**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

ННК "Інститут прикладного системного аналізу"
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

"___" _____2016 р.

# Дипломна робота

першого (бакалаврського) **рівня вищої освіти**
(першого (бакалаврського), другого (магістерського))

зі спеціальності7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код та назва спеціальності)

на тему: Сучасні засоби пошуку вразливостей Web-сайтів та серверів та аналіз їх можливостей і практичного використання

Виконав: студент 4 курсу, групи ДА-22
(шифр групи)

Унгул Володимир Валерійович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник к.т.н., доцент Цурін О.П. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант Економіка д.е.н., професор Семенченко Н.В. _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Нормоконтроль ст. викладач Бритов О.А. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2016 року

**Національний технічний університет України**
**«Київський політехнічний інститут»**

Факультет (інститут) ННК "Інститут прикладного системного аналізу"
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти Перший(Бакалаврський)
(перший (бакалаврський), другий (магістерський) або спеціаліста)

Спеціальність 7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____2016 р.

**ЗАВДАННЯ**
**на дипломний проект (роботу) студенту**
Унгулу Володимиру Валерійовичу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Сучасні засоби пошуку вразливостей Web-сайтів та серверів та аналіз їх можливостей і практичного використання

керівник проекту (роботи) Цурін Олег Пилипович, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 2016 р. №_____

2. Строк подання студентом проекту (роботи) _____

3. Вихідні дані до проекту (роботи) _____

Сканери вразливостей, перелік сайтів для сканування.
Форма реалізації – сторінка Web–сайту.
Мова програмування – PHP, Web-фреймворк – Yii 1.1.

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)

1. Провести розгляд вразливостей Web-сайтів та серверів, вибір та загальний огляд сканерів вразливостей, аналіз їх функціоналу.

2. Провести тестування обраними сканерами та проаналізувати отримані результати.
3. Розробити сторінку Web-сайту, яка міститиме інформацію про роботу з обраними сканерами вразливостей.
4. Виконати економічний аналіз програмного продукту.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, плакатів тощо)

1. Інтерфейс обраних сканерів.
2. Порівняльні характеристики обраних сканерів.
3. Ілюстрація сторінки Web-сайту з результатами роботи.

6. Консультанти розділів проекту (роботи)

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Економіка | Семенченко Н. В., професор | | |
| Основна частина | | | |

7. Дата видачі завдання 01.02.2016

Календарний план

| № з/п | Назва етапів виконання дипломного проекту (роботи) | Строк виконання етапів проекту (роботи) | Примітка |
|---|---|---|---|
| 1 | Отримання завдання | 01.02.2016 | |
| 2 | Збір інформації | 15.02.2016 | |
| 3 | Розгляд вразливостей Web-сайтів та серверів | 28.02.2016 | |
| 4 | Вибір та аналіз засобів пошуку вразливостей | 15.03.2016 | |
| 5 | Тестування обраних засобів та аналіз результатів | 30.03.2016 | |
| 6 | Розробка сторінки Web-сайту з використанням Yii 1.1. | 15.04.2016 | |
| 7 | Аналіз економічної обгрунтованості | 30.04.2016 | |
| 8 | Оформлення дипломної роботи | 31.05.2016 | |
| 9 | Отримання допуску до захисту та подача роботи в ДЕК | 14.06.2016 | |
| 10 | Захист дипломної роботи | 21.06.2016 | |

Студент _____ В.В.Унгул
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ О.П.Цурін
(підпис) (ініціали, прізвище)

# АНОТАЦІЯ

бакалаврської дипломної роботи Унгула Володимира Валерійовича
на тему : «Сучасні засоби пошуку вразливостей Web-сайтів та серверів та аналіз їх можливостей і практичного використання»

Дипломна робота присвячена аналізу засобів пошуку вразливостей Web-сайтів та серверів. Було розглянуто поширені вразливості Web-сайтів та серверів, обрано програмні засоби для їх сканування. Після цього було розглянуто їх інтерфейс та функціональні можливості.

