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

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи магістра

на тему: "Проєкт системи виявлення атак на основі Snort в IT-інфраструктурі на платформі Microsoft Active Directory"

Виконав студент 2-го курсу групи КБм-23-1
спеціальності 125 «Кібербезпека та захист інформації»
_____ М.А. Ульяніцький

Керівник старший викладач кафедри комп’ютерної інженерії та кібербезпеки
_____ І.Г. Фальковський

Рецензент доцент кафедри КІ та КБ, кандидат педагогічних наук, доцент
_____ О.С. Головня

Житомир – 2024

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютернихтехнологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека та захист інформації»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ
Завідувач кафедри КІ та КБ,
к.т.н., доцент
Андрій Єфіменко
17 вересня 2024 р.

ЗАВДАННЯ

на випускну кваліфікаційну роботу магістра

Студент: Ульяніцький Михайло Анатолійович

Тема роботи: "Проєкт системи виявлення атак на основі Snort в IT-інфраструктурі на платформі Microsoft Active Directory"

затверджена Наказом університету від 17 вересня 2024 р. № 479/с

Термін здачі студентом закінченої роботи 23 грудня 2024 р.

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

Обладнання та програмне забезпечення:

1. Топологія клієнт-серверної мережі;
2. Проєкт локальної клієнт-серверної мережі;
3. Windows Server 2019 Standart v.1809;
4. Windows 10. Professional. 22H2;
5. Kali Linux;
6. Мережевий емулятор GNS3 (v.2.2.49);
7. Гіпервізор Oracle VirtualBox (v.7.1.4).

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

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	А.А. Єфіменко	17.09.2024	17.09.2024
2	А.А. Єфіменко	10.10.2024	10.10.2024
3	А.А. Єфіменко	12.11.2024	12.11.2024

Студент _____ Михайло Ульяніцький

Керівник _____ Ігор Фальковський

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	01.09.2024 – 10.09.2024	Виконано
2	Аналіз літературних джерел та аналогічних рішень	11.09.2024 – 20.09.2024	Виконано
3	Розробка структури роботи та узгодження плану	21.09.2024 – 30.09.2024	Виконано
4	Дослідження Active Directory та систем виявлення атак	01.10.2024 – 15.10.2024	Виконано
5	Проєктування топології мережі	16.10.2024 – 31.10.2024	Виконано
6	Налаштування програмного забезпечення Snort	01.11.2024 – 15.11.2024	Виконано
7	Інтеграція Snort із сервісами Active Directory	16.11.2024 – 30.11.2024	Виконано
8	Проведення тестування системи на реальних загрозах	01.12.2024 – 10.12.2024	Виконано
9	Оформлення розділів пояснювальної записки	11.12.2024 – 13.12.2024	Виконано
10	Оформлення додатків та графічного матеріалу	14.12.2024 – 16.12.2024	Виконано
11	Підготовка до захисту та остаточне редагування	17.12.2024 – 19.12.2024	Виконано

Студент _____ Михайло Ульяніцький

Керівник _____ Ігор Фальковський

АНОТАЦІЯ

Випускна кваліфікаційна робота на тему: «Проєкт системи виявлення атак на основі Snort в IT-інфраструктурі на платформі Microsoft Active Directory» присвячена розробці та впровадженню ефективної системи виявлення атак для захисту клієнт-серверної мережі.

У роботі проаналізовано сучасні загрози, що виникають у середовищі Active Directory, та розглянуто методи захисту з використанням системи виявлення вторгнень (IDS) Snort. Було детально описано процес побудови мережевої топології, розроблено схему адресації та інтегровано Snort для моніторингу мережевого трафіку. Практичні результати включають налаштування ключових мережевих служб, таких як DNS, DHCP, NTP, і перевірку функціонування системи в умовах реальних загроз. Проведене тестування підтвердило високу ефективність запропонованих рішень у виявленні та запобіганні різноманітним кібератакам.

Робота містить 95 сторінок основного тексту, 89 рисунки, 4 таблиці, а також 20 сторінок додатків. Список використаних джерел включає 22 найменувань.

Ключові слова: СИСТЕМА ВИЯВЛЕННЯ ВТОРГНЕНЬ, ACTIVE DIRECTORY, SNORT, КІБЕРБЕЗПЕКА, МЕРЕЖЕВИЙ ТРАФІК, IDS, ПРОТОКОЛИ.

ANNOTATION

The Master's thesis titled "Design of an Attack Detection System Based on Snort in an IT Infrastructure on the Microsoft Active Directory Platform" focuses on the development and implementation of an effective intrusion detection system to protect a client-server network.

The study analyzes modern threats within the Active Directory environment and explores protection methods using the Snort intrusion detection system (IDS). The thesis provides a detailed description of the network topology design, addressing scheme development, and Snort integration for network traffic monitoring. Practical results include the configuration of key network services such as DNS, DHCP, and NTP, as well as testing the system's performance under real threat conditions. The testing confirmed the high effectiveness of the proposed solutions in detecting and mitigating various cyberattacks.

The work comprises 95 pages of main text, 89 figures, 20 tables, and 22 pages of appendices. The list of references includes 16 sources.

Keywords: INTRUSION DETECTION SYSTEM, ACTIVE DIRECTORY, SNORT, CYBERSECURITY, NETWORK TRAFFIC, IDS, PROTOCOLS.

ЗМІСТ

ВСТУП..... 8

РОЗДІЛ 1 Загальні відомості про систему виявлення атак в IT-інфраструктурі . 9

1.1 Теоретичні аспекти функціонування Windows Server 2019..... 9

1.2 Класифікація систем виявлення вторгнень..... 11

1.3 Системи виявлення вторгнень, що базуються на сигнатурах 13

1.4 Порівняння відомих систем виявлення вторгнень 14

1.5 Відмінності роботи між IDS та IPS в IT-інфраструктурі 20

1.6 Аспекти роботи зі Snort 25

РОЗДІЛ 2 ТОПОЛОГІЯ КОМП'ЮТЕРНОЇ МЕРЕЖІ ТА РОЗРОБКА СИСТЕМИ АДРЕСАЦІЇ..... 29

2.1 Побудова топології мережі..... 29

2.2 Вибір обладнання..... 31

2.3. Розробка схеми адресації мережі 35

2.4 Архітектура системи виявлення вторгнень SNORT 36

2.5. Розробка підсистеми захисту клієнт-серверної мережі організації 40

РОЗДІЛ 3. НАЛАГОДЖЕННЯ ТА ТЕСТУВАННЯ ПІДСИСТЕМИ ЗАХИСТУ КЛІЄНТ-СЕРВЕРНОЇ МЕРЕЖІ 46

3.1 Базове налагодження пристроїв мережі..... 46

3.2 Налагодження Active Directory і робочих станцій..... 52

3.3 Встановлення програми Snort..... 64

3.4 Налаштування програми Snort..... 69

3.5 Перевірка функціонування системи виявлення вторгнень 77

ВИСНОВКИ..... 92

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 93

ДОДАТКИ..... 95

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AD – Active Directory

IDS – Intrusion Detection System

IPS – Intrusion Prevention System

DNS – Domain Name System

DHCP – Dynamic Host Configuration Protocol

NTP – Network Time Protocol

IP – Internet Protocol

TCP – Transmission Control Protocol

UDP – User Datagram Protocol

LDAP – Lightweight Directory Access Protocol

SQL – Structured Query Language

VPN – Virtual Private Network

WS – Workstation

LAN – Local Area Network

HIDS – Host-based Intrusion Detection System

NIDS – Network-based Intrusion Detection System

ACL – Access Control List

ВСТУП

Актуальність теми. Тема "Проєкт системи виявлення атак на основі Snort в IT-інфраструктурі на платформі Microsoft Active Directory" є дуже актуальною в сучасному світі інформаційних технологій. Вона полягає у розв'язання проблем кібербезпеки в сучасних IT-середовищах, де кібератаки стають все більш поширеними та витонченими. Впровадження системи виявлення вторгнень (IDS) на основі Snort у поєднанні з платформою Microsoft Active Directory розширює можливості моніторингу мережі та виявлення загроз, підвищуючи таким чином загальні заходи безпеки. Тому ця тема буде дуже актуальною і зможе бути корисною для різних організацій, які використовують Active Directory на базі Windows Server і хочуть забезпечити найвищий рівень захисту своєї мережі.

Метою цієї роботи: є розробка та впровадження системи виявлення атак в IT-інфраструктурі з використанням Snort як системи виявлення вторгнень (IDS) і використання платформи Microsoft Active Directory. Таким чином, мета полягає в тому, щоб підвищити рівень кібербезпеки організації шляхом активного виявлення та пом’якшення потенційних загроз для мережі. Крім того, проєкт спрямований на дослідження ефективності інтеграції Snort з Active Directory для покращення можливостей виявлення загроз і реагування в реальних IT-середовищах.

Об’єктом дослідження: є система виявлення атак на основі Snort, реалізована в IT-інфраструктурі, яка базується на платформі Microsoft Active Directory.

Предметом дослідження є процес виявлення атак в IT-інфраструктурі з використанням системи Snort на платформі Microsoft Active Directory. Тобто, саме процес аналізу мережевого трафіку та ідентифікації потенційних загроз у контексті використання Snort в середовищі, побудованому на основі Active Directory, є предметом дослідження.

РОЗДІЛ 1 Загальні відомості про систему виявлення атак в IT-інфраструктурі

1.1 Теоретичні аспекти функціонування Windows Server 2019

Windows Server має два основні канали випуску: канал довгострокового обслуговування (LTSC) і річний канал (AC). LTSC забезпечує довгостроковий варіант, зосереджений на традиційному життєвому циклі оновлень безпеки та якості. AC забезпечує більш часті випуски, зосереджуючись на контейнерах і мікросервісах, тож ви можете швидше скористатися перевагами інновацій. Однією з ключових тем Windows Server 2019 є гібридна хмарна інтеграція, що означає можливість безперешкодного підключення різних локальних серверів та хмарних служб і ресурсів. Windows Server 2019 підтримує різні гібридні сценарії, такі як синхронізація файлів Azure, резервне копіювання Azure, відновлення сайту Azure, мережевий адаптер Azure та монітор Azure. Ці функції дозволяють синхронізувати, створювати резервні копії, відновлювати, отримувати доступ і контролювати ваші сервери в різних місцях і на різних платформах, використовуючи портал Azure як центральний інтерфейс керування. Інтеграція з гібридною хмарою може допомогти вам оптимізувати сховище, мережу та інфраструктуру безпеки, а також використовувати масштабованість і гнучкість хмари.

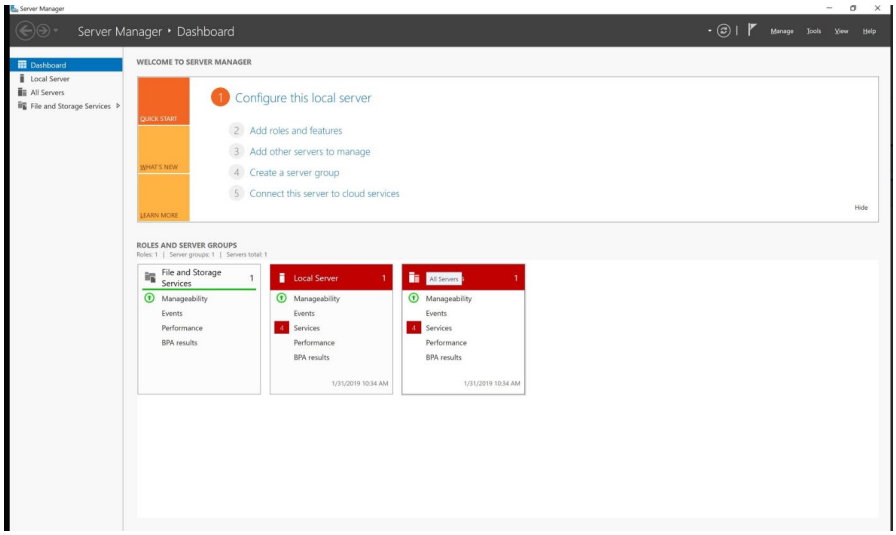


Рисунок 1.1 – Windows Server 2019

Ще одним важливим удосконаленням у Windows Server 2019 є підтримка гіперконвергентної інфраструктури (HCI), яка є способом об'єднання обчислювальних ресурсів, сховища та мережевих ресурсів в єдиний кластер серверів. HCI може спростити розгортання сервера та керування ним, зменшити витрати на обладнання та підвищити продуктивність і надійність. Windows Server 2019 представляє кілька нових функцій і інструментів для HCI, таких як Storage Spaces Direct, Storage Replica, Storage Migration Service, System Insights і Windows Admin Center. Ці функції та інструменти дозволяють створювати, налаштовувати, контролювати та переміщувати кластери HCI, а також отримувати статистичні дані та рекомендації на основі моделей робочого навантаження та даних про продуктивність.

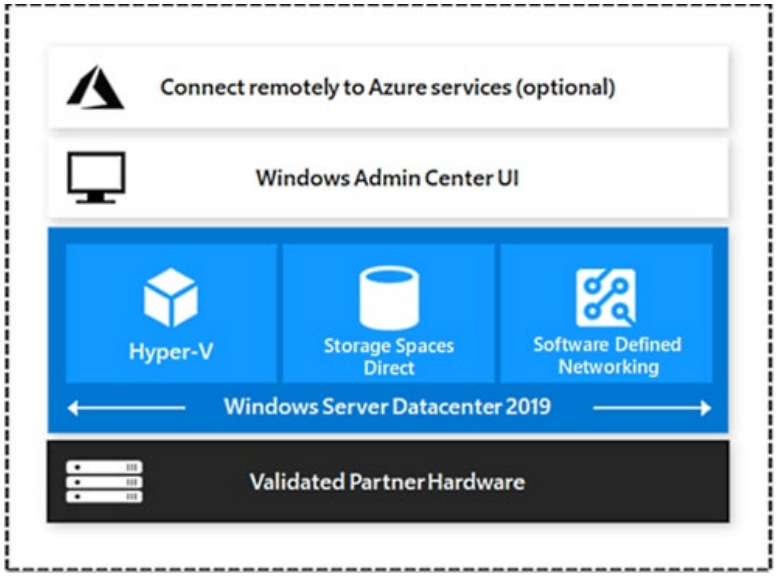


Рисунок 1.2 – Windows Server Datacenter 2019

Безпека завжди є головним пріоритетом для системних адміністраторів, і Windows Server 2019 поставляється з кількома вдосконаленнями безпеки, які допомагають захистити сервери від різних загроз і вразливостей. Ці функції та вдосконалення включають екрановані віртуальні машини, які зашифровані та ізольовані від хоста та інших віртуальних машин, запобігаючи несанкціонованому доступу та втручанню. Windows Defender Advanced Threat Protection (ATP) — це служба, яка виявляє розширені атаки на сервери та реагує на них за допомогою аналізу поведінки, машинного навчання та хмарного

інтелекту. Контейнери Windows Server — це легкі та ізольовані середовища, які можуть запускати програми, не впливаючи на хост чи інші контейнери.

Server 2019 потужніший і стабільніший, ніж будь-який із його попередників. Це означає, що основні технології інфраструктури працюватимуть краще, ніж будь-коли, і будуть більш безпечними, ніж будь-коли раніше, завдяки розміщенню цих існуючих робочих навантажень поверх Windows Server 2019.

1.2 Класифікація систем виявлення вторгнень

Система виявлення вторгнень (IDS) – це система, спрямована на виявлення потенційних атак шляхом аналізу поведінки та порівняння її з підозрілими моделями. Головною метою IDS є контроль та моніторинг мережевих ресурсів, а також виявлення незвичної та нерегулярної активності або можливих зловживань. Ця концепція існує протягом кількох років, проте нещодавно вона привернула значний інтерес дослідників та розробників систем інформаційної безпеки, оскільки стала ключовою складовою загальної інфраструктури безпеки мережі.

Система виявлення вторгнень аналізує мережевий трафік для виявлення будь-яких незвичайних або непередбачених активностей і сповіщає системного або мережевого адміністратора про виявлені аномалії. У деяких випадках IDS може вживати заходів проти небезпечного трафіку, таких як блокування або ізоляція IP-адрес користувача або джерела, щоб захистити мережу від можливих загроз.

Два найпоширеніші види IDS за місцем встановлення:

- Network Intrusion Detection System (NIDS);
- Host-based Intrusion Detection System (HIDS).

Перша працює лише на рівні мережі, а друга — лише на рівні окремо взятого хоста.

Технологія NIDS дає можливість встановити систему у стратегічно важливих місцях мережі та аналізувати вхідний/вихідний трафік всіх пристроїв

мережі. NIDS аналізують трафік на глибокому рівні, «заглядаючи» у кожен пакет із канального рівня до рівня додатків.

NIDS відрізняється від міжмережевого екрана, або фаєрвола. Фаєрвол фіксує лише атаки, що надходять зовні мережі, тоді як NIDS здатна виявити і внутрішню загрозу.

Мережеві системи виявлення вторгнень контролюють всю мережу, що дозволяє не витрачатися на додаткові рішення. Але є недолік: NIDS відстежують весь мережевий трафік, споживаючи велику кількість ресурсів. Чим більший обсяг трафіку, тим вища потреба в ресурсах CPU та RAM. Це призводить до помітних затримок обміну даними та зниження швидкості роботи мережі. Великий обсяг інформації також може приголомшити NIDS, змусивши систему пропускати деякі пакети, що робить мережу вразливою.

Альтернатива мережевим системам – хостові. Такі системи встановлюються на один хост усередині мережі та захищають тільки його. HIDS також аналізують всі вхідні та вихідні пакети, але лише одного пристрою. Система HIDS працює за принципом створення снапшотів файлів: робить знімок поточної версії і порівнює його з попередньою, виявляючи можливі загрози. HIDS краще встановлювати на критично важливі машини мережі, які рідко змінюють конфігурацію.

Крім NIDS та HIDS, доступні також PIDS (Perimeter Intrusion Detection Systems), які охороняють не всю мережу, а лише межі та сигналізують про їхнє порушення. Ще один різновид - VMIDS (Virtual Machine-based Intrusion Detection Systems). Це різновид систем виявлення загроз на основі технологій віртуалізації. Така IDS дозволяє уникнути розгортання системи виявлення на окремому пристрої. Достатньо розгорнути захист на віртуальній машині, яка відстежуватиме будь-яку підозрілу активність.

HIDS	Host Based (IDS) IDS на основі хоста
NIDS	Network based IDS IDS на основі мережі
HDS	Hybryd IDS Гібридний IDS
NNIDS	Network node intrusion detection Виявлення вторгнень мережевого вузла
AIDS	Anomaly based IDS Ідентифікатори на основі аномалій
MDS	Misuse detection system Система виявлення зловживань

Рисунок 1.3 – Типи систем виявлення вторгнень (IDS)

1.3 Системи виявлення вторгнень, що базуються на сигнатурах

Всі системи виявлення атак IDS працюють за одним принципом — пошук загрози шляхом аналізу трафіку. Відмінності криються у процесі аналізу. Існує три основні види: сигнатурні, засновані на аномаліях та засновані на правилах.

IDS цього різновиду працюють за схожим з антивірусним програмним забезпеченням принципом. Вони аналізують сигнатури та зіставляють їх із базою, яка має постійно оновлюватися для забезпечення коректної роботи. Відповідно, у цьому полягає головний недолік сигнатурних IDS: якщо з якихось причин база недоступна, мережа стає вразливою. Також якщо атака нова та її сигнатура невідома, є ризик того, що загроза не буде виявлена.

Сигнатурні IDS здатні відстежувати шаблони чи стани. Шаблони — це ті сигнатури, які зберігаються в базі, що постійно оновлюється.

Початковий стан системи – нормальна робота, відсутність атаки. Після успішної атаки система переходить у скомпрометований стан, тобто зараження пройшло успішно. Кожна дія (наприклад, встановлення з'єднання за протоколом, що не відповідає політиці безпеки компанії, активізація ПЗ тощо) здатна змінити стан. Тому сигнатурні IDS відстежують не дії, а стан системи.

Цей різновид IDS за принципом роботи у чомусь схожа з відстеженням станів, тільки має більше охоплення.

IDS, що базуються на аномаліях, використовують машинне навчання. Для правильної роботи таких систем виявлення загроз потрібний пробний період навчання. Адміністраторам рекомендується протягом перших кількох місяців повністю відключити сигнали тривоги, щоб система навчалася. Після тестового періоду вона готова до роботи. Система аналізує роботу мережі в даний момент, порівнює з аналогічним періодом та виявляє аномалії. Аномалії поділяються на три категорії:

- статистичні,
- аномалії протоколів,
- аномалії трафіку.

Статистичні аномалії виявляються, коли система IDS становить профіль штатної активності (обсяг вхідного/вихідного трафіку, програми, що запускаються і т.д.) і порівнює його з поточним профілем. Наприклад, для компанії характерне зростання трафіку по буднях на 90%. Якщо трафік раптом зросте не так на 90%, але в 900%, то система повідомить про загрозу.

Для виявлення аномалій протоколів IDS-система аналізує комунікаційні протоколи, їх зв'язки з користувачами, додатками та складає профілі. Наприклад, вебсервер повинен працювати на порту 80 для HTTP та 443 для HTTPS. Якщо для передачі інформації по HTTP або HTTPS буде використовуватись інший порт, IDS надішле повідомлення.

Також IDS здатні виявляти аномалії, будь-яку небезпечну чи навіть загрозливу активність у мережевому трафіку. Розглянемо, наприклад, випадок DoS-атаки. Якщо спробувати провести таку атаку «в лоб», її розпізнає та зупинить навіть фаєрвол. Креативні зловмисники можуть розсилати пакети з різних адрес (DDoS), що складніше виявити. Технології IDS аналізують мережевий трафік і завчасно запобігають подібним атакам.

1.4 Порівняння відомих систем виявлення вторгнень

Для забезпечення захисту IT-інфраструктури варто розглянути кілька систем виявлення атак (IDS), окрім Snort. Серед найбільш відомих рішень

виділяють Suricata, Zeek (Bro) та OSSEC. Кожна з цих систем має власні характеристики, переваги та обмеження. Ознайомлення з ключовими особливостями кожної допоможе визначити, яке рішення найкраще відповідає конкретним потребам та завданням.

Suricata – це багатофункціональна система виявлення атак, яка також підтримує функціонал IPS (Intrusion Prevention System). Її основною перевагою є можливість обробляти трафік паралельно за рахунок використання багато поточності, що забезпечує високу продуктивність. Suricata здатна працювати з правилами Snort, що робить її зручною для тих, хто вже використовує Snort і хоче розширити можливості. Крім того, Suricata може аналізувати не лише пакети, а й HTTP-запити та файли, що робить її більш універсальною для різних типів середовищ.



Рисунок 1.4 – SURICATA

Zeek (раніше відома як Bro) – це не просто IDS, а скоріше платформа для моніторингу мережевої безпеки з елементами аналізу мережевого трафіку. Основна відмінність Zeek від інших рішень полягає в тому, що він більше зосереджений на виявленні аномалій та незвичної активності, ніж на підписах атак. Zeek створює детальні журнали про мережеву діяльність, що допомагає аналізувати події після їхнього виникнення. Однак система потребує більше ресурсів і часу на налаштування, тому вона найкраще підходить для великих організацій із розвиненою кібербезпекою.



Рисунок 1.5 – Zeek

OSSEC відрізняється від інших IDS тим, що працює на рівні хостів, а не мережі. Це система виявлення вторгнень із відкритим кодом, яка спеціалізується на моніторингу файлів, аналізі журналів і виявленні аномалій на рівні серверів чи робочих станцій. OSSEC ефективно захищає від внутрішніх загроз та допомагає стежити за змінами в критично важливих файлах і конфігураціях. Вона також підтримує централізоване управління, що зручно для великих інфраструктур. Однак OSSEC не підходить для аналізу мережевого трафіку, тому її часто використовують у поєднанні з іншими IDS.

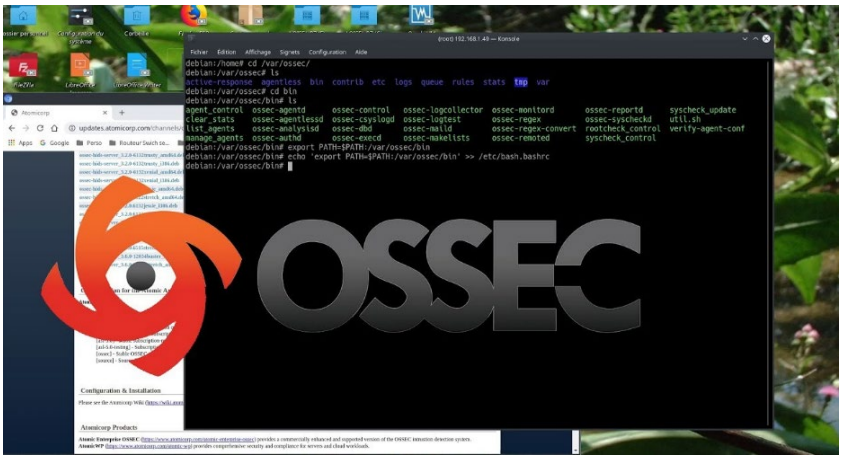


Рисунок 1.6 – OSSEC

Існує ще кілька альтернативних рішень, які можуть бути цікавими залежно від потреб організації. Наприклад, Wazuh – це форк OSSEC, який додав нові функції, зокрема інтеграцію з SIEM-системами та покращену аналітику. Snort 3

– оновлена версія Snort, що має поліпшену продуктивність і спрощену конфігурацію. Security Onion – це комплексне рішення, яке об’єднує кілька IDS та інших інструментів для моніторингу безпеки в єдиній платформі.

Вибір між Suricata, Zeek, OSSEC та іншими системами залежить від завдань, які стоять перед організацією. Suricata найкраще підходить для високонавантажених мереж, Zeek – для глибокого аналізу трафіку та виявлення аномалій, а OSSEC – для контролю на рівні хостів і серверів. У деяких випадках доцільно використовувати кілька IDS разом, щоб забезпечити всебічний захист мережі та даних.

Переваги та недоліки Snort у порівнянні з іншими рішеннями

Snort є однією з найпопулярніших систем виявлення атак (IDS/IPS) завдяки своїй простоті, ефективності та широкій підтримці спільноти. Проте, для ухвалення рішення щодо впровадження Snort важливо порівняти його з іншими рішеннями, такими як Suricata, Zeek (Bro) та OSSEC, зважаючи на їхні сильні та слабкі сторони.

Таблиця 1.1 – Переваги та недоліки Snort та порівняння з альтернативами.

Переваги Snort	Недоліки Snort	Порівняння з альтернативами
Доступність та підтримка спільноти Snort є відкритим ПЗ з великою спільнотою користувачів, що забезпечує регулярне оновлення правил та швидке реагування на нові загрози. Багато організацій використовують Snort, завдяки чому доступно багато документації та готових прикладів впровадження.	Обмежена продуктивність Snort обробляє мережевий трафік послідовно, що може створювати проблеми на високонавантажених мережах. Для великих інфраструктур Suricata виявляється більш продуктивним завдяки можливості паралельної обробки пакетів.	**Suricata** перевершує Snort у продуктивності, особливо на великих мережах, оскільки може використовувати багато поточність. Проте, його налаштування може бути складнішим для новачків.

Продовження таблиці 1.1

Гнучкість налаштувань Snort дозволяє створювати власні правила для виявлення атак та адаптувати їх під конкретні потреби. Це робить його придатним як для простих, так і для більш складних середовищ.	Zeek (Bro) спеціалізується на детальному аналізі трафіку та поведінковому моніторингу, що робить його більш придатним для виявлення складних загроз, таких як аномалії або невідомі атаки.	**Zeek (Bro)** надає розширений аналіз мережевого трафіку, виявляючи поведінкові аномалії, що робить його корисним для виявлення нових загроз.
Широке використання у промислових стандартах Багато організацій інтегрують Snort у свої системи безпеки завдяки його сумісності з іншими продуктами, що полегшує його впровадження та налаштування.	Якщо система працює з великою кількістю правил, Snort може втрачати ефективність, що ускладнює управління політиками безпеки. Suricata краще справляється з великими обсягами правил завдяки кращій оптимізації.	**OSSEC** фокусується на моніторингу хостів і аналізі журналів, що доповнює мережеві IDS, такі як Snort чи Suricata, але не може повноцінно замінити їх.
Підтримка режимів IDS та IPS Snort може працювати як у режимі моніторингу (IDS), так і у режимі активного блокування атак (IPS), що робить його універсальним рішенням.	Відсутність вбудованої підтримки для поведінкового аналізу Snort орієнтований на виявлення атак на основі підписів, тоді як Zeek та OSSEC можуть аналізувати поведінку в мережі чи на хостах, що дає можливість виявляти складніші загрози.	-

Snort, як система виявлення атак (IDS) на основі підписів, найкраще працює в середовищах, де важливо швидко розпізнавати відомі загрози та аномалії у мережевому трафіку. Завдяки своїй гнучкості й можливості створювати власні правила, Snort може бути ефективним у багатьох випадках. Ось кілька прикладів, де використання Snort стане оптимальним вибором.

1. Контроль корпоративної мережі

Snort особливо корисний для моніторингу середніх і великих корпоративних мереж, де потрібно виявляти загрози в реальному часі. Наприклад, він може виявляти сканування портів або спроби несанкціонованого доступу до серверів і робочих станцій. Це допомагає адміністраторам швидко реагувати на підозрілі активності й мінімізувати потенційну шкоду.

Приклад: Спроба атакуючого підібрати паролі до серверів через SSH можна виявити за допомогою спеціального правила в Snort. Завдяки цьому IT-відділ отримає сповіщення про можливий брутфорс-атаку й зможе вжити заходів для її припинення.

2. Захист вебдодатків від відомих атак

Snort ефективний у запобіганні поширеним вебатакам, таким як SQL-ін’єкції або XSS (міжсайтове виконання скриптів). Його підписні правила дозволяють оперативно виявляти такі загрози й захищати критично важливі вебдодатки.

Приклад: Компанія, яка надає клієнтам доступ до внутрішніх сервісів через вебплатформу, може налаштувати Snort для відстежування підозрілих запитів, що містять характерні для SQL-ін'єкцій конструкції.

3. Виявлення шкідливого ПЗ у трафіку

Snort може розпізнавати відомі зразки шкідливого програмного забезпечення, які передаються мережею. Це особливо корисно для підприємств, які обробляють великий обсяг даних та можуть стати мішенню для кібератак.

Приклад: Якщо користувачі компанії випадково завантажують заражений файл з Інтернету, Snort зможе виявити ці пакети й попередити про загрозу, перш ніж вірус потрапить у внутрішню систему.

4. Моніторинг віддалених підключень та VPN

Snort добре підходить для відстеження активності у VPN-з'єднаннях та інших віддалених підключеннях. Це важливо для організацій, що працюють з віддаленими співробітниками або підрядниками, оскільки такий доступ може стати вразливим місцем у безпеці.

Приклад: Якщо атакуючий намагається отримати доступ до корпоративної мережі через підроблений VPN-профіль, Snort зможе виявити аномальну активність і попередити про це адміністратора.

5. Комплаєнс та аудити безпеки

Snort також корисний для дотримання вимог стандартів безпеки (наприклад, PCI DSS або ISO 27001). Він дозволяє вести журнал виявлених загроз, що спрощує підготовку до аудитів та звітування про заходи безпеки.

Приклад: Фінансова установа використовує Snort для моніторингу мережі та запису всіх спроб несанкціонованого доступу до даних клієнтів. Ці дані можна використовувати для внутрішніх перевірок та зовнішніх аудитів.

6. Локальні мережі з обмеженим доступом до ресурсів

Snort є легким і не вимагає потужного обладнання, що робить його ефективним для невеликих організацій або мереж з обмеженими ресурсами. Це може бути корисним для шкіл, невеликих офісів або стартапів, яким потрібен базовий рівень захисту.

Приклад: Мала компанія, яка не має великого бюджету на кібербезпеку, може використовувати Snort для базового моніторингу мережевого трафіку й виявлення загроз.

1.5 Відмінності роботи між IDS та IPS в IT-інфраструктурі

Засоби захисту завжди розвивалися разом із появою нових методів атак. Це правило залишається актуальним і для сфери інформаційної безпеки. Сучасні рішення використовують брандмауери, системи виявлення вторгнень (IDS), вебфільтри, потокові антивірусні програми та підтримку VPN-з'єднань.

Брандмауери (firewall) здійснюють перевірку мережевих підключень – аналізують адресу та порт призначення й відправника, використовувані протоколи та стан сесії. Проте вони не здатні виявляти шкідливий чи атакуючий трафік, якщо він відповідає визначеним адміністратором правам і політикам доступу. У випадках, коли експлойт використовує вразливість у рамках дозволеного з'єднання, брандмауер не зможе заблокувати атаку. Для підвищення

рівня захисту впроваджуються нові технології, серед яких важливе місце займають системи виявлення (IDS) та запобігання (IPS) вторгненням.

Відмінності між IDS та IPS

IDS (Intrusion Detection System) – це система виявлення вторгнень, а IPS (Intrusion Prevention System) – система запобігання вторгненням. Ці технології настільки схожі, що їх часто розглядають як частину єдиного рішення. Раніше IDS існувала як окремий інструмент, але згодом стала складовою IPS. У порівнянні з традиційними засобами захисту, IDS та IPS забезпечують вищий рівень безпеки мережі, завдяки здатності своєчасно виявляти та блокувати загрози.

