Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(освітній рівень)

на тему: " Застосування техніки динамічного відкривання мережевих портів для підвищення безпеки корпоративних серверів "

Виконав: студент (ка)

Спеціальності:
125 «Кібербезпека»
(шифр і назва напряму підготовки, спеціальності)
Легкобит Олексій Юрійович
підпис (прізвище та ініціали)

Керівник Муж В.В.
підпис (прізвище та ініціали)

Нормоконтроль Тимощук Д. І.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент
підпис (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека
(шифр і назва спеціальності)

Студенту Легкобиту Олексію Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Застосування техніки динамічного відкривання мережевих портів для підвищення безпеки корпоративних серверів

Керівник роботи Муж Валерій Вікторович, к.ю.н., доцент кафедри КБ.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 15 » 04 2024 року № 4/7-350

2. Термін подання студентом завершеної роботи 12.06.2024

3. Вихідні дані до роботи Вимоги до безпеки серверів.
Операційні системи FreeBSD, Ubuntu Server, Red Hat Enterprise Linux, Windows

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ
1. Аналіз предметної області
2. Засоби створення техніки динамічного відкривання мережевих портів
3. Тестування техніки динамічного відкривання мережевих портів
4. Безпека життєдіяльності, основи охорони праці
Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Тема, мета, задачі. Метод захисту серверів з використання шифрованого VPN. Метод захисту серверів з використанням брандмауера. Метод захисту серверів з використанням ключів в SSH. Основи техніки port knocking. Варіанти реалізації port knocking. Безпека port knocking. Реалізація port knocking за допомогою iptables в RedHat Linux. Реалізація за допомогою knockd та UFW в Ubuntu Server. Port knocking в операційній системі FreeBSD. Встановлення та налаштування засобів тестування техніки port knocking в RedHat Linux. Тестування техніки port knocking в Ubuntu Server. Тестування техніки port knocking в FreeBSD. Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи хорони праці	Мариненко С. Ю., к.т.н. доцент кафедри МТ		

7. Дата видачі завдання 29.01.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	29.01.2024	
2.	Опрацювання джерел в галузі дослідження	02.02 – 30.01	
3.	Оформлення розділу «Огляд предметної області»	21.02 – 10.03	
4.	Оформлення розділу «Засоби створення техніки динамічного відкривання мережевих портів»	11.03 – 25.03	
5.	Оформлення розділу «Тестування техніки динамічного відкривання мережевих портів»	10.04 – 05.05	
6.	Оформлення розділу «Безпека життєдіяльності, основи охорони праці»	10.05 – 21.05	
7.	Оформлення кваліфікаційної роботи	23.05 – 06.06	
8.	Нормоконтроль	06.06 – 10.06	
9.	Перевірка на плагіат	11.06 – 12.06	
10.	Попередній захист кваліфікаційної роботи	14.06 – 15.06	
11.	Захист кваліфікаційної роботи	27.06.2024	

Студент _____ Легкобит О.Ю.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Муж В.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Застосування техніки динамічного відкривання мережевих портів для підвищення безпеки корпоративних серверів. // Кваліфікаційна робота ОР «Бакалавр» // Легкобит Олексій Юрійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБс-42 // Тернопіль, 2024 // С. 60, рис. – 25, табл. – 0, кресл. – 14, додат. – 1.

Ключові слова: port knocking, FreeBSD, Linux, Ubuntu, RedHat, SSH, iptables, knockd, ufw, PF, Windows, Oracle, MacOS.

У кваліфікаційній роботі бакалавра було проведено дослідження механізму динамічного відкривання мережевих портів з метою забезпечення безпеки корпоративних серверів. Проаналізовано поточні методи захисту механізмів адміністрування серверів, зокрема використання VPN, брандмауери та SSH-ключі. Хоча ці методи вважаються досить надійними, кожен з них має свої обмеження та недоліки, такі як можливість блокування VPN у деяких мережах або потенційна вразливість до атак на відкриті порти SSH.

Було розглянуто техніку port knocking. Описаний алгоритм роботи цієї техніки, за яким клієнт надсилає послідовність запитів до закритих портів на сервері, що призводить до автоматичного відкривання потрібного порту при виявленні правильної послідовності. Було досліджено різні методи реалізації port knocking в різних операційних системах та мережевому обладнанні.

У фінальному етапі було встановлено та налаштовано сервери з різними операційними системами та брандмауерами для реалізації техніки port knocking. Під час тестування було перевірено правильну роботу встановлених сервісів та налаштувань. Тести підтвердили ефективність та працездатність механізму port knocking, зокрема його здатність до відкриття потрібних портів та забезпечення безпеки підключення до серверів через SSH.

ANNOTATION

Application of dynamic port opening techniques to enhance the security of corporate servers. // Thesis of educational level "Bachelor"// Oleksii Lehkobyt // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, group СБс-42 // Ternopil, 2024 // P. 60, fig. - 25, tab. -0, chair. - 14 , added. - 1.

Keywords: port knocking, FreeBSD, Linux, Ubuntu, RedHat, SSH, iptables, knockd, ufw, PF, Windows, Oracle, MacOS.

In the bachelor's thesis, a study of the mechanism of dynamic opening of network ports was carried out in order to ensure the security of corporate servers.

The current methods of protecting server administration mechanisms, including the use of VPNs, firewalls, and SSH keys, were analyzed. Although these methods are considered to be quite reliable, each of them has its limitations and drawbacks, such as the possibility of VPN blocking in some networks or potential vulnerability to attacks on open SSH ports.

The port knocking technique was considered. The algorithm of this technique is described, according to which the client sends a sequence of requests to closed ports on the server, which leads to the automatic opening of the desired port when the correct sequence is detected. Various methods of implementing port knocking in different operating systems were investigated.

In the final stage, servers with different operating systems and firewalls were installed and configured to implement the port knocking technique. During the testing, we checked the correct operation of the installed services and settings. The tests confirmed the effectiveness and efficiency of the port knocking mechanism, in particular, its ability to open the necessary ports and ensure secure connection to servers via SSH.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

VPN	—	Virtual Private Network
SSH	—	Secure Shell
DDOS	—	Distributed Denial of Service
DOS	—	Denial of Service
TCP	—	Transmission Control Protocol
UDP	—	User Datagram Protocol
ACL	—	Access Control List
NAT	—	Network Address Translation
RPM	—	Red Hat Package Manager
LTS	—	Long-Term Support
UFW	—	Uncomplicated Firewall
UFS	—	Unix File System
ZFS	—	Zettabyte File System
FAT	—	File Allocation Table
PF	—	Packet Filter
ОС	—	Операційна система
ICMP	—	Internet Control Message Protocol

ВСТУП

Сучасні тенденції у сфері інформаційної безпеки свідчать про постійне зростання загроз інформаційним системам, зумовлене не тільки розвитком технологій, але й стійким бажанням зловмисників вдосконалювати методи атак на корпоративні сервери. Цей постійний еволюційний характер загроз вимагає постійного удосконалення та адаптації засобів і методів захисту інформаційних систем. В даному контексті, техніка динамічного відкривання мережевих портів, відома як port knocking, виступає як один із ефективних методів забезпечення додаткового рівня захисту для корпоративних серверів.

Метою даного дослідження є детальний аналіз, розробка та впровадження техніки динамічного відкривання мережевих портів для підвищення безпеки корпоративних серверів.

Для досягнення цієї мети посталено наступні задачі:

- розгляд існуючих методів захисту мережевих ресурсів та визначення їх недоліків;
- вивчення основ техніки динамічного відкривання мережевих портів;
- аналіз переваг і обмежень техніки port knocking;
- налаштування техніки port knocking в операційній системі FreeBSD;
- реалізація техніки port knocking в операційній системі Ubuntu Server;
- налаштування техніки port knocking в операційній системі RedHat Linux;
- тестування ефективності захисту, використовуючи операційні системи Windows 10, Oracle Linux та MacOS а також клієнти port knocking та SSH.

Об’єктом дослідження є сучасні системи безпеки корпоративних серверів, які використовуються для надійного захисту мережевих ресурсів та інформації.

Предметом дослідження є техніка port knocking, яка включає в себе динамічне відкривання мережевих портів на сервері в залежності від послідовності звернень на певні порти сервера.

Одержані результати дослідження сприятимуть створенню більш ефективних та надійних методів захисту корпоративних серверів від потенційних атак. Розробка та впровадження техніки динамічного відкривання

мережевих портів може відігравати важливу роль у забезпеченні конфіденційності та доступності корпоративної інформації, зменшуючи ризик несанкціонованого доступу до серверних ресурсів.

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд існуючих методів захисту механізмів адміністрування серверів

Сервери, які є важливою складовою інформаційної інфраструктури, піддаються різноманітним загрозам інформаційної безпеки. Віддалене адміністрування серверів є критично важливим аспектом для ефективного управління та підтримки інформаційної інфраструктури. Забезпечення безпеки цих сервісів вимагає вдосконаленої стратегії та використання передових методів захисту.

1.1.1 Використання шифрованого VPN

Шифрований VPN – це технологія, що дозволяє створити безпечне та зашифроване з'єднання між двома точками через ненадійну мережу, як-от Інтернет [1]. Основна мета – забезпечити конфіденційність даних під час їхньої передачі між двома вузлами.

VPN використовує тунель для передачі даних між двома вузлами через ненадійну мережу. Тунель може бути реалізований за допомогою різних протоколів, таких як IKEv2/IPsec, OpenVPN, L2TP/IPsec тощо.

Основний принцип – шифрувати дані перед їхньою передачею, а потім розшифровувати їх при отриманні. Це гарантує конфіденційність інформації та убезпечує від перехоплення зловмисниками. VPN використовує методи автентифікації, такі як паролі, сертифікати або двофакторну автентифікацію, для перевірки автентичності користувачів та забезпечення доступу тільки для авторизованих осіб.

Використання шифрованого VPN особливо корисне при використанні відкритих та ненадійних мереж, таких як кав'ярні, аеропорти чи готелі. VPN допомагає відсіяти багато типів атак, таких як Man-in-the-Middle, DNS-спуфінг та інші.

Недоліком використання VPN є те що деякі мережі можуть блокувати або обмежувати використання VPN. Наприклад, деякі країни чи організації можуть блокувати доступ до певних типів VPN або взагалі блокувати його використання що звісно вплине на можливість його використання для віддалено адміністрування серверів.

1.1.2 Використання брандмауера

Брандмауер є ключовим елементом в забезпеченні безпеки мережевих середовищ, включаючи сервери [2]. Основні функції брандмауера включають контроль доступу та захист від потенційно небезпечних з'єднань. Його застосування для захисту механізмів адміністрування серверів дозволяє ефективно управляти доступом та зменшити ризик небажаного трафіку [3].

Брандмауер дозволяє налаштовувати правила для обмеження доступу до сервера лише з визначених IP-адрес чи діапазонів IP, зменшуючи ризик несанкціонованого доступу. Брандмауер може вести журнали мережевого трафіку, що дозволяє адміністраторам виявляти аномальну активність та потенційні атаки [4].

Недоліком використання брандмауера для захисту механізмів адміністрування серверів є те що потреба в доступі до сервера може виникнути з IP-адрес, які не прописані, як дозволені в правилах брандмауера. В таких випадках використання лише брандмауера не є гнучким методом для надання доступу та може призвести до втрати доступу до сервера. Також важливо враховувати можливість використання динамічних IP-адрес.

1.1.3 Використання ключів в SSH

SSH є популярним протоколом для безпечного з'єднання з віддаленими серверами та виконання адміністративних завдань. Однак SSH також є мішенню для атак, зокрема brute force атак [5].

Brute force атака на SSH - це спроби незаконно отримати доступ до сервера, вгадуючи та перевіряючи всі можливі комбінації паролів. Зловмисники використовують автоматизовані інструменти для швидкого перебору паролів, доки не знайдуть правильний.

Використання довгих та складних паролі згідно політики паролів може значно ускладнити атаки методом перебору. Також налаштування обмеження на кількість невдалих спроб входу за певний час з блокуванням IP-адрес, які перевищили поріг покращить захист від атак на SSH.

Використання автентифікацію за допомогою SSH-ключів замість паролів надасть безпечний спосіб доступу який не чутливий до brute force атак. Ключі в SSH є одним із найбільш безпечних методів автентифікації, порівняно з введенням паролів [6].

Недоліком використання SSH-ключів для захисту механізмів адміністрування серверів є те що на порт SSH зловмисники можуть здійснити DOS або DDOS атаки. DDOS атака на SSH може стати серйозним викликом для безпеки віддалених серверів. У DDOS атаках зловмисники намагаються перевантажити сервер або мережу, здійснюючи велику кількість запитів. Це в кінцевому випадку може призвести до відмови сервісу SSH в зв’язку з перевантаженням.

1.2 Основи техніки port knocking

Мінімальний рівень безпеки для загальнодоступного сервера часто ґрунтується на принципі безпеки на основі хоста, що означає обмеження доступу до сервера лише з вказаних IP-адрес. Це виправдано для забезпечення безпеки, адже відкриття, наприклад, SSH-порту для всіх адрес може призвести до небажаної активності, такої як DDOS чи brute force атаки. Заборона доступу з усіх адрес може ускладнити налаштування, особливо якщо потрібно здійснювати доступ з різних частин світу. Потрібно уникати простого дозволу всім IP-адресам на підключення до критичних служб, таких як SSH, для уникнення можливих загроз і атак на сервер.

Port knocking - це метод динамічного відкриття порту, який за замовчуванням закритий для всіх вхідних IP-адрес, і може бути відкритий за наявності певного типу автентифікації на основі певного шаблону запитів [7]. У цьому методі клієнт відправляє певну послідовність запитів (зазвичай це TCP або UDP пакети) до набору закритих портів на сервері. Коли сервер виявляє правильну послідовність, він автоматично відкриває потрібний порт для дозволу з'єднань.

Ідея застосування цього механізму безпеки полягає в тому, щоб налаштувати на сервері шаблон, який буде випадковим і непрогнозованим для потенційних зловмисників. На сервері встановлюється брандмауер, який за замовчуванням блокує всі вхідні запити до портів, де працюють служби адміністрування, такі як SSH. Тільки при отриманні правильної послідовності запитів брандмауер розблоковує доступ до вказаних портів, надаючи можливість клієнту підключитися. Цей підхід дозволяє додатково зменшити ризик несанкціонованого доступу до служб адміністрування, таким чином підвищуючи рівень безпеки сервера.

Сучасні програмні брандмауери можуть блокувати запити на основі різноманітних параметрів, таких як мережеві порти, тип трафіку і вхідна IP-адреса. Навіть при налаштуванні брандмауера на блокування всіх запитів, він все одно бачить всі типи запитів, що надходять.

