DOI: 10.58254/viti.7.2025.18.202

УДК 004.056.57

PhD Фесьоха В. В. ORCID: 0000-0001-6612-1970 (ВІТІ ім. Героїв Крут)
Кисиленко Д. Ю. ORCID: 0000-0001-5491-6231 (ВІТІ ім. Героїв Крут)

МЕТОД САМОНАВЧАННЯ СИСТЕМИ КІБЕРЗАХИСТУ НА ОСНОВІ ІНВАРІАНТІВ ПОВЕДІНКИ ПОЛІМОРФНОГО ТА МЕТАМОРФНОГО ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Нові зразки шкідливого програмного забезпечення, зокрема поліморфні та метаморфні, характеризуються підвищеною складністю та здатністю змінювати поведінкові ознаки з метою уникнення виявлення існуючими системами кіберзахисту, що становить серйозну загрозу для кібербезпеки інформаційно-комунікаційних систем. Основною проблемою існуючих систем кіберзахисту в даному контексті є використання баз опису зразків шкідливого коду, які з часом втрачають актуальність і стають неспроможними забезпечити виявлення нових реалізацій кіберзагроз.

У статті запропоновано метод самонавчання системи кіберзахисту на основі інваріантів поведінки поліморфного та метаморфного шкідливого програмного забезпечення. Даний метод передбачає використання двоешелонної системи виявлення: на першому ешелоні здійснюється аналіз активності інформаційно-комунікаційної системи на предмет виявлення шкідливого програмного забезпечення за поведінковими ознаками, значення яких описані нечіткими лінгвістичними термами для кожного відомого його типу; другий ешелон використовує заздалегідь сформовану базу інваріантів поведінки, описаних нечіткими продукційними правилами для підвищення ефективності виявлення поліморфного та метаморфного шкідливого програмного забезпечення.

Ключовою особливістю запропонованого методу є автоматична генерація нових нечітких правил для опису поведінкових ознак невідомих зразків шкідливого програмного забезпечення, що досягається шляхом об'єднання інваріантної та поліморфної компонент у випадку детекції інваріантної компоненти. Додатково передбачено перевизначення інваріантних компонент шляхом повторного відбору таких ознак, які закономірно в сукупності повторюються для більшості екземплярів кожного типу шкідливого програмного забезпечення. Даний процес забезпечує самонавчання системи кіберзахисту, що дає змогу ефективніше протидіяти поліморфному та метаморфному шкідливому програмному забезпеченню.

Ключові слова: *поліморфне та метаморфне шкідливе програмне забезпечення, поведінковий аналіз, кібербезпека, нечітка логіка, машинне навчання, генетичні алгоритми, інваріантна компонента.*

V. Fesokha, D. Kysylenko. A method of self-learning of a cyber defence system based on behavioural invariants of polymorphic and metamorphic malware

New samples of malware, in particular polymorphic and metamorphic, are characterised by increased complexity and the ability to change behavioural characteristics to avoid detection by existing cyber defence systems, which poses a serious threat to the cybersecurity of information and communication systems. The main problem of existing cyber defence systems in this context is the use of databases describing malicious code samples, which lose their relevance over time and become unable to detect new implementations of cyber threats.

The article proposes a method of self-learning of a cyber defence system based on behavioural invariants of polymorphic and metamorphic malware. This method involves the use of a two-tier detection system: the first tier analyzes the activity of the information and communication system to detect malware by behavioural features, the values of which are described by fuzzy linguistic terms for each known type of malware; the second tier uses a pre-formed database of behavioural invariants described by fuzzy product rules to increase the efficiency of detecting polymorphic and metamorphic malware.

The key feature of the proposed method is the automatic generation of new fuzzy rules to describe the behavioural characteristics of unknown malware samples, which is achieved by combining the invariant and polymorphic components in the case of detecting the invariant component. Additionally, the system provides for the redefinition of invariant components by re-selecting such features that are naturally repeated in the aggregate for most instances of each type of malware. This process ensures self-learning of the cyber defence system, which makes it possible to more effectively counter polymorphic and metamorphic malware.

Keywords: *polymorphic and metamorphic malware, behavioural analysis, cybersecurity, fuzzy logic, machine learning, genetic algorithms, invariant component.*

Актуальність та постановка завдання в загальному вигляді. Існуючі темпи еволюції підходів до реалізації кіберзагроз призвели до зростання кількості нових і появи дедалі складніших варіантів деструктивного впливу, серед яких особливо небезпечними є поліморфне та метаморфне шкідливе програмне забезпечення (ШПЗ). Його здатність до динамічної адаптації ускладнює процес ідентифікації традиційними методами виявлення, які не забезпечують належного рівня детекції нових, раніше невідомих зразків ШПЗ. Крім того, використання зловмисниками загальнодоступних технологій штучного інтелекту (ШІ) для автоматизованої генерації поліморфних та метаморфних зразків ШПЗ створює серйозні виклики для систем кібербезпеки в контексті захисту ключових інформаційних систем (ІС), зокрема ІС військового призначення [1].

