

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи магістра

на тему: «Методи і технології підвищення рівня захищеності
системи "Розумний дім" на основі IoT honeypot»

Виконав студент 2-го курсу групи КБм-22-1
спеціальності 125 «Кібербезпека»



Д.С. Морозов

Керівник професор кафедри комп'ютерної інженерії та
кібербезпеки, доктор технічних наук, доцент



Ю.М. Россінський

Рецензент професор кафедри комп'ютерної інженерії та
кібербезпеки, доктор технічних наук, доцент



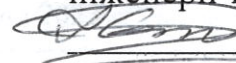
В.В. Воротніков

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
 Факультет інформаційно-комп'ютерних технологій
 Кафедра комп'ютерної інженерії та кібербезпеки
 Спеціальність 125 «Кібербезпека»
 Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко
13 жовтня 2023 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу магістра

Студента: Морозова Дмитра Сергійовича

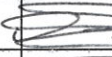
Тема роботи: «Методи і технології підвищення рівня захищеності системи "Розумний дім" на основі IoT honeypot»

затверджена Наказом університету від «13» жовтня 2023 р. № 572/с

Термін здачі студентом закінченої роботи 25 грудня 2023 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження): _____

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	А.А. Єфіменко	 05.09.23	 05.09.23
2	А.А. Єфіменко	 12.10.23	 12.10.23
3	А.А. Єфіменко	 28.10.23	 28.10.23
4	А.А. Єфіменко	 15.11.23	 15.11.23
5	А.А. Єфіменко	 27.11.23	 27.11.23

Студент  Д.С. Морозов

Керівник  Ю.М. Россінський

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі. Пошук і аналіз публікацій за тематикою дослідження. Формулювання технічного завдання. Опрацювання літературних джерел	5 вересня 2023 - 11 жовтня 2023	
2	Огляд і аналіз застосування IoT honeypot як засобів посилення захищеності IoT систем	12 жовтня 2023 – 27 жовтня 2023	
3	Розгортання і налаштування системи безпеки для «Розумного дому» на основі IoT honeypot	28 жовтня 2023 – 14 листопада 2023	
4	Дослідження атак на IoT honeypot системи «Розумний дім». Аналіз матеріалів по атакам зібраним за допомогою honeypot	15 листопада 2023 – 26 листопада 2023	
5	Розробка, розгортання і тестування telegram-боту як складової системи моніторингу IoT honeypot «Розумного дому»	27 листопада 2023 – 12 грудня 2023	
6	Захист	26 грудня 2023	

Студент  Д.С. Морозов

Керівник  Ю.М. Россінський

РЕФЕРАТ

Випускна кваліфікаційна робота магістра складається з проекту IoT honeypot системи для «Розумного будинку» та пояснювальної записки. Пояснювальна записка містить 108 сторінок, 18 ілюстрацій та 5 таблиць.

Метою роботи є підвищення безпеки системи «Розумного будинку» шляхом створення системи IoT honeypot для виявлення, попередження і інформування користувача про можливі атаки злоумисників на систему «Розумний будинок».

В роботі визначено основні завдання на розробку IoT honeypot системи, проаналізовано переваги і недоліки honeypot систем для виявлення вторгнень. Представлено схему розгортання і налаштування системи керування «Розумним будинком» на основі платформи Raspberry Pi, описано розгортання і налаштування ssh-honeypot Cowrie, а також розроблено telegram-бот для збільшення швидкості інформування користувача про можливі атаки на систему «Розумний дім». В роботі наводяться результати виявлених атак на систему «Розумний дім» під час тестування розгорнутої системи IoT honeypot і аналіз цих атак.

КЛЮЧОВІ СЛОВА: IOT, HONEYPOT, RASPBERRY PI, АТАКИ НА IOT, ВИЯВЛЕННЯ ВТОРГНЕНЬ

ABSTRACT

The master's thesis consists of a project of an IoT honeypot system for a "Smart Home" and an explanatory note. The explanatory note contains 108 pages, 18 illustrations and 5 tables.

The purpose of the work is to improve the security of the Smart Home system by creating an IoT honeypot system to detect, warn and inform the user about possible attacks on the Smart Home system.

The paper defines the main tasks for the development of the IoT honeypot system, analyzes the advantages and disadvantages of honeypot systems for intrusion detection. The paper presents a scheme for deploying and configuring a smart home control system based on the Raspberry Pi platform, describes the deployment and configuration of the Cowrie ssh-honeypot, and develops a telegram bot to increase the speed of informing the user about possible attacks on the smart home system. The paper presents the results of the detected attacks on the Smart Home system during the testing of the deployed IoT honeypot system and the analysis of these attacks.

KEYWORDS: IOT, HONEYPOT, RASPBERRY PI, ATTACKS ON IOT, INTRUSION DETECTION

ЗМІСТ

ВСТУП.....	9
РОЗДІЛ 1. ОГЛЯД І АНАЛІЗ КІБЕРЗАГРОЗ ДЛЯ «ІНТЕРНЕТУ РЕЧЕЙ»	12
1.1 Сучасні виклики і загрози кібербезпеки для України.....	12
1.2 Кібербезпека інтернету речей.....	14
1.3 Огляд загроз для IoT.....	22
1.4 Основні вразливості і напрямки атак IoT.....	26
Висновок до розділу 1	33
РОЗДІЛ 2. HONEYPOT ЯК ЕЛЕМЕНТ ПОСИЛЕННЯ КІБЕРБЕЗПЕКИ ІОТ СИСТЕМИ.....	34
2.1 Огляд досліджень використання honeypot для IoT.....	34
2.2 Використання honeypot як інструменту виявлення атак.....	38
2.3 Переваги і недоліки honeypot'ів	46
2.4 Огляд IoT honeypot'ів.....	49
2.5 Honeypot'и специфічні для шкідливого програмного забезпечення.....	53
Висновок до розділу 2.....	57
РОЗДІЛ 3. РОЗГОРТАННЯ І НАЛАШТУВАННЯ HONEYPOT'ІВ ДЛЯ ДОСЛІДЖЕННЯ БЕЗПЕКИ СИСТЕМИ «РОЗУМНИЙ ДІМ».....	59
3.1 Вибір honeypot для системи «Розумний дім».....	59
3.2 Розгортання і налаштування платформи та IoT honeypot.....	61
3.3 Розгортання і налаштування Cowrie.....	63
Висновок до розділу 3	69
РОЗДІЛ 4. АНАЛІЗ АТАК НА RASPBERRY PI SSH-HONEY- POT.....	70
Висновок до розділу 4	77
РОЗДІЛ 5. НАПИСАННЯ TELEGRAM-БОТУ ДЛЯ РЕАГУВАННЯ НА СПРОБИ ВЗЛОМУ COWRIE.....	78

5.1 Логування і відображення інформації про івенти в honeypot'і Cowrie.....	78
5.2 Розробка telegram-боту для відображення інформації про атаки на honeypot і реагування на них.....	82
Висновок до розділу 5	92
ВИСНОВКИ.....	93
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	96
ДОДАТКИ.....	103

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IoT	Internet of Things
ssh	Secure Shell Protocol
TTP	Tactics, techniques, and procedures

ВСТУП

Напрямки атак, методи проникнення і інструментарії кіберзловмисників постійно швидко еволюціонують. Поряд з загрозами атаками на державні установи, комерційні підприємства та дані користувачів існують стійкі загальносвітові тенденції до зростання кількості атак на пристрої IoT та різноманітних вбудованих систем. Цей напрямок кібератак хоч і не є настільки загрозливим для України в короткострокову перспективу як атаки на критичну інфраструктуру, проте, зважаючи на все ширше застосування інтернету речей в медицині, промисловості і комунальній інфраструктурі, він є доволі перспективним для кіберзлочинців з точки зору співвідношення зусиль на реалізацію таких атак до хаосу і збитків спричиненими цими атаками.

Відновлення і відбудова інфраструктури, промислових підприємств, муніципальних і транспортних систем України як в теперішній час так і після війни вимагатиме значного переосмислення підходів до їх побудови для більш ефективного використання ресурсів. Базуючись на кращих підходах останніх десятиліть і загальносвітових тенденцій для побудови нових і вдосконалення вже існуючих об'єктів інфраструктури будуть використовуватись значна кількість промислових і муніципальних рішень інтернету речей. А це, в свою чергу, створюватиме додаткові потенційні площини для атак кіберзлочинців. Тому пошук рішень по виявленню, аналізу і протидії цим загрозам стає вкрай актуальним.

Це загальносвітова тенденція, що формується останнє десятиліття. В більшості оглядів і досліджень кібербезпекових загроз зазначається, що майже кожна організація незабаром зіткнеться з проблемою кібербезпеки Інтернету речей безпосередньо у своїй корпоративній мережі або через третю сторону через ланцюжок поставок — якщо вона ще не постраждала.

Пристрої IoT піддаються в середньому 5200 кібератак на місяць [1]. За прогнозами аналітиків, до 2025 року буде від 20 до понад 25 мільярдів підключених пристроїв, від ноутбуків і медичних пристроїв до розумних замків, розумних приладів і розумних термостатів. Оскільки зазвичай ці пристрої мають обмежену об-

числювальну потужність і часто не мають вбудованих засобів захисту, вони особливо вразливі для хакерських атак, які намагаються отримати доступ до мережі. Оскільки Інтернет речей швидко поширюється на наше особисте та професійне життя, потенційна поверхня атак стає дедалі більшою.

У річному звіті Microsoft також зазначається загрозливо зростаючий список пристроїв з підключенням до Інтернету та Інтернету речей (IoT), які стають улюбленою мішенню хакерів через відсутність вбудованих елементів керування безпекою [2]. Інцидент кібербезпеки CommonSpirit Health змушує IT-системи працювати в автономному режимі. Згідно зі звітом, атаки на пристрої віддаленого керування неухильно зростали з червня 2021 року. Характер і напрямок цих атак постійно еволюціонує. Так минулого року спостерігалось значне зниження атак на звичайні протоколи IoT, наприклад Telnet, у деяких випадках на 60%. У той же час бот-мережі були перепрофільовані групами кіберзлочинців і національними державними акторами. Водночас деякі загрози залишаються стабільно високими вже декілька років. Стійкість шкідливих програм, таких як Mirai, підкреслює модульність цих атак і адаптивність до шаходів безпеки. Відповідно до звіту Microsoft Digital Defense Report, Mirai, який кілька разів перероблявся для адаптації до різних архітектур, заразив широкий спектр пристроїв IoT, включаючи камери Інтернет-протоколу, цифрові відеореєстратори камер безпеки та маршрутизатори.

Подібні загрози вимагають особливої уваги до захисту мереж підприємств і організацій, що використовують пристрої IoT в своїй діяльності. Програмні та архітектурні вразливості технологій, складність контролю рівня безпеки IoT мережі, що побудована на пристроях від різних вендорів, а головне адаптивність і постійне вдосконалення атак зловмисників на такі мережі – все це змушує шукати нові і дієві способи та інструменти захисту IoT мереж. Системні проблеми з безпекою пристроїв інтернету речей і мереж, що їх використовують, призводять до збільшення поверхні атаки такої мережі і труднощів її контролю з боку традиційних IDS систем.

Актуальність роботи полягає в тому, що використання сучасних підходів до побудови honeypot мереж для IoT є дієвим інструментом для посилення контролю

за діями зловмисників на етапі підготовки та проведення атак як на мережі інтернету речей так і на системи «Розумний будинок» зокрема.

Метою цього проекту є підвищення безпеки системи «Розумний будинок» шляхом побудови системи IoT honeypot для виявлення на атаки зловмисників та подальшого аналізу їх технік, тактик і процедур.

Для досягнення поставлених цілей розроблено і виконано наступні завдання:

- 1) Визначити основні кібербезпекові загрози для України в 2023 році та тенденції до їх змін стосовно атак на пристрої Інтернету речей.
- 2) Проаналізувати основні тенденції до використання honeypot систем для покращення безпеки IoT систем.
- 3) Проаналізувати і обрати IoT honeypot або низку honeypot систем для використанні їх в системі безпеки «Розумного дому».
- 4) Розгорнути, налаштувати і перевірити працездатність IoT honeypot системи для посилення безпеки системи «Розумного дому».
- 5) Проаналізувати результати виявлених атак на IoT honeypot систему безпеки «Розумного дому».

Об'єктом дослідження є процес посилення системи безпеки «Розумного дому».

Предметом дослідження є сучасні IoT honeypot системи для посилення безпеки «Розумного дому»

Практичне значення одержаного результату є високим, адже дозволяє ефективніше використовувати сучасні методи і технології захисту IoT систем взагалі і систем «Розумний будинок» зокрема, використовуючі сучасні підходи до побудови систем IoT honeypot.

Результати дослідження були опубліковані на конференції Proceedings of the 3rd Edge Computing Workshop (Morozov D.S., Vakaliuk T.A., Yefimenko A.A., Nitchuk T.M., Kolomiets R.O. Honeypot and cyber deception as a tool for detecting cyber attacks on critical infrastructure. Proceedings of the 3rd Edge Computing Workshop. Zhytomyr, Ukraine, April 7, 2023. Vol. 3374. P. 81-96.)

РОЗДІЛ 1. ОГЛЯД І АНАЛІЗ КІБЕРЗАГРОЗ ДЛЯ «ІНТЕРНЕТУ РЕЧЕЙ»

1.1 Сучасні виклики і загрози кібербезпеки для України

Щороку кількість загроз у сфері кібербезпеки невинно збільшується. Останні роки не були в цьому виключенням. Існує ряд загальносвітових тенденцій, серед яких слід зазначити традиційне зростання кількості атак програм-вимагачів, витоків даних і атак на критично важливу інфраструктуру. Проте, зараз найбільший вплив на глобальну кібербезпеку вносить вторгнення Росії в Україну.

24 лютого 2022 року світ кібербезпеки увійшов у епоху гібридної війни. За кілька годин до того, як були запущені ракети і колонни агресора перейшли через кордони, російські хакери здійснили масову руйнівну кібератаку проти українського уряду, технологічних компаній і фінансового сектора. Саме початок війни в кіберпросторі, як складова війни на землі, спричинив істотні зміни як в кількості так і в напрямках спричинених війною кібератак.

Поряд зі значною кількістю груп зловмисників за якими прямо чи опосередковано стоять держави з'явилась значна кількість загроз, що породжені діями «патріотичного хактивізму». Подібні атаки здійснюються доволі великою групою людей, що раніше жодним чином не була дотична до кіберзлочинності. Вони мають низьку кваліфікацію і зазвичай їм притаманні тільки найпростіші тактики і інструменти кіберзлочинців. Це, в свою чергу, накладає певні обмеження на вибір ними потенційних цілей і на вектори їх атак. Прикладом подібної діяльності є значні сплески DDOS-атак в 2022 році, «дефейсів», спорадичних і слабоскоординованих атак на адміністративні установи, що несли здебільшого психологічний вплив і зазвичай намагалися спричинити хаос в суспільстві ніж задати прямої економічної чи військової шкоди [3].

Зміна тактик, технік і процедур, поява нових загроз і напрямків атак відбуваються дуже динамічно і потребують швидкого виявлення і реагування. Так, в звіті за перший квартал 2023 року, SCPC (The State Cyber Protection Center of Ukraine)

зазначає, що порівняно з IV кварталом 2022, зафіксоване зменшення загальної кількості кібератак, організовуваних угрупованнями проросійських хактивістів, проте їхня систематичність та інтенсивність продовжують залишатись на високому рівні. Водночас від початку календарного року, порівняно з попереднім кварталом, у 1.5-2.9 рази для різних секторів знизилась кількість атак, організованих проросійськими угрупованнями хактивістів, націлених на комерційний, фінансовий сектор, Уряд та місцеві органи влади, сектор безпеки та оборони. При цьому інтенсивність атак на сектор енергетики та ЗМІ залишається на тому ж рівні [4].

Вдосконалення інструментарію виявлення, аналізу, вивчення і протидії кіберзагрозам має стати постійною задачею для фахівців відділів кібербезпеки не тільки під час конвенційної війни Росії проти України, а і після її завершення. The Computer Emergency Response Team of Ukraine (CERT-UA) under the SSSCIP (State Service of Special Communications and Information Protection of Ukraine) зазначає, що такі проти ІТ-інфраструктури можуть тривати ще багато років і навіть ті організації, в яких захист побудований за найкращими стандартами, повинні постійно переоцінювати ризики, тестувати і симулювати різні сценарії проникнення та можливості виявлення атак, оновлювати власні знання та документувати процеси, оскільки ворожі хакери використовують весь наявний арсенал та невинно еволюціонують [5].

Посилення кіберзахисту мереж критичної інфраструктури, державних і приватних підприємств і організацій України значно ускладнюють здійснення атак для російських кіберзлочинців. Проте еволюція їх тактик і інструментаріїв також постійно адаптується, а зловмисники активно шукають шляхи для здійснення своєї деструктивної діяльності. Так за результатами першого півріччя 2023 року SSSCIP (State Service of Special Communications and Information Protection of Ukraine) зазначає, що експерти також зафіксували збільшення кількості кібератак на комерційні структури протягом березня та квітня: майже вдвічі більше, ніж у січні-лютому 2023 року [6], а від середини минулого року прослідковується тенденція зростання кількості атак на ланцюжок постачання (supply chain attacks). Йдеться, зокрема, про ІТ-компанії, що розробляють рішення для критичної інфраструктури [7].

Така швидка динаміка загроз змушує вносити швидкі зміни як в роботу державних так і приватних установ в галузі кібербезпеки та шукати дієві способи протидії новим загрозам. Саме швидкість адаптації нових підходів і імплементації найкращих світових практик багато в чому дозволили уникнути більших збитків і руйнівних наслідків кібератак. Аналіз, вивчення і систематизація інформації про алгоритми, шляхи та технології, що використовувались в цих атаках і будуть з великою імовірністю застосовані знову є важливим фактором в побудові гнучкої і адаптивної системи кібербезпеки як на рівні держави так і на рівні окремих підприємств та організацій.

Слід розуміти, що збільшення кількості кіберзлочинців, що долучаються до атак на різноманітні об'єкти в Україні та за її межами, а також поступове набуття ними практичних навичок проведення атак на об'єкти певного типу і спеціалізація окремих угруповань на проведенні специфічних кібератак призводить до постійного зростання складності цих атак. Проводиться постійний пошук нових точок прикладення сили і постійне вдосконалення технік, тактик і процедур російських і афільованих з ними кіберзлочинців. На даний момент важко спрогнозувати наскільки руйнівними будуть ці атаки, а це змушує інтенсифікувати шляхи і способи захисту мережевих інфраструктур для об'єктів різного призначення і форм власності.

З високою імовірністю в подальшому до традиційних і цілеспрямованих атак на підприємства та організації України все частіше додаватимуться різноманітні високоавтоматизовані атаки на менш критично важливі об'єкти як IoT пристрої, системи «Розумний дім», різноманітні системи муніципального IoT тощо. Низький рівень захищеності, масовість і відносно прості способи атак подібних систем вже давно зробили їх легкою ціллю для світової кіберзлочинності.

1.2 Кібербезпека інтернету речей

За останнє десятиліття "Інтернет речей" (IoT) став звичною, доступною і популярною технологією. Це постійно зростаюча мережа фізичних об'єктів, які використовують IP-протокол для підключення до Інтернету, а також комунікацію, яка відбувається між цими об'єктами та іншими пристроями і системами з доступом до Інтернету. Ці вбудовані пристрої часто невеликі, з обмеженим енергоспоживанням і пам'яттю, і підключаються за допомогою певної бездротової технології. Сфера застосування Інтернету речей широка через його універсальний і неоднорідний характер, гетерогенну природу. Виробники постійно пропонують нові IoT рішення як для споживачів, так і для промисловості. Повсякденні предмети, такі як холодильники, кавоварки та лампочки тепер стають частинами типових розумних будинків, де кінцевий користувач дистанційно керує і контролює кожен пристрій.

Системи управління та виробництва також отримують вигоду від розширення Інтернету речей, підвищуючи ефективність повсякденних процесів, операцій та процедур.

Для кращого усвідомлення глибини проникнення IoT пристроїв в наше життя можна розглянути наступні приклади:

1. Датчики температури по всьому будинку, підключені до Wi-Fi, щоб ефективніше контролювати опалення в будинку.

2. IP-камери тепер підключаються до Інтернету і стають віддалено доступними через додаток з будь-якої точки світу для посилення безпеки помешкання.

3. Існуючі концепції освітлення і енергозбереження трансформуються у взаємопов'язані системи, такі як розумні системи освітлення, які можна повністю автоматизувати і контролювати через додаток за допомогою смартфона.

4. Звичайні побутові предмети, такі як пральна машина або холодильник, підключаються до Інтернету, щоб повідомляти про стан ваших речей або залишок продуктів у холодильнику.

По мірі того, як підключені пристрої стають стандартом, все більше типів пристроїв підключаються до домашніх мереж і стають доступними з Інтернету для

зручності. Згідно зі звітом компанії IoT Analytics, приблизно 16,7 мільярда пристроїв Інтернету речей використовуватимуться у 2023 році, і щорічний приріст становить 16-18% [8]. Поряд з домашніми мережами у вигляді Wi-Fi використовуються кілька спеціалізованих бездротових протоколів для мереж домашньої автоматизації. Ці протоколи, такі як Z-Wave і Zigbee, дозволяють забезпечити більш ефективний зв'язок між засобами автоматизації, які часто також можна вважати пристроями IoT. Ці мережі можуть бути підключені до Інтернету через міст, що робить їх доступними для інших пристроїв, таких як комп'ютери та смартфони, які не підтримують ці спеціалізовані протоколи, але мають Wi-Fi.

З промислового боку ринку тенденція IoT також починає чітко простежуватися. Замість дорогого періодичного технічного обслуговування, де для вимірювань потрібне спеціалізоване обладнання, промислові машини можуть бути оснащені дешевими інтегрованими датчиками, які можуть транслювати свої дані безпосередньо через мережу.

Коли датчик показує, що щось не так, технічне обслуговування можна проводити за потребою, а не періодично. Автоматизація може йти навіть далі, ніж просто дані з датчиків. Наприклад, машини можуть автоматично і динамічно змащуватися мастилом з резервуару на основі даних датчиків, які можуть вимірювати, наскільки добре працює машина і чи потрібне чи потрібне додаткове змащення для запобігання зносу. Це може заощадити багато дорогої робочої сили, оскільки для перевірки та обслуговування цих машин потрібен спеціалізований персонал. Пристрої Інтернету речей можуть генерувати багато даних в режимі реального часу, що дозволяє застосовувати нові методи аналізу за допомогою новітніх технологій, таких як машинне навчання. Датчики вимірюють весь термін служби машини, включаючи точки відмови. Маючи достатньо даних, можна передбачити характеристики поломки, розпізнавши індикатори, що призводять до таких відмов.

Якщо поломку дорогої промислової машини можна запобігти, проаналізувавши попередні дані датчиків до такої події, компанії можуть заощадити багато грошей і підвищити продуктивність.

Впровадження Інтернету речей також прискорюється завдяки наявності широко розгорнутих мереж зв'язку. Там, де традиційні глобальні мережі пропонують високошвидкісне з'єднання через оптоволокно, мідь або 4G, така технологія вимагає багато енергії для роботи. Оскільки пристрої Інтернету речей потребують лише рішення з низькою пропускнуою здатністю, були створені глобальні мережі з низьким енергоспоживанням як тип мережі, що відповідає цим вимогам. Існує ряд різних стандартів, серед яких LoRa є найбільш поширеним.

Якщо поломці дорогого промислового обладнання можна запобігти, проаналізувавши попередні дані датчиків до такої події, компанії можуть заощадити багато грошей і підвищити продуктивність.

Впровадження Інтернету речей також прискорюється завдяки наявності широко розгорнутих мереж зв'язку. Там, де традиційні глобальні мережі пропонують високошвидкісне з'єднання через оптоволокно, мідь або 4G, така технологія вимагає багато енергії для роботи. Оскільки пристрої Інтернету речей потребують лише рішення з низькою пропускнуою здатністю, були створені глобальні мережі з низьким енергоспоживанням як тип мережі, що відповідає цим вимогам. Існує ряд різних стандартів, з яких найбільш відомим є LoRa. Ця технологія дозволяє передавати дані на дуже великі відстані (понад 10 км) з низьким енергоспоживанням. Альянс LoRa - це мережа телекомунікаційних провайдерів по всьому світу, які експлуатують мережі LoRa. Для того, щоб надати цим пристроям необхідний зв'язок, доступні спеціальні мережеві модулі LoRa або вбудовані в IoT-пристрої. Таким чином, моніторинг в режимі реального часу став можливим за низьку ціну. Передавач LoRa може працювати протягом 15 років на двох батареях, що відкриває багато можливостей для широкого розгортання без високих витрат на обслуговування цих датчиків.

Незважаючи на те, що існує незліченна кількість переваг підключення об'єктів і пристроїв до Інтернету, швидке зростання Інтернету речей і пов'язані з ним проблеми безпеки створюють широке поле для атак кіберзлочинців.

Однією з основних характеристик пристроїв Інтернету речей є обмежені обчислювальні можливості, наприклад, зменшена обчислювальна потужність і обсяг

пам'яті в порівнянні зі звичайними комп'ютерами. Через ці обмеження залишається мало місця для реалізації складних механізмів безпеки механізмів, які б адекватно захищали пристрій. Крім того, бізнес Інтернету речей здебільшого орієнтований на отримання прибутку, що робить низьку собівартість і короткий час виходу на ринок важливими факторами для виробників IoT. Отже, питанням безпеки приділяється недостатньо уваги, і сьогодні на ринку присутня величезна кількість вразливих IoT-пристроїв.