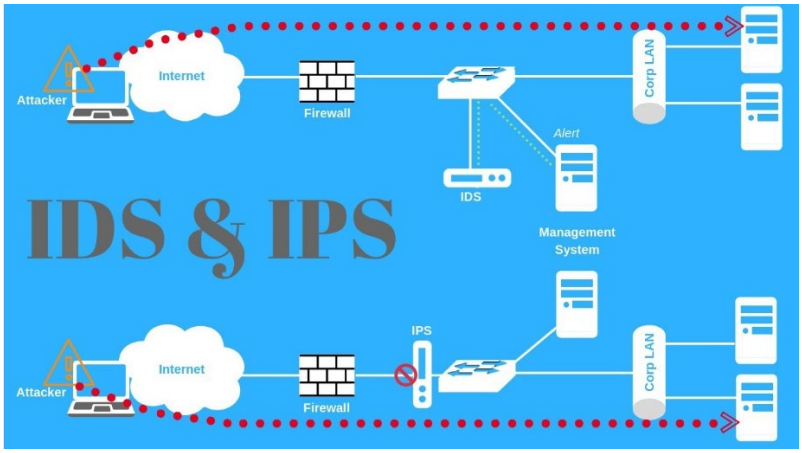


Рисунок 1.8 – IPS/IDS

Відмінності між IDS та IPS

Основна різниця між цими рішеннями полягає у способі реагування на інциденти. IDS виконує функцію моніторингу: вона розпізнає потенційно небезпечну активність та сповіщає адміністратора про загрозу. IPS, окрім виявлення проблеми, автоматично приймає заходи для боротьби з нею — розриває підозріле з'єднання або блокує IP-адресу, з якої здійснюється підозріла активність.

Розрізняються ці системи також за способом інтеграції в інфраструктуру та взаємодією з мережевим трафіком. IDS працює паралельно до потоку даних, обробляючи копії трафіку, що проходить мережею. У свою чергу, IPS

вбудовується в мережу, пропускаючи весь трафік через себе. Така архітектура робить IDS більш зручною в умовах великих навантажень, адже вона не впливатиме на загальну мережеву активність, навіть якщо її ресурси будуть перевантажені. Натомість IPS може стати «вузьким місцем», сповільнюючи роботу мережі та створюючи додаткове навантаження на брандмауер.

Що контролюють системи виявлення та запобігання вторгненням

Як уже зазначалося, IDS аналізує мережевий трафік та порівнює його з наявною базою загроз. Завдяки цьому вона здатна виявляти:

- Мережеві атаки – спроби порушити роботу системи через трафік;
- Несанкціонований доступ до даних – підозрілі спроби отримати конфіденційну інформацію;
- Активність шкідливого ПЗ – виявлення вірусів чи троянів у трафіку;
- Сканування портів – спроби знайти відкриті порти для подальших атак;
- Аномальну поведінку – незвичну активність, яка може свідчити про приховану загрозу;
- Використання ботнетів та майнінг-пулів – виявлення кіберзагроз, які можуть бути частиною масових атак або майнінгу криптовалют;
- Порушення політик безпеки – контроль за виконанням встановлених правил доступу та використання ресурсів.

Завдяки такому функціоналу IDS та IPS здатні забезпечити високий рівень захисту та знизити ризик кібератак на мережеву інфраструктуру.

Категорії IDS за місцем встановлення та принципом дії

Системи виявлення вторгнень (IDS) класифікуються за місцем встановлення та принципом роботи. Далі розглянемо обидва підходи.

Класифікація за місцем встановлення

- Система виявлення мережевих вторгнень (NIDS)

NIDS аналізує мережевий трафік усіх пристроїв у мережі, перевіряючи кожен пакет на всіх рівнях – від канального до прикладного. Така система здатна

виявити як зовнішні, так і внутрішні загрози. Однак зі зростанням обсягу трафіку збільшуються вимоги до процесора та оперативної пам'яті. В умовах перевантаження NIDS може не встигати обробляти всі пакети, що створює ризик пропуску загроз.

- Система виявлення вторгнень на хості (HIDS)

HIDS встановлюється на окремому хості всередині мережі та аналізує лише його трафік і активність. Принцип роботи полягає у створенні знімка системи та порівнянні його з попередньою версією, що дозволяє виявити небажані зміни чи загрози. Така система найкраще підходить для критично важливих серверів, де зміни конфігурації відбуваються рідко.

- Периметрова система виявлення вторгнень (PIDS)

PIDS — це базове рішення, яке контролює доступ до мережі на її межі, подібно до паркану навколо будинку. Вона не забезпечує повний захист мережі, а лише повідомляє про потенційні порушення периметра.

- Система виявлення вторгнень на основі віртуальних машин (VMIDS)

Ця система використовує можливості віртуалізації, що усуває потребу в розгортанні окремого пристрою для моніторингу. VMIDS розміщується на віртуальній машині, що дозволяє аналізувати підозрілу активність у віртуальному середовищі без значних змін у фізичній інфраструктурі.

- Система виявлення вторгнень на основі прикладних протоколів (APIDS)

APIDS аналізує пакети даних, які передаються через протоколи прикладного рівня. Прикладом може бути контроль SQL-запитів для виявлення шкідливої активності при взаємодії з базами даних.

- Гібридна система виявлення вторгнень (HyIDS)

HyIDS поєднує функції кількох систем, описаних вище, для досягнення більшого рівня захисту. Це комплексне рішення дозволяє використовувати сильні сторони різних підходів у рамках єдиної архітектури.

Цей підхід до класифікації IDS допомагає організаціям обрати оптимальне рішення для своїх потреб, залежно від особливостей інфраструктури та загроз, які необхідно контролювати.

Класифікація IDS за принципом дії:

Сигнатурні системи виявлення вторгнень працюють за принципом, схожим на антивірусне програмне забезпечення. Вони порівнюють трафік із відомими сигнатурами загроз, які зберігаються у постійно оновлюваній базі даних. Якщо доступ до бази буде відсутній або вона виявиться застарілою, ефективність такої системи знижується. Основний недолік сигнатурних IDS полягає в тому, що вони не здатні виявляти нові або невідомі загрози, для яких ще не створено сигнатури. Таким чином, ці системи можуть пропустити атаки з новими характеристиками. Крім того, вони зосереджені на відстеженні стану системи, а не на аналізі окремих подій.

Системи, що працюють на основі аномалій, застосовують технології машинного навчання для виявлення відхилень у поведінці мережі. Для коректної роботи такої системи спочатку необхідно провести навчання, яке дозволяє зрозуміти, як виглядає "нормальна" активність у мережі. Тривалість навчання залежить від складності IT-інфраструктури.

Принцип роботи IDS на основі аномалій полягає у порівнянні поточного трафіку та активності з попередніми періодами, щоб виявити три основні типи аномалій:

Статистичні аномалії – виявлення відхилень у кількості запитів або обсягах трафіку.

Аномалії протоколів – помилки чи нестандартне використання мережевих протоколів.

Такі системи є досить ефективними у виявленні нових загроз, але їхнє налаштування та підтримка можуть бути складними. Потрібен час для навчання системи, а також регулярне оновлення моделей для уникнення помилкових спрацьовувань.

1.6 Аспекти роботи зі Snort

Snort — Система, яка максимально задовольняє основні вимоги та відповідає потребам організацій.

запобігання вторгнень (IPS/IDS) із відкритим кодом у світі. Snort IPS/IDS використовує серію правил, які допомагають визначити зловмисну мережеву активність, і використовує ці правила для пошуку пакетів, які збігаються з ними, і генерує сповіщення для користувачів.

Snort також можна розгорнути всередині, щоб зупинити ці пакети. Snort має три основні призначення: як сніффер пакетів, як tcpdump, як реєстратор пакетів — що корисно для налагодження мережевого трафіку, або його можна використовувати як повномасштабну систему запобігання вторгненню в мережу. Snort можна завантажити та налаштувати як для особистого, так і для ділового використання. Ось деякі ключові аспекти Snort:

- Виявлення загроз: Snort аналізує мережевий трафік і виявляє аномалії, що можуть свідчити про наявність атак або вразливостей в мережі. Він може реагувати на різні типи атак, включаючи відомі шаблони атак, підозрілі дії користувачів та небезпечний мережевий трафік.
- Правила виявлення: Snort використовує правила для виявлення специфічних типів атак або аномальної поведінки. Ці правила можуть бути налаштовані для виявлення конкретних загроз або взаємодії з певними мережевими протоколами.
- Аналіз трафіку: Snort може аналізувати різні аспекти мережевого трафіку, включаючи заголовки пакетів, дані та сигнатури атак, щоб виявити підозрілі дії.
- Потужність та гнучкість: Snort є дуже потужною та гнучкою системою, яка може бути налаштована для виявлення різних видів загроз та адаптуватися до різних мережевих середовищ.

- Відкрите програмне забезпечення: Snort є відкритим програмним забезпеченням з відкритим вихідним кодом, що дозволяє користувачам вносити зміни та розробляти додаткові функції відповідно до їх потреб.
- Інтеграція з іншими системами безпеки: Snort може інтегруватися з іншими системами безпеки, такими як фаєрволи, системи журналювання подій та системи управління подіями, для створення комплексних захисних рішень.



Рисунок 1.7 – Snort

Правила Snort є важливим компонентом, що забезпечує ефективність роботи системи. Кожне правило в Snort складається з двох основних частин: заголовка та опцій. Заголовок містить основну інформацію, необхідну для визначення, які пакети потрібно перевіряти. Він включає тип протоколу (TCP, UDP, ICMP), IP-адреси джерела та призначення, порти та напрямок трафіку. Наприклад, правило може виглядати так: alert tcp any any -> 192.168.1.0/24 80, що означає, що система буде генерувати сповіщення для всіх TCP-пакетів, які надходять на порт 80 з будь-якої адреси до підмережі 192.168.1.0/24.

Опції правила уточнюють умови, які мають бути виконані для спрацьовування правила. Вони можуть містити різні параметри, такі як msg, яке визначає текст повідомлення, що буде виведено при спрацьовуванні правила, або content, яке вказує на конкретний вміст, що потрібно знайти в пакеті. Наприклад, правило може виглядати так: alert tcp any any -> 192.168.1.0/24 80 (msg:"HTTP GET detected"; content:"GET"; sid:1001; rev:1;), що означає, що буде згенеровано сповіщення при виявленні HTTP-запиту з текстом "GET" на порт 80.

Snort також підтримує регулярні вирази для створення складніших умов у правилах, що дозволяє більш точно налаштувати систему для виявлення специфічних загроз. Крім того, правила Snort можуть бути згруповані в категорії, що полегшує управління ними та їх налаштування відповідно до вимог безпеки конкретної організації.

Регулярне оновлення правил Snort є критично важливим для забезпечення захисту від нових загроз. Кіберзлочинці постійно розробляють нові методи атак, тому важливо, щоб система виявлення була адаптована до останніх тенденцій у безпеці. Snort дозволяє користувачам завантажувати нові правила з репозиторіїв спільноти, що забезпечує їх актуальність та ефективність.

Ще одна важлива функція Snort — це можливість інтеграції з іншими системами безпеки. Snort можна використовувати в комбінації з брандмауерами, системами управління подіями безпеки (SIEM) та іншими рішеннями, що дозволяє створити комплексну систему захисту. Це забезпечує більш широкий огляд безпеки в організації та дозволяє швидше реагувати на інциденти.

Snort також є відкритим програмним забезпеченням, що дозволяє користувачам вносити зміни та розробляти нові функції відповідно до своїх потреб. Це надає гнучкість у налаштуванні системи та дозволяє організаціям адаптувати Snort під свої специфічні вимоги.

У підсумку, Snort є потужним і гнучким інструментом для виявлення загроз у мережі. Його можливості в аналізі трафіку, виявленні аномалій і інтеграції з іншими системами безпеки роблять його важливим компонентом в арсеналі засобів інформаційної безпеки. Правила Snort, їх структура та адаптивність є ключовими для забезпечення високого рівня захисту інформаційних систем від сучасних кіберзагроз.

Висновки до першого розділу:

У першому розділі було розглянуто основні принципи та інструменти забезпечення захисту мережевих інфраструктур від атак, зокрема через використання IDS/IPS. Проаналізовано особливості роботи операційної системи

Windows Server 2019 та платформи Active Directory, що є фундаментальними елементами сучасної IT-інфраструктури. Крім того, значну увагу приділено системі виявлення вторгнень Snort та її порівнянню з іншими популярними рішеннями, такими як Suricata, Zeek (Bro) та OSSEC.

Розглянуті підходи до класифікації IDS за місцем встановлення та принципами дії допомогли визначити, в яких умовах кожна з них є найбільш ефективною. Сформовано базу знань про теоретичні та практичні аспекти захисту мережі, що є важливим етапом для подальшого проєктування та реалізації системи виявлення атак.

РОЗДІЛ 2 ТОПОЛОГІЯ КОМП'ЮТЕРНОЇ МЕРЕЖІ ТА РОЗРОБКА СИСТЕМИ АДРЕСАЦІЇ

2.1 Побудова топології мережі

Клієнт-серверна мережа — це архітектура, де пристрої поділяються на сервери та клієнти. Сервери надають ресурси й сервіси, такі як файли, бази даних, електронна пошта та автентифікація, а клієнти надсилають запити для їх отримання. Мережа зазвичай реалізується за зіркоподібною або гібридною топологією, де сервери є центром взаємодії. Основні переваги такої мережі — централізоване управління та підвищена безпека, адже критичні дані зберігаються на серверах. Водночас недоліками є залежність від серверів: у разі збою доступ до ресурсів може бути заблокований. Для забезпечення надійності важливо впроваджувати механізми резервування та продуману систему адресації.

У цій роботі було розглянуто клієнт-серверну мережу, в якій було використано 1 маршрутизатор: R-231-27-1. Також будуть залучені комутатори SW-231-27-1 та SW-231-27-2. Робочі станції будуть підключені до комутаторів SW-231-27-1 та SW-231-27-2. В даній мережі налаштовано доступ до інтернету. На мережевих пристроях налаштовано синхронізацію часу через NTP, відправку даних про події з використанням Syslog та аутентифікацію з використанням Radius. На Windows Server буде налаштовано AD, DHCP-сервер, DNS-сервер, NTP-сервер, Syslog-сервер, Radius-сервер. Інтернет-з'єднання забезпечує доступ до глобальних ресурсів та дозволяє організації ефективно співпрацювати з партнерами, клієнтами й іншими зовнішніми сторонами. Воно також є важливим для своєчасного отримання оновлень безпеки та використання хмарних платформ і сервісів.

Робочі станції і сервери включають: Windows 10, яка є ключовою операційною системою для сучасних робочих станцій, забезпечуючи сумісність із новітніми програмами та сервісами. Windows Server, який використовується

для централізованого адміністрування мережі, управління даними, контролю доступу та виконання інших серверних задач.

Трирівнева топологія мережі — це структура, яка поділяє мережу на три логічні рівні: ядро (Core), агрегацію (Distribution) та доступ (Access). Така архітектура забезпечує ефективну роботу мережі, підвищує її масштабованість і спрощує управління. Цей підхід часто застосовується в корпоративних мережах, де потрібно оптимізувати обробку великих обсягів трафіку та забезпечити надійність з'єднань.

Рівень ядра (Core) відповідає за швидку передачу даних між основними сегментами мережі. Він забезпечує високу пропускну здатність і мінімальні затримки, виконуючи функції швидкої маршрутизації. На цьому рівні зазвичай використовуються потужні маршрутизатори та комутатори, які здатні обробляти великий обсяг трафіку. Оскільки ядро є критичним компонентом, його проектують з урахуванням резервування для мінімізації простоїв у разі відмови обладнання.

Рівень агрегації (Distribution) виступає як проміжний шар між ядром і рівнем доступу. Він консолідує трафік від кількох комутаторів рівня доступу та передає його на рівень ядра. Крім того, на цьому рівні застосовуються політики безпеки та фільтрація трафіку, наприклад, за допомогою списків контролю доступу (ACL). Рівень агрегації також використовується для впровадження балансування навантаження та управління міжмережевими з'єднаннями.

Рівень доступу (Access) забезпечує підключення кінцевих пристроїв, таких як комп'ютери, принтери та IP-телефони, до мережі. На цьому рівні встановлюються комутатори, які забезпечують передачу даних між користувацькими пристроями та вищими рівнями мережі. Також тут можуть застосовуватися механізми контролю доступу, такі як автентифікація користувачів і захист від несанкціонованого доступу.

Трирівнева топологія допомагає розподілити навантаження в мережі та підвищити її надійність завдяки чіткому розмежуванню функцій кожного рівня.

Така архітектура дозволяє легко масштабувати мережу шляхом додавання нових пристроїв або сегментів без порушення загальної структури. Вона також полегшує управління, оскільки політики безпеки та правила маршрутизації можуть бути централізовано реалізовані на рівні агрегації та ядра.

У процесі проєктування була реалізована трирівнева модель мережі, яка має ієрархічну структуру. Ця модель передбачає поділ мережі на три взаємопов'язані рівні: ядро, розподіл та доступ. Кожен з рівнів виконує окремі функції, що забезпечує ефективність і надійність роботи мережі. Структура цієї мережі наочно представлена на рисунку нижче.

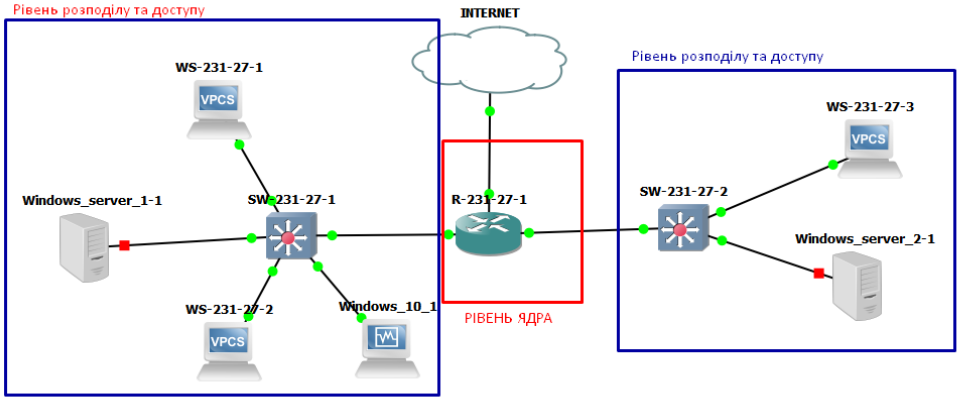


Рисунок 2.1 – Загальна топологія мережі

Рівень ядра складається з 1 маршрутизатора, він з’єднаний між собою канальними зв’язками технології Fast Ethernet.

Рівень розподілу та доступу має 2 комутатори, та підключені робочі станції, що під’єднані до ядра мережі. З’єднання до ядра використано за допомогою технології Ethernet.

2.2 Вибір обладнання

Мережа спроєктована на базі обладнання Cisco. Нижче представлені технічні характеристики пристроїв, які будуть використані в цій мережі.

Роутер Cisco C7200 — це потужний мережевий пристрій, яке широко використовується для маршрутизації в середніх і великих корпоративних мережах. Цей роутер, випущений у середині 90-х років, здобув популярність

завдяки своїй гнучкості, продуктивності та можливостям масштабування. Cisco C7200 оснащений потужними процесорами, що забезпечують високу швидкість обробки пакетів, і має можливість установки до 512 МБ оперативної пам'яті (RAM) та 128 МБ флешпам'яті, що дозволяє підтримувати велику кількість маршрутизаційних записів і функцій.

Роутер підтримує різноманітні модулі інтерфейсу, що дозволяє підключати різні типи мережевих з'єднань, включаючи Ethernet, T1/E1, T3/E3 та інші, завдяки чому він може адаптуватися під конкретні вимоги мережі. Продуктивність Cisco C7200 може досягати десятків тисяч пакетів на секунду, залежно від конфігурації, що робить його ідеальним для обробки великих обсягів трафіку.

Пристрій працює на основі операційної системи Cisco IOS, яка забезпечує широкий спектр функцій маршрутизації, комутації, безпеки та управління трафіком. Це програмне забезпечення дозволяє налаштовувати роутер для різноманітних сценаріїв використання, підвищуючи його гнучкість. Cisco C7200 розроблений для забезпечення високої надійності та безперервності роботи, підтримуючи функції резервування та автоматичного відновлення в разі відмови.

Додатково, роутер підтримує різноманітні механізми безпеки, включаючи доступ за списками, шифрування трафіку та автентифікацію користувачів, що дозволяє забезпечити безпечний обмін даними в мережі. Завдяки можливості додавання модулів інтерфейсу, C7200 може легко адаптуватися до зростаючих потреб бізнесу, що робить його універсальним рішенням для багатьох організацій.

У підсумку, роутер Cisco C7200 є надійним і потужним рішенням для маршрутизації, яке підходить для середніх і великих корпоративних мереж. Його гнучкість, масштабованість та підтримка різних мережевих технологій роблять його одним з кращих виборів для організацій, які прагнуть забезпечити ефективну і безпечну маршрутизацію трафіку.



Рисунок 2.2 – Високопродуктивний маршрутизатор Cisco C7200

Таблиця 2.1 – Характеристика маршрутизатора cisco c7200 adventerpricek9

Інтерфейси	Загальна інформація
Стандартна пам'ять	1 Gb
Максимальна пам'ять:	2 Gb
Технологія пам'яті	SDRAM
Флешпам'ять	64 Mb
Підтримується карта пам'яті	CompactFlash (CF) Card

2. Образ Комутатора Cisco i86bi

Комутатор Cisco i86bi є потужним мережевим пристроєм, призначеним для забезпечення високої продуктивності в сучасних корпоративних мережах. Цей комутатор належить до серії Cisco 8800 і розроблений для використання в умовах, що вимагають швидкої та надійної передачі даних. Cisco i86bi підтримує передові технології, такі як інтелектуальна маршрутизація, що дозволяє йому ефективно обробляти великі обсяги трафіку.

Комутатор підтримує різноманітні протоколи, включаючи Ethernet, VLAN, і TCP/IP, що робить його універсальним рішенням для різних мережевих середовищ. Завдяки своїй архітектурі, Cisco i86bi забезпечує високу пропускну

здатність і малу затримку, що є критично важливим для додатків, чутливих до затримок, таких як VoIP та потокове відео.

Обладнання оснащене багатьма портами, які дозволяють підключати різні пристрої, такі як комп'ютери, сервери та інші мережеві компоненти. Cisco i86bi підтримує технології управління трафіком, такі як Quality of Service (QoS), що дозволяє пріоритизувати трафік і забезпечувати стабільну роботу критичних застосунків. Крім того, комутатор забезпечує функції безпеки, включаючи контроль доступу та шифрування даних, що підвищує загальний рівень захисту мережі.

Cisco i86bi також має можливості для управління та моніторингу, що дозволяє адміністраторам легко контролювати стан мережі, а також виявляти та усувати проблеми в реальному часі. Завдяки підтримці SNMP та інших протоколів управління мережами, адміністратори можуть налаштовувати та управляти комутатором віддалено, що спрощує адміністрування мережі.

У підсумку, комутатор Cisco i86bi є потужним і гнучким рішенням для підприємств, які прагнуть забезпечити ефективність, надійність і безпеку своїх мереж. Завдяки підтримці новітніх технологій та можливості масштабування, Cisco i86bi підходить для використання в різних мережевих середовищах, що робить його ідеальним вибором для сучасних організацій.

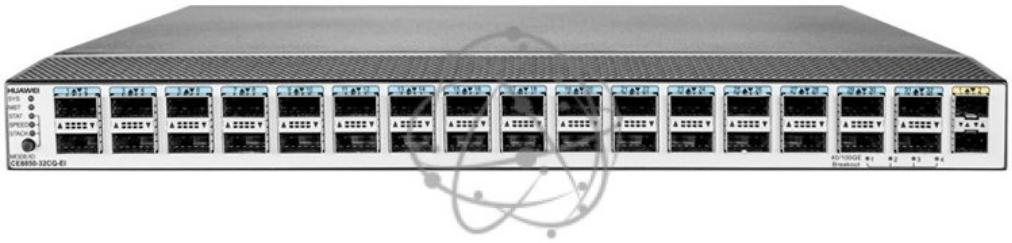


Рисунок 2.3 – Образ Cisco i86bi

У якості кінцевих робочих станцій використовуватимуться комп’ютери з операційною системою Windows 10. Ці станції будуть функціонувати з мінімальними характеристиками: 4-ядерний процесор, 8 ГБ оперативної пам'яті, накопичувач об'ємом 256 ГБ (HDD або SSD) та мережевий інтерфейс зі швидкістю 100 Mbit. Як сервер буде використовуватися комп’ютер з Windows Server 2019, обладнаний 8-ядерним процесором, 16 ГБ оперативної пам'яті, накопичувачем об'ємом 512 ГБ або більше (HDD чи SSD) та мережевим інтерфейсом зі швидкістю 1 Gbit.

2.3. Розробка схеми адресації мережі

Розробка схеми адресації для клієнт-серверної мережі є важливим кроком для забезпечення ефективного функціонування та управління мережевими ресурсами. Правильна адресація дозволяє оптимізувати трафік, підвищити безпеку та забезпечити зручне адміністрування всієї інфраструктури. У процесі розробки адресації для клієнт-серверної мережі зазвичай використовуються два основні підходи: статична і динамічна адресація. Для серверів, маршрутизаторів та інших ключових пристроїв рекомендується використовувати статичну IP-адресацію, яка забезпечує стабільний доступ і спрощує їхнє адміністрування. Динамічна адресація, реалізована через протокол DHCP, зазвичай використовується для клієнтських пристроїв. Це зменшує навантаження на адміністраторів та запобігає конфліктам IP-адрес у великих мережах. Важливим аспектом є поділ мережі на підмережі, що дозволяє забезпечити сегментацію трафіку для підвищення продуктивності та безпеки. Наприклад, можна виділити окремі підмережі для серверів, клієнтів та мережевих сервісів, що дозволить розмежувати доступ і налаштувати специфічні політики контролю доступу. Для зручності управління та моніторингу мережі зазвичай використовуються маски підмережі, що визначають кількість доступних IP-адрес у кожному сегменті. Водночас маршрутизація між підмережами забезпечується за допомогою комутаторів або маршрутизаторів третього рівня. Це дозволяє ефективно передавати трафік між сегментами без втрат продуктивності. Реплікація адрес і

резервування ключових маршрутів допомагає забезпечити безперервну роботу мережі навіть у випадку збоїв. Схема адресації є критичним елементом у проєктуванні мережі, який забезпечує баланс між продуктивністю, безпекою і гнучкістю розширення мережі у майбутньому.

Для локальної мережі буде залучено 2 адреси: 192.168.1.0/24 та 192.168.1.0/24

Для інтернету та для Роутера була залучена адреса: 202.192.27.0

Таблиця 2.2 – Параметри підключення пристроїв

Пристрій	Інтерфейс	Підключення до пристрою	Підключення до інтерфейсу
INTERNET	Eth 0/0	R-231-27-1	Eth 0/0
Router R-231-27-1	Eth 1/0	SW-231-27-1	Eth 0/0
	Eth 1/1	SW-231-27-2	Eth 0/0
Комутатор SW-231-27-1	Eth 0/1	WS-231-27-1	Eth 0
	Eth 1/0	Windows_server_1-1	Eth 0
	Eth 0/3	WS-231-27-2	Eth 0
	Eth 0/2	Windows_10_1	Eth 0
Комутатор SW-231-27-2	Eth 0/1	WS-231-27-3	Eth 0
	Eth 1/0	Windows_server_2-1	Eth 0

Здійсню розподіл адрес для локальних мереж. Для зручності результат наведений в таблиці 2.2.

Таблиця 2.3 Адресація локальної мережі

Призначення адрес	IP — addressaddress
Адреса мережі	192.168.1.0 – 192.168.2.0
Маска мережі	255.255.255.0(/24)
Мінімальна адреса	192.168.1.248
Шлюз за замовчуванням (1)	192.168.1.254
Призначені Windows Server (2)	192.168.1.252 – 192.168.2.252
Призначені комутаторам (2)	192.168.19.240 – 192.168.19.243

2.4 Архітектура системи виявлення вторгнень SNORT

Система SNORT є популярним інструментом для виявлення та запобігання вторгненням у мережу завдяки своїй гнучкій архітектурі та відкритому

вихідному коду. Її структура дозволяє ефективно аналізувати мережевий трафік у реальному часі, виявляти аномалії та потенційні загрози. Основними компонентами SNORT є модулі захоплення трафіку, механізм декодування, система застосування правил і модуль реагування. Ця архітектура забезпечує можливість гнучкого налаштування та масштабування, що робить SNORT придатним як для невеликих локальних мереж, так і для великих корпоративних середовищ.

Ключовим компонентом SNORT є модуль захоплення трафіку, який працює на рівні мережевого інтерфейсу та перехоплює пакети даних, що проходять мережею. Цей модуль діє подібно до сніффера, аналізуючи як вхідний, так і вихідний трафік. Захоплені пакети передаються далі в систему для більш глибокого аналізу. Завдяки цьому SNORT може фільтрувати трафік за протоколами, IP-адресами, портами та іншими параметрами.

Наступний етап обробки виконує декодувальний модуль, який розшифровує захоплені пакети та виділяє корисну інформацію. Він розбиває пакети на рівні відповідно до моделі OSI, що дозволяє аналізувати як заголовки, так і вміст даних. Цей етап є важливим для коректного розпізнавання атак, оскільки багато загроз маскують свій трафік на різних рівнях протоколів.

Після декодування пакети передаються в модуль застосування правил, який є основним компонентом SNORT. Цей модуль порівнює пакети з наявними правилами для виявлення потенційних загроз. Правила можуть бути спрямовані на виявлення відомих атак (сигнатурні правила) або аномальної поведінки в мережі. Вони задаються в спеціальному форматі та можуть включати інформацію про тип протоколу, джерело та призначення трафіку, ключові слова або шаблони вмісту. Наприклад, SNORT може шукати сигнатуру SQL-ін’єкції або спробу несанкціонованого доступу до сервера через SSH.

Ще одним важливим компонентом є препроцесори, які виконують додатковий аналіз трафіку перед застосуванням основних правил. Вони дозволяють підготувати дані для кращого розпізнавання атак, обробляючи

специфічні типи протоколів, як-от HTTP або DNS. Препроцесори можуть об'єднувати фрагментовані пакети, що дає змогу виявляти складні атаки, які намагаються обійти систему за рахунок розбиття трафіку.

Після того, як пакет відповідає певному правилу, SNORT активує модуль реагування. Цей модуль визначає, які дії необхідно виконати у разі виявлення загрози. Дії можуть включати створення сповіщення, запис інформації до журналу подій або активне блокування трафіку. У режимі IPS (Intrusion Prevention System) SNORT може не тільки виявляти, але й запобігати атакам, блокуючи шкідливий трафік у режимі реального часу. В режимі IDS (Intrusion Detection System) він лише фіксує підозрілу активність і повідомляє адміністратора про можливу загрозу.

Журнали подій, які генерує SNORT, відіграють важливу роль у забезпеченні безпеки, оскільки містять інформацію про виявлені загрози. Ці дані можуть бути проаналізовані адміністраторами для подальших дій, таких як оновлення політик безпеки або зміна правил моніторингу. SNORT підтримує інтеграцію з іншими інструментами, наприклад, з системами управління подіями безпеки (SIEM), що дозволяє централізовано відстежувати всі інциденти в мережі.

Гнучкість SNORT забезпечується завдяки його відкритому коду, що дозволяє користувачам створювати власні правила та розширювати функціональність системи. Це особливо важливо в умовах динамічних загроз, коли потрібно швидко реагувати на нові вразливості. SNORT також активно підтримується спільнотою, що дозволяє регулярно отримувати нові правила та оновлення.

Snort – це популярна програма з відкритим кодом, яка забезпечує гнучке налаштування та високу портативність як система виявлення мережевих вторгнень на основі шаблонів. Її легко розгорнути на різних мережевих вузлах, а ефективність роботи забезпечується мінімальним споживанням ресурсів, таких як оперативна пам'ять і процесорний час. Snort використовує набір підписів, що

визначають конкретні типи атак і шкідливої активності. Ці підписи відомі як правила Snort, і саме вони визначають, як система розпізнає та реагує на загрози.

Кожне правило в Snort складається з кількох основних компонентів: <дія правила> <протокол> <IP-адреса джерела> <порт джерела> <напрямок> <IP-адреса призначення> <порт призначення> <опції правила>. Дія правила визначає, як система повинна реагувати на виявлену активність – створити сповіщення, записати журнал чи просто проігнорувати пакет. Найпоширенішими є правила для створення сповіщень, які фіксують дані про підозрілу активність для подальшого аналізу.

Крім основних параметрів мережевого пакета, у правилах можуть бути вказані додаткові опції, що задаються у форматі пар "ключ-значення". Вони можуть містити інформацію про тип загрози, текст повідомлення для адміністратора, пріоритет оповіщення та інші параметри. Наприклад, система може призначати різні рівні пріоритетів загрозам: пріоритет 1 вказує на критичну небезпеку, тоді як пріоритет 4 означає менш серйозну загрозу.

Ось приклад правила Snort:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: 'SYN FIN Scan Detected'; flags: SF; reference: arachnids, 198; classtype: attempted-recon;).

У цьому прикладі система генерує сповіщення для TCP-трафіку, що надходить із зовнішньої мережі до внутрішньої. Вона перевіряє прапорці пакета (flags) на наявність комбінації SYN та FIN, що часто вказує на сканування портів. До правила також додано довідкову інформацію та класифікацію, що допомагає під час аналізу інцидентів.

Таким чином, Snort завдяки правилам здатний швидко виявляти та реагувати на різноманітні загрози, забезпечуючи надійний захист мережевої інфраструктури.