Обраними засобами було проведено сканування обраного переліку сайтів. Було порівняно результати роботи обраних сканерів, а для знайдених вразливостей були запропоновані можливі рішення. Опис роботи зі сканерами було виконано у вигляді сторінки Web-сайту, який уже було введено в дію.

Загальний обсяг 120 сторінок , 58 рисунків , 9 таблиць , 15 посилань.

Ключові слова : Web-сайт, вразливість, сканер, XSpider, Acunetix, Shadow Security.

# АННОТАЦИЯ

бакалаврской дипломной работе Унгула Владимира Валерьевича
на тему: «Разработка виртуальной лаборатории функционально-логического моделирования»

Дипломная работа посвящена анализу средств поиска уязвимостней Web-сайтов и серверов. Были рассмотрены распространенные уязвимости Web-сайтов и серверов, выбраны программные средства для их сканирования. После этого был рассмотрены их интерфейс и функциональные возможности.

Выбранными средствами было проведено сканирование выбранного перечня сайтов. Было проведено сравнение результатов работы выбранных сканеров, а для найденных уязвимостей были предложены возможные решения. Описание работы со сканерами было выполнено в виде страницы Web-сайта, который уже был введен в действие.

Общий объем 120 страницы, 58 рисунков, 9 таблиц, 15 ссылок.

Ключевые слова: Web-сайт, уязвимость, сканер, XSpider, Acunetix, Shadow Security.

# ANNOTATION

a bachelor’s degree work of Yakymets Roman
entitled: "Modern finding vulnerabilities of Web-sites and servers and analyzing opportunities and practical use"

Degree work is dedicated to the analysis tools find vulnerabilities Web-sites and servers. It was a review of common security vulnerabilities of Web-sites and servers, selected software for scanning. There were review of interface and functionality.

Elected means held scanning the selected Web-sites on the list. It compared the results of the selected scanners, and for vulnerabilities found were offered possible solutions. Description of work with scanners were designed as a Web-page site that has been put into effect.

The total amount of work 120 pages, 58 figures, 9 table, 15 references.

Keywords: Web-site, vulnerability, scanner, XSpider, Acunetix, Shadow Security.

# ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ , СКОРОЧЕНЬ І ТЕРМІНІВ

WVS – Web Vulnerability Scanner

SSS – Shadow Security Scanner

HTTP – HyperText Transfer Protocol

HTML – HyperText Markup Language

SQL – Structured query language

DOS – Denial of Service

XSS – Cross Site Scripting

TCP – Transmission Control Protocol

PHP – Hypertext Preprocessor

ПП – Програмний продукт

# ВСТУП

Web-сервери та Web-сайти - це об’єкти, які постійно піддаються небезпекі. Особливу увагу слід звернути на Web-сервери, серйозну загрозу для яких становлять хакери і віруси. Перші можуть отримати доступ до конфіденційної інформації, розміщеної на сервері, зламати сайти і змінити їх вміст, а також вивести з ладу сервер за допомогою розподіленої атаки (DDoS-атака). Віруси ж, вражаючи веб-сервери, перетворюють їх у джерело інфекції. Крім того, вони істотно сповільнюють його роботу, а також займають інтернет-канал. Багато вірусів, особливо інтернет-черв'яки, використовують для розповсюдження вразливості в програмному забезпеченні. Також і хакери прагнуть спрямовувати атаки на відомі «дірки» в програмному забезпеченні. Використовуючи вразливості, і ті й інші одержують досить легкий доступ до віддаленого комп'ютера навіть у тому випадку, якщо останній добре захищений.

Практично у майже будь-якій програмі є вразливості. Наявність вразливостей легко пояснюється через здатність людей допускати помилки. Велике програмне забезпечення (ПЗ) пише не одна людина, а ціла група. І досить часто помилки виникають при компонуванні модулів, створених різними програмістами. Крім того, наявність вразливостей далеко не завжди визначається якістю написання ПЗ.

На сьогоднішній день компанії дуже рідко замислюються про безпеку своєї інформації в мережі і зовсім не приділяють цьому питанню уваги, часто починаючи вживати заходів лише після витоку або втрати важливої інформації.