Крім того, пристрої Інтернету речей піддаються більшому ризику атак у порівнянні з іншими інформаційними системами з кількох причин. Одна з них полягає в тому, що розумні пристрої завжди увімкнені та підключені. Інша полягає в тому, що більшість пристроїв Інтернету речей, які продаються з дуже спрощеною інструкцією для користування працюють за принципом "підключи і працюй", не вимагаючи від кінцевого користувача особливих зусиль і технологічних знань.

Ця зручна для користувача концепція часто тягне за собою небезпечні налаштування за замовчуванням, включаючи стандартні і слабкі облікові дані для входу. Через загальну некомпетентність більшість людей ніколи не змінюють облікові дані доступу на своїх пристроях, якщо тільки їх не змусять до цього, або, що ще гірше, виробник пристрою повністю виключив таку можливість.

Слід зазначити, що облікові дані для входу за замовчуванням на подібних пристроях часто встановлюються виробником ідентичними, або написаними в посібнику користувача або надруковані десь на упаковці пристрою, що робить їх легко доступними для будь-кого. Прикладами можуть бути такі комбінації імен користувачів і паролів, як `admin/admin`, `user/user` та `root/root`.

Крім того, виробники публікують оновлення та патчі безпеки, але вони зазвичай не застосовуються до пристроїв автоматично. Як наслідок, багато пристроїв працюють з вразливою та застарілою прошивкою, оскільки користувачам бракує знань про те, як адміністрування своїх пристроїв.

У міру розвитку на ринок будуть виводитися нові продукти. Для вендорів економічно не вигідно підтримувати всю лінійку своїх продуктів доти, доки останній пристрій не буде виведений з експлуатації.

Підтримка програмного забезпечення коштує дорого, а отримання оновлень для пристроїв саме по собі є проблемою. Натомість, якщо пристрої взагалі підтримуються після закінчення гарантійного терміну, будь-яке оновлення програмного забезпечення часто припиняється невдовзі після виходу нової версії продукту. Бізнес-модель полягає не в тривалому терміні підтримки після покупки, а в самій покупці та швидкій заміні на продукт, який має більше можливостей або є дешевшим у виробництві. Однак продукти використовуються набагато довше, ніж період, на який надається підтримка. Якщо промисловий датчик може працювати самостійно протягом 15 років, навряд чи його замінять після закінчення гарантійного терміну. Те саме стосується і споживчих товарів. Навіть якби заміна була можлива, можна придумати безліч сценаріїв, коли пристрої Інтернету речей глибоко інтегровані в конструкції або інші місця, де пристрій просто недосяжний. Датчики, глибоко вбудовані в промислову установку, ймовірно, потребують повної зупинки заводу щоб отримати доступ до датчика. Просто залишити їх у спокої набагато життєздатніше, ніж витратити мільйони на заміну або оновлення. Це призводить до того, що продукти працюють на застарілому програмному забезпеченні, але все ще підключені до Інтернету. Навіть якщо пристрій зовсім новий, програмне забезпечення, яке ним керує, може бути не таким. Продавці часто покладаються на самих виробників обладнання

В дослідженні Ф. Ебберса [9] проведено аналіз даних 1,06 млн пристроїв, зібраних за допомогою пошукової системи Censys, а потім зіставлено результати з останньою версією програмного забезпечення, яку пропонує кожен виробник IoT пристроя. Аналіз показав, що лише 2,45 відсотка IoT-пристроїв мають найновішу доступну прошивку. Встановлені версії в середньому мають вік 19,2 місяця. Ці дані свідчать про те, що процеси і методи оновлення, які використовуються інженерами до цього часу, не є достатніми для того, щоб підтримувати пристрої Інтернету речей в актуальному стані.

Пристрої Інтернету речей часто безпосередньо підключаються до Інтернету для забезпечення зв'язку. Завдяки такому прямому підключенню ці пристрої мо-

жуть бути вразливими до дій злочинців, коли останні знаходять вразливості в їхньому програмному забезпеченні. Нові вразливості з'являються регулярно і можуть надати кіберзлочинцям необмежений доступ до пристрою. І не лише пристрої можуть містити вразливі місця. Оскільки нові протоколи поспішно впроваджуються у виробниче середовище, часто вони не оцінюються належним чином і можуть також містити більш фундаментальні недоліки.

Ці вразливості спричинені різними недоліками, такими як застаріле програмне забезпечення з відомими вразливостями, помилки в реалізації програмного забезпечення або використання облікових даних за замовчуванням. Такі вразливості дозволяють легко скомпрометувати пристрій. У порівнянні з компрометацією сервера компанії, що містить конфіденційну інформацію, вплив можна вважати обмеженим, оскільки типові пристрої IoT мають обмежені можливості. Однак, зважаючи на великі масштаби розгортання цих пристроїв та їх підключення до Інтернету, кількість компрометації може стати значною.

Оскільки пристрої Інтернету речей широко розгорнуті в масовій кількості, вони можуть бути легко використані, як тільки вразливість була опублікована. Зважаючи на те, що багато пристроїв ніколи не оновлюються, вони залишаються вразливими протягом усього свого життя. Щодо вразливостей, знайдених у настільному або серверному програмному забезпеченні, зловмисникам зазвичай доводиться діяти швидко. Серйозні загрози часто виправляються відносно швидко, що робить експлойти неефективними. Це призвело до зростання кількості шкідливих програм для Інтернету речей, які націлені саме на малопотужні пристрої, підключені до Інтернету.

Після успішного зламу пристрою зловмисник отримує повний контроль над ним і може робити на ньому все, що завгодно. У випадках, коли ці пристрої є частиною корпоративної мережі без належної сегментації, це може навіть означати, що зловмисники отримують постійний плацдарм всередині мережі, з якого вони можуть продовжувати свою діяльність далі. Комбінована атака може бути дуже ефективною, якщо сформувати так звану ботнет зі скомпрометованих малопотужних пристроїв.

Виробники ж і надалі не публікують оновлення для всіх своїх лінійок продуктів з різних причин: пристрої більше не підтримуються, патчі не доступні від постачальників або тому, що певна функціональність, необхідна для роботи пристрою, залежить від вразливого фрагмента коду.

Крім того, вартість дешевих і старих пристроїв може призвести до того, що постачальник вирішить, що інвестувати в оновлення більше не є економічно доцільним, або ж йому просто буде байдуже після того, як продукт буде продано.

Навіть якщо патч доступний, отримати оновлення для цих пристроїв може бути неможливо, наприклад, якщо він інтегрований у промислове обладнання. Оскільки оновлення програмного забезпечення не поширюються на ці пристрої, то і промислове обладнання залишається вразливими протягом усього терміну експлуатації. Це робить атаки на такі пристрої дуже ефективними і небезпечними.

Нарешті, часто за замовчуванням на IoT-пристроях відкриті порти для мережевих протоколів, таких як Telnet, SSH та HTTP. Що дає змогу зловмисникам використовувати різноманітні способи підключення до пристроїв з використанням цих портів. Це спричиняє порушення конфіденційності, цілісності та доступності даних, якщо зловмисники отримають віддалений доступ до пристрою.

1.3 Огляд основних загроз для IoT

Пристрої Інтернету речей є привабливими цілями для зловмисників через існуючі проблеми з безпекою. Атаки різняться за складністю, а також поширенням та потенціалом шкоди, залежно від загальної мети атаки. Деякі атаки здійснюються з метою лише розкриття інформації, в той час як інші спрямовані на повну компрометацію системи з використанням віддаленого або довільного виконання коду.

Найсерйознішою загрозою, з якою стикаються пристрої Інтернету речей, є шкідливе програмне забезпечення.

Існує безліч різних зразків шкідливого програмного забезпечення та сімейств шкідливих програм, і їх кількість збільшується з появою на ринку різних пристроїв IoT, які постійно випускаються на ринку. Різні шкідливі програми класифікуються на основі таких факторів, як те, що вони вони роблять та їхнє призначення. Деякі з найвідоміших типів - руткіти, програми-вимагачі, боти, фінансові шкідливі програми, логічні бомби, віруси, хробаки та троянські програми.

Руткіт - це тип шкідливого програмного забезпечення, який надає зловмиснику привілейований доступ до системи, наприклад наприклад, root-доступ, до системи. Це практично дає зловмиснику повний контроль над пристроєм, що робить його вразливим для подальших маніпуляцій.

Шкідливе програмне забезпечення, що блокує дані користувача з вимогою викупу має загальну мету - вимагати від користувача гроші. Це ПЗ зазвичай блокує пристрій або програмне забезпечення користувача, наприклад, за допомогою блокування екрану або шифрування даних. Потім, щоб видалити інфекцію і відновити нормальну поведінку, користувач повинен заплатити зловмиснику викуп.

Боти - це саморозповсюджуване шкідливе програмне забезпечення, яке заражає пристрій перед підключенням до центрального сервера, який зазвичай називають бот-майстром, для отримання подальших інструкцій. Заражені пристрої можуть бути використані для кількох цілей, наприклад, для зараження інших пристроїв, запуску DDoS-атаки або збору конфіденційних даних.

DDoS-атаки або для збору конфіденційної інформації та відправки її назад бот-майстру.

Фінансове шкідливе програмне забезпечення визначається як тип, основною метою якого є збір та надсилання інформації про банківські рахунки зловмиснику. Такою інформацією є часто отримують або шляхом збору даних безпосередньо з пристрою, або за допомогою підроблених мобільних банківських додатків.

Логічні бомби - це фрагменти коду, розміщені зловмисником всередині програмної системи, які спрацьовують при виконанні певних умов. При спрацьовуванні ініціюються зловмисні дії, які можуть завдати шкоди системі, наприклад, видаливши або змінивши дані або виконання шкідливого коду.

Віруси - це шкідливе програмне забезпечення, якому для розмноження потрібна програма і поширюється разом з програмою, в яку він вбудувався. Для того, щоб вірус необхідна для того, щоб вірус запустився, наприклад, шляхом запуску програми в якій він знаходиться.

На відміну від вірусів, шкідливі програми-черв'яки можуть діяти самостійно і не потребують втручання користувача для самовідтворення та розповсюдження.

Троянські коні (трояни) - це тип шкідливого програмного забезпечення, яке виглядає як легальне програмне забезпечення, але насправді вони мають зловмисні цілі і можуть отримати контроль над зараженим пристроєм. На відміну від вірусів і хробаків, трояни не можуть самовідтворюватися, але, як і віруси, вимагають втручання користувача. Подібно до вірусів, вони потребують взаємодії з користувачем для виконання своїх дій. Існує кілька типів троянських програм, залежно від дій, які вони виконують. Деякі з них найпоширеніших типів - Trojan Backdoor, Trojan DDoS, and Trojan Downloader. Trojan Backdoor створює "чорний хід" на пристрої, що полегшує подальші атаки, дозволяючи зловмиснику отримати як доступ, так і дистанційне керування. Типові дії, що виконуються на зараженому пристрої що виконуються на зараженому пристрої, - це надсилання та отримання файлів, а також запуск і видалення файлів. Trojan DDoS, як випливає з назви, виконує DDoS-атаки з інфікованих пристроїв на певну IP-адресу. Нарешті, Trojan Downloader непомітно

завантажує і встановлює шкідливі файли з віддаленого сервера, а потім запускає файли на зараженому пристрої.

Важливим вектором атаки на пристрої IoT є увімкнений сервіс Telnet. Telnet - це дуже старий протокол зв'язку між пристроями, який дозволяє адміністраторам віддалено входити на пристрій і виконувати команди. Протокол Telnet надає засоби для встановлення зв'язку з віддаленим сервером. Те, як цей зв'язок обробляється і які можливості відкриваються після підключення, залежить від реалізації сервера Telnet. Звичайні сервери Telnet надають стандартний термінал, дуже схожий на той, що можна отримати при підключенні монітора до пристрою без графічного інтерфейсу користувача. Інші користувацькі реалізації мають обмежений набір команд, які можна виконати, залежно від пристрою.

Telnet - це протокол прикладного рівня, який використовується для зв'язку з віддаленим хостом за допомогою інтерфейсу командного рядка. Протокол було розроблено у 1969 році, ще до того, як інтернет став загальнодоступним. Завдяки своєму ранньому створенню, він не застосовував жодної форми шифрування для зв'язку, що робить його застарілим з точки зору сучасної безпеки і не так широко використовується, як раніше. Таким чином, більш безпечніші протоколи, такі як SSH, все частіше замінюють Telnet. Тим не менш, існує кілька пристроїв Інтернету речей, таких як маршрутизатори, відеореєстратори та IP-камери, які реалізують Telnet у вбудованих системних додатках завдяки його відносно простій реалізації. Сканування за допомогою Shodan, виявляє, що понад п'ять мільйонів підключених гаджетів по всьому світу мають відкритий порт Telnet.

Оскільки належний рівень безпеки є важливим для протоколу віддаленої взаємодії, Telnet був витіснений більш сучасними альтернативами, такими як SSH. Однак, незважаючи на наявність протоколів, які набагато краще підходять для цієї мети, Telnet все ще має значну присутність в Інтернеті. Хоча існують деякі сервери Telnet, які навмисно залишаються в мережі, більшість легітимного використання Telnet належить до старих систем, які так і не отримали підтримки оновлення до сучасного протоколу зв'язку. Telnet також популярний на таких пристроях, як мо-

деми та маршрутизатори, як спосіб конфігурації цих пристроїв під час їх налаштування для виробничого використання. Перевагою Telnet є те, що він дуже легкий, оскільки не потребує криптографічних обчислень, які можуть бути дорогими для вбудованих мікросхем.

За замовчуванням сервер Telnet працює на 23 порту протоколу управління передачею (TCP), але його можна налаштувати щоб він був доступний і через порт 2323. Пристроєм, на яких один з цих двох портів відкритий, зловмисники можуть завдати значної шкоди. Оскільки при використанні Telnet з'єднання не шифрується, конфіденційна інформація, така як паролі та ідентифікатори, легко доступна зловмисникам шляхом підслуховування. Додаткова інформація про пристрій, наприклад, модель обладнання та модель програмного забезпечення, також може бути розкрита і використана зловмисниками. Також зловмисники можуть визначити, чи вимагає пристрій автентифікації. Якщо так, зловмисники можуть отримати несанкціонований доступ, підслухавши облікові дані, надіслані відкритим текстом, або або спробувавши відомі облікові дані за замовчуванням. Паролі для стандартних облікових записів, таких як root або admin, також можуть бути отримані за допомогою простих атак грубого перебору.

Існує кілька реалізацій Telnet, але жодна з них не стала домінуючою. З цієї причини не всі клієнти, які використовують Telnet, повністю реалізують стандарт, тобто не всі опції підтримуються однаково. Це дає можливість ідентифікувати певні пристрої, надсилаючи клієнтам запити на переговори з рідкісними опціями, щоб з'ясувати, які з них підтримуються.

Було представлено зашифровану альтернативу у вигляді TelnetS, але вона так і не стала такою популярною, як Telnet. Натомість SSH є найпопулярнішим протоколом, який має схожі функції з Telnet, але з набагато кращими функціями безпеки та шифрування.

Secure Shell (SSH) - це мережевий протокол прикладного рівня, який зазвичай використовується для отримання доступу до командного рядка (оболонки) на віддаленому хості. Він був в основному розроблений для заміни декількох застарілих

протоколів, серед яких протокол Telnet. SSH - це криптографічний протокол з архітектурою клієнт-сервер, який дозволяє безпечно працювати з мережевими безпечно працювати з мережевими сервісами через незахищену мережу. На відміну від протоколу Telnet, який надсилає всю інформацію у відкритому вигляді, SSH шифрує всі дані, що передаються між клієнтом і сервером. За замовчуванням TCP-порт для SSH - 22, але користувач може змінити його користувач може змінити його на інший порт.

Крім того, протокол надає спеціальні ключі SSH для більш безпечного і автоматизованого процесу автентифікації. Функціонально SSH-ключі - це облікові дані для автентифікації які замінюють імена користувачів і паролі, запобігаючи успішній атаці грубої сили.

У IoT ключі SSH можуть бути особливо корисними, оскільки слабкі паролі є однією з найбільших проблем безпеки. За допомогою цих ключів кожен пристрій отримує відкритий ключ, що відповідає приватному ключу виробника, що дозволяє постачальникам оновлювати та керувати пристроями віддалено. Таким чином, оскільки це асиметрична схема шифрування, кіберзлочинці не можуть використувати відкритий ключ для отримання доступу, якщо у них немає відповідного закритого ключа.

1.4 Основні вразливості і напрямки атак для IoT

Традиційно атака на пристрій часто складається з різних етапів. Класичний метод поділу цих етапів був запропонований Е. Хатчинсом та ін. [10] має назву Cyber Killchain (СК). Killchain - це метод відображення різних етапів атаки в єдину модель. Пов'язуючи ці окремі етапи разом, можна проаналізувати атаку в цілому або навіть як частину кампанії. Якщо два етапи часто використовуються разом, модель СК може допомогти виявити початкову розвідку, яку проводив зловмисник після того, як в системі було виявлено певний тип шкідливого програмного забезпечення, яке було виявлено в системі. Вона також робить можливим зворотний

процес, дозволяючи захисникам вжити конкретних заходів, як тільки атака буде виявлена на ранніх стадіях.

Етапами Cyber Killchain'a є:

1. Розвідка.
2. Озброєння.
3. Доставка.
4. Експлуатація.
5. Встановлення.
6. Командування та управління.
7. Дії на цілі.

На першому етапі, розвідці, метою є отримання якомога більше інформації про ціль. Оскільки модель СК охоплює всі атаки, пов'язані з кібербезпекою, це може бути зроблено будь-якими способами, як онлайн, так і офлайн. Залежно від цілі, можна придумати різні методи для проведення розвідки. Якщо метою є сервер, підключений до комп'ютера, зловмисники можуть використовувати сканери вразливостей, щоб знайти слабкі місця в підключених до нього сервісах.

Однак метою також може бути системний адміністратор компанії, і в цьому випадку можна подумати про спостереження та використання соціальних мереж як методи розвідки. Якщо розглядати атаки на пристрої Інтернету речей, ця фаза може, серед іншого, включати сканування Інтернету на наявність пристроїв з відкритими портами або перехоплення банерів з відкритих серверів Telnet.

Коли зловмисник зібрав достатньо розвідувальних даних для здійснення атаки, трояну для віддаленого доступу поєднується з експлойтом для створення корисного навантаження, що підлягає доставці. Експлойт - це частина, яка використовує вразливість у програмі або іншому засобі безпеки, що дозволяє виконати код на цільовій системі. Код, що виконується, встановлює троянську програму для віддаленого доступу - програму, яка дозволяє повністю контролювати систему, на якій вона виконується. Існує декілька стратегій для встановлення як експлойту, так і

трояну для віддаленого доступу, кожна з яких має різні характеристики та відповідні способи використання.

Вибір правильної комбінації та їх поєднання - це те, що відбувається на етапі взяття на озброєння. Якщо зіставити цю фазу з IoT-пристроєм, експлойт може бути таким же простим, як отримання набору облікових даних за замовчуванням, які, як відомо, працюють на пристрої, оскільки вони жорстко закодовані або не змінюються з моменту встановлення. Що стосується корисного навантаження, то може бути використаний варіант Mirai, заснований на його вихідному коді, але також можуть бути обрані інші шкідливі програми, сумісні з пристроями IoT.

На етапі доставки зломисник прагне доставити озброєний пакунок з попереднього етапу до цілі. Цього можна досягти різними способами, наприклад, за допомогою фішингу, заманювання жертви на шкідливий веб-сайт, підкидання USB-накопичувача або прямого підключення до сервера.

IoT-пристрої часто підключаються безпосередньо до Інтернету, що дозволяє здійснити тривіальну фазу доставки. Просте підключення до пристрою дозволяє доставити експлойт у вигляді облікових даних за замовчуванням, після чого код може бути виконаний через наданий доступ до командної оболонки.

Після того, як пакет успішно доставлений до цілі, експлойт використовує вразливість для виконання коду зломисника в системі. Це можна інтерпретувати по-різному, і не завжди ґрунтується на технічній вразливості в програмі. Якщо користувача заманюють до запуску шкідливої програми, код виконується без зловживання технічною вразливістю, але зазвичай зловживають довірою користувача.

Прикладом з іншого варіанту дій зломисника є, наприклад, помилка у програмному забезпеченні веб-сервера, що дозволяє виконувати код на машині без участі користувача.

Запуск коду зломисника в системі призводить до інсталяції троянської програми віддаленого доступу, що входить до складу троянської частини пакунка з «озброєнням». Так само, як і експлойт, існує багато різних типів троянів, кожен з яких має свої власні переваги та особливостями. Залежно від мети зломисника, вибирається і встановлюється той чи інший троянець.

Мета етапу встановлення полягає в тому, щоб підтримувати постійну роботу в системі, замість того, щоб покладатися на експлойти для запуску коду в системі. Це означає, що зловмисник має незалежну точку опори в системі, яку не може бути вирішена просто шляхом виправлення початкових вразливостей в системі. Де атаки на пристрої IoT здійснюються через погано захищене з'єднання Telnet, шкідливе програмне забезпечення, таке як Mirai, не покладається на Telnet для керування системою. Замість цього завантажується і виконується двійковий файл, що дозволяє контролювати пристрій незалежно від Telnet-з'єднання.

Троян для віддаленого доступу використовує власні методи зв'язку зі зловмисником. Цей зв'язок називається трафіком команд і управління, також відомим як C&C або C2-трафік. Оскільки це зв'язок між додатком і мережею зсередини, він часто не блокується брандмауером, на відміну від вхідних з'єднань. Троян під'єднується до сервера за межами мережі, де він може знайти інструкції про те, як продовжити атаку. Подальші варіанти розвитку подій дуже різноманітні, але вони залежать від функцій, які має троян. Поширені функції можуть включати відкриття оболонки в системі, вилучення файлів, створення скріншотів або поширення по внутрішній мережі. Це може бути як повністю автоматизовано, так і зроблено вручну, щоб уникнути можливих заходів виявлення. Ця завершальна фаза атаки є останньою фазою моделі КС, яка називається "Дії по досягненню цілей".

Цілі широко варіюються в залежності від зловмисника. Для IoT ботнетів "орєнда" пристроїв для масштабних DDoS-атак є звичайною метою, але більш просунуті зловмисники можуть використовувати пристрої Інтернету речей, щоб спочатку закріпитися в мережі компанії, а потім діяти далі. У попередніх дослідженнях, присвячених вивченню компрометації IoT [11], спостерігалися DDoS-атаки у вигляді UDP-флудів і багатьох різних типів TCP-атак. Крім того, спостерігалися атаки типу DNS water torture2 та SSL-атаки.

Для точного аналізу методів і мотивації зловмисника важливо, щоб кожен етап КС можна було дослідити незалежно. Створюючи "відбитки пальців" для кожного етапу, можна отримати шаблони для ідентифікації зловмисників або груп зловмисників. Що ще важливіше, пов'язуючи різні етапи разом в рамках однієї атаки або

кампанії, можна виявити атаки на значно більш ранніх стадіях КС, зменшуючи кількість шкоди, яку може завдати зловмисник.

Протягом багатьох років численні пристрої IoT, що працюють зі службою Telnet або SSH стали жертвами численних сімейств шкідливих програм, таких як Mirai, Hajime і Gafgyt, і це лише деякі з них. Спільним для багатьох з цих сімейств шкідливих програм є те, що вони використовують пристрої Інтернету речей для створення масових шкідливих мереж, також відомих як ботнети.

IoT-ботнети часто використовуються для атак на інші системи, наприклад шляхом запуску DDoS-атаки. Крім того, скомпрометовані пристрої можуть бути використані для інших негідних цілей, наприклад, для зараження інших пристроїв.

Як правило, ці IoT-атаки проходять три фази:

- фаза розвідки і вторгнення,
- фаза зараження,
- фаза монетизації .

Під час початкового етапу атаки, в фазі розвідки та вторгнення, зловмисники виконують автоматичне сканування діапазонів публічних IP-адрес, щоб знайти пристроїв, які приймають з'єднання через певний порт, наприклад, порт 22, порт 23 або порт 2323, перш ніж спробувати подолати захист самого пристрою. Одним з найпоширеніших методів вторгнення – перебір (брутфорс). При проведенні брутфорсу зловмисник, як правило, намагається підібрати набір часто використовуваних облікових даних для стандартних системних користувачів або заводські облікові дані за замовчуванням для конкретних пристроїв IoT.

Як BASHLITE (також відомий як Gafgyt, LizardStresser або Torlus) так Mirai, серед інших, використовують цей метод вторгнення за допомогою жорстко закодованого словник з обліковими даними за замовчуванням. Набір облікових даних, який використовує BASHLITE включає шість типових імен користувачів і 14 типових паролів, тоді як словник, що використовується

Mirai є більшим і містить 62 унікальні пари імен та паролів. На Рисунку 1.4.1 перелічено 46 унікальних паролів, включених до оригінального вихідного коду

Mirai, а також деякі деяких пристроїв IoT, що використовують ці паролі за замовчуванням. Очевидно, що саме атаки на IoT-пристрої є дуже цілеспрямованими, оскільки більшість паролів можуть бути підключені до декількох різних типів пристроїв, де IP-камери, відеореєстратори та маршрутизатори є одними з найпопулярніших.