Діапазон номерів портів від 1 до 65535 представляє велику кількість можливих комбінацій. Брандмауер аналізує інформацію про те, що запит надійшов від конкретної IP-адреси до певного порту призначення. Після отримання запиту він перевіряє встановлені користувачем правила, щоб визначити, які дії слід виконати щодо цього запиту. Отже, суть у тому, що всі запити, які надсилаються до сервера, доступні для моніторингу та аналізу брандмауером.

Операційна система володіє здатністю реєструвати всі мережеві запити, що надходять до її мережевих інтерфейсів, незалежно від того, чи активована служба, яка може відповісти на ці запити, чи ні. Ядро операційної системи налаштоване ініціювати переривання при отриманні кожного мережевого

запиту. Переривання є механізмом, який використовується ядром для того, щоб сповістити центральний процесор про необхідність обробки отриманого запиту.

Отже, основна ідея цього захисту полягає в тому, щоб використовувати певну програму, яка генерує серію запитів на основі шаблонів та відправляє їх до закритих портів (knocking). Далі інший демон відслідковує цей шаблон надсилання запитів і ініціює задану користувачем дію, якщо шаблон запитів є правильним [8].

Для прикладу, системний інженер, який прагне отримати віддалений доступ через SSH до віддаленого сервера налаштовує на сервері сценарій/демон, який моніторить отримані запити. Також він конфігурує сервер для виклику подій на основі шаблону запиту. Наприклад, він може встановити шаблон, який надсилає запити на порти 3003 TCP, 4004 UDP та 5005 TCP.

Припустимо, що системний інженер, перебуваючи в будь-якому місці, надсилає запити на підключення до цих портів у точно визначеному шаблоні (3003 TCP, 4004 UDP, 5005 TCP). Це ініціює подію, яку він попередньо налаштував на сервері. Таким чином, він може попросити сервер відкрити порт 22 для його вихідної IP-адреси (з якої він надсилав запити до портів сервера) (див.рисунок 1.1)..

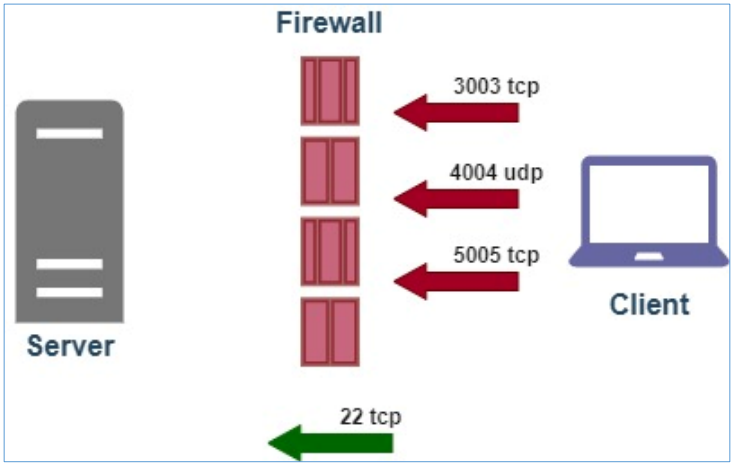

Рисунок 1.1 – Схема техніки port knocking

Ураховуючи загальну кількість доступних портів на сервері (від 1 до 65535), системний інженер може налаштувати випадкові порти, які

використовуватимуться як шаблон для автентифікації. Після виконання завдання системний інженер може закрити використаний порт. Закриття порту також є подією, ініційованою сервером, коли інший шаблон надсилається у вигляді запиту для цього порту або автоматично за певний проміжок часу.

1.3 Варіанти реалізації port knocking

Можливість налаштування техніки port knocking існує на різних операційних системах та обладнанні. В операційних системах Linux (Ubuntu, Debian, Red Hat, CentOS та інші) можливе використання програми knockd або власних сценаріїв для обробки запитів і налаштування правил ufw, iptables, nftables або firewalld. FreeBSD також використовує knockd або інші подібні інструменти в інтеграції з pf, ipfw або ipfilter. В Windows для реалізації port knocking можна використовувати сторонні програми або скрипти.

Щодо обладнання, налаштування port knocking можливе на різних типах мережевого обладнання, якщо воно підтримує відповідні функції. Наприклад, деякі маршрутизатори або брандмауери можуть мати можливість налаштування port knocking або аналогічні засоби для реалізації подібного механізу. Реалізація port knocking може залежати від конкретної моделі та версії операційної системи або обладнання, тому для конкретних випадків рекомендується докладно перевіряти документацію та ресурси виробників.

Реалізація техніки port knocking на обладнанні Cisco може варіюватися в залежності від конкретної моделі та версії IOS. В основі цього процесу зазвичай лежить використання ACL та тригерів, які ініціюють зміни в правилах фільтрації трафіку.

Для реалізації техніки port knocking на маршрутизаторах MikroTik використовується брандмауер та правила фільтрації пакетів. В Juniper Networks, реалізація техніки port knocking може варіюватися в залежності від конкретного обладнання та операційної системи Junos OS. Для налаштування port knocking в Juniper також використовується брандмауер та політики фільтрації трафіку.

В маршрутизаторах Palo Alto Networks, реалізація техніки port knocking може бути здійснена за допомогою правил безпеки Security Policy та Address Groups. В маршрутизаторах Fortinet, реалізація техніки port knocking може бути здійснена за допомогою правил застосування політик Policies і розширених правил фільтрації.

1.4 Застереження що до безпеки port knocking

Добрим аргументом що до зменшення безпеки техніки port knocking є можливість прослуховування пакетів зловмисником для виявлення використання port knocking. Якщо зловмисник виявить серію статичних SYN-запитів в прослуханому пакеті і побачить відсутність відповіді на ці SYN-запити, а потім, наприклад, буде спостерігати SSH-трафік, він може припустити, що насправді відбувається port knocking. Зловмисник може підтвердити це, виявивши, що порт SSH був закритий на сервері, який він відстежує, і після серії запитів SYN до випадкових портів спостерігається SSH-трафік.

Виникає проблема безпеки в техніці port knocking, пов'язана з відсутністю методу автентифікації клієнта. У зв'язку з цим важливо зазначити, що техніка port knocking не надає ніякого вбудованого механізму перевірки автентичності клієнта. Коли сервер отримує послідовність правильно сформованих запитів, він вважає, що це коректний сигнал для відкриття доступу до певного сервісу, такого як SSH.

Також використання техніки port knocking у сценаріях з використанням NAT може викликати проблеми з контролем доступу. Одним з основних труднощів є те, що за NAT одна зовнішня IP-адреса може представляти кілька внутрішніх пристроїв, і, отже, відкривши доступ для конкретної IP-адреси по суті відкривається доступ для всіх пристроїв за цією IP-адресою.

Враховуючи цю обставину, важливо мати додаткові альтернативні методи автентифікації та контролю доступу до портів адміністрування сервера, які не базуються на IP-адресах.

Використання довгих та складних паролі згідно політики паролів може значно ускладнити зловмиснику атаки методом перебору після отримання доступу до порту SSH. Також налаштування обмеження на кількість невдалих спроб входу за певний час з блокуванням IP-адрес, які перевищили поріг покращить захист від атак на SSH і не тільки. Використовуючи автентифікацію за допомогою SSH-ключів замість паролів створить безпечний спосіб доступу який не чутливий до brute force атак. Двофакторна автентифікація - це метод захисту, який вимагає введення двох різних видів ідентифікаційних даних для підтвердження особи. Зазвичай це є комбінація введення основного пароля або ідентифікаційних даних та підтвердження за допомогою другого фактора (SMS-код, Google Authenticator).

Доступ надається лише після успішної автентифікації обома факторами. Двофакторна автентифікація підвищує рівень безпеки, оскільки для незаконного доступу потрібно не тільки знати пароль, але і мати доступ до додаткового фактора, який зазвичай є більш складним для підміни чи вторгнення.

Ключів в SSH та двофакторна автентифікація є найбільш безпечні методи автентифікації, порівняно з введенням лише логіна та пароля.

1.5 Висновки до розділу

В першому розділі було здійснено огляд захисту механізмів адміністрування серверів. Показано що використання VPN для доступу до сервера, брандмауера та ключів SSH є надійними методами захисту механізмів адміністрування серверів, але з певними недоліками. Недоліком використання VPN є те що деякі мережі можуть блокувати або обмежувати використання VPN. Недоліком використання брандмауера для захисту механізмів адміністрування серверів є те що потреба в доступі до сервера може виникнути з IP-адрес, які не прописані, як дозволені в правилах брандмауера. Недоліком використання SSH-ключів для захисту механізмів адміністрування серверів є те що на порт SSH зловмисники можуть здійснити DOS або DDOS атаки.

Описано основи техніки динамічного відкривання портів (port knocking). У цьому методі клієнт відправляє певну послідовність запитів (зазвичай це TCP або UDP пакети) до набору закритих портів на сервері. Коли сервер виявляє правильну послідовність, він автоматично відкриває потрібний порт для дозволу з'єднань. Проаналізовано різні механізми реалізації port knocking в операційних системах та на різних типах мережевого обладнання.

Також у розділі висвітлено застереження що до безпеки техніки port knocking, а саме прослуховування пакетів зловмисником для виявлення використання port knocking та відкривання доступу до сервера з NAT IP-адрес.

РОЗДІЛ 2 ЗАСОБИ СТВОРЕННЯ ТЕХНІКИ ДИНАМІЧНОГО ВІДКРИВАННЯ МЕРЕЖЕВИХ ПОРТІВ

2.1 Загальний алгоритм роботи техніки port knocking

На рисунку 2.1 показана алгоритм роботи техніки port knocking для надання доступу до порту адміністрування сервера.

Рисунок 2.1 – Алгоритм роботи техніки port knocking

На рисунку показано алгоритм роботи техніки port knocking. Клієнт надсилає пакети на визначені порти сервера 3003 TCP, 4004 UDP, 5005 TCP.

Брандмауер має ідентифікувати вхідні пакети, які коректно сформовані. Якщо брандмауер не ідентифікує ці пакети як коректну частину протоколу TCP/IP, пакети будуть відкинуті. Після того як брандмауер розпізнає пакети як коректні, він передає послідовність пакетів до сервісу, який перевіряє, чи була послідовність надісланих пакетів вірною. Якщо послідовність пакетів була неправильною, пакети будуть відкинуті. Якщо послідовність була правильною, сервіс надсилає запит брандмауеру на тимчасове відкриття порту віддаленого адміністрування (наприклад SSH). Брандмауер відкриває порт для віддаленого адміністрування для IP-адреси клієнта, який виконав правильний порядок надсилання пакетів port knocking. Клієнт має можливість з'єднатися з сервером через SSH використовуючи відкритий порт.

Цей алгоритм показує роботу методу, який є одним зі способів зменшення поверхні атаки на сервер, забезпечуючи додатковий рівень захисту від несанкціонованого доступу до сервісів віддаленого адміністрування, які зазвичай слухають вхідні з'єднання, такі як SSH, RDP, WinBOX та інші.

2.2 Техніка port knocking в операційній системі Linux

Реалізація техніки port knocking в операційній системі Linux передбачає встановлення демона або налаштування скрипта, який відстежує послідовність запитів на мережеві порти і відповідає на цю послідовність відкриттям доступу до служби або ресурсів на сервері.

2.2.1 Реалізація за допомогою iptables в RedHat Linux

Red Hat Linux є одним з провідних комерційних дистрибутивів операційної системи Linux і входить до складу великої екосистеми програмного забезпечення Red Hat.

Red Hat Enterprise Linux (RHEL) є ентерпрайз-версією, яка призначена для використання в корпоративному середовищі та побудована на основі відкритого вихідного коду [9]. Вона надає надійність, безпеку та підтримку для підприємств, які використовують Linux-сервери. Red Hat забезпечує систему безпеки та оновлення через свій центр безпеки. Компанія також пропонує платний рівень підтримки для користувачів, які вимагають професійної допомоги.

RHEL використовує систему управління пакетами RPM для встановлення та управління програмним забезпеченням. Red Hat сприяє розвитку відкритих технологій та активно співпрацює зі спільнотою великих проектів, таких як Fedora.

Iptables є інструментом управління фільтрацією пакетів у системах Linux [10]. В основному, це засіб для конфігурації правил брандмауера в ядрі операційної системи. Пакет iptables призначений для налаштування та керування правилами брандмауера на операційних системах Linux. Він надає можливість фільтрації пакетів і визначення правил для їх обробки. Дозволяє адміністраторам визначати правила безпеки для вхідного та вихідного трафіку на мережевому рівні. Використовується для захисту системи від небажаного доступу, контролю мережевого трафіку та забезпечення безпеки. Iptables організований у вигляді таблиць та ланцюжків, де кожна таблиця відповідає конкретному аспекту мережевого трафіку. Існують різні таблиці, такі як filter (за замовчуванням), nat, mangle, тощо. Також є можливість створювати додаткові таблиці. Адміністратор може визначити правила для обробки пакетів на основі різних критеріїв, таких як IP-адреса, порти, протоколи тощо. Правила можуть вказувати, як обробляти пакет (приймати, відхиляти, пересилати тощо). Ціль визначає, що робити з пакетом, який відповідає певному правилу. Ціль може бути, наприклад, ACCEPT (прийняти), DROP (відхилити), REJECT (відхилити з відправкою відповіді), MASQUERADE (тип NAT), тощо. Iptables дозволяє виконувати різні дії з пакетами, такі як зміна їхніх адрес, визначення маршрутів, пересилання тощо. Це дозволяє гнучко налаштовувати обробку мережевого трафіку. Взаємодія з iptables відбувається через командний рядок. Користувачі можуть додавати,

видаляти та переглядати правила, налаштовувати таблиці та ланцюжки за допомогою команд.