Основні компоненти архітектури Snort наведені на рисунку 2.4.

Архітектура Snort є комплексною системою для виявлення мережевих вторгнень, що складається з кількох ключових компонентів. На початковому

етапі, мережевий інфраструктурний рівень, або "мережевий хребет", забезпечує передачу даних між різними пристроями в мережі. Сніфер, розташований на цьому рівні, перехоплює та фіксує весь мережевий трафік, що дозволяє Snort збирати інформацію про пакети для подальшого аналізу.

Наступним етапом є препроцесор, який проводить попередню обробку отриманих даних, фільтруючи та нормалізуючи трафік, що сприяє спрощенню аналізу. Основною частиною архітектури є двигун виявлення, який аналізує дані згідно з наборами правил. Цей двигун порівнює інформацію про пакети з сигнатурами атак, що визначені в правилах, і генерує сповіщення у разі виявлення підозрілого трафіку.

Набори правил є важливим елементом, оскільки вони описують характеристики атак і можуть бути налаштовані відповідно до специфічних потреб мережі. Вони також мають пріоритети, що дозволяє системі визначати серйозність загроз. Після виявлення атак, сповіщення передаються до модулів для журналювання, які фіксують інформацію про інциденти для подальшого аналізу та реагування. Таким чином, архітектура Snort забезпечує потужний механізм для виявлення та запобігання мережевим загрозам.

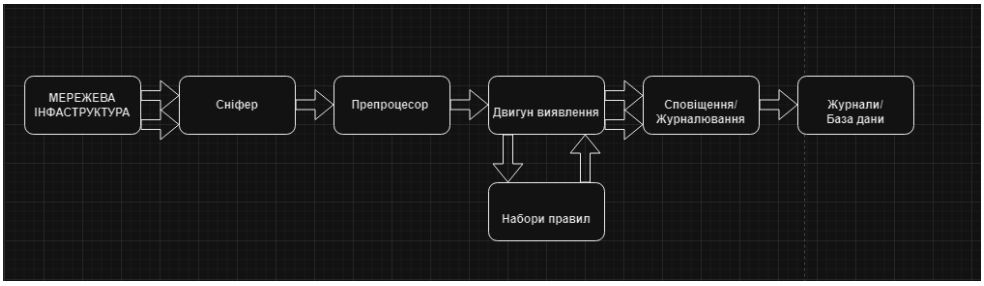


Рисунок 2.4 – Архітектура Snort

2.5. Розробка підсистеми захисту клієнт-серверної мережі організації

Підсистема захисту клієнт-серверної мережі організації є важливим компонентом, що забезпечує захист даних, зменшує ризики витоку інформації та запобігає несанкціонованому доступу до ресурсів мережі. Розробка надійної системи безпеки передбачає впровадження кількох рівнів захисту, які охоплюють як мережевий, так і прикладний рівень. Основним завданням

підсистеми є контроль доступу до ресурсів мережі, моніторинг активності користувачів і захист від зовнішніх та внутрішніх загроз. На першому етапі створення підсистеми захисту проводиться аудит існуючої мережевої інфраструктури, щоб визначити потенційні вразливості, зони ризику та визначити ключові точки, де потрібна посилена безпека.

Для захисту мережевої інфраструктури використовуються міжмережеві екрани або брандмауери, які фільтрують вхідний і вихідний трафік відповідно до політик безпеки. Брандмауери блокують несанкціонований доступ до критичних серверів та захищають клієнтські робочі станції від зовнішніх атак. Важливим компонентом є використання систем виявлення та запобігання вторгнень (IDS/IPS), які дозволяють виявляти аномальну активність і запобігати загрозам у реальному часі. Система IDS/IPS аналізує мережевий трафік і використовує сигнатури для розпізнавання відомих атак або аномальної активності, що може свідчити про потенційні загрози.

Окрім цього, підсистема захисту передбачає використання VPN (Virtual Private Network) для захищеного з'єднання з віддаленими користувачами та філіями. VPN шифрує трафік, що дозволяє запобігти перехопленню даних під час їхньої передачі через мережу. Для автентифікації користувачів впроваджується система двофакторної автентифікації (2FA), яка значно підвищує безпеку, вимагаючи підтвердження особи за допомогою одноразового пароля або біометричних даних. Контроль доступу до ресурсів мережі реалізується через системи управління обліковими записами, які дозволяють точно налаштовувати права доступу користувачів і регулярно перевіряти їхню активність.

Для налаштування клієнт-серверної мережі із забезпеченням безпеки, синхронізацією часу та централізованим управлінням, було реалізовано ряд ключових мережевих параметрів на маршрутизаторі та комутаторах, а також налаштовано серверні служби на Windows Server. У даній архітектурі мережі

використовуються один маршрутизатор і два комутатори для зв’язку та обміну даними, а всі робочі станції підключені до домену Misha.net.

На маршрутизаторі R-231-27-1 виконана початкова конфігурація адресації, що дозволяє організувати доступ до мережі, налаштований протокол синхронізації часу NTP для підтримання точного часу на мережевих пристроях, а також конфігурація Syslog для централізованого журналювання подій. Це дає змогу адміністратору централізовано моніторити активність і стежити за мережею. Протокол SSH налаштовано для захищеного підключення до маршрутизатора, що дозволяє безпечно керувати пристроєм віддалено.

Комутатори SW-231-27-1 та SW-231-27-2 також налаштовані з використанням статичної адресації та підключені до маршрутизатора R-231-27-1 для забезпечення зв’язку з іншими пристроями в мережі. На комутаторах також налаштовано NTP для синхронізації часу з сервером, що допомагає підтримувати єдиний час через всю мережу, а також активовано Syslog для запису системних подій і моніторингу мережевого трафіку. Це дозволяє збирати журнали подій із комутаторів на централізованому сервері. Захищене підключення через SSH також доступне на комутаторах, що забезпечує віддалений доступ і управління.

На серверних пристроях Windows Server_1 та Windows Server_2 налаштовані основні сервіси для забезпечення управління мережею і безпеки. На обох серверах розгорнуто Active Directory (AD), що надає централізоване управління користувачами і доступом до мережевих ресурсів. Усі робочі станції в мережі підключено до домену Misha.net, що дозволяє спростити керування користувачами та доступом до ресурсів мережі. На серверах також налаштовано Syslog-сервер, який приймає журнали подій від маршрутизатора і комутаторів, дозволяючи централізовано відстежувати події та швидко реагувати на потенційні проблеми.

Додатково налаштовано Radius-сервер, який забезпечує централізовану аутентифікацію для доступу до мережевих ресурсів, а також покращує контроль за доступом до мережевих пристроїв. Цей сервер дає змогу контролювати доступ

і налаштовувати правила аутентифікації для підвищення безпеки в мережі. Для забезпечення точної синхронізації всіх пристроїв мережі, на серверах налаштований NTP-сервер, який забезпечує єдиний часовий стандарт, що важливо для систем журналювання та мережевої безпеки. Доступ до серверів захищено за допомогою SSH, що дозволяє віддалено виконувати управлінські задачі без ризику компрометації даних. Окрім основних функцій, на Windows Server_1 та Windows Server_2 також налаштовано DNS-сервери та реалізовано реплікацію між ними. DNS забезпечує перетворення доменних імен на IP-адреси, що дозволяє робочим станціям і сервісам у домені Misha.net зручно знаходити необхідні ресурси за іменами, замість IP-адрес. Це спрощує доступ до мережевих ресурсів і забезпечує стабільність підключень, оскільки дозволяє легко управляти доменними іменами та відповідними їм IP-адресами.

Реплікація між Server_1 та Server_2 реалізована для забезпечення безперервності роботи мережі та захисту даних у випадку відмови одного з серверів. Реплікація гарантує, що всі дані та налаштування, створені на одному сервері, автоматично копіюються на інший, забезпечуючи актуальність інформації в реальному часі. Це особливо важливо для Active Directory, де синхронізація користувацьких облікових записів, політик доступу та інших налаштувань дозволяє зберігати єдиний стан домену Misha.net. Реплікація також охоплює DNS-записи, що забезпечує доступність домену з будь-якого сервера навіть у разі збою одного з них.

Резервування DNS і реплікації Active Directory підвищують надійність і стійкість до збоїв у мережі. Завдяки цьому мережа залишається доступною, а користувачі можуть підключатися до ресурсів навіть за наявності технічних проблем на одному із серверів.

Іншим важливим аспектом захисту є регулярне оновлення програмного забезпечення та встановлення патчів безпеки, що запобігає використанню вразливостей у програмах. Для управління та моніторингу подій безпеки використовуються системи SIEM (Security Information and Event Management),

які об’єднують дані про події з різних компонентів мережі й аналізують їх для виявлення загроз і можливих інцидентів безпеки. SIEM надає можливість централізовано відстежувати та керувати подіями в мережі, а також швидко реагувати на інциденти.

На рівні кінцевих пристроїв, таких як комп'ютери користувачів і сервери, використовуються антивірусне програмне забезпечення та засоби захисту від шкідливого програмного забезпечення. Це дозволяє запобігти проникненню шкідливих програм, які можуть бути використані для крадіжки даних або порушення роботи мережі. Резервне копіювання є ще одним елементом підсистеми захисту: регулярні бекапи даних гарантують, що інформація не буде втрачена навіть у разі атаки або збою. Завдяки резервному копіюванню можна швидко відновити роботу мережі після непередбачуваних подій.

Загалом, підсистема захисту клієнт-серверної мережі організації є багаторівневим комплексом, який поєднує інструменти контролю доступу, системи виявлення загроз, технології шифрування, антивірусні засоби та систему управління подіями безпеки. Всі ці елементи працюють разом для забезпечення надійного захисту даних та ресурсів мережі, мінімізуючи ризики несанкціонованого доступу і збоїв у роботі. Такий підхід дозволяє підвищити рівень безпеки мережі, забезпечити її стабільність та ефективність в умовах постійно зростаючих кіберзагроз.

Висновки до другого розділу:

У другому розділі магістерської роботи було здійснено детальний аналіз побудови клієнт-серверної мережі, зокрема її топології, вибору обладнання та розробки схеми адресації. Було визначено, що клієнт-серверна мережа найкраще відповідає потребам організації завдяки можливості централізованого управління ресурсами, що забезпечує більшу безпеку та зручність обслуговування. При побудові топології мережі враховано можливості розширення та оптимізації трафіку, що підвищує продуктивність мережевої інфраструктури.

Також було розглянуто необхідне мережеве обладнання — маршрутизатори та комутатори Cisco, які відповідають вимогам до продуктивності й надійності мережі. Схема адресації, що передбачає поділ мережі на підмережі, дозволяє оптимізувати трафік та підвищити рівень безпеки. Було впроваджено протоколи та сервіси для синхронізації часу (NTP), журналювання подій (Syslog), автентифікації (Radius) та доменної служби Active Directory.

Загалом, розроблена клієнт-серверна мережа відповідає сучасним стандартам безпеки та забезпечує стабільний доступ до ресурсів мережі для користувачів організації.

РОЗДІЛ 3. НАЛАГОДЖЕННЯ ТА ТЕСТУВАННЯ ПІДСИСТЕМИ ЗАХИСТУ КЛІЄНТ-СЕРВЕРНОЇ МЕРЕЖІ

3.1 Базове налагодження пристроїв мережі

У процесі базового налагодження пристроїв мережі виконано конфігурацію ключових компонентів, включаючи адресацію, ідентифікацію пристроїв, налаштування мережевих сервісів, а також впровадження DHCP і DNS з доменним іменем Misha.net.

Адресація пристроїв

Було створено дві локальні мережі:

192.168.1.0/24 — для першої підмережі;

192.168.2.0/24 — для другої підмережі.

Для всіх пристроїв мережі налаштовано статичну IP-адресацію в межах відповідних підмереж. Кожен пристрій отримав унікальну IP-адресу для забезпечення коректного функціонування мережі.

Ідентифікація пристроїв

Для зручності адміністрування та моніторингу всі пристрої були перейменовані згідно зі стандартами ідентифікації:

Маршрутизатори: отримали назви R-231-27-1 та R-231-27-2.

Комутатори: були перейменовані на SW-231-27-1 і SW-231-27-2.

Комп'ютери: отримали імена WS-231-27-1, WS-231-27-2 та WS-231-27-3.

Сервери: для двох серверів було також присвоєно унікальні імена для їхньої ідентифікації в мережі.

Налаштування DHCP

Для автоматизації призначення IP-адрес клієнтам мережі був налаштований DHCP-сервер. Конфігурація включала:

Визначення діапазонів виділення IP-адрес для обох підмереж:

Для підмережі 192.168.1.0/24: діапазон адрес 192.168.1.100–192.168.1.254.

Для підмережі 192.168.2.0/24: діапазон адрес 192.168.2.100–192.168.2.254.

Вказівку шлюзу за замовчуванням (default gateway) для кожної підмережі.

Налаштування резервування IP-адрес для критичних пристроїв, таких як сервери та маршрутизатори.

Налаштування DNS

Для забезпечення зручності доступу до мережевих ресурсів було налаштовано DNS-сервер із доменним іменем Misha.net. Основні налаштування включали:

Призначення основного доменного імені мережі — Misha.net.

Створення записів для імен хостів, наприклад:

R-231-27-1.Misha.net — для основного маршрутизатора;

WS-231-27-1.Misha.net — для робочих станцій;

Server1.Misha.net — для серверів.

Визначення DNS-сервера як основного джерела для розв’язання імен у локальній мережі.

Налаштування мережевих сервісів

Для забезпечення функціональності мережі виконано налаштування таких сервісів:

NTP (Network Time Protocol):

На маршрутизаторах, серверах і критичних пристроях налаштовано синхронізацію часу за допомогою NTP.

Це забезпечило узгоджений системний час, що важливо для логування та роботи мережевих протоколів.

Syslog:

Налаштовано централізоване збирання системних журналів на виділеному сервері Syslog.

Це забезпечило моніторинг мережевих подій і швидке виявлення потенційних проблем.

SSH (Secure Shell):

Для всіх пристроїв мережі активовано SSH для забезпечення безпечного віддаленого доступу.

Налаштовано захищені облікові записи та паролі для унеможливлення несанкціонованого доступу.

Перевірка конфігурацій

Після виконання налаштувань було проведено тестування:

Перевірено доступність пристроїв за допомогою команд ping між підмережами.

Перевірено функціонування DHCP, зокрема автоматичне отримання IP-адрес клієнтами.

Виконано тестування роботи DNS, включаючи розв’язання доменних імен.

Атестовано синхронізацію часу через NTP, а також перевірено логування подій на сервері Syslog.

Верифіковано безпечність і доступність SSH-з’єднань для кожного пристрою.

У результаті виконаних робіт було налаштовано стабільну та функціональну мережу, що включає автоматичне розподілення адрес, підтримку доменних імен, синхронізацію часу та безпечний віддалений доступ до пристроїв.

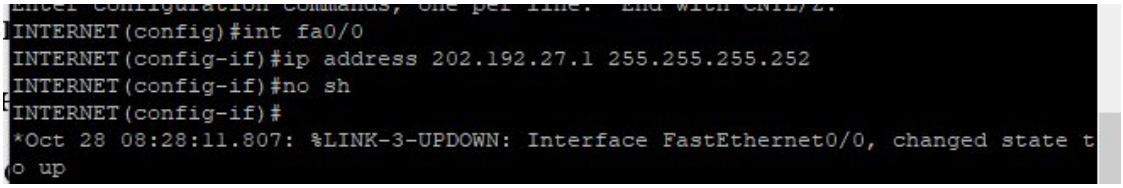


Рисунок 3.1 – Задана адреса Internet

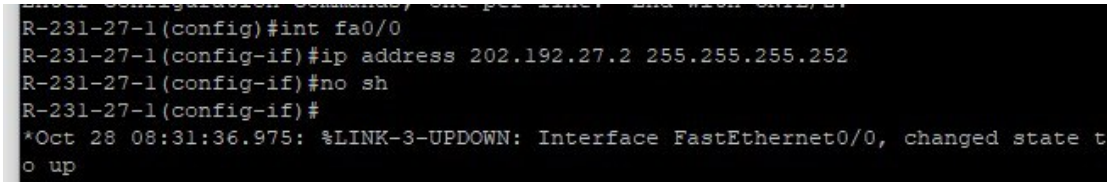


Рисунок 3.2 – Задана адреса R -231– 27– 1

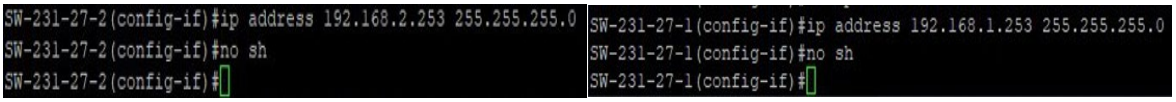


Рисунок 3.3 – Задана адреса SW– 231– 27– 1, SW– 231– 27– 2

```
WS-231-27-2> ip 192.168.1.250 192.168.1.1
Checking for duplicate address...
WS-231-27-2 : 192.168.1.250 255.255.255.0 gateway 192.168.1.1

WS-231-27-2> save
Saving startup configuration to startup.vpc
.  done

WS-231-27-2>
```

WS-231-27-3 - PuTTY

```
WS-231-27-3> show ip

NAME        : WS-231-27-3[1]
IP/MASK     : 192.168.2.251/24
GATEWAY     : 192.168.2.254
DNS         :
MAC         : 00:50:79:66:68:02
LPORT       : 20023
RHOST:PORT  : 127.0.0.1:20024
MTU         : 1500
```

Рисунок 3.4 – Налаштовані адреси на комп’ютерах

WS-231-27-1 - PuTTY

```
^C
WS-231-27-1> ping 192.168.1.254

84 bytes from 192.168.1.254 icmp_seq=1 ttl=255 time=9.404 ms
84 bytes from 192.168.1.254 icmp_seq=2 ttl=255 time=6.712 ms
84 bytes from 192.168.1.254 icmp_seq=3 ttl=255 time=8.269 ms
84 bytes from 192.168.1.254 icmp_seq=4 ttl=255 time=12.425 ms
84 bytes from 192.168.1.254 icmp_seq=5 ttl=255 time=14.433 ms

WS-231-27-1> ping 192.168.1.250

84 bytes from 192.168.1.250 icmp_seq=1 ttl=64 time=0.713 ms
84 bytes from 192.168.1.250 icmp_seq=2 ttl=64 time=1.321 ms
84 bytes from 192.168.1.250 icmp_seq=3 ttl=64 time=1.093 ms
84 bytes from 192.168.1.250 icmp_seq=4 ttl=64 time=1.167 ms
84 bytes from 192.168.1.250 icmp_seq=5 ttl=64 time=2.237 ms
```

Рисунок 3.5 – Всі адреси на комп’ютері пінгуються

WS-231-27-3 - PuTTY

```
WS-231-27-3> ping 192.168.2.248

84 bytes from 192.168.2.248 icmp_seq=1 ttl=128 time=2.904 ms
84 bytes from 192.168.2.248 icmp_seq=2 ttl=128 time=2.033 ms
84 bytes from 192.168.2.248 icmp_seq=3 ttl=128 time=8.170 ms
84 bytes from 192.168.2.248 icmp_seq=4 ttl=128 time=13.198 ms
84 bytes from 192.168.2.248 icmp_seq=5 ttl=128 time=6.580 ms

WS-231-27-3> ping 192.168.2.254

84 bytes from 192.168.2.254 icmp_seq=1 ttl=255 time=17.628 ms
84 bytes from 192.168.2.254 icmp_seq=2 ttl=255 time=6.513 ms
84 bytes from 192.168.2.254 icmp_seq=3 ttl=255 time=13.663 ms
84 bytes from 192.168.2.254 icmp_seq=4 ttl=255 time=3.524 ms
84 bytes from 192.168.2.254 icmp_seq=5 ttl=255 time=3.401 ms

WS-231-27-3>
```

Рисунок 3.6 – Всі адреси на комп’ютері пінгуються

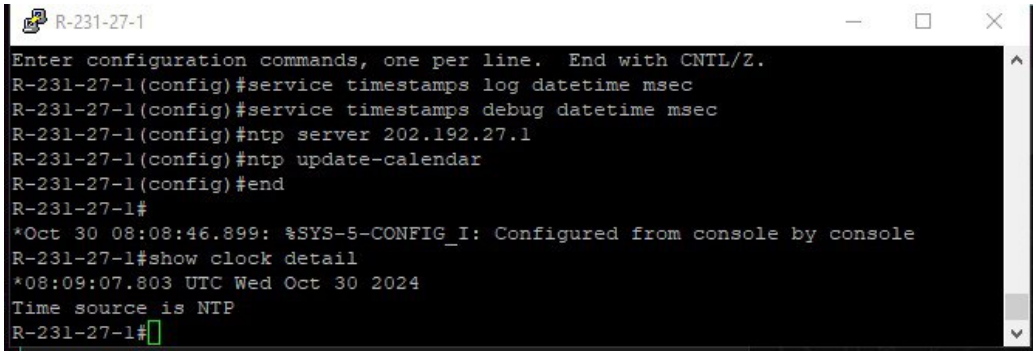


Рисунок 3.7 – Налаштування ntp та демонстрація, що він працює

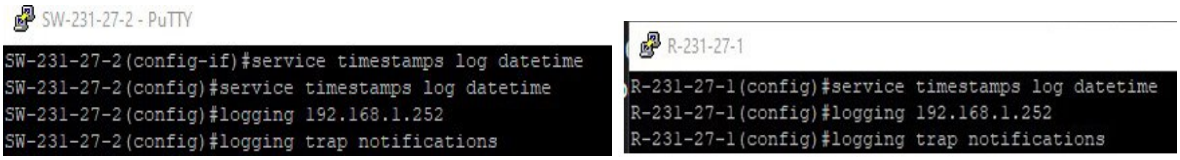


Рисунок 3.8 – Налаштування syslog

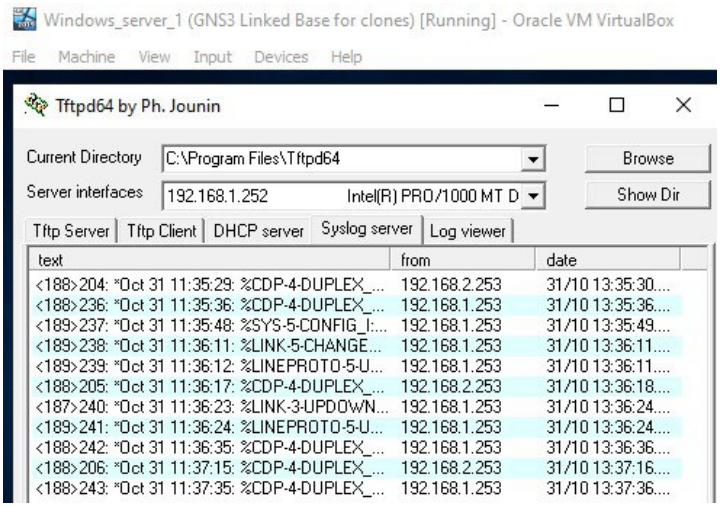


Рисунок 3.9 – Syslog працює

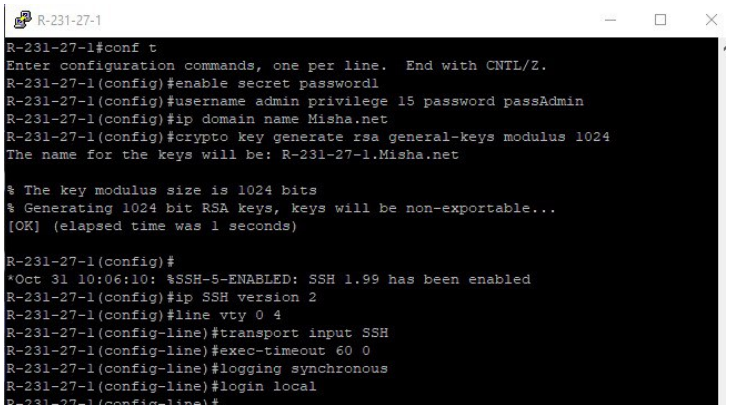


Рисунок 3.10 – Налаштований ssh на R– 231– 27– 1

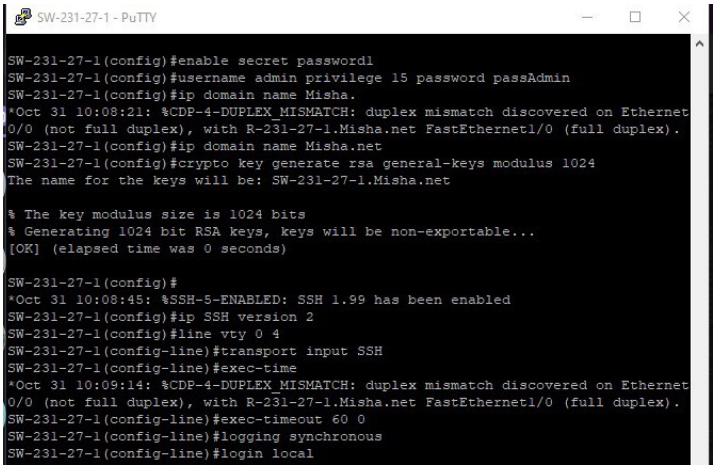


Рисунок 3.11 – Налаштований ssh на SW– 231– 27– 1

3.2 Налагодження Active Directory і робочих станцій

Спершу ми конфігурували сервер для роботи з постійною IP-адресою 192.168.1.252. Це гарантує надійний доступ до сервера та запобігає можливим конфліктам з іншими пристроями у мережі.

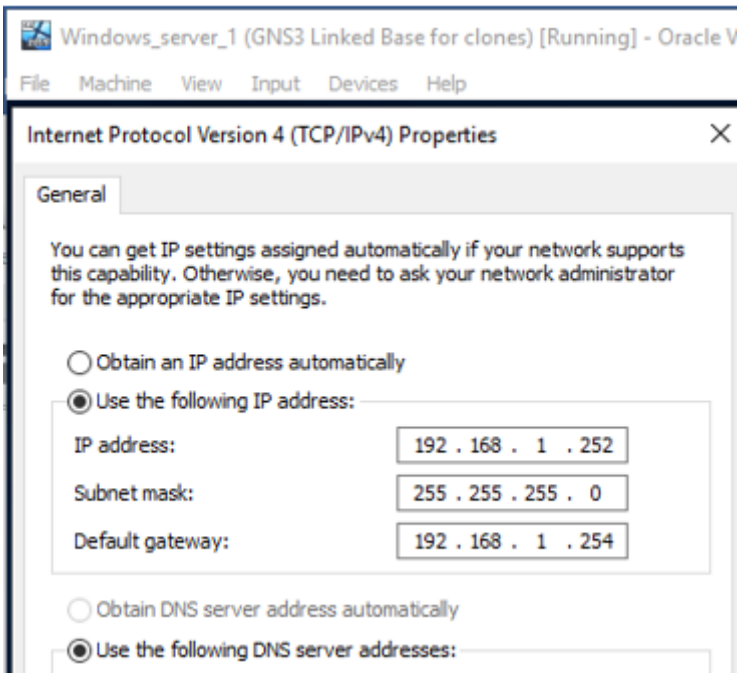


Рисунок 3.12 – Налаштування IP адреси

Далі ми налаштували сервер, додавши до нього потрібні ролі. Серед них були служби Active Directory Domain Services (AD DS), DNS-сервер та DHCP-сервер.

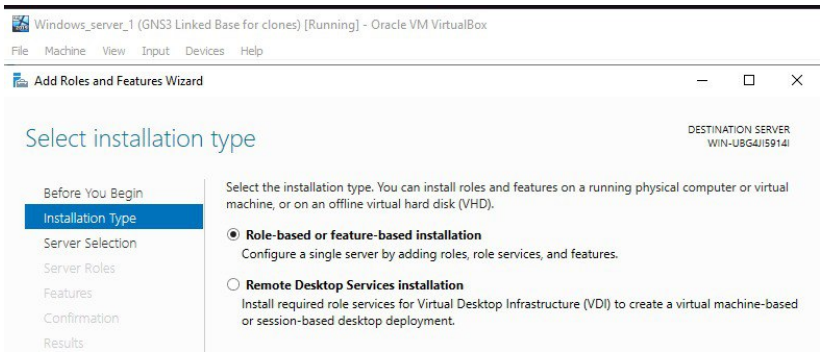


Рисунок 3.13 – Додавання ролей

Під час налаштування ми вибрали сервер для розгортання цих ролей, що забезпечило коректну роботу домену та мережевих служб.

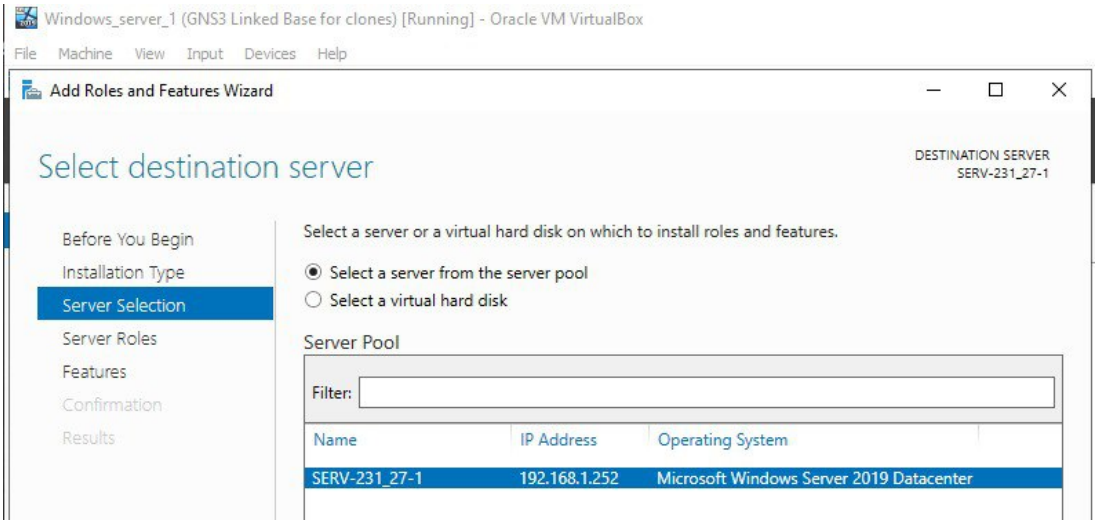


Рисунок 3.14 – Вибір сервера призначення

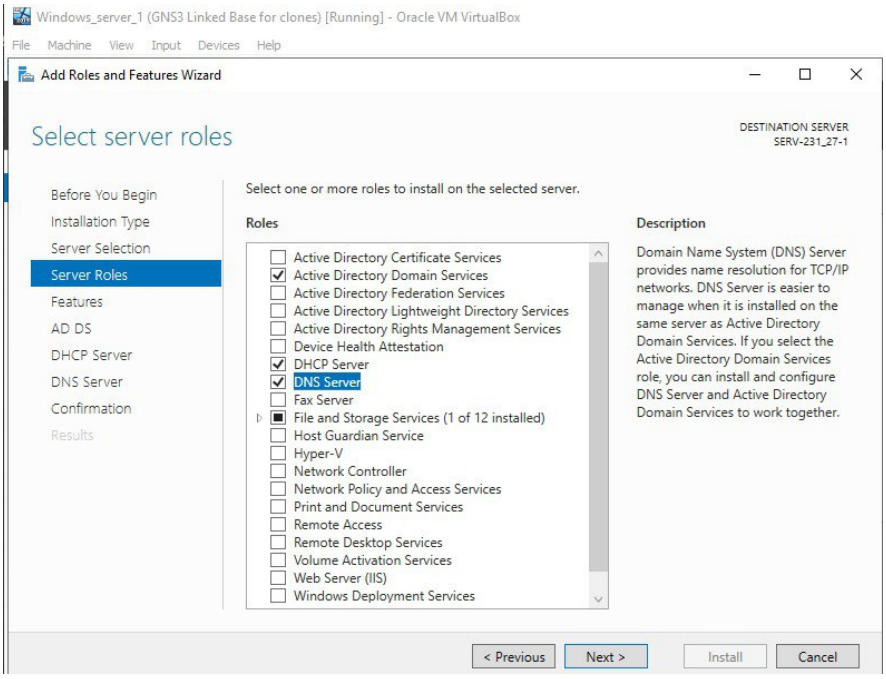


Рисунок 3.15 – Додавання ролей

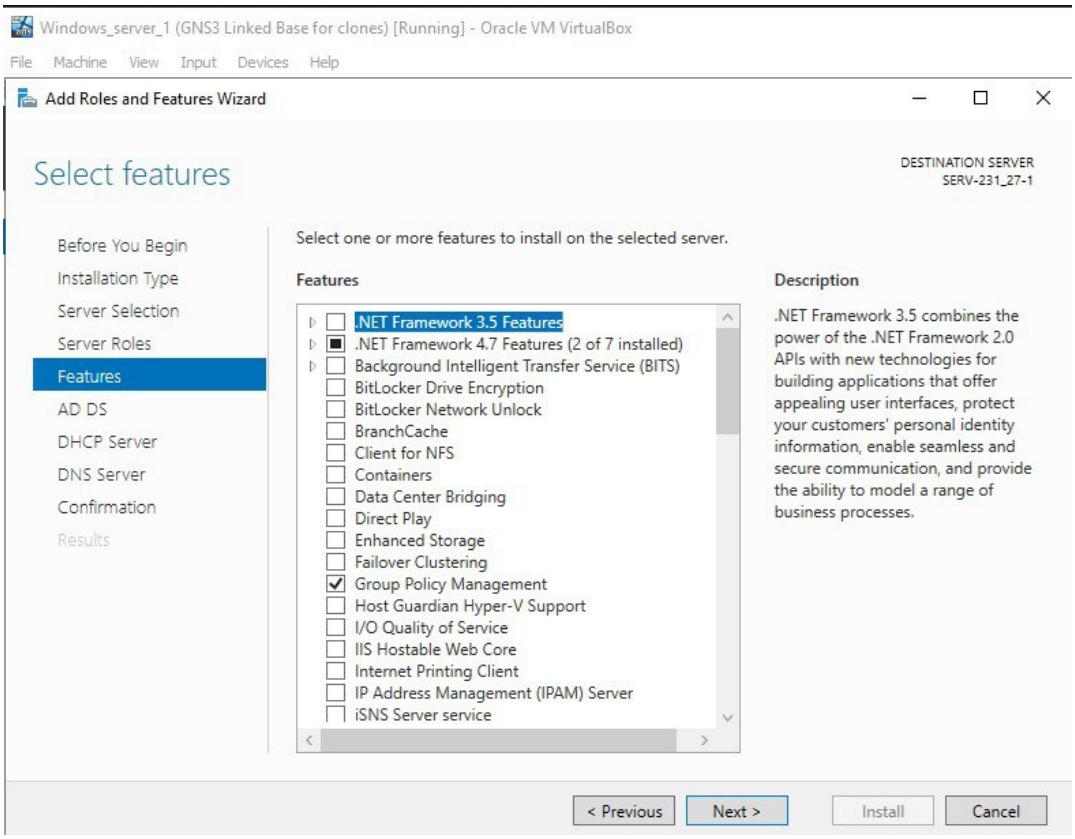


Рисунок 3.16 – Додавання ролей

Для організації домену був створений новий домен із назвою Misha.net, який надає можливість централізованого управління користувачами, пристроями та іншими мережевими ресурсами.