Одним із доцільних підходів до виявлення поліморфного та метаморфного ШПЗ є застосування моделі визначення поведінкових інваріантів, властивих відомим типам ШПЗ на основі інтеграції нечіткої логіки та генетичних алгоритмів (ГА) [2]. Зазначена модельґрунтується на визначенні поведінкової інваріантної компоненти для кожного відомого типу ШПЗ, що залишається незмінною незалежно від проведених модифікацій. Так, досліджувана множина поведінкових ознак описується нечіткими лінгвістичними термами, на основі яких формуються нечіткі продукційні правила для подальшого визначення засобами ГА інваріантної компоненти як підмножини ознак, що характеризує відповідний тип ШПЗ. Такий підхід дозволяє досягти високої точності виявлення поліморфних та метаморфних шкідливих програм на основі заздалегідь визначених інваріантів поведінки.

Поряд з цим, актуальність знань (правил) даної моделі, як і решти подібних підходів до виявлення поліморфного та метаморфного ШПЗ [3–6], стрімко знижується внаслідок безпосередньої залежності від швидкості еволюції методів маскування та модифікації шкідливого коду, що використовуються зловмисниками.

Це зумовлює необхідність подальших досліджень, які полягають у підвищенні ефективності виявлення поліморфного та метаморфного ШПЗ шляхом постійного доповнення знань (правил) та адаптації систем кіберзахисту до нового ШПЗ.

Аналіз попередніх досліджень [3–6] щодо адаптації систем кіберзахисту до ШПЗ, здатного до самомодифікації, показує, що переважна їх кількість базується на використанні моделей машинного навчання, зокрема для обробки великих масивів даних, аналізу PE-заголовків, визначення концептуального дрейфу між схожими зразками шкідливих програм, створення представлень про нові зразки ШПЗ засобами генеративних моделей ШІ. Однак зазначені підходи хоч і дозволяють системі певною мірою своєчасно самонавчатися (донавчатися), наприклад, у випадках використання інкрементального навчання (Incremental Learning), навчання з перенесенням (Transfer Learning), навчання з псевдопозначками (Self-training with Pseudo-labeling) або потокового навчання (Online Learning/Continual Learning), проте потребують постійного контролю для запобігання втрати попередніх знань, а також не забезпечують верифікацію та інтерпретацію прийнятих рішень, що у свою чергу ускладнює як дослідження кіберінцидентів, так і відстеження еволюції ШПЗ. Решта досліджень [7; 8], які базуються на використанні баз знань (правил), хоча й забезпечують верифікацію та інтерпретацію прийнятих рішень, водночас потребують залучення експертів, що, у свою чергу, уповільнює процес навчання системи та вносить елемент суб’єктивності суджень.

У зв’язку з цим постає актуальне наукове завдання щодо розробки методу самонавчання для системи кіберзахисту, в основі якої покладено модель визначення інваріантної компоненти в поведінці ШПЗ на основі інтеграції нечіткої логіки та ГА без залучення експертів [2].

Метою статті є розробка методу самонавчання системи кіберзахисту на основі інваріантів поведінки поліморфного та метаморфного ШПЗ.

Метод самонавчання системи кіберзахисту на основі інваріантів поведінки поліморфного та метаморфного ШПЗ. З урахуванням виявлених недоліків існуючих підходів щодо адаптації систем кіберзахисту до поліморфного та метаморфного ШПЗ, а також особливостей реалізації моделі, запропонованої в [2], пріоритетним напрямком вирішення сформульованого наукового завдання доцільно розглядати автоматичне доповнення новими правилами існуючої бази нечітких правил, отриманих внаслідок результатів детекції специфічних поведінкових патернів ШПЗ. Реалізація даного підходу передбачає використання двоешелонованого способу відслідковування активності поліморфного та метаморфного ШПЗ [9].

Перший ешелон аналізує трафік ІС на предмет виявлення шкідливої активності засобами бази нечітких правил, яка описує відомі поведінкові ознаки ШПЗ різноманітних типів.

Другий ешелон аналізує трафік ІС на предмет виявлення шкідливої активності засобами бази нечітких правил, що описує інваріантні компоненти поведінки поліморфного та метаморфного ШПЗ. Якщо ознаки активності ІС відповідають описаній нечіткими лінгвістичними термами інваріантній компоненті поведінки ШПЗ, то дана активність класифікується як шкідлива.

Вихідні дані:

$T=\{t_i\}$ – трафік ІС, де t_i – окремий фрагмент трафіка або подія в системі;

db_1 – база нечітких правил першого ешелону, яка описує відомі поведінкові ознаки ШПЗ;

db_2 – база нечітких правил другого ешелону, що містить інваріантні компоненти поведінки поліморфного та метаморфного ШПЗ;

$F_1 : t_i \rightarrow \{0,1\}$ – функція детекції першого ешелону, яка визначає, чи відповідає t_i правилу з db_1:

$$F_1(t_i) = \begin{cases} 1, \text{якщо } t_i \text{ відповідає хоча б одному } r_1^j \in db_1, \\ 0, \text{інакше.} \end{cases} \quad (1)$$

$F_2 : t_i \rightarrow \{0,1\}$ – функція детекції другого ешелону, яка визначає, чи відповідає t_i правилу з db_2:

$$F_2(t_i) = \begin{cases} 1, \text{якщо } t_i \text{ відповідає хоча б одному } r_2^k \in db_2, \\ 0, \text{інакше.} \end{cases} \quad (2)$$

Обмеження та допущення:

db_2 – містить інваріантні компоненти поведінки ШПЗ, які є незалежними від поліморфізму чи метаморфізму;

поліморфна $P(t_i)$ або метаморфна $M(t_i)$ компоненти може бути зафіксована, якщо t_i не детектується першим ешелоном, але детектується другим;

процес оновлення db_1 відбувається за рахунок додавання нових правил $r_2^j \in db_1$, тоді як оновлення db_2 відбувається через певний період, після накопичення достатньої кількості нових правил в db_1.

