DOI 10.36074/logos-18.08.2023.40

ПОВЕДІНКОВІ РЕАКЦІЇ HONEYNET НА ПРИКЛАДАХ ХАРАКТЕРНИХ МЕРЕЖЕВИХ АТАК

ORCID ID: 0000-0002-0326-2394 **Колованова Євгенія Павлівна**
канд. техн. наук, доцент кафедри
Харківський національний університет імені В.Н. Каразіна

ORCID ID: 0000-0001-8826-1616 **Малахов Сергій Віталійович**
канд. техн. наук, ст. науковий співробітник, доцент кафедри
Харківський національний університет імені В.Н. Каразіна

ORCID ID: 0000-0002-1134-2925 **Мелкозьорова Ольга Михайлівна**
канд. техн. наук, доцент кафедри
Харківський національний університет імені В.Н. Каразіна

УКРАЇНА

Вступ. В роботі розглянуто особливості функціонування та впровадження технологій активного мережевого захисту на прикладі мережі *Honeynet*. Визначено можливі напрями використання технологій штучного інтелекту *(AI)* та машинного навчання *(ML)* для реалізації поведінкових реакцій кластерних мережевих пасток (*Honeypot)*, що входять до складу єдиної *Honeynet* системи.

Імітаційне моделювання зворотних реакцій складових *Honeynet*, виконано на прикладі реалізації спрощеної (тестової) мережевої структури в умовах відтворення кількох найбільш поширених різновидів мережевих атак. Наведено приклад системи фільтрації трафіку. Програмний імітатор *Honeynet*, реалізовано на мові програмування *Python*.

Основна частина. В загальному випадку можливості технологій активного захисту розглянуто в роботах [1-3]. Однак, з урахуванням специфіки поширення функцій мережевих пасток (*Honeypot, далі HP*) на відповідне мережеве утворення (функціональну підсистему), котра входить до інфраструктури інформаційної системи, яка захищається, є необхідність акцентувати увагу на певних аспектах, що обумовлені генезисом моделі активного захисту інформаційних ресурсів:

- активний захист, це модель захисту, яка передбачає використання технологій, що спонукають мережевих зловмисників до їх завчасного виявлення, ресурсного виснаження *(включаючи витрачений час)* та/чи введення в оману, замість того, щоб тільки захищати відповідну систему/ресурс від зовнішніх вторгнень;
- *HP*, це програмно-апаратне рішення, яке імітує/відтворює функціонування реального інформаційного ресурсу *(системи, що захищається)*. *HP* може передбачати кластерну реалізацію з єдиним, або розгалуженим центром управління;
- *HP*, котрі здатні адаптуватись під особливості функціонування їх мережевого оточення, без втручання людини, називають «*Dynamic Honeypots*»;
- технологія *Cyber Deception* [2] розширює ідеї *Honeypot* та передбачає використання додаткових структурних компонентів у якості приманки для

зловмисника *(безвідносно частки антропогенності джерела загроз, тобто: - людина, робот або зв'язка людина-машина...)*;

- кластерний принцип організації *HP* є подальшим розвитком практики використання одиночних мережевих пасток і забезпечує еволюційний перехід до мережевих інфраструктурних утворень (*Honeynet*) з підтримкою функцій структурної віртуалізації та самонавчання на основі імплементації можливостей штучного інтелекту [2];

- основним призначенням *HP* є, максимальне привертання уваги і ресурсних зусиль зловмисників *(включаючи залучені бот-системи)* для отримання більш повної інформації про використовувані інструменти й методики нападу та забезпечення виграшу часу для протидії можливим (вже очікуваним) атакам [4-5];

- за рахунок ширшого сценарного (поведінкового) поля, кластерні *HP* забезпечують більший діапазон активної протидії і мають кращу стійкість до їх компрометації та/чи деанонімізації їх функціонування.