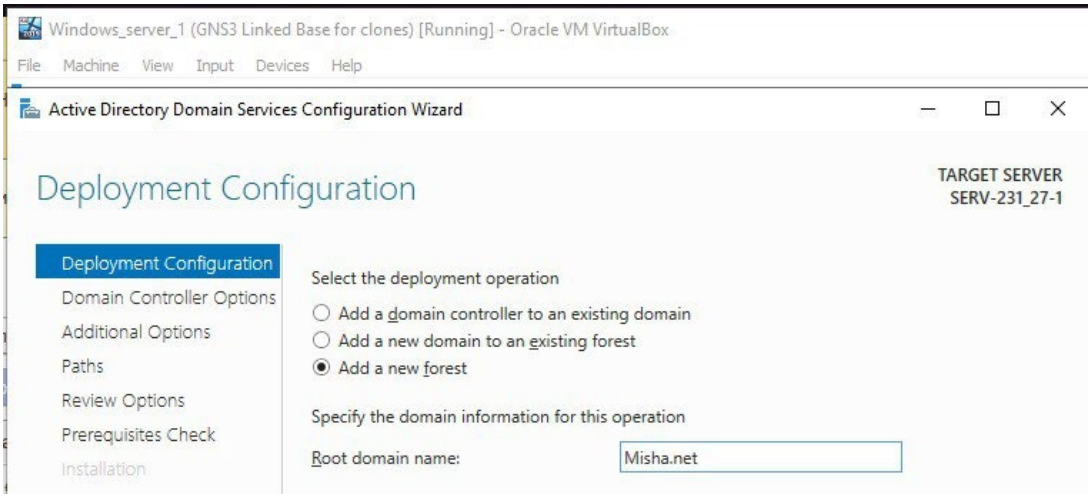


Рисунок 3.17 – Створення домену

Потім була налаштована область DHCP під назвою Lan_scope, яка забезпечує автоматичне призначення IP-адрес клієнтським пристроям у мережі.

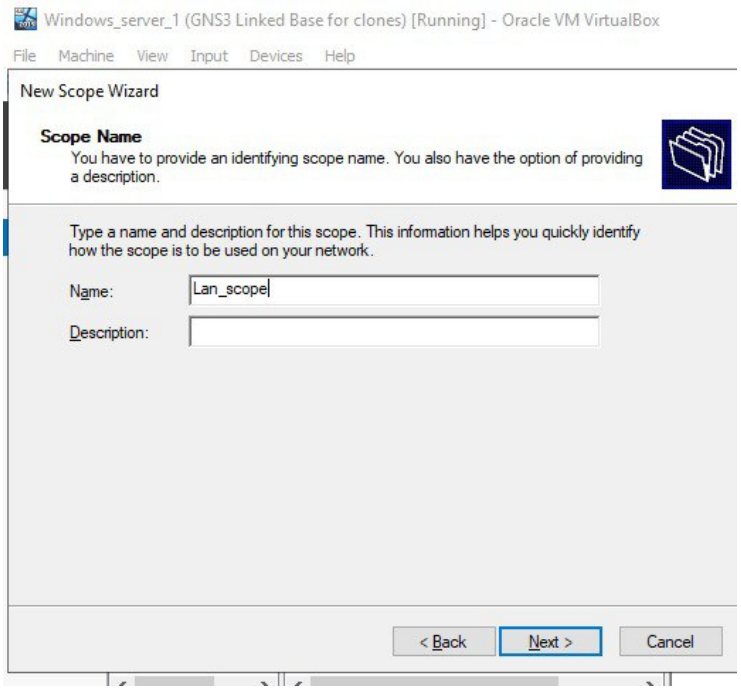


Рисунок 3.18 – Створення області Lan_scope

У рамках області Lan_scope був заданий діапазон IP-адрес для клієнтських пристроїв, наприклад, від 192.168.19.1 до 192.168.19.254, який використовується для автоматичного призначення.

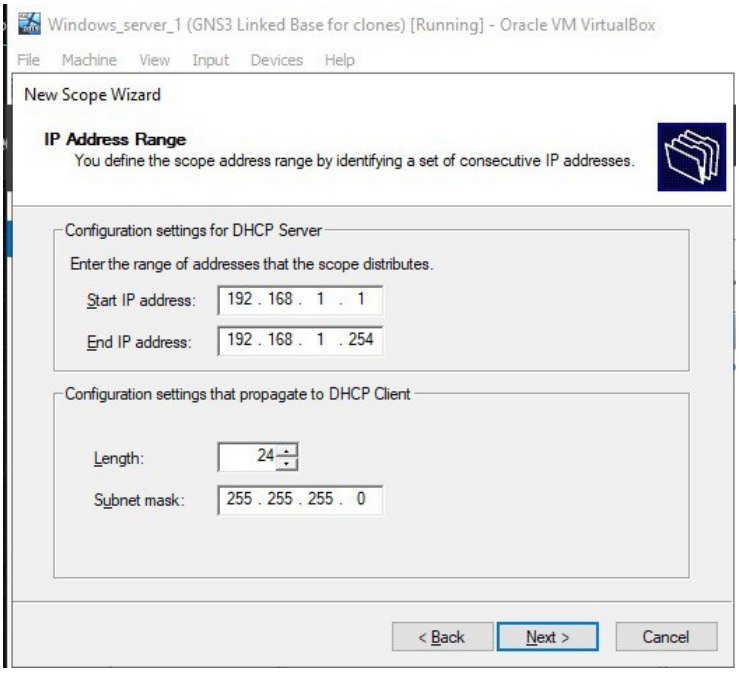


Рисунок 3.19 – Визначення діапазону роздачі адрес

З діапазону розподілу були виключені статичні IP-адреси, які призначені для сервера та інших критичних пристроїв, зокрема 192.168.19.250–192.168.19.254.

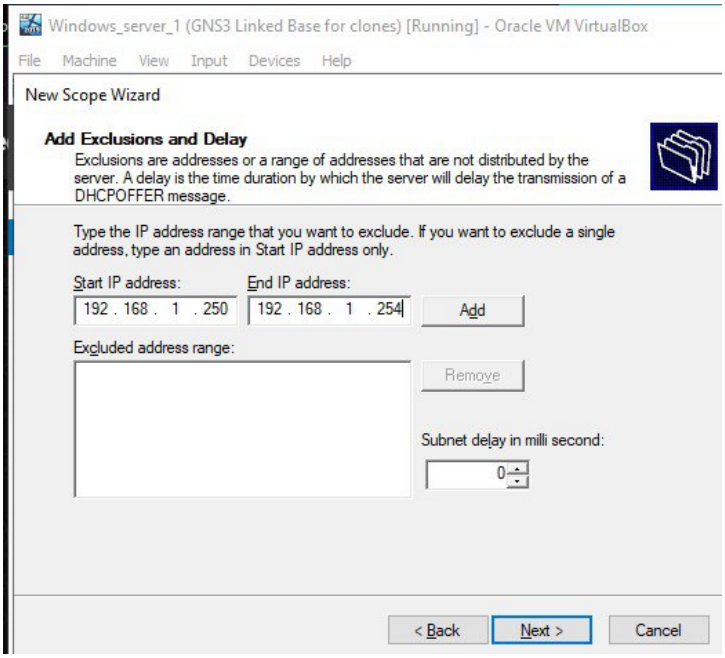


Рисунок 3.20 – Виключення статичних адрес з діапазону роздачі

Був встановлений термін оренди IP-адрес (lease time) для клієнтських пристроїв, що складає 8 днів. Це гарантує своєчасне оновлення адрес та мінімізує ризик виникнення конфліктів

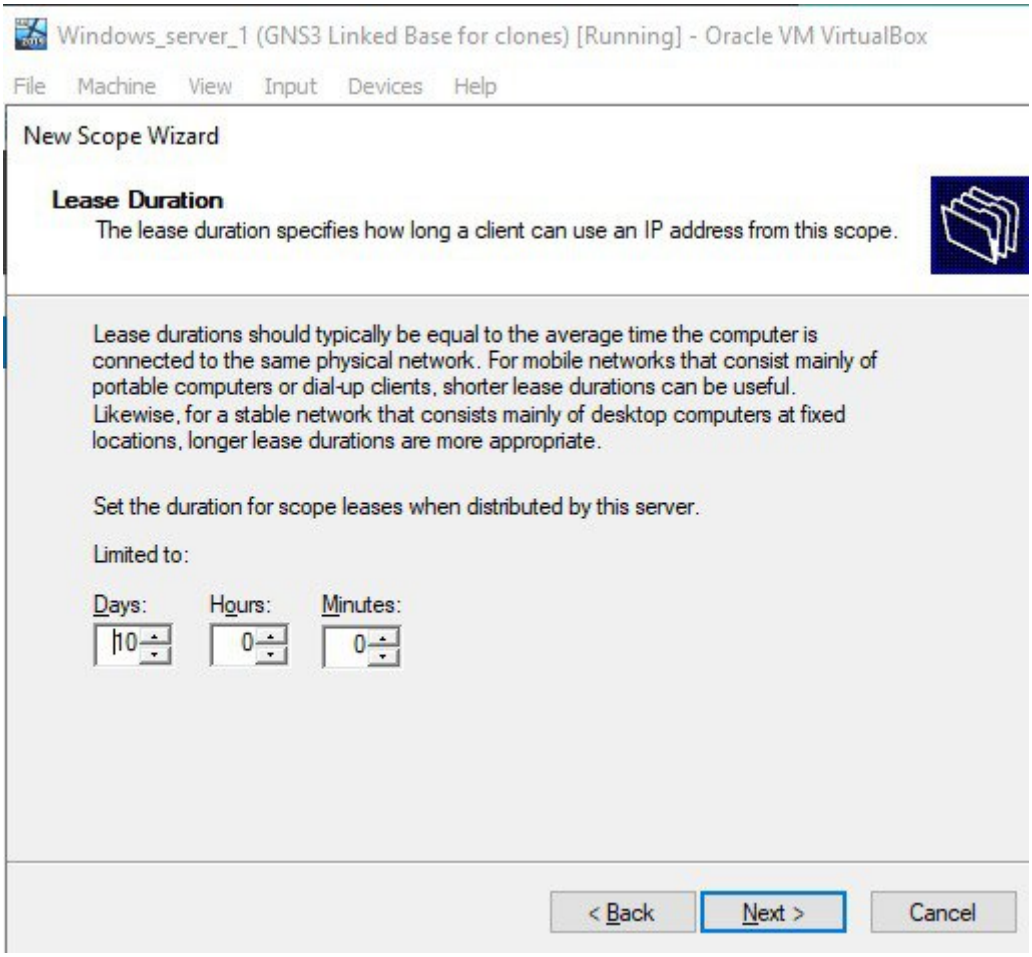
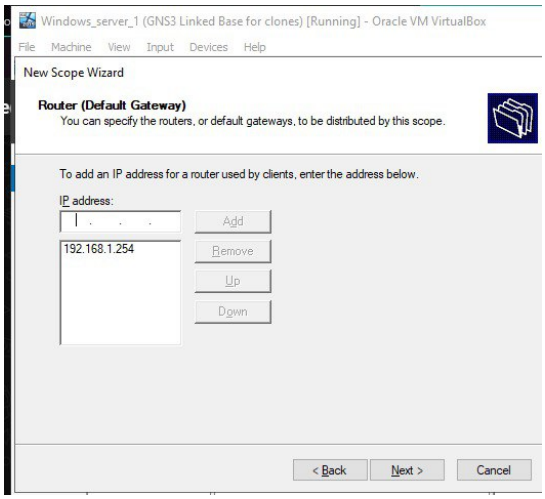
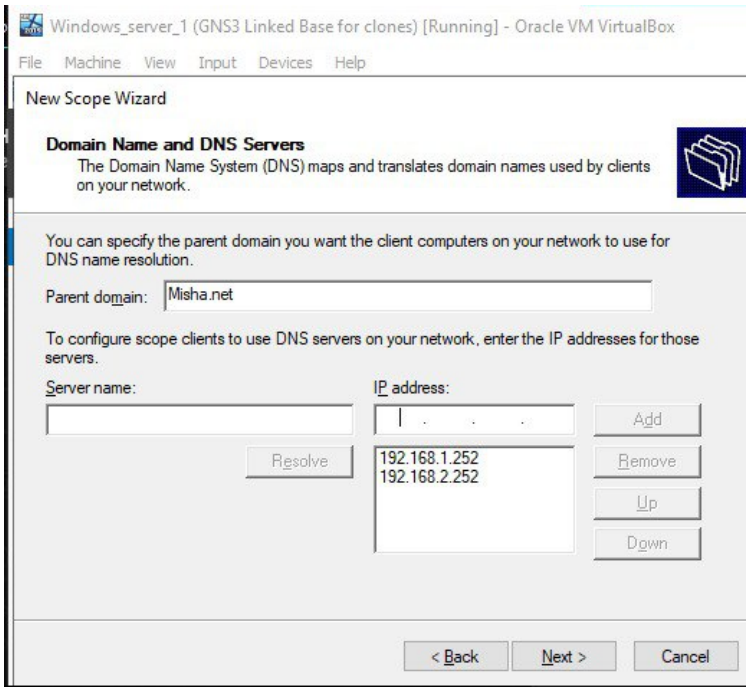


Рисунок 3.21 – Вибір терміну часу видачі адрес

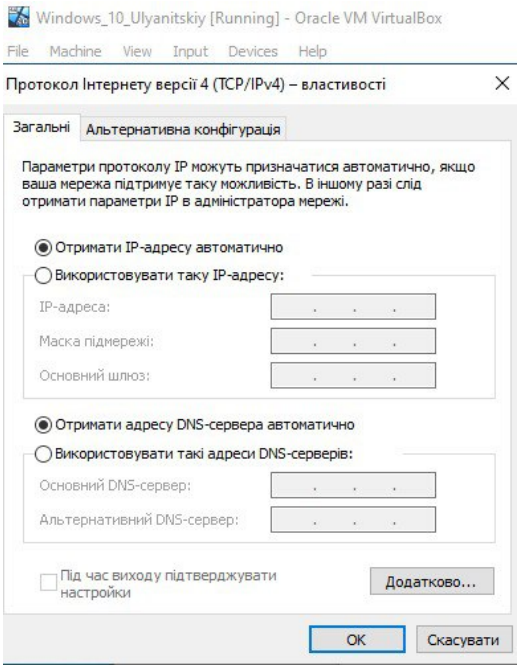
У налаштуваннях DHCP був вказаний шлюз за замовчуванням (default gateway) з адресою 192.168.1.254, що забезпечує з'єднання з Інтернетом.



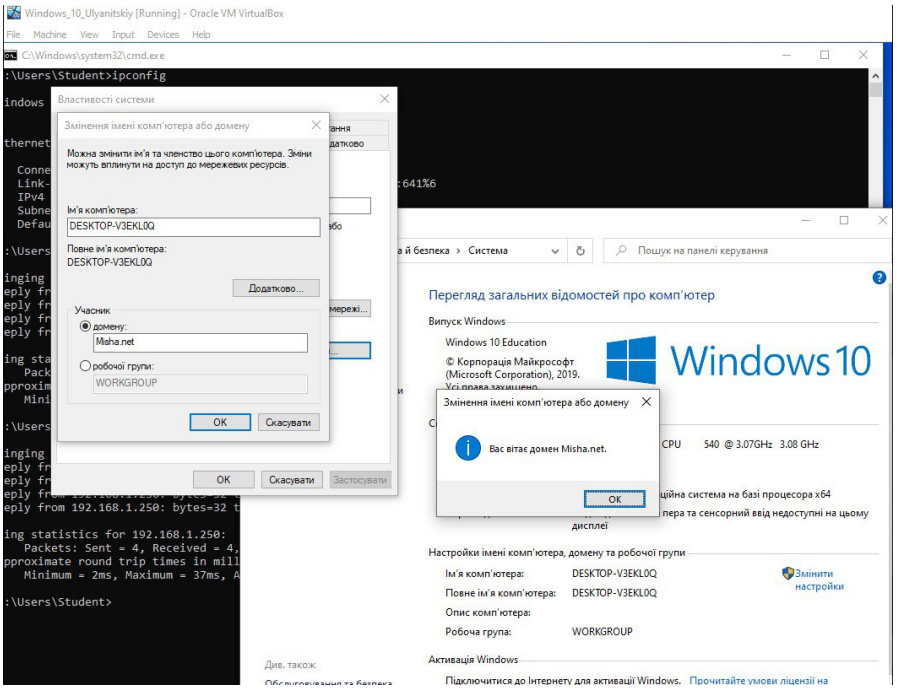
До налаштувань DHCP були додані адреси DNS-серверів, що дає можливість клієнтським пристроям використовувати DNS для розв'язання доменних імен. Основні DNS-сервери мають адреси 192.168.1.251 та 192.168.2.251.



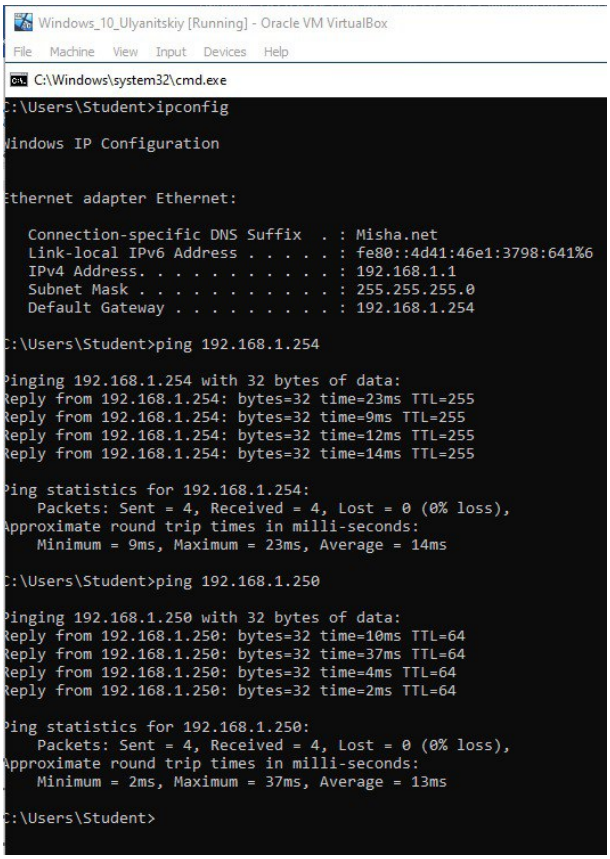
Клієнтська станція (Windows 10) була налаштована на автоматичне отримання IP-адрес через DHCP.



Станцію з ОС Windows 10 було підключено до домену Misha.net, що дозволяє використовувати централізоване управління користувачами та ресурсами.

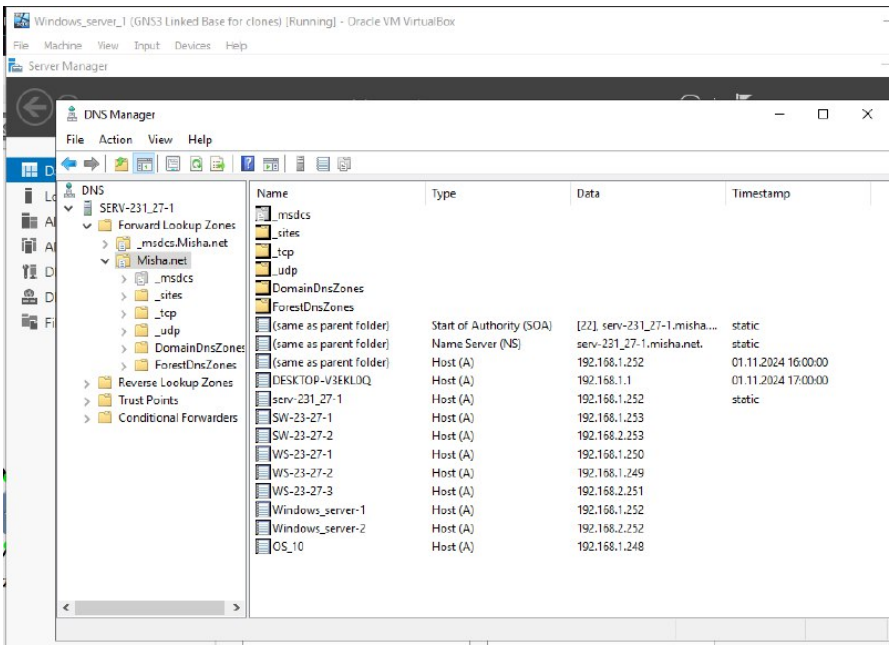


25 – Підключення станції з ОС Windows 10 до домену

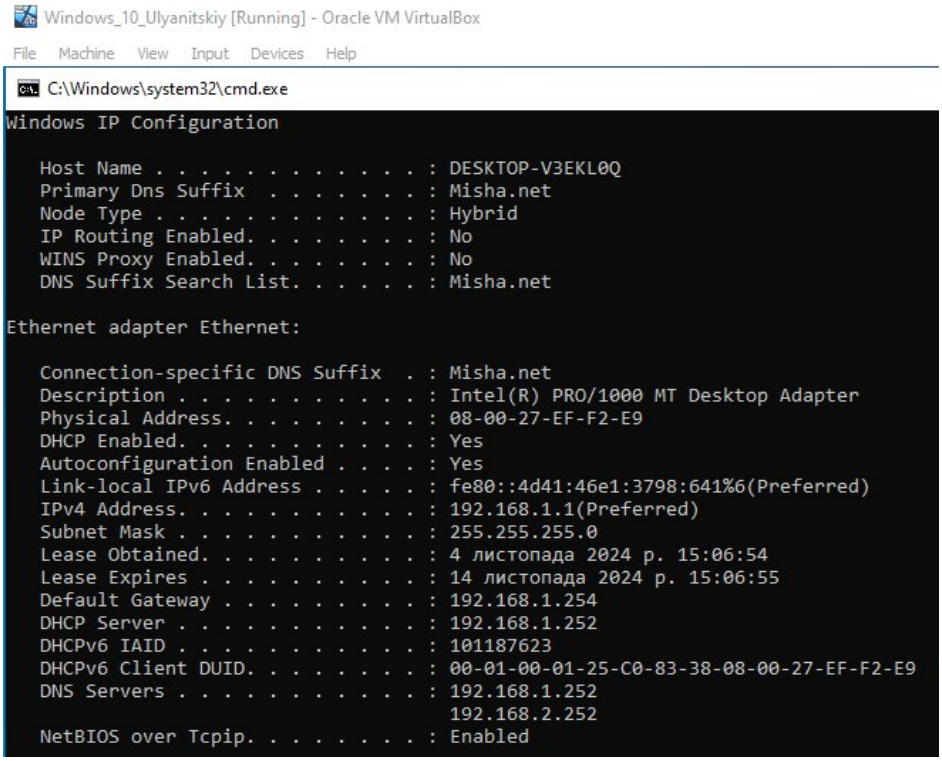


26 – Перевірено чи проходять пінги на Windows 10

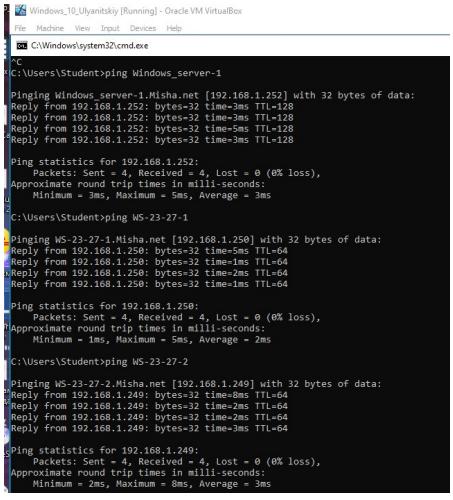
Були створені відповідні DNS-записи для забезпечення коректного розв'язання імен у домені Misha.net.



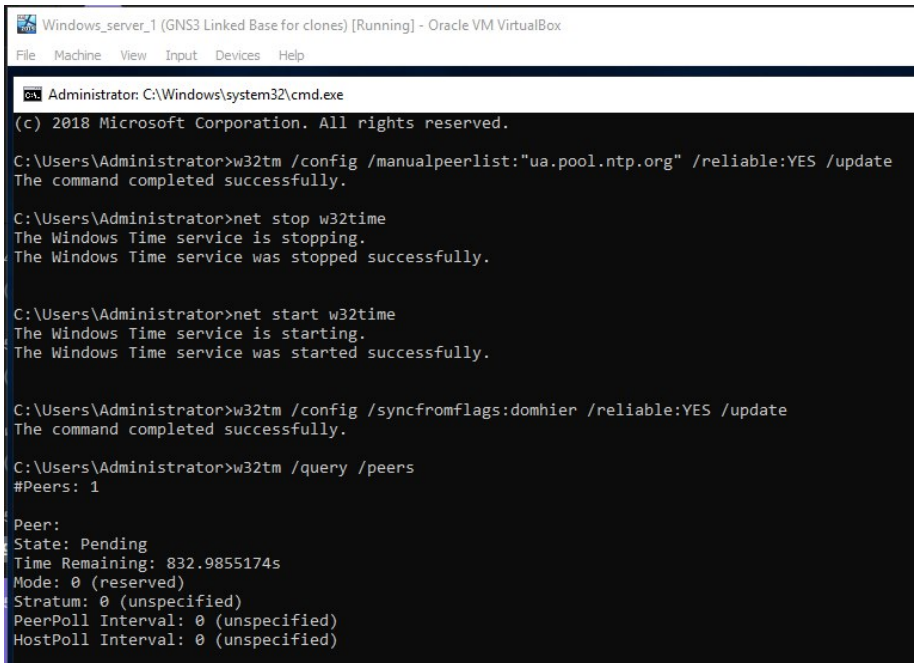
Було виконано перевірку всіх налаштувань, щоб упевнитися, що всі пристрої правильно отримують IP-адреси та мають змогу підключатися до домену.



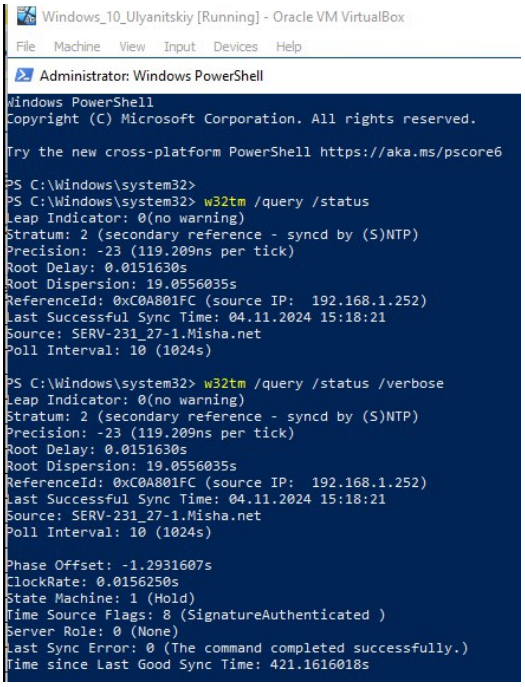
Було перевірено з'єднання з маршрутизатором і забезпечено доступ до Інтернету для всіх клієнтських станцій.



Для синхронізації часу в мережі було налаштовано NTP-сервер на контролері домену, який забезпечує точний час для всіх клієнтських пристроїв.

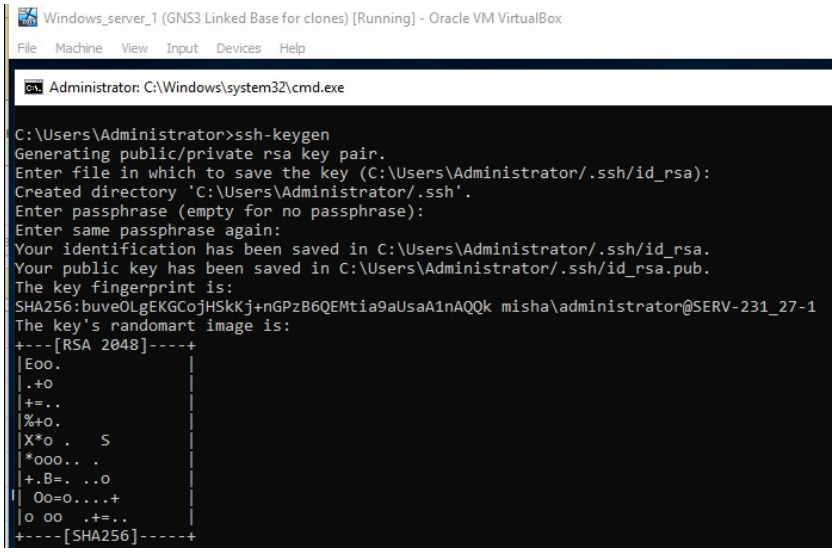


Клієнтські станції з ОС Windows 10 були конфігуровані для автоматичної синхронізації часу з NTP-сервером домену.

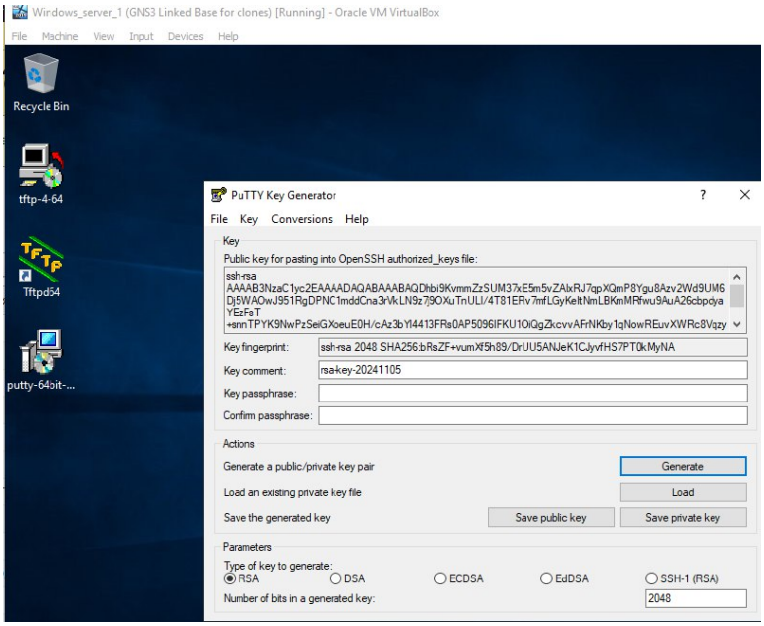


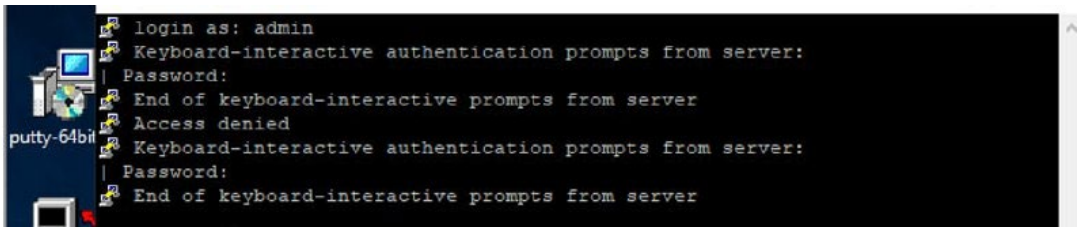
NTP на ОС Windows 10

Далі було налаштовано SSH (Secure Shell) для забезпечення безпечного віддаленого доступу до серверів і мережевих пристроїв. Це дозволяє адміністратору виконувати конфігураційні зміни та керувати пристроями через зашифроване з’єднання, що забезпечує захист даних від несанкціонованого доступу.