На рисунку 2.2 показано команди, які використовуються для інсталяції та активації iptables в RedHat Linux.

```
[root@sr3-redhat93 ~]# dnf install iptables-services
Updating Subscription Management repositories.
Last metadata expiration check: 0:21:12 ago on Sat 09 Mar 2024 10:04:27 PM EET.
Package iptables-nft-services-1.8.8-6.el9_1.noarch is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@sr3-redhat93 ~]# systemctl enable iptables
[root@sr3-redhat93 ~]# systemctl start iptables
[root@sr3-redhat93 ~]# systemctl status iptables
● iptables.service - IPv4 firewall with iptables
     Loaded: loaded (/usr/lib/systemd/system/iptables.service; enabled; preset: disabled)
     Active: active (exited) since Sat 2024-03-09 21:57:34 EET; 28min ago
   Main PID: 38062 (code=exited, status=0/SUCCESS)
        CPU: 22ms

Mar 09 21:57:33 sr3-redhat93 systemd[1]: Starting IPv4 firewall with iptables...
Mar 09 21:57:34 sr3-redhat93 iptables.init[38062]: iptables: Applying firewall rules: [  OK  ]
Mar 09 21:57:34 sr3-redhat93 systemd[1]: Finished IPv4 firewall with iptables.
[root@sr3-redhat93 ~]#
```

Рисунок 2.2 – Команди інсталяції та активації iptables в RedHat Linux

На рисунку 2.3 показано скрипт, який використовує iptables для реалізації механізму port knocking на сервері. Це скрипт для оболонки Linux, який конфігурує брандмауер, щоб тимчасово відкрити порт SSH лише після отримання коректної послідовності звернень на порти 3003 TCP, 4004 UDP, 5005 TCP. Параметр iptables –F видаляє всі правила з усіх ланцюжків. Параметр iptables –X видаляє всі користувацькі ланцюжки. Параметр iptables –Z скидає всі лічильники пакетів та байтів у всіх правилах та ланцюжках. Потім створюються нові ланцюги для керування процесом port knocking: STATE0, STATE1, STATE2 та STATE3. У ланцюзі STATE0, якщо пакет приходить на порт TCP 3003, його вважають першою послідовністю (KNOCK1) і позначають за допомогою модуля recent для відстеження, а потім відкидають. У ланцюзі STATE1, якщо раніше був зафіксований KNOCK1 і пакет приходить на порт UDP 4004, він вважається другою послідовністю (KNOCK2), позначається і відкидається. Пакети без попереднього KNOCK1 відкидаються.

У ланцюзі STATE2, якщо були виявлені KNOCK1 і KNOCK2 і пакет приходить на порт TCP 5005, він вважається третьою послідовністю (KNOCK3), позначається і відкидається. Пакети без попередніх KNOCK1 та KNOCK2 відкидаються.

У ланцюзі STATE3, якщо KNOCK3 був зафіксований, сервер відкриває доступ до порту TCP 22 для IP-адреси, яка відправила коректну послідовність пакетів. Це відбувається шляхом прийняття ICMP та SSH трафіку від цієї IP-адреси.

Також в скрипті присутні правила для прийняття вхідного трафіку, який вже було встановлено (ESTABLISHED та RELATED) та для локального трафіку (з мережі 127.0.0.0/8).

```
#!/bin/sh -x

iptables -F
iptables -X
iptables -Z

iptables -N STATE0
iptables -A STATE0 -p tcp --dport 3003 -m recent --name KNOCK1 --set -j DROP
iptables -A STATE0 -j DROP

iptables -N STATE1
iptables -A STATE1 -m recent --name KNOCK1 --remove
iptables -A STATE1 -p udp --dport 4004 -m recent --name KNOCK2 --set -j DROP
iptables -A STATE1 -j STATE0

iptables -N STATE2
iptables -A STATE2 -m recent --name KNOCK2 --remove
iptables -A STATE2 -p tcp --dport 5005 -m recent --name KNOCK3 --set -j DROP
iptables -A STATE2 -j STATE0

iptables -N STATE3
iptables -A STATE3 -m recent --name KNOCK3 --remove
iptables -A STATE3 -p tcp --dport 22 -j ACCEPT
iptables -A STATE3 -p icmp -j ACCEPT
iptables -A STATE3 -j STATE0

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -s 127.0.0.0/8 -j ACCEPT

iptables -A INPUT -m recent --name KNOCK3 --rcheck -j STATE3
iptables -A INPUT -m recent --name KNOCK2 --rcheck -j STATE2
iptables -A INPUT -m recent --name KNOCK1 --rcheck -j STATE1
iptables -A INPUT -j STATE0
```

Рисунок 2.3 – Команди iptables для реалізації механізму port knocking в RedHat Linux

Отже, цей скрипт налаштовує правила брандмауера так, що доступ до порту SSH можливий тільки після послідовного коректного надсилання пакетів на певні порти.

2.2.2 Реалізація за допомогою knockd та UFW в Ubuntu Server

Ubuntu Linux є однією з найпопулярніших та широко використовуваних дистрибутивів операційної системи Linux, яка розробляється компанією Canonical Ltd. Ubuntu Server - це версія операційної системи Ubuntu Linux, спеціально призначена для використання на серверах [11]. Вона базується на відкритих стандартах, надаючи надійну та ефективну платформу для розгортання різних серверних застосунків.

Ubuntu є вільним та відкритим програмним забезпеченням. Це означає, що є можливість завантажити, встановити та користуватися ним безкоштовно, а також змінювати його вихідний код за власним бажанням. Випуски Ubuntu регулярно оновлюються і виходять за фіксованим графіком. Є версії з довгостроковою підтримкою (LTS) та регулярні випуски. Ubuntu має широкий набір програмного забезпечення, що дозволяє легко встановлювати та оновлювати різноманітні програми. В комплектацію Ubuntu включено багато корисних програм та драйверів, що дозволяє користувачам швидко почати роботу. Ubuntu підтримує різні архітектури, включаючи x86, x86-64, ARM тощо, що робить його універсальним для різних пристроїв.

Операційна система регулярно отримує оновлення безпеки та підтримку з боку спільноти. Ubuntu Server розроблено з урахуванням потреб високоякісного масштабування. Є підтримка різних серверних аплікацій, віртуалізації та контейнеризації.

Засоби безпеки, такі як UFW, сприяють захисту системи від потенційних загроз. UFW є інтерфейсом для налаштування брандмауера в операційних системах Ubuntu та інших дистрибутивах [12]. Він надає зручний спосіб управління iptables, який є більш розширеним інструментом для керування правилами мережевої безпеки в Linux. UFW зазвичай вже встановлено в

операційних системах Ubuntu. Для активації використовується команда `ufw enable`. Після активації брандмауер буде запущено при кожному завантаженні системи. Для деактивації використовується команда `ufw disable`.

В UFW команди зазвичай мають інтуїтивно зрозумілий синтаксис та використовують ключі та параметри для налаштування брандмауера. Основні команди виглядають наступним чином:

- додати дозвільне правило для конкретного порту `ufw allow [порт]`;
- додати дозвільне правило для конкретного порту та протоколу `ufw allow [порт]/[протокол]`;
- видалити правило для конкретного порту `ufw delete allow [порт]`;
- видалити правило для конкретного порту та протоколу `ufw delete allow [порт]/[протокол]`;
- заборонити доступ до певного порту `ufw deny [порт]`;
- відхилити підключення до певного порту `ufw reject [порт]`;
- обмежити частоту спроб звернень на певний порт/протокол `ufw limit [порт/протокол]`;
- перевірити стан брандмауера та список правил `ufw status`;
- переглянути нумерований список правил `ufw status numbered`;
- скинути всі правила до стану за умовчанням `ufw reset`;
- увімкнути або вимкнути логування подій `ufw logging [on/off]`.

UFW має вбудовані профілі для різних служб, наприклад, `ufw allow 'OpenSSH'` дозволяє SSH-з'єднання, і `ufw allow 'Nginx HTTP'` - HTTP-з'єднання для Nginx.

В Ubuntu server використовується knockd сервіс, який дозволяє реалізувати техніку port knocking на сервері [13]. Встановлення knockd в Ubuntu server виконується командою `apt install knockd`.

На рисунку 2.4 показано створену конфігурацію сервісу knockd у системі systemd Ubuntu Linux.

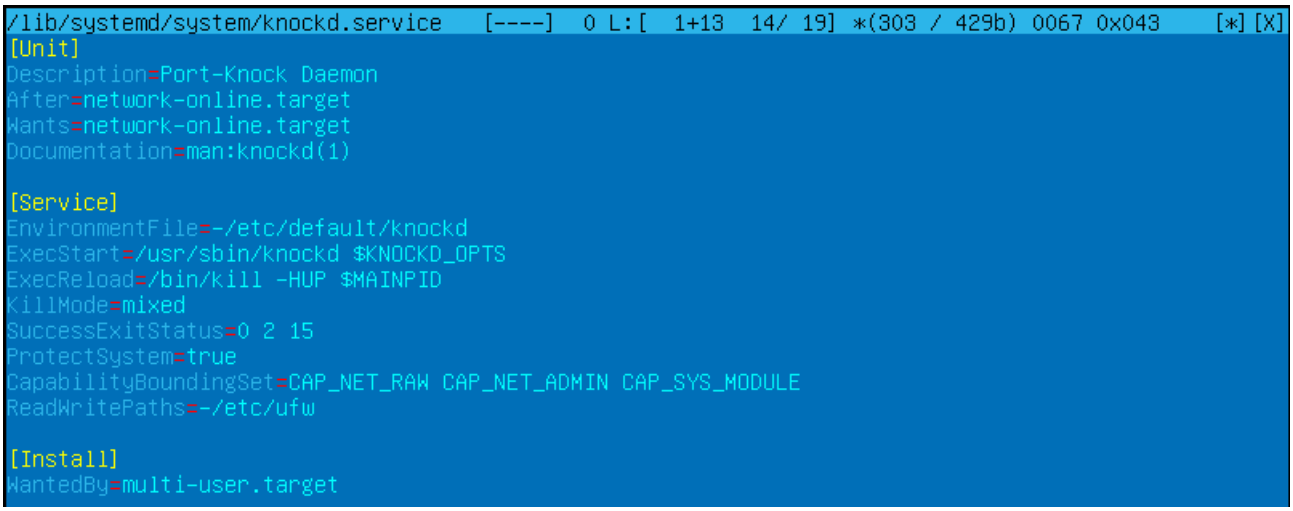

Рисунок 2.4 – Конфігурація сервісу knockd у системі systemd Ubuntu Linux

Цей конфігураційний файл визначає параметри та поведінку сервісу knockd в системі systemd.

Параметри цього файлу мають наступне значення:

1) `Description` - опис сервісу, який відображає призначення сервісу (Port-Knock Daemon);

2) `After` - вказує, що сервіс повинен запуститися після запуску `network-online.target` (мережі);

3) `Wants` - вказує, що сервіс knockd залежить від сервісу `network-online.target`;

4) `EnvironmentFile` - вказує шлях до файлу з оточенням (`/etc/default/knockd`), де можуть бути визначені додаткові змінні середовища.

5) `ExecStart` - команда, яка виконується при запуску служби, у даному випадку, це виклик knockd.

6) `ExecReload` - команда для перезавантаження служби.

7) `KillMode` - вказує, як слід завершувати процеси при зупинці служби (у даному випадку, mixed - спочатку надсилається сигнал завершення, а потім зупиняється процес).

8) `SuccessExitStatus` - коди виходу, які розглядаються як успішні (0 2 15).

9) `ProtectSystem` - включає або виключає захист файлової системи (в даному випадку, ввімкнено).

10) `ReadWritePaths` – розташування в які служба має право читати та записувати (в даному випадку `/etc/ufw/`).

11) `WantedBy` - вказує, що служба повинна бути запущена в багатьох режимах (в даному випадку `multi-user.target`).

На рисунку 2.5 показано конфігураційний файл knockd.comf, який визначає правила для knockd в Ubuntu Linux.

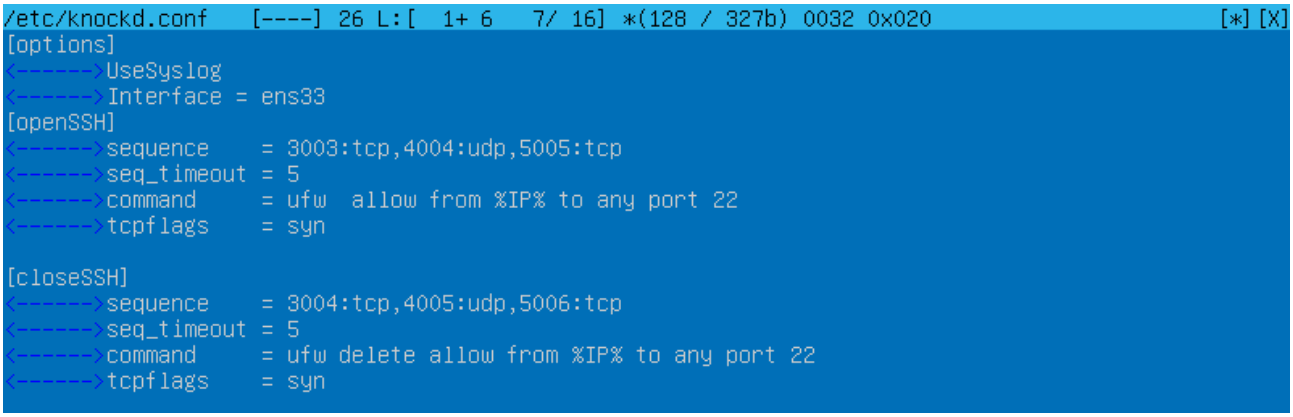

Рисунок 2.5 – Конфігураційний файл knockd.comf в Ubuntu Linux

Конфігураційні параметри цього файлу мають наступне значення:

1) `UseSyslog` - вказує, що knockd повинен використовувати syslog для журналювання подій.

2) `Interface` - вказує мережевий інтерфейс, який слід використовувати.

3) `openSSH` - це ім'я правила для відкриття SSH.

4) `sequence` - послідовність, яку потрібно надіслати, щоб відкрити SSH.

5) `seq_timeout` - таймаут для очікування повної послідовності.

6) `command` - команда, яка виконується при вірній послідовності (додає правило в ufw для доступу до порту 22).

7) `tcpflags` - вказує прапорці TCP, які має мати пакет для виклику події.

8) `closeSSH` - це ім'я правила для закриття SSH (аналогічно openSSH, але використовує ufw delete для видалення правила).

На рисунку 2.6 показано вивід команди `systemctl status knockd`, яка використовується для перевірки статусу сервісу в системі.

