

[5] Farhana Haque Sumi, Lokesh Dutta and Farhana Sarker, "A Review on Cyberattacks and Their Preventive Measures," *International Journal of Cyber Research and Education (IJCRE),* vol. 1, no. 2, pp. 12-29, 01 07 2019.

[6] Thanh Vu, S. N., Stege, M., El-Habr, P. I., Bang, J., & Dragoni, N., "A survey on botnets: Incentives, evolution, detection and current trends," *Future Internet,* vol. 13, p. 198, 2021.

[7] Kumar, S., Pathak, S. K., & Singh, J., "A Comprehensive Study of XSS Attack and the Digital Forensic Models to Gather the Evidence," *ECS Transactions,* vol. 107, 2022.

[8] Aryaman, Naik, N. S., Vollala, S., & Amin, R., "Detecting and predicting countermeasures against clickjacking," *Security and Privacy,* vol. 6, no. 5, 2023.

[9] І. Г. Бондарєв, Телекомунікаційні системи та мережі. Принципи функціонування, технології та протоколи, Львів: Львівська політехніка, 2016.

[10] "Господарський кодекс України 436-IV редакція від 01.01.2024," [Online]. Available: https://zakon.rada.gov.ua/laws/show/436-15#Text.

[11] "SME definition," European Comission, 2021. [Online]. Available: https://single-market-economy.ec.europa.eu/smes/sme-definition_en. [Accessed 20 01 2024].

[12] "Why Cybersecurity Matters More Than Ever For SMBs in 2022," Leading IT, 04 02 2022. [Online]. Available: https://www.goleadingit.com/why-cybersecurity-matters-more-than-ever-for-smbs-in-2022/.

[13] "The State of Cloud Security 2021 Report," Fugue +, 2021. [Online]. Available: https://resources.fugue.co/state-of-cloud-security-2021-report. [Accessed 26 02 2023].

[14] C. Insiders, "2021 CLOUD SECURITY REPORT," 2021.

[15] C. Insiders, "2023 Cloud Security Report," 2023.

[16] D. Emm, "IT threat evolution Q3 2023," Securelist, 01 12 2023. [Online]. Available: https://securelist.com/it-threat-evolution-q3-2023/111171/.

[17] Awais Rashid, Howard Chivers, George Danezis, Emil Lupu, Andrew Martin, CyBOK: Cyber Security Body of Knowledge, National Cyber Security Centre, 2019.

[18] Van Niekerk, J. F., Von Solms, R., "Information security culture: A management perspective.," *Computers & Security,* vol. 29(4), p. 476–486, 2010.

[19] "SMB Networking Environments and Solutions Design Considerations," Cisco Press, 8 10 2004. [Online]. Available: https://www.ciscopress.com/articles/article.asp?p=342887&seqNum=2.

[20] V. Lytvynov, N. Stoianov, I. Stetsenko, I. Skiter, O. Trunova, A. Hrebennyk, V. Nekhai, I. Burmaka, Attacks defense of computer nets by tools using extended information about environment : monograph, Chernihiv : Chernihiv Politechnic National University , 2021. - 212c..

[21] Alvarez, J. R. N., Zamora, Y. P., Pina, I. B., & Angarita, E. N., "Demilitarized network to secure the data stored in industrial networks.," *International Journal of Electrical & Computer Engineering*, vol. 11, 2021.

[22] "What is a DMZ Network?," [Online]. Available: https://www.fortinet.com/resources/cyberglossary/what-is-dmz.

[23] Kwon, W. C., Lee, J. K., & Kim, K. H., "Application of access control policy in ScienceDMZ-based network configuration.," *Convergence Security Journal*, vol. 21, pp. 3-10, 2021.

[24] Rababah, B., Zhou, S., & Bader, M., "Evaluation the Performance of DMZ.," *International Journal of Wireless and Microwave Technologies*, vol. 1, 2018.

[25] Skiter, I., Burmaka, I., & Sigayov, A., "Design of Technical Methods for Analysing Network Security Based on Identification of Network Traffic Anomalies," *Information & Security*, vol. 47, no. 3, pp. 306-316, 2020.

[26] N. H. Nguyen, Essential Cyber Security Handbook In Ukrainian, 2018.

[27] D. K. James Michael Stewart, Network Security, Firewalls, and VPNs, Jones & Bartlett Learning, 2020.

[28] Wes Noonan, Ido Dubrawsky, Firewall Fundamentals, Cisco Press, 2006.

[29] H. Andrea, Cisco ASA Firewall Fundamentals, Amazon Digital Services, 2014.

[30] "IPFilter (IPF)," OmniOS, [Online]. Available: https://omnios.org/info/ipfilter?. [Accessed 11 12 2023].

[31] Peter N. M. Hansteen, The Book of PF: A No-Nonsense Guide to the OpenBSD Firewall, No Starch Press, 2008.

[32] Jingyao, S., Chandel, S., Yunnan, Y., Jingji, Z., & Zhipeng, Z., "Securing a network: how effective using firewalls and VPNs are?," *Advances in Information and Communication: Proceedings of the 2019 Future of Information and Communication Conference (FICC)*, vol. 2, 2020.

[33] Krishna, T. V., & Karthik, P., "Dominance of Hardware Firewalls and Denial of Firewall Attacks (Case Study BlackNurse Attack)," *International Journal of Science and Research (IJSR)*, vol. 11, pp. 28-33, 2022.

[34] Teng, L., Hung, C. H., & Wen, C. H. P., "P4SF: A high-performance stateful firewall on commodity P4-programmable switch.," *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, 2022.