Концептуально, основні вимоги до кластерної реалізації *Honeypot*, тобто *HoneyNet (*надалі *HN,* або система*)*, можна окреслити наступним чином:

- система повинна бути наділена відповідними механізмами взаємодії із нападником (*хакером та/чи атакуючою бот-системою*) задля створення у нього/неї хибного «уявлення» про успішність вже здійснених дій та «перспективність» реалізації наступних кроків;

- система повинна мати механізми сповіщення про нетипові інциденти безпеки і аномалії мережевого трафіку й підтримувати функції неявної (прихованої) протидії фактам ведення зовнішньої мережевої розвідці, та/чи спробам вторгнення у внутрішню структуру, відповідно до чинного сценарного поля відтворюваного аватару (-рів) *HP* [1];

- механізми активної протидії повинні бути «розміщені» у компоненті, що адмініструє доступом до інших ресурсів в структурі мережі *HN*;

- система повинна однозначно розрізняти трафік і поведінкові сигнатури «*порядних*» користувачів від аномального мережевого трафіку та/чи деструктивних дій з боку атакуючої сторони *(наприклад, тестування і «зняття» телеметрії роботи мережевого устаткування та ін.)*.

Загальна ідея та місце інтеграції *HN* в інформаційну структуру, що захищається, наведено на рис.1. В контексті зазначених вище тез, до найбільш принципових особливостей даної схеми слід віднести наступне:

- ***Firewall 1*** *(або вхідний маршрутизатор)* – обробляє трафік із «зовнішньої» мережі згідно із встановленими правилами *ACL*. Відповідно до цих правил, відбувається маршрутизація вхідних пакетів до комутатору у «внутрішньої» частині мережі (на рис.1 позначено, як *Firewall 2*) та/або до одного чи відразу кількох з активних *HoneyPot* (*фізичних та/чи віртуальних*) в межах цієї *HN*;

- ***Honeynet system*** – система-пастка, що може складатися з декількох окремих *Honeypot (з єдиним або автономними центрами управління)* та мати кластерну структуру з різним рівнем вкладеності та/чи розпаралелювання (*резервування складових*);

- центри управління *HN* можуть бути виконані локально й віддалено. Центри управління наявних/підтримуваних *Honeypot (включаючи віртуалізовані),* можуть підтримувати змішану схему управління, тобто частина з них передбачає виключно локальне адміністрування, а частина мати й зовнішній канал обслуговування/ініціювання їх роботи;

- *Firewall 2* – вихідний, відносно даної *HN,* міжмережевий екран або комутатор, що виступає своєрідним шлюзом, через який відбувається безпосередній доступ до ресурсів, що захищаються.