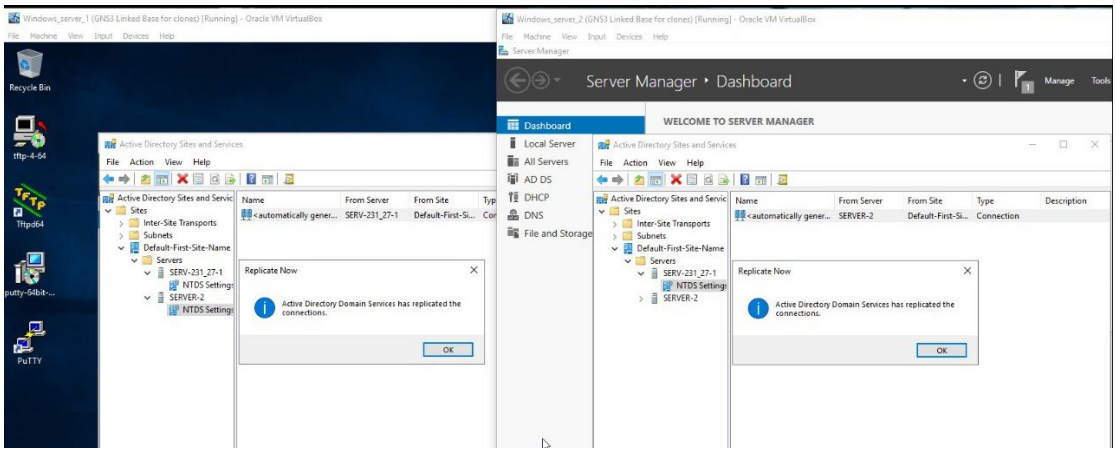
cmd





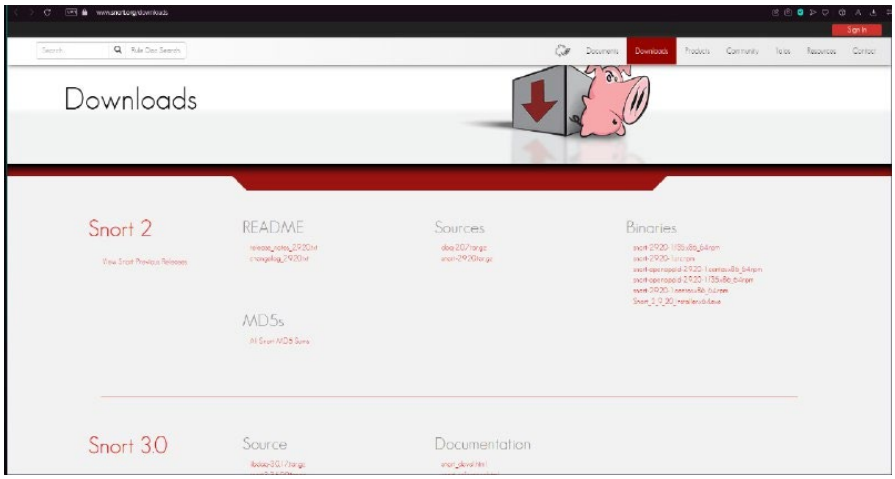
SSH під’єднався до маршрутизатора

Далі була налаштована реплікація між серверами, що дозволяє забезпечити синхронізацію даних та налаштувань між основним сервером і резервним. Всі аналогічні конфігурації, включаючи налаштування DHCP, DNS, NTP та SSH, були повторно застосовані на другому сервері для забезпечення безперервності роботи та відмовостійкості мережевих сервісів.

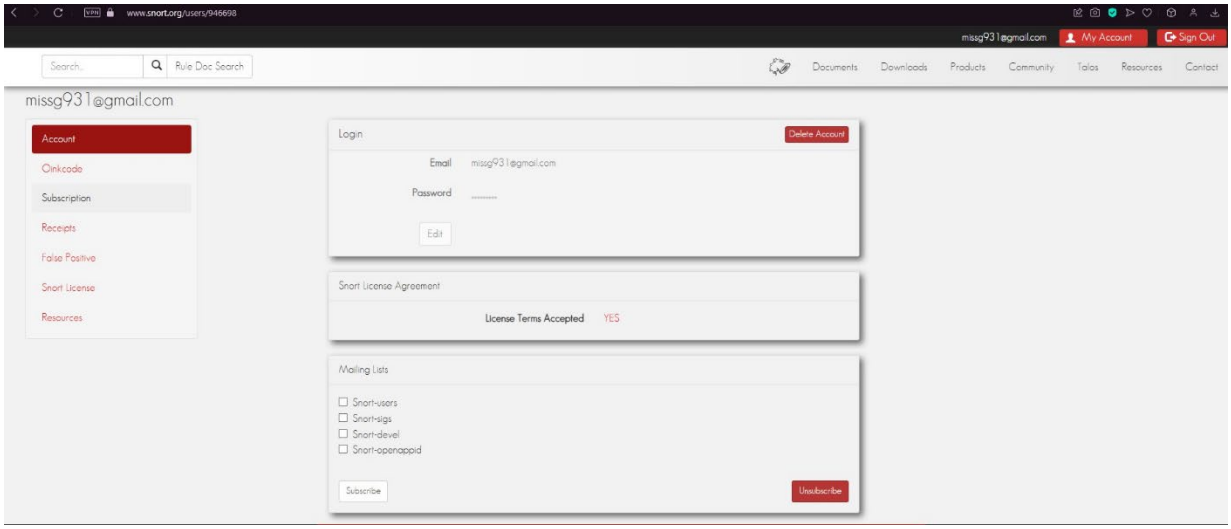


3.3 Встановлення програми Snort

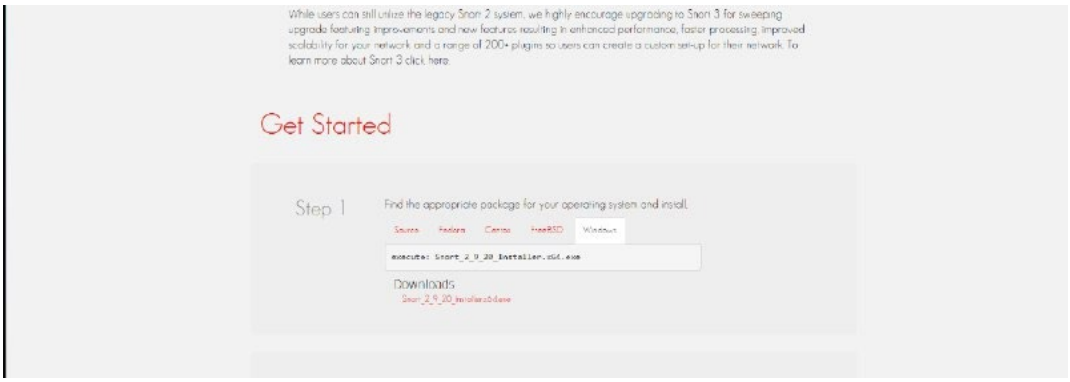
Щоб завантажити програму Snort, потрібно перейти на офіційний вебсайт за посиланням: https://www.snort.org/downloads і вибрати відповідну версію програми. На рисунку 3.36 показано сторінку сайту Snort із доступними варіантами версій для завантаження.



Щоб отримати доступ до ширшого вибору версій для завантаження, користувачу потрібно зареєструватися на сайті Snort. На рисунку 3.37 представлено інтерфейс особистого кабінету користувача.



Вибираємо версію, сумісну з операційною системою Windows, і приступаємо до її підготовки для встановлення. У нашому випадку це версія 2.9.20 (див. рисунок 3.38).



8 – Встановлення версії Snort 2.9.20 для Windows

На першому етапі необхідно прийняти умови ліцензійної угоди (див. рисунок 3.39).

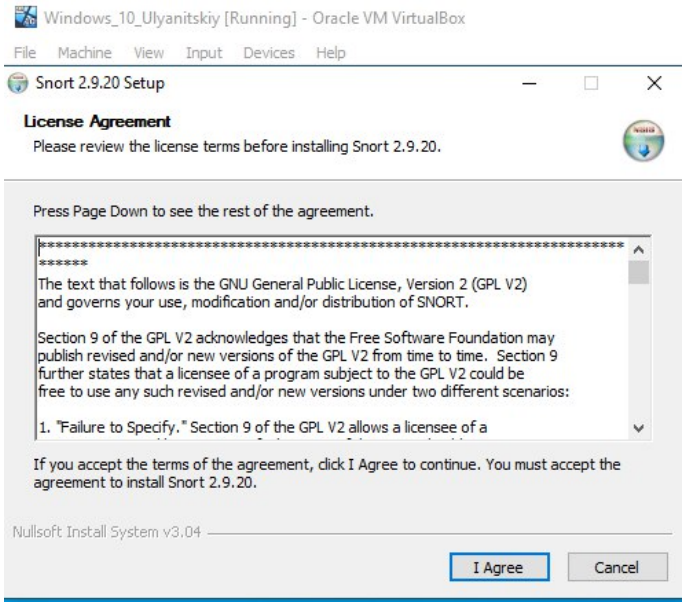


Рисунок 3.39 – Перший крок встановлення Snort

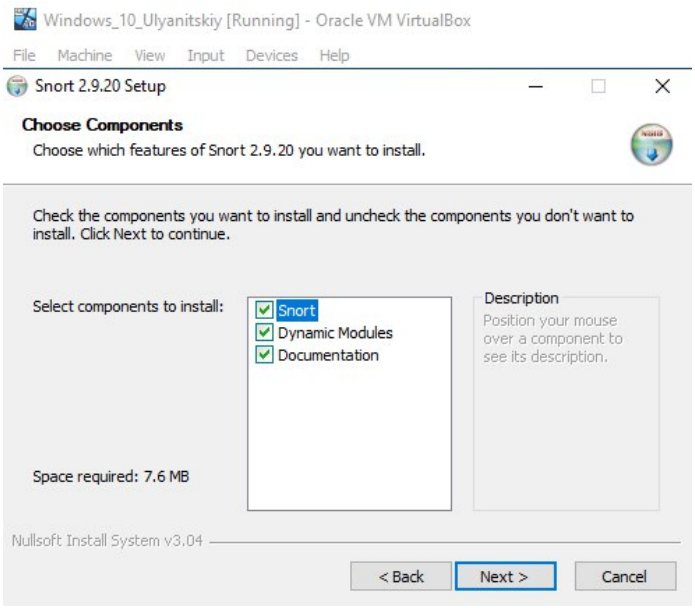


Рисунок 3.40 – Вибір компонентів Snort

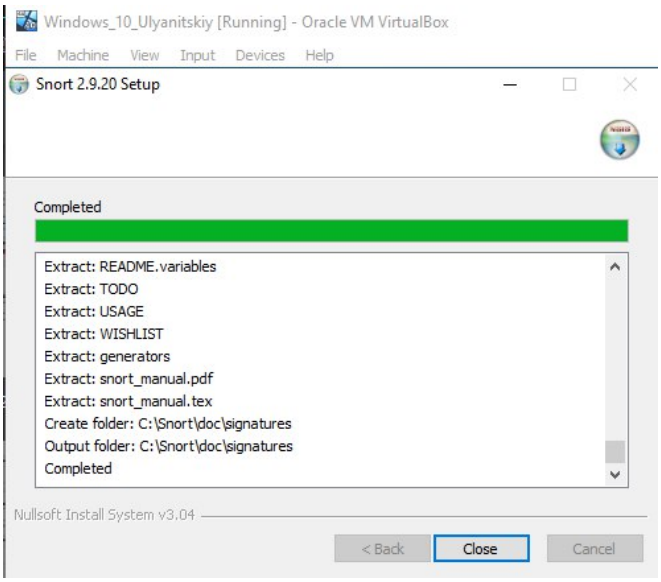
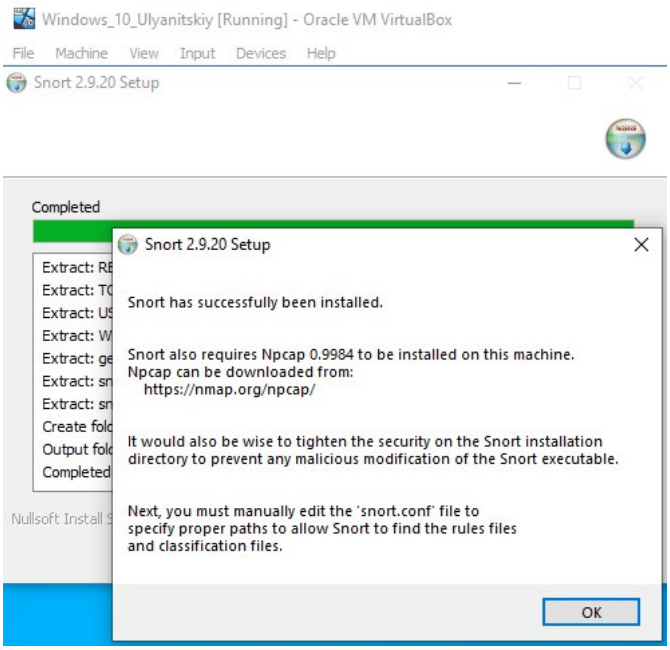


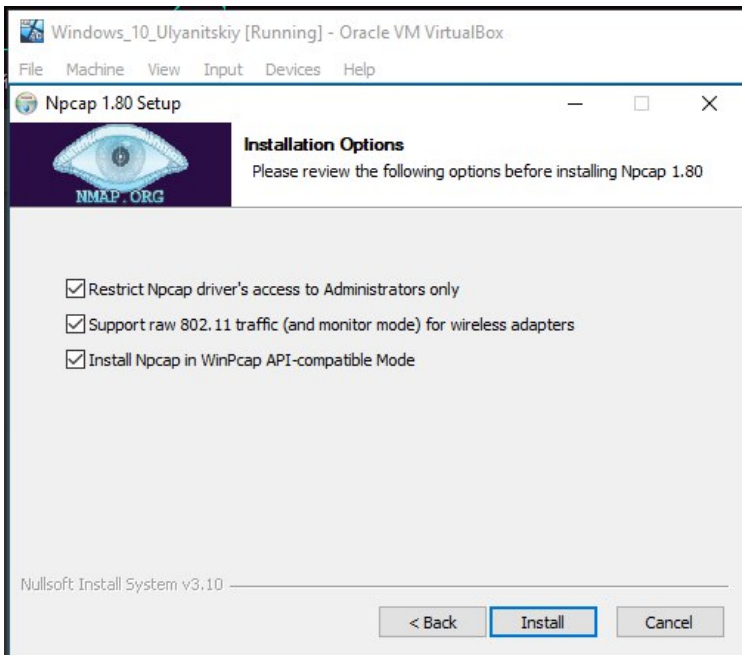
Рисунок 3.41 – Успішне встановлення Snort

npcap з офіційного сайту www.nmap.org.



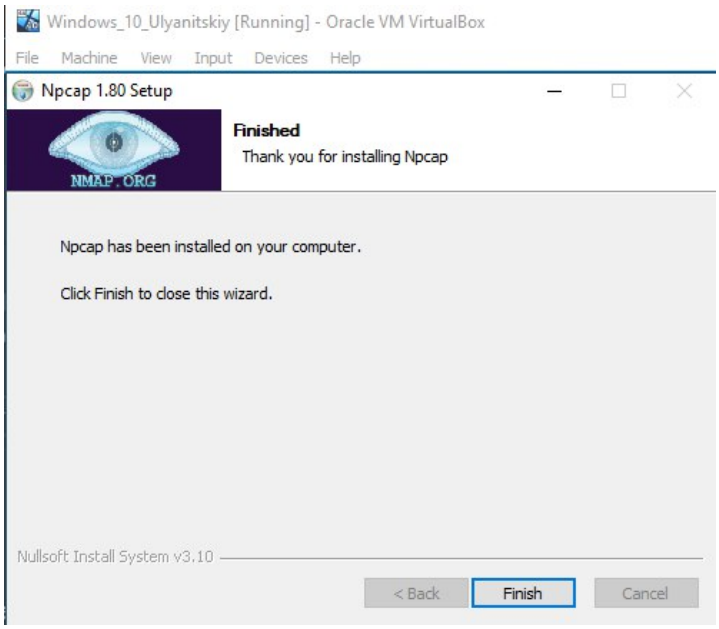
Snort 2.9.20 завершено

Переходимо на сайт www.nmap.org/npcap/ і завантажуємо необхідний додаток для програми Snort. Потім розпочинаємо процес установки, обираючи всі три доступні параметри (див. рисунок 3.43).



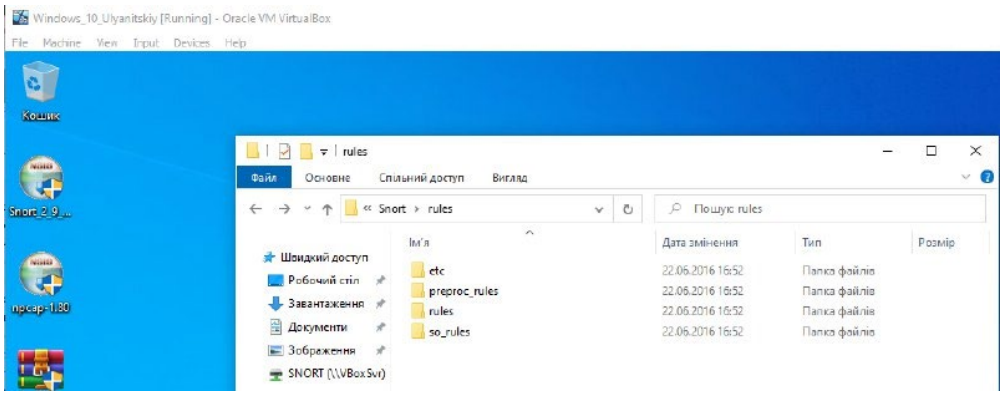
Npcap

Натискаємо кнопку "Install" і переконуємося, що Npcap було успішно встановлено (див. рисунок 3.44).



Npcap

Після успішної інсталяції Snort та Npcap переходимо до налаштування правил. Їх можна знайти на офіційному сайті Snort. Вибираємо правила, що відповідають версії Snort, завантажуємо архів та розпаковуємо його. Після цього повинні з'явитися такі папки: “rules”, “so_rules”, “etc” та “preproc_rules” (див. рисунок 3.45).



3.4 Налаштування програми Snort

Для перевірки правильності встановлення Snort потрібно відкрити командний рядок (з правами адміністратора) і спочатку вказати шлях до локального диска, де було завантажено Snort, використовуючи команду “cd C:\”.

Далі вводимо команду “cd Snort” для переходу в папку Snort, після чого за допомогою команди “cd bin” потрапляємо в підкаталог bin. Для перевірки версії Snort набираємо команду “Snort -V”. На рисунку 3.46 показано результат успішного встановлення Snort через командний рядок.

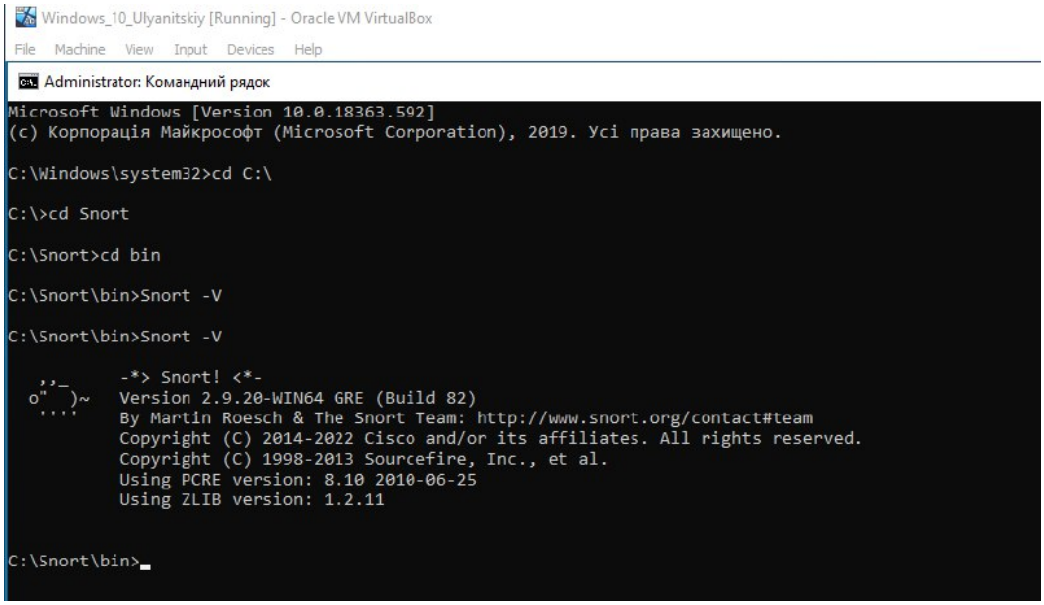


Рисунок 3.46 – Інформація про Snort у командному рядку.

Копіюємо весь вміст папки rules з розпакованими правилами в директорію c:\Snort\rules. Аналогічно, переносимо вміст папки preproc_rules до c:\Snort\preproc_rules, а вміст папки so_rules — до c:\Snort\so_rules (див. рисунок 3.47).

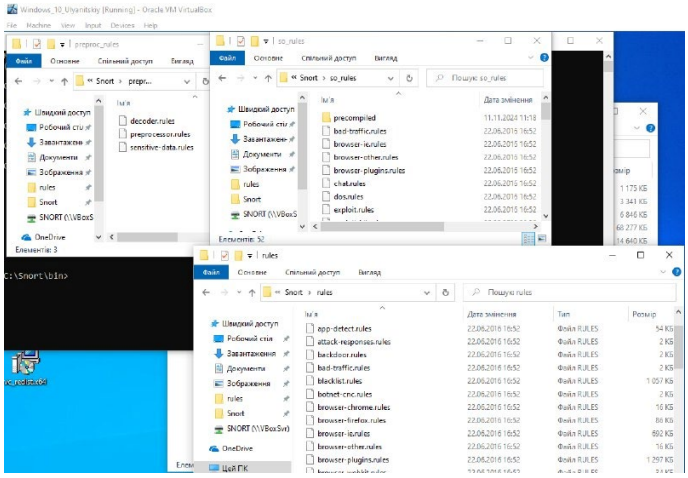


Рисунок 3.47 – Переміщення файлів із розпакованого архіву Snort

Наступним кроком переходимо до c:/Snort/etc і відкриваємо файл Snort.conf за допомогою стандартного Блокнота. Після цього прокручуємо файл вниз і в рядку ipvar HOME_NET any змінюємо "any" на IP-адресу нашого пристрою. Така сама зміна має бути внесена і в 48-й рядок (див. рисунок 3.48).

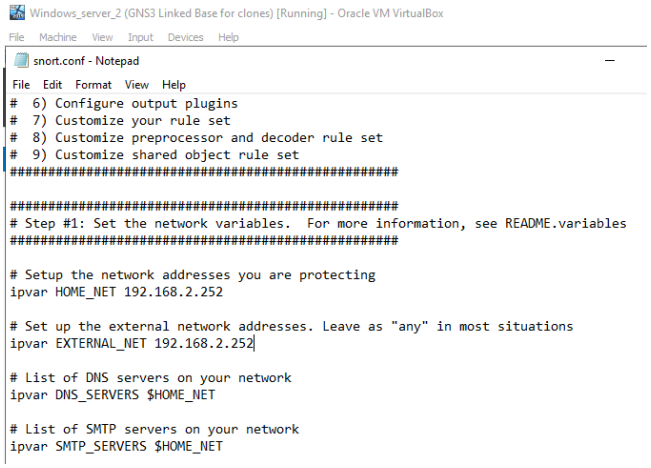
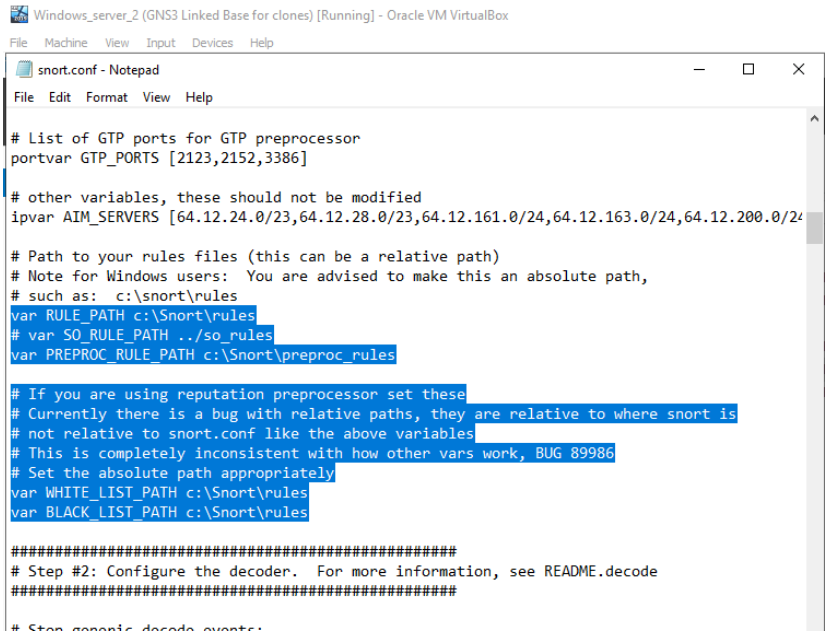
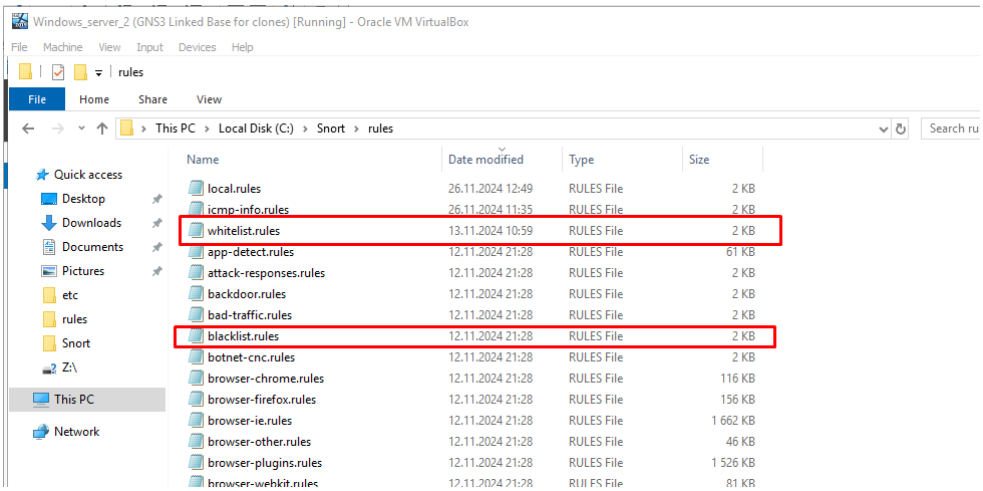


Рисунок 3.48 – Редагування IP-адреси у файлі Snort.conf

Після цього прокручуємо файл до рядка RULE_PATH, де замінюємо ../rules на c:\Snort\rules, ../so_rules на c:\Snort\so_rules, а також ../preproc_rules на c:\Snort\preproc_rules. Додатково, змінюємо значення WHITE_LIST_PATH і BLACK_LIST_PATH з ../rules на c:\Snort\rules (див. рисунок 3.49).



Створюємо два файли — whitelist.conf та blacklist.conf, розміщуючи їх в директорії C:\Snort\rules (див. рисунок 3.50).



whitelist".conf та "blacklist".conf

У файлі Snort.conf замінюємо рядок #config logdir: на config logdir:c:\Snort\log (див. рисунок 3.51).

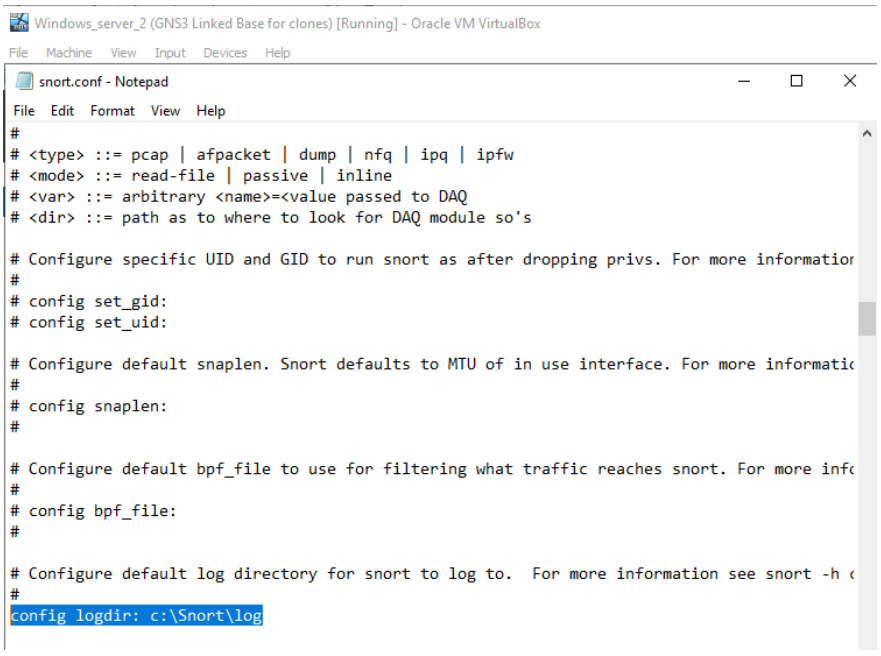


Рисунок 3.51 – Заміна команди #config logdir

Крім того, у файлі здійснюємо заміну наступних бібліотек:

usr/local/lib/snort_dynamicpreprocessor на C:\Snort\lib\snort_dynamicpreprocessor,

usr/local/lib/snort_dynamicengine/libsf_engine.so на C:\Snort\lib\snort_dynamicengine\sf_engine.dll.

Після того, як бібліотеки будуть замінені, закоментуємо їх за допомогою знака #.

Також додаємо коментарі (за допомогою #) перед усіма препроцесорами, які йдуть під вбудованою нормалізацією пакетів, оскільки вони викликають помилки при виконанні. Процес заміни бібліотек показано на рисунку 3.52.

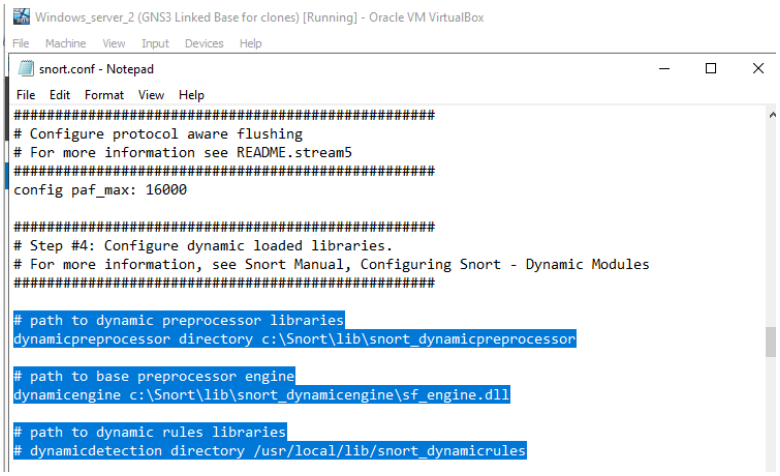
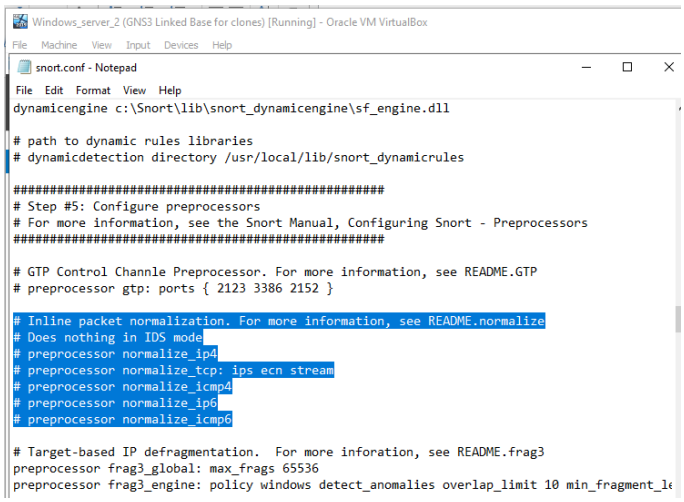
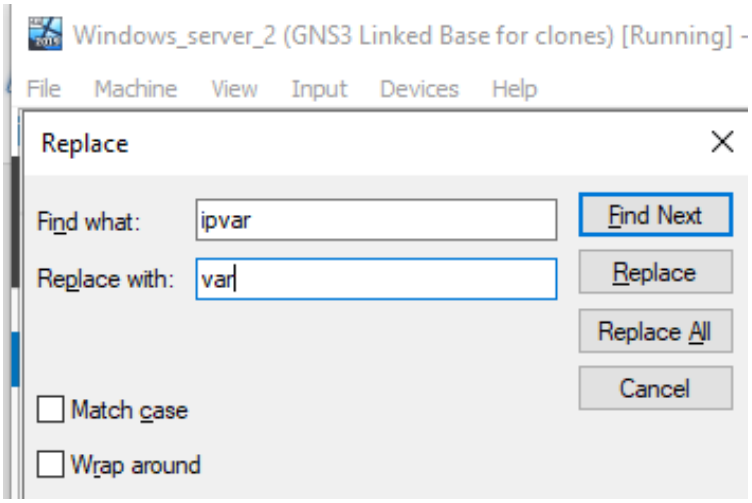


Рисунок 3.52 – Заміна бібліотек

Додаємо символ коментаря (#) перед усіма препроцесорами, що належать до вбудованої нормалізації пакетів, оскільки вони викликають помилки під час виконання. Список цих закоментованих препроцесорів можна побачити на рисунку 3.53.