[35] M. Rash, Linux Firewalls: Attack Detection and Response with iptables, psad, and fwsnort, No Starch Press, 2027.

[36] І. Бурмака, "Класифікація систем виявленя вторгнень в розподілені інформаційні системи".*Проблеми зняття з експлуатації об'єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 17): збірник матеріалів IV Міжнародної конференції (25–27 квітня 2017, м. Славутич). – Чернігів : ЧНТУ, 2017. – с. 59-63.*

[37] Pradhan, M., Nayak, C. K., & Pradhan, S. K., "Intrusion detection system (IDS) and their types.," *Securing the internet of things: Concepts, methodologies, tools, and applications* , pp. 481-497, 2020.

[38] "SolarWinds Security Event Manager Features," Solarwoinds, [Online]. Available: https://www.solarwinds.com/security-event-manager/use-cases. [Accessed 25 06 2023].

[39] "About Zeek," Zeek, 2022. [Online]. Available: https://zeek.org/about/. [Accessed 20 01 2024].

[40] Tiwari, A., Saraswat, S., Dixit, U., & Pandey, S., "Refinements In Zeek Intrusion Detection System," *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS),* vol. 1, pp. 974-979, 2022.

[41] "What are Community Rules?," Snort.org, [Online]. Available: https://www.snort.org/faq/what-are-community-rules. [Accessed 12 11 2023].

[42] Waleed Abdul, Abdul Fareed Jamali, and Ammar Masood., "Which open-source ids? snort, suricata or zeek.," *Computer Networks,* vol. 213, 2022.

[43] "Suricata Documentation," The Open Information Security Foundation , 2023. [Online]. Available: https://suricata.io/documentation/.

[44] "McAfee Network Security Platform," Trellix, 16 02 2023. [Online]. Available: https://docs.trellix.com/bundle/network-security-platform-10.1.x-product-guide/page/GUID-373C1CA6-EC0E-49E1-8858-749D1AA2716A.html.

[45] "The NIST Cybersecurity Framework 2.0," NIST, 8 08 2023. [Online]. Available: https://csrc.nist.gov/pubs/cswp/29/the-nist-cybersecurity-framework-20/ipd.

[46] "Netgate Appliances," Netgate, [Online]. Available: https://www.netgate.com/appliances. [Accessed 20 01 2024].

[47] David Zientara, pfSense 2.x Cookbook: Manage and maintain your network using pfSense, 2nd Edition, Packt Publishing, 2018.

[48] "Fortinet Inc. Annual Report 10K," Fortinet.com, 24 02 2023. [Online]. Available: https://investor.fortinet.com/node/23411/html.

[49] E. Messmer, "Fortinet unveils high-speed security blade for Fortigate-5000 chassis; FortiGate-5001B blade combines firewall, VPN, IPS, application controls, filtering," *Network World,* 30 11 2010.

[50] N. Eddy, "Fortinet Releases Internal Network Firewall," eWeek, 23 01 2015. [Online]. Available: https://www.eweek.com/small-business/fortinet-releases-internal-network-firewall/. [Accessed 7 11 2023].

[51] N. Eddy, "Fortinet Security Platform Hits Amazon Web Services," eWeek, 27 03 2014. [Online]. Available: https://www.eweek.com/small-business/fortinet-security-platform-hits-amazon-web-services/. [Accessed 16 10 2023].

[52] Chou Dylan, and Meng Jiang, "A survey on data-driven network intrusion detection.," *ACM Computing Surveys (CSUR),* vol. 54.9, pp. 1-36.

[53] M. Lewis, "Cisco ASA series part one: Intro to the Cisco ASA," nccgroup, 20 09 2017. [Online]. Available: https://research.nccgroup.com/2017/09/20/cisco-asa-series-part-one-intro-to-the-cisco-asa/. [Accessed 17 08 2023].

[54] "Cisco Secure Firewall ASA," Cisco, [Online]. Available: https://www.cisco.com/c/en/us/products/security/adaptive-security-appliance-asa-software/index.html. [Accessed 09 01 2024].

[55] "Cisco Firepower 2100 Series," Cisco, [Online]. Available: https://www.cisco.com/site/us/en/products/security/firewalls/firepower-2100-series/index.html#tabs-ca9b217826-item-9e6cde6a19-tab. [Accessed 10 12 2023].

[56] L. Lamport, "The part-time parliament," *ACM Transactions on Computer Systems,* vol. 16, no. 2, pp. 133-169, 1998.

[57] Narayanan A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S., Bitcoin and cryptocurrency technologies: a comprehensive introduction, Princeton University Press, 2016.

[58] N. S., "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf. [Accessed 14 06 2023].

[59] M. Swan, Blockchain: Blueprint for a new economy, O'Reilly Media, 2015.

[60] "Secure Hash Standard (SHS), FEDERAL INFORMATION PROCESSING STANDARDS 180-4," Information Technology Laboratory Gaithersburg, 2015. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.180-4.

[61] "Hash Functions," Information Technology Laboratory NIST, 04 01 2017. [Online]. Available: https://csrc.nist.gov/projects/hash-functions. [Accessed 03 09 2023].

[62] " SHA-3 Standard: Permutation Based Hash and Extendable-Output Functions, Federal Information Processing Standards (FIPS) Publication 202," National Institute of Standards and Technology, 08 2015. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf.

[63] "Lightweight Directory Access Protocol," [Online]. Available: https://ldap.com/. [Accessed 11 10 2023].

[64] Yaga, D., Mell, P., Roby, N., & Scarfone, K., "Blockchain technology overview," *arXiv preprint arXiv:1906.11078*, 2019.