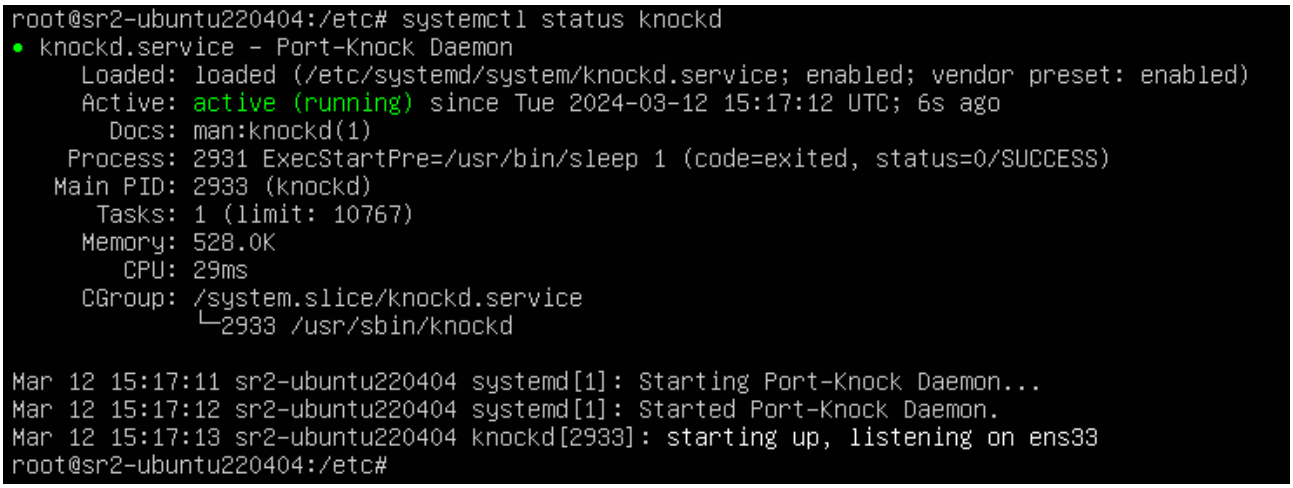

Рисунок 2.6 – Вивід команди перевірки статусу сервісу `knockd` в системі в Ubuntu Linux

Сервіс запущений (active (running)) та очікує на отримання звернень.

2.3 Техніка port knocking в операційній системі FreeBSD

FreeBSD - це безкоштовна та відкрита операційна система, яка базується на UNIX, а саме на системі UNIX BSD [14]. Основною метою FreeBSD є надання стабільної та надійної операційної системи для різних типів обчислювальних пристроїв, включаючи сервери, настільні комп'ютери та вбудовані системи. FreeBSD славиться своєю стабільністю та надійністю. Вона відома як операційна система, яка працює без перезавантажень протягом тривалих періодів часу.

Розповсюджується FreeBSD під ліцензією BSD, що дає користувачам велику свободу використання, модифікації та розповсюдження коду. Операційна система показує високу продуктивність в різних областях від вебсерверів до мережевих пристроїв. FreeBSD має велику кількість вбудованих функцій та інструментів для різних завдань, таких як мережеві послуги, безпека, віртуалізація тощо. FreeBSD має велику та активну спільноту користувачів та розробників, які надають підтримку, розвивають нові функції та вирішують проблеми. ОС підтримує різні файлові системи, включаючи UFS, ZFS та FAT. Файлові системи відповідають за організацію та збереження файлів на диску. FreeBSD має підтримку різних протоколів мережевого зв'язку, таким як TCP/IP,

IPsec. Це дозволяє FreeBSD використовуватися як мережевий сервер або маршрутизатор. FreeBSD має власну систему управління пакетами, яка називається pkg. Ця система дозволяє користувачам легко встановлювати, оновлювати та видаляти програми та бібліотеки з репозиторіїв FreeBSD. ОС має різні оболонки командного рядка, такі як tcsh, csh, bash, які надають користувачам інтерфейс для взаємодії з операційною системою.

Брандмауер PF в операційній системі FreeBSD надає широкі можливості для захисту мережі та керування мережевим трафіком [15]. Він дозволяє фільтрувати пакети на основі різних критеріїв, таких як IP-адреси, порти, протоколи тощо. PF також може бути використаний для перенаправлення мережевого трафіку на інші інтерфейси або сервери, реалізації NAT та контролю за політикою безпеки. Він працює як частина ядра операційної системи і надає потужний та надійний інструмент для налаштування правил фільтрації та обробки мережевого трафіку.

PF здатний бачити стан всіх з'єднань, які проходять через брандмауер. Це дозволяє фільтрувати пакети на основі їхнього стану (stateful filtering), що дозволяє забезпечити безпеку мережі та ефективне управління трафіком. PF дозволяє контролювати швидкість мережевого трафіку та призначати пріоритети за допомогою обмежень на швидкість або використання черг (traffic shaping). Це корисно для забезпечення надійності мережі та підтримки якості обслуговування. PF може вести журнал всіх вхідних та вихідних пакетів, що дозволяє адміністраторам відслідковувати та аналізувати мережевий трафік. Це корисно для виявлення атак та вирішення проблем мережі.

В операційній системі FreeBSD використовується демон knockd для динамічного відкривання портів на основі надсилання послідовності пакетів на відповідні порти [16]. Він дозволяє захистити сервер, приховавши сервіси, які прослуховуються, за допомогою закритих портів, та відкривати їх лише у випадку виявлення певної послідовності пакетів, що надходять на інші порти.

Для встановлення knockd в FreeBSD можна використати пакетний менеджер pkg та команду `pkg install knockd`.

Після встановлення knockd потрібно налаштувати його конфігураційний файл `/usr/local/etc/knockd.conf` згідно вимог. На рисунку 2.7 показано конфігураційний файл `knockd.conf` в FreeBSD.

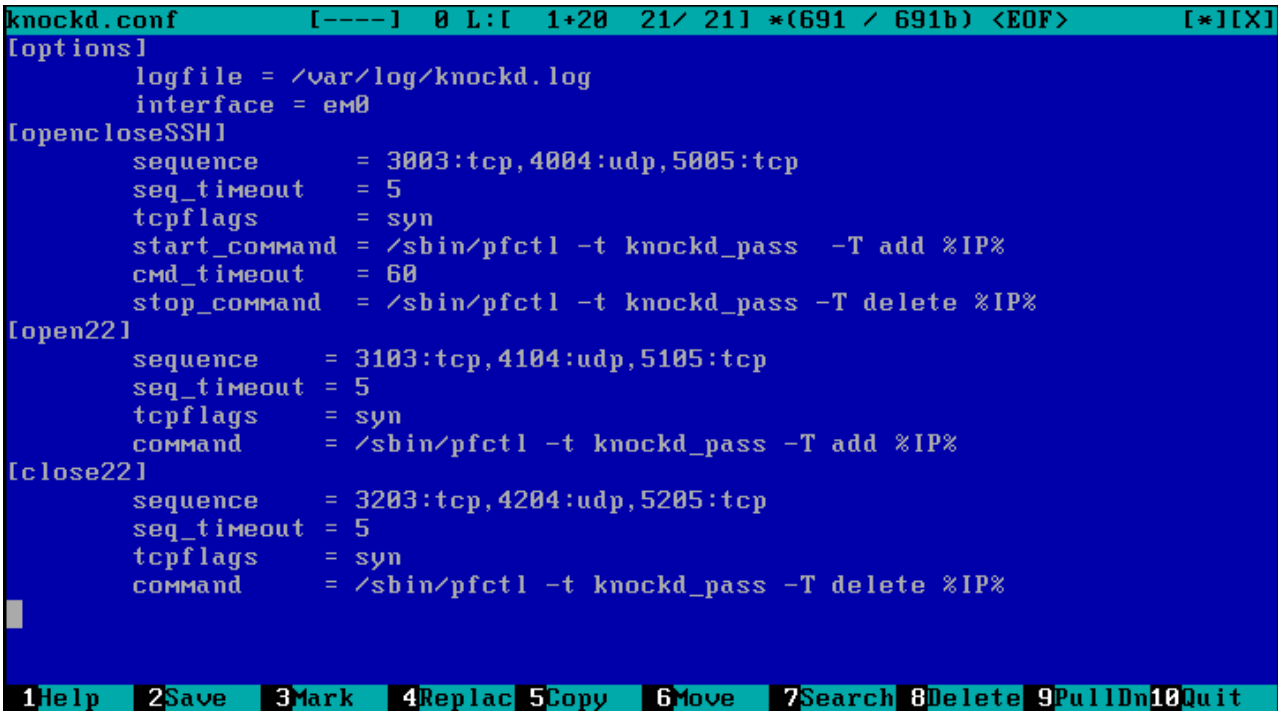

Рисунок 2.7 – Конфігураційний файл `knockd.conf` в FreeBSD

В цьому файлі визначаються послідовності пакетів та дії, які knockd має виконати при їх виявленні.

У файлі визначено чотири секції: options, opencloseSSH, open22 і close22 кожна з яких містить різні параметри.

Конфігураційні параметри цього файлу мають наступне значення:

1) `logfile` - вказує шлях до файлу, де будуть зберігатися логи сервісу (`/var/log/knockd.log`);

2) `interface` - визначає мережевий інтерфейс, на якому knockd буде слухати послідовності пакетів (`em0`);

3) `sequence` - послідовність, яку потрібно надіслати для запуску start_command (`3003:tcp,4004:udp,5005:tcp`).

4) `seq_timeout` - таймаут для очікування повної послідовності (`5`).

5) `tcpflags` - вказує прапорці TCP, які має мати пакет для виклику події (`syn`).

6) `start_command` - команда, що виконується при коректній послідовності (`/sbin/pfctl -t knockd_pass -T add %IP%`).

7) `cmd_timeout` - час у секундах протягом якого буде доступ до порту сервісу SSH (`60`).

8) `stop_command` - команда, що виконується для заборони доступу до порту сервісу SSH (`/sbin/pfctl -t knockd_pass -T delete %IP%`).

Секція open22 налаштована лише для відкриття доступу, використовуючи іншу послідовність портів (3103 TCP, 4104 UDP 5105 TCP) і команди pfctl. Секція close22 налаштована лише для закриття доступу, використовуючи іншу послідовність портів (3203 TCP, 4204 UDP 5205 TCP).

Після налаштування конфігураційного файлу потрібно запустити knockd за допомогою команди `service knockd start`.

Щоб knockd запускався автоматично при завантаженні системи потрібно додати запуск сервісу до файлу `/etc/rc.conf` за допомогою команди `sysrc knockd_enable="YES"`.

На рисунку 2.8 показано вмістом файлу pf.conf, який є конфігураційним файлом для PF брандмауера в ОС FreeBSD.

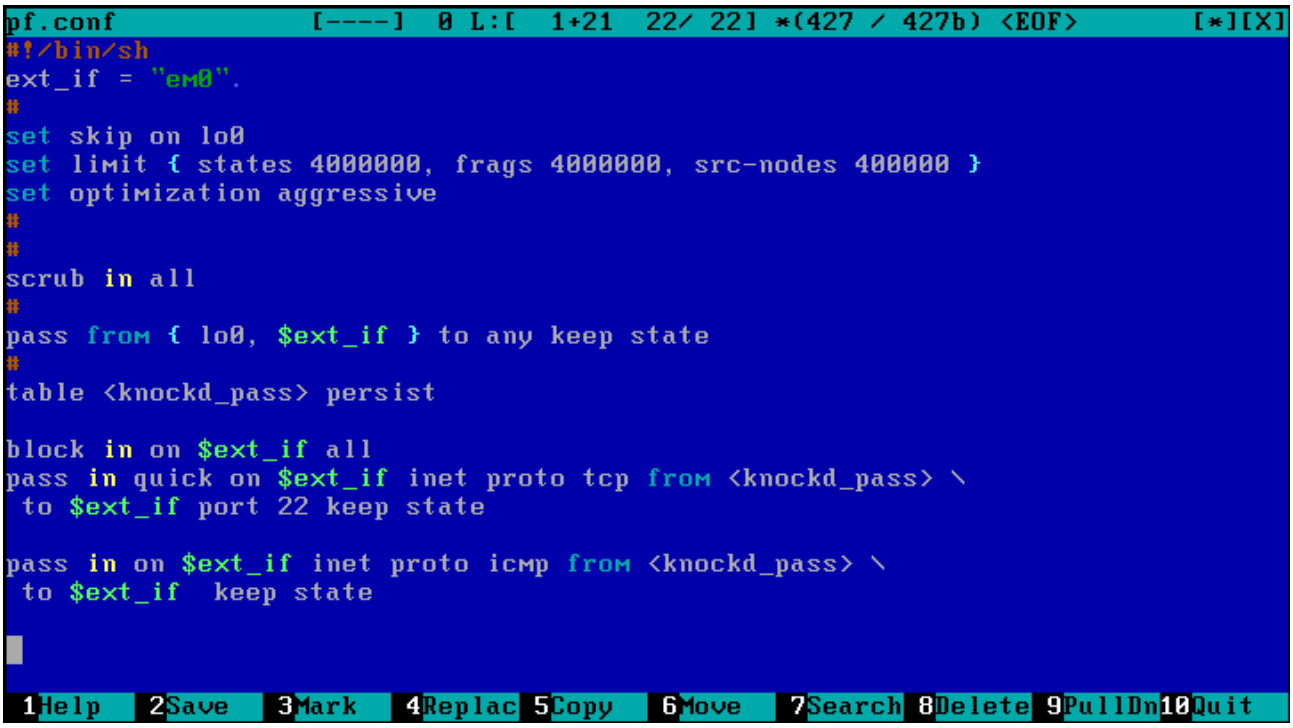

Рисунок 2.8 – Конфігураційний файл `pf.conf` в FreeBSD

Конфігураційні параметри цього файлу мають наступне значення:

1) `ext_if="em0"` - це присвоєння змінної, яка вказує на зовнішній інтерфейс мережі, який буде використовуватися в PF;

2) `set skip on lo0` – вказує PF пропускати весь трафік, який йде через інтерфейс lo0, який є локальним інтерфейсом петлі;

3) `set limit { states 4000000, frags 4000000, src-nodes 4000000 }` - встановлює ліміти для станів, фрагментів і вузлів джерел, що визначають максимальну кількість кожного, яку PF може обробляти;

4) `set optimization aggressive` - встановлює агресивний режим оптимізації для PF, який може покращувати продуктивність для певних сценаріїв;

5) `scrub in all` - правило, яке очищає всі пакети, що надходять. Може нормалізувати пакети для додаткової безпеки;

6) `pass from { lo0, $ext_if } to any keep state` - дозволяє вихідний трафік з інтерфейсів lo0 та зовнішнього інтерфейсу до будь-якого призначення, зберігаючи стан з'єднання;

7) `block in on $ext_if all` - блокує весь вхідний трафік на зовнішньому інтерфейсі.

Таблиця `knockd_pass` використовується для зберігання IP-адрес, які надіслали коректну послідовність до демона knockd. Подальші правила у файлі визначають, як поводитися з пакетами, що відповідають критеріям в таблиці `knockd_pass`.

Правило `pass in quick on $ext_if inet proto tcp from <knockd_pass> to $ext_if port 22 keep state` пропускає TCP трафік на порт SSH з IP-адресами, зазначеними в таблиці `knockd_pass`. Також є правило для обробки ICMP пакетів для IP-адрес з таблиці `knockd_pass`.

На рисунку 2.9 показано вивід команди `service knockd status`, яка використовується для перевірки статусу сервісу в системі.

```
root@sr1-freebasd14:/etc# service knockd status
knockd is running as pid 1802.
root@sr1-freebasd14:/etc#
```

Рисунок 2.9 – Вивід команди перевірки статусу сервісу `knockd` в FreeBSD

Сервіс knockd працює та має ідентифікатор процесу (PID) 1802.

