ІНФОКОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ ТА ЕЛЕКТРОННА ІНЖЕНЕРІЯ
INFORMATION AND COMMUNICATION TECHNOLOGIES,
ELECTRONIC ENGINEERING

№ 2 (2), 2021 https://doi.org/10.23939/ictee2021.02.032

РОЗРОБЛЕННЯ СИСТЕМИ АНАЛІЗУ СЕСІЙ З ПРИСТРОЯМИ IOT ДЛЯ БОРОТЬБИ ІЗ БОТНЕТАМИ

С. Тукало, О. Шпур, О. Костів

Національний університет "Львівська політехніка", вул. С. Бандери, 12, Львів, 79013, Україна

Відповідальний за рукопис: С. Тукало (e-mail: sviatoslav.tukalo.mtr.2020@lpnu.ua).

(Подано 1 грудня 2021)

Розроблено систему аналізу сесій із пристроями IoT для боротьби із ботнетами і, як наслідок, – захисту пристроїв мережі Інтернету речей від проникнення зловмисних мереж ботів. Для її реалізації запропоновано власний ботнет на основі протоколу SSH. Задля забезпечення високої надійності та децентралізованості ботнет здійснює керування через окремий сервер баз даних, в якому міститься інформація про стан ботів, а також загальна інформація про кожного з них. Запропонована система аналізу сесій реалізована за принципом Honeynet мереж, але по суті є гібридною, оскільки використовує модель автономних агентів, модель моніторингу мережі та модель виявлення вторгнень на основі поведінки. Командний сервер може викрадати файли із зараженого бота, виконувати будь-які операції від імені адміністратора, а також вражати розумні пристрої. Для дослідження використано смарт-годинник, який працює за допомогою Bluetooth LE. Як результат створено власну систему захисту від ботнетів, яка дає змогу аналізувати хост та виявляти основні ознаки наявності цього хоста в мережі ботів. Це дозволяє оперативно зреагувати та почати протидіяти такому зараженню. Система дає змогу отримати дані про встановлені активні з'єднання SSH, команди, які віддалено запускаються на цьому хості, а також автоматично заблокувати встановлені з'єднання та не допустити проникнення нових. У результаті тестування запропонованої системи здійснено атаку на пристрій IoT та заблоковано зловмисника, що підтверджує ефективність розробки.

Ключові слова: *IoT; бот; ботмережа; SSH; SSH-botnet; Honeynet.*

УДК 004.45

1. Вступ

Сьогоднішні темпи розвитку сфери інформаційних технологій набули чималих обертів, вона розвивається дуже стрімко. Стає очевидним той факт, що сфера Інтернету речей є однією з галузей інформаційних технологій, які зростають найстрімкіше. Кількість пристроїв, підключених до цієї мережі, вже у 2025 р., за пронозами вчених, досягне 21,5 млрд [1]. Така популярність зумовлена низкою істотних переваг: порівняно низька собівартість продукту, популярність серед населення, поліпшення функціонування підприємств та заводів, покращення маркетингових рейтингів. Оскільки Інтернет речей зачіпає такі важливі галузі нашого життя, постає питання безпеки персональних даних та IoT пристроїв загалом.

На цей момент у світі не існує єдиного стандарту розроблення IoT пристроїв. Це означає, що кожен виробник може скористатися будь-якими стандартами безпеки або ж взагалі їх не використовувати. Дуже часто це призводить до того, що IoT пристрій стає жертвою кіберзлочинців. Для мінімізації ризиків, пов'язаних із доступом до пристроїв IoT та підміною даних на них,

необхідний постійний моніторинг сесій, як TCP, так і UDP, які створюються у разі їх підключення. Причин атаки на ґаджети доволі багато, проте основні з них – залучення пристрою до бот-мережі або крадіжка персональних даних. Отже, розроблення системи аналізу сесій з пристроями IoT для боротьби із ботнетами і, як наслідок, захисту пристроїв мережі Інтернету речей від проникнення зловмисних мереж ботів є актуальним завданням.

2. Аналіз відомих методів деактивації ботнетів та способи захисту від інфікування

Першим кроком у боротьбі з ботнетами є їх виявлення. Для цього використовують такі техніки:

- Honeypots та Honeynet – це середовище, яке створено спеціально вразливим до кібератак для зараження його ботом. Принцип полягає в тому, що після інфікування Honypot'а ботом системі вдасться внести дані про бота до бази даних антивірусних програм та продовжити аналіз всієї ботнет-мережі [2].
- Система виявлення ботнету на підставі його "підпису" – такі системи працюють на основі відомих підписів та ознак шкідливого ПЗ. Ці системи працюють у мережі, аналізуючи весь прохідний трафік [3].
- Система виявлення ботнету на основі аномалій. Ця методика виявляє ботнет на основі параметрів мережевого трафіку, а саме: затримки мережі, підозріло великих обсягів трафіку, незвичної поведінки системи та трафіку на невідомих портах [4].
- Метод виявлення ботнету на основі DNS – найвідоміший та найпопулярніший метод виявлення, ґрунтується на моніторингу трафіку, оскільки боти використовують DNS для отримання команд від C&C сервера. Враховуючи нерегулярність таких запитів, ботнети легко помітити [5].
- Оцінка потоку (Flux Score) – це метрика, основана на кількох параметрах: кількість відображень IP-доменів у всіх пошуках DNS, кількість записів серверів імен в одному пошуку одного домена [6].
- Рекурсивна DNS методика – контролює трафік DNS та зберігає всю інформацію в централізований колектор даних [7].
- Пакетний метод – це відстеження різних властивостей доменного імені для виявлення будь-якого несанкціонованого домена, що генерується за допомогою алгоритмів доменного потоку. Метод відстежує такі параметри: властивості WHOIS, тривалість TTL, а також географічне положення.
- Графік відмов DNS – оскільки DNS сервери дуже надійні та рідко виходять з ладу за звичних умов роботи, цей метод є одним із найефективніших.