Оскільки за замовчуванням рядок ipvar не розпізнається Snort, змінюємо його на var. Для цього використовуємо комбінацію клавіш Ctrl+H (див. рисунок 3.54).



ipvar на var

Заключним етапом є видалення зворотного слешу та додавання символу коментаря (#) в рядки, що містять тег "reputation preprocessor". Після виконання цих змін зберігаємо файл і закриваємо його (див. рисунок 3.55).

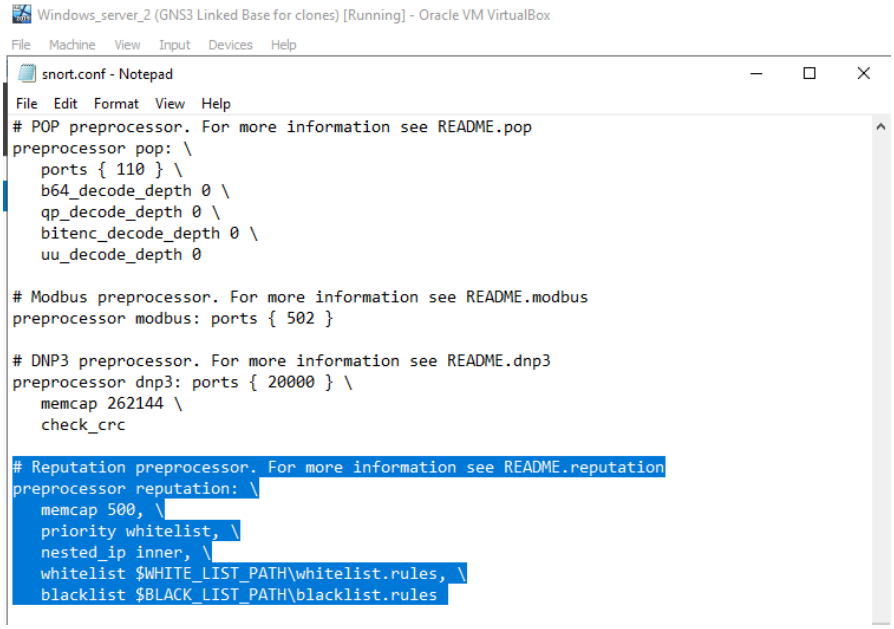


Рисунок 3.55 – Останній крок перед запуском Snort у файлі Snort.conf

Залишається встановити правило для Snort. Для цього переходимо до директорії c:\Snort\rules та відкриваємо файл icmp-info.rules за допомогою Блокнота (див. рисунок 3.56). Наприкінці файлу додаємо наступне правило (обов'язково):

alert tcp any any -> any any(msg: "Testing Alert" ; sid:1000001).

Це правило не має конкретних критеріїв, тому воно буде спрацьовувати на будь-який ICMP-пакет, який надходить. Важливо зазначити, що в правилі вказано унікальний ідентифікатор підпису (sid), який є обов'язковим. За умовами, власні правила Snort повинні починатися з 999999.

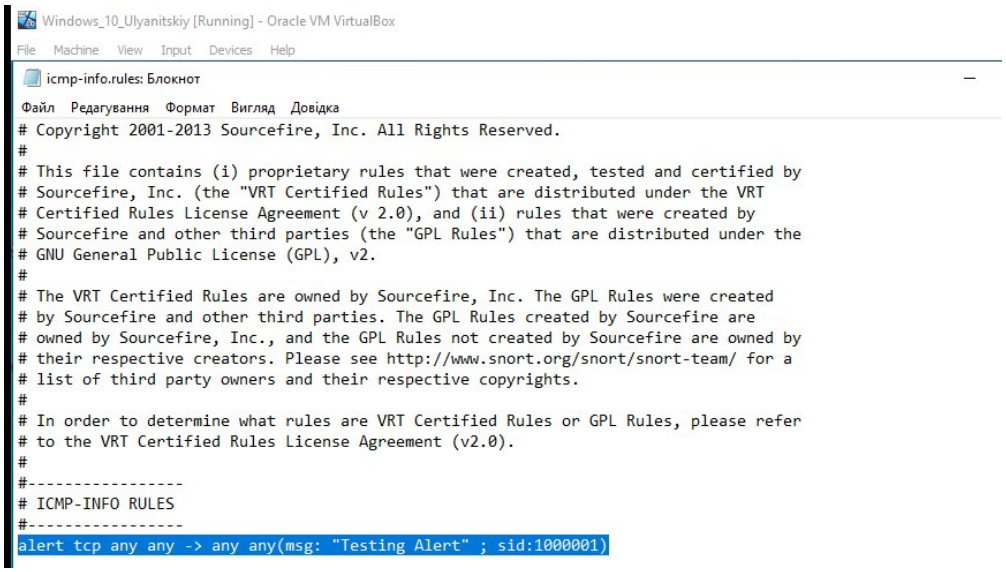


Рисунок 3.56 – Додавання правила у файлі icmp-info.rules

Щоб перевірити, чи Snort дійсно створює сповіщення, відкриваємо командний рядок, переходимо до директорії c:\Snort\bin та вводимо команду:

snort -iX -A console -c C:\snort\etc\snort.conf -l C:\Snort\log -K ascii

де X — це індексний номер вашого мережевого інтерфейсу. Після натискання Enter Snort готовий до роботи.

На рисунку 3.57 показано успішний запуск Snort на операційній системі Windows через командний рядок.

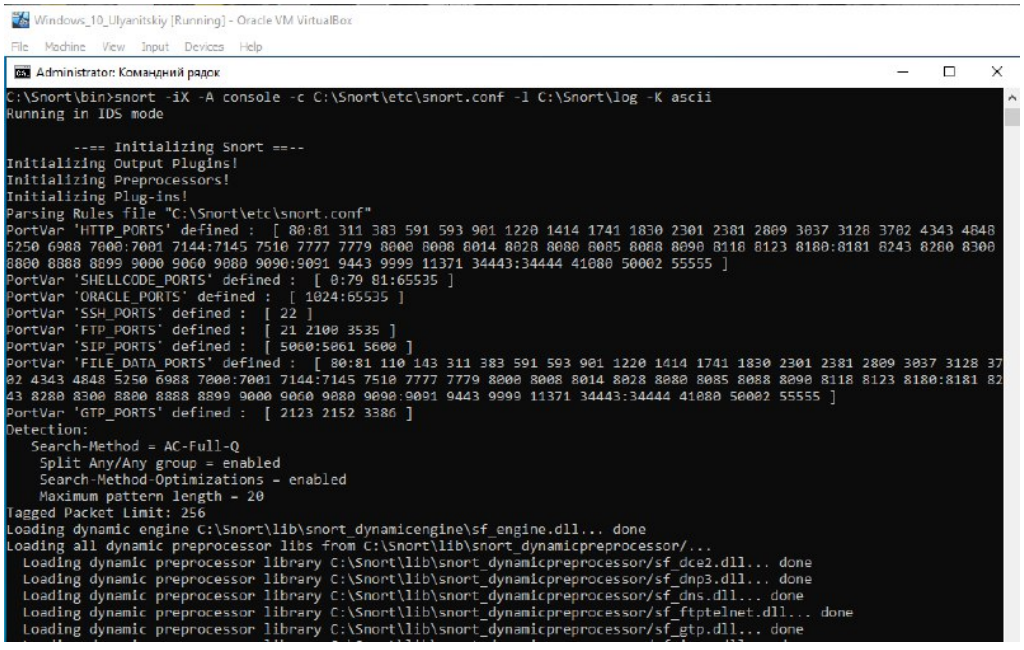


Рисунок 3.57 – Успішний запуск Snort через командну стрічку

3.5 Перевірка функціонування системи виявлення вторгнень

Щоб впевнитися в коректній роботі, необхідно протестувати Snort. Перший тест здійснюється через командний рядок за допомогою команди:

snort -i 1 -c c:\Snort\etc\snort.conf -T.

На рисунку 3.58 показано виконання першого етапу тестування.

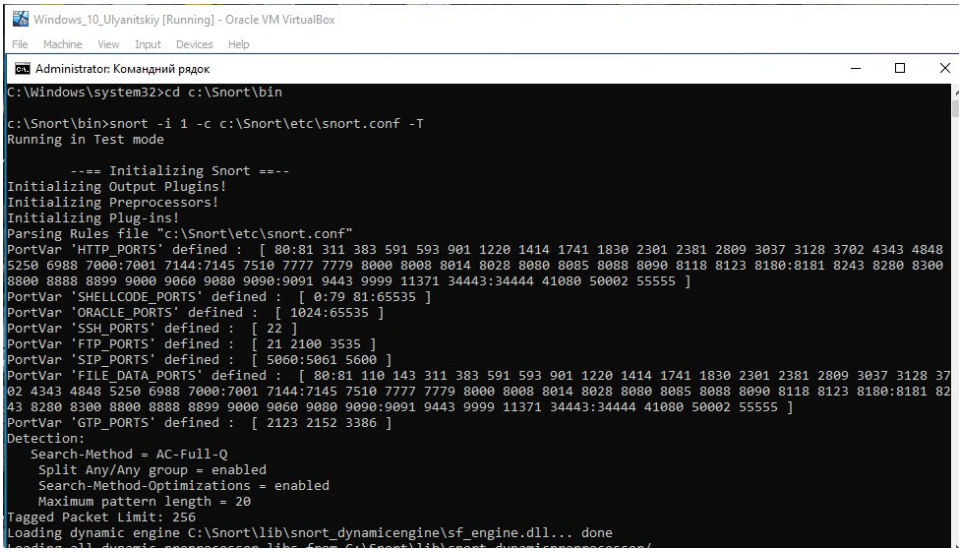


Рисунок 3.58 – Запуск першого тестування Можна спостерігати успішне виконання всіх препроцесорів (див. рисунок 3.59).

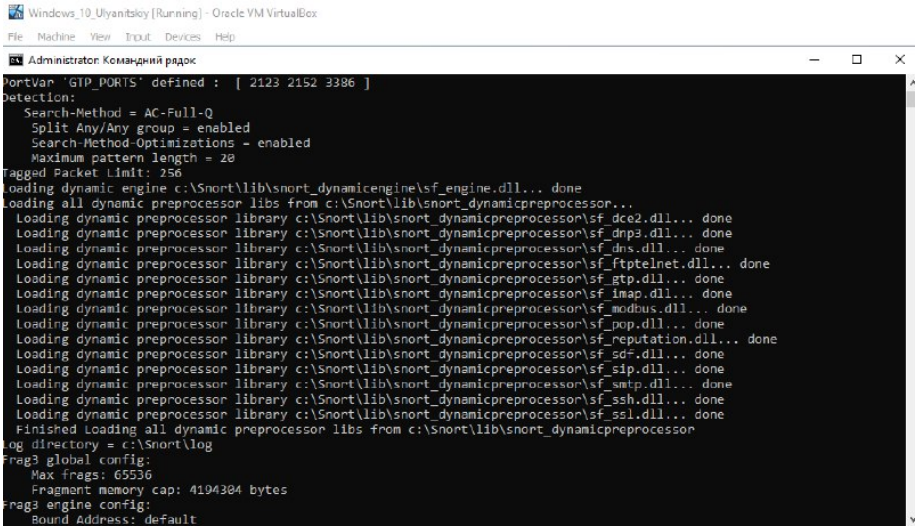


Рисунок 3.59 – Успішне виконання препроцесорів

Крім того, проводимо аналіз і переконуємося, що правила були успішно завантажені. Загалом було опрацьовано 11051 різноманітних правил (див. рисунок 3.60). Перевірка препроцесорів завершена успішно.

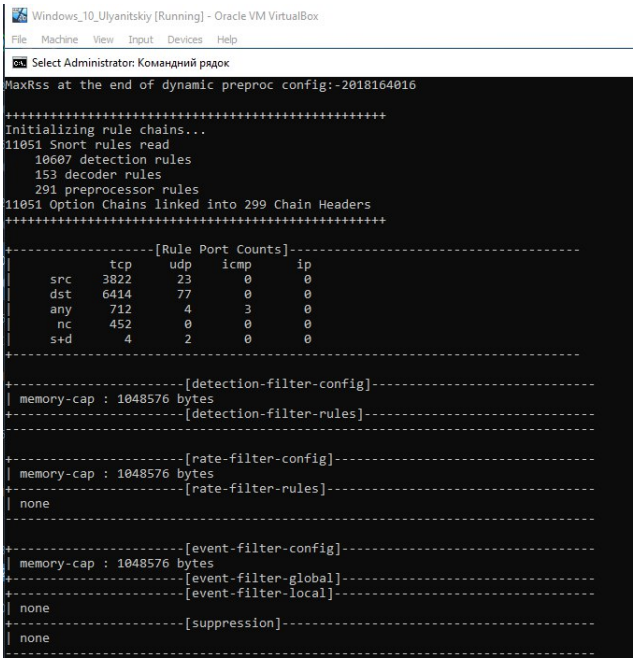


Рисунок 3.60 – Успішне виконання правил

У підсумку можемо спостерігати, що тестування було виконано успішно (див. рисунок 3.61).

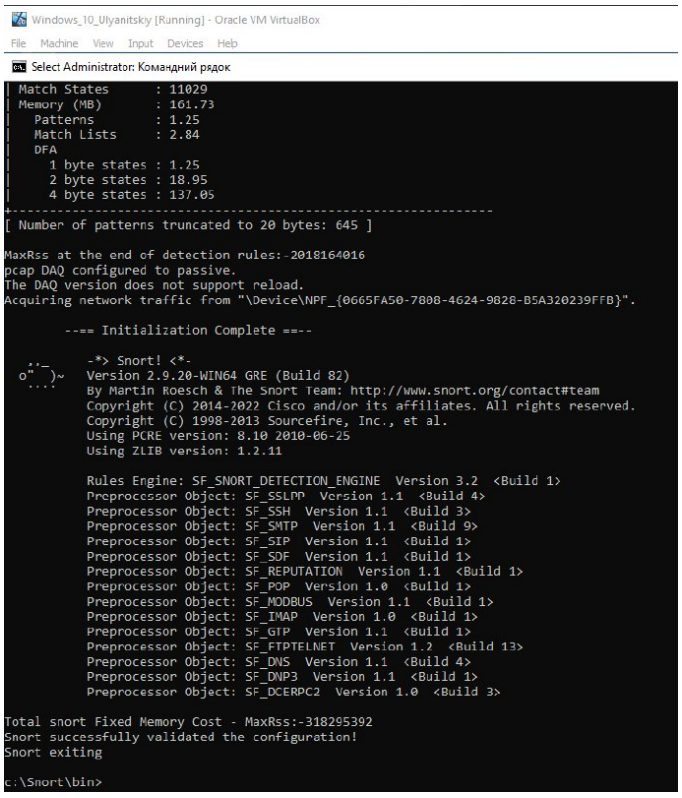


Рисунок 3.61 – Успішне перше тестування Snort

Як можна побачити, перший тест завершився успішно. Тепер перейдемо до виконання більш складного тестування через командний рядок. Спочатку необхідно перейти до папки c:\Snort\rules і відкрити файл local.rules. У цьому файлі додаємо наступні правила для генерації сповіщень:

alert icmp any any -> any any (msg:”Testing ICMP”;sid:1000001;)

alert tcp any any -> any any (msg:”Testing TCP”;sid:1000002;)

alert udp any any -> any any (msg:”Testing udp”;sid:1000003;)

На рисунку 3.62 зображено, як додати ці оповіщення до файлу local.rules.

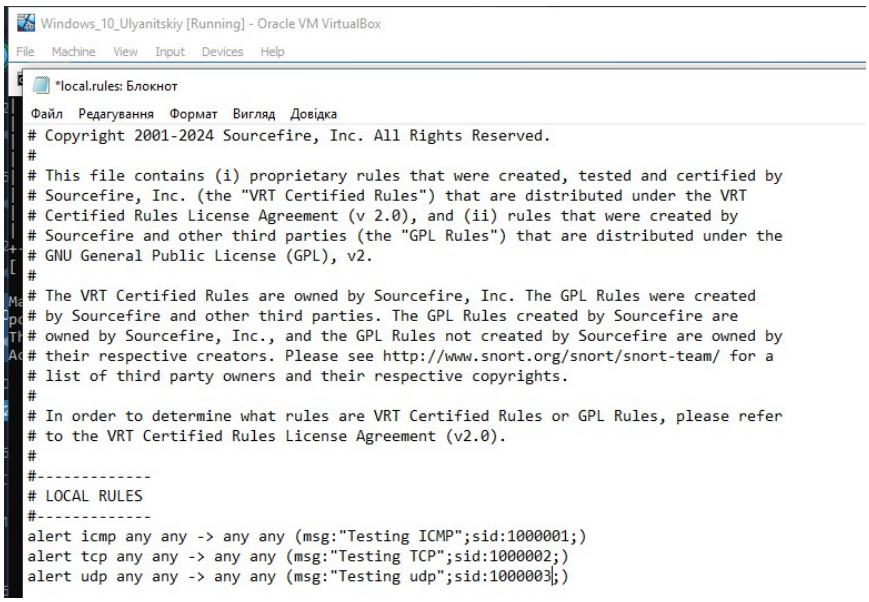


Рисунок 3.62 – Додавання оповіщень до файлу local.rules

Далі запускаємо тестування через командний рядок, використовуючи команду: snort -i 1 -c c:\Snort\etc\snort.conf -A console.

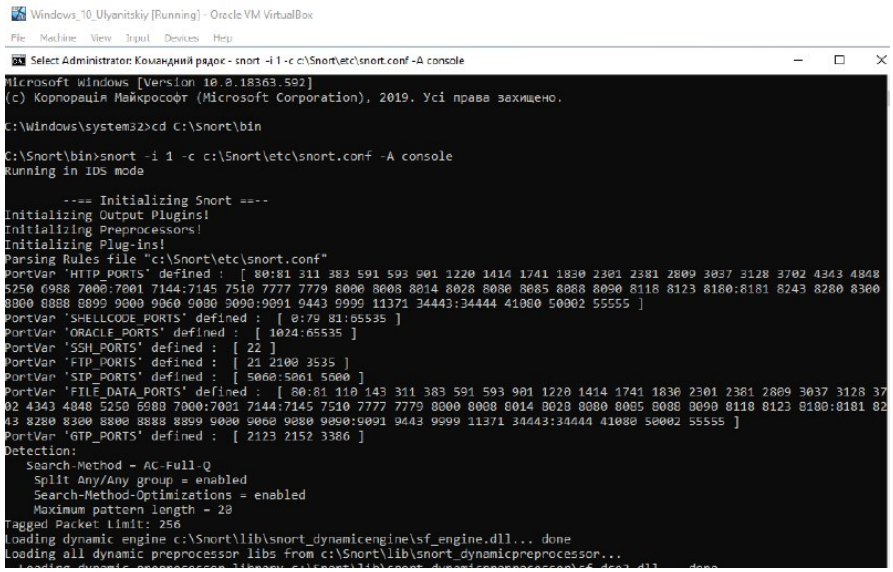


Рисунок 3.63 – Запуск другого тесту.

Після виконання другого тестування спостерігаємо успішне завантаження препроцесорів (рисунок 3.64).

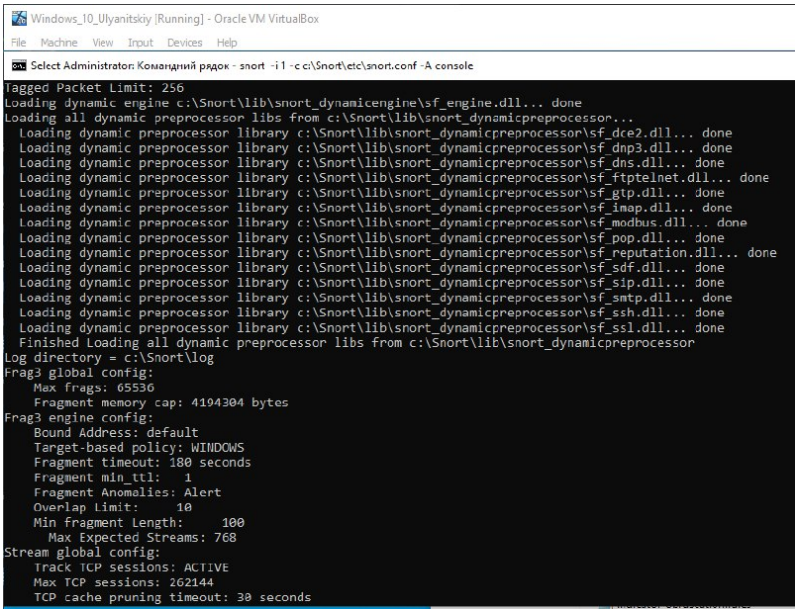


Рисунок 3.64 – Успішний запуск препроцесорів другого тесту.

Як показано на рисунку 3.65, всі правила були успішно завантажені, а перевірка препроцесорів виконана без помилок.

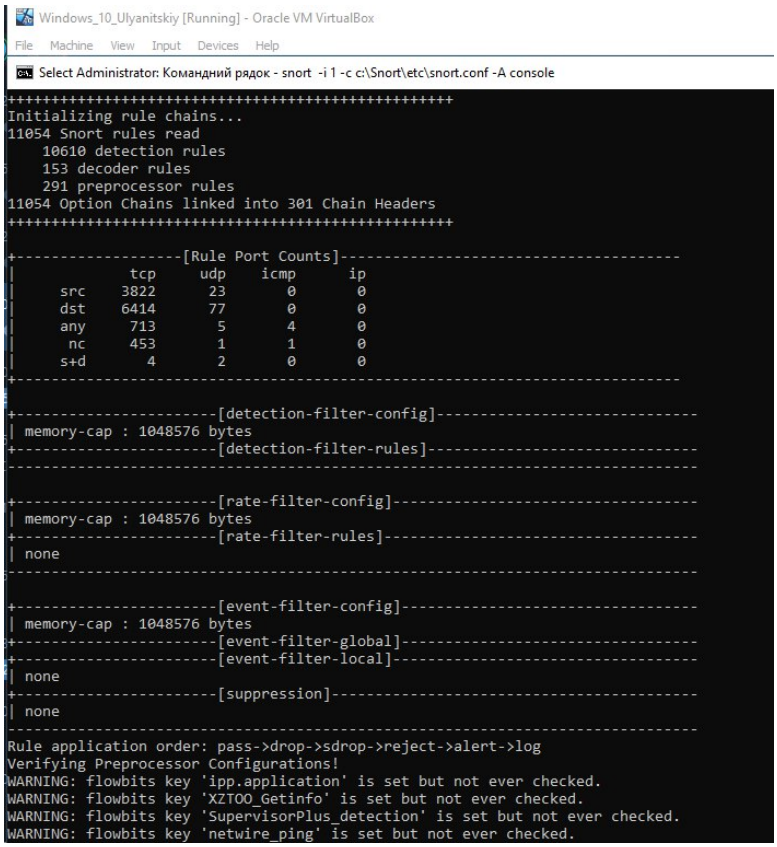


Рисунок 3.65 – Успішне зчитування правил та верифікація препроцесорів

На рисунку 3.66 видно, що процес ініціалізації Snort завершився успішно.

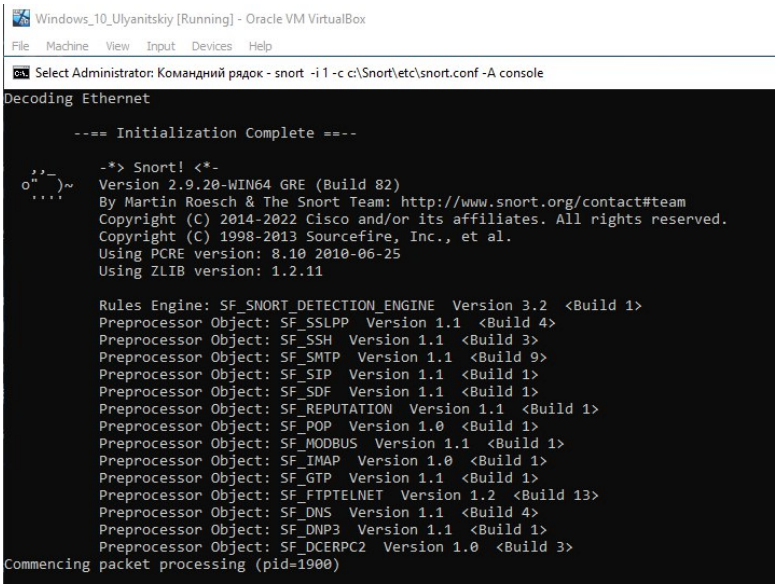


Рисунок 3.66 – Успішна ініціалізація Snort

На зображенні 3.67 відображено результат використання інструменту Zenmap, графічного інтерфейсу для Nmap, для виконання інтенсивного

сканування мережевого пристрою за IP-адресою 192.168.1.248. Під час сканування було виявлено відкритий порт 135/tcp, який використовується для Microsoft Windows RPC. Крім того, за допомогою функції OS Detection було визначено, що пристрій працює на операційній системі Microsoft Windows (версія Windows 10 або Windows Server 2019) з точністю 90-95%. Також виконано сканування MAC-адреси, що вказує на віртуальний адаптер Oracle VirtualBox з адресою 08:00:27:EF:FE:19. У результаті трасування маршруту було підтверджено, що IP-пакет досяг свого призначення за 1 хоп. Для збору додаткової інформації було використано скрипти Nmap, а загальний час сканування склав 35.30 секунд. Всі дані вказують на робочий стан пристрою та допомагають зрозуміти його конфігурацію в мережі.

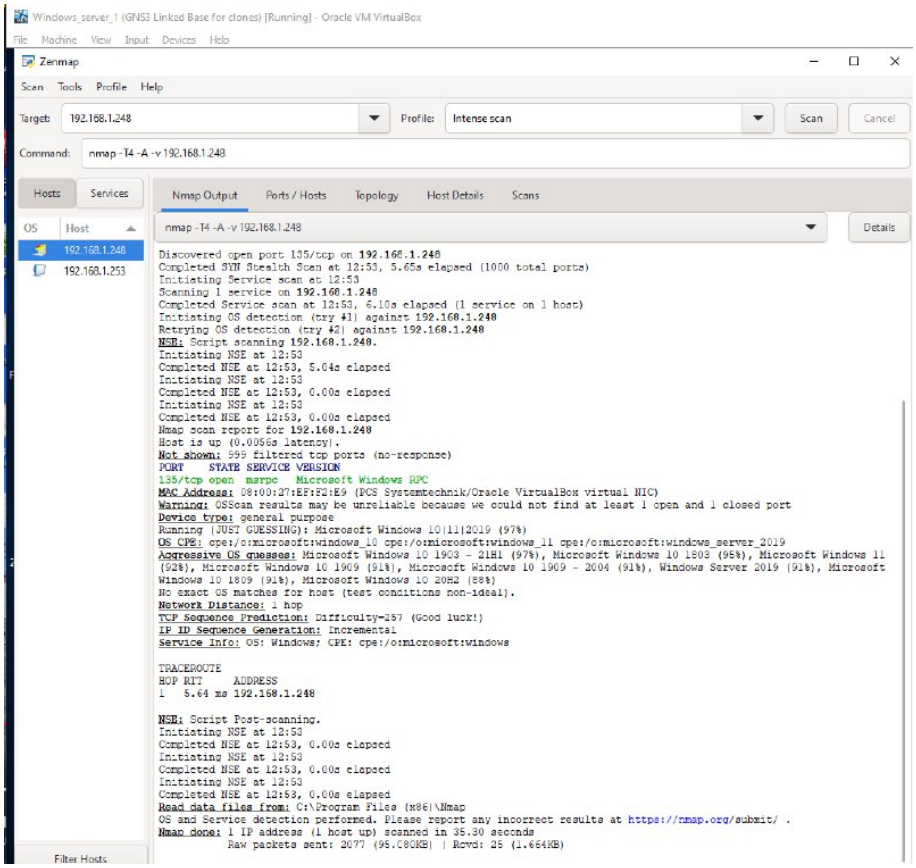


Рисунок 3.67 – відображення інформації в Zenmap

На зображенні показано результат тестування системи виявлення вторгнень Snort, яка здійснює моніторинг мережевих пакетів і генерує

сповіщення на основі заданих правил. Під час тесту Snort перевіряє різні типи трафіку, зокрема UDP і ICMP. Спочатку фіксуються пакети UDP з повідомленням "Testing udp", що вказує на їх перевірку між IP-адресами 192.168.1.253 і 192.168.1.248. Потім відбувається тестування для ICMP (зазвичай пінг-запитів), де також реєструються сповіщення про пріоритет та адреси джерела й призначення. Це свідчить про успішну роботу Snort, яка здатна виявляти і реєструвати пакети в реальному часі, забезпечуючи моніторинг та безпеку мережі.

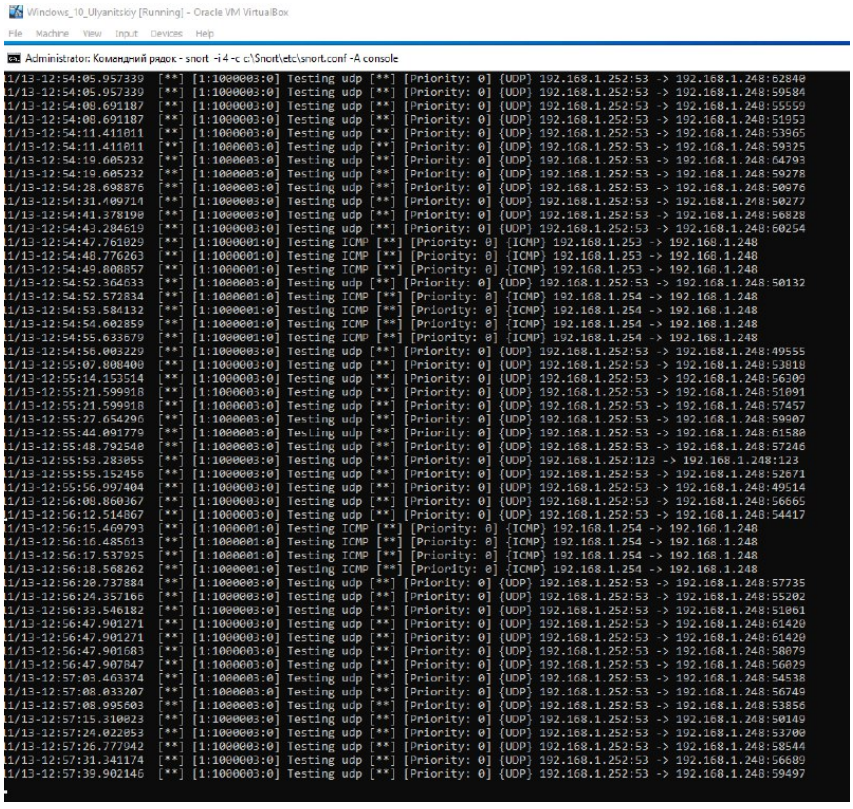


Рисунок 3.68 – Успішний процес другого тесту.

Щоб переглянути статистику, потрібно відкрити термінал і натискати комбінацію клавіш Ctrl+C. У результаті з'явиться статистика, яка містить інформацію про всі параметри Snort та виконані процеси. (рисунок 3.69)

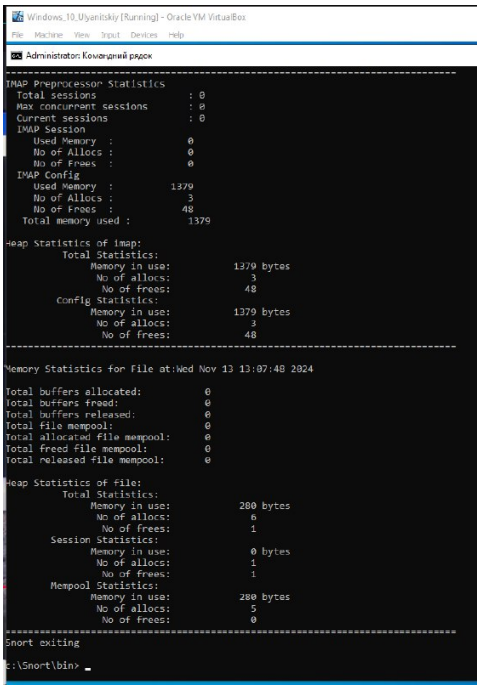


Рисунок 3.69 – Статистика Snort.

Далі показано 2 правила Snort, які записані у файлі local.rules:

Перше правило:

alert icmp any any -> any any (msg:"ICMP alert detected";sid:1000001;rev:1;)

Це правило спрацьовує при виявленні будь-якого ICMP пакету (з будь-якого джерела та до будь-якого призначення). Якщо Snort виявить такий пакет, він генерує сповіщення з повідомленням "ICMP alert detected" і має унікальний ідентифікатор (SID) 1000001.

Друге правило:

alert $HOME_NET any -> $HOME_NET any (msg:"Nmap scan FYI detected";flow:stateless;flags:S;f:12;reference:arachnids,27;classtype:attempted-recon;gid:180002;rev:4;)

Це правило генерує сповіщення, якщо в мережі з $HOME_NET буде виявлено певний тип мережевого сканування (наприклад, сканування порту через Nmap). Зокрема, перевіряється наявність специфічного набору прапорців TCP для виявлення SYN-сканування, що є типовим методом сканування портів. Якщо таке сканування виявлено, сповіщення містить текст "Nmap scan FYI detected" і має додаткову інформацію щодо класу спроби розвідки (recon).