Необхідно:

визначити поліморфну $P(t_i)$ або метаморфну $M(t_i)$ компоненти в поведінці шляхом порівняння поведінки з відомими інваріантами I_i;

сформувати нові нечіткі правила $R_i : I(t_i) + \{P(t_i), M(t_i)\}$ за умови: *якщо* $I(t_i) \approx I_i$, *то* $t_i \approx$ ШПЗ, де $I(t_i)$ – інваріантна компонента;

додати згенероване правило R_i *до* db_1.

Метод самонавчання системи кіберзахисту на основі поведінкових інваріантів поліморфного та метаморфного ШПЗ передбачає реалізацію наступних етапів:

1. *Ініціалізація вихідних даних.*

На даному етапі ініціалізується множина поведінкових ознак ШПЗ із відомих наборів даних про активність ШПЗ, наприклад, таких як ML-Based NIDS Datasets [10], на основі яких заповнюються бази 1-го та 2-го ешелонів [2].

2. *Аналіз активності 1-м ешелоном*.

На цьому етапі відбувається первинне сканування трафіку ІС із використанням бази правил першого ешелону db_1 з метою визначення відповідності фрагмента трафіка t_i ІС існуючим правилам (1). Якщо фрагмент трафіка t_i класифіковано як шкідливий (деструктивний), відбувається негайне його блокування. В іншому випадку трафік передається для подальшого аналізу на 2-й ешелон [2].

3. *Аналіз активності 2-м ешелоном*.

На цьому рівні проводиться перевірка на наявність інваріантних компонент у ознаках фрагмента трафіка t_i ІС. Визначення відповідності здійснюється шляхом порівняння поведінкових ознак з базою db_2 (2), що містить інформацію про інваріантні компоненти $I(t_i)$ відомих класів ШПЗ $I(t_i) \in db_2$. Якщо трафік t_i містить ознаки ШПЗ, які не було виявлено на попередньому етапі, то t_i класифікується як частина нового зразка ШПЗ такого типу, до якого належала інваріантна компонента $I(t_i)$. Далі відбувається блокування трафіка t_i [2].

4. *Фіксація вектора значень ознак поліморфної* $P(t_i)$ *або метаморфної* $M(t_i)$ *компоненти.* Блокування трафіка t_i на попередньому етапі свідчить про присутність в інфраструктурі ІС поліморфного або метаморфного ШПЗ. Це також є свідченням того, що у базі правил 1-го ешелону db_1 відсутнє правило $r_1^j \notin db_1$, яке дає змогу виявити новий зразок ШПЗ.

У зв'язку з цим, доцільним підходом до вирішення даного завдання є генерація нового нечіткого правила $r^j{}_1$ *для бази* db_1 з метою забезпечення своєчасного самонавчання системи кіберзахисту. Для реалізації описаного, при визначенні відповідності фрагмента трафіка t_i ІС як нового зразка ШПЗ на 3-му етапі, необхідно зафіксувати як активоване правило $r_2^k \in db_2$, що описує певну інваріантну компоненту, так і поліморфну $P(t_i)$ або метаморфну $M(t_i)$ компоненти. У такому випадку система фіксує нові ознаки поліморфної чи метаморфної компонент.

5. *Генерація нових правил.*

На основі зафіксованих даних на попередньому етапі щодо активності в системі поліморфного та метаморфного ШПЗ здійснюється формування нового нечіткого правила $r^{new}{}_1$ для бази db_1:

Нехай:

$I(t)$ – інваріантна компонента поведінки ШПЗ, що вже є в базі db_2;

$P(t)$ – нова поліморфна компонента, що визначається як різниця між загальною поведінкою $B(t)$, початково обмежену відповідністю множині ознак із офіційного набору даних про активність ШПЗ [10] та інваріантною компонентою $I(t)$ (3):

$$P(t) = B(t) - I(t). \tag{3}$$

$\mu db_1(t)$ – ступінь відповідності поведінки t правилам бази db_1.

$\mu db_2(t)$ – ступінь відповідності правилам бази db_2.

Θ_1, Θ_2 – порогові значення відповідності правилам, де Θ_1 – *низьке,* Θ_2 – *високе*.

Тоді, якщо $\mu db_1(t) < \Theta_1$, $\mu db_2(t) \geq \Theta_2$, то нове нечітке правило формується як (4):

$$r^{new}: \text{якщо } I(t) \text{ та } P(t), \text{ то } t \in \text{ШПЗ} \quad (4)$$

6. *Додавання нового правила r^{new} до бази db_1.*

Згенероване нечітке правило додається до бази правил 1-го ешелону (5), що дає змогу блокувати схожу шкідливу активність на ранніх етапах протидії поліморфному та метаморфному ШПЗ.

$$db_1 \leftarrow db_1 \cup r^{new} \quad (5)$$

7. *Оновлення бази інваріантних компонент db_2.*

Для забезпечення актуальності бази інваріантних компонент db_2 на даному етапі здійснюється їх перевизначення на основі [2] шляхом відбору таких ознак, які закономірно в сукупності повторюються для більшості екземплярів кожного типу ШПЗ. Даний процес забезпечує самонавчання системи кіберзахисту, що дає змогу ефективніше протидіяти поліморфному та метаморфному ШПЗ.