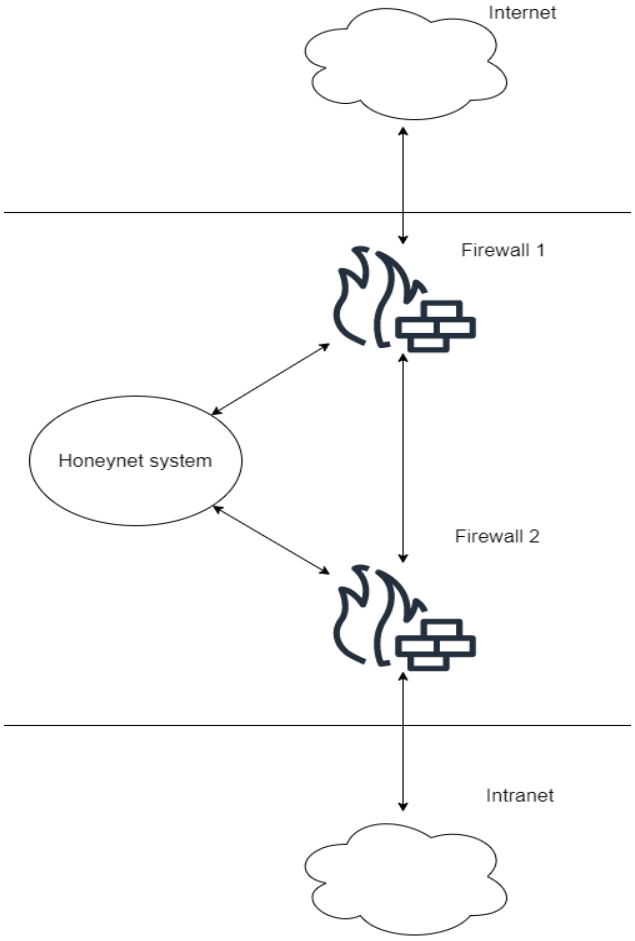

Рис. 1. **Спрощена структура HN**

З рис.1 слід, що елементи *Firewall 1* та *Firewall 2,* фактично, сегментують [6] канал зв'язку *(безвідносно фізичної природи середовища передачі даних/трафіку)* між зовнішньою і внутрішньою частинами взаємодіючих інформаційних систем, де *HN* виконує роль інтелектуального мережевого адаптеру, котрий відтворює потрібні поведінкові реакції персоналу й мережевого устаткування, які притаманні для різних варіантів наявного сценарного полю у межах відтворюваних поведінкових аватарів *Honeypot* (див. рис.1 в роботі [1]), та/або всього *HN,* тобто всієї системи.

Структура зв'язків і логіка взаємодії елементів інтегрованого модулю управління *Honeynet system,* коротко розглянуто в роботі [3]. Спрощена структурна схема *Honeynet,* що дозволяє відтворювати відповідні умови для вирішення завдань імітаційного моделювання процедур активної моделі захисту інформаційних ресурсів та досліджувати процеси циркуляції тестового мережевого трафіку, наведено на рис.2. Дана схема не враховує особливості, стосовно багатофакторної сегментації наявних ресурсів та/чи мережевого трафіку [6], але надає потрібні умови для моделювання поведінкових реакцій *HN* для різних типів атакуючих дій та/або нештатної (недекларованої) мережевої активності користувачів [7].

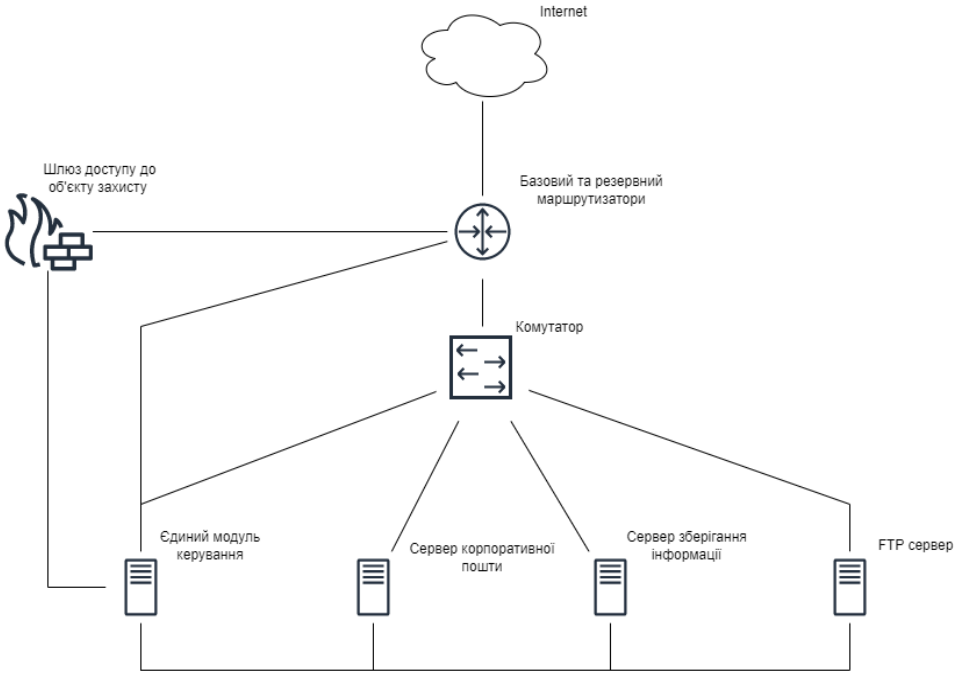

Рис. 2. **Імітаційна структура мережі HoneyNet** *(Варіант)*

Сценарне поле *HP* або *HN* містить набір дій, що є сукупністю зворотних реакцій системи на потенційні дії атакуючої сторони. Задля створення й підтримки ефективного сценарного полю *HN* потрібно своєчасно визначати і періодично корегувати (*доповнювати*) відповідний лист загроз безпеки [8-13] та реакцій системи на кожну із них. Коректно сформоване та оновлюване сценарне поле *HP/HN,* дозволяє ефективно протидіяти атакам й створювати відповідні багатоходові ланцюжки зворотних дій системи активного захисту, щодо протидії різним комбінаціям атак, наприклад: - сповільнення та/або перенаправлення трафіку; - блокування трафіку від конкретного вузлу *(IP та/чи MAC адреси)*; - хибна (підставна) авторизація тощо. Більш докладно о цілях й принципах використання поведінкових профілів для створюваного аватару *HP*, розглянуто в [1]. Приклади реалізації правил *ACL* (*Access Control Lists* - списки доступу, або *prefix-list)* для *HP* при вирішенні завдань парірування типових атак, наведено в табл.1-5 *(авторська версія)*.