2.4 Висновки до розділу

У другому розділі була показано загальний алгоритм роботи техніки port knocking для надання доступу до порту адміністрування сервера. Яка є одним зі способів зменшення поверхні атаки на сервер, забезпечуючи додатковий рівень захисту від несанкціонованого доступу до сервісів віддаленого адміністрування, таких як SSH, RDP, WinBOX та інші.

Встановлено та налаштовано Red Hat Enterprise Linux з брандмауером iptables та здійснено реалізацію техніки port knocking лише засобами iptables. Скрипт з командами iptables налаштовує правила брандмауера так, щоб доступ до порту SSH був можливий тільки після послідовного коректного надсилання пакетів на певні порти.

Встановлено та налаштовано Ubuntu Server з брандмауером UFW та здійснено реалізацію техніки port knocking за допомогою сервісу knockd та брандмауера UFW.

Встановлено та налаштовано FreeBSD з брандмауером PF та здійснено реалізацію техніки port knocking за допомогою сервісу knockd та брандмауера PF з використанням таблиць та стану з’єднань.

РОЗДІЛ 3 ТЕСТУВАННЯ ТЕХНІКИ ДИНАМІЧНОГО ВІДКРИВАННЯ МЕРЕЖЕВИХ ПОРТІВ

3.1 Встановлення та налаштування засобів тестування

Для тестування техніки динамічного відкривання портів скористаємось інструментами knock.exe для Windows, а також knock для Oracle Linux та MacOS. Ці утиліти дозволять надсилати послідовності пакетів на визначені порти сервера, щоб ініціювати відкриття необхідних портів.

Утиліта knock.exe для Windows дозволяє створювати та надсилати послідовності пакетів на вказані порти сервера. Утиліта knock є аналогічною до knock.exe, але призначена для операційних систем Oracle Linux та MacOS. Вона також дозволяє створювати та відправляти послідовності пакетів для ініціювання port knocking.

Використання цих утиліт дозволить виконати тестування техніки port knocking на різних операційних системах і перевірити, як сервер реагує на ці запити.

В додатку А представлено файл вихідного коду knock.c на мові програмування C, який буде використано для компілювання. На рисунку 3.1 показано параметри компілювання утиліти knock.exe та knock.

```
root@ubuntu2204ws:/home/admin/knock# x86_64-w64-mingw32-gcc knock.c -lws2_32 -o
knock.exe
root@ubuntu2204ws:/home/admin/knock# gcc knock.c -o knock
root@ubuntu2204ws:/home/admin/knock#
```

Рисунок 3.1 – Компілювання утиліти knock.exe та knock в Ubuntu Linux

Перша команда компіляції використовує компілятор x86_64-w64-mingw32-gcc для компіляції програми з файлу knock.c у виконуваний файл knock.exe для платформи Windows. Опція -lws2_32 додає бібліотеку ws2_32 для роботи з мережевими функціями в середовищі Windows.

Друга команда компілює файлу knock.c і створює виконуваний файл з ім'ям knock для платформи Linux та MacOS.

3.2 Тестування техніки port knocking в RedHat Linux

Щоб протестувати техніку port knocking на RedHat Linux з Windows 10 використаємо утиліту knock.exe для надсилання послідовності звернень на певні порти та PuTTY для підключення через SSH.

На рисунку 3.2 показано мережеві налаштування операційної системи RedHat Linux.

```
[root@sr3-redhat93 ~]# ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.142  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::20c:29ff:fea3:d24  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:a3:0d:24  txqueuelen 1000  (Ethernet)
        RX packets 490688  bytes 740478729 (706.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 124867  bytes 8866176 (8.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 451  bytes 44766 (43.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 451  bytes 44766 (43.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@sr3-redhat93 ~]#
```

Рисунок 3.2 – Мережеві налаштування операційної системи RedHat Linux

Згідно налаштувань брандмауера iptables будь-які вхідні з’єднання до RedHat Linux заборонені (див. рисунок 2.3) включно з протоколом ICMP (див.рисунок 3.3)

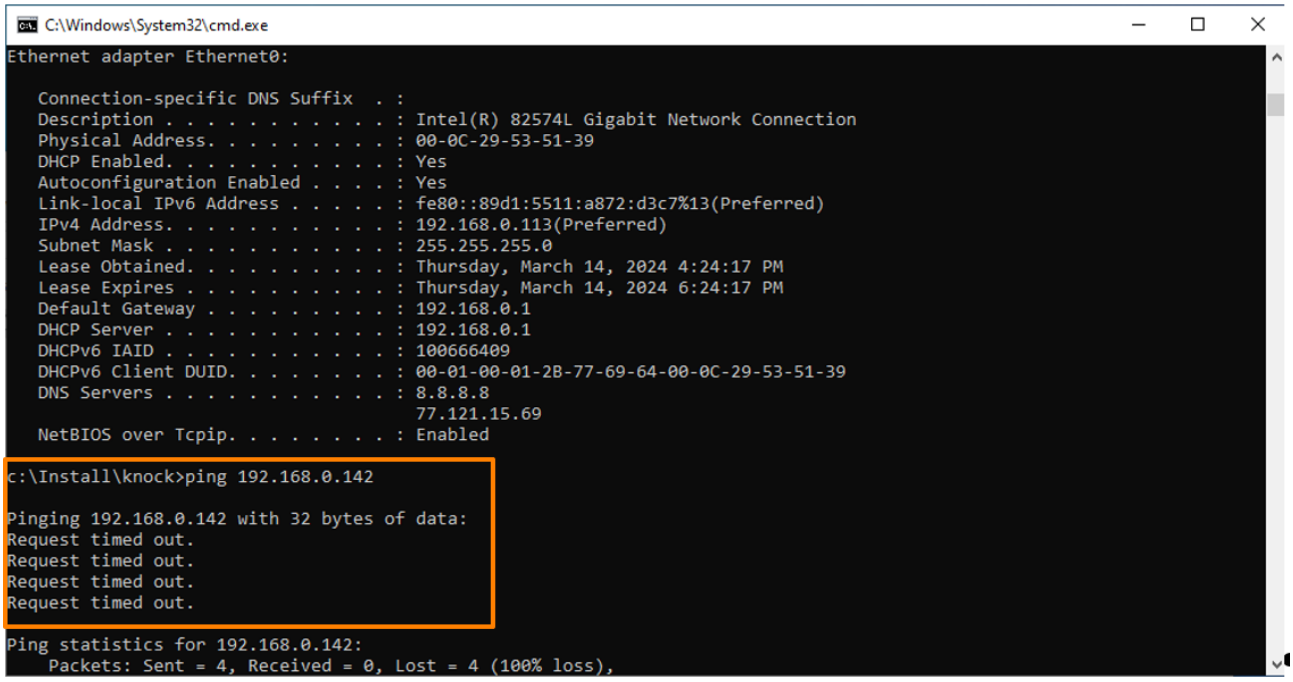

Рисунок 3.3 – Перевірка доступності хоста RedHat Linux

На рисунку 3.4 показано вивід утиліта knock.exe, яка відправила послідовність пакетів на вказані порти сервера з IP-адресою 192.168.0.142.

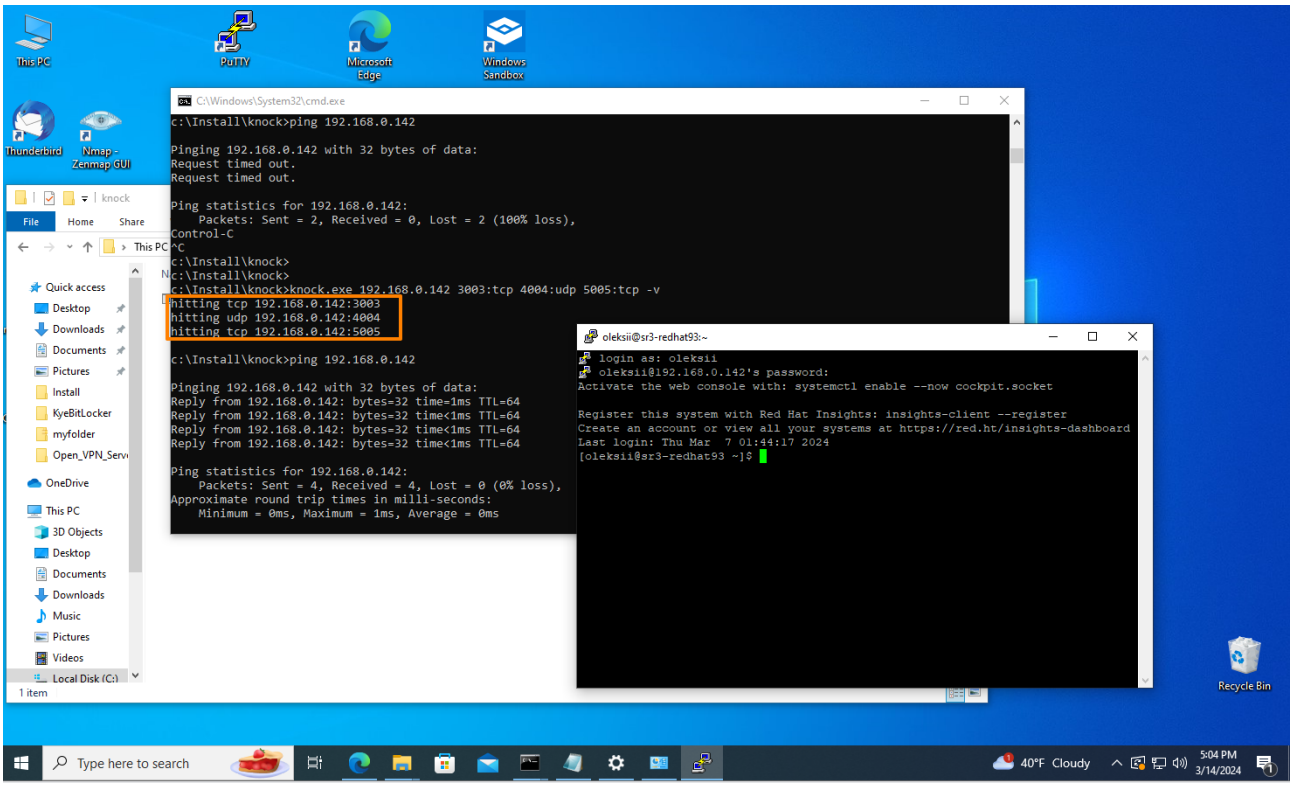

Рисунок 3.4 – Відкриття доступу по SSH до хоста RedHat Linux

Порт TCP 22 був відкритий за допомогою техніки port knocking та здійснено підключитися до сервера за допомогою SSH та клієнта PuTTY.

3.3 Тестування техніки port knocking в Ubuntu Server

Щоб протестувати техніку port knocking на Ubuntu Server використаємо утиліту knock в MacOS для надсилання послідовності звернень на певні порти та клієнт для підключення через SSH.

На рисунку 3.5 показано мережеві налаштування операційної системи Ubuntu Server.

```
root@sr2-ubuntu220404:/etc# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:29:20:e1 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.0.141/24 brd 192.168.0.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe29:20e1/64 scope link
       valid_lft forever preferred_lft forever
root@sr2-ubuntu220404:/etc#
```

Рисунок 3.6 – Мережеві налаштування операційної системи Ubuntu Server

Згідно налаштувань брандмауера UFW блокуються будь-які з’єднання до порту SSH в Ubuntu Server (див.рисунок 3.7)

```
macos1251ws:~ root#
macos1251ws:~ root# ifconfig en0
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=40b<RXCSUM,TXCSUM,VLAN_HWTAGGING,CHANNEL_IO>
        ether 00:0c:29:d1:69:6c
        inet6 fe80::fc:6d7b:8ad:4221%en0 prefixlen 64 secured scopeid 0x4
        inet 192.168.0.113 netmask 0xffffff00 broadcast 192.168.0.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect (1000baseT <full-duplex>)
        status: active
macos1251ws:~ root# ssh oleksii@192.168.0.141
ssh: connect to host 192.168.0.141 port 22: Operation timed out
macos1251ws:~ root#
```

Рисунок 3.7 – Перевірка доступності SSH на хості Ubuntu Server

Спроба з'єднатися з IP-адресою 192.168.0.141 через SSH не булу успішною. На це вказує повідомлення про помилку "ssh: connect to host 192.168.0.141 port 22: Operation timed out".

На рисунку 3.8 показано вивід утиліта knock, яка відправила послідовність пакетів на вказані порти сервера з IP-адресою 192.168.0.141 з операційної системи MacOS.

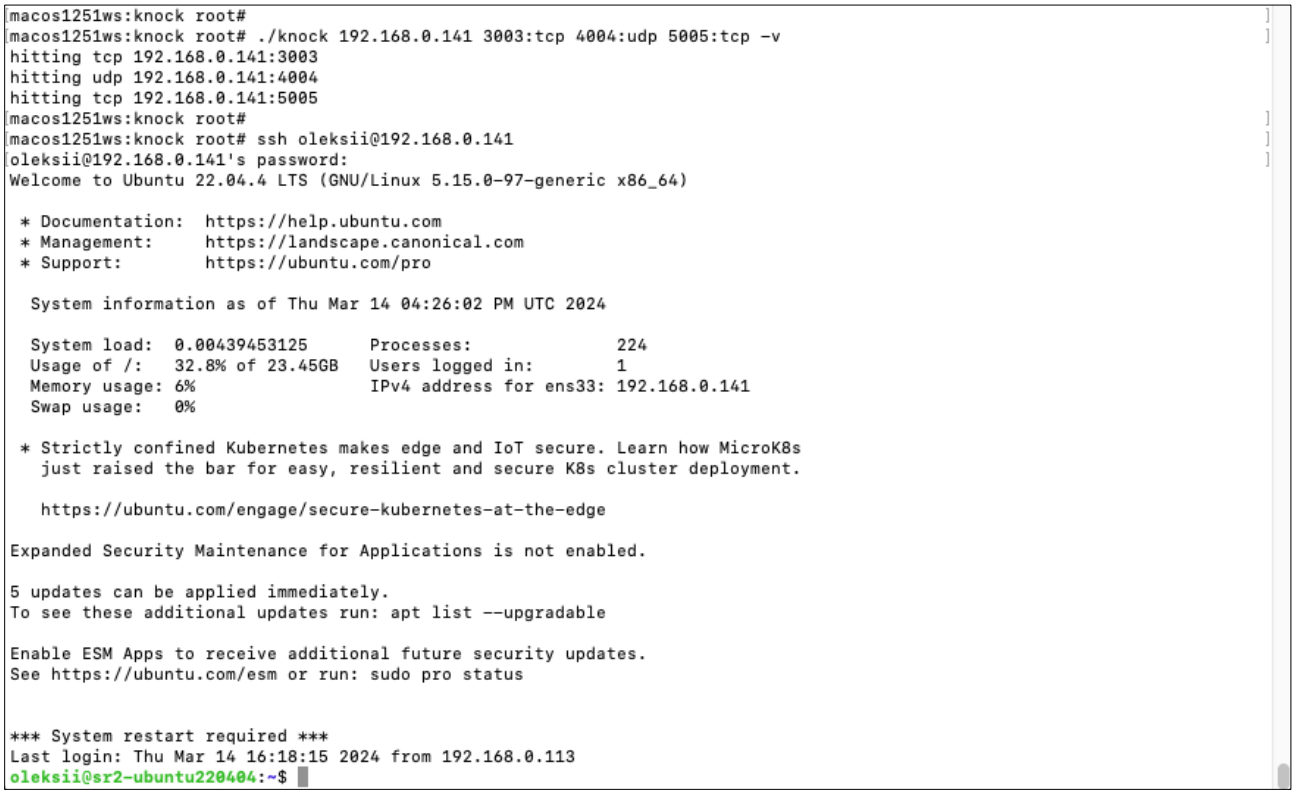

Рисунок 3.8 – Відкриття доступу по SSH до хоста Ubuntu Server

Порт TCP 22 був відкритий за допомогою техніки port knocking та здійснено підключитися до сервера за допомогою клієнта SSH.