Таким чином, запропонований метод самонавчання системи кіберзахисту на основі інваріантів поведінки поліморфного та метаморфного ШПЗ забезпечує гнучке та ефективне його виявлення. Використання інваріантних компонент у поєднанні з автоматичним оновленням баз правил дає змогу системі швидко реагувати на нові зразки ШПЗ, зменшуючи ймовірність їхнього обходу.

Приклад. Розглянемо виявлення активності одного з поширених типів ШПЗ – Generic, який спрямований на криптографічні системи, зокрема на створення колізій у блокових шифрах [11].

Відповідно до офіційного набору даних про ШПЗ ML-Based NIDS Datasets, а саме NF-UQ-NIDS-v2 [10], наведено відомі поведінкові ознаки Generic, на основі яких будуються правила для 1-го та 2-го (виділено зеленим кольором) ешелонів (табл. 1).

Поведінка екземпляра Generic характеризується наступними значеннями ознак:

*Generic** = <1043, 53, 17, 0, 556703, 2, 0, 0, 0, 0, 0, 4563288, 0, 0, 254, 254, 57, 57, 0, 57, 114, 0, 0, 0, 0, 0, 912000, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 5120, 255, 2000, 0>.

Композиційну таблицю, на основі якої було створено нечітке лінгвістичне правило для 1-го ешелону відслідковування активності ШПЗ типу Generic, наведено у таблиці 2.

Нечітке лінгвістичне правило 1-го ешелону має вигляд:

ЯКЩО $L4\_SRC\_PORT = t_1$ & $L4\_DST\_PORT = t_1$ & $PROTOCOL = t_1$ & $L7\_PROTO = t_1$ & $IN\_BYTES = t_1$ &
& $TPC\_FLAGS = t_1$ & $CLIENT\_TPC\_FLAGS = t_1$ & $SERVER\_TPC\_FLAGS = t_1$ & $FLOW\_DURATION\_MILLSECONDS = t_1$ &
& $MIN\_TTL = t_1$ & $MAX\_TTL = t_1$ & $SHORTEST\_FLOW\_PKT = t_1$ & $MIN\_IP\_PKT\_LEN = t_1$ &
& $NUM\_PKTS\_256\_TO\_512\_BYTES = t_1$ & $NUM\_PKTS\_512\_TO\_1024\_BYTES = t_1$ & $TCP\_WIN\_MAX\_IN = t_1$ &
& $TCP\_WIN\_MAX\_OUT = t_1$ & $ICMP\_TYPE = t_1$ $ICMP\_IPv4\_TYPE = t_1$ & $DNS\_QUERY\_ID = t_1$ &
& $DNS\_QUERY\_TYPE = t_1$ & $DNS\_TTL\_ANSWEAR = t_1$ & $FTP\_COMMAND\_RET\_CODE = t_1$, ТО $t \in Generic$

Очевидно, правило 1-го ешелону не активується через те, що значення ознак *in_bytes, flow_duration_milliseconds, dns_ttl_answear* виходять за межі діапазонів значень вказаних лінгвістичних термів, що у свою чергу не дозволяє виявити шкідливий трафік ШПЗ Generic. У такому випадку трафік передається для подальшого аналізу на 2-й ешелон.

Нечітке лінгвістичне правило, що було створене для 2-го ешелону, наведено у таблиці 3.

Нечітке лінгвістичне правило 2-го ешелону має вигляд:

ЯКЩО $IN\_PKTS = t_1$ & $OUT\_BYTES = t_1$ & $OUT\_PKTS = t_1$ & $DURATION\_IN = t_1$ &
& $DURATION\_OUT = t_1$ & $LONGEST\_FLOW\_PKT = t_1$ & $MAX\_IP\_PKT\_LEN = t_1$ &
& $SRC\_TO\_DST\_SECOND\_BYTES = t_1$ & $DST\_TO\_SRC\_SECOND\_BYTES = t_1$ & $RETRANSMITTED\_IN\_BYTES = t_1$ &
& $RETRANSMITTED\_IN\_PKT = t_1$ & $RETRANSMITTED\_OUT\_BYTES = t_1$ & $RETRANSMITTED\_IN\_PKT = t_1$ &
& $SRC\_TO\_DST\_AVG\_THROUGHPUT = t_1$ & $DST\_TO\_SRC\_AVG\_THROUGHPUT = t_1$ &
& $NUM\_PKTS\_UP\_TO\_128\_BYTES = t_1$ & $NUM\_PKTS\_128\_TO\_256\_BYTES = t_1$ & $NUM\_PKTS\_1024\_TO\_1514\_BYTES = t_1$,
ТО $t \in Generic$

Таким чином, правило 2-го ешелону відслідковування шкідливої активності успішно активується, використовуючи попередньо визначену інваріантну компоненту для типу ШПЗ Generic.

Для формування нового нечіткого правила r^{new} виявлення екземпляра ШПЗ Generic з метою реалізації самонавчання системи кіберзахисту необхідно зафіксувати нову поліморфну компоненту в його поведінці. Вихід значень ознак *in_bytes, flow_duration_milliseconds, dns_ttl_answear* за межі діапазонів значень поточних лінгвістичних термів призводить до генерації нових лінгвістичних термів t^j_{i+1}, діапазони значень яких визначаються но основі отриманих даних. Як показано в таблиці 4, нові лінгвістичні терми дозволяють коректно описати мінливі поведінкові ознаки, що дає змогу сформувати нове нечітке лінгвістичне правило на основі поєднання поточної інваріантної та визначеної поліморфної (метаморфної) компонент.