Зловмисні мережі визначають за набором параметрів, серед яких: короткий TTL, швидкість зміни набору вирішених IP-адрес, що повертаються з кожного запиту, велика кількість вирішених IP-адрес та розв'язані IP-адреси у багатьох різних мережах. У разі використання ботнету кількість відмов у DNS стрімко зростатиме, що дасть змогу застосувати одну із відомих практик для боротьби із ними [8, 9, 16, 17].

Загальні практики для боротьби з ботнетом можна поділити на дві групи: технічні та нетехнічні. Технічні практики для боротьби з ботнетами часто використовують такі ж методи, як і самі ботнети. До них належать:

- DDoS атаки на C&C вузли.
- Виведення з ладу C&C вузлів.
- Захоплення DNS імен, які використовуються C&C.
- Блокування IP адрес [18, 10].

Методи боротьби з ботнетами типу Peer-2-Peer

Оскільки реалізація захисту централізованих ботнетів (Client-Server) достатньо проста, розробники шкідливого ПЗ вимушені були перейти до інших архітектурних рішень. Насамперед ботнети з архітектурою P2P створюють певні перешкоди для їх детектування в мережі:

- Протокол динамічного налаштування хоста DHCP – динамічний перерозподіл тієї самої IP адреси може призвести до появи двох або більше заражених машин, що детектуватимуться як одна.
- Проксі-сервер – схоже на проблему, спричинену DHCP. Боти отримуватимуть доступ до інтернету через прозорий або анонімний проксі-сервер і детектуватимуться як єдиний бот.
- Зашифрована комунікація – якщо бот використовує зашифрований трафік, єдиним методом для його дослідження буде reverse engineering, а ключ для дешифрування повинен міститися на клієнтській частині бота.
- NAT – механізм, який дає змогу змінювати IP-адреси у заголовку пакета, тим самим відображаючи у мережі кілька ботів як один.

Використання цих методів перешкоджання відслідковуванню ботнетів приводить до доволі тривалого їх функціонування. Тривалість функціонування ботів залежно від їхнього типу наведено на рис. 1.

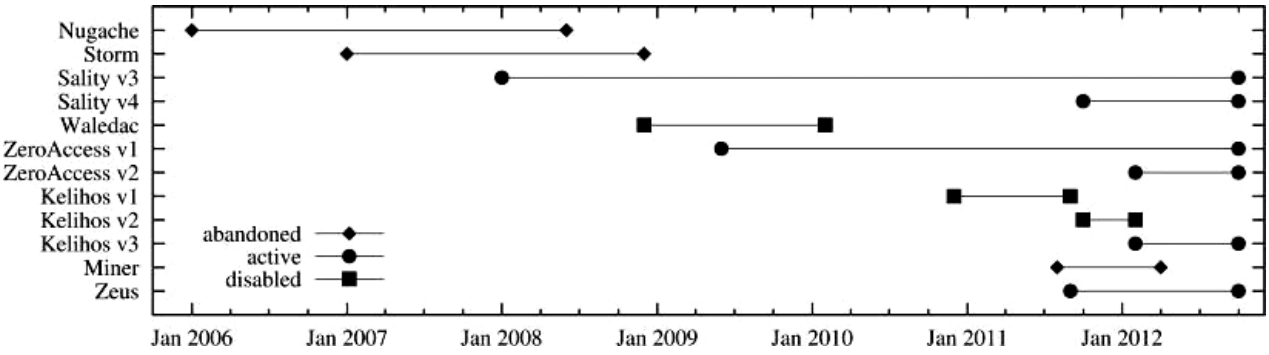

Рис. 1. Період функціонування різних типів ботнетів [11]

З рис. 1 видно, що найменший період часу функціонував ботнет для майнінгу криптовалюти – вісім місяців, а ботнет сім'ї Sality активно пропрацював понад п'ять років. Це говорить про те, що ботнети, побудовані на основі P2P архітектури, є дуже стійкими через їх властивість, яка дає змогу перебудувати мережу і зробити її непомітною для вже створених систем боротьби з нею. Такий тип архітектури дасть змогу забезпечити швидку реконфігурацію мережі та забезпечити стабільний перехідний період функціонування системи загалом. Таку архітектуру використаємо для побудови досліджуваної мережі ботнету та розроблення системи аналізу сесій.

3. Розроблення сценарію реалізації, керування та роботи мережі ботнетів

Для розроблення системи аналізу сесій із пристроями IoT потрібно розробити власний ботнет, оскільки це дасть змогу вести дослідження, спираючись на реальну модель, і отримати чіткі вхідні дані. Для цього необхідно визначити загальні положення та застосування ботнету. Прийнято рішення, що ботнет використовуватиметься для атакування пристроїв IoT, які переважно містяться у приватних мережах. Розроблення ботнету зображено на рис. 2.

Оскільки більшість пристроїв IoT підключені до мережі Wi-Fi або Bluethoth, стає очевидним, що потрібно атакувати ці пристрої не зі стороннього ПЗ або ж через проникнення до їх мереж, а через інші вразливіші пристрої у цій мережі. Саме тому за основу взято підключення через протокол SSH. Вибравши протокол, за допомогою якого працюватиме ботнет, а також встановивши цілі для атак, необхідно чітко розуміти, як можна інфікувати комп'ютер. Є кілька варіантів вирішення цієї проблеми:

- використовувати шкідливе ПЗ;
- здійснювати проникнення за допомогою соціальної інженерії.

Оскільки мета – зробити ботнет якомога непомітнішим для систем виявлення шкідливого ПЗ, використаймо соціальну інженерію. Соціальною інженерією (надалі – СІ) називають психологічні

маніпуляції людиною, за допомогою яких можна отримати доступ до конфіденційної інформації. Очевидно, що такий спосіб інфікування є менш надійним та неефективним у разі атакування досвідчених адміністраторів, проте може стати доволі ефективним, коли атакуватимуть звичайних користувачів ПК.