[65] Meng, W., Li, W., Yang, L. T., & Li, P., "Enhancing challenge-based collaborative intrusion detection networks against insider attacks using blockchain.," *nternational Journal of Information Security*, vol. 19, pp. 279-290, 2020.

[66] Yli-Huumo J., Ko, D., Choi, S., Park, S., & Smolander, K., "Where is current research on blockchain technology?—a systematic review," *PloS one*, vol. 11(10), pp. 1-27, 2016.

[67] Wang Luqin, and Yong Liu., "Exploring miner evolution in bitcoin network," *Passive and Active Measurement: 16th International Conference, PAM 2015, New York, NY, USA, March 19-20, 2015, Proceedings 16. Springer International Publishing* , pp. 290-302, 2015.

[68] Kolokotronis, N., Brotsis, S., Germanos, G., Vassilakis, C., & Shiaeles, S., " On blockchain architectures for trust-based collaborative intrusion detection.," *In 2019 IEEE world congress on services (SERVICES)*, vol. 2642, pp. 21-28, 2019.

[69] Li, W., Tug, S., Meng, W., & Wang, Y., "Designing collaborative blockchained signature-based intrusion detection in IoT environments.," *Future Generation Computer Systems*, vol. 96, pp. 481-489, 2019.

[70] Fung, C. J., Baysal, O., Zhang, J., Aib, I., & Boutaba, R., "Trust management for host-based collaborative intrusion detection," *In Managing Large-Scale Service Deployment: 19th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, DSOM 2008, Samos Island, Greece, September 22-26, 2008.Proceedings 19. Springer Berlin Heidelberg*, pp. 109-122, 2008.

[71] Li, W., Meng, W., & Kwok, L. F., "Design of intrusion sensitivity-based trust management model for collaborative intrusion detection networks," *Trust Management VIII: 8th IFIP WG 11.11 International Conference, IFIPTM 2014, Singapore, July 7-10, 2014. Springer Berlin Heidelberg.*, vol. Proceedings 8, pp. 61-67, 2014.

[72] Saravanan, V., Madiajagan, M., Rafee, S. M., Sanju, P., Rehman, T. B., & Pattanaik, B., "IoT-based blockchain intrusion detection using optimized recurrent neural network.," *Multimedia Tools and Applications*, 2023.

[73] Bediya, A. K., & Kumar, R., "A novel intrusion detection system for internet of things network security," *Bediya, Arun Kumar, and Rajendra Kumar. "A novel intrusion detection system for internet of things network security." Research Anthology on Convergence of Blockchain, Internet of Things, and Security. IGI Global,* pp. 330-348, 2023.

[74] Kably, S., Benbarrad, T., Alaoui, N., & Arioua, M., "Multi-Zone-Wise Blockchain Based Intrusion Detection and Prevention System for IoT Environment.," *Computers, Materials & Continua,* vol. 75, no. 1, 2023.

[75] Al-E'mari, S., Anbar, M., Sanjalawe, Y., Manickam, S., & Hasbullah, I., "Intrusion Detection Systems Using Blockchain Technology: A Review, Issues and Challenges.," *Computer Systems Science & Engineering,* vol. 40, no. 1, 2022.

[76] І. А. Бурмака, "Архітектура розподіленої системи виявлення вторгнень на основі blockchain технології".*Проблеми зняття з експлуатації об'єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 2020) в режимі онлайн: збірник матеріалів V Міжнародної конференції (27–29 квітня 2020, м. Славутич). – Чернігів : ЧНТУ, 2020. с.54-59.*

[77] Burmaka, I., Dorosh, M., Skiter, I., & Lytvyn, S, "Architecture of Distributed Blockchain Based Intrusion Detecting System for SOHO Networks," *Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MODS, 2021 June 28–July 01, Chernihiv, Ukraine. Springer Nature,* pp. 313-326, 2021.

[78] Burmaka, I., Zlobin, S., Lytvyn, S., & Nekhai, V., "Detecting flood attacks and abnormal system usage with artificial immune system," *Mathematical Modeling and Simulation of Systems: Selected Papers of 14th International*

Scientific-Practical Conference, MODS, 2019 June 24-26, Chernihiv, Ukraine, pp. 131-143, 2019.

[79] J. Brownlee, Clever Algorithms: Nature-Inspired Programming Recipes, Lulu.com, 2011.

[80] Okada, H., Yamasaki, S., & Bracamonte, V., "Proposed classification of blockchains based on authority and incentive dimensions," *19th international conference on advanced communication technology (ICACT). IEEE*, pp. 593-597, 2017.

[81] Wüst, K., & Gervais, A., "Do you need a blockchain?," *2018 crypto valley conference on blockchain technology (CVCBT) IEEE*, pp. 45-54, 2018.

[82] M. Vukolić, "Rethinking permissioned blockchains.," *Proceedings of the ACM workshop on blockchain, cryptocurrencies and contracts*, pp. 3-7, 2017.

[83] Meng, W., Tischhauser, E. W., Wang, Q., Wang, Y., & Han, J. , "When intrusion detection meets blockchain technology: a review.," *Ieee Access, 6, 10179-10188.*, 2018.

[84] Burmaka, I., Stoianov, N., Lytvynov, V., Dorosh, M., & Lytvyn, S., "Proof of stake for blockchain based distributed intrusion detecting system.," *Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MODS, 2020 June 29–July 01, Chernihiv, Ukraine Springer Nature.*, vol. 1265, pp. 237-247, 2020.

[85] Castro, Miguel, and Barbara Liskov., "Practical byzantine fault tolerance.," *OsDI.*, vol. 99, pp. 173-186, 1999.