Щоб забезпечити або ж усунути існуючу проблему, пов'язану із захистом інформації, застереження від атак зловмисників корпоративного сайту, його бази даних або всередині мережі додатків, у даній темі буде розглянуто рішення для діагностики вразливостей і моніторингу комп’ютерів в мережі,

спеціальні сканери - програмні або апаратні засоби, скануючі систему на предмет виявлення можливих проблем в безпеці, що дозволяють виявляти, оцінювати і усувати вразливості в мережі.

Сканери уразливості діляться на дві основні групи:

1. Сканери корпоративних мереж, призначення яких полягає в аналізі мережі на наявність відкритих портів, а також вразливостей в операційних системах і додатках.
2. Сканери уразливості веб-додатків. На даний момент їхня популярність зростає в силу того, що більшість комерційних організацій і банків використовують у своїй діяльності інтернет ресурси, захист яких стає важливим фактором. У цій роботі буде розглянуто більше інформації саме по цій групі.

Пропоновані продукти мають всі можливості і засоби для ефективного виявлення і управління виправленнями вразливостей, які створені після аналізу та фільтрації результатів.

Метою роботи є огляд і оцінка можливостей сучасних засобів пошуку вразливостей. Кінцевою метою є збір та аналіз результатів, наведенння шляхів боротьби зі знайденими вразливастями за допомогою обраних засобів, на основі сканування переліку сайтів:

- kpi.ua;
- cad.kpi.ua;
- cad.edu.kpi.ua;
- iasa.kpi.ua;
- its.kpi.ua.

Для опису роботи з обраними засобами буде розроблено сторінку Web-сайту. Отримані результати можна використати для покращення роботи просканованих сайтів.

# 1. ОГЛЯД ВРАЗЛИВОСТЕЙ WEB-САЙТІВ ТА СЕРВЕРІВ ТА ВИБІР ЗАСОБІВ ЇХ ПОШУКУ

В даному розділі будуть розглянуті відомі типи вразливостей Web-сайтів та серверів. Також будуть обрані та розглянуті сканери для пошуку вразлвостей. Далі буде проведено порівняння їх інтерфесу та функціоналу.

## 1.1. Вразливості програмного забезпечення та конфігурації

Коли мова йде про вразливість веб-серверів, то часто йдеться про «дірки» в їх програмному забезпеченні. Це відноситься до самих програм-серверів, таким як Apache, Microsoft Internet Information Server та інш.. Це програмне забезпечення досить об'ємне і складне, так що «дірки» в ньому обов'язково є. Сучасний веб-сервер неможливо уявити собі без багатьох додаткових функцій, наприклад без підтримки мов програмування типу Perl, PHP та інш., а також без систем управління базами даних. Все це стає можливим завдяки установці на веб-сервер додаткового ПЗ, яке теж може мати вразливості.

Головною особливістю виробничих вразливостей є їх прихильність до певних версій програмного забезпечення. Справа в тому, що «дірки» часто зустрічаються не у всій лінійці веб-серверів, а тільки в деяких їх релізах. А ще варто відзначити, що чим популярніше те чи інше програмне забезпечення, тим частіше для нього знаходять нові вразливості.

Захиститися від розглянутого типу вразливостей програмного забезпечення можна лише своєчасною установкою всіх розроблених виробниками патчів (оновлених версій). Справа в тому, що розробники софта регулярно викладають на офіційних сайтах оновлення для своїх продуктів. При виявленні критичної для безпеки «дірки» патч випускається швидко (якщо, звичайно, компанія дійсно піклується про своїх клієнтів). Якщо ж знову знайдені вразливості несуть швидше теоретичну, ніж реальну загрозу, то в міру їх накопичення випускаються кумулятивні патчі.

### 1.1.1. Вразливості конфігурації

Вразливості можуть виникати через некоректне налаштування програмного забезпечення веб-сервера. Переважна більшість веб-серверів мають досить великий набір параметрів, що стосуються практично всіх аспектів його роботи. Таким чином, безпека багато в чому залежить від адміністраторів, що займаються їх обслуговуванням. Через неуважність, недостатню кваліфікацію чи ще з якихось причин вони також можуть помилятися. І ці помилки можуть відкрити дорогу до веб-сервера хакеру або вірусу.