Рис. 2. Процес розроблення ботнету

Атак на основі СІ існує чимало, проте найпоширеніший фішинг атаки. Після виконання усіх попередніх етапів постає ще одна невирішена проблема – комунікація між ботами. Оскільки це однорангова мережа, а саме шкідливе ПЗ ніяк не взаємодіє із ОС інфікованих хостів, то власник мережі не має інформації про статус ботів.

Для вирішення цього завдання можна використовувати кілька методів:

- запис станів до бази даних;
- виконання команди ping для встановлення статусу хоста.

Використаймо для комунікації базу даних, у яку записуватимемо стани ботів для подальшої роботи з ними. На рис. 3 наведено загальний вигляд архітектури мережі ботів.

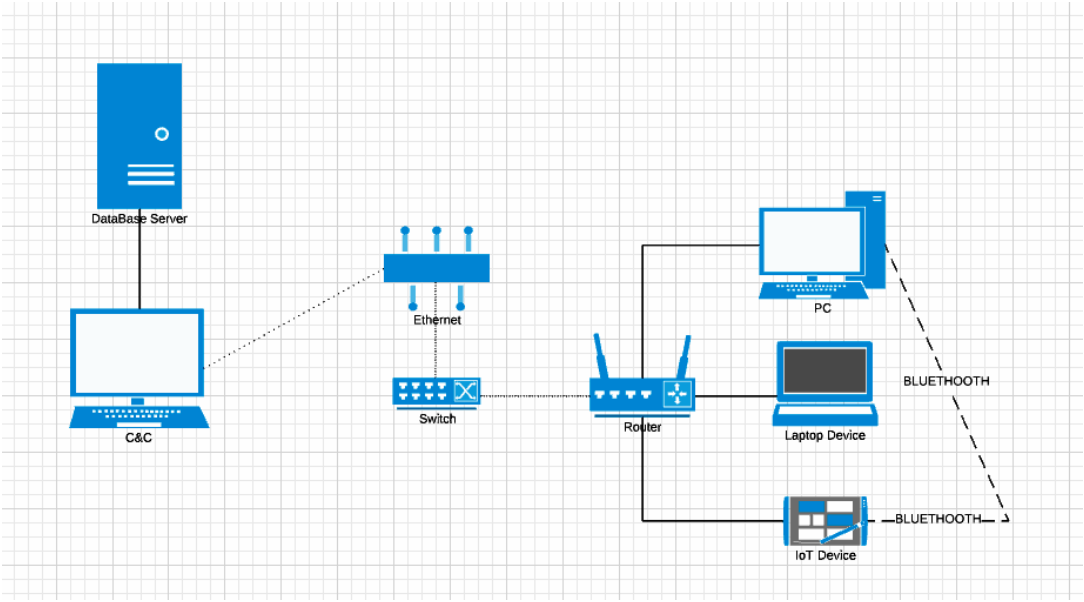

Рис. 3. Розроблена архітектура мережі ботів для захисту пристроїв мережі Інтернету речей

Інфікування хоста відбувається за рахунок CI. Оскільки використовується фішингова атака, вона зобов'язана налякати користувача та не дати йому часу для роздумів. Оскільки ми використовуємо протокол SSH, необхідно захопити пароль користувача. Це реалізовано за рахунок "невідкладного" оновлення системи, для якого необхідно встановити пакети.

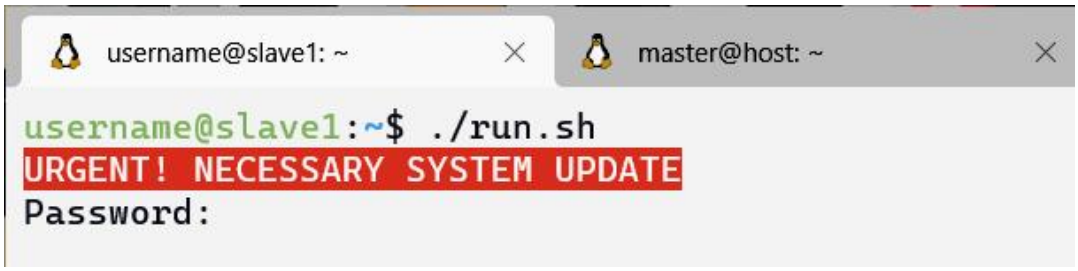

Рис. 4. Фішингова атака

Після введення пароля адміністратора вірус почне робити вигляд, що оновлює пакети, а це викличе довіру у власника ПК, на який у цей момент часу відбувається атака, що зображено на рис. 5. Проте небезпека цього вірусу полягає у тому, що він використовує усі легальні сервіси і не провокує антивірусні програмні засоби.