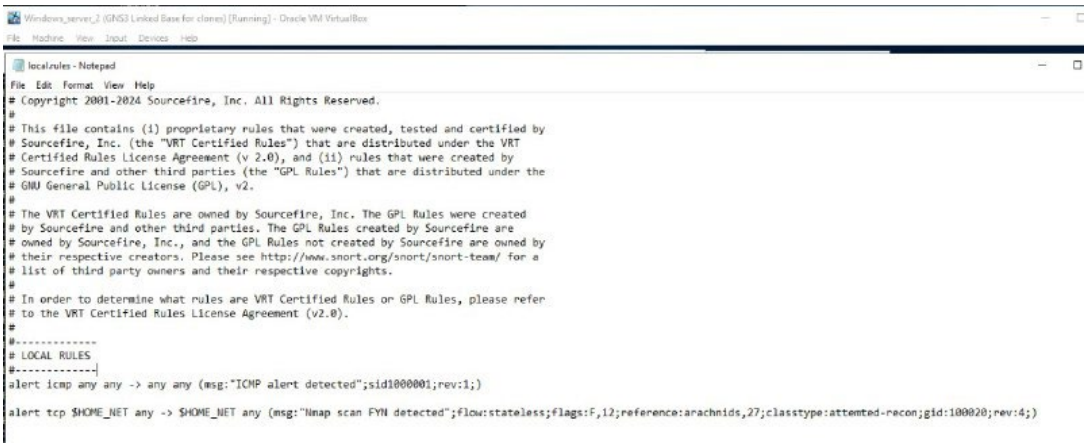


Рисунок 3.70 – Правила Snort

На рисунку 3.71 відображено результати сканування мережі за допомогою інструменту Nmap. Було проведено сканування IP-адреси 192.168.2.252, в результаті якого було визначено, що на цій адресі знаходиться комп'ютер під управлінням операційної системи Microsoft Windows 2019. Було виявлено відкриті порти та відповідні сервіси, такі як DNS, Kerberos, SMB, LDAP, HTTP та інші. Також було визначено тип пристрою та операційну систему. Ця інформація може бути використана для подальшого аналізу безпеки мережі та виявлення потенційних вразливостей.

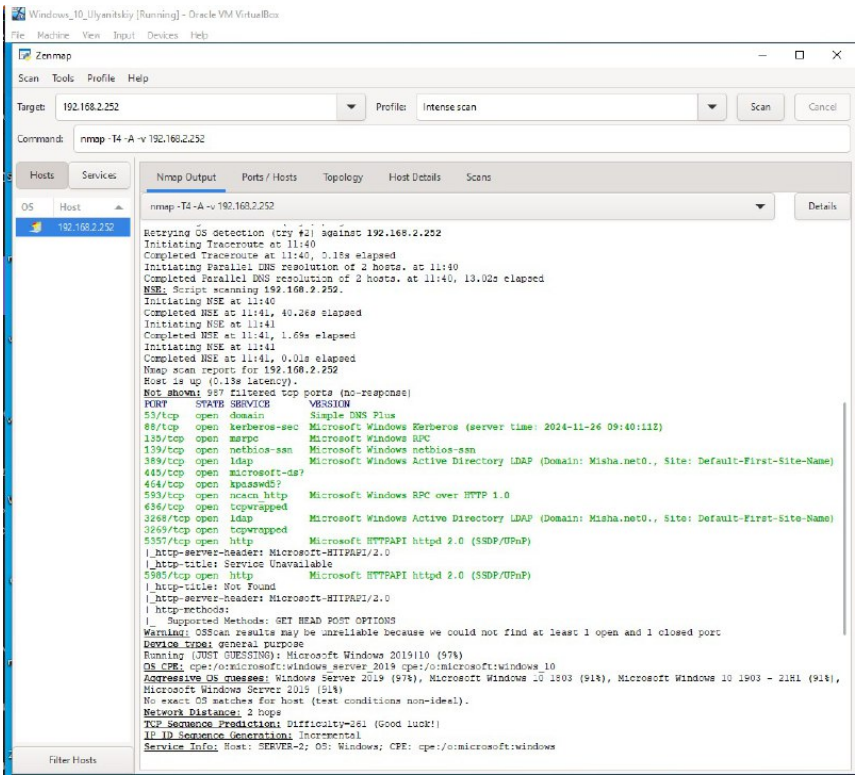


Рисунок 3.71 – Сканування Zenmap

На рисунку 3.72 представлені лог-файли системи виявлення вторгнень (IDS) Snort, яка моніторила мережевий трафік. Snort зафіксував велику кількість спроб підключення до різних портів на комп'ютері з IP-адресою 192.168.2.252. Ці спроби були класифіковані як потенційно шкідливий трафік, що може свідчити про спробу сканування або атаки на цей комп'ютер. Це можуть бути спроби підбору паролів, експлуатації вразливостей або інші види кібератак.

Більшість зафіксованих подій пов'язані з відсутністю тристороннього рукостискання TCP, що є характерною ознакою сканування портів або спроб встановити несанкціоновані з'єднання.

Загалом, цей лог свідчить про активність, спрямовану на комп'ютер з IP-адресою 192.168.2.252, і вимагає подальшого аналізу для визначення точної природи загрози та вжиття відповідних заходів безпеки.

Для додаткової перевірки було встановлено Kali Linux, а також налаштовано статичну IP-адресу для системи.

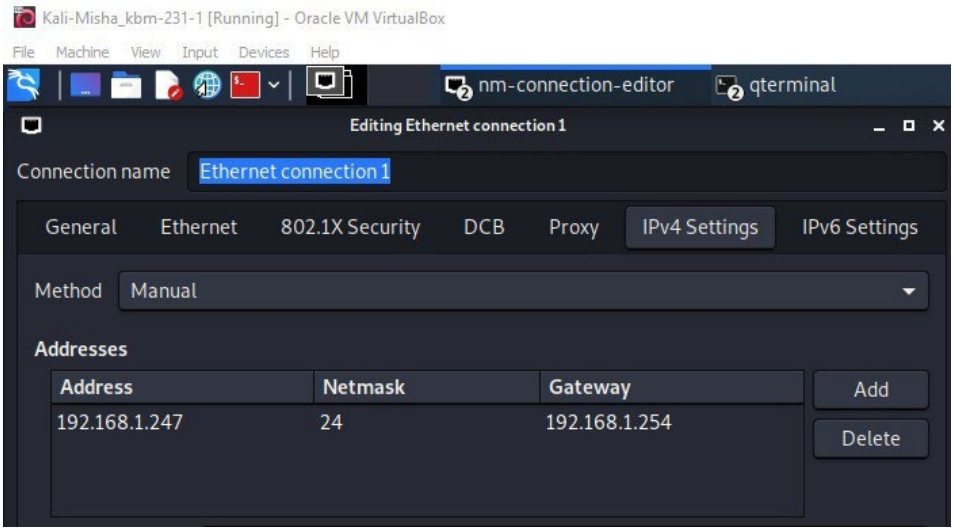


Рисунок 3.72 – Встановлений Kali Linuks

На зображенні 3.73 показано файл local.rules, який містить налаштування правил для Snort – системи виявлення вторгнень (IDS). У цьому файлі присутнє правило:

alert tcp any any -> $HOME_NET 21 (msg:"FTP Attempted"; sid:60001; rev:1;).

Це правило відстежує TCP-трафік, що надходить з будь-якої IP-адреси та будь-якого порту на порт 21 у домашній мережі $HOME_NET (порт 21 використовується для FTP). При збігу умови Snort згенерує сповіщення з повідомленням "FTP Attempted", використовуючи унікальний ідентифікатор правила (sid) 60001. Правило має ревізію (rev) 1, що означає першу версію цього правила.

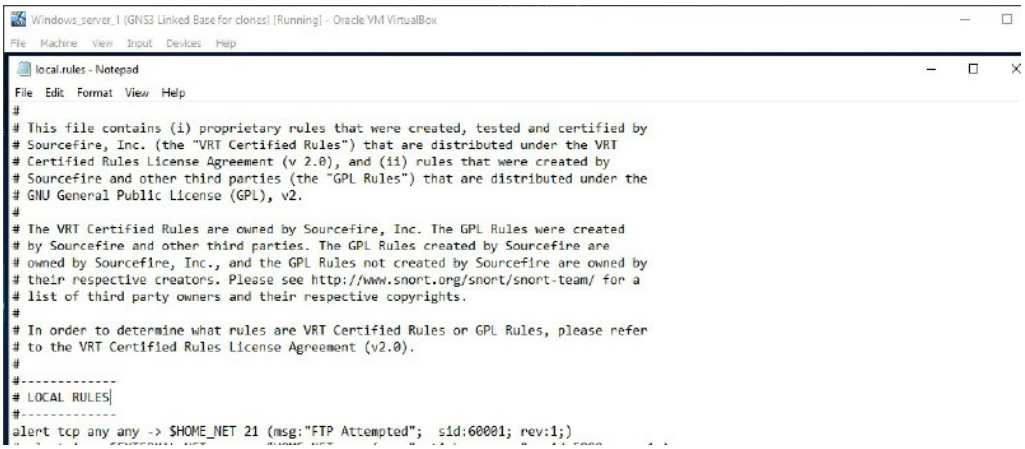


Рисунок 3.73 – Правила Snort

Наступним кроком підключаємося до FTP-сервера Windows з Kali Linux для перевірки з'єднання та тестування правил.

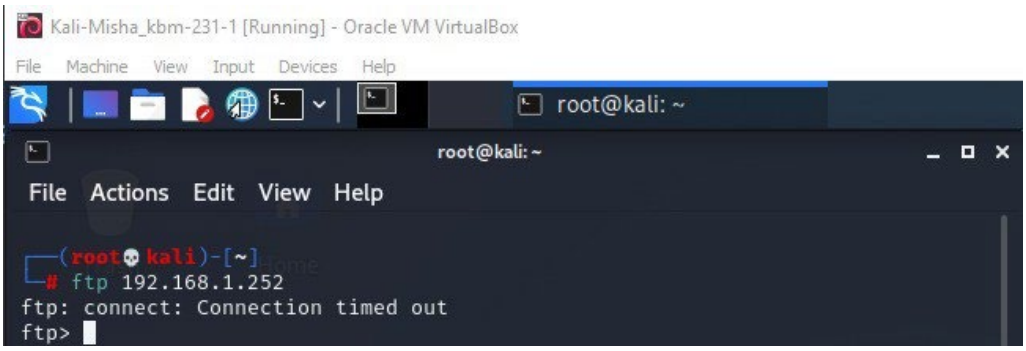


Рисунок 3.74 – Підключення по ftp

На Рисунку 3.75 можемо побачити що тестування правила пройшло успішно. Snort використовує конфігураційний файл snort.conf і завантажує різні препроцесори (наприклад, для FTP, HTTP, SMTP тощо). Після запуску процесу обробки пакетів Snort виявив спроби підключення до FTP-сервера з IP-адреси 192.168.1.247 на IP-адресу 192.168.1.252 порт 21. Це відповідає правилу FTP Attempted, що показано в логах з пріоритетом 0.

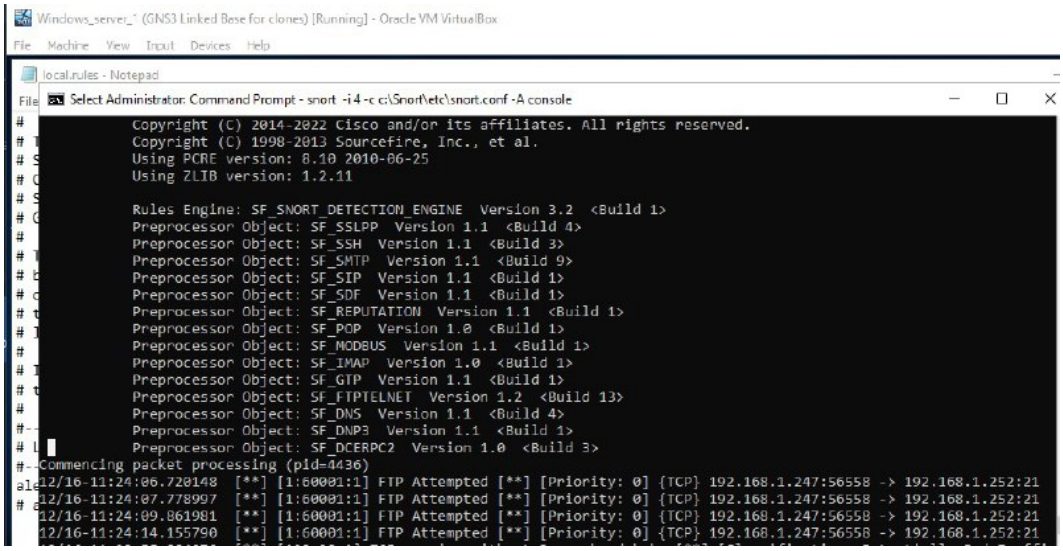


Рисунок 3.75 – Перевірка правила

Далі ми бачимо файл який містить конфігурацію для системи безпеки, яка дозволяє виявляти потенційні загрози в мережі.

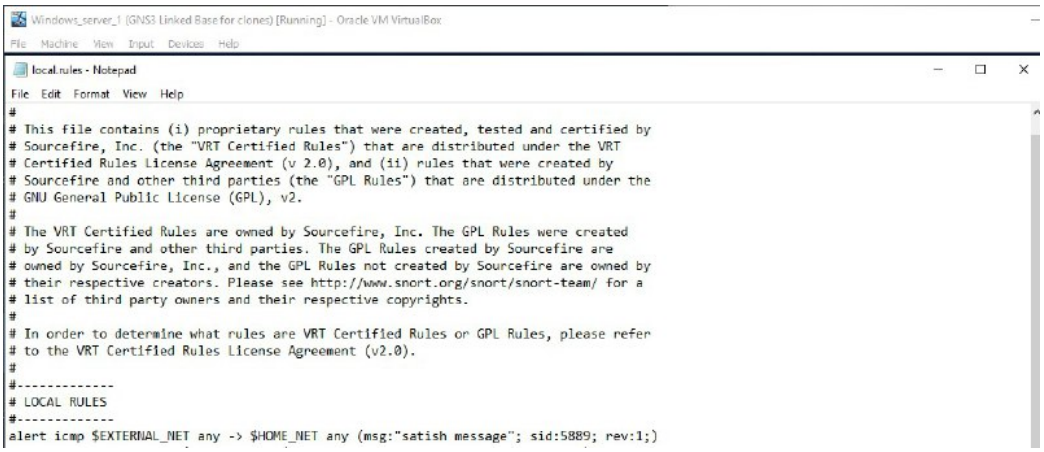


Рисунок 3.76 – Правила Snort

Виконано успішне тестування з'єднання з пристроєм за IP-адресою 192.168.1.252 за допомогою команди ping

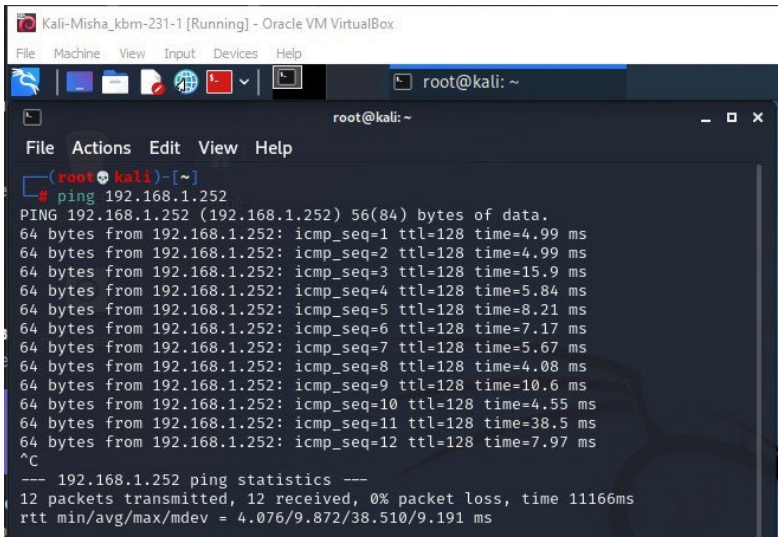


Рисунок 3.77 – Перевірка зв’язку

На зображенні 3.78 показано фрагмент логу системи виявлення вторгнень Snort, яка зафіксувала велику кількість спроб встановити несанкціоновані TCP-з'єднання. Це може свідчити про атаку або спробу сканування мережі.

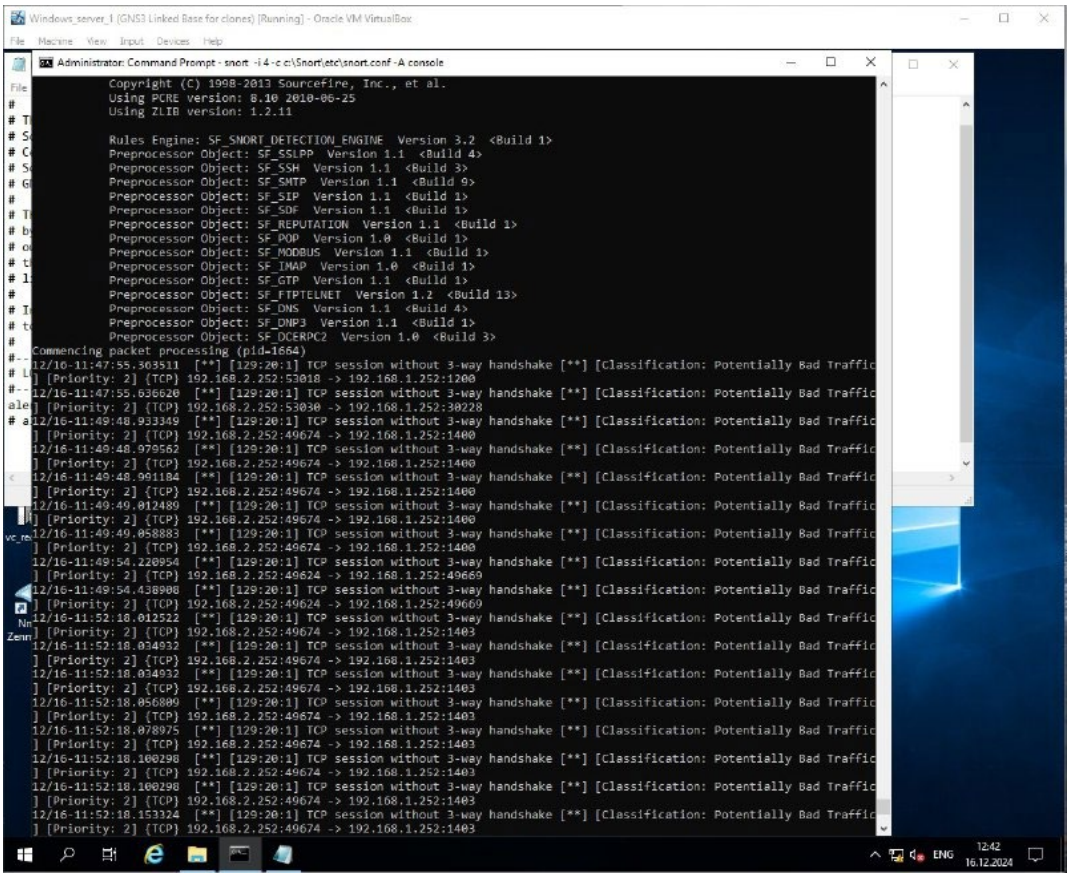


Рисунок 3.77 – Перевірка правила

Висновки до третього розділу:

У третьому розділі було детально розглянуто практичні аспекти впровадження підсистеми захисту клієнт-серверної мережі, що базується на платформі Microsoft Active Directory та системі виявлення вторгнень Snort. Було виконано налаштування ключових мережевих служб, таких як DNS, DHCP і NTP, які забезпечують централізоване управління мережею та її ефективне функціонування.

Особливу увагу приділено інтеграції Snort для моніторингу мережевого трафіку і виявлення потенційних атак, що дозволяє оперативно реагувати на загрози, забезпечуючи високу стійкість мережі до кіберзагроз. Проведене тестування підтвердило працездатність запропонованої системи захисту, яка продемонструвала здатність виявляти загрози, включаючи спроби несанкціонованого доступу і підозрілі активності.

Таким чином, запропонована архітектура дозволяє досягти високого рівня безпеки, зберігаючи при цьому стабільність роботи мережевих служб. Розроблені рішення є актуальними і можуть бути ефективно впроваджені в корпоративному середовищі для забезпечення надійного захисту від загроз.

ВИСНОВКИ

У магістерській роботі було проведено дослідження, спрямоване на розробку та впровадження системи виявлення атак на основі Snort в IT-інфраструктурі, що працює на платформі Microsoft Active Directory. У процесі виконання роботи досягнуто поставлених цілей, які включали теоретичне обґрунтування вибору технологій, проєктування мережевої архітектури, розробку системи адресації, впровадження системи виявлення вторгнень (IDS) та її тестування в реальних умовах.

У першому розділі було розглянуто загальні принципи функціонування операційної системи Windows Server 2019, її інтеграцію з платформою Active Directory та теоретичні аспекти роботи систем виявлення вторгнень (IDS). Проведено детальну класифікацію IDS, зокрема на основі місця встановлення та принципу дії, а також порівняння різних рішень, таких як Snort, Suricata, Zeek (Bro) і OSSEC. З’ясовано, що Snort є найбільш оптимальним вибором для впровадження завдяки його високій гнучкості, продуктивності та доступності.

Другий розділ було присвячено проєктуванню клієнт-серверної мережі. У роботі було розроблено топологію мережі з використанням трирівневої архітектури, що включає рівні ядра, агрегації та доступу. Виконано вибір обладнання, зокрема маршрутизаторів, комутаторів та робочих станцій, а також розроблено ефективну схему адресації. Усі елементи мережі були спроєктовані з урахуванням вимог до продуктивності, масштабованості та безпеки.

Третій розділ роботи включав реалізацію підсистеми захисту та тестування її працездатності. Було налаштовано ключові мережеві служби, такі як DNS, DHCP, NTP, а також інтегровано систему виявлення вторгнень Snort для моніторингу мережевого трафіку. У ході тестування перевірено здатність системи виявляти потенційні загрози, включаючи спроби сканування портів, несанкціоновані підключення та інші кібератаки. Отримані результати підтвердили високу ефективність впровадженого рішення.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Клієнт–серверна архітектура та ролі серверів [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Клієнт-серверна_архітектура
2. Snort – система виявлення вторгнень: основні принципи роботи [Електронний ресурс] – Режим доступу: https://www.snort.org
3. Роль Active Directory у забезпеченні безпеки мережевої інфраструктури [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/en-us/windows-server/identity/active-directory
4. Suricata IDS: порівняння можливостей з іншими рішеннями [Електронний ресурс] – Режим доступу: https://suricata-ids.org
5. Особливості моніторингу мережевого трафіку для виявлення загроз [Електронний ресурс] – Режим доступу: https://www.networkworld.com
6. Впровадження системи Snort у корпоративну мережу: покрокова інструкція [Електронний ресурс] – Режим доступу: https://cybersecuritymagazine.com/snort-setup
7. Взаємодія Snort із Windows Server 2019 [Електронний ресурс] – Режим доступу: https://www.windowsserversecurity.com/snort
8. Cisco та інструменти для забезпечення безпеки мережі [Електронний ресурс] – Режим доступу: https://www.cisco.com
9. Zeek (Bro): система моніторингу мережевої безпеки [Електронний ресурс] – Режим доступу: https://zeek.org
10. Windows Server 2019: сучасні функції для корпоративної інфраструктури [Електронний ресурс] – Режим доступу: https://techcommunity.microsoft.com
11. Налаштування DHCP, DNS та NTP на Windows Server [Електронний ресурс] – Режим доступу: https://theitbros.com/dns-dhcp-ntp-setup
12. Кіберзагрози у сучасному IT-середовищі [Електронний ресурс] – Режим доступу: https://cybersecurityventures.com

13. Wazuh як альтернатива Snort для інтеграції з SIEM-системами [Електронний ресурс] – Режим доступу: https://wazuh.com

14. Методи захисту клієнт-серверної архітектури [Електронний ресурс] – Режим доступу: https://www.securecomputing.net

15. Аналіз та порівняння систем виявлення вторгнень (IDS) [Електронний ресурс] – Режим доступу: https://securityweekly.com/ids-comparison

16. NIDS і HIDS: переваги та недоліки у застосуванні [Електронний ресурс] – Режим доступу: https://www.sans.org/reading-room

17. Локальні IDS системи: Переваги OSSEC [Електронний ресурс] – Режим доступу: https://ossec.github.io

18. Intrusion Detection in IT Infrastructures: Case Study [Електронний ресурс] – Режим доступу: https://journals.plos.org/cybersec

19. Windows Server: Кращі практики налаштування NTP [Електронний ресурс] – Режим доступу: https://ntp.org/server2019

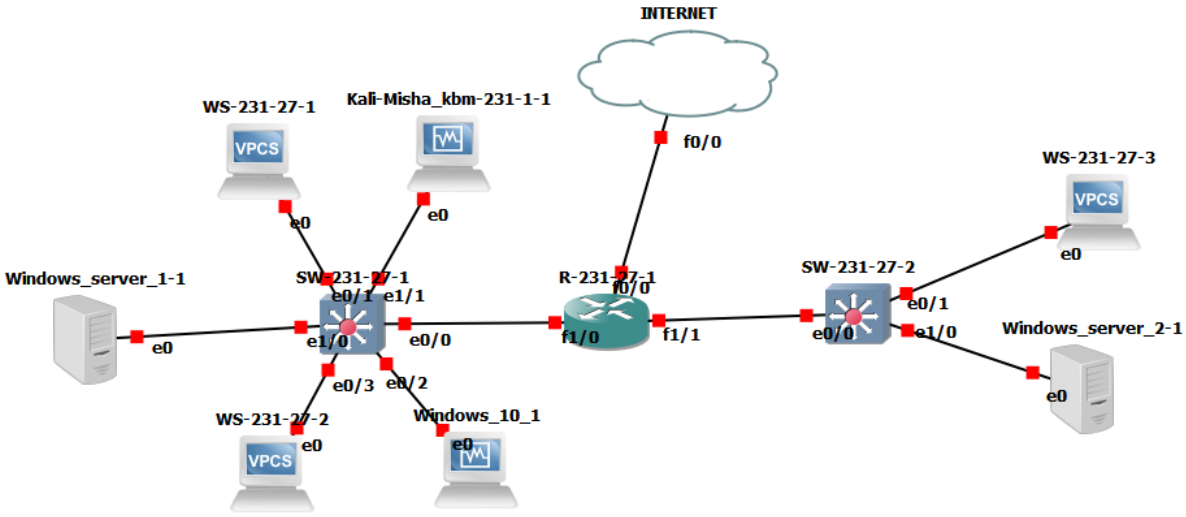
20. Захист мережевої інфраструктури за допомогою IDS [Електронний ресурс] – Режим доступу: https://techrepublic.com/network-security

21. Ульяніцький М.А., Фальковський І.Г. Проект системи виявлення атак на основі snort в IT-інфраструктурі // Тези доповідей XIV Міжнародної науково-технічної конференції «Інформаційно-комп'ютерні технології – 2024», м. Житомир, 02-03 грудня 2024 року – Житомир: Житомирська політехніка, 2024.

22. Ульніцький М.А., Бродський Ю. Б. Підсистема захисту клієнт-серверної мережі від кібератак // Тези доповідей VII Всеукраїнської науково-технічної конференції «Комп'ютерні технології: інновації, проблеми, рішення», м. Житомир, 02-03 грудня 2024 року – Житомир: Житомирська політехніка, 2024. – С. 27–28.

ДОДАТКИ

Додаток А – Детальна схема побудованої мережі



```
Current configuration : 1723 bytes
!
! Last configuration change at 09:51:38 UTC Tue Dec 17 2024
!
version 15.2
service timestamps debug datetime msec
service timestamps log datetime
no service password-encryption
service compress-config
!
hostname SW-231-27-1
!
boot-start-marker
boot-end-marker
!
!
logging discriminator EXCESS severity drops 6 msg-body drops EXCESSCOLL
logging buffered 50000
logging console discriminator EXCESS
enable secret 5 $1$5cz5$YDhypzydhA2JSc98FGBTf.
!
username admin privilege 15 password 0 passAdmin
no aaa new-model
!
!
!
!
!
no ip routing
no ip icmp rate-limit unreachable
!
!
!
no ip domain-lookup
ip domain-name Misha.net
no ip cef
no ipv6 cef
!
!
!
spanning-tree mode rapid-pvst
spanning-tree extend system-id
!
vlan internal allocation policy ascending
!
ip tcp synwait-time 5
!
!
!
!
!
!
!
!
!
!
!
!
!
interface Ethernet0/0
!
interface Ethernet0/1
!
interface Ethernet0/2
!
interface Ethernet0/3
!
interface Ethernet1/0
!
interface Ethernet1/1
!
interface Ethernet1/2
!
interface Ethernet1/3
!
interface Ethernet2/0
!
interface Ethernet2/1
!
interface Ethernet2/2
!
interface Ethernet2/3
!
interface Ethernet3/0
!
interface Ethernet3/1
!
interface Ethernet3/2
!
interface Ethernet3/3
```

```
!
interface Vlan1
 ip address 192.168.1.253 255.255.255.0
 no ip route-cache
!
ip default-gateway 192.168.1.254
ip forward-protocol nd
!
no ip http server
no ip http secure-server
!
ip ssh version 2
!
!
logging host 192.168.1.252
!
!
!
control-plane
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line vty 0 4
 exec-timeout 60 0
 logging synchronous
 login local
 transport input ssh
!
ntp update-calendar
ntp server 202.192.27.1
!
End
Current configuration : 1211 bytes
!
upgrade fpd auto
version 15.2
service timestamps debug datetime msec
service timestamps log datetime
no service password-encryption
!
hostname R-231-27-1
!
boot-start-marker
boot-end-marker
!
!
!
no aaa new-model
no ip icmp rate-limit unreachable
!
!
!
!
!
!
no ip domain lookup
ip cef
no ipv6 cef
!
multilink bundle-name authenticated
!
!
!
!
!
!
!
!
!
!
!
redundancy
!
!
ip tcp synwait-time 5
!
!
!
!
!
!
!
!
!
!
!
!
interface FastEthernet0/0
 ip address 202.192.27.2 255.255.255.252
 duplex half
!
interface FastEthernet1/0
```

```
 ip address 192.168.1.254 255.255.255.0
 duplex auto
 speed auto
!
interface FastEthernet1/1
 ip address 192.168.2.254 255.255.255.0
 duplex auto
 speed auto
!
ip forward-protocol nd
no ip http server
no ip http secure-server
!
!
ip route 0.0.0.0 0.0.0.0 202.192.27.1
!
logging trap notifications
logging host 192.168.1.252
no cdp log mismatch duplex
!
!
!
control-plane
!
!
!
mgcp profile default
!
!
!
gatekeeper
 shutdown
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line vty 0 4
 login
 transport input all
!
```

```
 ntp update-calendar
 ntp server 202.192.27.1
 !
 End
 Current configuration : 1723 bytes
 !
 ! Last configuration change at 09:51:55 UTC Tue Dec 17 2024
 !
 version 15.2
 service timestamps debug datetime msec
 service timestamps log datetime
 no service password-encryption
 service compress-config
 !
 hostname SW-231-27-2
 !
 boot-start-marker
 boot-end-marker
 !
 !
 logging discriminator EXCESS severity drops 6 msg-body drops EXCESSCOLL
 logging buffered 50000
 logging console discriminator EXCESS
 enable secret 5 $1$vIdA$lYvNVZzz7QHw1N5UuVwEU1
 !
 username admin privilege 15 password 0 passAdmin
 no aaa new-model
 !
 !
 !
 !
 !
 no ip routing
 no ip icmp rate-limit unreachable
 !
 !
 !
 no ip domain-lookup
 ip domain-name Misha.net
 no ip cef
 no ipv6 cef
 !
 !
```

```
!
spanning-tree mode rapid-pvst
spanning-tree extend system-id
!
vlan internal allocation policy ascending
!
ip tcp synwait-time 5
!
!
!
!
!
!
!
!
!
!
!
!
!
interface Ethernet0/0
!
interface Ethernet0/1
!
interface Ethernet0/2
!
interface Ethernet0/3
!
interface Ethernet1/0
!
interface Ethernet1/1
!
interface Ethernet1/2
!
interface Ethernet1/3
!
interface Ethernet2/0
!
interface Ethernet2/1
!
interface Ethernet2/2
!
interface Ethernet2/3
!
interface Ethernet3/0
!
interface Ethernet3/1
!
interface Ethernet3/2
!
interface Ethernet3/3
!
interface Vlan1
 ip address 192.168.2.253 255.255.255.0
 no ip route-cache
!
ip default-gateway 192.168.2.254
ip forward-protocol nd
!
no ip http server
no ip http secure-server
!
ip ssh version 2
!
!
logging host 192.168.1.252
!
!
!
control-plane
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line vty 0 4
 exec-timeout 60 0
 logging synchronous
 login local
 transport input ssh
!
ntp update-calendar
ntp server 202.192.27.1
!
```

```
# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.1.248

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET 192.168.1.248

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
ipvar SIP_SERVERS $HOME_NET
```