[86] Cachin, Christian, Simon Schubert, and Marko Vukolić., "Non-determinism in byzantine fault-tolerant replication.," arXiv preprint arXiv:1603.07351., 2016. [Online]. Available: https://arxiv.org/pdf/1603.07351.pdf(Access.

[87] Ren, Y., Liu, Y., Ji, S., Sangaiah, A. K., & Wang, J., "Incentive mechanism of data storage based on blockchain for wireless sensor networks," *Mobile Information Systems*, 2018.

[88] Lindsey, S., & Raghavendra, C. S., "Power-efficient gathering in sensor information systems," *Proceedings, IEEE aerospace conference,* vol. 3, p. 3, 2002.

[89] Mao, J., Zhang, Y., Li, P., Li, T., Wu, Q., & Liu, J., "A position-aware Merkle tree for dynamic cloud data integrity verification," *Soft Computing*, vol. 21, 2017.

[90] Gupta, Y., Shorey, R., Kulkarni, D., & Tew, J., "The applicability of blockchain in the Internet of Things.," *10th International Conference on Communication Systems & Networks (COMSNETS) IEEE.*, pp. 561-564, 2018.

[91] A. Antonopoulos, Mastering Bitcoin: Programming the Open Blockchain 2nd Edition, O'Reilly Media, 2017.

[92] "Standard agent contacts network," Anylogic, [Online]. Available: https://anylogic.help/anylogic/agentbased/connections-and-networks.html. [Accessed 26 04 2023].

[93] "What Is The Bitcoin Block Size Limit?," Bitcoin Magazine, 4 7 2022. [Online]. Available: https://bitcoinmagazine.com/guides/what-is-the-bitcoin-block-size-limit. [Accessed 05 11 2022].

[94] Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., & Capkun, S., "On the security and performance of proof of work blockchains.," *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* , pp. 3-16, 2016.

[95] S. King, "Primecoin: Cryptocurrency with prime number proof-of-work.," July 7th 2013.

[96] Zhang, F., Eyal, I., Escriva, R., Juels, A., & Van Renesse, R., "Resource-Efficient Mining for Blockchains," *26th USENIX Security Symposium (USENIX Security 17)*, pp. 1427-1444.

[97] Tosh, D. K., Shetty, S., Liang, X., Kamhoua, C., & Njilla, L, "Consensus protocols for blockchain-based data provenance: Challenges and opportunities.," *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, pp. 469-474, 2017.

[98] Wendl, M., Doan, M. H., & Sassen, R., "The environmental impact of cryptocurrencies using proof of work and proof of stake consensus algorithms: A systematic review.," *ournal of Environmental Management*, vol. 326, 2023.

[99] V. P., "Blackcoins proof-of-stake protocol v2," Blackcoin, 2014. [Online]. Available: https://blackcoin.co/blackcoin-pos-protocol-v2-whitepaper.pdf. [Accessed 10 11 2021].

[100] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *Self-Published Paper*, vol. 19, 2012.

[101] Chen, L., Xu, L., Shah, N., Gao, Z., Lu, Y., & Shi, W., "On security analysis of proof-of-elapsed-time (poet)," *Stabilization, Safety, and Security of Distributed Systems: 19th International Symposium, SSS 2017, Boston, MA, USA, November 5–8, 2017, Proceedings 19*, pp. 282-297, 2017.

[102] Menon, A. A., Saranya, T., Sureshbabu, S., & Mahesh, A. S., "A Comparatıve Analysis on Three Consensus Algorithms: Proof of Burn, Proof of Elapsed Time, Proof of Authority," *Computer Networks and Inventive Communication Technologies: Proceedings of Fourth ICCNCT 2021*, pp. 369-383, 2022.

[103] D. Batmunkh, "Private blockchain consensus mechanisms," Medium, 23 11 2018. [Online]. Available: https://medium.com/@arigatodl/private-blockchain-consensus-mechanisms-8e6fc48c8fb. [Accessed 2024].

[104] "Bitshares - your share in the decentralized exchange," Bitshares, [Online]. Available: https://bitshares.org/whitepaper/. [Accessed 01 07 2023].

[105] D. Mazieres, "The stellar consensus protocol: A federated model for internet-level consensus," *Stellar Development Foundation,* vol. 32, 2015.

[106] Singh, A., Kumar, G., Saha, R., Conti, M., Alazab, M., & Thomas, R., "A survey and taxonomy of consensus protocols for blockchains.," *Journal of Systems Architecture,* vol. 127, 2022.

[107] A. T. Inc, "Consensus protocol for permissioned blockchain networks," Medium, 07 07 2020. [Online]. Available: https://accubits-tech.medium.com/consensus-protocol-for-permissioned-blockchain-networks-2d9d842659a. [Accessed 2024].

[108] Dylan Yaga, Peter Mell, Nik Roby, Karen Scarfone, "Blockchain Technology Overview," *NISTIR 8202 https://doi.org/10.6028/NIST.IR.8202,* p. 68, 10 2018.

[109] I. Burmaka, "Consensus algorithm comparison for blockchain based intrusion detecting system".*Безпека ресурсів інформаційних систем : збірник тез І Міжнародної науково-практичної конференції(м. Чернігів 16-17 квітня 2020р.). –Чернігів : НУЧП, 2020. –с.6-14.*

[110] Xiao, Y., Zhang, N., Lou, W., & Hou, Y. T., "A survey of distributed consensus protocols for blockchain networks," *IEEE Communications Surveys & Tutorials,* vol. 22, no. 2, 2020.