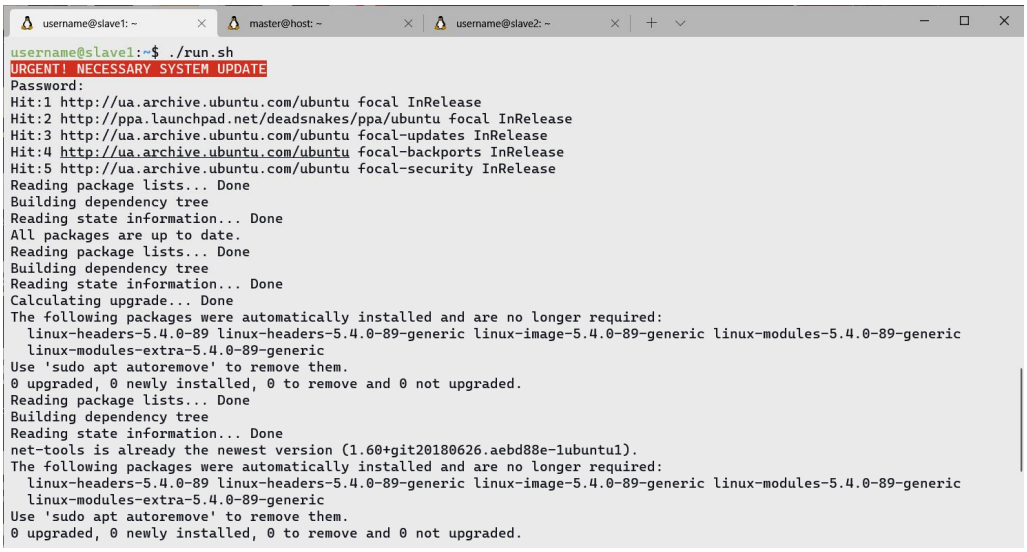

Рис. 5. Робота шкідливого ПЗ

Це шкідливе програмне забезпечення, окрім оновлення офіційних пакетів дистрибутива Linux, здійснює встановлення потрібних для роботи програми сервісів, а також, отримавши пароль адміністратора, відкриває необхідні мережеві порти у налаштуванні брандмауера. Наступним етапом роботи цього вірусу є збирання інформації про ПК жертви:

- IP адреса;
- ім'я користувача;
- пароль адміністратора;
- ввімкнення/вимкнення ПК жертви.

На останньому третьому етапі вірусне програмне забезпечення встановлює з'єднання із базою даних зловмисника та записує туди усю зібрану інформацію. Після виконання усіх цих кроків файл із зловмисним шкідливим текстом програми видаляється та повідомляє користувача про успішне встановлення пакетів оновлення, як зображено на рис. 6.

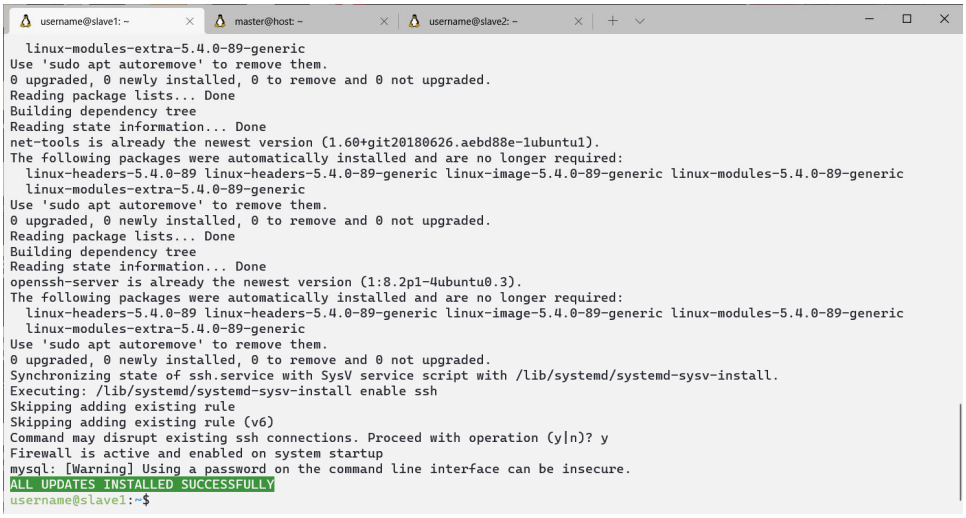

Рис. 6. Завершення процесу інфікування ПК

Зазначимо, що текст програми написаний мовою програмування Bash, яка підтримується всіма дистрибутивами Linux, тому вона може виконуватися без додаткового програмного забезпечення у вигляді компіляторів.

4. Тестування вразливості IoT пристроїв, атакованих ботом

Розроблену архітектуру мережі ботів та запропоновану атаку створено для дослідження стійкості системи у разі атакування із не відомого раніше пристрою. Виконаємо дослідження стійкості IoT пристрою, на який здійснюватимемо атаку. Для прикладу візьмемо фітнес-браслет Xiaomi Mi Band 3, який працює із використанням стандарту Bluetooth LE. Сама атака відбувалась у кілька етапів:

- Інфікування хоста.
- Залучення його до зловмисної мережі ботів.
- Копіювання за допомогою створеного функціонала файлів для подальших атак.
- Наказ боту проаналізувати всі відкриті та можливі з'єднання BLUETOOTH.
- Створення пари із пристроєм Xiaomi Ma Band 3.
- Початок спам-атаки.

Насамперед необхідно захопити файл логів про надсилання сповіщення на пристрій та проаналізувати їх за допомогою програми Wireshark (рис. 7).

На рис. 7 бачимо велику кількість переданих даних, проте очевидно, що не всі значення нам необхідні для успішного встановлення з'єднання між ботом та розумним годинником. Якщо проаналізувати дані, стає очевидним, що Mi Band 3 використовує протокол ATT запису даних. Важливим тут також є Generic Attribute Profile, побудований на вершині ATT, щоб надавати послуги високого рівня виробнику, який реалізує LE. Ці послуги використовують здебільшого для систематичнішого управління процесом передавання даних.

Опрацювавши наведені на рис. 7 дані, бачимо хендшейк між ботом та смарт-годинником у рядку під номером 102, який визначає девайс як 0xf000, та отримаємо усі необхідні атрибути відносно протоколу ATT. Значення атрибутів наведено на рис. 8.

Тепер, зібравши всі ці дані та проаналізувавши їх, бот може розпочати поетапну атаку на розумний годинник. Атака здійснюється за допомогою додатків дистрибутива Linux – hcitool та gatttoll

Перший етап сканування – наявність відкритих з'єднань. Наступні етапи – отримання MAC-адреси пристрою та підключення між ботом та смарт-годинником. Результат наведено на рис. 9.