Password	Device Type	Password	Device Type	Password	Device Type
123456	ACTI IP Camera	klv1234	HiSilicon IP Camera	1111	Xerox Printer
anko	ANKO Products DVR	jvbsd	HiSilicon IP Camera	Zte521	ZTE Router
pass	Axis IP Camera	admin	IPX-DDK Network Camera	1234	Several IP Cameras
888888	Dahua DVR	system	IQinVision Cameras	12345	Several IP Cameras
666666	Dahua DVR	meinsm	Mobotix Network Camera	root	Samsung IP Camera
vizxv	Dahua IP Camera	54321	Packet8 VOIP Phone	password	Routers
7ujMko0vizrv	Dahua IP Camera	00000000	Panasonic Printer	fucker	Unknown
7ujMko0admin	Dahua IP Camera	realtek	RealTek Routers	guest	Unknown
666666	Dahua IP Camera	1111111	Samsung IP Camera	admin1234	Unknown
dreambox	Dreambox TV Receiver	xmhdipc	Shenzhen Anran Camera	default	Unknown
juantech	Guangzhou Juan Optical	smcadmin	SMC Routers	service	Unknown
xc3511	H.264 Chinese DVR	ikwb	Toshiba Network Camera	support	Unknown
OrhlwSC8	HiSilicon IP Camera	ubnt	Ubiquiti AirOS Router	tech	Unknown
eat1029	HiSilicon IP Camera	supervisor	VideoIQ	user	Unknown
hi3518	HiSilicon IP Camera	blank	Vivotek IP Camera	xlxx.	Unknown
klv123	HiSilicon IP Camera				

Рисунок 1.4.1 – Типові паролі за замовчуванням для IoT пристроїв

Після того, як зловмисник отримав доступ до командної оболонки, наступним кроком зазвичай є фаза зараження. Під час цієї фази здійснюється спроба отримати повний контроль над пристроєм і налаштувати його для будь-якої, яку він матиме на фінальній стадії монетизації. Етап зараження часто включає в себе завантаження бінарного файлу, а отже, саме на цьому етапі на пристрої з'являється шкідливе програмне забезпечення.

Перш ніж завантажити та встановити бінарні файли шкідливого програмного забезпечення, зловмисник готує середовище, до якого здійснюється доступ, перевіряючи та налаштовуючи його. Зазвичай ця процедура здійснюється шляхом надсилання фіксованої серії команд, що залежить від конкретної атаки, через експлуатований сервіс.

Одна з найвідоміших послідовностей команд, що виконуються шкідливими програмами, націленими на служби Telnet і використовується такими шкідливими програмами, як Mirai і Hajime, складається з наступних п'яти рядків:

```
enable
system
shell
sh
/bin/busybox <random_string>
```

Метою виконання перших чотирьох команд є увімкнення доступу до командного інтерпретатора. Мета останньої команди останньої команди - перевірити, чи присутній BusyBox2, щоб визначити, чи належить чи належить система до IoT-пристроїв. Якщо відповідь буде

```
/bin/busybox: No such file or directory,
```

то зловмисник найчастіше розриває з'єднання.

Якщо в системі є BusyBox, відповідь буде

```
<random_string>: applet not found,
```

і, таким чином, вважається дійсним для подальшої експлуатації зловмисником.

Ці початкові команди не є типовими для SSH-інфекцій, однак наступні дії схожі. Зловмисник часто продовжує досліджувати параметри пристрою до якого він отримав доступ, визначаючи такі характеристики, як архітектура процесора, платформа і версію ядра, а також видаляє потенційно присутні файли, завантажені конкуруючим шкідливим програмним забезпеченням. Далі для завантаження шкідливого двійкового файлу зазвичай використовуються wget, tftp, curl або echo.

Потім, права доступу до двійкового файлу зазвичай підвищуються за допомогою chmod, щоб зробити його доступним для читання, запису і виконання, після чого завантажується виконуваний файл. Нарешті, перед розривом з'єднання багато зловмисників намагаються видалити докази своєї діяльності, видаляючи всі завантажені файли і очищаючи історію bash.

Однак часто дії зломисників з протоколом SSH не передбачають зараження шкідливим програмним забезпеченням після успішного входу в систему. Скомпрометований IoT-пристрій зазвичай використовується як проксі-сервер, що використовує можливості переадресації портів протоколу SSH. Зломисник надсилає TCP/IP-запит на переадресацію трафіку на вказану IP-адресу і порт призначення, використовуючи пристрій Інтернету речей як проміжну службу. Це може бути використано для використовувати для надсилання спаму або HTTP-трафіку до сервісу або веб-сайту-жертви.

На останньому етапі, під час фази монітезації, противник використовує скомпрометований пристрій або пристрої в подальших операціях. Однією з найпоширеніших групових атак з використанням великої кількості заражених пристроїв, є DDoS-атака. DDoS-атаки мають на меті перешкодити нормальній роботі та доступності, націлюючись на сервер, службу або мережу з великим навантаженням трафіку. Цей потік трафіку генерується за допомогою централізованого сервера управління і контролю (C&C), керованого зломисником, для управління кількома зараженими пристроями, що складають бот-мережу, з метою одночасно надсилати пакети з постійною швидкістю, щоб перевантажити жертву. Таке перевантаження трафіку може, в свою чергу, спричинити збій або відмову в обслуговуванні легітимному трафіку. DDoS-атаки добре відомі і здійснюються вже багато років, ще задовго до появи Інтернету речей. Однак величезна кількість незахищених пристроїв Інтернету речей підключених до інтернету, відкрила можливість для створення більш масових і потужніших бот-мереж, ніж будь-коли раніше.

Висновок до РОЗДІЛУ 1

В першому розділі було розглянуто основні тенденції розвитку кібербезпеки систем Інтернету речей. Проаналізовано причини як відставання систем безпеки IoT пристроїв так і шляхи посилення кібербезпеки IoT систем.

Дешевизна, масовість і популярність подібних систем призводить до того, що багато пристроїв ніколи не оновлюються, вони залишаються вразливими протягом усього свого життя. Це значно полегшує зловмисникам створення автоматизованих систем для пошуку і розповсюдження шкідливих програм для Інтернету речей, які націлені саме на малопотужні пристрої, підключені до Інтернету. У випадках, коли ці пристрої є частиною корпоративної мережі без належної сегментації, це може означати, що зловмисники отримують постійний плацдарм всередині мережі, з якого вони можуть продовжувати свою діяльність далі. Такі комбіновані атаки є дуже ефективними, якщо сформувати ботнет зі скомпрометованих IoT пристроїв.

Виробники часто взагалі не публікують оновлення для всіх своїх лінійок продуктів з різних причин: пристрої більше не підтримуються, патчі не доступні від постачальників або тому, що певна функціональність, необхідна для роботи пристрою, залежить від вразливого фрагмента коду. Крім того, вартість дешевих і старих пристроїв може призвести до того, що постачальник вирішить, що інвестувати в оновлення більше не є економічно доцільним, або ж йому просто буде байдуже після того, як продукт буде продано.

Одним з традиційних способів для зловмисників отримати доступ до IoT пристрою є використання того, що часто за замовчуванням на IoT-пристроях відкриті порти для мережевих протоколів, таких як Telnet, SSH та HTTP. А це, в свою чергу, дає змогу зловмисникам використовувати різноманітні способи підключення до пристроїв з використанням цих портів. Це спричиняє порушення конфіденційності, цілісності та доступності даних, якщо зловмисники отримають віддалений доступ до пристрою.

Всі ці проблеми характерні як для всіх IoT пристроїв загалом так і для систем «Розумний будинок» зокрема.

РОЗДІЛ 2. HONEYPOT ЯК ЕЛЕМЕНТ ПОСИЛЕННЯ КІБЕРБЕЗПЕКИ ІОТ СИСТЕМИ

2.1 Огляд досліджень використання honeypot для IoT

Інструментарій і шляхи проникнення кіберзлочинців в мережі є дуже широким та постійно еволюціонує. Саме тому сучасні дослідження deception технологій і honeypot різного рівня інтерактивності відрізняються різноманіттям підходів до побудови дослідницьких мереж зі штучними вразливостями безпеки для збору інформації в «польових умовах». Так в роботах Фраунхольца і Шоттена [12-13] описано низку досліджень honeypot в університетських мережах. У цих статтях автори провели аналіз поведінки як реальних зловмисників, так фахівців з перевірки мереж на вразливості у сконфігурованій вразливій системі. Дослідники надали їм атаковану систему, в якій використовували ресурси honeypot, і стежили за їхньою поведінкою. Це були серверні honeypot-програми: подрібний файл robots.txt, змінені повідомлення про помилки, адаптивна затримка та різноманітні honey-files. Було перевірено більше тисячі відвідувань зловмисників. В подальшому було проаналізовано поведінку зловмисників протягом шести місяців за допомогою низки honeypots, розгорнутих на одному клієнті та п'яти серверах веб-хостингу. Було відстежено більше десяти мільйонів відвідувань. У honeypot об'єктах використовувалися протоколи honeypotTokenTP, honey-tokenTPS, FTP, POP3, SMTP, SSH та Telnet. Також імітувалися промислові протоколи зв'язку для дослідження загроз промисловим додаткам.

Кіберзлочинці постійно шукають способи виявлення фактів використання deception платформ і honeypot в мережі яку вони атакують. Це викликає потребу постійного вдосконалення подібних систем і збільшення їх правдоподібності. Так в роботі Д. Реті і його дослідницької команди [14] вводиться поняття інтерференційних елементів-приманок як правдоподібне розширення існуючих структур обману, спрямовуючи зловмисників на атаку елементів-приманок. Пропонуються їх моделі і еталонні реалізацію. Аналізується поведінкові модулі зловмисників при взаємодії

з новим типом приманок. Перевагою запропонованих рішень є збільшення взаємодії між зловмисником і розгорнутими елементами-приманками, що збільшує ймовірність викликати у зловмисника невпевненість, втрачаючи при цьому час і ресурси зловмисника. Запропонована система здатна покращити процес виявлення вторгнень, а також затримувати та перешкоджати поточній розвідувальній діяльності.

В останні роки посилений інтерес дослідників викликають проблеми побудови високоінтерактивних honeypot-систем для інтернету речей, оскільки проблема безпеки в IoT і вдосконалення інструментарію злочинців не дозволяють використувати традиційні підходи мереж загального призначення. В роботі Фраунхольца і Шотена [15] розглядається Falcom — приманка високої взаємодії, яка надає повноцінну операційна система, що максимізує її взаємодію з зловмисниками і розроблена для вбудованих архітектур. Будь-яка взаємодія з цією приманкою є підозрілою та направляється на подальше розслідування. Аналізуючи спостережувані параметри атак, можна виявити останні тенденції, нові вектори атак і поточні спроби вторгнень. В роботі розглядаються особливості побудови приманок для вбудованих систем, архітектури процесорів, а також системні ресурси, що обираються для правдоподібної імітації вбудованих пристроїв. У еталонній реалізації досліджується механізм автентифікації схильний до атак грубої сили та словникових атак.

Одним з головних завдань honeypot є збір доказової бази для розслідування кіберінцидентів і подальшого притягнення кіберзлочинців до відповідальності. В роботах [15-16] розглядається система приманок середньої взаємодії, яка пропонує послуги telnet і SSH. За допомогою цього honeypot протягом трьох місяців виконувався збір інформації про взаємодії з ними. Ці дані використовувались для статистичного та поведінкового аналізу. В роботі був проаналізований розподіл атак і різні IP-адреси зловмисників, країни походження, використовувані служби анонімізації, рівень навичок супротивника та вбудовані пристрої, на які зазвичай націлено. В роботі використані методи машинного навчання, які здатні ідентифікувати унікальні типи сеансів на основі виданих команд і наданих облікових даних. Зібрані дані були проаналізовані на ознаки, які дозволили класифікувати типи зловмисників і сеанси.

Відмінності в архітектурі і ресурсах промислових мереж інтернету речей різного призначення призводять до необхідності пошуку правильних методик розгортання honeypot для максимальної ефективності їх використання. В роботі А. Асієн [17] розглядаються різноманітні існуючі honeypot системи для промислових IoT-систем і методи їх розгортання. При цьому проаналізовано пошукові системи, що використовуються зловмисниками для пошуку і ідентифікації PLC і SCADA-систем та вразливостей в них на етапі підготовки атаки. Досліджені методики розгортання систем приманок з використанням хмарних технологій. В якості системи збору інформації про взаємодію з приманками використаний популярний стек ELK (ElasticSearch, Logstash і Kibana). В роботі продемонстрована методика розгортання приманок шляхом проведення перевірки концепції на основі атак в контрольованому середовищі.

Не зважаючи на свій вік протокол telnet досі широко застосовується в мережах IoT і часто стає ціллю атаки зловмисників. В дослідженні Х. Семіча [18] досліджується реалізація honeypot, що виявляє та повідомляє про атаки на telnet на пристрої Інтернету речей. Розглянутий honeypot дозволяє виявляти як атаки зловмисника так і атаки на основі ботнета Mirai. Багатокомпонентна конструкція реалізована для досягнення достатнього впливу шкідливого трафіку та безпеки зібраних даних. В роботі досліджується гнучкий дизайн приманки, який дозволяє легко модифікувати honeypot для емуляції різних пристроїв IoT.

Одним потенційно небезпечних напрямків атаки є атаки зловмисників на пристрої IoT, які використовують протокол Universal Plug and Play. В роботі М. Хакіма [19] запропоновано і досліджено фреймворк U-PoT для побудови honeypot для пристроїв інтернету речей. Запропонований фреймворк автоматично створює приманку з документів опису пристроїв UPnP і може бути розширена до будь-яких типів пристроїв або постачальників, які використовують UPnP для зв'язку. Проведені експериментальні дослідження показали, що емульовані пристрої здатні імітувати поведінку справжнього пристрою IoT і обманювати надані постачальниками програми керування пристроєм або популярні пошукові системи IoT, що використовуються щзлочинцями для пошуку вразливих пристроїв.

2.2 Використання honeypot як інструменту виявлення атак

Історично honeypot-системи були розроблені для пошуку та вивчення дій зловмисників у скомпрометованій системі. Термін honeypot використовується для системи, яка була налаштована з наміром бути скомпрометованою. Зазвичай вона містить більш старе і вразливе програмне забезпечення з уразливостями або дірками в безпеці, пов'язаними з неправильним налаштуванням програми. За рахунок свого розташування в межах DMZ та в середині мережі підприємства вона має слугувати пріоритетною цілью і дає змогу отримати інформацію про методи та інструменти зловмисника. Honeypot-система дає змогу знизити кількість помилкових спрацьовувань, що видаються IDS/IPS-системами. Honeypot можуть бути легко використані для виявлення та систематизації інформації про нові методи атак і вдосконалення системи інформації про перспективні загрози [20]. На ранніх етапах зловмисник сканує мережу для пошуку вразливих комп'ютерів, внаслідок чого виявляє приманку, яка навмисне вразлива для залучення атак. Якщо надалі зловмисник спробує підключитися до honeypot, система негайно виявить та зафіксує дію, тому що звичайний користувач не повинен взаємодіяти із системою [21].

Залежно від технології, що використовується, системи honeypot можна класифікувати наступним чином:

- Malware Honeypot: реплікують і використовують вектори атак для ідентифікації шкідливого програмного забезпечення.
- Spam Honeypot: налаштовані на імітацію відкритих проксі-серверів і поштових ретрансляторів. Виявляє та блокує велику кількість спаму.
- Database Honeypots: ефективні проти таких дій, як SQL-ін'єкції, які часто не відстежуються брандмауерами.
- Client Honeypots: допомагають відстежувати шкідливі сервери, які в першу чергу відповідають за атаки на клієнтів. Функціонують на основі технології віртуалізації та реалізує стратегію стримування для обмеження ризиків.

- Honeynets: Це єдина система, що складається з різних honeypot'ів мережевої безпеки. Метою honeynets є тактичне відстеження мотивів і методів зловмисника, одночасно стримуючи всі види вихідного і вхідного трафіку.

Однак, основною класифікаційною ознакою honeypot систем є ступінь їхньої інтерактивності. Під інтерактивністю розуміється рівень відкритих мережевих сервісів, доступних зловмиснику. Honeypot-системи бувають низькоінтерактивними та високоінтерактивними.

Низькоінтерактивний honeypot, як правило, включає один або кілька мережевих сервісів, що є об'єктами атаки. Такими сервісами є POP3, SMTP, IMAP, FTP, HTTP та інші. Такий honeypot встановлюється на комп'ютер під керуванням MS Windows або GNU/Linux як звичайний сервіс. Цей сервіс відразу закріплює за собою порти для прослуховування активності мережі. Число відкритих портів визначається кількістю емульованих сервісів. У більшості випадків, емуляція сервісів відбувається на поверхневому рівні - програми не реалізують всі вимоги RFC, а лише імітують команди, що найбільш часто викликаються [22].

10-15 років тому це вважалося достатнім, але зараз це стає однією з основних причин можливого викриття. Зловмисники намагатимуться виявити присутність honeypotів в мережі за допомогою підставних сервісів, що працюють в системі. Ці зонди можуть включати створення шкідливих пакетів, таких як HTTPS, SMTPS, на які honeypot дасть неправильну відповідь або взагалі не відреагує. Наприклад, порти, які повинні працювати, але відмовляються від тристороннього рукошлякування, можуть вказувати на наявність honeypot.

Зловмисники можуть виявити екземпляри, що працюють на віртуальній машині VMWare, проаналізувавши MAC-адреси. Посилаючись на стандарти IEEE для поточного діапазону MAC-адрес, призначених VMWare Inc., зловмисник може виявити наявність honeypots на базі VMWare.

Honeyd - це генератор honeypotів, за допомогою якого можна легко створити тисячі honeypotів. На отримані SMTP-запити вони відповідатимуть фальшивими відповідями. Зловмисник може виявити наявність honeypot за допомогою методів TCP-фінгерпринтингу, що базуються на часових параметрах.

Зловмисники можуть виявити `snort_inline honeypot`, аналізуючи вихідні пакети. Якщо вихідний пакет втрачено, це може виглядати для зловмисника як чорна діра. Коли `snort_inline` модифікує вихідний пакет, зловмисник може перехопити модифікований пакет через іншу хост-систему і визначити модифікацію пакета.

Виявлення `honeypot` буде залежати від аналізу вмісту і часу відповідей. Такий аналіз потребуватиме часу для зловмисників, щоб виявити та перевірити існування `honeypot` в мережі організації. Втім, вже існують рішення, які спрощують процес виявлення `honeypot` в системі [23-26]. Окрім традиційних програмних детекторів низькоінтегрованих гаманців, активно розробляються системи виявлення гаманців за допомогою штучного інтелекту [27]. Такі інструменти вимагають від зловмисника глибокого розуміння особливостей як системи, що атакується, так і можливих "медових горщиків", які можуть бути використані для захисту такої системи. Інструменти виявлення "медових горщиків" також корисні для аналітиків безпеки. Вони можуть використовувати їх для сканування своїх систем на наявність таких особливостей перед розгортанням їх в Інтернеті або у виробництві, щоб уникнути помилок

Переваги низькоінтерактивних `honeypot`:

- Відносна простота реалізації.
- Простота встановлення та супроводу.
- Легкість налаштування.
- Працює поверх штатної операційної системи.
- Багато приманок, розкиданих по мережі можна об'єднати у систему.

Недоліки низькоінтерактивних `honeypot`:

- Обмежена кількість емульованих сервісів.
- Низька скритність від виявлення
- Низька (порівняно з високоінтерактивними `honeypot`) ефективність при відстеженні дій зловмисника.

Високоінтерактивний `honeypot` - програмний комплекс призначений для емуляції операційної системи вцілому. На відміну від низькоінтерактивного, високоінтерактивний `honeypot` дозволяє набагато частіше переконати хакера в тому, що

той знаходиться на зламаній машині, користується командним рядком або графічним інтерфейсом, виконує на ній команди. Така система виглядає значно реалістичнішою, ніж просто емуляція окремих сервісів – зловмисник розуміє, що частково досяг своєї мети – один із комп'ютерів мережі вже зламаний. Якщо до цього основна інформація, що збирається про хакера, полягала в основному в протоколах сеансів мережевої активності, то тепер хакер усі свої дії здійснює на машині-приманці, що дозволяє протоколювати його діяльність також на системному рівні або заходами операційної системи, або окремими програмами, які збирають всю інформацію про його дії.

Функції високоінтерактивних системнабагато ширші за низькоінтерактивні:

- Збір та контроль даних (прослуховування мережевого трафіку та ведення журналів для подальшого аналізу).
- Виявлення атак і їх джерел атаки.
- Ідентифікація зловмисника та інформації про нього (IP-адреса, протокол передачі даних, порт, країна, User agent, операційна система).
- Контроль і протоколювання дій зловмисника.
- Реагування на дії зловмисника, зокрема блокування його активності.
- Введення зловмисника в оману шляхом приховання або підміни інформації, за якою він може зрозуміти, що атакує не реальну систему, а honeypot, а також шляхом зміни конфігурації системи.

Переваги високоінтерактивного honeypot:

- Максимум інформації про дії зловмисника.
- Зловмиснику важче відрізнити високоінтерактивний honeypot від звичайного вузла.
- Можливість встановлення будь-яких програм, що містять реальні вразливості.
- Можливість виявити раніше невідомі вразливості системи.

Недоліки високоінтерактивного honeypot:

- Необхідність розгортання кваліфікованою командою фахівців.
- Проблема аналізу даних після злому honeypot.
- Наявність демаскуючих ознак. Якщо зловмисник може з допомогою будь-яких ознак визначити, що система є високоінтерактивною, а не реальною, така система перестає бути стійкою до виявлення.
- Можливість використання системи зловмисником в якості інструменту для взлому

Після розгортання високоінтерактивні honeypot вимагають багато уваги і кваліфікації фахівців, що їх використовують. Ці люди повинні забезпечувати якісне обслуговування системи та гарантувати, що під час захоплення honeypot не використовують для атаки справжніх систем.

Для складання уявлення про зловмисника необхідно визначити інформацію, яку збиратиме високоінтерактивна система. Опис атаки на високоінтерактивну систему зазвичай містить інформацію за такими критеріями:

- Масштаб і глибина - масштаб атаки описується кількістю порушених машин, а глибина - це рівень впливу на систему.
- Складність – характеризує рівень знань, необхідні виконання конкретної атаки.
- Маскування – якість приховування слідів своєї присутності в системі зловмисником.
- Джерело атаки – зловмисник має бути, наскільки це можливо, ідентифіковано.
- Вразливість – це недолік у системі/протоколі, що дозволяє здійснити атаку.
- Інструменти - інструменти, використані при атаці, наприклад, руткіти або бекдори.
- Масштаб і глибина можуть бути отримані з частоти атак, ступеня впливу атаки та за ступенем зараження системи.

Маскування зловмисника визначається тим, наскільки якісно він приховує сліди своєї присутності у системі. Вразливість, що використовується зловмисником, повинна бути визначена для ведення подальшої статистики. Це необхідно в силу того, що зазвичай у високоінтерактивній системі присутні відразу кілька вразливостей, статистика збирається за кожною з них. Крім цього, один зловмисник може одночасно атакувати відразу кілька вразливостей і не всі його спроби атак будуть успішними. Джерело атаки зазвичай можна визначити за допомогою метаданих мережових пакетів, проте першоджерело атаки може бути складно ідентифікувати, оскільки зловмисник намагатиметься приховувати свою присутність у системі.

Однак, більшість open source як низько так і високоінтерактивних honeypot давно відома досвідченим зловмисникам. Методики їх ідентифікації і обходу дозволяють виявляти такі honeypot на етапі сканування і огляду системи і не потрапляти в розставлені пастки. До таких методів виявлення відносяться вимірювання round-trip time, відправка пошкоджених пакетів і аналіз відповідей на них, дослідження поведінки функціоналу сервісу, аномалії в поведінці системних викликів, аналіз мережевого трафіку, визначення аномалій апаратного забезпечення та інші [28].

Однак основним загальним недоліком honeypot систем, що виявляються зловмисниками про проникненні в мережу і пошуку вразливих систем для подальшого латерального руху є відсутність правдоподібного мережевого трафіку від систем приманок. Одинокостоячий honeypot, що має вразливості, відкриті порти, але не здійснює взаємодії з рештою мережі дозволяє швидко ідентифікувати його як пастку.

Метою розгортання honeypot зазвичай є або збір інформації для дослідницьких цілей або для забезпечення безпеки у виробничих мережах. Дослідницькі honeypot'и, як випливає з назви, розгортаються для дослідницьких цілей. Вони збирають інформацію про поведінку хакерів, їхні інструменти, методи та методи атак. Крім того, вони виявляють слабкі місця системи, які активно використовуються кі-

берзлочинцями, щоб розробити нові стратегії захисту. Головною метою їх розгортання є отримання нових знань про зловмисників і про те, як супротивники здійснюють зловмисну діяльність.

Дослідницькі honeypot'и, як правило, є високоінтерактивними лабораторіями, що надають кіберзлочинцям більше можливостей для взаємодії, проникнення та контролю над системою. Таким чином, ризики розгортання дослідницьких honeypot'ів вищі, ніж при розгортанні виробничих honeypot'ів. Найчастіше дослідницькі організації, такі як військові, університети та охоронні компанії є тими, хто розгортає ці типи honeypot.