[111] V. Buterin, Proof of stake: The making of Ethereum and the philosophy of blockchains, Seven Stories Press, 2022.

[112] J. Roberto, "Understanding Proof of Stake: The Nothing at Stake Theory," Medium, 07 06 2018. [Online]. Available: https://medium.com/coinmonks/understanding-proof-of-stake-the-nothing-at-stake-theory-1f0d71bc027.

[113] Bentov, I., Lee, C., Mizrahi, A., & Rosenfeld, M, "Proof of activity: Extending bitcoin's proof of work via proof of stake," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 3, pp. 34-37, 2014.

[114] V. Buterin, "Long-Range Attacks: The Serious Problem With Adaptive Proof of Work," Ethereum Foundation, 15 05 2014. [Online]. Available: https://blog.ethereum.org/2014/05/15/long-range-attacks-the-serious-problem-with-adaptive-proof-of-work. [Accessed 04 11 2023].

[115] N. community, "Nxt Whitepaper," 07 02 2016. [Online]. Available: https://nxtdocs.jelurida.com/Nxt_Whitepaper.

[116] Rabin, T., & Ben-Or, M., "Verifiable secret sharing and multiparty protocols with honest majority.," *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pp. 73-85, 1989.

[117] J. Kwon, "Tendermint: Consensus without mining.," *Draft v. 0.6*, pp. 1-11, 2014.

[118] Gilad, Y., Hemo, R., Micali, S., Vlachos, G., & Zeldovich, N., "Algorand: Scaling byzantine agreements for cryptocurrencies.," *Proceedings of the 26th symposium on operating systems principles*, pp. 51-68, 2017.

[119] Nakamura, R., Jimba, T., & Harz, D., " Refinement and verification of CBC casper," *Crypto Valley Conference on Blockchain Technology (CVCBT) IEEE*, pp. 26-38, 2019.

[120] D. Larimer, "DPOS BFT— Pipelined Byzantine Fault Tolerance," Medium, 16 05 2018. [Online]. Available: https://medium.com/eosio/dpos-bft-pipelined-byzantine-fault-tolerance-8a0634a270ba. [Accessed 12 10 2023].

[121] "Cosmos SDK documentation," Cosmos, [Online]. Available: https://docs.cosmos.network/. [Accessed 12 10 2023].

[122] Li, W., Andreina, S., Bohli, J. M., & Karame, G., "Securing proof-of-stake blockchain protocols.," *Data Privacy Management, Cryptocurrencies and Blockchain Technology: ESORICS 2017 International Workshops, DPM 2017 and CBT 2017, Oslo, Norway, September 14-15, 2017,* pp. 297-315, 2017.

[123] І. А. Бурмака, М. С. Дорош, "Оптимізація використання обчислювальних ресурсів розподіленою системою виявлення вторгнень на основі blockchain".*Проблеми зняття з експлуатації об'єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 21) : збірникматеріалів VI Міжнародної конференції (27–29 квітня 2021,м. Славутич). – Чернігів : НУ «Чернігівська політехніка», 2021. – с. 47-50.*

[124] Burmaka, I. A., Lytvynov, V. V., Skiter, I. S., & Lytvyn, S. V., "Evaluating a blockchain-based network performance for the intrusion detection system.," *Математичні машини і системи,* vol. 1, pp. 99-109, 2020.

[125] Zhang, S., & Lee, J. H., "Analysis of the main consensus protocols of blockchain," *ICT express,* vol. 6, no. 2, pp. 93-97, 2020.

[126] "Закон України Про інформацію, Редакція від 27.07.2023," [Online]. Available: https://zakon.rada.gov.ua/laws/show/2657-12#Text.

[127] "Закон України Про захист інформації в інформаційно-комунікаційних системах Редакція від 31.12.2023," [Online]. Available: https://zakon.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80#Text.

[128] "PFCTL(8) System Manager's Manual," OpenBSD manual page server, [Online]. Available: https://man.openbsd.org/pfctl.8. [Accessed 12 01 2024].

[129] Burmaka, I., Stoianov, N., Lytvynov, V., Dorosh, M., & Lytvyn, S., "Proof of stake for blockchain based distributed intrusion detecting system," *Dorosh, M., & Lytvyn, S. (2020, August). Proof of Stake for Blockchain Based Distributed Intrusion Detecting System. In Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MOD*, vol. 1265, p. 237, 2020.

ДОДАТКИ

ДОДАТОК А
СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

[1] Burmaka, I., Stoianov, N., Lytvynov, V., Dorosh, M., & Lytvyn, S., "Proof of stake for blockchain based distributed intrusion detecting system," *Dorosh, M., & Lytvyn, S. (2020, August). Proof of Stake for Blockchain Based Distributed Intrusion Detecting System. In Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MOD,* vol. 1265, p. 237, 2020. https://doi.org/10.1007/978-3-030-58124-4_23 (SCOPUS)

[2] Burmaka, I., Dorosh, M., Skiter, I., & Lytvyn, S, "Architecture of Distributed Blockchain Based Intrusion Detecting System for SOHO Networks," *Mathematical Modeling and Simulation of Systems (MODS'2020): Selected Papers of 15th International Scientific-practical Conference, MODS, 2021 June 28–July 01, Chernihiv, Ukraine. Springer Nature,* pp. 313-326, 2021. https://doi.org/10.1007/978-3-030-89902-8_24 (SCOPUS)

[3] Burmaka, I., Zlobin, S., Lytvyn, S., & Nekhai, V., "Detecting flood attacks and abnormal system usage with artificial immune system," *Mathematical Modeling and Simulation of Systems: Selected Papers of 14th International Scientific-Practical Conference, MODS, 2019 June 24-26, Chernihiv, Ukraine*, pp. 131-143, 2019. https://doi.org/10.1007/978-3-030-25741-5_14 (SCOPUS)