Таблиця 1

Приклад таблиці *ACL* для *вхідного* інтерфейсу *HP* *(Варіант)*

Номер правила	Запис у маршрутизаторі	Зміст процедури
1	***access-list 1 permit any any***	Дозволити будь-який трафік від будь-якого джерела до будь-якого пункту призначення

Таблиця 2

ACL для міжелементної взаємодії HP *(Варіант)*

Номер правила	Запис у комутаторі	Зміст процедури
1	***access-list 1 permit IP host 192.168.3.111 host 192.168.3.113***	Дозволити будь-який трафік між *«Модулем керування»* та *«Сервером зберігання інформації»* (*вар.*).

Продовження табл. 2

Номер правила	Запис у комутаторі	Зміст процедури
2	***access-list 1 deny 192.168.0.0 0.0.0.255***	Заборонити будь-який трафік між вузлами однієї приватної мережі крім *«Модулю керування HP» (наприклад з «Сервером зберігання інформації»).*
3	***access-list 1 permit any any***	Дозволити будь-який трафік від будь-якого джерела до будь-якого елементу *(вузлу)* призначення.

Таблиця 3

ACL для випадку здійснення ARP *spoofing* *(Варіант)*

Номер правила	Запис у комутаторі	Зміст процедури
1	***access-list 1 deny arp host <IP-адреса атакуючого> any***	Заборонити проходження трафіку, що містить ознаки *ARP*.
2	***access-list 1 permit any any***	Дозволити будь-який трафік від будь-якого джерела до будь-якого елементу *(вузлу)* призначення.

Таблиця 4

ACL для випадку DoS атаки *(Варіант)*

Номер правила	Запис у комутаторі	Зміст процедури
1	***access-list 1 deny tcp host <IP-адреса атакуючого> any fragments***	Заборонити трафік, що складається із фрагментованих *TCP* пакетів.
2	***access-list 1 deny icmp host <IP-адреса хакеру> any packet-too-big***	Заборонити *ICMP* трафік, занадто великого розміру.
3	***access-list 1 permit any any***	Дозволити будь-який трафік від будь-якого джерела до елементу *(вузлу)* призначення.

Таблиця 5

ACL для випадку протидії атакам типу XSS *(Варіант)*

Номер правила	Запис у комутаторі	Зміст процедури
1	***access-list 1 deny tcp host <IP-адреса атакуючого> any eq www method <назва методу>***	Заборонити доступ до певних методів протоколу HTTP *(наприклад, POST, PUT, GET, PATCH та ін.).*
2	***access-list 1 deny tcp host <IP-адреса атакуючого> any eq www host <IP-адреса хосту>***	Заборонити доступ до певних *IP* адрес.
3	***access-list 1 deny tcp host <IP-адреса атакуючого> any eq www header User-Agent "malicious-agent"***	Заборонити доступ до певних заголовків *HTTP.*
4	***access-list 1 permit any any***	Дозволити будь-який трафік від будь-якого джерела до будь-якого елементу *(вузлу)* призначення.

```python
def teardrop_check(ip_header_array):
    Teardrop = False

    for i in range(len(ip_header_array) - 1):
        current_packet = ip_header_array[i]
        next_packet = ip_header_array[i + 1]

        packet_shift = next_packet - current_packet

        if packet_shift != 1:
            Teardrop = True
            break

    return correct_shifts
```