З іншого боку, виробничі honeypot'и переважно розгортаються в межах виробничих мереж корпорацій для зменшення ризиків. Вони часто імітують реальні виробничі системи або послуги і є простими у використанні та розгортанні. Мета створення виробничих honeypot'ів - ввести в оману і зайняти кіберзлочинців, змусивши їх витратити час і ресурси на отримання доступу до фальшивих ресурсів на отримання доступу до фальшивих сервісів. Таким чином, вони дозволяють корпораціям оцінити і виправити внутрішні слабкі місця та досягти більшої безпеки в реальному житті [29].

Honeypot'и мережевих систем. Їхнє завдання - захищати систему, виявляючи атаки та повідомляючи про це системних адміністраторів. Виробничі honeypot'и збирають набагато менше інформації про атаки, порівняно з дослідницькими, і тому в першу чергу є низько взаємодіючими honeypot'ами. Проте, виробничі honeypot'и активно посилюють функції безпеки організації.

Відповідно до моделі безпеки Брюса Шнайера, безпека поділяється на виявлення, запобігання та реагування, а виробничі honeypot'и забезпечують значну цінність у всіх трьох категоріях.

Поширеною проблемою, коли йдеться про виявлення порушень безпеки в організації є величезна кількість журналів даних, які необхідно проаналізувати. Щоб виявити та виявлення атак та експлойтів, такі механізми безпеки, як системи виявлення вторгнень (Intrusion Detection System, IDS), часто використовуються для виявлення атак та експлойтів.

Системи виявлення вторгнень (IDS) часто є стандартними реалізаціями. Однак, вони створюють багато хибнопозитивних сповіщень, що робить процес виявлення ще більш неефективним і трудомістким. Розгортаючи виробничі honeypoti, кількість цих типів сповіщень буде різко зменшуватися. Виробничі honeypot не мають функціонального призначення для авторизованих користувачів, а це означає, що більшість виявлених активностей, пов'язаних з ними, є нелегітимними, а отже, має високу цінність для організації.

Ще однією проблемою IDS є помилкові спрацьовування, які виникають, коли система не в змозі виявити зловмисну активність через нові та незареєстровані методи атак. Honeypoti вирішують цю проблему, оскільки вони виявляють як відому, так і невідому шкідливу активність.

Таким чином, honeypot'и не запобігають проникненню хакерів у виробничі системи. Однак вони додають можливості для запобігання, оскільки зловмисники змушені витрачати час і ресурси на атаку емуляції систем замість реальних систем. Вразливості виявлені в honeypot після компрометації, також можуть бути присутніми в оригінальних виробничих системах, які потім можуть бути виправлені до того, як ними хтось скористається. Оскільки хакери експлуатують конкретні лазівки в honeypot, вони також показують, яку саме інформацію шукають кіберзлочинці.

Для того, щоб правильно відреагувати на інцидент, необхідна детальна інформація про зловмисника, як він потрапив у систему і що він робив, перебуваючи в ній, є важливими факторами для правильного реагування на інцидент. Оскільки виробничі honeypot не виконують жодної фактичної функції для організації, вони можуть бути легко вилучені в будь-який час для проведення криміналістичного аналізу у разі виникнення інциденту. Також зникають побоювання щодо забруднення даних оскільки з системою взаємодіють лише неавторизовані користувачі, а вся перехоплена активність вважається зловмисною. Виробничі honeypot мають велику цінність, оскільки вони надають необхідну інформацію для ініціювання ефективної та швидкої реакції на зловмисні інциденти.

Фізичні honeypot'и включають, як вказує назва, фізичну машину або пристрій. Оскільки вони працюють на реальному обладнанні, їх зазвичай відносять до категорії honeypot'ів з високим рівнем взаємодії. Отже, мета полягає в тому, щоб система була повністю скомпрометована. Як і у випадку з високовзаємодіючими honeypot'ами, фізичні honeypot'и зазвичай дорого встановлювати через вимоги до ресурсів. Крім того, вони можуть забирати багато часу на обслуговування через свою складність. Отже, ці типи не є особливо масштабованими.

Віртуальні вулики, навпаки, є надзвичайно масштабованими. Замість того, щоб замість того, щоб для розгортання кожного honeypot потрібна фізична машина, вони працюють на програмному забезпеченні.

Фізична машина може бути розгорнута на декількох віртуальних машинах, що діють як honeypot. Таким чином, віртуальні горщики значно дешевші, а також потребують простішого розгортання та обслуговування, ніж фізичні. Поширене програмне забезпечення для налаштування віртуальних honeypot'ів є VMWare та User-Mode Linux (UML).

2.3 Переваги і недоліки honeypot'ів

У порівнянні з іншими існуючими механізмами безпеки, які часто використовуються, honeypot'и мають кілька явних переваг.

По-перше, одна з головних переваг honeypot'ів полягає в тому, що всі дії, спрямовані на них, і та взаємодія з ними вважаються зловмисними. Це, в свою чергу, призводить до суттєво менші набори зібраних даних у порівнянні з такими механізмами безпеки, як брандмауери та IDS. На відміну від них, honeypots не потрібно обробляти значні журнали даних згенерованих величезною кількістю мережевого трафіку, що надходить до них. Крім того, їм не потрібно розрізняти, чи є перехоплені пакети легітимними чи ні. Таким чином простір, необхідний для зберігання зібраних даних, значно менший, і вони також уникають виснаження ресурсів. Як брандмауери, так і IDS потенційно не здатні працювати корек-

тно, якщо навантаження трафіку на них стає занадто високим. Якщо таблиці брандмауера таблиці брандмауера переповнюються, він може заблокувати всі з'єднання, навіть санкціоновані.

Аналогічно, IDS можуть відкидати пакети, якщо буфер стає переповненим, що призводить до несанкціонований трафік може пройти.

По-друге, розмір наборів даних honeypot робить аналіз інформації набагато більш керованим. Honeypots дозволяють вивчати всі типи атак, як відомих, так і невідомих (нульових днів), оскільки вони відстежують всі дії, які на них на них. Як зазначалося раніше, вони можуть отримувати розвіддані, пов'язані з зловмисника, наприклад, в якій точці світу знаходиться зловмисник, які методи і прийоми, а також які інструменти використовуються. Своєю чергою, це може бути використано для покращити інформаційну безпеку та запобігти майбутнім атакам. Крім того, немає потреби у великих ресурсах та надмірному бюджеті, оскільки практично будь-яка система, комп'ютер чи пристрій практично будь-яка система, комп'ютер або пристрій можуть бути використані як "медовий горщик". Крім того, вони відносно легко встановлювати, налаштовувати та обслуговувати. Їм не потрібно отримувати великі бази даних, що містять підписи, які потрібно постійно оновлювати та підтримувати. Крім того, немає потреби в розробці складних алгоритмів або правил, які потенційно можуть призвести до неправильних конфігурацій.

Як зазначалося вище, кількість ризиків і недоліків, пов'язаних з honeypot'ами варіюється залежно, наприклад, від ступеня їхньої складності. Незважаючи на те, що підводних каменів не так багато, вони є причиною того, що honeypot не є адекватною заміною повністю замінити сучасні стандартні механізми безпеки. Тому зазвичай вони співіснують з такими механізмами безпеки, як брандмауери та IDS, щоб зробити свій внесок у загальну систему безпеки системи.

Одним з головних недоліків honeypots є те, що це може бути складним завданням зробити їх надійними для досвідчених кіберзлочинців. Досвідчені кіберзлочинці здатні проводити розвідку і ідентифікацію системи, а це означає, що вони можуть ідентифікувати справжню систему, оскільки вона має певні очікувані характеристики або поведінку.

Зловмиснику достатньо такої простої помилки, як орфографічна, щоб зрозуміти, що він або вона не взаємодіє з реальною системою. Це може мати критичні наслідки як для виробничих, так і для дослідницьких платформ. Якщо зловмисник виявить, що компанія використовує honeypots у своїй виробничій мережі, він може ввести організацію в оману підробляючи атаки проти неї. Це призведе до генерування хибних тривог, які надсилатимуться адміністратору, в той час як зловмисник виконує справжні атаки на реальну виробничу систему.

Для дослідницьких систем це ще більший ризик. У разі виявлення зловмисники можуть надати неправдиві або некоректні дані, щоб не бути виявленими. Висновки на основі цієї інформації дадуть хибне уявлення дії зловмисників.

Іншим фактором, що впливає на достовірність даних є здатність зловмисників видавати себе за інші комп'ютерні системи, приховуючи свою справжню ідентичність. Зловмисники можуть підмінити IP-адресу джерела атакуючого трафіку, використовуючи такі заходи таких як VPN-сервіси або проксі-сервери, що призводить до отримання невірної інформації про походження атаки.

Ще одним суттєвим недоліком є те, що вони можуть відстежувати активність лише в тому випадку, якщо якщо зловмисник безпосередньо націлений на них. Вони не здатні зібрати жодних даних про атаки, якщо вони здійснюються проти будь-якої іншої системи в мережі. Отже, навіть якщо дані, зібрані в ідеальному випадку, мають дуже високу цінність, обмежене поле зору бджіл може виключити події, що відбуваються навколо них.

Нарешті, існує ризик захоплення honeypot'а хакером. Як згадувалося вище, ризик зростає зі збільшенням складності. Якщо зловмисник має повний доступ до операційної системи зловмиснику повний доступ до ОС, має більше шансів бути скомпрометованим, ніж той, що лише імітує невеликий набір сервісів. Чим більше можливостей взаємодії має зловмисник, тим тим більша ймовірність того, що він або вона отримає доступ до реальної системи. Потенційним недоліком успішного поглинання є те, що успішного поглинання полягає в тому, що honeypot може бути використаний для запуску пасивних або активних атак на інші системи як самостійно, так і в складі ботнету.

Коли зловмисник компрометує honeypot, це дозволяє оператору цієї honeypot'f повністю проаналізувати шлях атаки. Це може призвести до небажаної уваги або навіть викрити самого зловмисника, якщо будуть знайдені певні вказівники, наприклад, у двійковому коді або в інфраструктурі командного управління. З цієї причини шкідливе програмне забезпечення зазвичай має функціонал, який виявляє середовище- honeypot, і в цей момент шкідливе програмне забезпечення може перервати свою атаку. Оскільки ми зацікавлені в аналізі поведінки зловмисників і відстеженні супротивників у часі, важливо замаскувати індикатори honeypot від зловмисників.

Відомий приклад такої функціональності був присутній у програмі-вимагачі WanaCrypt. Дослідник безпеки, відомий як MalwareTech, знайшов доменне ім'я під час аналізу зразка шкідливого програмного забезпечення, яке ще не було зареєстроване. Honeypot'и часто зберігають всі DNS-запити, намагаючись обдурити шкідливе програмне забезпечення, щоб воно перенаправляло трафік на систему, контрольовану дослідником. У цьому випадку WanaCrypt припустив, що якщо цей домен дозволений, то він повинен працювати в контрольованому середовищі, що зупинить подальше зараження. Зареєструвавши цей домен, шкідливе програмне забезпечення змогло зберегти домен в глобальному масштабі, чого воно не очікувало в реальному середовищі. Таким чином, вона помилково вважала, що всі системи є середовищем, схожим на honeypot, що фактично зупинило її подальше поширення. Хоча це був погано реалізований метод виявлення honeypot, сам принцип залишається важливим аспектом для розгляду.

2.4. Огляд IoT honeypotів

Кількість атак на промислові, муніципальні, медичні мережі, а також на елементи критичної інфраструктури, що мають в своєму складі IoT пристрої щороку

збільшується на 15-20% по всьому світу [30]. Зважаючи на постійну загрозу кібератак на елементи критичної інфраструктури України цьому питанню потрібно приділяти особливу увагу.

Наслідки атак на подібні системи можуть бути наступними:

- Відмова в обслуговуванні. Найбільша кількість атак, що здійснюються, призводить відмові в обслуговуванні, а саме до несправностей, що призводять до часткового або повного припинення роботи вбудованого пристрою.
- Виконання шкідливого коду. Наслідком атаки може бути виконання пристроєм шкідливого коду, введенного зловмисником. Це також включає різні веб-скрипти і SQL-ін'єкції, що може змінити поведінку пристрою.
- Порушення цілісності. Результатом атак є порушення цілісності деяких даних або вихідного коду прошивки пристроїв. Це включає зміну файлів і параметрів конфігурації, а також додатків на пристрої.
- Витік інформації. У деяких випадках результат атаки – це несанкціоноване отримання зловмисником певної інформації.
- Несанкціонований доступ. Багато атак призводять до того, що зловмисник отримує несанкціонований доступ до пристрою. Це не тільки включає випадки, коли зловмиснику, який не має доступу до пристрою, вдається логічно його отримати, але також і випадки, коли зловмисник, маючи доступ, підвищує привілеї.
- Зниження рівня захищеності пристрою. Зловмисник своїми діями може призвести до того, що пристрій буде використовувати слабші алгоритми або політики безпеки, ніж ті, що він фактично підтримує.

В наш час використання honeypot в мережах загального призначення є доволі суперечливою і часто неефективною практикою через високу вірогідність їх виявлення. Однак застосування спеціалізованих honeypot/honeynet в IoT мережах досі є доволі ефективним [31]. Через особливості архітектури і протоколів зв'язку з IoT пристроями використання навіть низькоінтенсивних honeypot є ефективним маркером початку атаки. Особливо якщо розміщувати такі honeypot/honeynet системно і відслідковувати актуальні загрози безпеки IoT пристроїв для модифікації прима-

нок. Щоб збільшити збір даних і отримати краще розуміння про загрози, приманкам доцільно використовували різні рівні взаємодії. Крім того, потрібно циклічно змінювати їх IP-адреси, щоб приманки не були позначені як приманки, що зменшить кількість атак і кількість корисної інформації, яку можна було зібрати.

Без сумніву, SSH, Telnet і веб-сервери є одними з найбільш часто використовуваних і доступних послуг в сфері Інтернету речей, що робить їх привабливою мішенню для зловмисників. Крім того, що пристрої IoT зазвичай використовують безліч обчислювальних архітектур, які суттєво відрізняються від тих, що використовуються традиційними комп'ютерними мережами. Саме через це зловмисники з більшою ймовірністю запуснуть своє програмне забезпечення, коли вони отримають доступ до honeypot і не перевірять, яку архітектуру вони використовують. Завдяки цьому дослідники можуть відстежити джерела інструментів атак, які використовували зловмисники, що дозволяє їм набагато ефективніше вивчати їх пізніше.

Пристрої IoT мають певні особливості які потрібно враховувати при створенні honeypot/honeynet [32]. Щоб максимізувати шанси хакера знайти та використати вразливості, важливо, щоб honeypot залишався анонімним, імітуючи реальну систему, щоб запобігти його легкому ідентифікуванню зловмисниками. Через особливість пристроїв Інтернету речей, а також через нездатність повністю зрозуміти природу та діяльність зловмисника, для ефективної приманки знадобиться інший підхід.

IoT honeypot успадковують деякі характеристики від honeypot загального застосування, включаючи здатність реагувати на події, коли вони виникають. Незважаючи на те, що ці honeypot не розроблені спеціально для IoT, наразі вони іноді використовуються для дослідження IoT honeypot. Прикладом є Honeyd, що дозволяє не тільки створювати віртуальні медіа, але й інтегрувати машини. Є кілька протоколів, які підтримуються цим honeypot, зокрема: UDP, TCP, FTP, SMTP, Telnet, IIS, POP і telnet. Різноманітні дослідження досліджували, чи можна використовувати HoneyD для створення ефективних honeypot, які приваблюють зловмисників.

Dionaea [33] — це програмне забезпечення з відкритим вихідним кодом, яке дозволяє користувачам створювати приманки середньої взаємодії, які імітують різні служби (наприклад, FTP, HTTP, MQTT тощо). Ця програма спрямована на зловмисників, які атакують хости в Інтернеті за допомогою вразливих служб. За допомогою Cowrie можна створювати масштабовані honeypot середнього та високого рівня взаємодії, які можуть контролювати та контролювати різні поведінки. Як honeypot середньої взаємодії, він реєструє взаємодію оболонки зловмисника в імітованій системі UNIX за допомогою емуляції кількох команд. Як honeypot з високою взаємодією, це проксі для SSH і Telnet для спостереження за взаємодією зловмисника в іншій системі. По суті, він діє як проксі-сервер між зловмисником і групою віртуальних машин, які налаштовані на сервері серверу, що дозволяє гнучку конфігурацію.

Найуніверсальніші IoT honeypot здатні емулювати будь-який пристрій, підключений до Інтернету. З повною емуляцією пристрою зловмисникам важче виявити приманку, що додає honeypot більшої реалістичності. За допомогою платформи ThingPot повну платформу Інтернету речей можна емулювати та підтримувати на рівні додатків, гарантуючи, що ваша система Інтернету речей є масштабованою, віртуальною, відкритою та масштабованою.

Також варто зазначити IoT CandyJar [34], що може відтворювати поведінку пристроїв IoT без ризику бути скомпрометованими, оскільки вони розумні та імітують поведінку автентичних пристроїв IoT. Вони називаються приманками інтелектуальної взаємодії.

Conpot [35] є одним із найпопулярніших honeypot ICS, який використовується дослідниками протягом багатьох років. Conpot підтримує багато промислових протоколів, у тому числі Building Automation and Control Network, Guardian AST, Kamstrup, Modbus, S7comm і багато інших, як-от HTTP, FTP, SNMP, Intelligent Platform Management Interface і TFTP. У комплекті є шаблони для ПЛК класу Siemens S7, систем моніторингу резервуарів Guardian AST, а також інтелектуальних лічильників Kamstrup.

Kippo2 [36] - популярний SSH-софт, написаний на Python. Наразі він більше не розробляється, але був форкований Мішелем Остерхофом (Michel Oosterhof), який назвав його Cowrie3. Cowrie - це SSH- і Telnet-софт із середнім рівнем взаємодії, призначений для ведення журналу атак грубої сили і взаємодії з командним інтерфейсом, що виконується зловмисником. Крім типових функцій з низьким рівнем взаємодії, таких як збір облікових даних і підроблені оболонки, можна перенаправляти невідомі команди на екземпляр SSH, використовуючи функцію "Виконати". Ця функція не вважається справжньою пасткою з високим рівнем взаємодії, виходячи з визначення з, але через динамічну природу її часто називають середнім рівнем взаємодії.

У Cowrie є багато функціональних можливостей, які роблять його схожим на honeypot з високою взаємодією. Наприклад, є підтримка фальшивої файлової системи, тому якщо зловмисник намагається отримати доступ до файлів через емульовану оболонку, деякі файли насправді доступні. Це дозволяє користувачам конфігурувати honeypot з високим рівнем гнучкості, але з набагато меншим ризиком, ніж типовий honeypot з високим рівнем взаємодії.

Ще однією популярною програмою для взаємодії в середовищі є Kojoney24. Він пропонує схожу функціональність, що й Cowrie, але зосереджений на SSH, а не на SSH і Telnet. У Kojoney2 запити на завантаження, зроблені зловмисниками, фактично виконуються у фоновому режимі, щоб їх можна було проаналізувати пізніше. Ці файли не можуть бути використані в емульованому середовищі, тому немає ризику зараження через виконання бінарних файлів.

2.5 Honeypot'и специфічні для шкідливого програмного забезпечення

Крім того, доступні також різноманітні специфічні для Mirai honeypot. На основі аналізу шкідливого програмного забезпечення Mirai, ці honeypot розроблені таким чином, щоб реакції на відомі команди відповідали критеріям відбору Mirai.

Це робить їх дуже корисними для збору зразків шкідливого програмного забезпечення Mirai і для дослідження зловмисників з дуже схожим шляхом атаки, як у Mirai. Однак для більш загальних досліджень і виявлення раніше невідомого шкідливого програмного забезпечення ці honeypot недостатньо просунуті. Ці реалізації з низьким рівнем взаємодії можуть бути корисними для початкового збору даних, але для більш просунутих вимог потрібен високоінтерактивний honeypot.

МTPot - це проект Symmetria5, призначений для захоплення семплів Mirai. Він призначений для прослуховування Telnet-з'єднань і поводить себе саме так, як очікує Mirai, запускаючи завантаження семплів. Це чистий медовий горщик з низьким рівнем взаємодії, із заздалегідь визначеними очікуваними командами і відповідними відповідями. Програма корисна для дослідження обставин, за яких Mirai спробує заразити комп'ютер. Однак, в цілому в цілому, він занадто простий і йому бракує багатьох параметрів конфігурації, щоб відповідати нашим цілям.

Mirai знаходить своїх жертв, скануючи Інтернет у псевдовипадковому порядку псевдовипадковому порядку за допомогою TCP SYN-зондів. Ці сканування виконуються на портах 23 і 2323, і певні діапазони IP-адрес заносяться в чорний список, щоб запобігти приверненню уваги відомих компаній або урядів. Після того, як потенційна жертва знайдена за допомогою сканування, встановлюється з'єднання Telnet встановлюється з'єднання Telnet і вибираються 10 випадкових записів з попередньо налаштованого списку з 62 облікових даних. У разі успішного входу в систему, IP-адреса жертви та правильні облікові дані надсилаються на сервер звітів. Окремо від сервера звітів, програма-завантажувач входить на комп'ютер жертви, використовуючи отримані облікові дані.

Ця програма-завантажувач визначає базову архітектуру, після чого завантажує і виконує конкретне шкідливе програмне забезпечення. Після успішного зараження Mirai намагається сховатися, видаляючи виконувані файли і використовуючи псевдовипадкові імена для свого процесу. В результаті, Mirai не зберігається при перезавантаженні, що ускладнює розслідування.

Щоб запобігти зараженню пристрою іншими шкідливими програмами, процеси, прив'язані до портів 22 і 23, а також відомі процеси від конкуруючих шкідливих програм, таких як .anime і Qbot, знищуються. Нарешті, заражений пристрій буде прослуховувати трафік команд і управління, одночасно скануючи інші потенційні жертви за допомогою описаного вище методу.

Там, де інфраструктура контролю та управління спочатку базувалася на IP-адресах, пізніше вона була переведена на доменний в середині вересня. Крім того, приблизно в той самий час шкідливе програмне забезпечення почало видаляти свої власні двійкові файли та приховувати ідентифікатор процесу. В одному з останніх зразків, зібраних перед випуском вихідного коду, були виявлені такі особливості, як закриття портів зараження і агресивне знищення конкурентних інфекцій спостерігалось закриття портів зараження та агресивне знищення конкурентних інфекцій. Після релізу шкідливе програмне забезпечення швидко еволюціонувало, а різні учасники додавали різні функції. Ці функції включають нові набори облікових даних, зміни в чорному списку IP-адрес і сканування на портів, пов'язаних з протоколом CWMP.

На основі набору даних DNS у поєднанні з отриманими зразками шкідливого програмного забезпечення, Антонакакіс та ін. змогли кластеризувати IP-адреси серверів управління і контролю та спільну мережеву інфраструктуру. Ці кластери сильно відрізнялися за розміром: найменший з них складався з одного хоста, а найбільший містив 11 доменів з 92 IP-адресами.

Оскільки ці кластери не мають спільної інфраструктури, цілком можливо, що кілька суб'єктів активно керували ботнетами, використовуючи Mirai. Ці кластери демонстрували різні рівні еволюції, але велика кількість функцій не дорівнює успішним обсягам зараження.

За період моніторингу було зафіксовано тисячі DDoS-атак, здійснених Mirai. Поряд з гучними атаками на "Krebs on Security", Dyn та ліберійський Lonestar, Mirai продемонстрував ті ж характеристики, що і так звані сервіси-стрейнери або бутери. Ці послуги дозволяють клієнтам платити за DDoS-атаки на обрані ними цілі. Деякі оператори Mirai націлювалися на ігрові платформи, такі як Steam, Minecraft і

Runescape. Ці атаки використовували широкий спектр стратегій, таких як об'ємні атаки, виснаження стану TCP та атаки на рівні додатків. Цікаве спостереження було зроблено щодо однієї з цілей атак. На 7-му місці за популярністю була IP-адреса, пов'язана з одним із серверів командування і управління. З 484 спостережуваних IP-адрес командно-контрольних серверів Mirai 47 самі були об'єктами DDoS-атак.

При кластеризації за командами атаки було виявлено 93 унікальних кластери, з яких 26 кластерів були атаковані принаймні один раз. Це підтверджує гіпотезу про те, що Mirai насправді складається з декількох конкуруючих операторів ботнету.

Якщо звичайні DDoS-атаки покладаються на стратегії ампліфікації, то Mirai в основному використовував прямі невідображаючі атаки. Навіть не покладаючись на стратегії посилення, Mirai все одно був здатен завдати серйозної шкоди, про що свідчать згадані вище гучні атаки.

Ботнет Mirai показує, що навіть відносно прості словникові атаки можуть призвести до компрометації сотень тисяч пристроїв IoT. У майбутньому, ймовірно, пристрої Інтернету речей також стануть мішенню для програмних вразливостей, тому виробникам вкрай необхідно адаптувати набагато суворішу стратегію захисту для своїх пристроїв.

Ще одним відомим ботнетом, що складається з пристроїв Інтернету речей, є Najime. Najime був детально проаналізований Edwards та Profetis. Було виділено три етапи, а саме Етап розвідки та зараження Етап розвідки Najime проходить через сеанс Telnet, дуже схожий на інші шкідливі програми для IoT, про які ми говорили раніше. Він сканує Інтернет, використовуючи алгоритм рандомізації який генерує випадкову IP-адресу, до якої встановлюється з'єднання. Коли з'єднання з'єднання вдається встановити через порт 23, він послідовно перевіряє список облікових даних. Це суттєва відмінність від Mirai та його різновидів, які перевіряють облікові дані у випадковому порядку. Коли комбінація імені користувача/пароля відхиляється, Najime від'єднується, а потім знову підключається, щоб спробувати інший набір облікових даних. Таким чином, за один сеанс можна спробувати лише одну комбінацію за сеанс, при цьому продовжуючи переглядати словник.