[4] Skiter, I., Burmaka, I., & Sigayov, A., "Design of Technical Methods for Analysing Network Security Based on Identification of Network Traffic Anomalies," *Information & Security,* vol. 47, no. 3, pp. 306-316, 2020 https://doi.org/10.11610/isij.4722

[5] Burmaka, I. A., Lytvynov, V. V., Skiter, I. S., & Lytvyn, S. V., "Evaluating a blockchain-based network performance for the intrusion detection

system" *Математичні машини і системи* vol. 1, pp. 99-109, 2020. DOI: 10.34121/1028-9763-2020-1-99-109

[6]. Lytvynov, N. Stoianov, I. Stetsenko, I. Skiter, O. Trunova, A. Hrebennyk, V. Nekhai, I. Burmaka. Attacks defense of computer nets by tools using extended information about environment : monograph – Chernihiv : Chernihiv Politechnic National University, 2021. – 212 с.

[7] I. Burmaka, «CONSENSUS ALGORITHM COMPARISON FOR BLOCKCHAIN BASED INTRUSION DETECTING SYSTEM». Безпека ресурсів інформаційних систем : збірник тез I Міжнародної науково-практичної конференції(м. Чернігів 16-17 квітня 2020р.). –Чернігів : НУЧП, 2020. –с.6-14

[8] Бурмака І.А., «КЛАСИФІКАЦІЯ СИСТЕМ ВИЯВЛЕНЯ ВТОРГНЕНЬ В РОЗПОДІЛЕНІ ІНФОРМАЦІЙНІ СИСТЕМИ». Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 17): збірник матеріалів IV Міжнародної конференції (25–27 квітня 2017, м. Славутич). – Чернігів : ЧНТУ, 2017. – с. 59-63

[9] Бурмака Іван Анатолійович, «Архітектура розподіленої системи виявлення вторгнень на основі blockchain технології». Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 2020) в режимі онлайн: збірник матеріалів V Міжнародної конференції (27–29 квітня 2020, м. Славутич). – Чернігів : ЧНТУ, 2020. с.54-59

[10] І. А. Бурмака, М. С. Дорош «Оптимізація використання обчислювальних ресурсів розподіленою системою виявлення вторгнень на

основі blockchain». Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 21) : збірникматеріалів VI Міжнародної конференції (27–29 квітня 2021,м. Славутич). – Чернігів : НУ «Чернігівська політехніка», 2021. – с. 47-50

ДОДАТОК Б
ДОВІДКИ ПРО ВПРОВАДЖЕННЯ

Міністерство культури та інформаційної політики України
Департамент культури і туризму, національностей та релігій
Чернігівської обласної державної адміністрації
ЧЕРНІГІВСЬКИЙ ОБЛАСНИЙ ФІЛАРМОНІЙНИЙ ЦЕНТР ФЕСТИВАЛІВ ТА КОНЦЕРТНИХ ПРОГРАМ
Проспект Миру, 15, м. Чернігів, 14000, тел./факс (0462) 777-368, 675-424;
тел. 676-032, 675-893
E-mail: centre2000@ukr.net www.nota.net.ua

Від 07.05.2024 року р. № 131/01-08

Проректору з наукової
роботи національного університету
“Чернігівська політехніка”
Вікторії МАРГАСОВІЙ
14035, м. Чернігів, в.Шевченка, 95

ДОВІДКА ПРО ВПРОВАДЖЕННЯ
наукових результатів дисертаційної роботи
БУРМАКИ Івана Анатолійовича на тему: “Інформаційна технологія виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств на основі blockchain”,
представленої на здобуття ступеня доктора філософії
зі спеціальності 122 - “Комп’ютерні науки”

Запропонована в дисертаційній роботі БУРМАКИ Івана Анатолійовича на тему “Інформаційна технологія виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств на основі blockchain” технологія виявлення та аналізу аномальних подій для захисту комп’ютерних мереж малих та середніх підприємств дозволила в Чернігівському обласному філармонійному центрі фестивалів та концертних програм:

-провести аналіз та виявити аномальні події зовнішнього та внутрішнього походження в комп’ютерній мережі підприємства;

-визначити основні вектори атак на комп’ютерну мережу підприємства;

-розробити заходи з подальшого укріплення захисту мережевої інфраструктури Чернігівського обласного філармонійного центру.

В.о. генерального директора-
керівника художнього Сергій ТЕРЕБУН

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЧЕРНІГІВСЬКА ПОЛІТЕХНІКА»

вул. Шевченка, 95, Чернігів, 14035, Україна

тел. +38(0462) 665-103;
факс +38(0462) 665-105
E-mail: cstu@stu.cn.ua
www.stu.cn.ua
Код ЄДРПОУ 05460798

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

CHERNIHIV POLYTECHNIC NATIONAL UNIVERSITY

95, Shevchenko str., Chernihiv, 14035, Ukraine

14.05.2024 № 203/08-1128/вс

На № _____ від _____

Довідка
про впровадження результатів дисертаційної роботи
БУРМАКИ Івана Анатолійовича
на тему «Інформаційна технологія виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain»
на здобуття наукового ступеня доктора філософії за спеціальністю 122 «Комп'ютерні науки»

Основні положення та результати дисертаційного дослідження БУРМАКИ Івана Анатолійовича на тему «Інформаційна технологія виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain» були впроваджені в навчальному процесі Національного університету «Чернігівська політехніка» у такому вигляді:

1. В складі лекцій та лабораторних робіт для здобувачів вищої освіти за освітнім ступенем бакалавр спеціальності 121 – «Інженерія програмного забезпечення» освітньо-професійної програми «Інженерія програмного забезпечення» в рамках дисципліни «Системи захисту обчислювальних мереж» та за освітнім ступенем магістр спеціальності 121 – «Інженерія програмного забезпечення» освітньо-професійної програми «Інженерія програмного забезпечення» в рамках дисципліни «Моделювання, аналіз та інструментальні засоби безпеки комп'ютерних мереж», які містять рекомендації щодо побудови захищених комп'ютерних мереж та налаштування систем виявлення та попередження мережевих вторгнень.