Додаток В. Відтестована та відредагована конфігурація Snort

```
# List of ports you run web servers on
portvar HTTP_PORTS [80,81,311,383,591,593,901,1220,1414,1741,1830,2301,2381,2809,3037,3128,3702,4343,4848,5250,6988,7000,7001,7144,7145,7510,7777,7779,8000,8008,8014,8028,8080,8085,8088,8090,8118,8123,8180,8181,8243,8280,8300,8800,8888,8899,9000,9060,9080,9090,9091,9443,9999,11371,34443,34444,41080,50002,55555]

# List of ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80

# List of ports you might see oracle attacks on
portvar ORACLE_PORTS 1024:

# List of ports you want to look for SSH connections on:
portvar SSH_PORTS 22

# List of ports you run ftp servers on
portvar FTP_PORTS [21,2100,3535]

# List of ports you run SIP servers on
portvar SIP_PORTS [5060,5061,5600]

# List of file data ports for file inspection
portvar FILE_DATA_PORTS [$HTTP_PORTS,110,143]

# List of GTP ports for GTP preprocessor
portvar GTP_PORTS [2123,2152,3386]

# other variables, these should not be modified
ipvar AIM_SERVERS [64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.188.3.0/24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24,205.188.179.0/24,205.188.248.0/24]

# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH c:\Snort\rules
# var SO_RULE_PATH ../so_rules
var PREPROC_RULE_PATH c:\Snort\preproc_rules
```

```
# If you are using reputation preprocessor set these
# Currently there is a bug with relative paths, they are relative to where snort is
# not relative to snort.conf like the above variables
# This is completely inconsistent with how other vars work, BUG 89986
# Set the absolute path appropriately
var WHITE_LIST_PATH c:\Snort\rules
var BLACK_LIST_PATH c:\Snort\rules

##################################################
# Step #2: Configure the decoder.  For more information, see README.decode
##################################################

# Stop generic decode events:
config disable_decode_alerts

# Stop Alerts on experimental TCP options
config disable_tcpopt_experimental_alerts

# Stop Alerts on obsolete TCP options
config disable_tcpopt_obsolete_alerts

# Stop Alerts on T/TCP alerts
config disable_tcpopt_ttcp_alerts

# Stop Alerts on all other TCPOption type events:
config disable_tcpopt_alerts

# Stop Alerts on invalid ip options
config disable_ipopt_alerts

# Alert if value in length field (IP, TCP, UDP) is greater th elength of the packet
# config enable_decode_oversized_alerts

# Same as above, but drop packet if in Inline mode (requires enable_decode_oversized_alerts)
# config enable_decode_oversized_drops

# Configure IP / TCP checksum mode
config checksum_mode: all

# Configure maximum number of flowbit references.  For more information, see README.flowbits
# config flowbits_size: 64

# Configure ports to ignore
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53

# Configure active response for non inline operation. For more information, see REAMDE.active
# config response: eth0 attempts 2

# Configure DAQ related options for inline operation. For more information, see README.daq
#
# config daq: <type>
# config daq_dir: <dir>
# config daq_mode: <mode>
# config daq_var: <var>
#
# <type> ::= pcap | afpacket | dump | nfq | ipq | ipfw
# <mode> ::= read-file | passive | inline
# <var> ::= arbitrary <name>=<value passed to DAQ
# <dir> ::= path as to where to look for DAQ module so's

# Configure specific UID and GID to run snort as after dropping privs. For more information see snort -h command line options
#
# config set_gid:
# config set_uid:

# Configure default snaplen. Snort defaults to MTU of in use interface. For more information see README
#
# config snaplen:
```

```
#

# Configure default bpf_file to use for filtering what traffic reaches snort. For more information see snort -h command line options (-F)
#
# config bpf_file:
#

# Configure default log directory for snort to log to.  For more information see snort -h command line options (-l)
#
config logdir: c:\Snort\log


##################################################
# Step #3: Configure the base detection engine.  For more information, see README.decode
##################################################

# Configure PCRE match limitations
config pcre_match_limit: 3500
config pcre_match_limit_recursion: 1500

# Configure the detection engine  See the Snort Manual, Configuring Snort - Includes - Config
config detection: search-method ac-split search-optimize max-pattern-len 20

# Configure the event queue.  For more information, see README.event_queue
config event_queue: max_queue 8 log 5 order_events content_length

##################################################
## Configure GTP if it is to be used.
## For more information, see README.GTP
##################################################

# config enable_gtp

##################################################
# Per packet and rule latency enforcement
# For more information see README.ppm
##################################################

# Per Packet latency configuration
#config ppm: max-pkt-time 250, \
#   fastpath-expensive-packets, \
#   pkt-log

# Per Rule latency configuration
#config ppm: max-rule-time 200, \
#   threshold 3, \
#   suspend-expensive-rules, \
#   suspend-timeout 20, \
#   rule-log alert

##################################################
# Configure Perf Profiling for debugging
# For more information see README.PerfProfiling
##################################################

#config profile_rules: print all, sort avg_ticks
#config profile_preprocs: print all, sort avg_ticks

##################################################
# Configure protocol aware flushing
# For more information see README.stream5
##################################################
config paf_max: 16000

##################################################
# Step #4: Configure dynamic loaded libraries.
```

```
# For more information, see Snort Manual, Configuring Snort - Dynamic Modules
##################################################

# path to dynamic preprocessor libraries
dynamicpreprocessor directory c:\Snort\lib\snort_dynamicpreprocessor

# path to base preprocessor engine
dynamicengine c:\Snort\lib\snort_dynamicengine\sf_engine.dll

# path to dynamic rules libraries
# dynamicdetection directory /usr/local/lib/snort_dynamicrules

##################################################
# Step #5: Configure preprocessors
# For more information, see the Snort Manual, Configuring Snort - Preprocessors
##################################################

# GTP Control Channle Preprocessor. For more information, see README.GTP
# preprocessor gtp: ports { 2123 3386 2152 }

# Inline packet normalization. For more information, see README.normalize
# Does nothing in IDS mode
# preprocessor normalize_ip4
# preprocessor normalize_tcp: ips ecn stream
# preprocessor normalize_icmp4
# preprocessor normalize_ip6
# preprocessor normalize_icmp6

# Target-based IP defragmentation. For more inforation, see README.frag3
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10 min_fragment_length 100 timeout 180

# Target-Based stateful inspection/stream reassembly. For more inforation, see README.stream5
preprocessor stream5_global: track_tcp yes, \
  track_udp yes, \
  track_icmp no, \
  max_tcp 262144, \
  max_udp 131072, \
  max_active_responses 2, \
  min_response_seconds 5
preprocessor stream5_tcp: log_asymmetric_traffic no, policy windows, \
  detect_anomalies, require_3whs 180, \
  overlap_limit 10, small_segments 3 bytes 150, timeout 180, \
   ports client 21 22 23 25 42 53 79 109 110 111 113 119 135 136 137 139 143 \
      161 445 513 514 587 593 691 1433 1521 1741 2100 3306 6070 6665 6666 6667 6668 6669 \
      7000 8181 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779, \
   ports both 80 81 311 383 443 465 563 591 593 636 901 989 992 993 994 995 1220 1414 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7907 7000 7001 7144 7145 7510 7802 7777 7779 \
      7801 7900 7901 7902 7903 7904 7905 7906 7908 7909 7910 7911 7912 7913 7914 7915 7916 \
      7917 7918 7919 7920 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090 9091 9443 9999 11371 34443 34444 41080 50002 55555
preprocessor stream5_udp: timeout 180

# performance statistics. For more information, see the Snort Manual, Configuring Snort - Preprocessors - Performance Monitor
# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt 10000
```

```
# HTTP normalization and anomaly detection.  For more information, see README.http_inspect
preprocessor http_inspect: global iis_unicode_map unicode.map 1252 compress_depth 65535 decompress_depth 65535
preprocessor http_inspect_server: server default \
    http_methods { GET POST PUT SEARCH MKCOL COPY MOVE LOCK UNLOCK NOTIFY POLL BCOPY BDELETE BMOVE LINK UNLINK OPTIONS HEAD DELETE TRACE TRACK CONNECT SOURCE SUBSCRIBE UNSUBSCRIBE PROPFIND PROPPATCH BPROPFIND BPROPPATCH RPC_CONNECT PROXY_SUCCESS BITS_POST CCM_POST SMS_POST RPC_IN_DATA RPC_OUT_DATA RPC_ECHO_DATA } \
    chunk_length 500000 \
    server_flow_depth 0 \
    client_flow_depth 0 \
    post_depth 65495 \
    oversize_dir_length 500 \
    max_header_length 750 \
    max_headers 100 \
    max_spaces 200 \
    small_chunk_length { 10 5 } \
    ports { 80 81 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000 7001 7144 7145 7510 7777 7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180 8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090 9091 9443 9999 11371 34443 34444 41080 50002 55555 } \
    non_rfc_char { 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 } \
    enable_cookie \
    extended_response_inspection \
    inspect_gzip \
    normalize_utf \
    unlimited_decompress \
    normalize_javascript \
    apache_whitespace no \
    ascii no \
    bare_byte no \
    directory no \
    double_decode no \
    iis_backslash no \
    iis_delimiter no \
    iis_unicode no \
    multi_slash no \
    utf_8 no \
    u_encode yes \
    webroot no

# ONC-RPC normalization and anomaly detection.  For more information, see the Snort Manual, Configuring Snort - Preprocessors - RPC Decode
preprocessor rpc_decode: 111 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779 no_alert_multiple_requests no_alert_large_fragments no_alert_incomplete

# Back Orifice detection.
# preprocessor bo

# FTP / Telnet normalization and anomaly detection.  For more information, see README.ftptelnet
preprocessor ftp_telnet: global inspection_type stateful encrypted_traffic no check_encrypted
preprocessor ftp_telnet_protocol: telnet \
    ayt_attack_thresh 20 \
    normalize ports { 23 } \
    detect_anomalies
preprocessor ftp_telnet_protocol: ftp server default \
    def_max_param_len 100 \
    ports { 21 2100 3535 } \
    telnet_cmds yes \
    ignore_telnet_erase_cmds yes \
    ftp_cmds { ABOR ACCT ADAT ALLO APPE AUTH CCC CDUP } \
    ftp_cmds { CEL CLNT CMD CONF CWD DELE ENC EPRT } \
    ftp_cmds { EPSV ESTA ESTP FEAT HELP LANG LIST LPRT } \
    ftp_cmds { LPSV MACB MAIL MDTM MIC MKD MLSD MLST } \
    ftp_cmds { MODE NLST NOOP OPTS PASS PASV PBSZ PORT } \
```

```
        ftp_cmds { PROT PWD QUIT REIN
REST RETR RMD RNFR } \
        ftp_cmds { RNTO SDUP SITE SIZE
SMNT STAT STOR STOU } \
        ftp_cmds { STRU SYST TEST
TYPE USER XCUP XCRC XCWD } \
        ftp_cmds { XMAS XMD5 XMKD
XPWD XRCP XRMD XRSQ XSEM } \
        ftp_cmds { XSEN XSHA1
XSHA256 } \
        alt_max_param_len 0 { ABOR CCC
CDUP ESTA FEAT LPSV NOOP PASV
PWD QUIT REIN STOU SYST XCUP
XPWD } \
        alt_max_param_len 200 { ALLO
APPE CMD HELP NLST RETR RNFR STOR
STOU XMKD } \
        alt_max_param_len 256 { CWD
RNTO } \
        alt_max_param_len 400 { PORT } \
        alt_max_param_len 512 { SIZE } \
        chk_str_fmt { ACCT ADAT ALLO
APPE AUTH CEL CLNT CMD } \
        chk_str_fmt { CONF CWD DELE
ENC EPRT EPSV ESTP HELP } \
        chk_str_fmt { LANG LIST LPRT
MACB MAIL MDTM MIC MKD } \
        chk_str_fmt { MLSD MLST MODE
NLST OPTS PASS PBSZ PORT } \
        chk_str_fmt { PROT REST RETR
RMD RNFR RNTO SDUP SITE } \
        chk_str_fmt { SIZE SMNT STAT
STOR STRU TEST TYPE USER } \
        chk_str_fmt { XCRC XCWD
XMAS XMD5 XMKD XRCP XRMD XRSQ
} \
        chk_str_fmt { XSEM XSEN
XSHA1 XSHA256 } \
        cmd_validity ALLO < int [ char R
int ] > \
        cmd_validity EPSV < [ { char 12 |
char A char L char L } ] > \
        cmd_validity MACB < string > \
        cmd_validity MDTM < [ date
nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
        cmd_validity MODE < char ASBCZ
> \
        cmd_validity PORT < host_port > \
        cmd_validity PROT < char CSEP >
\
        cmd_validity STRU < char FRPO [
string ] > \
        cmd_validity TYPE < { char AE [
char NTC ] | char I | char L [ number ] } >
    preprocessor ftp_telnet_protocol: ftp
client default \
        max_resp_len 256 \
        bounce yes \
        ignore_telnet_erase_cmds yes \
        telnet_cmds yes


    # SMTP normalization and anomaly
detection. For more information, see
README.SMTP
    preprocessor smtp: ports { 25 465 587
691 } \
        inspection_type stateful \
        b64_decode_depth 0 \
        qp_decode_depth 0 \
        bitenc_decode_depth 0 \
        uu_decode_depth 0 \
        log_mailfrom \
        log_rcptto \
        log_filename \
        log_email_hdrs \
        normalize cmds \
        normalize_cmds { ATRN AUTH
BDAT CHUNKING DATA DEBUG EHLO
EMAL ESAM ESND ESOM ETRN EVFY } \
        normalize_cmds { EXPN HELO
HELP IDENT MAIL NOOP ONEX QUEU
QUIT RCPT RSET SAML SEND SOML } \
        normalize_cmds { STARTTLS
TICK TIME TURN TURNME VERB VRFY
X-ADAT X-DRCP X-ERCP X-EXCH50 } \
        normalize_cmds { X-EXPS X-
LINK2STATE XADR XAUTH XCIR
XEXCH50 XGEN XLICENSE XQUE XSTA
XTRN XUSR } \
        max_command_line_len 512 \
        max_header_line_len 1000 \
        max_response_line_len 512 \
        alt_max_command_line_len 260 {
MAIL } \
        alt_max_command_line_len 300 {
RCPT } \
        alt_max_command_line_len 500 {
HELP HELO ETRN EHLO } \
```

```
        alt_max_command_line_len 255 { EXPN VRFY ATRN SIZE BDAT DEBUG EMAL ESAM ESND ESOM EVFY IDENT NOOP RSET } \
        alt_max_command_line_len 246 { SEND SAML SOML AUTH TURN ETRN DATA RSET QUIT ONEX QUEU STARTTLS TICK TIME TURNME VERB X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
        valid_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN EVFY } \
        valid_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND SOML } \
        valid_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-DRCP X-ERCP X-EXCH50 } \
        valid_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
        xlink2state { enabled }

    # Portscan detection. For more information, see README.sfportscan
    preprocessor sfportscan: proto { all } memcap { 10000000 } sense_level { low }

    # ARP spoof detection. For more information, see the Snort Manual - Configuring Snort - Preprocessors - ARP Spoof Preprocessor
    # preprocessor arpspoof
    # preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00

    # SSH anomaly detection. For more information, see README.ssh
    preprocessor ssh: server_ports { 22 } \
                autodetect \
                max_client_bytes 19600 \
                max_encrypted_packets 20 \
                max_server_version_len 100 \
                enable_respoverflow enable_ssh1crc32 \
                enable_srvoverflow enable_protomismatch

    # SMB / DCE-RPC normalization and anomaly detection. For more information, see README.dcerpc2
    preprocessor dcerpc2: memcap 102400, events [co ]
    preprocessor dcerpc2_server: default, policy WinXP, \
        detect [smb [139,445], tcp 135, udp 135, rpc-over-http-server 593], \
        autodetect [tcp 1025:, udp 1025:, rpc-over-http-server 1025:], \
        smb_max_chain 3, smb_invalid_shares ["C$", "D$", "ADMIN$"]

    # DNS anomaly detection. For more information, see README.dns
    preprocessor dns: ports { 53 } enable_rdata_overflow

    # SSL anomaly detection and traffic bypass. For more information, see README.ssl
    preprocessor ssl: ports { 443 465 563 636 989 992 993 994 995 7801 7802 7900 7901 7902 7903 7904 7905 7906 7907 7908 7909 7910 7911 7912 7913 7914 7915 7916 7917 7918 7919 7920 }, trustservers, noinspect_encrypted

    # SDF sensitive data preprocessor. For more information see README.sensitive_data
    preprocessor sensitive_data: alert_threshold 25

    # SIP Session Initiation Protocol preprocessor. For more information see README.sip
    preprocessor sip: max_sessions 40000, \
      ports { 5060 5061 5600 }, \
      methods { invite \
            cancel \
            ack \
            bye \
            register \
            options \
```

```
                refer \
                subscribe \
                update \
                join \
                info \
                message \
                notify \
                benotify \
                do \
                qauth \
                sprack \
                publish \
                service \
                unsubscribe \
                prack }, \
            max_uri_len 512, \
            max_call_id_len 80, \
            max_requestName_len 20, \
            max_from_len 256, \
            max_to_len 256, \
            max_via_len 1024, \
            max_contact_len 512, \
            max_content_len 2048

        # IMAP preprocessor. For more information see README.imap
        preprocessor imap: \
          ports { 143 } \
          b64_decode_depth 0 \
          qp_decode_depth 0 \
          bitenc_decode_depth 0 \
          uu_decode_depth 0

        # POP preprocessor. For more information see README.pop
        preprocessor pop: \
          ports { 110 } \
          b64_decode_depth 0 \
          qp_decode_depth 0 \
          bitenc_decode_depth 0 \
          uu_decode_depth 0

        # Modbus preprocessor. For more information see README.modbus
        preprocessor modbus: ports { 502 }

        # DNP3 preprocessor. For more information see README.dnp3
        preprocessor dnp3: ports { 20000 } \
          memcap 262144 \
          check_crc

        # Reputation preprocessor. For more information see README.reputation
        preprocessor reputation: \
          memcap 500, \
          priority whitelist, \
          nested_ip inner, \
          whitelist $WHITE_LIST_PATH\whitelist.rules, \
          blacklist $BLACK_LIST_PATH\blacklist.rules

        ##################################################
        # Step #6: Configure output plugins
        # For more information, see Snort Manual, Configuring Snort - Output Modules
        ##################################################

        # unified2
        # Recommended for most installs
        # output unified2: filename merged.log, limit 128, nostamp, mpls_event_types, vlan_event_types

        # Additional configuration for specific types of installs
        # output alert_unified2: filename snort.alert, limit 128, nostamp
        # output log_unified2: filename snort.log, limit 128, nostamp

        # syslog
        # output alert_syslog: LOG_AUTH LOG_ALERT

        # pcap
        # output log_tcpdump: tcpdump.log

        # metadata reference data.  do not modify these lines
        include classification.config
        include reference.config


        ##################################################
        # Step #7: Customize your rule set
```

```
        # For more information, see Snort Manual, Writing Snort Rules
        #
        # NOTE: All categories are enabled in this conf file
        ############################################################

        # site specific rules
        include $RULE_PATH\local.rules

        include $RULE_PATH\app-detect.rules
        include $RULE_PATH\attack-responses.rules
        include $RULE_PATH\backdoor.rules
        include $RULE_PATH\bad-traffic.rules
        include $RULE_PATH\blacklist.rules
        include $RULE_PATH\botnet-cnc.rules
        include $RULE_PATH\browser-chrome.rules
        include $RULE_PATH\browser-firefox.rules
        include $RULE_PATH\browser-ie.rules
        include $RULE_PATH\browser-other.rules
        include $RULE_PATH\browser-plugins.rules
        include $RULE_PATH\browser-webkit.rules
        include $RULE_PATH\chat.rules
        include $RULE_PATH\content-replace.rules
        include $RULE_PATH\ddos.rules
        include $RULE_PATH\dns.rules
        include $RULE_PATH\dos.rules
        include $RULE_PATH\experimental.rules
        include $RULE_PATH\exploit-kit.rules
        include $RULE_PATH\exploit.rules
        include $RULE_PATH\file-executable.rules
        include $RULE_PATH\file-flash.rules
        include $RULE_PATH\file-identify.rules
        include $RULE_PATH\file-image.rules
        include $RULE_PATH\file-multimedia.rules
        include $RULE_PATH\file-office.rules
        include $RULE_PATH\file-other.rules
        include $RULE_PATH\file-pdf.rules
        include $RULE_PATH\finger.rules
        include $RULE_PATH\ftp.rules
        include $RULE_PATH\icmp-info.rules
        include $RULE_PATH\icmp.rules
        include $RULE_PATH\imap.rules
        include $RULE_PATH\indicator-compromise.rules
        include $RULE_PATH\indicator-obfuscation.rules
        include $RULE_PATH\indicator-shellcode.rules
        include $RULE_PATH\info.rules
        include $RULE_PATH\malware-backdoor.rules
        include $RULE_PATH\malware-cnc.rules
        include $RULE_PATH\malware-other.rules
        include $RULE_PATH\malware-tools.rules
        include $RULE_PATH\misc.rules
        include $RULE_PATH\multimedia.rules
        include $RULE_PATH\mysql.rules
        include $RULE_PATH\netbios.rules
        include $RULE_PATH\nntp.rules
        include $RULE_PATH\oracle.rules
        include $RULE_PATH\os-linux.rules
        include $RULE_PATH\os-other.rules
        include $RULE_PATH\os-solaris.rules
        include $RULE_PATH\os-windows.rules
        include $RULE_PATH\other-ids.rules
        include $RULE_PATH\p2p.rules
        include $RULE_PATH\phishing-spam.rules
        include $RULE_PATH\policy-multimedia.rules
        include $RULE_PATH\policy-other.rules
```

include $RULE_PATH\policy.rules
include $RULE_PATH\policy-social.rules
include $RULE_PATH\policy-spam.rules
include $RULE_PATH\pop2.rules
include $RULE_PATH\pop3.rules
include $RULE_PATH\protocol-finger.rules
include $RULE_PATH\protocol-ftp.rules
include $RULE_PATH\protocol-icmp.rules
include $RULE_PATH\protocol-imap.rules
include $RULE_PATH\protocol-pop.rules
include $RULE_PATH\protocol-services.rules
include $RULE_PATH\protocol-voip.rules
include $RULE_PATH\pua-adware.rules
include $RULE_PATH\pua-other.rules
include $RULE_PATH\pua-p2p.rules
include $RULE_PATH\pua-toolbars.rules
include $RULE_PATH\rpc.rules
include $RULE_PATH\rservices.rules
include $RULE_PATH\scada.rules
include $RULE_PATH\scan.rules
include $RULE_PATH\server-apache.rules
include $RULE_PATH\server-iis.rules
include $RULE_PATH\server-mail.rules
include $RULE_PATH\server-mssql.rules
include $RULE_PATH\server-mysql.rules
include $RULE_PATH\server-oracle.rules
include $RULE_PATH\server-other.rules
include $RULE_PATH\server-webapp.rules
include $RULE_PATH\shellcode.rules
include $RULE_PATH\smtp.rules
include $RULE_PATH\snmp.rules
include $RULE_PATH\specific-threats.rules
include $RULE_PATH\spyware-put.rules
include $RULE_PATH\sql.rules
include $RULE_PATH\telnet.rules
include $RULE_PATH\tftp.rules
include $RULE_PATH\virus.rules
include $RULE_PATH\voip.rules
include $RULE_PATH\web-activex.rules
include $RULE_PATH\web-attacks.rules
include $RULE_PATH\web-cgi.rules
include $RULE_PATH\web-client.rules
include $RULE_PATH\web-coldfusion.rules
include $RULE_PATH\web-frontpage.rules
include $RULE_PATH\web-iis.rules
include $RULE_PATH\web-misc.rules
include $RULE_PATH\web-php.rules
include $RULE_PATH\x11.rules

##
Step #8: Customize your preprocessor and decoder alerts
For more information, see README.decoder_preproc_rules
##

decoder and preprocessor event rules
include $PREPROC_RULE_PATH\preprocessor.rules
include $PREPROC_RULE_PATH\decoder.rules
include $PREPROC_RULE_PATH\sensitive-data.rules

##
Step #9: Customize your Shared Object Snort Rules

```
        # For more information, see http://vrt-blog.snort.org/2009/01/using-vrt-certified-shared-object-rules.html
        ##################################################################

        # dynamic library rules
        # include $SO_RULE_PATH/bad-traffic.rules
        # include $SO_RULE_PATH/chat.rules
        # include $SO_RULE_PATH/dos.rules
        # include $SO_RULE_PATH/exploit.rules
        # include $SO_RULE_PATH/icmp.rules
        # include $SO_RULE_PATH/imap.rules
        # include $SO_RULE_PATH/misc.rules
        # include $SO_RULE_PATH/multimedia.rules
        # include $SO_RULE_PATH/netbios.rules
        # include $SO_RULE_PATH/nntp.rules
        # include $SO_RULE_PATH/p2p.rules
        # include $SO_RULE_PATH/smtp.rules
        # include $SO_RULE_PATH/snmp.rules
        # include $SO_RULE_PATH/specific-threats.rules
        # include $SO_RULE_PATH/web-activex.rules
        # include $SO_RULE_PATH/web-client.rules
        # include $SO_RULE_PATH/web-iis.rules
        # include $SO_RULE_PATH/web-misc.rules
```

Додаток Г. Теза доповіді на тему проект системи виявлення атак на основі snort в IT-інфраструктурі

Ульяніцький М. А., магістрант,

Фальковський І. Г., ст. викладач,

Державний університет «Житомирська політехніка»

ПРОЕКТ СИСТЕМИ ВИЯВЛЕННЯ АТАК НА ОСНОВІ SNORT В IT-ІНФРАСТРУКТУРІ

IDS/IPS відстежує весь трафік у мережі, щоб виявити будь-яку відому шкідливу поведінку. Одним із способів, якими зловмисник намагатиметься скомпрометувати мережу, є використання вразливості в пристрої або програмному забезпеченні. IDS/IPS ідентифікує ці спроби використання та блокує їх до того, як вони успішно скомпрометують будь-яку кінцеву точку в мережі.

Для виявлення інцидентів зазвичай використовуються три методології виявлення IDS:

- Виявлення на основі сигнатур порівнює сигнатури з спостережуваними подіями, щоб визначити можливі інциденти. Це найпростіший метод виявлення, оскільки він порівнює лише поточну одиницю активності (наприклад, пакет або запис у журналі зі списком підписів) за допомогою операцій порівняння рядків.

- Виявлення на основі аномалій порівнює визначення того, що вважається нормальною активністю, із спостережуваними подіями, щоб виявити значні відхилення. Цей метод виявлення може бути дуже ефективним для виявлення раніше невідомих загроз.

- Аналіз протоколу з урахуванням стану порівнює попередньо визначені профілі загальноприйнятих визначень доброякісної активності протоколу для кожного стану протоколу з спостережуваними подіями, щоб виявити відхилення.

Snort – це система виявлення вторгнень (IDS) та інтрузійно-профілактична система (IPS) в одному рішенні. Вона використовується для аналізу мережевого трафіку на предмет шкідливої або небажаної активності. У режимі IDS Snort виявляє потенційні загрози, а в режимі IPS може блокувати або відхиляти цей трафік в реальному часі.

Suricata – це конкурент системи Snort на ринку середнього бізнесу з відкритим вихідним кодом, що вперше була представлена у 2010 році. Однією з переваг Suricata є те, що вона є молодою системою, що дозволяє уникнути великої кількості застарілого коду. Крім того, вона використовує новіші технології, що сприяє її швидкій роботі порівняно з конкурентами. Розробники Suricata також піклуються про сумісність зі стандартними утилітами аналізу результатів, що дозволяє їй підтримувати ті ж модулі, що й Snort. Ця система може виявляти загрози за допомогою сигнатур і підходить для середніх і великих компаній.

McAfee Network Security Platform за стартову ціну близько $10 000. Ця система виявлення вторгнень (IDS) здатна ефективно блокувати широкий спектр загроз, включаючи доступ до шкідливих сайтів та запобігання DDoS-атакам. Проте через свою розмахність, вона може сповільнити роботу мережі. Тут важливо вирішити, що пріоритетнішим: інтеграція з іншими сервісами або максимальний рівень безпеки.

Zeek – це безкоштовна система виявлення вторгнень з відкритим вихідним кодом. Вона підтримує як стандартний режим виявлення вторжень, так і режим виявлення шкідливих сигнатур. Окрім цього, Zeek може виявляти події і дозволяє користувачам визначати власні скрипти політик. Однак недоліком Zeek є складність

взаємодії з інструментом через відсутність графічного інтерфейсу, оскільки розробка спрямована на функціональність.

Системи виявлення вторгнень (IDS) і системи запобігання вторгненням (IPS) постійно спостерігають за вашою мережею, виявляючи можливі інциденти та записуючи інформацію про них, зупиняючи інциденти та повідомляючи про них адміністраторам безпеки. Крім того, деякі мережі використовують IDS/IPS для виявлення проблем із політиками безпеки та запобігання порушенням політик безпеки.

У доповіді буде представлено детальний огляд систем виявлення вторгнень (IDS) і систем запобігання вторгненням (IPS), їх роль і значення в інфраструктурі безпеки мережі. Доповідь розгляне принципи роботи цих систем, їхні можливості виявлення та реагування на потенційні загрози, а також їхні переваги та недоліки. Також буде розглянуто важливість IDS/IPS у забезпеченні відповідності політикам безпеки та їхню роль у запобіганні порушенням цих політик. На доповіді буде розглянуто сучасні тенденції та рекомендації з використання IDS/IPS у контексті сучасних загроз та вимог до безпеки мережі.

Список використаних джерел

1. "IDS/IPS: Intrusion Detection and Prevention Systems" – R. Lippmann, D. P. Fried, K. W. Ingols URL: https://www.researchgate.net/publication/368928254

2. Augmenting zero trust architecture to end points using blockchain: A state-of-the-artreview URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/spy2.191

Додаток Д. Теза доповіді на тему підсистема захисту клієнт-серверної мережі від кібератак

Ульяніцький М. А., магістрант,

Бродський Ю. Б., к.т.н.,доцент

Державний університет «Житомирська політехніка»

ПІДСИСТЕМА ЗАХИСТУ КЛІЄНТ-СЕРВЕРНОЇ МЕРЕЖІ ВІД КІБЕРАТАК

Кібератаки можуть виникнути в будь-який момент і на будь-якій інфраструктурі, тому питання кібербезпеки є надзвичайно важливим.

Підсистема захисту клієнт серверної мережі від атак на Active Directory є дуже актуальною в сучасному світі інформаційних технологій. Active Directory є одним з основних компонентів сучасних корпоративних мереж, і важливо забезпечити його захист від різного роду атак, таких як злом паролів, використання недійсних логінів і багато інших. До складу Active Directory входить кілька різних служб. Основною службою є служби домену, але Active Directory також включає служби полегшених каталогів (AD LDS), протокол доступу до спрощених каталогів (LDAP), служби сертифікатів або AD CS, служби федерації (AD FS) і служби керування правами (AD RMS). Різні команди працюють по всій Україні, і люди можуть підключатися до незахищених мереж, що робить їхні дані доступними для зловмисників.

Головна мета дослідження полягає в аналізі, розробці та впровадженні ефективної підсистеми захисту клієнтсерверної мережі з метою запобігання кібератак.

Мета дослідження реалізована на основі використання передових технологій та методів захисту, зокрема впровадження сучасних систем виявлення вторгнень, застосування шифрування даних, використання механізмів аутентифікації та авторизації, а також моніторингу та реагування на інциденти безпеки в реальному часі.

Захист клієнт-серверних мереж від кібератак є надзвичайно важливим завданням у сучасному цифровому світі. Для забезпечення безпеки таких мереж використовуються різноманітні технології та методи:

 Firewalls та Інтрузійна Детекція/Запобігання (IDS/IPS): Захищають мережу від несанкціонованого доступу та виявляють аномалії для запобігання атак;

 шифрування Даних та VPN: Забезпечують конфіденційність інформації та захищають передачу даних через зашифровані канали;

 сегментація мережі та політики безпеки: розділяє мережу для обмеження поширення атак та встановлення правил безпеки;

 антивірусне ПЗ та оновлення безпеки: захищають від вірусів та забезпечують встановлення патчів для закриття вразливостей.

Ці технології та методи становлять ключову підсистему захисту клієнт-серверних мереж від різноманітних кібератак.

Аналіз загроз у сфері кіберзахисту клієнт-серверних мереж полягає у виявленні потенційних атак та вразливостей, що можуть бути використані для несанкціонованого доступу до даних чи систем. Технічні рішення для захисту таких мереж включають в себе методи захисту, такі як застосування брандмауерів, впровадження систем виявлення вторгнень (IDS) та запобігання вторгнень (IPS), шифрування даних, регулярне оновлення програмного забезпечення, використання сегментації мережі, резервне копіювання даних, антивірусні заходи та навчання персоналу щодо кібербезпеки. Ці заходи спрямовані на попередження атак та зменшення

можливості їхнього успішного впровадження, що дозволяє забезпечити більш високий рівень безпеки в клієнт-серверних мережах.

Інноваційні підходи до кіберзахисту клієнт-серверних мереж включають застосування штучного інтелекту та машинного навчання для розпізнавання аномальної активності та передбачення потенційних загроз. Техніки аналізу поведінки користувачів та систем в реальному часі дозволяють виявляти неочікувані відхилення та швидко реагувати на кіберзагрози.

Отже, кібератаки стають все більшою загрозою для організацій та інфраструктури, особливо в умовах

сучасних конфліктів. Забезпечення кіберзахисту та розробка підсистем захисту для клієнт-серверних мереж є невід'ємною складовою в умовах постійно зростаючих кіберзагроз. Аналіз загроз та застосування технологій захисту, таких як шифрування, мережеві брандмауери, системи виявлення вторгнень та інші, демонструють необхідність постійного вдосконалення та адаптації заходів безпеки до нових викликів кіберпростору. Тому, лише комплексний, системний підхід та поєднання різноманітних технологій та методів захисту можуть забезпечити ефективний захист клієнт-серверних мереж від сучасних кібератак.

Список використаних джерел

1. Active Directory. URL: https://www.cyberark.com/what-is/active-directory/