Після того, як набір облікових даних прийнято, виконується низка команд для визначення придатності скомпрометованого пристрою. Вибирається відповідне місце у файловій системі і визначається архітектура системи шляхом перевірки програми "відлуння". Після того, як ціль вважається придатною для зараження, двійковий файл завантажується шляхом введення закодованих байтів через сеанс терміналу.

Двійковий файл, завантажений наприкінці першого етапу, є простою програмою, метою якої є підключення до віддаленого сервера і збереження всіх отриманих байтів у файлі. Автори помітили, що кожна архітектура отримує окремий двійковий файл, написаний вручну спеціально для цієї архітектури. Вони дійшли такого висновку на основі двох індикаторів, які є характеристиками, що не підходять для компілятора, який може створювати виконувані файли для різних архітектур з одного і того ж вихідного коду. Завантажувач DHT Двійковий файл, завантажений на попередньому етапі, є завершальною стадією хробака Najime.

Цей двійковий файл відповідає за отримання і виконання додаткового корисного навантаження за допомогою однорангової мережі (P2P), що складається з інфікованих хостів. Він використовує кілька протоколів, що застосовуються в BitTorrent. Наприклад, протокол DHT використовується для виявлення інших інфікованих хостів, а транспортний протокол uTorrent використовується для обміну даними по мережі.

З внутрішньої роботи бінарного файлу на фінальній стадії видно, що Najime підтримує низку різних модулів. До моменту публікації в Інтернеті був помічений лише модуль експлуатації "exp". Інші модулі наразі не активні, хоча Najime підтримує завантаження нових і, можливо, шкідливих модулів через свою P2P-мережу, що все ще становить серйозну загрозу.

Висновок до Розділу 2

В розділі 2 розглядаються різноманітні honeypot і honeynet системи в якості засобу захисту мережевої інфраструктури і виявлення атак зловмисників. Були розглянуті основні наукові публікації за тематикою і напрямки досліджень. Було проаналізовано переваги і недоліки honeypot систем як загального призначення так і спеціалізованих рішень для вузьких задач.

Постійна еволюція інструментарію, технік, тактик і процедур зловмисників змушують їх шукати нові і нові способи виявлення honeypot'ів та введення в оману IDS/IPS систем. Засоби виявлення honeypot'ів, що їх використовують досвідчені кіберзлочинці дозволяють їм доволі часто оминати всі пастки захисників реальних мереж загального призначення, що значно знижує відсоток ефективності використання honeypot'ів в подібних мережах.

Проте для мереж «Інтернету речей» honeypot'и досі залишаються доволі дієвим інструментом виявлення атак, оскільки по відношенню до систем IoT зловмисники зазвичай використовують автоматизовані рішення, що діють менш вибірково і працюють більш «по площі», а не точково. Тому навіть низькоінтерактивні honeypot системи для IoT залишаються вкрай ефективними.

РОЗДІЛ 3. РОЗГОРТАННЯ І НАЛАШТУВАННЯ ІОТ HONEYPOT ДЛЯ ДОСЛІДЖЕННЯ БЕЗПЕКИ СИСТЕМИ «РОЗУМНИЙ ДІМ»

3.1 Вибір ssh-honeypot для системи «Розумний дім»

Зважаючи на проаналізовану в Розділах 2 і 3 інформацію про основні напрямки атак на IoT системи взагалі та на системи «Розумний дім» зокрема, а також, беручи до уваги популярність ssh як засобу підключення для налаштування систем керування «Розумним будинком», вибір IoT honeypot для даного дослідження буде проводитись серед низько- і середньоінтерактивних ssh-honeypot рішень. Розглянемо найвідоміші сучасні ssh-honeypot системи.

Kippo [36] - SSH-honeypot, написаний на Python, був розроблений для виявлення та реєстрації атак грубої сили і, що найголовніше, повної історії командного інтерпретатора, виконаного зломисником. Він доступний для більшості сучасних дистрибутивів Linux і пропонує як керування та налаштування за допомогою клієнтських команд, так і веб-інтерфейс. Kippo пропонує фальшиву файлову систему і можливість пропонувати зломисникам фальшивий вміст (наприклад, файли паролів користувачів тощо), а також потужну систему статистики під назвою Kippo Graph.

Blacknet 2 [37] - це низькоінтерактивна SSH-система з двома компонентами (клієнт і сервер) з можливістю ведення журналів. Вона дозволяє використовувати їх для збору всіх спроб SSH, що виконуються на декількох IPv4-адресах наших пристроїв, а також для отримання та експорту статистики. Спеціальний веб-інтерфейс дозволяє в режимі реального часу відстежувати, що відбувається на пристроях, які IP-адреси підключались до системи і звідки.

Kojoney2 [38] - це SSH-honeypot для взаємодії в середовищі, написаний на Python з використанням бібліотек Twisted Conch. Kojoney2 буде слухати на порту 22 для з'єднань від клієнтів ssh. Після спроби з'єднання Kojoney2 автентифікує користувачів, порівнюючи надані імена користувачів і паролі з внутрішнім списком фальшивих користувачів. Більшість облікових даних буде прийнято, надаючи зломисникам доступ до імітованої оболонки, де вони можуть віддавати команди.

Kojoney2 імітує відповідь на багато легітимних команд оболонки, щоб обдурити зловмисників. На відміну від honeypot'ів з низьким рівнем взаємодії, Kojoney2 фактично завантажує файли, запитані зловмисником за допомогою команд `wget` або `curl`, використовуючи власні бібліотеки пошуку URL-адрес Python. Ці файли зберігаються в пісочниці в каталозі завантаження для аналізу, але вони не з'являються в імітованій оболонці Kojoney2. Завантажені файли порівнюються за допомогою `md5sum(1)` з існуючими файлами, щоб запобігти дублікатам (і відмові в обслуговуванні через вичерпання ресурсів файлової системи).

Cowrie honeypot [38], розроблений Мішелем Остерхофом, - це система, призначена для перехоплення як Telnet, так і SSH-з'єднань. Вона заснована на принципі середньої взаємодії. Cowrie працює шляхом емуляції командної оболонки. Він пропонує фальшиву файлову систему на основі Debian 5.0, що дозволяє вам додавати і видаляти файли за вашим бажанням. Ця програма також зберігає всі завантажені і вивантажені файли в безпечній і карантинній зоні, щоб ви могли виконати подальший аналіз, якщо це необхідно. Окрім емуляції оболонки SSH, вона може використовуватися як проксі SSH і Telnet, а також дозволяє перенаправляти SMTP-з'єднання на інший SMTP-сервер.

Проаналізувавши вищезгадані ssh-honeypot'и на основі критеріїв функціоналу, зручності розгортання і моніторингу, можливості використання додаткових скриптів для обробки результатів їх роботи і часу їх останнього оновлення, в якості ssh-honeypot'а для системи «Розумний дім» на базі платформи Raspberry Pi було обрано Cowrie.

Cowrie містить фальшиву файлову систему, що дозволяє додавати та видаляти файли. Більше того, можна додати фальшивий вміст файлів, щоб зробити honeypot більш достовірним, щоб зловмисник міг прочитати такі файли, як `/etc/passwd`. За замовчуванням, honeypot містить повну фальшиву файлову систему, що нагадує інсталяцію Debian, що підходить для Raspberry Pi OS. Втім існує можливість вибрати іншу файлову систему також можна вибрати іншу файлову систему для емулявання honeypot для інших задач.

Нарешті, всі файли, завантажені зловмисниками в honeypot, зберігаються для подальшого вивчення. Cowrie пропонує зловмиснику обмежену взаємодію, коли при роботі в якості Medium Interaction Honeypot. Однак, в Cowrie є більш широкі можливості для налаштування щоб зробити його більш реалістичним ніж інші вищезгадані ssh-honeypot.

Тим не менш, існує імовірність, що Cowrie може бути ідентифікований зловмисниками, оскільки існують автоматичні скрипти, які можуть виявити, що взаємодія відбувається з цим типом honeypot.

3.2 Розгортання і налаштування платформи та IoT honeypot

В якості основи для системи керування «Розумним будинком» було обрано доволі традиційне рішення на базі Raspberry Pi 3. Ця платформа дозволить побудувати honeypot з низьким рівнем взаємодії, який би емулював реальний сервіс на пристроях і генерував "невелику" поверхню атаки у внутрішньому середовищі, щоб відстежувати будь-які дії зловмисників.

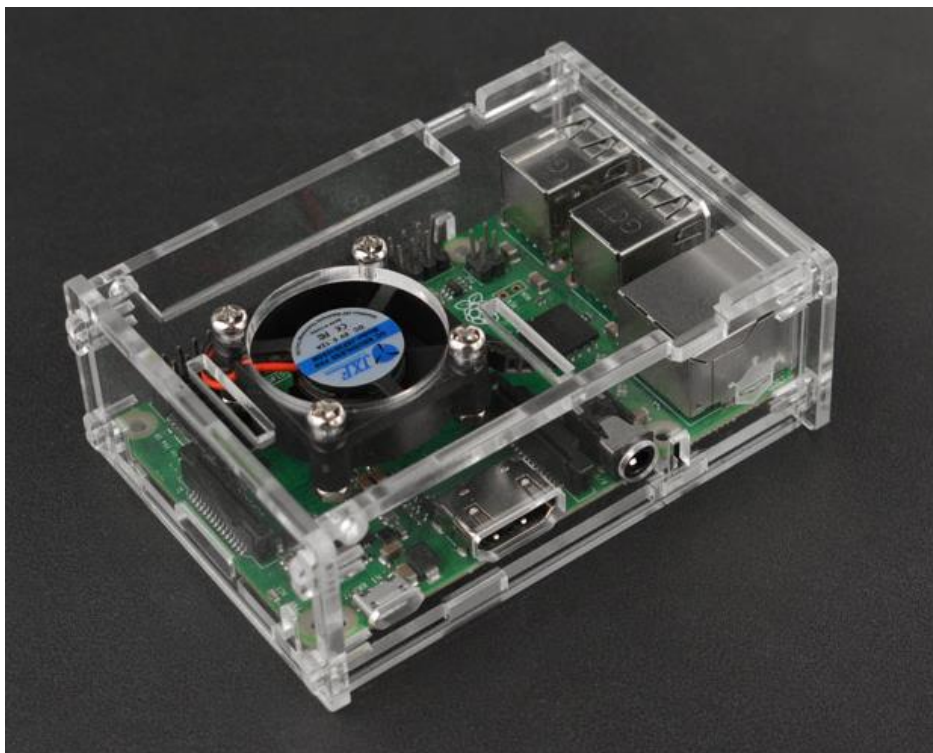


Рисунок 3.2 Raspberry Pi 3B

В якості платформи було використано Raspberry Pi Model 3B, що має наступні характеристики:

- 1GB RAM.
- 64GB Micro SD card.
- Wireless/LAN.
- Raspberry Pi OS

В якості скриптів для взаємодії компонентів було використано скрипти на Python. Для моніторингу конфігурації вбуло використано Supervisorд

Формат логування:

- в /var/log кожного honeypot генерується текстовий лог
- syslog сконфігуровано для можливого подільшого використання SIEM
- tcpdump було використано для зберігання пакетів

Підготовка та налаштування Raspberry Pi

1. Встановлення Pi OS

1.1 З офіційного сайту проекту (<https://www.raspberrypi.org/software/>) було завантажено Raspberry Pi OS imager. З його допомогою на sd-карту було записано нову версію операційної системи Raspberry Pi OS with Desktop. Після прошивки на SD-карті в кореневому розділі було створено порожній файл з ім'ям SSH (без розширення).

1.2 Після підключення по SSH до Raspberry Pi з іменем користувача *pi* і паролем *raspberry* було виконано наступні підготовчі дії:

- Змінено пароль за замовчуванням *pi* при завантаженні.
- Перевірено, чи можна виконати `sudo`, набравши "`sudo -i`"
- Запущено `raspi-config` для налаштування параметрів системи.
- Виконано `rpi-update`

2. Встановлення Supervisor. Він використовувався для моніторингу працездатності системи. Він також використовувався для запуску python-скрипта і його перезапуску, якщо було виявлено будь-які помилки. Налаштування виконувалось наступним чином:

- `sudo apt-get install supervisor`
- `cd /etc/supervisor`
- `sudo supervisor start`".

```
; supervisor config file[unix_http_server]
file=/var/run/supervisor.sock ; (the path to the socket file)
chmod=0700 ; sockef file mode (default 0700)
chown = nobody:nogroup
username = ## (UserName)
password = ## (Password)[supervisord]
logfile=/var/log/supervisor/supervisord.log ; (main log file;default
$CWD/supervisord.log)
pidfile=/var/run/supervisord.pid ; (supervisord pidfile;default supervi-
sord.pid)
childlogdir=/var/log/supervisor ; ('AUTO' child log dir, default
$TEMP)
user=root
; the below section must remain in the config file for RPC
; (supervisorctl/web interface) to work, additional interfaces may be
; added by defining them in separate rpcinterface: sections
[rpcinterface:supervisor]
supervisor.rpcinterface_factory = supervisor.rpcinterface:make_main_rpcin-
terface[supervisorctl]
serverurl=unix:///var/run/supervisor.sock ; use a unix:// URL for a unix
socket
username = ## (Your UserName)
password = ## (Your Password)
; The [include] section can just contain the "files" setting. This
; setting can list multiple files (separated by whitespace or
; newlines). It can also contain wildcards. The filenames are
; interpreted as relative to this file. Included files *cannot*
; include files themselves.[include]
files = /etc/supervisor/conf.d/*.conf
```

Рисунок 3.2.1 Вміст конфігураційного файлу supervisord.conf

3.3 Розгортання і налаштування Cowrie

Cowrie – це дуже популярний medium/low interaction ssh-honeypot для логування bruteforce-атак та дослідження взаємодії злоумисника з командним рядком.

Для роботи було налаштовано наступний функціонал:

- Підробна файлова система.

- Вхід по SSH з паролем root/password (встановлюється в userdb.txt)
- Підробний SSH-сервіс налаштовано на запуск через порт 22 з автоприв'язкою, в той час як справжній SSH налаштовано на запуск через порт 3393.
- Було створено окремий *account cowrie* без пароля, тобто тільки користувач root міг виконувати sudo до цього облікового запису і виконувати відповідні сервіси.
- Щоб обмежити взаємодію цього honeypot, і щоб дозволяти зловмисникам входити наосліп, в файлі /home/cowrie/cowrie/etc/userdb.txt було створено фальшивий обліковий запис root з 12 символами. Тільки цей обліковий запис мав доступ до цієї фальшивої файлової системи.

Покрокове налаштування виконувлось наступним чином:

1. Було змінено порт SSH за замовчуванням на порт 3393 (оскільки cowrie буде розміщувати фальшиву SSH-систему на порту 22)
2. Для цього в терміналі було введено команду

```
$ nano /etc/ssh/sshd_config
```

3. Відредаговано порт за замовчуванням з 22 на 3393
4. Збережено і закрито файл
5. Перезавантажено сервіс sshd
6. Після перезавантаження наступні підключення по ssh до сервера (Raspberry Pi) виконувались використовуючи новий порт 3393 (тобто <ip>:3393)

Далі було встановлено залежності

```
$ sudo apt-get install git python-virtualenv libssl-dev libffi-dev build-essential libpython3-dev python3-minimal authbind virtualenv
```

Створено аккаунт користувача

```
$ sudo adduser --disabled-password cowrie
Adding user 'cowrie' ...
Adding new group 'cowrie' (1002) ...
Adding new user 'cowrie' (1002) with group 'cowrie' ...
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
```

```

Home Phone []:
Other []:
Is the information correct? [Y/n]

$ sudo su - cowrie

```

Виконано встановлення і активація віртуального оточення

```

$ pwd
/home/cowrie/cowrie
$ virtualenv --python=python3 cowrie-env
New python executable in ./cowrie/cowrie-env/bin/python
Installing setuptools, pip, wheel...done.

$ source cowrie-env/bin/activate
(cowrie-env) $ pip install --upgrade pip
(cowrie-env) $ pip install --upgrade -r requirements.txt

```

Далі було налаштовано Authbind для запуску служб від імені не-користувача (non-root) на порту 22

```

$ sudo apt-get install authbind
$ sudo touch /etc/authbind/byport/22
$ sudo chown cowrie:cowrie /etc/authbind/byport/22
$ sudo chmod 770 /etc/authbind/byport/22
Edit bin/cowrie and modify the AUTHBIND_ENABLED settingChange
the listening port to 22 in cowrie.cfg:
[ssh]
listen_endpoints = tcp:22:interface=0.0.0.0

```

Наступною дією було налаштування моніторингу Cowrie в Supervisor. Для цього в файл `/etc/supervisor/conf.d/cowrie.conf` було поміщено наступну конфігурацію:

```

program:cowrie]command=/home/cowrie/cowrie/bin/cowrie startdi-
rectory=/home/cowrie/cowrie/
user=cowrie
autostart=true
autorestart=unexpected
exitcodes=1
redirect_stderr=true
stdout_logfile=NONE

```

Важливим етапом було внесення змін до `cowrie.cfg`. для того, щоб він надсилав лог до `syslog`. Для цього було внесено зміни в налаштуваннях:

```
[output_localsyslog]
facility = SYSLOG
format = text
```

Для подальшого дослідження корисного навантаження, що його потенційно можуть спробувати записати зловмисники, було налаштовано ввімкнення `tcpdump` під час завантаження. `Tcpdump` використовується для увімкнення перехоплення пакетів на портах 22,445, 3389, 443 і 80. Перехоплені пакети зможуть записати корисне навантаження, виконане зловмисниками на `honeypot`, і надати додаткове джерело інформації. Мінусом цього підходу на `Raspberry Pi` є те, що `tcpdump` може вичерпати пам'ять. Тому на початку дослідження планувалося додати зовнішній жорсткий диск, проте в подальшому не було виявлено спробу завантажити зловмисниками чогось на `honeypot`.

Налаштування виконувались наступним чином:

```
su pi
cd /home/pi
echo "sudo -u dump tcpdump -p -f port 445 or port 139 or port 3389
or port 80 or port 22 or port 443 or port 7777 -w
/var/log/pcap/trace-%Y-%m-%d-%H-%M-%S-%s.pcap -G 3600 -C 50 " >>
test.sh
chmod 700 test.sh
apt-get install tcpdump
sudo groupadd pcap
adduser -r -s /bin/nologin dump
sudo chgrp pcap /usr/sbin/tcpdump
sudo chmod 750 /usr/sbin/tcpdump
sudo setcap cap_net_raw,cap_net_admin=eip /usr/sbin/tcpdump
```

Далі було встановлено нову конфігурацію супервізора в розділі `/etc/supervisor/conf.d/tcpdump.conf`

```
[program:dump]command=sh /home/pi/test.sh
directory=/home/pi/ user=pi
autostart=true
autorestart=unexpected
stdout_logfile=/var/log/pcap/app.log
stdout_logfile_maxbytes=10MB
stdout_logfile_backups=30
redirect_stderr=true
stderr_logfile=/var/log/pcap/err.log
```

Для ефективної роботи з логами подій на cowrie була виконана конфігурація та налаштовано ведення журналу Syslog (Cowrie було налаштовано для передачі логів в syslog). Керування дозволами на ведення журналу^

```
chown -R smb:smb /var/log/smb
chown -R cowrie:cowrie /var/log/cowrie
chown -R tcruise:tcruise /var/log/rdp
chown -R dump:pcap /var/log/pcap/
```

Для більш зручної роботи з логами було встановлено syslog-ng (програма автоматично пересилає журнал на вказану IP-адресу). Готовий конфігураційний файл syslog-ng.conf наведено в Додатку А

Також було налаштовано IP tables (для обмеження доступу лише до певних портів) та підтримки iptables при завантаженні:

```
iptables -F INPUT DROP
iptables -F OUTPUT ACCEPT
sudo iptables -A INPUT -p tcp --dport 7777 -m conntrack --
ctstate NEW,ESTABLISHED -j ACCEPT
sudo iptables -t nat -A PREROUTING -p tcp --dport 445 -j REDIRECT --to-port 7777
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 3389 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 3393 -j ACCEPT
apt-get install iptables-persistent13.Iptables-save
```

В якості початкового інструменту для аналізу отриманих логів від Cowrie було вирішено обрати Splunk на базі дроплету сервісу Digital Ocean. Використання Splunk є зручним і популярним рішенням, що дозволило швидко і зручно аналізувати логи подій використовуючи візуалізацію результатів та зручні фільтри Splunk.

Для цього було створено дроплет (сутність) Digital Ocean і запусчено Splunk для обробки логів, зібраних Cowrie honeypot.

Було використано безкоштовну версію Splunk, для чого було зареєстровано обліковий запис Splunk. Далі було перейдено на сторінку завантажень і для віртуального сервера Ubuntu Digital Ocean VPS було запущена наступна команда, щоб завантажити останню версію Splunk:

```
wget -O splunk-9.0.7-3b814538f686-linux-4.2-amd64.deb
'https://www.splunk.com/bin/splunk/DownloadActivityServlet?ar-
chitecture=x86_64&platform=linux&version=9.0.7&prod-
uct=splunk&filename=splunk-9.0.7-3b814538f686-linux-4.2-
amd64.deb&wget=true'
```

Після завантаження Splunk було встановлено за допомогою команди

```
sudo dpkg -i splunk-9.0.7-3b814538f686-linux-4.2-amd64.deb
```

Наступною дією був запуск Splunk за допомогою команди

```
sudo /opt/splunk/bin/splunk start
```

Щоб легко обробляти логи Cowrie в Splunk, було використано поулярне рішення на основі додатку Splunk під назвою Tango Honeypot Intelligence [39]. Завантаження додатку і його встановлення в Splunk здійснено за допомогою функції "install app from file".

Після встановлення в Splunk було налаштовано Splunk Universal Forwarder в Cowrie Honeypot. Для цього на Raspberry Pi було виконано команду:

```
git clone https://github.com/aplura/Tango.git /tmp/tango; chmod
+x /tmp/tango/uf_only.sh
cd /tmp/tango/
./uf_only.sh
```

Це дозволило налаштувати і завантажити Splunk Universal Forwarder на Raspberry Pi OS. Далі, дотримуючись підказок було налаштовано «Форвардер» на взаємодію з Splunk. В якості порта отримувача Splunk, було використано порт 9997.

Вибір цього порта пояснюється тим, що за замовчуванням всі скрипти для Tango Honeypot Intelligence налаштовані на 9997.

В результаті попередніх налаштувань honeypot почав відправляти логи і далі потрібно було створити створити «слухача» в Splunk.

Для цього на дроплеті було перейдено в

`$splunk/etc/system/default/` на Splunk і відкрито файл `inputs.conf`

```
cd /opt/splunk/etc/system/default/
sudo nano inputs.conf
```

Всередині `inputs.conf` було додано наступні рядки

```
[splunktcp://9997]
disabled = 0
```

Після цього Splunk було перезапущено, щоб зміни вступили в силу, а потім відкрито додаток Tango Honeypot Intelligence. На цьому моменті я почав бачити дані, що надходять з Raspberry Pi від Cowrie honeypot'a.

Висновки до Розділу 3.

В Розділі 3 були виконані підготовчі розгортання і налаштування Raspberry Pi як платформи, що імітуватиме систему керування «Розумного будинку» на якій було встановлено ssh-honeypot Cowrie. Для роботи з логамі на цьому етапі дослідження було використано популярне рішення – пересилка і обробка логів в Splunk. Це дозволяє зручно і швидко переглядати великі об'єми інформації, що збирає Cowrie під час дій зловмисників. Розгляд і аналіз отриманої інформації буде наведено в наступному розділі.

РОЗДІЛ 4. АНАЛІЗ АТАК НА RASPBERRY PI SSH-HONEYPOT

У період з 16 листопада 2023 року по 7 грудня 2023 року було розгорнуто ssh-honeypot Cowrie на базі Raspberry Pi, що імітувала модуль керування «Розумним будинком». Фізично дослідницька система була розгорнута на базі Державного університету «Житомирська Політехніка» в одній з навчальних лабораторій з метою збору інформації про зловмисників, які взаємодіяли з Raspberry Pi, а також про те, як вони взаємодіяли з нею. На початковому етапі система була розгорнута для взаємодії з платформою Splunk, яка обробляла дані, записані системою, на дроплеті Splunk, розгорнутому за допомогою Digital Ocean. Для інтерпретації журналів Cowrie і надання розвідки та аналізу кіберзагроз і статистики використовувався додаток Tango Honeypot Intelligence. Загалом honeypot успішно приймав кілька з'єднань і сканувань на день, надаючи корисну інформацію про загрози і зразки шкідливого програмного забезпечення для аналізу.

За 22 дні, що honeypot був активним, з ним взаємодіяли 174 різні IP-адреси у вигляді успішних або неуспішних з'єднань. За весь час існування сайту було здійснено 452 спроб входу, з яких 24 були успішними, а 428 - невдалими. Це дорівнює приблизно 8 різним IP-адресам, які підключалися до honeypot'а в день, з 5 спробою входу в систему в день. Деяке пожвавлення активності зловмисників припадало на п'ятницю, суботу або неділю, що означає, що зловмисники були активні у вихідні дні. Днем найменшої активності, окрім дня запуску honeypot, був понеділок. У таблиці 4.1 наведено список найпопулярніших імен користувачів, які використовували зловмисники для підключення до ssh.