Від некоректного налаштування не може допомогти установка патчів так як, при оновленні ПЗ його конфігурація не змінюється. Це означає, що вразливість в системі захисту після інсталяції патча швидше за все залишиться. Головною небезпекою розглянутого типу «дірок» є складність їх виявлення. Єдиний спосіб дійсно надійного захисту від таких вразливостей - використання спеціальних сканерів безпеки. Ці програми за допомогою спеціальних методів досліджують захист веб-серверів і знаходять потенційно небезпечні місця.

### 1.1.2. Вразливості власного програмного забезпечення

Скрипти веб-сайтів теж можуть містити вразливості. Сучасний веб-сервер і супроводжуюче ПЗ дуже часто служать своєрідною базою для виконання програм, що написані власноруч користувачем. Йдеться про скрипти, які працюють на більшості сучасних сайтів. Справа в тому, що більшість мов веб-програмування є серверними. Це означає, що скрипти, написані на них, виконуються прямо на сервері, а на комп'ютер користувача (в даному випадку - відвідувача сайту) відправляються тільки результати їх роботи. В цьому є досить серйозна небезпека бо скрипти для сайтів далеко не завжди розробляються дійсно гарними спеціалістами. На багатьох веб-проектах використовуються безкоштовно поширювані програми або ж софт власного написання, який теж може містити помилки. Причому деякі з них можуть бути

дуже серйозними, що дозволяють зловмисникам дістати несанкціонований доступ до самого серверу.

Виявити «дірки» в скриптах можна за допомогою сканерів безпеки. Якщо власник веб-сервера дійсно піклується про безпеку свого сайту, то повинен періодично перевіряти його бо хакери постійно вигадують нові способи віддалених атак. Крім того, постійно виявляються нові «дірки» в оригінальному ПЗ, які можуть у поєднанні зі скриптами, які раніше вважалися безпечними, являти собою реальну загрозу.

## 1.2. Класифікація вразливостей і атак.

Розглянемо класифікацію основних вразливостей та атак, побудованих на використанні даних вразливостей.

### 1.2.1. Аутентифікація (Authentication)

Аутентифікація використовує як мінімум один з трьох механізмів (факторів): "щось, що ми маємо", "щось, що ми знаємо" або "щось, що ми є". У цьому розділі описуються атаки, спрямовані на обхід або експлуатацію вразливостей в механізмах реалізації аутентифікації Web-серверів.

#### 1.2.1.1. Підбір (Brute Force)

Багато систем дозволяють використовувати слабкі паролі або ключі шифрування, і користувачі часто вибирають легко вгадувані або наявні у словниках парольні фрази.

Існує два види підбору: прямий і зворотний. При прямому підборі використовуються різні варіанти пароля для одного імені користувача. При зворотному перебираются різні імена користувачів, а пароль залишається незмінним. У системах з мільйонами облікових записів ймовірність використання різними користувачами одного пароля досить велика. Не

дивлячись на популярність і високу ефективність, підбір може займати кілька годин, днів або років.

Приклад:

- Им’я користувача = Jon
- Паролі = smith, michael-jordan, [pet names], [birthdays], [car names], ...
- Имена користувачів = Jon, Dan, Ed, Sara, Barbara, .....
- Пароль = 12345678

#### 1.2.1.2. Недостатня аутентифікація (Insufficient Authentication)

Ця вразливість виникає, коли Web-сервер дозволяє атакуючому отримувати доступ до важливої інформації або функцій сервера без належної аутентифікації. Інтерфейси адміністрування через Web - яскравий приклад критичних систем.

Залежно від специфіки програми, подібні компоненти не повинні бути доступні без належної аутентифікації. Щоб не використовувати аутентифікацію деякі ресурси "ховаються" за певною адресою, на яку немає посилань з основних сторінках сервера або інших загальнодоступних ресурсах. Однак, подібний підхід не більш ніж "безпека через приховування". Важливо розуміти, що, не дивлячись на те, що зловмисник не знає адреси сторінки, вона все одно доступна через Web. Необхідний URL може бути знайдений перебором типових файлів і директорій (таких як / admin /), з використанням повідомлень про помилки, журналів перехресних посилань або шляхом простого читання документації. Подібні ресурси повинні бути захищені адекватно важливості їх вмісту і функціональних можливостей.