На рисунку 3.9 показано вивід команду `ufw status numbered` в операційній системі Ubuntu Server.

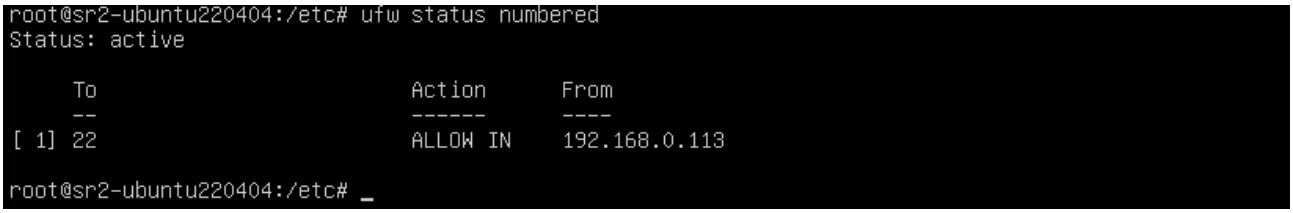

Рисунок 3.9 – Активні правила брандмауера UFW в Ubuntu Server

Вивід команди показує одне активне правило для брандмауера з номером 1, яке дозволяє вхідні з'єднання (ALLOW IN) на порт 22 з хоста з IP-адреси 192.168.0.113. Це правили створилось сервісом knockd в результаті опрацювання коректної послідовності пакетів на вказані порти сервера.

3.4 Тестування техніки port knocking в FreeBSD

Щоб протестувати техніку port knocking в FreeBSD використаємо утиліту knock в Oracle Linux для надсилання послідовності звернень на певні порти та клієнт SSH для підключення до сервісу віддаленого керування.

Файл rc.conf в операційній системі FreeBSD є основним конфігураційним файлом, який містить параметри запуску служб та різноманітні налаштування для системи включно з налаштуваннями мережі. На рисунку 3.10 показано вміст файлу rc.conf.

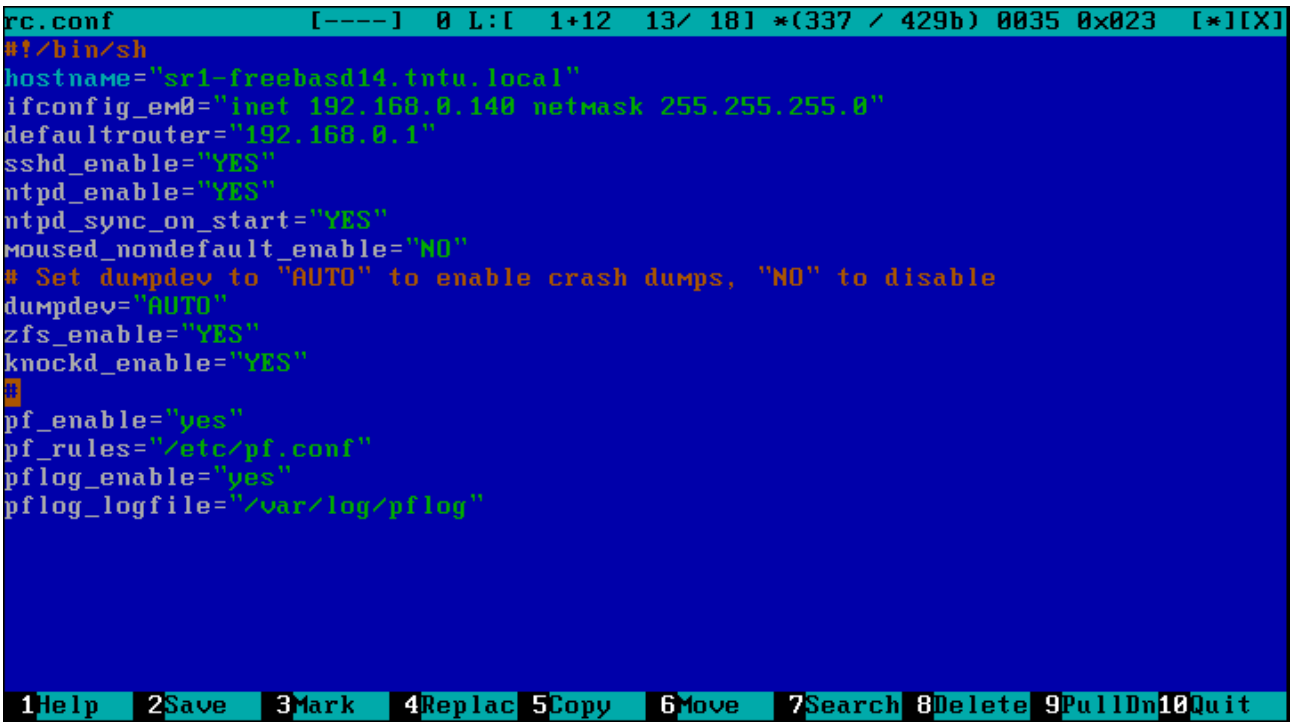

Рисунок 3.10 – Параметри налаштування операційної системи FreeBSD

Конфігураційні параметри цього файлу мають наступне значення:

1) `hostname="sr1-freebsd14.tntu.local"` – встановлення імені хоста;

2) `ifconfig_em0="inet 192.168.0.140 netmask 255.255.255.0"` – налаштування мережевого інтерфейсу em0 зі статичною IP-адресою;

3) `defaultrouter="192.168.0.1"` – встановлення шлюзу за замовчуванням;

4) `sshd_enable="YES"` – активація SSH демона при запуску системи;

5) `knockd_enable="YES"` – активація демона knockd при запуску системи;

6) `pf_enable="yes"` – активація PF при запуску системи;

7) `pf_rules="/etc/pf.conf"` – шлях до файлу конфігурації PF;

8) `pflog_enable="yes"` – активація логування для PF;

9) `pflog_logfile="/var/log/pflog"` – шлях до файлу логів PF.

Згідно налаштувань брандмауера PF будь-які вхідні з'єднання до FreeBSD (див. рисунок 2.8) заборонені включно з протоколом ICMP (див.рисунок 3.11)

```
[root@oraclelinux92kvm /]#
[root@oraclelinux92kvm /]# ifconfig br0
br0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.103  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::474:2515:d7cb:6810  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:aa:23:66  txqueuelen 1000  (Ethernet)
        RX packets 31591  bytes 89695097 (85.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15041  bytes 1036535 (1012.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@oraclelinux92kvm /]# ping 192.168.0.140
PING 192.168.0.140 (192.168.0.140) 56(84) bytes of data.
^C
--- 192.168.0.140 ping statistics ---
35 packets transmitted, 0 received, 100% packet loss, time 34830ms

[root@oraclelinux92kvm /]# ssh admin@192.168.0.140

^C
[root@oraclelinux92kvm /]#
```

Рисунок 3.11 – Перевірка доступності хоста FreeBSD з Oracle Linux

Було використано команду ping 192.168.0.140 для тестування з'єднання з хостом з IP-адресою 192.168.0.140. Результати виконання команди ping показують, що усі 35 пакетів втрачено (100% втрат). Це свідчить про проблему з мережевим з'єднанням з хостом 192.168.0.140. Спроба підключення по SSH до хоста FreeBSD також була невдалою.

На рисунку 3.12 показано вивід утиліта knock, яка відправила послідовність пакетів на вказані порти сервера з IP-адресою 192.168.0.140 з операційної системи Oracle Linux.

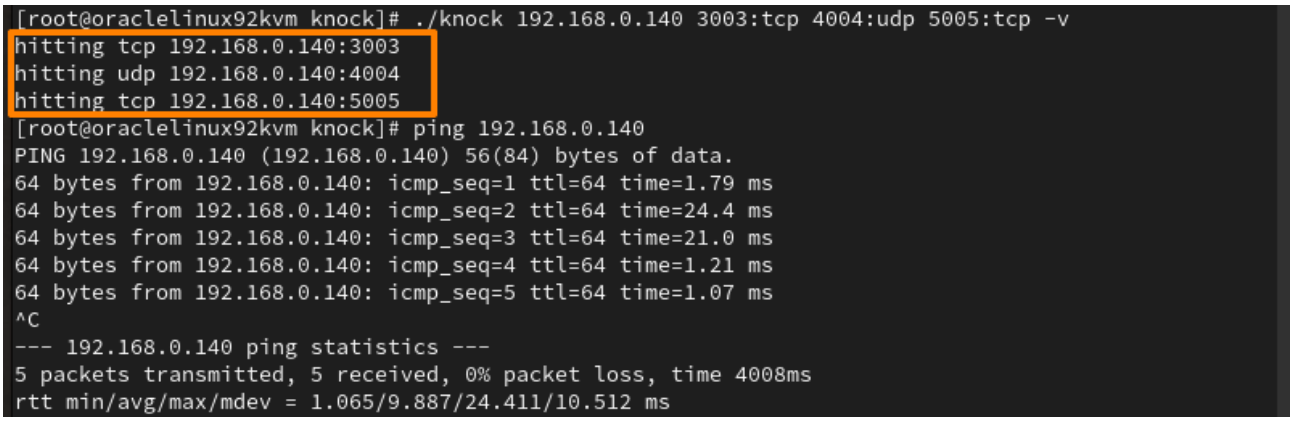

Рисунок 3.12 – Відкриття доступу по SSH до хоста FreeBSD з Oracle Linux

Також було виконано команду ping 192.168.0.140, щоб перевірити зв'язок з FreeBSD після виконання команди knock. На рисунку 3.13 показано вікно терміналу з успішним входом з Oracle Linux в систему FreeBSD через SSH.

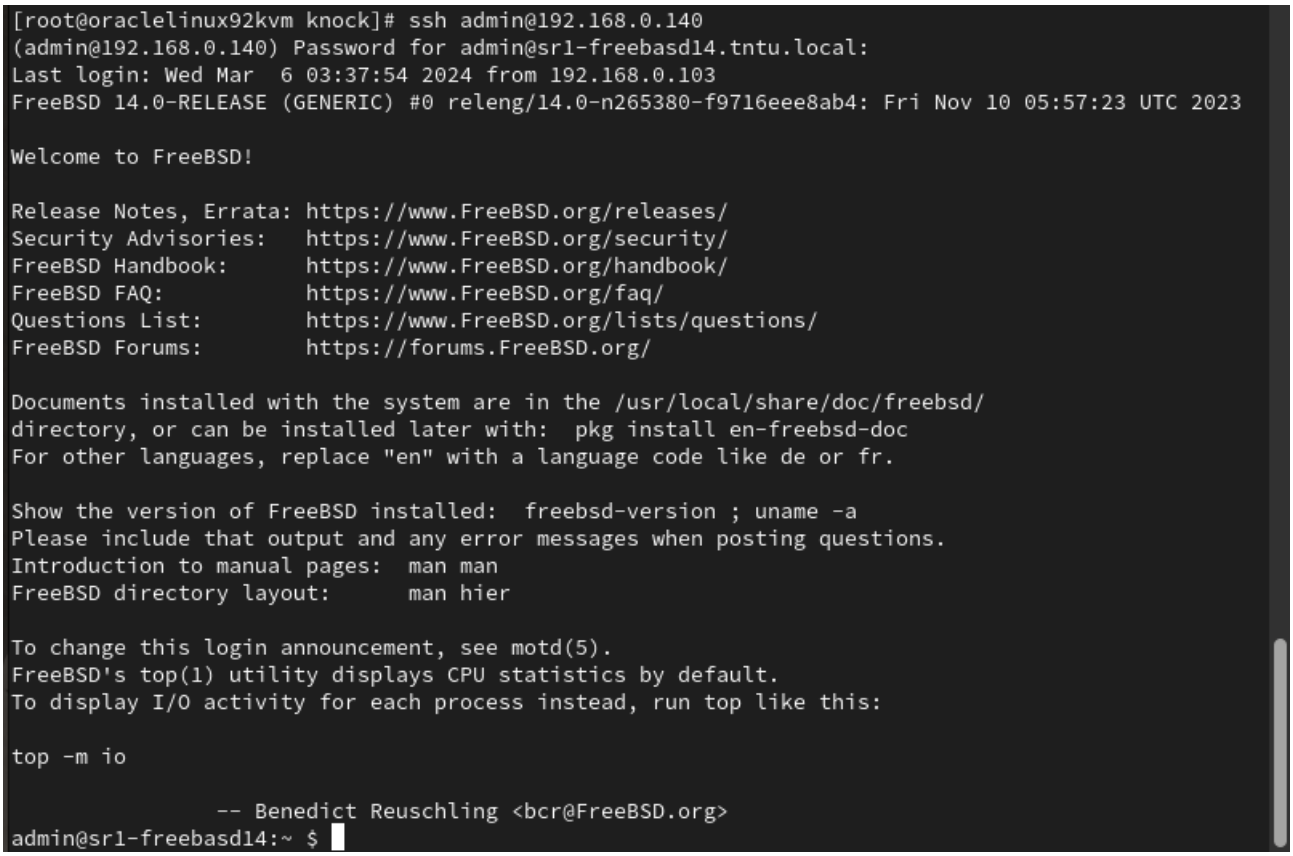

Рисунок 3.13 – Вхід по SSH до хоста FreeBSD з Oracle Linux

На рисунку 3.14 показано вивід вмісту таблиці `knockd_pass` брандмауера PF.

```
root@sr1-freebasd14:/etc#
root@sr1-freebasd14:/etc# pfctl -t knockd_pass -T show
   192.168.0.103
root@sr1-freebasd14:/etc#
```

Рисунок 3.14 – Вивід вмісту таблиці `knockd_pass` брандмауера PF в FreeBSD

Команда `pfctl -t knockd_pass -T show` використовується для перегляду поточного вмісту таблиці `knockd_pass` в брандмауері PF операційної системи FreeBSD [16]. У таблиці `knockd_pass` зберігаються IP-адреси з яких булу надіслано коректні послідовності звернень на певні порти для динамічного відкривання порту SSH.