Ln: 7, Col: 9

Рис. 3. **Алгоритм виявлення для *Teardrop***

Приклади реалізації виявлення аномалій мережевого трафіку для атак типу *Teardrop* та *XSS*, тобто міжсайтовий скриптинг, наведено на рис.3-4.

```python
import re

def is_XSS(packet_data):
    # Преобразование пакета в строку
    packet_str = packet_data.decode('utf-8')

    # Поиск HTTP-запросов в пакете
    http_requests = re.findall(r'(GET|POST|PUT|DELETE|HEAD) (.*?) HTTP/1.[01]', packet_str)

    for method, path in http_requests:
        # Извлечение параметров из пути запроса
        params = re.findall(r'[\?&]([^=]+)=([^&]+)', path)
        for param, value in params:
            # Проверка значения параметра на наличие потенциально опасных символов или шаблонов
            if re.search(r'<script|javascript:|on\w+=', value):
                # Потенциальная XSS-атака
                return True

    return False
```

Рис. 4. **Алгоритм виявлення для атак *XSS*** ***(варіант)***

Приклади реалізації виявлення аномалій мережевого трафіку для атак типу *ARP-spoofing* та *Path traversal*, наведено на рис.5-6.

```python
import struct
import socket

def is_ARP_spoofing(eth_frame_data):
    # Чтение заголовка Ethernet-кадра
    eth_header = eth_frame_data[:14]
    eth_header_unpack = struct.unpack('!6s6sH', eth_header)
    destination_mac = eth_header_unpack[0]
    source_mac = eth_header_unpack[1]
    eth_type = eth_header_unpack[2]

    # Проверка типа Ethernet-кадра на ARP
    if eth_type == 0x0806:
        # Чтение заголовка ARP
        arp_header = eth_frame_data[14:42]
        arp_header_unpack = struct.unpack('2s2s1s1s2s6s4s6s4s', arp_header)
        arp_opcode = arp_header_unpack[4]
        arp_sender_mac = arp_header_unpack[5]
        arp_sender_ip = socket.inet_ntoa(arp_header_unpack[6])

        # Проверка корректности ARP-запроса или ответа
        if arp_opcode == b'\x00\x01' and arp_sender_mac != source_mac:
            # Потенциальная атака ARP Spoofing
            return True
        if arp_opcode == b'\x00\x02' and arp_sender_mac != source_mac:
            # Потенциальная атака ARP Spoofing
            return True

    return False

```

Рис. 5. **Алгоритм виявлення для атак типу *ARP-spoofing*** *(варіант)*

```python
import re

def is_path_traversal(packet_data):
    # Преобразование пакета в строку
    packet_str = packet_data.decode('utf-8')

    # Поиск HTTP-запросов в пакете
    http_requests = re.findall(r'(GET|POST|PUT|DELETE|HEAD) (.*?) HTTP/1.[01]', packet_str)

    for method, path in http_requests:
        # Проверка пути запроса на наличие потенциально опасных шаблонов
        if re.search(r'\.\./|\.\.\\|/\.\./|/etc/passwd|C:\\\\Windows\\\\System32', path):
            # Потенциальная атака Path Traversal
            return True

    return False
```

Рис. 6. **Алгоритм виявлення для атак *Path traversal*** *(варіант)*

Висновки.

1. Розглянутий приклад програмного імітатору *Honeynet*, забезпечує імплементацію концепції активного мережевого захисту, що враховує особливості топології і специфіку використання мережевих ресурсів, які захищаються.

2. Ефективність використання *Honeypot* (як складових системи *HN*), можливо значно покращити за рахунок впровадження в дійсні логічні ланцюги алгоритмів управління *HN*, елементів *AI* та машинного навчання.

3. Існуючі зразки *Honeypot*, що використовують елементи *AI* та *LM*, в своєї більшості, є пропрієтарними і не підлягають незалежному аудиту. Тому, використання таких систем повністю «базується» на довірі до їх розробника.

4. Альтернативою пропрієтарним *Honeypot* є створення й впровадження *Open Source* систем із підтримкою гнучкого конфігурування, як окремих складових елементів *Honeypot*, так і всієї системи (*Honeynet*) в цілому.

5. Недоліком концепції активного захисту слід вважати наявність в складі *HN* додаткових елементів захисту, котрі ускладнюють обслуговування всієї системи та, потенційно, можуть представляти інтерес для кіберзлочинців (що залежить від кваліфікації атакуючого), що може бути менш ефективно із точки зору безпеки ніж пасивні методи захисту. Тому слід розглядати таку модель як корисне доповнення до традиційних методів захисту, задля посилення загального рівня захисту від інформаційних загроз. Крім того, напрацювання в галузі *AI* та *LM*, помітно зменшує навантаження з обслуговуючого персоналу, концентруючи його увагу тільки на найбільш неоднозначних ситуаціях.