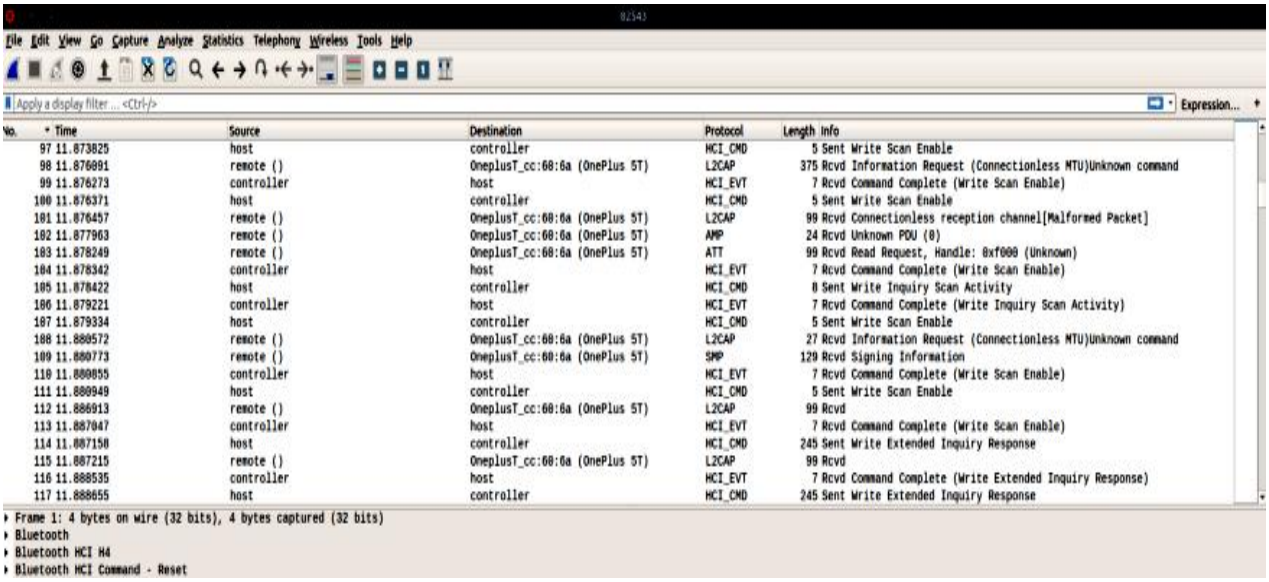

Рис. 7. Аналіз мережевого пакета, отриманого за допомогою Wireshark

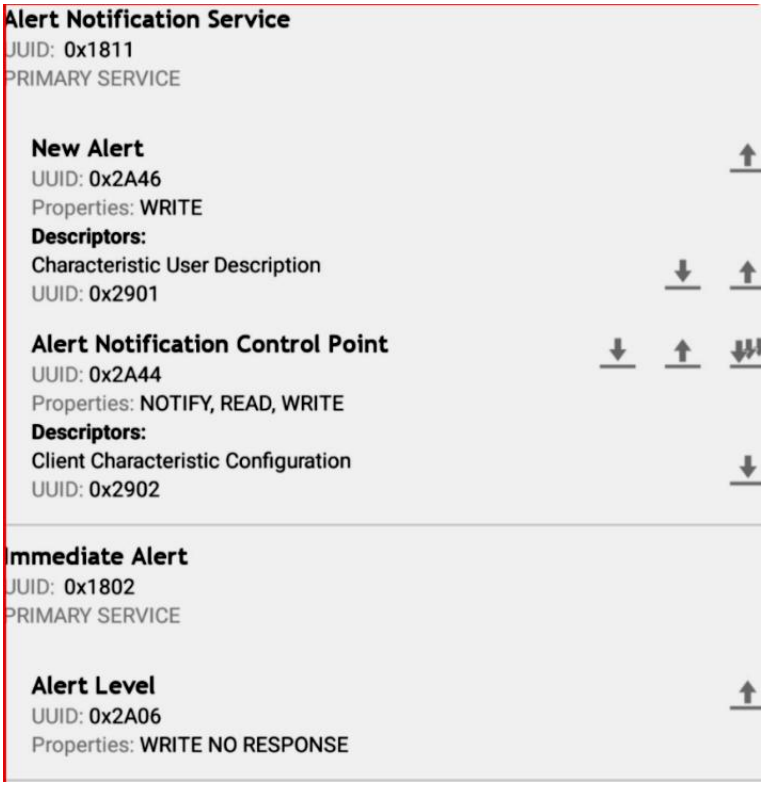

Рис. 8. Атрибути пристрою, на який здійснюється атака

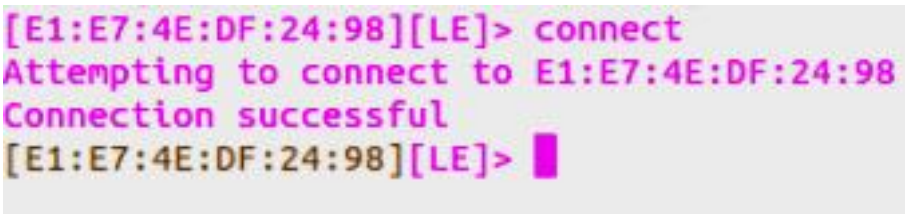

Рис. 9. Успішне підключення до Mi Band 3

Тепер у нас є можливість отримати список усіх первинних служб та команд. Для того, щоб ними скористатися, потрібно мати UUID та дескриптор пристрою. Отримані дані, такі як сервіси та характеристики, подано на рис. 10.

```
[E1:E7:4E:DF:24:98][LE]> connect
Attempting to connect to E1:E7:4E:DF:24:98
Connection successful
[E1:E7:4E:DF:24:98][LE]> primary
attr handle: 0x0001, end grp handle: 0x0007 uuid: 00001800-0000-1000-8000-00805f9b34fb
attr handle: 0x0008, end grp handle: 0x000b uuid: 00001801-0000-1000-8000-00805f9b34fb
attr handle: 0x000c, end grp handle: 0x0016 uuid: 0000180a-0000-1000-8000-00805f9b34fb
attr handle: 0x0017, end grp handle: 0x001c uuid: 00001530-0000-3512-2118-0009af100700
attr handle: 0x001d, end grp handle: 0x0023 uuid: 00001811-0000-1000-8000-00805f9b34fb
attr handle: 0x0024, end grp handle: 0x0026 uuid: 00001802-0000-1000-8000-00805f9b34fb
attr handle: 0x0027, end grp handle: 0x002c uuid: 0000180d-0000-1000-8000-00805f9b34fb
attr handle: 0x002d, end grp handle: 0x0057 uuid: 0000fee0-0000-1000-8000-00805f9b34fb
attr handle: 0x0058, end grp handle: 0x006c uuid: 0000fee1-0000-1000-8000-00805f9b34fb
```

Рис. 10. Список первинних сервісів

Отже, отримано дескриптори та унікальні ідентифікатори кожного із сервісів розумного годинника. Тепер за допомогою цих даних можна атакувати сервіси. Атаки можуть бути спрямовані на постійне відволікання жертви від роботи на сповіщення, повідомлення або ж недійсні дзвінки. Також це дасть змогу кіберзловмиснику шантажувати користувача різними повідомленнями. Та найнебезпечніша тут власне доступність до даних про серцебиття, записаних на скомпрометованому розумному годиннику. Протокол ATT потребує аутентифікації. Для успішної аутентифікації із розумним годинником Mi Band 3 потрібно виконати такі кроки:

- налаштування сповіщень про аутентифікацію (процес дебагу) відправленням двобайтового запиту – \x01\x00;
- надсилання 16-байтового ключа шифрування із типом змінної char та додавання до нього двох надлишкових байтів інформації - x01\x00 + ЗНАЧЕННЯ_КЛЮЧА;
- запит випадкового ключа від пристрою – здійснюється за допомогою відправлення двобайтового повідомлення \x02\x00;
- отримання випадкового ключа з відповіді пристрою (останні 16 байтів);
- шифрування цього випадкового числа за допомогою нашого 16-байтового ключа із використанням алгоритму шифрування AES/ECB/NoPadding (з Crypto.Cipher import AES) і відправлення його назад (\x03\x00 + закодовані дані);