Нове нечітке лінгвістичне правило має вигляд:

ЯКЩО $L4\_SRC\_PORT = t_1$ & $L4\_DST\_PORT = t_1$ & $PROTOCOL = t_1$ & $L7\_PROTO = t_1$ & $IN\_BYTES = t_3$ &
& $TPC\_FLAGS = t_1$ & $CLIENT\_TPC\_FLAGS = t_1$ & $SERVER\_TPC\_FLAGS = t_1$ & $FLOW\_DURATION\_MILLSECONDS = t_3$ &
& $MIN\_TTL = t_1$ & $MAX\_TTL = t_1$ & $SHORTEST\_FLOW\_PKT = t_1$ & $MIN\_IP\_PKT\_LEN = t_1$ &
& $NUM\_PKTS\_256\_TO\_512\_BYTES = t_1$ & $NUM\_PKTS\_512\_TO\_1024\_BYTES = t_1$ & $TCP\_WIN\_MAX\_IN = t_1$ &
& $TCP\_WIN\_MAX\_OUT = t_1$ & $ICMP\_TYPE = t_1$ $ICMP\_IPv4\_TYPE = t_1$ & $DNS\_QUERY\_ID = t_1$ &
& $DNS\_QUERY\_TYPE = t_1$ & $DNS\_TTL\_ANSWEAR = t_2$ & $FTP\_COMMAND\_RET\_CODE = t_1$, ТО $t \in Generic$

Таким чином, сформоване нове нечітке лінгвістичне правило додається до бази правил 1-го ешелону, що сприяє адаптації системи кіберзахисту до виявлення нових зразків поліморфного та метаморфного ШПЗ.

Висновки. У статті вирішується актуальне наукове завдання, яке полягає у розробці методу самонавчання для системи кіберзахисту, який базується на використанні моделі визначення інваріантної компоненти в поведінці ШПЗ на основі інтеграції нечіткої логіки та ГА. Основна ідея методу, що відрізняє його від існуючих, полягає в реалізації способу автоматичного доповнення бази правил новими нечіткими правилами, сформованими на основі об'єднання інваріантної поведінкової компоненти ШПЗ із виявленими поліморфними або метаморфними змінами.

Ключовими результатами дослідження є:

розроблено новий підхід до самонавчання, який забезпечує адаптивність системи кіберзахисту до нових зразків ШПЗ без втрати накопичених знань;

запропоновано двоешелонну систему аналізу активності ІС, яка поєднує класичний поведінковий аналіз із перевіркою на відповідність інваріантним компонентам, що підвищує ефективність детекції;

забезпечено виключення фактору суб'єктивних експертних суджень.

Запропонований метод може бути використано для вдосконалення існуючих стратегій кіберстійкості ключових ІС, зокрема у випадках, коли необхідно забезпечити швидке реагування на нові варіанти кіберзагроз.

Перспективним напрямком подальших наукових досліджень є відслідковування еволюції ШПЗ з урахуванням концептуального дрейфу, а також розробка механізму контролю, який запобігатиме надмірній кількості хибних спрацювань нових правил.