2. У науковій роботі кафедри у проектах «Cyber Rapid Analisys for Defense Awareness of Real-time Situation - CyRADARS» за грантом NATO SPS (grant agreement number: G5286).

Проректор з наукової роботи В.Г. Маргасова

Україна

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ
«ІНФОРМАЦІЙНІ СИСТЕМИ ЗАХИСТУ»

14000, м. Чернігів, вул. Гонча, 23, тел. (050) 355-55-68, пошта: a_rudenok@ukr.net, www.isz.kyiv.ua

№ 54 від 13.05.2024

АКТ ПРО ВПРОВАДЖЕННЯ

наукових результатів дисертаційної роботи БУРМАКИ Івана Анатолійовича на тему: "Інформаційна технологія виявлення та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств на основі blockchain", представленої на здобуття ступеня доктора філософії зі спеціальності 122 - "Комп'ютерні науки"

Під час проектування та впровадження комплексних систем захисту інформації в інформаційно-комунікаційних системах були використані результати дисертаційної роботи БУРМАКИ Івана Анатолійовича.

Запропонована в дисертаційній роботі інформаційна технологія виявлення та та аналізу аномальних подій для захисту комп'ютерних мереж малих та середніх підприємств дозволила:

-визначити основні вразливості та вектори атак на інформаційну інфраструктуру підприємства;

-розробити комплекс заходів для усунення виявлених вразливостей та підвищення рівня захищеності мережевої інфраструктури.

Також це надало можливість підвищити швидкість реагування на кіберінциденти та оперативно усувати їх наслідки.

Директор ТОВ «Інформаційні системи захисту» Олександр Руденок

ДОДАТОК В

ВИДИ КІБЕРАТАК

Таблиця В.1 – ВИДИ КІБЕРАТАК

Назва	Тип атаки	Напрямок атаки	Опис
1	2	3	4
SYN Flood Attack	Активна,DOS	Відмова в обслуговуванні, TCP протокол	Спрямована на системи, що використовують TCP протокол, для перевантаження системи зловмисник відкриває багато з’єднань, через що система не відповідає на запити користувачів
TCP Reset Attack	Активна,DOS	Відмова в обслуговуванні, TCP протокол	Спрямована на системи, що використовують TCP протокол. Зловмисник прослуховує TCP-з'єднання жертви і надсилає фальшивий пакет TCP RESET жертві. Тоді це змушує жертву ненавмисно розірвати своє TCP-з'єднання.
ICMP Attack	Активна,DOS	Відмова в обслуговуванні, ICMP протокол, пропускна здатність	Пакети exo-запиту ICMP надсилаються через атаку маскування на IP-адреси широкомовної трансляції, це призводить до того, що велика кількість пакетів відповідей echo ICMP надсилається від посередника на адресу жертви, це спричиняє перевантаження мережі та збої.
UDP Storm Attack	Активна,DOS	Відмова в обслуговуванні, Пропукна здатність	Перешкоджає роботи хостів та служб в мережі, сповільнює мережу. Виконується шляхом встановлення зв’язку між двома службами UDP, кожна з яких виробляє дуже велику кількість пакетів.

Продовження Таблиці В.1

1	2	3	4
DNS Request Attack	Активна,DOS	Пропускна здатність мережі, процесорні ресурси	Надсилається велика кількість запитів DNS на сервери імен, підроблюючи IP-адресу джерела. Сервер імен, виконує роль проміжної сторони в атаці, відповідає, посилаючи результати на підроблену IP-адресу жертви, перевантажуючи інтернет канал.
CGI Request Attack	Активна,DOS	Процесорні ресурси	Зловмисник надсилає багато запитів CGI на цільовий сервер, тоді зловмисник споживає весь ресурс ЦП жертви. Сервер вимушений аварійно завершувати сервіси.
Mail Bomb Attack	Активна,DOS	Перевантаження поштових сервісів, заповнення дискового простору	Надсилання великої кількості електронної пошти може просто заповнити дисковий простір одержувача на сервері, або, в деяких випадках, може перевантажити сервер і спричинити припинення роботи сервера.
ARP Storm Attack	Активна,DOS	Процесорні ресурси, ARP	Система превантажується великою кількістю ARP запитів.
Algorithmic Complexity Attack	Активна,DOS	Процесорні ресурси	DDOS-атака з низькою пропускною здатністю, що створює перевантаження у найгіршому випадку роботи алгоритмів, що використовуються програмами. Наприклад, цілями атаки можуть бути двійкові дерева або хеш-таблиці з підібраними входами для значного споживання системних ресурсів.
Spam Attack	Активна,DOS	Поштові служби	Спам атака порушує роботу поштових служб різних організацій, що завдає серйозних проблем користувачам. Може призвести до заповнення дискового простору.