Для успішного тестування треба зрозуміти, яка із характеристик на рис. 7 відповідає за конкретний сервіс. Для цього перехопимо пакет під час активації сервісу та проаналізуємо його (рис. 11).

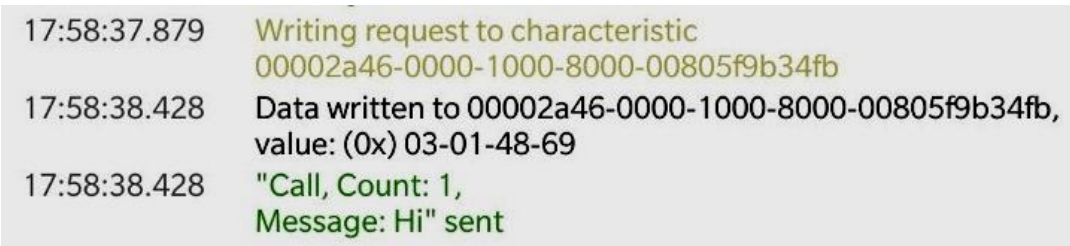

Рис. 11. Аналіз перехопленого пакета під час сповіщення

Рис. 12. Спам-атака, спрямована на розумний годинник

Як видно з рис. 11, запит відправлено до однієї із характеристик - 00002A46–0000–1000–8000–00805f9b34fb, перші два байти якої характеризують тип сповіщення. Серед них можна виділити такі:

- 01 – сповіщення про E-mail;
- 02 – сповіщення про дзвінок;
- 03 – сповіщення про пропущений дзвінок;
- 04 – сповіщення про отримане SMS-повідомлення.

Знаючи структуру та зміст даних, якими обмінюються пристрої мережі та фітнес-браслет, інфікованому хосту вдалося приєднатися до розумного годинника Mi Band 3 та розпочати спам-атаку на нього. Результат атаки подано на рис. 12.

5. Розроблення системи аналізу сесій для боротьби із ботнетами

Проаналізувавши всі дані, отримані у попередньому підпункті, розробимо систему для протидії ботнету. Оскільки операційна система Linux пропонує широкий спектр утиліт, за допомогою яких можна просканувати як мережу, так і процеси, які відбуваються у цей момент часу, прийнято рішення розробляти таку систему за допомогою мови програмування Bash. Очевидно, що будь-яку систему необхідно правильно розгортати. Розроблення систем на основі Honeynet є найоптимальнішим варіантом для швидкого сканування систем на ураження ботнетом. Оскільки ботнет використовує протокол SSH і під час інфікування хоста встановлює SSH сесію, достатньо просто перевірити його статус, як показано на рис. 13.

Наступним етапом розгортання є встановлення усіх необхідних пакетів. Для цього використаємо таку команду: – sudo apt install – y python-virtualenv libssl-dev libffi-dev build-essential libpython3-dev python3-minimal. Система розроблена так, що вона працює динамічно, не використовуючи значення за замовчуванням, проте все ж необхідно внести певні зміни в файли конфігурації, наприклад, значення ssh порту (за замовчуванням 22), а також IP-адресу інтерфейсу (за замовчуванням 0.0.0.0). Процес конфігурації зображено на рис. 14.

```
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enab
   Active: active (running) since Wed 2021-06-16 23:46:36 -03; 6s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 16561 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
Main PID: 16562 (sshd)
    Tasks: 1 (limit: 4915)
   Memory: 1.5M
   CGroup: /system.slice/ssh.service
           └─16562 /usr/sbin/sshd -D
```

Рис. 13. Статус SSH.service

Здійснивши необхідні зміни, можна перейти до запуску сервісу за допомогою команди bot_analyser.service start, а також виконати валідацію розгорнутої системи.