Список використаних джерел:

[1] Кохановська, Т., Нарежний, О., & Дьяченко, О. (2020). Дослідження можливостей технології Honeypot. *Комп'ютерні науки та кібербезпека*, 1(1), 33-42. Вилучено з URL: http://surl.li/hglmt

[2] Михайленко, Д., Чорна, Т. & Малахов, С. Використання можливостей AI при реалізації Static та Dynamic Honeypot для покращення параметрів захисту інформаційних ресурсів. *Технології, інструменти та стратегії реалізації наукових досліджень:* матеріали IV *Міжнародної наукової конференції,* (с. 54-57). 7.10.2022 р. Суми, Україна: МЦНД. DOI 10.36074/mcnd-07.10.2022

[3] Михайленко Д., Нємцев М. Особливості технології мережевих пасток як інструменту активного захисту та аналізу дій атакуючої сторони. Proceedings of the XXI International Scientific and Practical Conference. Melbourne, Australia – 2023 – Pp. 483-487. URL: https://isg-konf.com/scientists-and-methods-of-using-modern-technologies/

[4] Рузудженк, С., Погоріла, К., Кохановська, Т., & Малахов, С. (2020). Особливості захисту корпоративних ресурсів за допомогою технології Honeypot. Комп'ютерні науки та кібербезпека, (4), 22-29. Вилучено з URL: https://periodicals.karazin.ua/cscs/article/view/15751/14600

[5] CrowdStrike team. HONEYPOTS IN CYBERSECURITY EXPLAINED [Електронний ресурс] – 2022. Вилучено з: http://surl.li/hgncb

[6] Джон Маллери, & Джейсон Занн (2007). *Безопасная сеть вашей компании.* (Е. Линдеманн, пер. с англ.). Москва: НТ Пресс.

[7] Азаров, С., Нємцев, М., & Малахов, С. Огляд аналогій та обгрунтування принципів створення демон юнітів відстеження мережевої активності користувачів. Proceedings of the XX International Scientific and Practical Conference. Graz, Austria. 2023. Pp. 447-453. URL: https://isg-konf.com/technologies-innovative-and-modern-theories-of-scientists/

[8] Яремчук, К., Воскобойников, Д., & Мелкозьорова, О. (2022). Сучасні загрози та способи забезпечення безпеки веб-застосунків. *Комп'ютерні науки та кібербезпека*, (2), 28-34. https://periodicals.karazin.ua/cscs/article/view/21038/19744

[9] Богданова, Є., Чорна, Т., & Малахов, С. (2022). Огляд поточного стану загроз, що обумовлені впливом експлойтів. *Комп'ютерні науки та кібербезпека*, (2), 35-40. http://surl.li/kaafo

[10] Лахтін, І., Михайленко, Д., & Нарєжній, О. (2022). Порівняння комерційних сканерів вразливостей веб-додатків та сканерів з відкритим кодом. *Комп'ютерні науки та кібербезпека*, (2), 41-49. https://periodicals.karazin.ua/cscs/article/view/21040/19746

[11] Рондалєв, Д., Мелкозьорова, О., & Нарєжній, О. (2019). Особливості функціонування корпоративного міжмережевого екрану та питання взаємодії з системою IDS. *Комп'ютерні науки та кібербезпека*, (3), 11-21. https://periodicals.karazin.ua/cscs/article/view/15614/14707

[12] Рогоза, П., & Єсін, В. (2022). Використання нейронної мережі замість бази знань у експертній системі детектору зловмисного трафіку до веб-ресурсів. *Комп'ютерні науки та кібербезпека*, (1), 6-15. https://periodicals.karazin.ua/cscs/article/view/20908/19612

[13] Попов, Ю., Рузудженк, С., & Погоріла, К. (2019). SQL-ін'єкції: огляд потенційних способів захисту. *Комп'ютерні науки та кібербезпека*, (3), 22-26. https://periodicals.karazin.ua/cscs/article/view/15615/14710