Таблиця 1

Поведінкові ознаки ШПЗ Generic згідно з NF-UQ-NIDS-v2

L4_SRC_PORT	Початковий порт на 4-му рівні OSI – порт для встановлення з'єднання
L4_DST_POTR	Кінцевий порт на 4-му рівні OSI – порт для прийому з'єднання
PROTOCOL	Протокол на 3-му або 4-му рівнях OSI – визначає тип комунікації (TCP, UDP, ICMP)
L7_PROTO	Протокол на 7-му рівні OSI – визначає передачу даних між додатками (HTTP, FTP, DNS, SMTP)
IN_BYTES	Кількість отриманих байтів – обсяг отриманих даних у мережі
IN_PKTS	Кількість вхідних пакетів, отриманих мережею
OUT_BYTES	Кількість переданих байтів – на обсяг відправлених даних з пристрою
OUT_PKTS	Кількість вихідних пакетів
TPC_FLAGS	TCP-прапорці, показують стан з'єднання або команду у TCP
CLIENT_TPC_FLAGS	TCP-прапорці клієнта, що визначають дії, які виконує клієнт
SERVER_TPC_FLAGS	TCP-прапорці сервера, які вказують, як сервер реагує на запити клієнта
FLOW_DURATION_MILLSECONDS	Тривалість потоку в мілісекундах, час взаємодії між двома вузлами в мережі
DURATION_IN	Тривалість вхідного трафіку, час прийому даних
DURATION_OUT	Тривалість вихідного трафіку, час відправлення даних
MIN_TTL	Мінімальний час життя пакету
MAX_TTL	Максимальний час життя пакету
LONGEST_FLOW_PKT	Найбільший пакет у потоці даних
SHORTEST_FLOW_PKT	Найменший пакет у потоці даних
MIN_IP_PKT_LEN	Найменший розмір IP-пакету, визначає характер трафіку або виявити аномалії
MAX_IP_PKT_LEN	Найбільший розмір IP-пакету, який може бути використаний для великих файлів або атак
SRC_TO_DST_SECOND_BYTES	Кількість байтів від джерела до призначення за певний час, відстежує обсяг даних в одному напрямку
DST_TO_SRC_SECOND_BYTES	Кількість байтів від призначення до джерела, відображає зворотний трафік
RETRANSMITTED_IN_BYTES	Кількість повторно переданих байтів (вхідних), через втрату або пошкодження пакетів
RETRANSMITTED_IN_PKTS	Кількість повторно переданих вхідних пакетів, через втрату або пошкодження пакетів
RETRANSMITTED_OUT_BYTES	Кількість повторно переданих байтів (вихідних), після невдалих спроб передачі
RETRANSMITTED_OUT_PKTS	Кількість повторно переданих вихідних пакетів, які потребували повторної передачі через невдалі спроби
SRC_TO_DST_AVG_THROUGHPUT	Середня пропускна здатність (вхідна) – середній обсяг переданих даних від джерела до призначення за певний час
DST_TO_SRC_AVG_THROUGHPUT	Середня пропускна здатність (вихідна) – середній обсяг переданих даних від призначення до джерела за певний час
NUM_PKTS_UP_TO_128_BYTES	Пакети до 128 байт – малий трафік, використовується для виявлення аномалій
NUM_PKTS_128_TO_256_BYTES	Пакети від 128 до 256 байт – середній трафік у мережі
NUM_PKTS_256_TO_512_BYTES	Пакети від 256 до 512 байт – середній трафік або типи пакетів для протоколів
NUM_PKTS_512_TO_1024_BYTES	Пакети від 512 до 1024 байт – середній або стандартний трафік
NUM_PKTS_1024_TO_1514_BYTES	Пакети від 1024 до 1514 байт – максимальний розмір Ethernet-пакету
TCP_WIN_MAX_IN	Максимальний розмір TCP-вікна (вхідний) – максимальна кількість байт без підтвердження для вхідного трафіку
TCP_WIN_MAX_OUT	Максимальний розмір TCP-вікна (вихідний) – максимальна кількість байт без підтвердження для вихідного трафіку
ICMP_TYPE	Тип повідомлення ICMP – діагностичні або помилкові повідомлення (пінг, відстеження маршруту)
ICMP_IPV4_TYPE	Тип повідомлення ICMP для IPv4
DNS_QUERY_ID	Ідентифікатор запиту DNS – унікальний код для відповідності запитів і відповідей
DNS_QUERY_TYPE	Тип запиту DNS визначає тип запитуваної інформації
DNS_TTL_ANSWEAR	Час дії DNS-запису до його оновлення
FTP_COMMAND_RET_CODE	Код відповіді FTP – результат виконання команди FTP-сервером

Таблиця 2

Композиційна таблиця формування нечіткого правила для 1-го ешелону

L4_SRC_PORT	L4_DST_PORT	PROTOCOL	L7_PROTO	IN_BYTES	IN_PKTS	OUT_BYTES	OUT_PKTS	TPC_FLAGS	CLIENT_TPC_FLAGS	SERVER_TPC_FLAGS	FLOW_DURATION_MILLSECONDS	DURATION_IN	DURATION_OUT	MIN_TTL	MAX_TTL	LONGEST_FLOW_PKT	SHORTEST_FLOW_PKT	MIN_IP_PKT_LEN	MAX_IP_PKT_LEN	SRC_TO_DST_SECOND_BYTES	DST_TO_SRC_SECOND_BYTES	RETRANSMITTED_IN_BYTES	RETRANSMITTED_IN_PKTS	RETRANSMITTED_OUT_BYTES	RETRANSMITTED_OUT_PKTS	SRC_TO_DST_AVG_THROUGHPUT	DST_TO_SRC_AVG_THROUGHPUT	NUM_PKTS_UP_TO_128_BYTES	NUM_PKTS_128_TO_256_BYTES	NUM_PKTS_256_TO_512_BYTES	NUM_PKTS_512_TO_1024_BYTES	NUM_PKTS_1024_TO_1514_BYTES	TCP_WIN_MAX_IN	TCP_WIN_MAX_OUT	ICMP_TYPE	ICMP_IPV4_TYPE	DNS_QUERY_ID	DNS_QUERY_TYPE	DNS_TTL_ANSWEAR	FTP_COMMAND_RET_CODE
t_1 0 23053	t_1 0 22159	t_1 0 95	t_1 0 3	t_1 4 44997	t_1 1 611	t_1 0 48283600	t_1 0 243	t_1 0 2	t_1 0 2	t_1 0 4	t_1 0 2146670	t_1 0 562	t_1 0 32	t_4 200 255	t_3 200 255	t_1 28 16011	t_1 28 404	t_1 0 40	t_1 28 16011	t_1 40 1120056005600	t_1 0 106918656	t_1 0 14504	t_1 0 14	t_1 0 7015	t_1 0 16	t_1 0 359976000	t_1 0 1432816000	t_1 0 700	t_1 0 152	t_1 0 66	t_1 0 35	t_1 0 153	t_1 0 8192	t_1 0 8666	t_1 0 38893	t_1 0 152	t_1 0 23043	t_1 0 6	t_1 0 100	t_1 0 150
t_2 23160 65535	t_2 22218 65535	t_2 96 255	t_2 7 9	t_2 46345 539723	t_2 700 4200	t_2 53459328 106918656	t_2 260 2338	t_2 16 19	t_2 16 20	t_2 16 20	t_2 2146685 4294952	t_2 611 1624	t_2 46 63	t_1 0 45	t_1 0 64	t_2 32160 65212	t_2 440 1369	t_2 48 60	t_2 32160 65212	t_2 56002240005600 2240016800560000	t_2 160093726 320187451	t_2 39056 70551	t_2 17 29	t_2 131587 154139	t_2 96 116	t_2 370760000 4068696000	t_2 1516960000 4283280000	t_2 750 4200	t_2 168 695	t_2 73 206	t_2 45 73	t_2 157 897	t_2 16383 21216	t_2 16125 17155	t_2 40231 65280	t_2 157 255	t_1 23296 65282	t_2 16 46	t_2 80000 97000	t_2 156 331
			t_3 18 41					t_3 21 31	t_3 24 31	t_3 22 31			t_3 78 94	t_2 46 64	t_2 110 128			t_3 76 88		t_3 44800616005600 56006720016800				t_3 35 52	t_3 220639 241085	t_3 173 184						t_3 89 101		t_3 29200 35947	t_3 25787 29200				t_3 125 255		
			t_4 78 85										t_4 109 188	t_3 120 128				t_4 105 245		t_4 112002800039200 194324480016800					t_4 363369 383369	t_4 268 346								t_4 45260 65335	t_4 38235 55845				t_4 22769 32769		
			t_5 92 100										t_5 203 360					t_5 327 385		t_5 224001680056000 448006160056000					t_5 452969 457471										t_5 60328 65335						
			t_6 125 231										t_6 422 500					t_6 422 576		t_6 560022400056000 112006720056000.0																					