Необхідно визначити та встановити активні з'єднання через протокол SSH. Оскільки SSH використовує мережевий порт 22, розроблена система в режимі реального часу сканує саме цей порт. Результати цього сканування подано на рис. 16.

```
# (use systemd: endpoint for systemd activation)
# listen_endpoints = systemd:domain=INET:index=0
# For both IPv4 and IPv6: listen_endpoints = tcp6:2222:interface=\:\:
# Listening on multiple endpoints is supported with a single space seperator
# e.g listen_endpoints = "tcp:2222:interface=0.0.0.0 tcp:1022:interface=0.0.0.0" will result li$
# use authbind for port numbers under 1024

listen_endpoints = tcp:22:interface=0.0.0.0

# Enable the SFTP subsystem
# (default: true)
sftp_enabled = true


# Enable SSH direct-tcpip forwarding
# (default: true)
forwarding = true
```

Рис. 14. Файл конфігурації системи

```
root@LinuxHint:~# netstat -tan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State

tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
```

Рис. 15. Валідація роботи розробленої системи аналізу сесій

```
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
00192.168.217.13022192.168.217.132
```

Рис. 16. Сканування відкритих з'єднань

Як видно із рис. 16, система встановила, що до хоста 192.168.217.130 підключено хост 192..168.217.132 через порт 22. Оскільки система використовується як Honeynet, у цей момент часу до неї не підключений жоден верифікований користувач. Це означає, що система встановила IP-адресу бота, який передає команди нашому хосту. Наступний етап – розроблення функціонала для аналізу виконуваних команд. Результати наведено на рис. 17.

Останнім, проте не менш важливим елементом цієї истеми є режим блокування встановленої IP-адреси ботнету. Результат роботи системи подано на рис. 18.

Аналізуючи мережевий трафік, незалежно від типу сесій – чи TCP чи UDP, та запустивши усі розроблені підсистеми, матимемо можливість заблокувати бота, який атакував. Успішне виконання такого аналізу проілюстровано на рис. 19 та 20.

```
username@slave2:~$ ./analyser.sh
RemoteHostIP 192.168.217.132
execution
command scp -1 master@192.168.217.132:/tmp/bluezzatack.py username@192.168.217.130:/tmp
--------------------------------------------------------------------------------
RemoteHostIP 192.168.217.132
execution
command cd /tmp && python3 bluezzatack.py -scan
--------------------------------------------------------------------------------
```

Рис. 17. Режим прослуховування віддалених команд

```
username@slave2:~$ ./analyser.sh
Skipping adding existing rule
Status: inactive

network traffic from 192.168.217.132 is successfully blocked
username@slave2:~$
```

Рис. 18. Результат блокування IP адреси

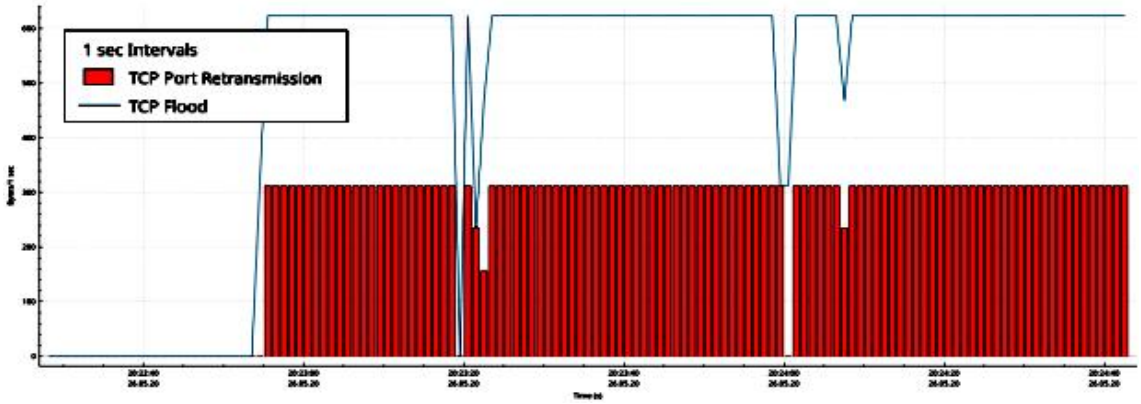

Рис. 19. Аналіз мережевого трафіку після блокування TCP трафіку

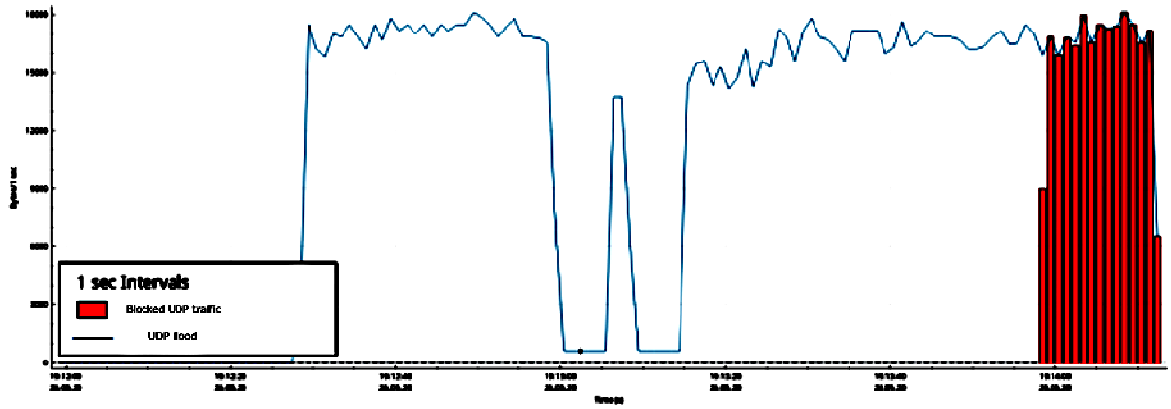

Рис. 20. Аналіз мережевого трафіку після блокування UDP трафіку

Як видно із рис. 19 та 20, реалізована система аналізу та захисту від ботнетів дає змогу проаналізувати хост та виявити основні ознаки його наявності в мережі ботів. Це дає можливість оперативно зреагувати та почати протидіяти такому зараженню. Система дозволяє отримати дані про встановлені активні з'єднання SSH, команди, які віддалено запускаються на цьому хості, а також автоматично заблокувати встановлені з'єднання та не допустити проникнення нових.