Серед найпопулярніших імен користувачів можна було побачити як стандартні імена, такі як "admin", "root", "pi" і "test", а також декілька неочікуваних ("admin0", "test1"). Ім'я користувача "pi" - ім'я користувача за замовчуванням для Raspberry Pi OS, а це означає, що системи на базі Raspberry Pi OS користуються популярністю у зловмисників в якості потенційної цілі.

Таблиця 4.1.

Імена користувача (username) які використовували зловмисники у спробах підключення до ssh-honeypot'а Cowrie

Username	Кількість спроб
admin	148
root	119
pi	64
test	36
ubuntu	31
user	25
debian	5
test1	2
admin0	1
raspi	1

В таблиці 4.2 наведено список найпопулярніших паролів, які намагалися використовувати зловмисники при підборі:

Таблиця 4.2.

Паролі які використовували зловмисники у спробах підключення до ssh-honeypot'а Cowrie

Password	Кількість спроб
	175
password	63
123456	50
admin	41
1234	24
pi	13
admin0	7
default	3
admin1	2
admin1234	1

Більшість паролів, що використали зловмисники - це ті, які можна було б очікувати, як стандартні для Raspberry Pi. Тобто варіант «без пароля», «password» і «123456», як традиційна трійка найпопулярніших паролів. З високою імовірністю, зловмисники використовували стандартний файл словника паролів для перебору паролів для входу в Raspberry Pi. У таблиці 4.3 наведено найпопулярніші комбінації імен користувачів та паролів:

Таблиця 4.3

Комбінації логінів і паролей, які використовували зловмисники у спробах підключення до ssh-honeypot'а Cowrie

Комбінація Username/Password	Кількість спроб
root/	119
admin/password	58
admin/123456	43
root/admin	36
admin/1234	21
pi/raspberry	12
admin/	8
ubuntu/admin	2
admin/admin1	2
admin/admin123	1

Додаток Tango Honeypot Intelligence [40] надає дані про місцезнаходження IP-адрес, які взаємодіють з honeypot. На цю інформацію не можна покладатися або використовувати для атрибуції, оскільки вона лише визначає місцезнаходження пристрою, який взаємодіяв з пристроєм. Зловмисник може взаємодіяти з honeypot через VPN або VPS, розташовані в іншій країні.

На рисунку 4.1 показано топ-5 країн, які сканували honeypot:

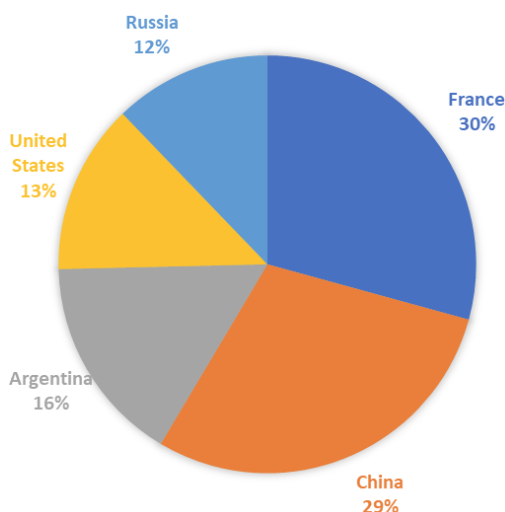


Рисунок 4.1 – Топ-5 країн з яких відбувалися сканування honeypot'а на основі інформації від додатку Tango Honeypot Intelligence

В таблиці 4.4 зазначені IP-адреси з яких було здійснено найбільша кількість спроб підключення до ssh-honeypot’у Cowrie на базі Raspberry Pi системи керування «Розумним будинком»

Таблиця 4.4

IP-адреси зловмисників, які намагались здійснити підключення до ssh-honeypot’а Cowrie

IP-адреса зловмисника	Країна	Кількість спроб
141.94.110.90	FR	34
61.177.173.21	CN	27
201.219.232.9	AR	19
173.82.51.214	US	14
179.60.147.157	RU	6
37.230.146.226	RU	3
129.205.102.242	NG	2
204.152.210.184	US	2
196.229.67.9	TN	2
154.13.1.114	DE	2

При перевірці IP-адрес зловмисників через відкритий сервіс аналізу підозрілих IP-адрес на платформі VirusTotal (www.virustotal.com) було виявлено, що ці адреси вже раніше були помічені в проведенні подібних атак (брутфорс SSH), а тому помічені в багатьох базах як джерело небезпеки:

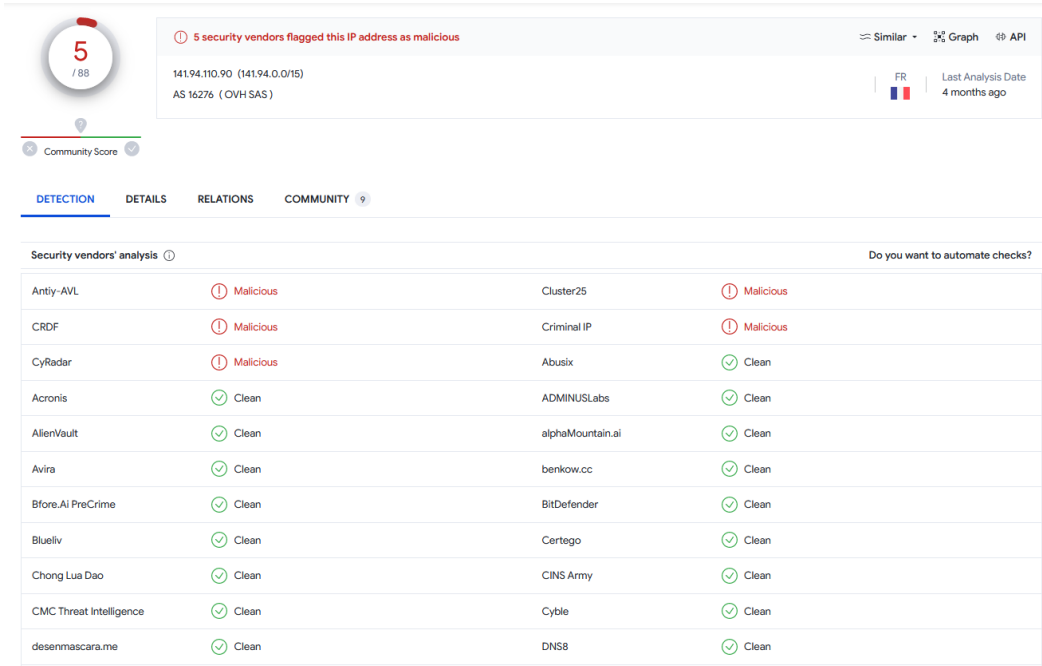


Рисунок 4.2 - Фрагмент репутаційної інформації про підозрілу IP-адресу з якої здійснювалися спроби підключення до ssh-honeypot’а з платформи VirusTotal

Провівши аналогічне дослідження IP-адрес зловмисників на платформі CriminalIP (www.criminalip.io) було також виявлено, що дані IP-адреси були неодноразово помічені в проведенні брутфорс-атак на ssh.

141.94.110.90

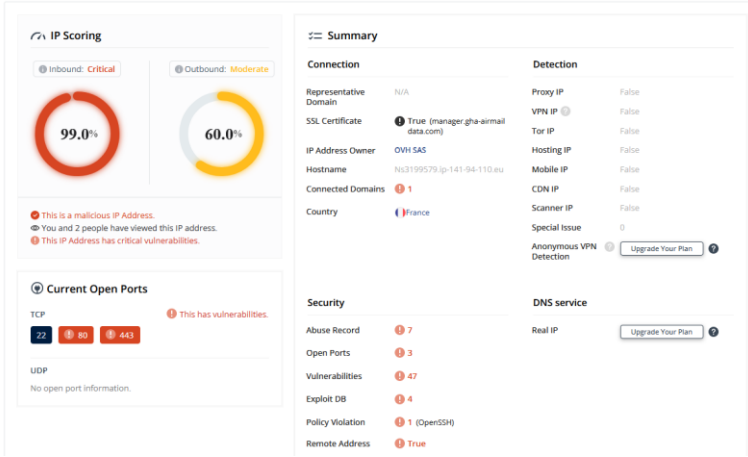


Рисунок 4.3. Фрагмент репутаційної інформації про підозрілу IP-адресу з якої здійснювалися спроби підключення до ssh-honeypot’а з платформи CrimnalIP

В базі AbuseIPDB (www.abuseipdb.com) IP-адреси зловмисників також зазначались як адреси з яких виконувались брутфорс-атаки на SSH

IP Abuse Reports for 141.94.110.90:

This IP address has been reported a total of 17,473 times from 799 distinct sources. 141.94.110.90 was first reported on October 26th 2022, and the most recent report was 1 month ago

Old Reports: The most recent abuse report for this IP address is from 1 month ago. It is possible that this IP is no longer involved in abusive activities.

Reporter	IoA Timestamp	Comment	Categories
✓ Anonymous	2023-11-08 17:38:29 (1 month ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-10-01 15:06:05 (2 months ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-08-11 19:22:37 (3 months ago)	\$f2bV_matches	Brute-Force SSH
🇫🇷🇯🇦	2023-07-12 14:00:37 (4 months ago)	ssh login attempt	Brute-Force SSH
✓ Anonymous	2023-07-04 09:54:31 (5 months ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-07-01 16:54:52 (5 months ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-05-28 11:29:49 (6 months ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-04-12 17:38:48 (7 months ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-03-06 22:53:02 (9 months ago)	\$f2bV_matches	Brute-Force SSH
✓ Anonymous	2023-03-03 17:45:34 (9 months ago)	\$f2bV_matches	Brute-Force SSH

Рисунок 4.4 - Фрагмент репутаційної інформації про підозрілу IP-адресу з якої здійснювалися спроби підключення до ssh-honeypot’а з платформи AbuseIPDB

Виконавши аналогічні дії з IP-адресами з яких були виконано найбільше спробу підключення до ssh honeypot'a на базі Raspberry Pi було виявлено, що абсолютна більшість з них була помічена в спробах брутфорсу SSH. Звіти про події за участю деяких адрес дають змогу побачити, що аналогічна діяльність проводиться з них протягом багатьох місяців (від 6 до 24 місяців).

Наступний блок корисної інформації від Cowrie було отримано проаналізувавши успішні сесії зломисників з ssh honeypot'у. В якості успішної «сесії» (session) прийнято вважати успішний вхід в систему і виконання команд в honeypot. З 24 успішних входів було лише 5 сесій з виконанням команд. У таблиці 4.5 наведено команди, що виконувалися в honeypot:

Таблиця 4.5

Команди, що вводились зломисниками під час успішних підключень до ssh-honeypot'a Cowrie

Команда, що вводилась	Кількість спроб
uname -a	4
free -m	2
ps x	2
uname	1
cat /proc/cpuinfo	1

Два зломисника, яким вдалося підключитися до honeypot виконали 'uname -a' як єдину команду, а потім завершували сеанс. Ще два зломисники виконували три перші команди: "uname -a", "free -m" і "ps x" за один сеанс. Імовірно, така поведінка відповідає роботі автоматизованого скрипту або бота, призначеного для брутфорсу в SSH і подальшого збору інформації про комп'ютер жертви шляхом отримання версії системи, інформації про пам'ять і запущені процеси. Автоматизація цього процесу дозволяє зломиснику переглянути інформацію, щоб визначити потенційні цілі для подальшого дослідження. Взаємодія людини і Tango Honeypot Intelligence надає дані про те, що оцінюється як взаємодія людини і що оцінюється як взаємодія бота. Ця оцінка виконується на основі часових міток взаємодії. Взає-

модії, де кількість введених команд здається підозрілою з огляду на тривалість сеансу, часто ідентифікуються як боти, а взаємодії, які тривають довше, будуть ідентифіковані як потенційно людські. Всі п'ять успішних сесій, що виконали злоумисники взаємодіяючи з honeypot мали тривалість менше 2 секунд. Тому, з високою імовірністю їх виконували боти або автоматизовані скрипти.

Висновок до Розділу 4

В розділі 4 були викладені результати дослідження, що проводилось на розгорнутій системі з IoT honeypot Cowrie, яка імітувала ssh системи керування «Розумним будинком». Дослідна система була розгорнута з 16 листопада 2023 року по 7 грудня 2023 року. За 22 дні, що honeypot був активним, з ним взаємодіяли 174 різні IP-адреси у вигляді успішних або неуспішних з'єднань. За весь час існування сайту було здійснено 452 спроб входу, з яких 24 були успішними, а 428 - невдалими.

В ході дослідження було проаналізовано IP-адреси з яких відбувались підключення злоумисників, комбінації логінів і паролей за допомогою яких вони намагались підключитись до honeypot, тривалість їх атак, а також декілька команд, що їх ввели злоумисники яким вдалося підібрати правильні логін і пароль.

Підсумовуючи цей етап дослідження слід зазначити, що розгортання, налаштування і отримана інформація про взаємодії Cowrie honeypot'а було ефективним для отримання цінних дослідницьких даних про кіберзагрози щодо злоумисників та їхньої діяльності. Зі зібраних даних було проаналізовано, що велику кількість спроб брутфорсу для проникнення в систему. Абсолютна більшість з них, імовірно була зроблена за допомогою автоматизованих скриптів і ботів. Переважна більшість таких взаємодій з honeypot оцінюється як боти, які ведуть розвідувальну діяльність, отримуючи інформацію про систему. З великою імовірністю ці боти потім передають отриману інформацію злоумисникам, щоб ті могли оцінити, чи є система потенційною цінністю. Подібні розвіддані про кіберзагрози, отримані honeypot, дозволяють фахівцям мережевої безпеки зміцнити свої мережі від виявлених загроз шляхом аналізу активності злоумисників, їх технік, тактик і процедур та векторів атак.

РОЗДІЛ 5. НАПИСАННЯ TELEGRAM-БОТУ ДЛЯ РЕАГУВАННЯ НА СПРОБИ ВЗЛОМУ COWRIE

5.1 Логування і відображення інформації про івенти в honeypot'і Cowrie

На даний момент одним з найбільш популярних способів логування і відображення інформації про інциденти в honeypot'і Cowrie є використання Elasticsearch Stack (ELK Stack). Тобто вискоєфективна комбінація платформ Elasticsearch/Logstash/Kibana. Elasticsearch використовується для швидкого пошуку і формування інформації про івенти. Logstash займається обробкою, зберіганням та індексуванням логів про події. Kibana використовується для візуального відображення отриманої інформації про можливі спроби вторгнення.

```
> May 10, 2021 @ 21:08:09.636 cloud_image_id: ami-82e2a5679226e293c cloud.region: sa-east-1 cloud.instance.id: i-0e4ebfc38769a8327 cloud.provider: aws cloud.account.id: 191982846051
cloud.availability_zone: sa-east-1c cloud.machine.type: t2.small ecs.version: 1.5.0 input.type: log @timestamp: May 10, 2021 @ 21:08:09.636 duration: 4.865
geosp.country_code3: CN geosp.location.lon: 113.727 geosp.location.lat: 34.773 geosp.longitude: 113.727 geosp.country_code2: CN geosp.ip: 27.128.244.13 geosp.latitude: 34.773
geosp.timezone: Asia/Shanghai geosp.continent_code: AS geosp.country_name: China @version: 1 host.hostname: ip-172-31-45-193 host.mac: 0a:0e:73:c3:a5:f8 host.os.name: Debian
GNU/Linux host.os.codename: buster host.os.family: debian host.os.platform: debian host.os.kernel: 4.19.0-16-cloud-amd64 host.os.version: 10 (buster) host.name: ip-172-31-45-193

> May 10, 2021 @ 21:08:09.635 cloud_image_id: ami-82e2a5679226e293c cloud.region: sa-east-1 cloud.instance.id: i-0e4ebfc38769a8327 cloud.provider: aws cloud.account.id: 191982846051
cloud.availability_zone: sa-east-1c cloud.machine.type: t2.small ecs.version: 1.5.0 input.type: log @timestamp: May 10, 2021 @ 21:08:09.635 duration: 95.373
geosp.country_code3: CN geosp.location.lon: 113.727 geosp.location.lat: 34.773 geosp.longitude: 113.727 geosp.country_code2: CN geosp.ip: 27.128.244.13 geosp.latitude: 34.773
geosp.timezone: Asia/Shanghai geosp.continent_code: AS geosp.country_name: China @version: 1 host.hostname: ip-172-31-45-193 host.mac: 0a:0e:73:c3:a5:f8 host.os.name: Debian
GNU/Linux host.os.codename: buster host.os.family: debian host.os.platform: debian host.os.kernel: 4.19.0-16-cloud-amd64 host.os.version: 10 (buster) host.name: ip-172-31-45-193

> May 10, 2021 @ 21:08:09.635 cloud.provider: aws cloud_image_id: ami-82e2a5679226e293c cloud.instance.id: i-0e4ebfc38769a8327 cloud.region: sa-east-1 cloud.account.id: 191982846051
cloud.availability_zone: sa-east-1c cloud.machine.type: t2.small ecs.version: 1.5.0 input.type: log @timestamp: May 10, 2021 @ 21:08:09.635 duration: 14.766
geosp.country_code3: PA geosp.location.lon: -80 geosp.location.lat: 9 geosp.longitude: -80 geosp.country_code2: PA geosp.ip: 45.227.255.206 geosp.latitude: 9
geosp.timezone: America/Panama geosp.continent_code: NA geosp.country_name: Panama @version: 1 host.hostname: ip-172-31-45-193 host.mac: 0a:0e:73:c3:a5:f8 host.os.name: Debian
GNU/Linux host.os.codename: buster host.os.family: debian host.os.platform: debian host.os.kernel: 4.19.0-16-cloud-amd64 host.os.version: 10 (buster) host.name: ip-172-31-45-193

> May 10, 2021 @ 21:08:09.634 cloud.region: sa-east-1 cloud.provider: aws cloud.instance.id: i-0e4ebfc38769a8327 cloud_image_id: ami-82e2a5679226e293c cloud.account.id: 191982846051
cloud.availability_zone: sa-east-1c cloud.machine.type: t2.small ecs.version: 1.5.0 input.type: log @timestamp: May 10, 2021 @ 21:08:09.634 duration: 118.097
geosp.country_code3: PA geosp.location.lon: -80 geosp.location.lat: 9 geosp.longitude: -80 geosp.country_code2: PA geosp.ip: 45.227.255.207 geosp.latitude: 9
geosp.timezone: America/Panama geosp.continent_code: NA geosp.country_name: Panama host.hostname: ip-172-31-45-193 host.mac: 0a:0e:73:c3:a5:f8 host.os.name: Debian GNU/Linux
host.os.codename: buster host.os.family: debian host.os.platform: debian host.os.kernel: 4.19.0-16-cloud-amd64 host.os.version: 10 (buster) host.name: ip-172-31-45-193
```

Рисунок 5.1 - Приклад фрагменту логу подій від honeypot'у Cowrie відображеного засобами Kibana

Даний підхід є доречним при використанні великої низки різноманітних honeypot'ів (як Cowrie так і інших honeypot'ів, що підтримують ELK Stack) в умовах реальної мережі підприємства чи дослідницької лабораторії, бо дозволяє зручно реалізовувати графічні інтерфейси для відображення великого об'єму інформації, що поступає в режимі реального часу. Дашборди, фільтри і готові плагіни дозволяють оперативно реагувати на події і зберігати величезний об'єм інформації про дії зловмисників.

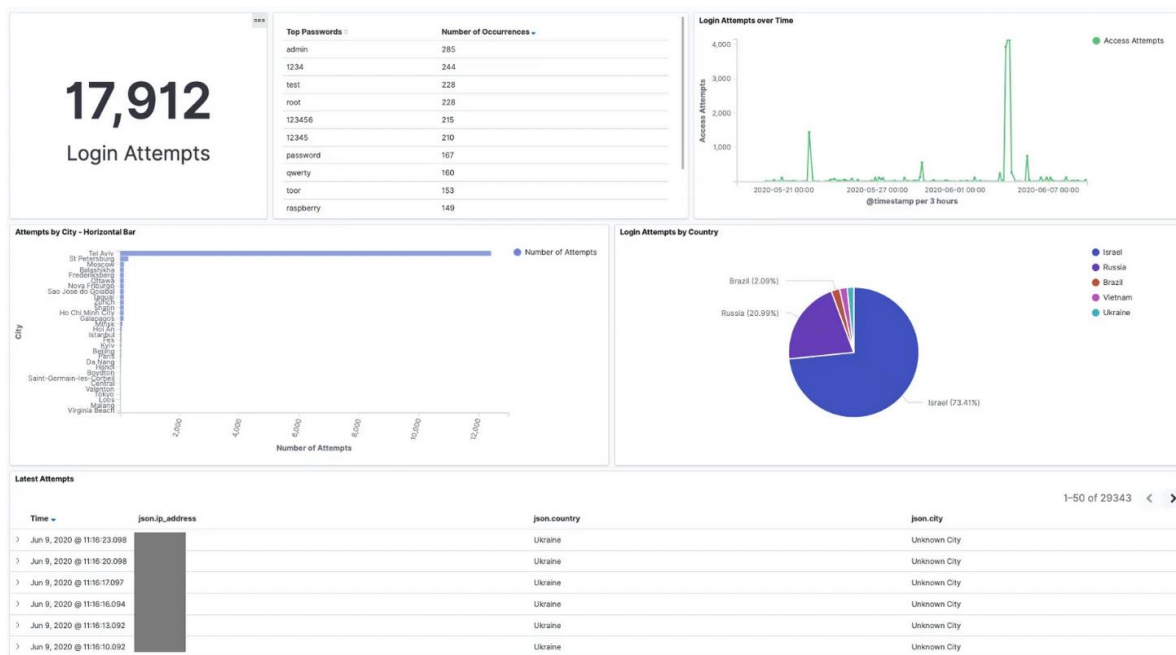


Рисунок 5.2 - Приклад дашборду логу подій від honeypot'у Cowrie відображеного засобами Kibana

В той же час використання ELK Stack є дещо надлишковим з точки зору співвідношення часу на його налаштування і деплой до кількості інформації, що оброблятиметься на ньому від одного єдиного honeypot'а, що виступатиме в ролі «першого рубежа» виявлення початку атаки на платформу керування «Розумного будинку». В якості альтернативи, що дозволить максимально повноцінно і зручно використати весь об'єм отриманої від Cowrie інформації про можливі інциденти, а також дасть змогу оперативно інформувати про їх виникнення мною пропонується рішення на базі telegram-боту через який і буде відобуватися інформування користувача системи «Розумний будинок».

Однією з дуже зручних особливостей Cowrie є можливість зручного налаштування формату видачі інформації про події для їх подальшої обробки сторонніми програмами чи для пересилки в інші сервіси. Зручним і популярним рішенням є використання формату JSON для пересилки інформації про спроби атак. JSON - структурований формат обміну даними, який легко читати для людей і легко обробляти для програм.

Всі події та дані, зібрані Honeypot Cowrie, можуть бути збережені у вигляді JSON-файлів, де кожен запис містить різні атрибути та параметри, пов'язані із спробою атаки, включаючи IP-адресу нападника, використані команди, дані аутентифікації і т. д.

JSON, як і будь який інший формат збереження даних, може бути налаштований за допомогою конфігураційних файлів Honeypot Cowrie, і він може змінюватися залежно від версії програми або конфігурації, використовуваної в конкретному випадку. На рисунку 5.3 наведено приклад простого конфігураційного файлу для Honeypot Cowrie.

```
{
  "honeypot": {
    "hostname": "Smart_Home",
    "listen_host": "0.0.0.0",
    "listen_port": 22,
    "log_file": "/path/to/cowrie/log/cowrie.json",
    "log_level": "INFO",
    "json_logging": true
  },
  "ssh": {
    "enabled": true,
    "listen_port": 2222,
    "auth_none": false,
    "auth_test_mode": false
  },
  "output": {
    "json": true
  }
}
```

Рисунок 5.3 - Приклад файлу конфігурації для збереження інформації про атаки в форматі JSON

Вищезазначений конфігураційний файл містить наступні параметри:

- "hostname": Ім'я хосту, яке буде використовуватися для honeypot сервера.
- "listen_host": IP-адреса, на якій слухається сервер.
- "listen_port": Порт, на якому слухається сервер SSH.
- "log_file": Шлях до файлу журналу, в якому зберігатиметься інформація про події.

- "log_level": Рівень логування (може бути "DEBUG", "INFO", "WARNING", "ERROR", "CRITICAL").
- "json_logging": Включення запису подій у форматі JSON.
- "enabled": Увімкнення або вимкнення прослуховування SSH.
- "listen_port": Порт, на якому слухається сервер SSH для Cowrie.
- "auth_none": Включення або вимкнення можливості аутентифікації без пароля.
- "auth_test_mode": Включення або вимкнення режиму тестування аутентифікації.
- "json": Включення виводу результатів у форматі JSON.

Формат JSON є зручним для пересилки інформації про атаки на Cowrie до telegram-боту з метою їх подальшого внесення в базу даних подій, відображення інформації про подію і використанні в додаткових сервісах.