У нашому випадку в таблиці `knockd_pass` вказана лише одна IP-адреса - 192.168.0.103. Це IP-адреса Oracle Linux. Правила брандмауера (див.рисунок 2.8), які використовують цю таблицю, дозволяють з'єднання з цієї IP-адреси по SSH до FreeBSD.

На рисунку 3.15 показано вікно терміналу FreeBSD, де в режимі реального часу відстежуються останні записи в журналі `knockd.log` за допомогою команди `tail -f`.

```
root@sr1-freebasd14:/var/log# tail -f knockd.log

[2024-03-06 03:42] 192.168.0.103: opencloseSSH: Stage 1
[2024-03-06 03:42] 192.168.0.103: opencloseSSH: Stage 2
[2024-03-06 03:42] 192.168.0.103: opencloseSSH: Stage 3
[2024-03-06 03:42] 192.168.0.103: opencloseSSH: OPEN SESAME
[2024-03-06 03:42] opencloseSSH: running command: /sbin/pfctl -t knockd_pass  -T
 add 192.168.0.103

[2024-03-06 03:43] 192.168.0.103: opencloseSSH: command timeout
[2024-03-06 03:43] opencloseSSH: running command: /sbin/pfctl -t knockd_pass -T
delete 192.168.0.103

```

Рисунок 3.15 – Вивід записів журналу `knockd.log` в FreeBSD

Стрічки цього виводу мають наступне значення:

1) `[2024-03-06 03:42] 192.168.0.103: opencloseSSH: Stage 1` - з IP-адреси 192.168.0.103 надійшов запит на порт 3003 TCP, що відповідає першому етапу (Stage 1) процесу port knocking;

2) `[2024-03-06 03:42] 192.168.0.103: opencloseSSH: Stage 2` - з цієї ж IP-адреси надійшов запит на наступний порт 4004 UDP, що відповідає другому етапу (Stage 2) процесу port knocking;

3) `[2024-03-06 03:42] 192.168.0.103: opencloseSSH: Stage 3` - запит на порт 5005 TCP, що відповідає третьому етапу (Stage 3) процесу port knocking було отримано від тієї самої IP-адреси;

4) `[2024-03-06 03:42] 192.168.0.103: opencloseSSH: OPEN SESAME` - відбувся успішний процес port knocking;

5) `[2024-03-06 03:42] opencloseSSH: running command: /sbin/pfctl -t knockd_pass -T add 192.168.0.103` - система виконала команду для додавання IP-адреси 192.168.0.103 до таблиці `knockd_pass` брандмауера PF, що дозволило цій IP-адресі з'єднання по SSH з FreeBSD;

6) `[2024-03-06 03:43] 192.168.0.103: opencloseSSH: command timeout` – термін перебування IP-адреси 192.168.0.103 в таблиці `knockd_pass` брандмауера PF вийшов;

7) `[2024-03-06 03:43] opencloseSSH: running command: /sbin/pfctl -t knockd_pass -T delete 192.168.0.103` - відповідно до таймауту, система виконала команду для видалення IP-адреси 192.168.0.103 з таблиці `knockd_pass`, скасувавши попередньо наданий доступ.

Файл журналу показує, що port knocking ініційований з IP-адреси Oracle Linux був успішним.

3.5 Висновки до розділу

В третьому розділі було встановлено та налаштовано операційні системи Windows 10, Oracle Linux та MacOS для використання в якості клієнтів port knocking. Було проведено тестування механізму port knocking налаштованого в RedHat Linux з використанням брандмауера iptables. Тестування проводилось з

операційної системи Windows 10 з використанням утиліти knock.exe з надсилання послідовності звернень на порти 3003 TCP, 4004 UDP ТА 5005 TCP та PuTTY для підключення через SSH.

Також було проведено тестування механізму port knocking налаштованого в Ubuntu Server з використанням брандмауера UFW та knockd сервісу. Тестування проводилось з операційної системи MacOS з використанням утиліти knock для надсилання послідовності звернень на порти ідентичні як і при тестування в RedHat Linux та клієнта SSH для віддаленого адміністрування. В операційні системі FreeBSD також було проведено тестування механізму port knocking налаштованого з використанням брандмауера PF та knockd сервісу. Тестування проводилось з операційної системи Oracle Linux з використанням утиліти knock для надсилання послідовності звернень на визначені порти та клієнта SSH для підключення до сервісу віддаленого керування.

Тести підтвердили коректність роботи зазначених сервісів на налаштувань. У всіх випадках порт 22 TCP був відкритий за допомогою техніки port knocking та здійснено підключитися до сервера за допомогою SSH.

Отже, port knocking є ефективним інструментом для підвищення безпеки серверів, оскільки він дозволяє приховувати сервіси за закритими портами і відкривати їх лише відповідно до надісланої певної послідовності пакетів.

РОЗДІЛ 4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Долікарська допомога при термічних опіках

Долікарська допомога при термічних опіках є важливим кроком у зменшенні наслідків та запобіганню подальших ускладнень. Термічні опіки відбуваються, коли шкіра або м’які тканини пошкоджуються високими температурами, які можуть виникнути в результаті вогню, гарячих речовин або пари, сонячного випромінювання або інших джерел тепла. Опіки можуть бути серйозними і потребують негайного медичного втручання.

При наданні домедичної допомоги враховувати ступінь пошкодження шкіри та м’яких тканин:

1) I ступінь (еритема) - почервоніння шкіри, набряклість і біль.

2) II ступінь (утворення пухирів) - сильний біль із інтенсивним почервонінням, відшаруванням епідермісу з утворенням міхурів, наповнених рідиною.

3) III ступінь: пошкодження всієї товщі шкіри з утворенням щільного струпу, під яким перебувають ушкоджені тканини.

4) IV ступінь (обвуглення): пошкодження всієї товщі шкіри з ушкодження м’язів, сухожиль, кісток.

Наказ Міністерства охорони здоров'я України від 09.03.2022 р. № 441 " Про затвердження порядків надання домедичної допомоги особам при невідкладних станах" встановлює порядки надання домедичної допомоги постраждалим при термічних опіках. У цьому порядку термін «термічний опік» вживається у такому значенні - це невідкладний стан, спричинений дією високих температур, в результаті чого виникає пошкодження шкіри та м’яких тканин. [17].

Надання домедичної допомоги постраждалим при термічних опіках передбачає такі кроки:

1) Переконайтеся у відсутності небезпеки для себе, оточуючих та постраждалого.

2) Припиніть дію високої температури на постраждалого та, при необхідності, зніміть тліючий одяг.
3) Зніміть прикраси, які можуть бути на ділянці опіку.
4) Заспокойте постраждалого та поясніть йому свої подальші дії.
5) Зателефонуйте на екстрений номер, щоб здійснити виклик екстреної медичної допомоги, та слухайте вказівки диспетчера.
6) Охолодіть місце опіку протягом щонайменше 20 хвилин, промиваючи його водою кімнатної температури. Це важливо, якщо площа опіку не перевищує 20% у дорослих або 10% у дітей.
7) Після охолодження накладіть на місце опіку чисту, стерильну суху марлеву пов'язку. Пов'язка не повинна надмірно тиснути на м'які тканини.
8) У разі наявності міхурів не пошкоджуйте їх. Якщо випадково міхур пошкоджено, накладіть пов'язки, як описано вище.
9) Якщо опіки охоплюють більше ніж 20% площі тіла у дорослих або 10% у дітей, накрийте постраждалого термопокривалом або покривалом;
10) Забезпечте постійний нагляд за постраждалим до прибуття бригади швидкої медичної допомоги.
11) У разі погіршення стану постраждалого до прибуття бригади швидкої медичної допомоги повторно зателефонуйте диспетчеру.
12) Якщо можливо, зберіть якомога більше інформації про обставини отримання травми від постраждалого або свідків. Передайте цю інформацію фахівцям бригади швидкої медичної допомоги або диспетчеру.

Це загальна послідовність дій, яку слід виконати, але завжди важливо дотримуватись інструкцій медичних фахівців та адаптувати допомогу до конкретної ситуації. Виконання цих кроків допоможе забезпечити постраждалому першу необхідну допомогу та зберегти його життя до прибуття медичних фахівців.

4.2 Техніка безпеки при роботі з ПК

До самостійної роботи на комп’ютерах допускаються особи, які пройшли медичний огляд, навчання по професії, вступний інструктаж з охорони праці та первинний інструктаж з охорони праці на робочому місці. В подальшому вони проходять повторні інструктажі з охорони праці на робочому місці один раз на півріччя, періодичні медичні огляди один раз на два роки.

Під час роботи на комп’ютерах можуть діяти такі небезпечні та шкідливі фактори, як:

- фізичні;
- психофізіологічні.

Основним обладнанням робочого місця користувача комп’ютера є монітор, системний блок та клавіатура, мишка.

Робочі місця мають бути розташовані на відстані не менше 1,5 м від стіни з вікнами, від інших стін на відстані 1м, між собою на відстані не менше 1,5 м. Відносно вікон робоче місце доцільно розташовувати таким чином, щоб природне світло падало на нього збоку, переважно зліва.

Робочі місця слід розташовувати так, щоб уникнути попадання в очі прямого світла. Джерела освітлення рекомендується розташовувати з обох боків екрану паралельно напрямку погляду. Для уникнення світлових відблисків екрану, клавіатури в напрямку очей користувача, від світильників загального освітлення або сонячних променів, необхідно використовувати анти полискові сітки, спеціальні фільтри для екранів, захисні козирки, на вікнах – жалюзі.

Монітор повинен бути розташований на робочому місці так, щоб поверхня екрана знаходилася в центрі поля зору на відстані 400-700 мм від очей користувача. Рекомендується розміщувати елементи робочого місця так, щоб витримувалася однакова відстань очей від екрана, клавіатури, тексту.

Зручна робоча поза при роботі з комп’ютером забезпечується регулюванням висоти робочого столу, крісла та підставки для ніг. Раціональною робочою позою може вважатися таке положення, при якому ступні працівника розташовані горизонтально на підлозі або підставці для ніг, стегна зорієнтовані

у горизонтальній площині, верхні частини рук – вертикальні. Кут ліктєвого суглоба коливається в межах 70-90°, зап'ястя зігнуті під кутом не більше ніж 20°, нахил голови 15-20°.

Для нейтралізації зарядів статичної електрики в приміщенні, де виконується робота на комп'ютерах, в тому числі на лазерних та світлодіодних принтерах, рекомендується збільшувати вологість повітря за допомогою кімнатних зволожувачів. Не рекомендується носити одяг з синтетичних матеріалів.

Згідно статті 18 Закону України "Про охорону праці" працівник зобов'язаний:

- знати і виконувати вимоги нормативних актів про охорону праці, правила поводження з устаткуванням та іншими засобами виробництва, користуватися засобами колективного та індивідуального захисту;
- дотримуватись зобов'язань щодо охорони праці, передбачених колективним договором та правилами внутрішнього трудового розпорядку підприємства;
- співробітничати з власником у справі організації безпечних і нешкідливих умов праці, особисто вживати посильних заходів щодо усунення будь-якої виробничої ситуації, яка створює загрозу його життю чи здоров'ю, або людей, які його оточують, повідомляти про небезпеку свого безпосереднього керівника або іншу посадову особу.

Вимоги безпеки перед початком роботи:

- увімкнути систему кондиціювання в приміщенні;
- перевірити надійність встановлення апаратури на робочому столі. Повернути монітор так, щоб було зручно дивитися на екран – під прямим кутом (а не збоку) і трохи зверху вниз, при цьому екран має бути трохи нахиленим, нижній його край ближче до оператора;
- перевірити загальний стан апаратури, перевірити справність електропроводки, з'єднувальних шнурів, штепсельних вилок, розеток, заземлення захисного екрана;
- відрегулювати освітленість робочого місця;

- відрегулювати та зафіксувати висоту крісла, зручний для користувача нахил його спинки;
- приєднати до системного блоку необхідну апаратуру. Усі кабелі, що з'єднують системний блок з іншими пристроями, слід вставляти та виймати при вимкненому комп'ютері;
- ввімкнути апаратуру комп'ютера вимикачами на корпусах в послідовності: монітор, системний блок, принтер (якщо передбачається друкування);
- відрегулювати яскравість свічення монітора, мінімальний розмір світної точки, фокусування, контрастність. Не слід робити зображення надто яскравим, щоб не втомлювати очей.

Рекомендується:

- яскравість свічення екрана – не менше 100Kg/м2;
- відношення яскравості монітора до яскравості оточуючих його поверхонь в робочій зоні – не більше 3:1;
- мінімальний розмір точки свічення не більше 0,4 мм для монохромного монітора і не менше 0,6 мм для кольорового, контрастність зображення знаку – не менше 0,8.

При виявленні будь-яких несправностей роботу не розпочинати, повідомити про це керівника.

Вимоги безпеки під час виконання роботи:

- необхідно стійко розташовувати клавіатуру на робочому столі, не опускати її хитання. Під час роботи на клавіатурі сидіти прямо, не напружуватися;
- для забезпечення несприятливого впливу на користувача пристроїв типу "миша" належить забезпечувати вільну велику поверхню столу для переміщення "миші" і зручного упору ліктьового суглоба;
- не дозволяються сторонні розмови, подразнюючі шуми;
- періодично при вимкненому комп'ютері прибирати ледь змоченою мильним розчином бавовняною ганчіркою порох з поверхонь апаратури. Екран протирають ганчіркою, змоченою у спирті. Не дозволяється

використовувати рідинні або аерозольні засоби чищення поверхонь комп’ютера.

Психофізіологічне розвантаження є одним з варіантів зменшення стресу.

Дана практика включає в себе застосування методу аутогенного тренування, що передбачає свідоме використання комплексу прийомів психічної саморегуляції та виконання простих фізичних вправ зі словесним самонавіюванням. Основна увага приділяється розслабленню м'язів (релаксації).

Під час сеансів психофізіологічного розвантаження рекомендується використовувати три періоди, що відповідають фазам відновлення:

Перший період - абстрагування від виробничого середовища, що відповідає фазі залишкового збудження. В цей час відтворюється повільна мелодійна музика та звуки пташиного співу. Працівники знаходять зручну позу та психологічно готуються до наступних періодів.