Запропонована система аналізу сесій реалізована за принципом Honeynet мереж, але, по суті, є гібридною, оскільки використовує модель автономних агентів, модель моніторингу мережі та модель виявлення вторгнень на основі поведінки. В результаті тестування запропонованої системи здійснено атаку на пристрій IoT та заблоковано зловмисника, що підтверджує її ефективність.

Висновки

Запропоновано систему аналізу сесій з пристроями IoT для боротьби із ботнетами і, як наслідок, захисту пристроїв мережі Інтернету речей від проникнення зловмисних мереж ботів. Запропонована система аналізу сесій реалізована за принципом Honeynet мереж, але, по суті, є гібридною, оскільки використовує модель автономних агентів, модель моніторингу мережі та модель виявлення вторгнень на основі поведінки. Використання цієї системи дає змогу проаналізувати хост та виявити основні ознаки його наявності в мережі ботів. Дані про встановлені активні з'єднання SSH, команди, які віддалено запускаються на хостах, а також автоматичне блокування встановлених з'єднань дадуть змогу отримати інформацію про стан IoT пристроїв та не допустити втрати або підміни персональних даних.

Список використаних джерел

[1] Botnets and their types // EC-Council. URL: https://blog.eccouncil.org/botnets-and-their-types/.

[2] Sochor T., Zuzcak M. (2014), Study of Internet Threats and Attack Methods Using Honeypots and Honeynets. In: Kwiecień A., Gaj P., Stera P. (eds) Computer Networks. CN 2014. Communications in Computer and Information Science, Vol. 431. Springer, Cham. URL:https://doi.org/10.1007/978-3-319-07941-7_12

[3] Livadas C., Walsh R., Lapsley D., and Strayer W. (2006) "Using machine learning techniques to identify botnet traffic", in Proceedings of the 2nd IEEE LCN Workshop on Network Security (WoNS'2006).

[4] Binkley J. and Singh S. (2006), "An algorithm for anomaly-based botnet detection, in Proceedings of USENIX SRUTI'06.

[5] Kang B. B. H. (2011), DNS-Based Botnet Detection. In: van Tilborg H. C. A., Jajodia S. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA. URL:https://doi.org/10.1007/978-1-4419-5906-5_845

[6] Cafuta D., Sruk V., Dodig I. (2018),"Fast-Flux Botnet Detection Based on Traffic Response and Search Engines Credit Worthiness", Technical Gazette, 25, 2(2018), pp. 390–40.

[7] Xingguo Li, JunfengWang, Xiaosong Zhang (2017), "Botnet Detection Technology Based on DNS", Future Internet, 9, 55. DOI:10.3390/fi9040055.

[8] Karawash A. (2015), Data protection and Brute Force attack, ResearchGate.

[9] The Attribute Protocol (ATT) // Dialog Semiconductor. URL: http://lpccs-docs.dialog-semiconductor.com/tutorial-custom-profile-DA145xx/att.html.

[10] Lee D. (2016), Recursive DNS: What It Is And Why You Should Care, Neystar. URL: https:// www.home.neustar/blog/recursive-dns-what-it-is-and-why-you-should-care.

[11] Water Torture: A Slow Drip DNS DDoS Attack (2014), Secure64 Software Corporation. URL: https:// secure64.com/2014/02/25/water-torture-slow-drip-dns-ddos-attack/.

[12] 1Modeling and Evaluating the Resilience of Peer-to-Peer Botnets (2013) / [C. Rossow, D. Andriesse, T. Werner та ін.], IEEE.

DEVELOPMENT OF SESSION ANALYSIS SYSTEMS WITH IOT DEVICES FOR PROTECTION AGAINST BOTNETS

S. Tukalo, O. Shpur, O. Kostiv

Lviv Polytechnic National University, 12, S. Bandery Str., Lviv, 79013, Ukraine

This paper has been devoted to the development of session analysis systems with IoT devices for protection against botnets and as a consequence of the protection of Internet of Things devices from the intrusion of malicious bot networks. To implement it, our own botnet based on the SSH protocol has been developed. To

ensure high reliability and decentralization, the botnet manages through a separate database server, which contains information about the status of bots, as well as general information about each of them. The proposed system of session analysis is implemented on the principle of Honeynet networks, but is essentially hybrid, as it uses a model of stand-alone agents, a model of network monitoring, and a model of intrusion detection based on behavior. The command server can steal files from an infected bot, perform any operations on behalf of the administrator, and affect smart devices. A smartwatch using Bluetooth LE was used for the study. As a result, we have created our own botnet protection system, which allows us to analyze the host and identify the main signs of the presence of this host in the bot network. This allows you to react quickly and start counteracting such an infection. The system allows you to obtain data on established active SSH connections, commands that are launched remotely on this host, as well as automatically block established connections and prevent the intrusion of new ones. As a result of testing the proposed system, an attack was made on the IoT device and an attacker was blocked, which confirms the effectiveness of its development.

Key words: *IoT; bot; botnet; SSH; SSH-botnet; Honeynet.*