В якості інформації, що пересилатиметься на telegram-бот від honeypot було обрано наступний формат повідомлення:

```
{
  "timestamp": "2023-11-10T12:30:45",
  "event_type": "ssh_bruteforce",
  "attacker_ip": "141.94.110.90",
  "attempt_number": 3,
  "credentials": {
    "username": "root",
    "password": "password123"
  },
  "commands_attempted": "ls -la"
}
```

Рисунок 5.4 - Коротка інформація в форматі JSON про атаку на ssh-honeypot, що надсилатиметься в телеграм-бот для обробки.

Цей формат JSON буде використаний для створення об'єкта, який буде переданий телеграм-боту для обробки і відправлення повідомлення користувачу «Розумного будинку» у випадку атаки на honeypot. Він містить наступні поля:

- timestamp - Час і дата події у форматі ISO 8601.

- event_type - Тип події, в даному випадку "ssh_bruteforce".
- attacker_ip - IP-адреса зловмисника.
- attempt_number - Номер спроби проникнення з цієї IP-адреси.
- credentials - Інформація про використані логін та пароль.
- username - логін, що використано для атаки.
- password - пароль, що використовується для атаки.
- commands_attempted - список команд, які зловмисник намагався виконати після успішного підключення до системи (у випадку успішного вторгнення).

5.2 Розробка telegram-боту для відображення інформації про атаки на honeypot і реагування на них.

Використання telegram-ботів є простим і доступним способом отримання інформації про різноманітні типи подій на мережевій інфраструктурі. Відносно простий Telegram Bot API дозволяє швидко розробити і запустити боти різноманітного призначення та функціоналу.

В якості основного функціоналу telegram-боту, що був розроблений в ході даної роботи є відображення короткої інформації про спроби атак («бріфу»), що здійснюються зловмисниками на honeypot.

Другою важливою функцією є перевірка інформації про IP-адреси, використані зловмисниками в ході атаки, в відкритих безкоштовних сервісах для репутаційної перевірки IP-адрес. В якості основної платформи для перевірки IP-адрес зловмисників було використано VirusTotal з його Public API, що дає змогу здійснити до 500 запитів на день та містить величезну базу даних по різноманітним IP-адресам, що помічені в протизаконній діяльності.

Третьою, додатковою функцією, стала функція пересилання IP-адреси зловмисника на мережеве обладнання (в якості прикладу буде обрано мережеве обладнання від компанії Mikrotik під керуванням операційної системи RouterOS) для подальшого модифікування правил брандмауера. Мережеве обладнання від компанії

Mikrotik є дуже популярним в Україні та за її межами, а можливості, що надає її операційна система RouterOS дають змогу максимально гнучко і зручно взаємодіяти з пристроями з використанням чудово документованого Mikrotik API.

Розглядаючи процес розробки telegram-боту для оперативного інформування про атаки на honeypot необхідно визначити на основі якого фреймворку буде розроблятися бот. Також слід зазначити, що якості мови на якій було написано бот обрано python, так як це дозволило використати велику кількість готових бібліотек для низки проміжних задач і значно скоротити як час розробки так і об'єм коду. Обираючи між різними фреймворками для написання telegram-ботів на мові python абсолютним лідером за функціоналом та зручністю є aiogram, що робить процес обрання фреймворку практично безальтернативним. В підтвердження останньої тези буде наведена коротка інформація про цей фреймворк.

Особливості фреймворка aiogram:

- Асинхронність (async/await): Одна з ключових особливостей aiogram - це використання асинхронного програмування з використанням async/await. Це дозволяє створювати швидкі і ефективні боти, зокрема, якщо бот повинен взаємодіяти з багатьма користувачами одночасно.
- Простота використання: Aiogram має простий і зрозумілий інтерфейс, що полегшує розробку. Він включає в себе високорівневий API для зручності, але також надає можливість робити HTTP-запити безпосередньо, якщо потрібно.
- Розширюваність: Aiogram дозволяє легко розширювати функціональність бота за допомогою middleware (проміжних обробників), які можуть виконувати певні дії при обробці кожного повідомлення.
- Підтримка багатьох API Telegram: Aiogram підтримує як HTTP API, так і Bot API, що робить його більш гнучким і придатним для різних випадків використання.

Перевагами aiogram є :

- Асинхронність: Дана особливість дозволяє створювати високоефективні боти, особливо в умовах великої кількості одночасних запитів.

- Зручний інтерфейс: Aiogram має досить зрозумілий інтерфейс, що полегшує розробку ботів, навіть для початківців.
- Активна спільнота: Фреймворк активно підтримується і розвивається, що гарантує наявність нових функцій і виправлення помилок.

Проте aiogram все ж таки має деякі недоліки:

- Менша кількість вбудованих функцій: У порівнянні з іншими фреймворками, aiogram може мати менше вбудованих функцій. Проте це може бути компенсовано розширенням функціональності за допомогою middleware.
- Вимоги до асинхронного програмування: Для коректної роботи фреймворка потрібно знати або вивчити асинхронне програмування, що може бути викликом для розробників, які раніше не працювали з асинхронним кодом.

Основні чинники, що вплинули на вибір aiogram в якості основи для боту:

- Асинхронність: якщо бот потребуватиме високої продуктивності та ефективності, aiogram може бути відмінним вибором завдяки підтримці асинхронного програмування.
- Простота використання: можливість швидко отримати зручний інтерфейс і швидкість розробки.
- Розширюваність: якщо в майбутньому потрібно буде розширювати функціональність бота, aiogram надасть зручний механізм middleware для цієї цілі.

В якості ілюстрації принципів роботи фреймворка aiogram на Рисунку 5.5 наведено приклад telegram-бота зробленого на цьому фреймворку. Цей код створює простого бота, який відповідає на команду /start і повторює текст, який введено користувачем. В прикладі ілюструється використання асинхронного програмування через ключове слово `async` та `await`.

```
import logging
from aiogram import Bot, Dispatcher, types

# Встановлення рівня логування для отримання інформації про виконання програми
logging.basicConfig(level=logging.INFO)
```

```

# 'TOKEN' - фактичний токен бота, що отримується в @BotFather
API_TOKEN = 'TOKEN'

# Ініціалізація бота та диспетчера
bot = Bot(token=API_TOKEN)
dp = Dispatcher(bot)

# Обробник команди /start
@dp.message_handler(commands=['start'])
async def send_welcome(message: types.Message):
    await message.answer("Привіт! Я бот. Я можу відповідати на ко-
манду /start.")

# Обробник текстових повідомлень
@dp.message_handler()
async def echo(message: types.Message):
    await message.answer(f"Ви написали: {message.text}")

if __name__ == '__main__':
    from aiogram import executor
    executor.start_polling(dp, skip_updates=True)

```

Рисунок 5.5 - Приклад використання фреймворку aiogram для створення telegram-боту.

Код на вищенаведеному прикладі працює наступним чином:

1. Імпорт необхідних модулів.
2. Ініціалізація бота та диспетчера.
3. Створення обробника для команди /start, який відправляє привітання.
4. Створення обробника для всіх інших текстових повідомлень, який відправляє повідомлення з ехо (повторює текст користувача).
5. Запуск бота за допомогою executor.start_polling.

Важливим функціоналом telegram-боту, що інформуватиме користувача про можливі спроби атак на ssh на honeypot Cowrie є додаткове надсилання інформації з логів Cowrie (IP-адреса зловмисника, що виконав атаку) на сервіс Virus Total шляхом використання відкритого публічного API (Public API). Це дає змогу отримати у відповідь репутаційну інформацію з відкритих баз даних Virus Total API, що по-

стійно оновлюється і містить величезну кількість коментарів різноманітних користувачів сервісу стосовно IP-адрес зломисників і їх дій під час атак на конкретні ресурси.

Доступ до Virus Total Public API надається кожному зареєстрованому користувачу шляхом надання індивідуального API-ключа, що використовуватиметься в запитах до Virus Total Public API. На Рисунку 5.6 показано інтерфейс відповідного розділу особистого кабінету для роботи з API сервісу. Рамками виділено приватний API Key і обмеження безкоштовної версії Virus Total Public API.

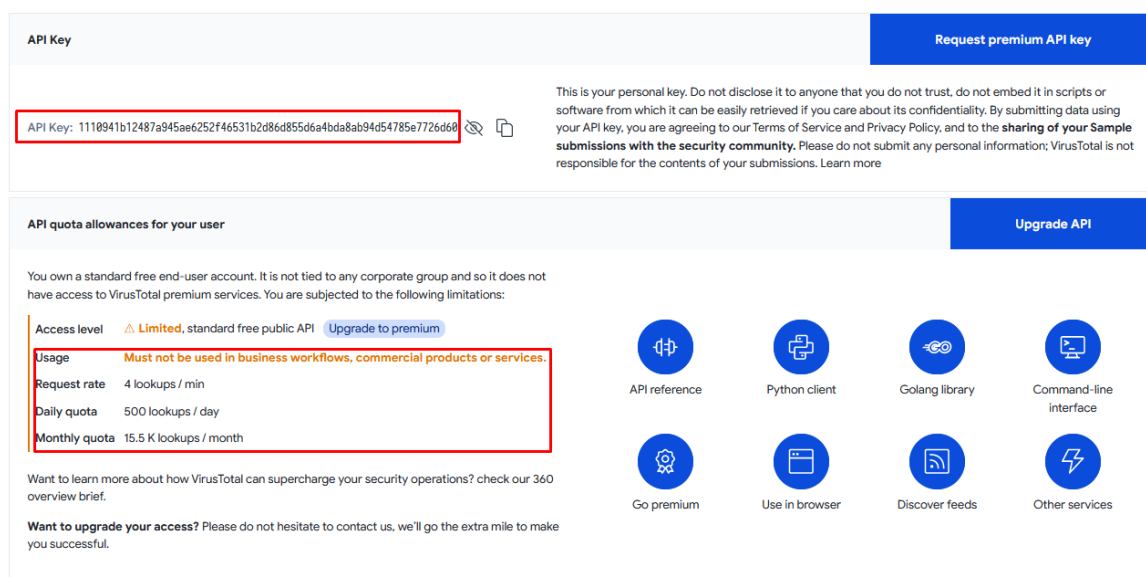


Рисунок 5.6 - Зовнішній вигляд розділу особистого кабінету для роботи з API сервісу Virus Total.

Virus Total Public API є безкоштовним відкритим сервісом, проте, як і будь який безкоштовний сервіс він має ряд обмежень в порівнянні з платним (Premium API). До таких обмежень відноситься вимога не використовувати Public API в комерційних продуктах чи сервісах, швидкість звернень до сервісу – не більше 4 запитів за хвилину, не більше 500 запитів в день і, відповідно, не більше 15500 запитів в місяць. Аналізуючи кількість атак, що зазнавав Cowrie honeypot на попередньому етапі дослідження, особливо з врахуванням кількості унікальних IP-адрес з яких виконувались атаки, можна вважати, що безкоштовної версії Virus Total Public API є достатньо для проведення подальших досліджень.

```
import requests
```

```
def get_virustotal_info(api_key, ip_address):
    url = f'https://www.virustotal.com/api/v3/ip_addresses/{ip_address}'
```

```

headers = {
    'x-apikey': api_key
}

response = requests.get(url, headers=headers)

if response.status_code == 200:
    data = response.json()

    print(f"IP-адреса: {ip_address}")
    print(f"Країна:
{data['data']['attributes']['last_analysis_stats']['country']}")
    print(f"Час останнього повідомлення:
{data['data']['attributes']['last_analysis_stats']['last_analysis_date']}")
    print("Інформація про зловмисну активність:")

    for result in data['data']['attributes']['last_analysis_results']:
        print(f"Деталі: {result['result']}")
        print(f"Детальний опис: {result['result_description']}")
        print(f"Час аналізу: {result['last_analysis_date']}")
        print("-----")

else:
    print(f"Помилка при виклику API: {response.status_code}")

if __name__ == "__main__":
    api_key = 'PRIVAT_API_KEY'
    target_ip = 'TARGET_IP_ADDRESS'
    get_virustotal_info(api_key, target_ip)

```

Рисунок 5.7 - Приклад використання Virus Total Public API для отримання інформації про IP-адресу.

В якості прикладу використання Virus Total Public API на Рисунку 5.7 наводиться код програми на мові python, що здійснює запит до сервісу з метою визначення того чи містяться в базах даних сервісу Virus Total інформація про IP-адресу з якої відбувались протизаконні дії, отримати інформацію стосовно характеру протизаконної активності (тип атак) в якій помічена ця IP-адреса, а також отримати інформацію про час останніх повідомлень про атаки в яких була помічена ця IP-адреса.

Останнім етапом розробки telegram-боту є додавання функції надсилання IP-адреси з якої відбувалася спроба взлому ssh в honeypot'і до правил брендмауєру Mikrotik-роутеру під керуванням RouterOS.

Мережеве обладнання від компанії Mikrotik є дуже популярним в Україні та за її межами, а можливості, що надає її операційна система RouterOS дають змогу максимально гнучко і зручно взаємодіяти з пристроями з використанням чудово документованого Mikrotik API.

Для взаємодії з API мережевого обладнання MikroTik зазвичай використовують бібліотеку, таку як librouteros для Python.

На рисунку 5.8 наведено приклад використання MikroTik API для модифікації правил firewall та створення логування.

В цій програмі на мові python шляхом звертання до Mikrotik API пристрою буде модифікуватися правила firewall на пристрої шляхом заборони підключення з певної IP-адреси як до самого пристрою так і до мережі за цим пристроєм. Додатково створюється окреме правило для логування спроб підключення чи пересилки пакетів через пристрій з IP-адреси, що вноситься в заборонні правила брандмауера.

```
from librouteros import connect

mikrotik_ip = "192.168.88.1" # IP-адреса MikroTik-пристрою
mikrotik_username = "largeadminlogin"
mikrotik_password = "largecomplexpassword111"

connection = connect(username=mikrotik_username, password=mikrotik_password,
host=mikrotik_ip)

firewall_rule = {
    "action": "drop",
    "chain": "input",
    "src-address": "141.94.110.90", # IP-адреса, що буде блокуватися
    "comment": "The attacker's IP address is blocked"
}

connection.path("/ip/firewall/filter").add(firewall_rule)

logging_rule = {
    "action": "log",
    "chain": "input",
    "src-address": "141.94.110.90",
    "comment": "Attempt to connect from an attacker's IP address"
}

connection.path("/ip/firewall/filter").add(logging_rule)
connection.close()
```

Рисунок 5.8 - Приклад використання Mikrotik API додавання IP-адреси зловмисника в блокуюче правило firewall'а Mikrotik-пристрою.

Для реалізації всіх вищезазначених функцій було створено telegram-бот Smart_Home_Honey_bot. Його повний код на мові python наведено в Додатку Б.

Приклад роботи боту продемонстровано на Рисунках 5.9-5.10.

Алгоритм роботи телеграм-боту наступний:

1. Після виникнення події (наприклад, спроба ssh-bruteforce атаки на Cowrie) Cowrie надсилає в Smart_Home_Honey_bot JSON з короткою інформацією з характеристикою події (“SSH Attack Brief”). Ця інформація містить час початку атаки (Time), тип події (“Event_type”. Наприклад, “ssh_bruteforce”), IP-адресу з якої було виконано спробу підключення до Cowrie (“IP”), інформація про те скільки спроб атак вже було виконано з відповідної IP-адреси (“Attempt”), логін і пароль які були використані зловмисником в спробі підключитися до ssh в ході атаки (“Credentials”), список команд виконаних зловмисником в терміналі Cowrie у випадку його успішного підключення до ssh (“Command Attempted”).
2. За командою “/start” бот розпочинає роботу і виводить на екран перше з неопрацьованих повідомлень про атаку на honeypot. Повідомлення містить як інформацію отриману від Cowrie в форматі JSON так і результат запиту до VirusTotal Public API стосовно репутаційної інформації про IP-адреси з якої відбувалася спроба підключення до honeypot. Ця репутаційна інформація містить дані про країну в якій зареєстрована IP-адреса (“Country”), а також види протизаконної активності в якій було помічено цю IP-адресу (“Malicious Activity”) якщо ця IP-адреса вже значиться в базах даних VirusTotal. Приклад повідомлення показано на Рисунку 5.9

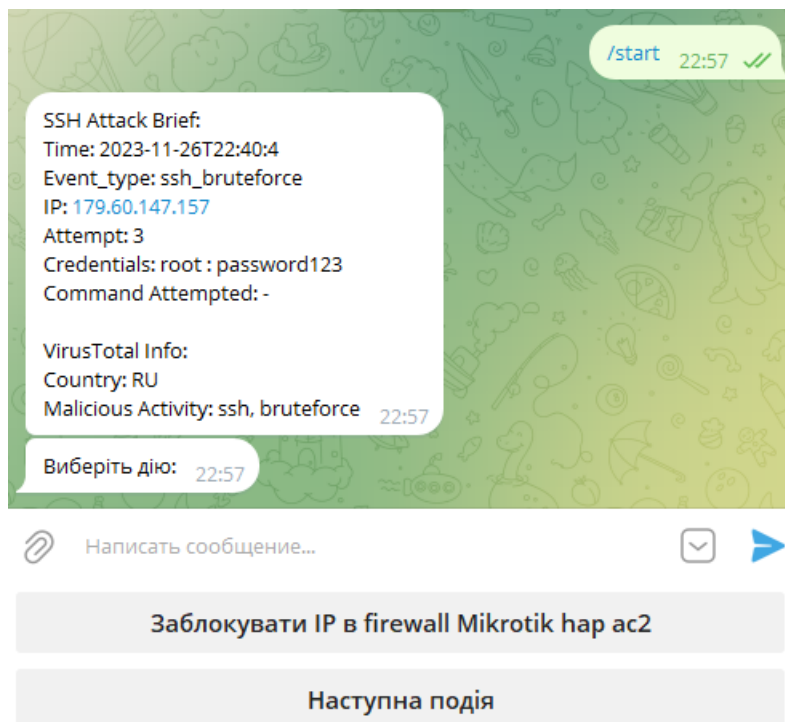


Рисунок 5.9 - Приклад повідомлення Smart_Home_Honey_bot про спробу атаки на honeypot.

3. Після виведення повідомлення про подію Smart_Home_Honey_bot виводить повідомлення «Виберіть дію» пропонуючи користувачеві скористатись відповідними кнопками в меню боту.
4. У випадку активації кнопки меню «Заблокувати IP в firewall Mikrotik» бот виконає запит до API Mikrotik шляхом додавання правила, що блокуватиме всі спроби підключення з відповідної IP адреси як до самого пристрою Mikrotik так і до мережевих ресурсів за ним ("chain": "input", "action": "drop"). Крім цього до правил логування пристрою було додано окреме правило для логування спроб підключення чи пересилки пакетів з відповідної IP-адреси з відповідним коментарем ("Attempting to connect from a blocked IP address"). Після цієї дії виводиться повідомлення «IP-адреса додана до правил блокування на Mikrotik». Приклад виконання активації кнопки меню «Заблокувати IP в firewall Mikrotik» показано на Рисунку 5.10.

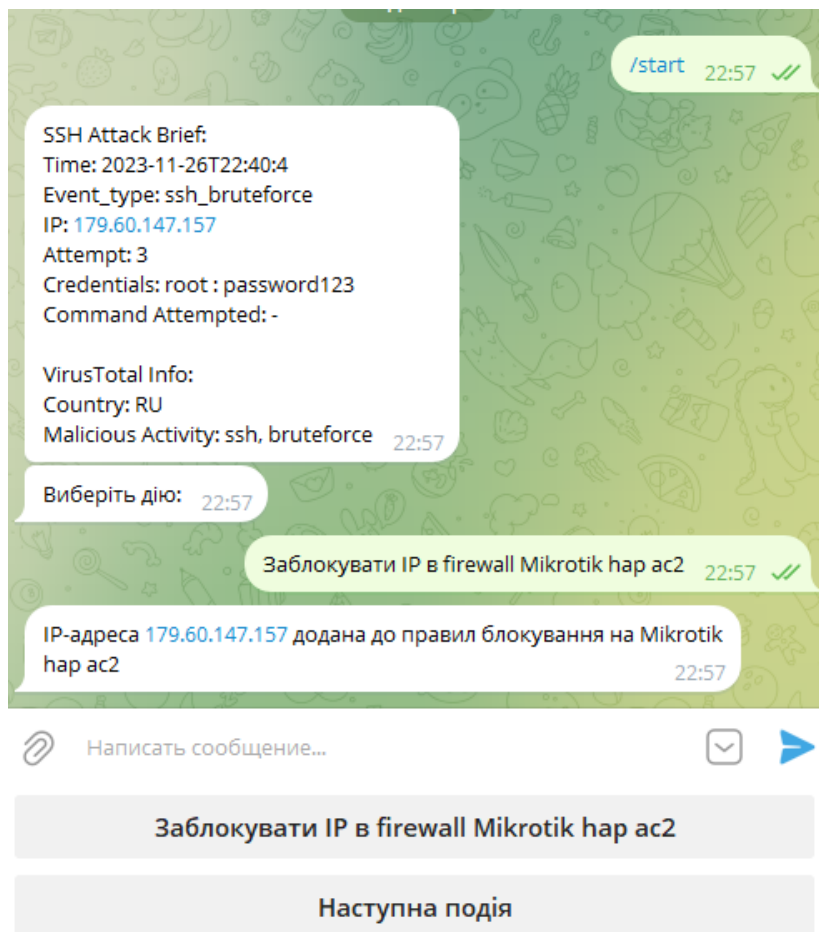


Рисунок 5.10 - Приклад виконання активації кнопки меню «Заблокувати IP в firewall Mikrotik».

5. У випадку активації кнопки меню «Наступна подія» бот завантажить наступне повідомлення (“Brief”) від honeypot та VirusTotal Public API. Якщо це була остання зареєстрована спроба атаки на honeypot, то буде виведено повідомлення «Події відсутні». Приклад повідомлення на Рисунку 5.11.

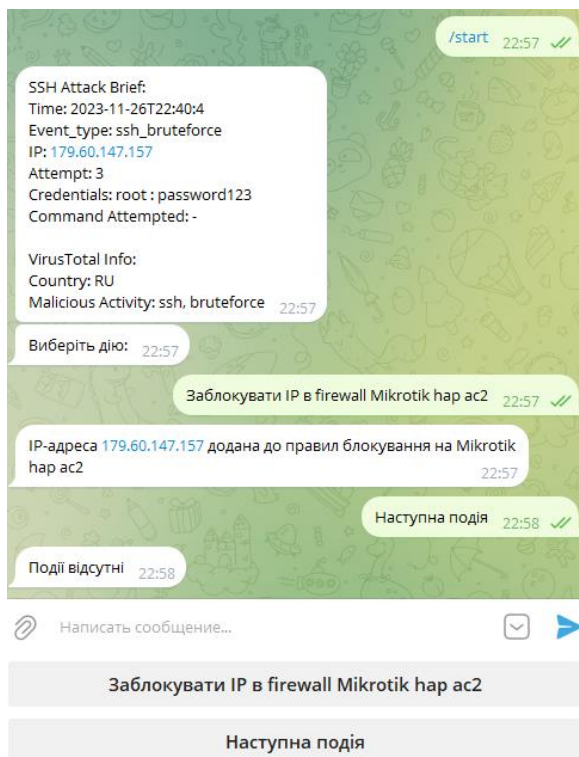


Рисунок 5.11 - Приклад виконання активації кнопки меню «Наступна подія».

Висновок до Розділу 5

В розділі 5 було розглянуто декілька підходів до логування і відображення інформації про події на Cowrie honeypot. Традиційний метод відображення інформації з використанням ELK Stack є зручним для великого об'єму інформації, проте цей спосіб вимагає кращих навичок від користувача для налаштування відображення та автоматизації реагування на події.

В якості альтернативи було запропоновано і розроблено telegram-бот Smart_Home_Honey_bot, що має зручний інтерфейс для відображення інформації про події та реагування на них. Бот є інструментом для оперативного інформування користувача про спроби атак зловмисників на honeypot та для швидкого реагування на атаки шляхом посилення безпеки мережевої інфраструктури «Розумного будинку».

ВИСНОВКИ

За останні роки кількість кібератак має стійку тенденцію до збільшення, а еволюція методів проникнення і інструментарів кіберзловмисників постійно прискорюється. Поряд з загрозами традиційними атаками на державні установи, комерційні підприємства та дані користувачів існують стійкі загальносвітові тенденції до зростання кількості атак на пристрої IoT та різноманітних вбудованих систем. Цей напрямок кібератак досі залишався не настільки загрозливим для України в короткостроковій перспективі як атаки на критичну інфраструктуру. Однак, зважаючи на все ширше застосування інтернету речей в медицині, промисловості і комунальній інфраструктурі, він є доволі перспективним для кіберзлочинців з точки зору співвідношення зусиль на реалізацію таких атак до хаосу і збитків спричиненими цими атаками.

Оскільки зазвичай ці пристрої мають обмежену обчислювальну потужність і часто не мають вбудованих засобів захисту, вони особливо вразливі для хакерських атак, які намагаються отримати доступ до мережі. Оскільки Інтернет речей швидко поширюється на наше особисте та професійне життя, потенційна поверхня атак стає дедалі більшою. Відновлення і відбудова інфраструктури, промислових підприємств, муніципальних і транспортних систем України як в теперішній час так і після війни вимагатиме значного переосмислення підходів до їх побудови для більш ефективного використання ресурсів. Базуючись на кращих підходах останніх десятиліть і загальносвітових тенденцій для побудови нових і вдосконалення вже існуючих об'єктів інфраструктури будуть використовуватись значна кількість промислових і муніципальних рішень інтернету речей. А це, в свою чергу, створюватиме додаткові потенційні площини для атак кіберзлочинців. Тому пошук рішень по виявленню, аналізу і протидії цим загрозам стає вкрай актуальним.