Другий період - заспокоєння, що відповідає фазі відновлювального гальмування. Показуються фотослайди з зображеннями природи, таких як квітучі луки, березові гаї, ставки і т.д. Звуковий супровід включає спокійну музику та заспокійливі формули аутогенного тренування.

Третій період - активізація, що відповідає фазі підвищеної збудженості. Спочатку світло повністю вимикається, а потім на екрані появляється червона пляма, розмір і яскравість якої поступово збільшуються. В кінці періоду звучить бадьора музика, а працівники виконують мобілізуючі формули аутогенного тренування, попередньо зробивши глибоке вдихання та видихання.

Сеанси психофізіологічного розвантаження можуть проводитись за єдиною програмою через індивідуальні навушники і складатись із двох періодів по 5 хвилин кожний: повне розслаблення та активізація працездатності. При необхідності, на фоні музики можуть використовуватись фрази, що сприяють відпочинку, покращенню самопочуття та бадьорості на заключному етапі. Після сеансів психофізіологічного розвантаження працівники відчувають зменшення втоми, з'являється бадьорість та гарний настрій, а загальний стан помітно поліпшується.

Додатково до сеансів психофізіологічного розвантаження, працівники також можуть скористатись іншими методами для зниження стресу та покращення психологічного благополуччя.

Одним з ефективних підходів є впровадження регулярних перерв під час робочого дня. Це можуть бути короткі паузи, під час яких працівники займаються розслаблюючими вправами, дихальними техніками або просто відпочивають. Це допомагає знизити напругу і покращити фокусування під час робочого процесу.

Також важливо створити комфортне робоче середовище для працівників. Це може включати забезпечення комфортних стільців і столів, добре освітлення та достатню вентиляцію. Природне освітлення та наявність рослин у приміщенні також можуть позитивно вплинути на настрій та самопочуття працівників.

Підтримка від керівництва та колег також має важливе значення. Створення сприятливого та підтримуючого робочого середовища, де працівники можуть відчути підтримку та співпрацю, сприяє зниженню стресу та покращує загальний настрій в колективі.

Крім того, особисте самоуправління і здібність до саморегуляції є важливими навичками для працівників. Це включає уміння регулювати власні емоції, реагувати на стресові ситуації та знаходити способи їх подолання, наприклад, за допомогою медитації, йоги або інших релаксаційних технік.

Усі ці підходи сприяють створенню здорової та продуктивної робочої атмосфери, де працівники можуть ефективно керувати стресом, забезпечуючи своє фізичне та емоційне благополуччя.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи бакалавра було розглянуто та реалізовано механізм динамічного відкривання мережевих портів з метою забезпечення безпеки корпоративних серверів.

У першому розділі було проаналізовано методи захисту механізмів адміністрування серверів, такі як використання VPN, брандмауери та ключів SSH. Зазначено, що хоча ці методи надійні, вони також мають свої недоліки, такі як можливі обмеження щодо використання VPN або потенційна можливість DOS або DDOS атак на відкриті порти адміністрування.

Також було детально описано техніку динамічного відкривання портів (port knocking), де клієнт відправляє певну послідовність запитів до закритих портів, що спричиняє автоматичне відкриття потрібного порту на сервері. Розглянуто різні методи реалізації port knocking у різних операційних системах та мережевому обладнанні.

Висвітлено потенційні загрози безпеці при використанні техніки port knocking, зокрема прослуховування пакетів зловмисниками для виявлення використання цієї техніки та відкриття доступу до сервера з NAT IP-адрес.

У другому розділі було представлено загальний алгоритм роботи техніки port knocking для надання доступу до сервісів віддаленого адміністрування, таких як SSH, RDP, WinBOX та інші.

Було встановлено та налаштовано сервери з різними операційними системами та брандмауерами для реалізації техніки port knocking. В Red Hat Enterprise Linux було використано брандмаур iptables для цієї мети, де скрипт з командами iptables налаштовував правила брандмауера так, щоб доступ до порту 22 був можливий лише після послідовного коректного надсилання пакетів на певні порти.

Також було налаштовано Ubuntu Server з використанням брандмауера UFW та реалізовано техніку port knocking за допомогою сервісу knockd. Для FreeBSD використовувався брандмауер PF з використанням таблиць та стану з’єднань та здійснена реалізація port knocking за допомогою knockd.

У третьому розділі було встановлено та налаштовано операційні системи Windows 10, Oracle Linux та MacOS для використання в якості клієнтів для тестування механізму port knocking в RedHat Linux з брандмауером iptables, Ubuntu Server з брандмауером ufw та knockd сервісом, а також в FreeBSD з брандмауером PF та knockd сервісом.

У процесі тестування механізму port knocking на RedHat Linux була використана операційна система Windows 10 для відправлення послідовності звернень на певні порти за допомогою утиліти knock.exe, а також клієнт SSH PuTTY для підключення до сервера через SSH після відкриття потрібного порту.

Аналогічно, на Ubuntu Server та FreeBSD було використано операційні системи MacOS та Oracle Linux як клієнти для надсилання послідовності звернень на визначені порти за допомогою утиліти knock та подальше віддалене адміністрування через SSH.

Тести підтвердили правильну роботу встановлених сервісів та налаштувань. У всіх випадках техніка port knocking була успішно застосована для відкриття порту 22 TCP, що дозволило підключитися до сервера за допомогою SSH. Ці тести допомогли перевірити ефективність та працездатність механізму port knocking в різних операційних системах та підтвердили можливості забезпечення додаткового рівня безпеки для віддаленого адміністрування серверів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Tymoshchuk, V., Karnaukhov, A., & Tymoshchuk, D. (2024). USING VPN TECHNOLOGY TO CREATE SECURE CORPORATE NETWORKS. Collection of scientific papers «ΛΌГОΣ», (June 21, 2024; Seoul, South Korea), 166-170. https://doi.org/10.36074/logos-21.06.2024.034

2. What Is a Firewall? URL: https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html (дата звернення: 16.03.2024).

3. Демчук, В., Тимощук, В., & Тимощук, Д. (2023). ЗАСОБИ МІНІМІЗАЦІЇ ВПЛИВУ SYN FLOOD АТАК. Collection of scientific papers «SCIENTIA», (November 24, 2023; Kraków, Poland), 130-130.

4. Іваночко, Н., Тимощук, В., Букатка, С., & Тимощук, Д. (2023). РОЗРОБКА ТА ВПРОВАДЖЕННЯ ЗАХОДІВ ЗАХИСТУ ВІД UDP FLOOD АТАК НА DNS СЕРВЕР. Матеріали конференцій МНЛ, (3 листопада 2023 р., м. Вінниця), 177-178.

5. Бекер, І., Тимощук, В., Маслянка, Т., & Тимощук, Д. (2023). МЕТОДИКА ЗАХИСТУ ВІД ПОВІЛЬНИХ ТА ШВИДКИХ BRUTE-FORCE АТАК НА IMAP СЕРВЕР. Матеріали конференцій МНЛ, (17 листопада 2023 р., м. Львів), 275-276.

6. Basic overview of SSH Keys URL: https://www.ssh.com/academy/ssh-keys (дата звернення: 16.03.2024).

7. Port knocking. URL: https://nordvpn.com/uk/cybersecurity/glossary/port-knocking/ (дата звернення: 16.03.2024).

8. Тимощук, В., Долінський, А., & Тимощук, Д. (2024). ВИКОРИСТАННЯ ТЕХНІКИ ДИНАМІЧНОГО ВІДКРИВАННЯ МЕРЕЖЕВИХ ПОРТІВ ДЛЯ ПІДВИЩЕННЯ БЕЗПЕКИ СЕРВЕРІВ. Collection of scientific papers «ΛΌГОΣ», (May 24, 2024; Zurich, Switzerland), 233-234.

9. Red Hat Enterprise Linux. URL: https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux. (дата звернення: 16.03.2024).

10. Netfilter/iptables project. URL: https://netfilter.org/documentation/ (дата звернення: 16.03.2024).

11. Ванца, В., Тимощук, В., Стебельський, М., & Тимощук, Д. (2023). МЕТОДИ МІНІМІЗАЦІЇ ВПЛИВУ SLOWLORIS АТАК НА ВЕБСЕРВЕР. Матеріали конференцій МЦНД, (03.11. 2023; Суми, Україна), 119-120.

12. UFW - Uncomplicated Firewall. URL: https://help.ubuntu.com/community/UFW (дата звернення: 16.03.2024).

13. Knockd - Linux man page. URL: https://linux.die.net/man/1/knockd (дата звернення: 16.03.2024).

14. ZAGORODNA, N., STADNYK, M., LYPA, B., GAVRYLOV, M., & KOZAK, R. (2022). Network Attack Detection Using Machine Learning Methods. Challenges to national defence in contemporary geopolitical situation, 2022(1), 55-61.

15. Тимощук, В., Долінський, А., & Тимощук, Д. (2024). СИСТЕМА ЗМЕНШЕННЯ ВПЛИВУ DOS-АТАК НА ОСНОВІ MIKROTIK. Матеріали конференцій МЦНД, (17.05. 2024; Ужгород, Україна), 198-200. https://doi.org/10.62731/mcnd-17.05.2024.008

16. FreeBSD Port-Knocking. URL: https://in4bsd.com/notes/freebsd-port-knocking/ (дата звернення: 16.03.2024).

17. Про затвердження порядків надання домедичної допомоги особам при невідкладних станах. URL: https://zakon.rada.gov.ua/laws/show/z0356-22#n769 (дата звернення: 16.03.2024).

ДОДАТКИ

Додаток А Файл вихідного коду knock.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <sys/types.h>

#ifdef __WIN32__
# include <winsock2.h>
# include <ws2tcpip.h>
#else
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <resolv.h>
# include <sys/socket.h>
#endif

#if defined(__FreeBSD__) || defined(__APPLE__)
#include <netinet/in.h>
#endif
#include <unistd.h>
#include <string.h>
#include <getopt.h>
#include <fcntl.h>

static char version[] = "0.7";

#define PROTO_TCP 1
#define PROTO_UDP 2

/* function prototypes */
void vprint(char *fmt, ...);
void ver();
void usage();

int o_verbose = 0;
int o_udp     = 0;
int o_delay   = 0;

int main(int argc, char** argv)
{
    int sd;
    struct sockaddr_in addr;
    int opt, optidx = 1;
    static struct option opts[] =
    {
        {"verbose",   no_argument,       0, 'v'},
        {"udp",       no_argument,       0, 'u'},
        {"delay",     required_argument, 0, 'd'},
        {"help",      no_argument,       0, 'h'},
        {"version",   no_argument,       0, 'V'},
        {0, 0, 0, 0}
    };

#ifdef __WIN32__
    WSADATA wsaData;
    if (WSAStartup(MAKEWORD(2,2), &wsaData) == SOCKET_ERROR) {
        fprintf (stderr, "Error initialising WSA.\n");
        return -1;
    }
#endif
```

```
while((opt = getopt_long(argc, argv, "vud:hV", opts, &optidx))) {
    if(opt < 0) {
        break;
    }
    switch(opt) {
        case 0:   break;
        case 'v': o_verbose = 1; break;
        case 'u': o_udp = 1; break;
        case 'd': o_delay = (int)atoi(optarg); break;
        case 'V': ver();
        case 'h': /* fallthrough */
        default: usage();
    }
}
if((argc - optind) < 2) {
    usage();
}

if(o_delay < 0) {
    fprintf(stderr, "error: delay cannot be negative\n");
    exit(1);
}

struct addrinfo hints, *res;
memset(&hints, 0, sizeof hints);
hints.ai_family = AF_INET;
hints.ai_socktype = SOCK_RAW;
if (getaddrinfo(argv[optind++], NULL, &hints, &res)) {
  fprintf(stderr, "Cannot resolve hostname [%s]\n", argv[optind-1]);
  exit(1);
}
for(; optind < argc; optind++) {
    unsigned short port, proto = PROTO_TCP;
    char *ptr, *arg = strdup(argv[optind]);

    if((ptr = strchr(arg, ':'))) {
        *ptr = '\0';
        port = atoi(arg);
        arg = ++ptr;
        if(!strcmp(arg, "udp")) {
            proto = PROTO_UDP;
        } else {
            proto = PROTO_TCP;
        }
    } else {
        port = atoi(arg);
    }

    if(o_udp || proto == PROTO_UDP) {
        sd = socket(PF_INET, SOCK_DGRAM, 0);
        if(sd == -1) {
            fprintf(stderr, "Cannot open UDP socket\n");
            exit(1);
        }
    } else {
```

```
#ifdef __WIN32__
            u_long flags;
#else
            int flags;
#endif
            sd = socket(PF_INET, SOCK_STREAM, 0);
            if(sd == -1) {
                fprintf(stderr, "Cannot open TCP socket\n");
                exit(1);
            }
#ifdef __WIN32__
            flags = 1;
            ioctlsocket(sd, FIONBIO, &flags);
#else
            flags = fcntl(sd, F_GETFL, 0);
            fcntl(sd, F_SETFL, flags | O_NONBLOCK);
#endif
        }
        memset(&addr, 0, sizeof(addr));
        addr.sin_family = AF_INET;
        addr.sin_addr.s_addr = ((const struct sockaddr_in *)res->ai_addr)->sin_addr.s_addr;
        addr.sin_port = htons(port);
        if(o_udp || proto == PROTO_UDP) {
            vprint("hitting udp %s:%u\n", inet_ntoa(addr.sin_addr), port);
            int total = sendto(sd, "", 1, 0, (struct sockaddr*)&addr, sizeof(addr));
        } else {
            vprint("hitting tcp %s:%u\n", inet_ntoa(addr.sin_addr), port);
            connect(sd, (struct sockaddr*)&addr, sizeof(struct sockaddr));
        }
        close(sd);
        usleep(1000*o_delay);
    }

    return(0);
}

void vprint(char *fmt, ...)
{
    va_list args;
    if(o_verbose) {
        va_start(args, fmt);
        vprintf(fmt, args);
        va_end(args);
        fflush(stdout);
    }
}

void usage() {
    printf("usage: knock [options] <host> <port[:proto]> [port[:proto]] ...\n");
    printf("options:\n");
    printf("  -u, --udp           make all ports hits use UDP (default is TCP)\n");
    printf("  -d, --delay <t>     wait <t> milliseconds between port hits\n");
    printf("  -v, --verbose       be verbose\n");
    printf("  -V, --version       display version\n");
    printf("  -h, --help          this help\n");
    printf("\n");
    printf("example:  knock myserver.example.com 123:tcp 456:udp 789:tcp\n");
    printf("\n");
    exit(1);
}
```