Таблиця 3

Композиційна таблиця формування нечіткого правила для 2-го ешелону

IN_PKTS	OUT_BYTES	OUT_PKTS	DURATION_IN	DURATION_OUT	LONGEST_FLOW_PKT	MAX_IP_PKT_LEN	SRC_TO_DST_SECOND_BYTES	DST_TO_SRC_SECOND_BYTES	RETRANSMITTED_IN_BYTES	RETRANSMITTED_IN_PKTS	RETRANSMITTED_OUT_BYTES	RETRANSMITTED_OUT_PKTS	SRC_TO_DST_AVG_THROUGHPUT	DST_TO_SRC_AVG_THROUGHPUT	NUM_PKTS_UP_TO_128_BYTES	NUM_PKTS_128_TO_256_BYTES	NUM_PKTS_1024_TO_1514_BYTES
t_1 1 611	t_1 0 48283600	t_1 0 243	t_1 0 562	t_1 0 32	t_1 28 16011	t_1 28 16011	t_1 40 1120056005600	t_1 0 106918656	t_1 0 14504	t_1 0 14	t_1 0 7015	t_1 0 16	t_1 0 359976000	t_1 0 1432816000	t_1 0 700	t_1 0 152	t_1 0 153
t_2 700 4200	t_2 53459328 106918656	t_2 260 2338	t_2 611 1624	t_2 46 63	t_2 32160 65212	t_2 32160 65212	t_2 56002240005600 2240016800560000	t_2 160093726 320187451	t_2 39056 70551	t_2 17 29	t_2 131587 154139	t_2 96 116	t_2 370760000 4068696000	t_2 1516960000 4283280000	t_2 750 4200	t_2 168 695	t_2 157 897
				t_3 78 94			t_3 44800616005600 56006720016800			t_3 35 52	t_3 220639 241085	t_3 173 184					
				t_4 109 188			t_4 112002800039200 194324480016800				t_4 363369 383369	t_4 268 346					
				t_5 203 360			t_5 224001680056000 448006160056000				t_5 452969 457471						
				t_6 422 500			t_6 560022400056000 11200672005600.0										