Оскільки зазвичай ці пристрої мають обмежену обчислювальну потужність і часто не мають вбудованих засобів захисту, вони особливо вразливі для хакерських атак, які намагаються отримати доступ до мережі. Оскільки Інтернет речей швидко

поширюється на наше особисте та професійне життя, потенційна поверхня атак стає дедалі більшою.

Одним з важливих напрямків посилення кібербезпеки приватних мереж користувачів є пошук оперативних дієвих методів виявлення спроб атак на ці традиційно менш захищені мережі. Розповсюдження Інтернету речей і систем «Розумний будинок» зокрема є зручною ціллю для кіберзлочинців, що використовують так слабкозахищені ресурси в своїй протизаконній активності. Одним зі шляхом виявлення атак на подібні мережі є побудова систем honeypot'ів, що ставатимуть першою ціллю атаки і даватимуть змогу проінформувати користувачів як про сам факт атаки на мережу «Розумного будинку» так і збиратимуть інформацію про дії зловмисника для його подальшого притягнення до відповідальності.

Метою цього проекту було підвищення безпеки системи «Розумний будинок» шляхом побудови системи IoT honeypot для виявлення на атаки зловмисників та подальшого аналізу їх технік, тактик і процедур.

Для досягнення поставлених цілей розроблено і виконано наступні завдання:

1. Визначити основні кібербезпекові загрози для України в 2023 році та тенденції до їх змін стосовно атак на пристрої Інтернету речей.
2. Проаналізувати основні тенденції до використання honeypot систем для покращення безпеки IoT систем.
3. Проаналізувати і обрати IoT honeypot або низку honeypot систем для використання їх в системі безпеки «Розумного дому».
4. Розгорнути, налаштувати і перевірити працездатність IoT honeypot системи для посилення безпеки системи «Розумного дому».
5. Проаналізувати результати виявлених атак на IoT honeypot систему безпеки «Розумного дому».

В ході дослідження, що проводилось на розгорнутій дослідній системі з IoT honeypot Cowrie, яка імітувала ssh системи керування «Розумним будинком» з 16 листопада 2023 року по 7 грудня 2023 року було отримано низку результатів, що дає змогу вважати запропонований підхід до виявлення атак на мережу «Розумного будинку» ефективною. За 22 дні, що honeypot був активним, з ним взаємодіяли 174

різні IP-адреси у вигляді успішних або неуспішних з'єднань. За весь час існування сайту було здійснено 452 спроб входу, з яких 24 були успішними, а 428 - невдалими.

В ході дослідження було проаналізовано IP-адреси з яких відбувались підключення злоумисників, комбінації логінів і паролей за допомогою яких вони намагались підключитись до honeypot, тривалість їх атак, а також декілька команд, що їх ввели злоумисники яким вдалося підібрати правильні логін і пароль.

В додаткового інструменту для відображення атак на IoT honeypot запропоновано і розроблено telegram-бот Smart_Home_Honey_bot, що має зручний інтерфейс для відображення інформації про події та реагування на них. Бот є інструментом для оперативного інформування користувача про спроби атак злоумисників на honeypot та для швидкого реагування на атаки шляхом посилення безпеки мережевої інфраструктури «Розумного будинку».

Таким чином розгортання, налаштування і отримана інформація про взаємодії Cowrie honeypot'а було ефективним методом для отримання цінних дослідницьких даних про кіберзагрози щодо злоумисників та їхньої діяльності. Зі зібраних даних було проаналізовано, що велику кількість спроб брутфорсу для проникнення в систему. Абсолютна більшість з них, імовірно була зроблена за допомогою автоматизованих скриптів і ботів. Переважна більшість таких взаємодій з honeypot оцінюється як боти, які ведуть розвідувальну діяльність, отримуючи інформацію про систему. З великою імовірністю ці боти потім передають отриману інформацію злоумисникам, щоб ті могли оцінити, чи є система потенційною цінністю. Подібні розвіддані про кіберзагрози, отримані honeypot, дозволяють фахівцям мережевої безпеки зміцнити свої мережі від виявлених загроз шляхом аналізу активності злоумисників, їх технік, тактик і процедур та векторів атак.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Cybersecurity in 2022 – A Fresh Look at Some Very Alarming Stats [Електронний ресурс]// Cybersecurity in 2022 – A Fresh Look at Some Very Alarming Stats. - Режим доступу: <https://www.forbes.com/sites/chuckbrooks/2022/01/21/cybersecurity-in-2022-a-fresh-look-at-some-very-alarming-stats/?sh=6a4b6a4f6b61> (дата звернення: 12.12.2023). – Назва з екрану.
2. Microsoft Digital Defense Report 2022 [Електронний ресурс]// Microsoft Digital Defense Report 2022. - Режим доступу: <https://www.microsoft.com/en-us/security/business/microsoft-digital-defense-report-2022> (дата звернення: 12.12.2023). – Назва з екрану.
3. Talos Year in Review 2022 [Електронний ресурс]// Talos Year in Review 2022. - Режим доступу: <https://blog.talosintelligence.com/talos-year-in-review-2022> (дата звернення: 12.12.2023). – Назва з екрану.
4. SCPC Report for Q1 2023 [Електронний ресурс]// SCPC Report for Q1 2023. - Режим доступу: <https://scpc.gov.ua/en/articles/306> (дата звернення: 12.12.2023). – Назва з екрану.
5. Enemy hackers are attacking Ukrainian critical infrastructure: enhancement of defense requires continuous effort [Електронний ресурс]// Enemy hackers are attacking Ukrainian critical infrastructure: enhancement of defense requires continuous effort . - Режим доступу: <https://cip.gov.ua/en/news/vorozhi-khakeri-atakuyut-kritichnu-infrastrukturu-ukrayini-pracyuvati-nad-posilennyam-zakhistu-treba-postiino> (дата звернення: 12.12.2023). – Назва з екрану.

6. The number of cyberattacks against commercial sector has tripled y-t-d: Statistics [Електронний ресурс]// The number of cyberattacks against commercial sector has tripled y-t-d: Statistics - Режим доступу: <https://cip.gov.ua/en/news/vid-pochatku-roku-zroslo-kilkist-kiberatak-na-komerциinii-sektor-statistika> (дата звернення: 12.12.2023). – Назва з екрану.

7. Attacks against IT companies and specialized software developers as a threat to critical infrastructure [Електронний ресурс]// Attacks against IT companies and specialized software developers as a threat to critical infrastructure - Режим доступу: <https://cip.gov.ua/en/news/ataki-na-it-kompaniyi-ta-specializovanikh-rozrobnikiv-pz-yak-zagroza-kritichnii-infrastrukturi> (дата звернення: 12.12.2023). – Назва з екрану.

8. State of IoT 2023: Number of connected IoT devices growing 16% to 16.7 billion globally [Електронний ресурс]// State of IoT 2023: Number of connected IoT devices growing 16% to 16.7 billion globally - Режим доступу: <https://iot-analytics.com/number-connected-iot-devices/> (дата звернення: 12.12.2023). – Назва з екрану.

9. F. Ebberts. A Large-Scale Analysis of IoT Firmware Version Distribution in the Wild. IEEE Transactions on Software Engineering, vol. 49, no. 02, pp. 816-830, 2023. DOI: 10.1109/TSE.2022.3163969

10. E. Hutchins, M. Cloppert, A. Rohan. Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. Leading Issues in Information Warfare & Security Research. 1. 2011

11. Pooja Kumari, Ankit Kumar Jain, A comprehensive study of DDoS attacks over IoT network and their countermeasures, Computers & Security, Volume 127, 2023. <https://doi.org/10.1016/j.cose.2023.103096>.

12. D. Fraunholz, M. Zimmermann, H.D. Schotten. Towards Deployment Strategies for Deception Systems // *Advances in Science, Technology and Engineering Systems Journal*. 2017. Vol. 2.

13. Fraunholz D., Schotten H.D. Defending Web Servers with Feints, Distraction and Obfuscation // *Proceedings of the International Conference on Computing, Networking and Communications (ICNC, Maui, USA, 5–8 March 2018)*. Piscataway, NJ: IEEE, 2018. DOI:10.1109/ICCNC.2018.8390365

14. D. Reti, D. Fraunholz, J. Zemitis, D. Schneider, H. D. Schotten. Deep Down the Rabbit Hole: On References in Networks of Decoy Elements// *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, Dublin, Ireland, 2020, pp. 1-11, DOI: 10.1109/CyberSecurity49315.2020.9138850.

15. D. Fraunholz, D. Krohmer, H. D. Schotten and C. Nogueira. Introducing Falcom: A Multifunctional High-Interaction Honeypot Framework for Industrial and Embedded Applications// *2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, Glasgow, UK, 2018, pp. 1-8, doi: 10.1109/Cyber-SecPODS.2018.8560675.

16. D. Fraunholz, D. Krohmer, S. D. Anton, H.D. Schotten. Investigation of cyber crime conducted by abusing weak or default passwords with a medium interaction honeypot// *2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security)*, London, UK, 2017, pp. 1-7, DOI: 10.1109/Cyber-SecPODS.2017.8074855.

17. A. Acien, A. Nieto, G. Fernandez, and J. Lopez. A comprehensive methodology for deploying IoT honeypots// *15th International Conference on Trust, Privacy and Security in Digital Business (TrustBus 2018)* vol. LNCS 11033, pp. 229243, 2018. http://doi.org/https://doi.org/10.1007/978-3-319-98385-1_16

18. H. Šemić and S. Mrdovic. IoT honeypot: A multi-component solution for handling manual and Mirai-based attacks// 2017 25th Telecommunication Forum (TELFOR), Belgrade, Serbia, 2017, pp. 1-4, doi: 10.1109/TELFOR.2017.8249458.
19. M. A. Hakim, H. Aksu, A. S. Uluagac, K. Akkaya. U-PoT: A Honeypot Framework for UPnP-Based IoT Devices// 2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC), Orlando, FL, USA, 2018, pp. 1-8, doi: 10.1109/PCCC.2018.8711321.
20. S. Ravji and M. Ali. Integrated Intrusion Detection and Prevention System with Honeypot in Cloud Computing// 2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE), 2018, pp. 95-100, doi: 10.1109/iCCECOME.2018.8658593.
21. A. Pashaei, M. Akbari, M. Lighvan, A. Charmin, Early Intrusion Detection System using honeypot for industrial control networks, Results in Engineering, Volume 16, 2022, 100576, ISSN 2590-1230, <https://doi.org/10.1016/j.rineng.2022.100576>.
22. D. Zielinski, H. Kholidy (2022). An Analysis of Honeypots and their Impact as a Cyber Deception Tactic. 10.48550/arXiv.2301.00045.
23. Checkpot. Honeypot Checker [Електронний ресурс]// Checkpot. Honeypot Checker - Режим доступу: <https://github.com/honeynet/checkpot> - (дата звернення: 12.12.2023). – Назва з екрану.
24. Send-Safe Honeypot Hunter [Електронний ресурс]// Send-Safe Honeypot Hunter - Режим доступу: <https://send-safe-honeypot-hunter.apponic.com/> - (дата звернення: 12.12.2023). – Назва з екрану.

25. Honeypot or not. [Електронний ресурс]// Honeypot or not. - Режим доступу: <https://honeyscore.shodan.io/> - (дата звернення: 12.12.2023). – Назва з екрану.

26. Detect-kippo-cowrie [Електронний ресурс]// Detect-kippo-cowrie - Режим доступу: https://github.com/blazeinfosec/detect-kippo-cowrie__ - (дата звернення: 12.12.2023). – Назва з екрану.

27. Cheng Huang, Jiaxuan Han, Xing Zhang, and Jiayong Liu (2019). Automatic Identification of Honeypot Server Using Machine Learning Techniques// Security and Communication Networks, Volume 2019, Article ID 2627608. <https://doi.org/10.1155/2019/2627608>.

28. M. Tsikerdekis, S. Zeadally, A. Schlesener, N. Sklavos. (2018). Approaches for Preventing Honeypot Detection and Compromise. 10.1109/GIIS.2018.8635603.

29. W. Fan, Z. Du and D. Fernández. Taxonomy of honeynet solutions// 2015 SAI Intelligent Systems Conference (IntelliSys), 2015, pp. 1002-1009, doi: 10.1109/IntelliSys.2015.7361266.

30. IoT Security Statistics 2023 [Електронний ресурс]// IoT Security Statistics 2023 - Режим доступу: <https://webinarcare.com/best-iot-security-software/iot-security-statistics/#3> - (дата звернення: 12.12.2023). – Назва з екрану.

31. Y. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, C. Rossow. (2016). IoT-POT: A novel honeypot for revealing current IoT threats. Journal of Information Processing. 24. 522-533. 10.2197/ipsjjip.24.522.

32. J. Franco, A. Aris, B. Canberk and A. S. Uluagac. A Survey of Honeypots and Honey-nets for Internet of Things, Industrial Internet of Things, and Cyber-Physical Systems//IEEE Communications Surveys & Tutorials, vol. 23, no. 4, pp. 2351-2383, Fourth-quarter 2021, doi: 10.1109/COMST.2021.3106669.

33. Dionea honeypot [Електронний ресурс]// Dionea honeypot - Режим доступу: <https://github.com/DinoTools/dionaea> - (дата звернення: 12.12.2023). – Назва з екрану.

34. Intelligent-IoT-Honeypot [Електронний ресурс]// Intelligent-IoT-Honeypot - Режим доступу: <https://github.com/as2d3/Intelligent-IoT-Honeypot> - (дата звернення: 12.12.2023). – Назва з екрану.

35. Conpot honeypot [Електронний ресурс]// Conpot honeypot - Режим доступу: <https://github.com/mushorg/conpot> - (дата звернення: 12.12.2023). – Назва з екрану.

36. Kippo honeypot [Електронний ресурс]// Kippo honeypot - Режим доступу: <https://github.com/desaster/kippo> - (дата звернення: 12.12.2023). – Назва з екрану.

37. Blacknet 2 honeypot [Електронний ресурс]// Blacknet 2 honeypot - Режим доступу: <https://github.com/morian/blacknet> - (дата звернення: 12.12.2023). – Назва з екрану.

38. Kojoney2 honeypot [Електронний ресурс]// Kojoney2 honeypot - Режим доступу: <https://github.com/madirish/kojoney2> - (дата звернення: 12.12.2023). – Назва з екрану.

39. Cowrie honeypot [Електронний ресурс]// Cowrie honeypot - Режим доступу: <https://github.com/cowrie/cowrie> - (дата звернення: 12.12.2023). – Назва з екрану.

40. Tango Honeypot Intelligence [Електронний ресурс]// Tango Honeypot Intelligence
- Режим доступу: <https://splunkbase.splunk.com/app/2666> - (дата звернення:
12.12.2023). – Назва з екрану.

ДОДАТКИ

Вміст файлу syslog-ng.conf

```

@version: 3.19
@include "scl.conf"

# Syslog-ng configuration file, compatible with default Debian sys-
logd
# installation.

# First, set some global options.
options { chain_hostnames(off); flush_lines(0); use_dns(no);
use_fqdn(no);
         dns_cache(no); owner("root"); group("adm"); perm(0640);
         stats_freq(0); bad_hostname("^gconfd$");
};

#####
# Sources
#####
# This is the default behavior of sysklogd package
# Logs may come from unix stream, but not from another machine.
#
source s_src {
    system();
    internal();
};

source honeypot {
    file("/var/log/rdpy/app.log" program-override("rdpy") flags(no-
parse));
    #file("/var/log/cowrie/audit.log" flags(no-parse));
    file("/var/log/smb/app.log" program-override("smb") flags(no-
parse));
    file("/var/log/lighttpd/access.log" program-override("lighttpd")
flags(no-parse));
    #file("/var/log/supervisor/supervisord.log" flags(no-parse));
};

# If you wish to get logs from remote machine you should uncomment
# this and comment the above source line.
#
#source s_net { tcp(ip(127.0.0.1) port(1000)); };

#####
# Destinations
#####
# First some standard logfile
#
destination d_auth { file("/var/log/auth.log"); };
destination d_cron { file("/var/log/cron.log"); };
destination d_daemon { file("/var/log/daemon.log"); };

```

```

destination d_kern { file("/var/log/kern.log"); };
destination d_lpr { file("/var/log/lpr.log"); };
destination d_mail { file("/var/log/mail.log"); };
destination d_syslog { file("/var/log/syslog"); };
destination d_user { file("/var/log/user.log"); };
destination d_uucp { file("/var/log/uucp.log"); };

# This files are the log come from the mail subsystem.
#
destination d_mailinfo { file("/var/log/mail.info"); };
destination d_mailwarn { file("/var/log/mail.warn"); };
destination d_mailerr { file("/var/log/mail.err"); };

# Logging for INN news system
#
destination d_newscrit { file("/var/log/news/news.crit"); };
destination d_newserr { file("/var/log/news/news.err"); };
destination d_newsnotice { file("/var/log/news/news.notice"); };

# Some 'catch-all' logfiles.
#
destination d_debug { file("/var/log/debug"); };
destination d_error { file("/var/log/error"); };
destination d_messages { file("/var/log/messages"); };

# The root's console.
#
destination d_console { usertty("root"); };

# Virtual console.
#
destination d_console_all { file(`/tty10`); };

# The named pipe /dev/xconsole is for the nsole' utility. To use
it,
# you must invoke nsole' with the -file' option:
#
#    $ xconsole -file /dev/xconsole [...]
#
destination d_xconsole { pipe("/dev/xconsole"); };

# Send the messages to an other host
#
destination d_net { tcp("192.168.x.x" port(9042)
log_fifo_size(1000)); };

# Debian only
destination d_ppp { file("/var/log/ppp.log"); };

#####
# Filters
#####
# Here's come the filter options. With this rules, we can set which

```

```
# message go where.
```

```
filter f_dbg { level(debug); };
filter f_info { level(info); };
filter f_notice { level(notice); };
filter f_warn { level(warn); };
filter f_err { level(err); };
filter f_crit { level(crit .. emerg); };

filter f_debug { level(debug) and not facility(auth, authpriv, news,
mail); };
filter f_error { level(err .. emerg) ; };
filter f_messages { level(info,notice,warn) and
                    not facility(auth,authpriv,cron,dae-
mon,mail,news); };

filter f_auth { facility(auth, authpriv) and not filter(f_debug); };
filter f_cron { facility(cron) and not filter(f_debug); };
filter f_daemon { facility(daemon) and not filter(f_debug); };
filter f_kern { facility(kern) and not filter(f_debug); };
filter f_lpr { facility(lpr) and not filter(f_debug); };
filter f_local { facility(local0, local1, local3, local4, local5,
                    local6, local7) and not filter(f_debug); };
filter f_mail { facility(mail) and not filter(f_debug); };
filter f_news { facility(news) and not filter(f_debug); };
filter f_syslog3 { not facility(auth, authpriv, mail) and not fil-
ter(f_debug); };
filter f_user { facility(user) and not filter(f_debug); };
filter f_uucp { facility(uucp) and not filter(f_debug); };

filter f_cnews { level(notice, err, crit) and facility(news); };
filter f_cother { level(debug, info, notice, warn) or facility(dae-
mon, mail); };

filter f_ppp { facility(local2) and not filter(f_debug); };
filter f_console { level(warn .. emerg); };
```

```
#####
```

```
# Log paths
```

```
#####
```

```
log { source(s_src); filter(f_auth); destination(d_auth); };
log { source(s_src); filter(f_cron); destination(d_cron); };
log { source(s_src); filter(f_daemon); destination(d_daemon); };
log { source(s_src); filter(f_kern); destination(d_kern); };
log { source(s_src); filter(f_lpr); destination(d_lpr); };
log { source(s_src); filter(f_syslog3); destination(d_syslog); };
log { source(s_src); filter(f_user); destination(d_user); };
log { source(s_src); filter(f_uucp); destination(d_uucp); };

log { source(s_src); filter(f_mail); destination(d_mail); };
#log { source(s_src); filter(f_mail); filter(f_info); destina-
tion(d_mailinfo); };
```

```

#log { source(s_src); filter(f_mail); filter(f_warn); destina-
tion(d_mailwarn); };
#log { source(s_src); filter(f_mail); filter(f_err); destina-
tion(d_mailerr); };

log { source(s_src); filter(f_news); filter(f_crit); destina-
tion(d_newscrit); };
log { source(s_src); filter(f_news); filter(f_err); destina-
tion(d_newser); };
log { source(s_src); filter(f_news); filter(f_notice); destina-
tion(d_newsnotice); };
#log { source(s_src); filter(f_cnews); destination(d_console_all);
};
#log { source(s_src); filter(f_cother); destination(d_console_all);
};

#log { source(s_src); filter(f_ppp); destination(d_ppp); };

log { source(s_src); filter(f_debug); destination(d_debug); };
log { source(s_src); filter(f_error); destination(d_error); };
log { source(s_src); filter(f_messages); destination(d_messages); };

log { source(s_src); filter(f_console); destination(d_console_all);
      destination(d_xconsole); };
log { source(s_src); filter(f_crit); destination(d_console); };
log { source(honeypot); destination(d_syslog); };
# All messages send to a remote site
#
log { source(s_src); destination(d_net); };
log { source(honeypot); destination(d_net); };
###
# Include all config files in /etc/syslog-ng/conf.d/
###
@include "/etc/syslog-ng/conf.d/*.conf"

```

Код боту Smart_Home_Honey_bot

```

import json
import requests
from aiogram import Bot, Dispatcher, types
from aiogram.types import ParseMode
from aiogram.types import InlineKeyboardMarkup, InlineKeyboardButton

TOKEN = '6875251593:AAEun_nBjwXtna-Nkfe3Inw-pPdMXfpTIWo'

bot = Bot(token=TOKEN)
dp = Dispatcher(bot)

# Дані для Mikrotik API
MIKROTIK_API_URL = 'http://192.168.88.2/api'
MIKROTIK_API_USERNAME = 'admin101m1kr0t1k'
MIKROTIK_API_PASSWORD = 'm1kr0t1kpa88w0rd'

def get_virustotal_info(ip_address):
    # Ключ для доступу до VirusTotal API
    virustotal_api_key = '1110941b12487a945ae6252f46531b2d86d855d6a4bda8ab94d54785e7726d60'
    url = f'https://www.virustotal.com/vtapi/v2/ip-address/report?apikey={virustotal_api_key}&ip={ip_address}'
    response = requests.get(url)
    if response.status_code == 200:
        result = response.json()
        return result
    return None

def modify_firewall(ip_address):
    # Формуємо дані для додавання правила в firewall Mikrotik
    rule_data = {
        'chain': 'input',
        'action': 'drop',
        'src-address': ip_address,
        'comment': f'Blocked by Telegram Bot'
    }

    # Створюємо URL для додавання правила в firewall Mikrotik
    api_url = f'{MIKROTIK_API_URL}/ip/firewall/filter/add'

    # Відправляємо POST-запит з даними авторизації та правилом
    response = requests.post(
        api_url,
        auth=(MIKROTIK_API_USERNAME, MIKROTIK_API_PASSWORD),
        data=rule_data
    )

    # Виводимо повідомлення в телеграм-бот
    if response.status_code == 200:
        await bot.send_message(message.chat.id, f"Firewall modified successfully. IP {ip_address} is blocked.")

```

```

else:
    await bot.send_message(message.chat.id, f"Error modifying firewall. Status code: {response.status_code}")

@dp.message_handler(commands=['start'])
async def start(message: types.Message):
    # При отриманні команди /start обробляємо отриманий JSON та надсилаємо бріфи

    attack_info = json.loads(json_data)

    # Використовуємо VirusTotal API для отримання інформації
    vt_info = get_virustotal_info(attack_info['attacker_ip'])

    # Формуємо бріф
    brief_text = f"<b>SSH Attack Brief:</b>\n"
    brief_text += f"Time: {attack_info['timestamp']}\n"
    brief_text += f"IP: {attack_info['attacker_ip']}\n"
    brief_text += f"Attempt #{attack_info['attempt_number']}\n"
    brief_text += f"Credentials: {attack_info['credentials']['username']}: {attack_info['credentials']['password']}\n"
    brief_text += f"Command Attempted: {attack_info['command_attempted']}\n"

    # Додаємо інформацію від VirusTotal, якщо є
    if vt_info:
        brief_text += "\n<b>VirusTotal Info:</b>\n"
        brief_text += f"Country: {vt_info.get('country', 'N/A')}\n"
        brief_text += f"Malicious Activity: {vt_info.get('as_owner', 'N/A')}\n"

    # Надсилаємо користувачеві
    keyboard = InlineKeyboardMarkup()
    modify_firewall_button = InlineKeyboardButton("Модифікувати firewall",
callback_data='modify_firewall')
    keyboard.add(modify_firewall_button)

    await message.answer(brief_text, parse_mode=ParseMode.HTML, reply_markup=keyboard)

@dp.callback_query_handler(lambda c: c.data == 'modify_firewall')
async def modify_firewall_callback(callback_query: types.CallbackQuery):
    # Викликаємо функцію модифікації firewall при натисканні на кнопку
    # Витягуємо IP-адресу з attack_info
    ip_address = attack_info['attacker_ip']
    modify_firewall(ip_address, callback_query.message)

    await bot.answer_callback_query(callback_query.id, text="Firewall modified successfully")

if __name__ == '__main__':
    from aiogram import executor

    executor.start_polling(dp, skip_updates=True)

```