Продовження Таблиці В.1

1	2	3	4
Підслухову-вання	Пасивна, Підслуховування	Конфіденційна інформація	Атакуючий намагається викрасти інформацію, яку користувач передає через мережу. Зловмисник використовує незахищені мережеві комунікації для доступу до даних, що надсилаються та отримуються користувачем.
Spear Phishing	Активна, Фішинг	Конфіденційна інформація	Направляється на конкретного користувача або організацію. Зловмисник спочатку збирає інформацію про користувача, щоб підвищити ймовірність успішного проведення атаки.
Clone Phishing	Активна, Фішинг	Конфіденційна інформація	Створення схожого на легітимний листа, з підміною посилання на фішинговий сайт, який копіює справжній, але має мету вкрасти конфіденційні дані.
Whaling	Активна, Фішинг	Конфіденційна інформація	Націлений на впливових кінцевих споживачів, таких як керівники вищої ланки корпорацій, політики та бізнесмени. Вміст таких фішингових сторінок орієнтований на відповідну цільову аудиторію.
Маніпуляція посиланнями	Активна, Фішинг	Конфіденційна інформація	Маніпулювання посиланнями використовує певну форму технічного обману, щоб посилання в електронному листі перенаправляло на підроблений сайт.
Підробка вебсайту	Активна, Фішинг	Конфіденційна інформація	Фішинговий сайт маскується під легітимний сайт, використовуючи схожий дизайн та підміну адреси. Основна мета — крадіжка персональних та банківських даних.
Ухилення від фільтру			Використання зображення замість тексту, щоб антифішинг-фільтри не могли виявити текст , який зазвичай використовується у фішинг-листах. Це призвело до еволюції більш досконалих

Продовження Таблиці В.1

1	2	3	4
			антифішинг-фільтрів, здатних відновити прихований текст на зображеннях(OCR).
Телефонний фішинг			Зловмисник надсилає повідомлення, яке, як стверджується, надійшло від банку, щоб користувач набрав номер телефону щодо проблем із його банківським рахунком. Коли користувач набирає номер телефону, пропонується ввести номер рахунку та PIN-код.
Vishing (голосовий фішинг)			Використання VoIP технології для підробки ідентифікаторів абонентів (телефонних номерів).
Міжсайтовий сценарій (XSS)	XSS	Веб застосунки	Використовує вразливість у веб-програмах, що дозволяє зловмиснику вводити код на веб-сторінки, які переглядають інші користувачі.
Атаки прямого доступу	Атаки прямого доступу	Персональні дані	Крадіжка персональних дани користувача шляхом отримання фізичного доступу до пристрою.
Мережі ботів	Мережі ботів	Встановлення шкідливого ПЗ для ботнета	Шкідливе ПЗ (троянські програми) захоплюють пристрої, які потім можуть використовуватися для здійснення атак DOS або здійснення інших шкідливих дій.
Likejacking	Clickjacking	Просування публікацій у соціальних мережах обманним шляхом	Зловмисник “змушує” користувачів ставити лайки публікації, на яку користувачі не хотіли лайкати.

Продовження Таблиці В.1

1	2	3	4
Cursorjacking	Clickjacking	Виконання зловмисного коду	Шляхом приховування реального курсора та малювання підробного в іншому місці, зловмисник змушує користувача натиснути певну кнопку, щоб запустити вразливий код.
Атака менеджера паролів	Clickjacking	Крадіжка паролів	Деякі менеджери паролів пропонують автозаповнення паролів для http версії, не дивлячись на те, що пароль був збережений для https версії. Зловмисник може створити підроблену http версію сайту і вкрасти пароль.
Комп’ютерний хробак	Backdoor	Персонльні дані	Тип шкідливого програмного забезпечення, яке використовується для викрадення особистих даних.
Object Code Backdoors	Backdoor	Крадіжка даних, виконання шкідливого коду	Шкідливі фрагменти вбудовуються в програмний код на одному з етапів створення чи розповсюдження програмного продукта.
Asymmetric Backdoors	Backdoor	Криптографічні дані	Використовується для атак на асиметричну криптографію і робить шифрування ненадійним, наприклад шляхом модифікації процесу генерації приватних та публічних ключів так, що приватний ключ можна відновити з публічного.
Compiler Backdoor	Backdoor	Крадіжка даних, виконання шкідливого коду	Заражений компілятор дає на виході програми які також заражені шкідливим кодом.
ARP Spoofing Attack	Підробка (Spoofing)	Крадіжка даних, Компромнтація аккаунтів.	За рахунок підробки ARP відповідей дозволяє зв’язати IP адресу хоста на який здійснюється атака з MAC адресою зловмисника, і перенаправити трафік зловмиснику для подальшого здійснення інших атак.

Продовження Таблиці В.1

1	2	3	4
Атака підміни DNS	Підробка (Spoofing)	DNS	Підробка DNS дозволяє зловмиснику перенаправити користувача на іншу адресу для подальшого здійснення інших атак.
IP Spoofing Attack	Підробка (Spoofing)	Обхід систем захисту, маскування	Підмінюючи IP адресу зловмисник маскується під легального користувача і отримує доступ до даних або проводить втручання від імені користувача.
Web Spoofing	Підробка (Spoofing)	Крадіжка конфіденційних даних	Атака при якій зловмисник створює шкідливий сайт схожий на надійний. Якщо користувач не помітить підробку і введе конфіденційні дані, вони потраплять до зловмисника.
Tampering	Активна, Втручання	Фізичний доступ до пристрою	Отримання фізичного доступу до залишеного без нагляду пристрою, подальшими цілями може бути крадіжка , модифікація або видалення інформації. Можливе також фізичне порушення роботи системи.