Таблиця 4

Композиційна таблиця генерації нового нечіткого правила

L4_SRC_PORT	L4_DST_PORT	PROTOCOL	L7_PROTO	IN_BYTES	IN_PKTS	OUT_BYTES	OUT_PKTS	TPC_FLAGS	CLIENT_TPC_FLAGS	SERVER_ TPC_FLAGS	FLOW_DURATION_MILLSECONDS	DURATION_IN	DURATION_OUT	MIN_TTL	MAX_TTL	LONGEST_FLOW_PKT	SHORTEST_FLOW_PKT	MIN_IP_PKT_LEN	MAX_IP_PKT_LEN	SRC_TO_DST_SECOND_BYTES	DST_TO_SRC_SECOND_BYTES	RETRANSMITTED_IN_BYTES	RETRANSMITTED_IN_PKTS	RETRANSMITTED_OUT_BYTES	RETRANSMITTED_OUT_PKTS	SRC_TO_DST_AVG_THROUGHPUT	DST_TO_SRC_AVG_THROUGHPUT	NUM_PKTS_UP_TO_128_BYTES	NUM_PKTS_128_TO_256_BYTES	NUM_PKTS_256_TO_512_BYTES	NUM_PKTS_512_TO_1024_BYTES	NUM_PKTS_1024_TO_1514_BYTES	TCP_WIN_MAX_IN	TCP_WIN_MAX_OUT	ICMP_TYPE	ICMP_IPV4_TYPE	DNS_QUERY_ID	DNS_QUERY_TYPE	DNS_TTL_ANSWEAR	FTP_COMMAND_RET_CODE
t_1 ∋ 23053	t_1 ∋ 22159	t_1 ∋ 95	t_1 ∋ 3	t_3 54671 57893	t_1 1 611	t_1 ∋ 48283600	t_1 ∋ 243	t_1 ∋ 2	t_1 ∋ 2	t_1 ∋ 4	t_3 443920 484435	t_1 ∋ 562	t_1 ∋ 32	t_4 200 255	t_3 200 255	t_1 28 16011	t_1 28 404	t_1 ∋ 40	t_1 28 16011	t_1 40 1120056005600	t_1 ∋ 106918656	t_1 ∋ 14504	t_1 ∋ 14	t_1 ∋ 7015	t_1 ∋ 16	t_1 ∋ 359976000	t_1 ∋ 1432816000	t_1 ∋ 700	t_1 ∋ 152	t_1 ∋ 66	t_1 ∋ 35	t_1 ∋ 153	t_1 ∋ 8192	t_1 ∋ 8666	t_1 ∋ 38893	t_1 ∋ 152	t_1 ∋ 23043	t_1 ∋ 6	t_2 1500 10000	t_2 150
t_2 23160 65535	t_2 22218 65535	t_2 96 255	t_2 7 9	t_1 4 44997	t_2 700 4200	t_2 53459328 106918656	t_2 260 2338	t_2 16 19	t_2 16 20	t_2 16 20	t_1 0 2146670	t_2 611 1624	t_2 46 63	t_1 0 45	t_1 0 64	t_2 32160 65212	t_2 440 1369	t_2 48 60	t_2 32160 65212	t_2 56002240005600 2240016800560000	t_2 160093726 320187451	t_2 39056 70551	t_2 17 29	t_2 131587 154139	t_2 96 116	t_2 370760000 4068696000	t_2 1516960000 4283280000	t_2 750 4200	t_2 168 695	t_2 73 206	t_2 45 73	t_2 157 897	t_2 16383 21216	t_2 16125 17155	t_2 40231 65280	t_2 157 255	t_1 23296 65282	t_2 16 46	t_1 0 100	t_2 156 331
			t_3 18 41	t_2 46345 539723				t_3 21 31	t_3 24 31	t_3 22 31	t_2 2146685 4294952		t_3 78 94	t_2 46 64	t_2 110 128			t_3 76 88		t_3 4480061600560 0 56006720016800			t_3 35 52	t_3 220639 241085	t_3 173 184						t_3 89 101		t_3 29200 35947	t_3 25787 29200				t_3 125 255	t_3 80000 87000	
			t_4 78 85										t_4 109 188	t_3 120 128				t_4 105 245		t_4 112002800039200 194324480016800				t_4 363369 383369	t_4 268 346							t_4 45260 65335	t_4 38235 55845				t_4 22769 32769		
			t_5 92 100										t_5 203 360					t_5 327 385		t_5 224001680056000 448006160056000				t_5 452969 457471									t_5 60328 65335						
			t_6 125 231										t_6 422 500					t_6 422 576		t_6 560022400056000 1120067200560 0.0																			

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Shimony E., Tsarfati O. Chatting our way into creating a polymorphic malware. *Identity Security and Access Management Leader. CyberArk*. URL:https://www.cyberark.com/resources/threat-research/chatting-our-way-into-creating-a-polymorphic-malware.

2. Фесьоха В., Кисиленко Д. Модель визначення інваріантної компоненти в поведінці шкідливого програмного забезпечення на основі інтеграції нечіткої логіки та генетичних алгоритмів. *Системи і технології зв’язку, інформатизації та кібербезпеки*. 2024. Vol. 1, no. 6. P. 232–242. URL: https://doi.org/10.58254/viti.6.2024.19.232.

3. Хандрос А. В., Ткач В. М. Застосування алгоритмів машинного навчання для детекції шкідливого програмного забезпечення через аналіз PE-заголовків. *Теоретичні і прикладні проблеми фізики, математики та інформатики: матеріали XXII Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених*. 2024. С. 179–181.

4. Domain adaptation-based deep learning framework for android malware detection across diverse distributions / S. Xiong et al. *Artificial intelligence advances*. 2024. Vol. 6, no. 1. P. 13–24. URL: https://doi.org/10.30564/aia.v6i1.6718.

5. MORPH: towards automated concept drift adaptation for malware detection / M. Alam et al. *ArXiv preprint arxiv:2401.12790*. 2024. URL: https://openreview.net/forum?id=Y5OSp4yq6X.

6. Varikuti H., Vatsavayi V. K. A hybrid malware detection framework with drift adaptation for timestamped data. *Journal of theoretical and applied information technology*. 2024. Vol. 102, no. 9. P. 4137–4144. URL: https://www.jatit.org/volumes/Vol102No9/35Vol102No9.pdf.

7. A lightweight malware detection model based on knowledge distillation / C. Miao et al. *Mathematics*. 2024. Vol. 12, no. 24. P. 4009. URL: https://doi.org/10.3390/math12244009 (date of access: 01.03.2025).

8. Semantic data representation for explainable windows malware detection models / P. Švec et al. 2024. URL: https://doi.org/10.13140/RG.2.2.29257.76648.

9. Фесьоха В. В., Кисиленко Д. Ю. Адаптивний підхід до виявлення шкідливого програмного забезпечення з властивістю самомодифікації. *IV Міжнародна науково-технічна конференція – ВІТІ імені Героїв Крут – 2024 «Системи і технології зв’язку, інформатизації та кібербезпеки: актуальні питання і тенденції розвитку»*. 2024. С. 143. URL: https://doi.org/10.61929/viti.mntk.4.2024.

10. ML-Based NIDS Datasets. *School of Information Technology and Electrical Engineering*. URL: https://staff.itee.uq.edu.au/marius/NIDS_datasets/#RA6.

11. Virus (Generic) | F-Secure Labs. *Schutz für digitale Momente | F-Secure*. URL: https://www.f-secure.com/v-descs/virus-w32-generic.shtml.