#### 1.2.1.3. Небезпечне відновлення паролів (Weak Password Recovery Validation)

Ця уразливість виникає, коли Web-сервер дозволяє атакуючому несанкціоновано отримувати, модифіковані або відновлювати паролі інших

користувачів. Функція відновлення пароля є важливою складовою що надається Web-серверами сервісу.

Система відновлення пароля може бути скомпрометована шляхом використання підбору, вразливостей системи або легко вгадується відповіді на секретне питання.

Приклад - слабкі методи відновлення паролів.

Перевірка інформації

Багато серверів вимагають від користувача вказати його email в комбінації з домашньою адресою і номером телефону. Ця інформація може бути легко отримана з мережевих довідників. В результаті, дані, які використовуються для перевірки, не є великим секретом. Крім того, ця інформація може бути отримана зловмисником з використанням інших методів, таких як міжсайтового виконання сценаріїв.

Парольні підказки

Сервер, що використовує підказки для полегшення запам'ятовування паролів, може бути атакований, оскільки підказки допомагають у реалізації підбору паролів. Користувач може використовувати стійкий пароль, наприклад, "221277King" з відповідною підказкою: "д-р + люб письменник". Атакуючий може припустити, що для користувача пароль складається з дати народження та імені улюбленого автора користувача. Це допомагає сформувати відносно короткий словник для атаки шляхом перебору.

Таємне питання та відповідь

Припустимо, відповідь користувача "Київ", а секретне питання "Місце народження". Зловмисник може обмежити словник для підбору секретної відповіді назвами міст. Більш того, якщо атакуючий має в своєму

розпорядженні деякою інформацією про користувача, дізнатися його місце народження не складно.

### 1.2.2. Авторизація (Authorization)

Даний розділ присвячений атакам, направленим на методи, які використовуються Web-сервером для визначення того, чи має користувач, служба або програма необхідний дозвіл для вчинення дії. Багато Web-сайтів дозволяють тільки певним користувачам отримувати доступ до деякого вмісту або функцій програми. Доступ іншим користувачам повинен бути обмежений. Використовуючи різні технології, зловмисник може підвищити свої привілеї і отримати доступ до захищених ресурсів.

#### 1.2.2.1. Передбачуване значення ідентифікатора сесії (Credential/Session Prediction)

Передбачуване значення ідентифікатора сесії дозволяє перехоплювати сесії інших користувачів. Подібні атаки виконуються шляхом передбачення або вгадування унікального ідентифікатора сесії користувача. Ця атака також як і перехоплення сесії (Session Hijacking) у разі успіху дозволяє зловмисникові послати запит Web-сервера з правами скомпрометованого користувача. Дизайн багатьох серверів припускає аутентифікацію користувача при першому зверненні та подальше відстеження його сесії. Для цього користувач вказує комбінацію імені та пароля. Замість повторної передачі ім'я користувача та пароль при кожній транзакції, Web-сервер генерує унікальний ідентифікатор, який присвоюється сесії користувача. Наступні запити користувача до сервера містять ідентифікатор сесії як доказ того, що аутентифікація була успішно пройдено. Якщо атакуючий може передбачити або вгадати значення ідентифікатора іншого користувача, це може бути використано для проведення атаки.

Приклад

Багато серверів генерують ідентифікатори сесії, використовуючи алгоритми власної розробки. Подібні алгоритми можуть просто збільшувати значення ідентифікатора для кожного запиту користувача. Інший поширений варіант - використання функції від поточного часу або інших специфічних для комп'ютера даних. Ідентифікатор сесії зберігається в cookie, прихованих полях форм або URL. Якщо атакуючий має можливість визначити алгоритм, використовуваний для генерації сесії, він може виконати наступні дії:

1) підключитися до сервера, використовуючи поточний ідентифікатор сесії;

2) обчислити або підібрати наступний ідентифікатор сесії;

3) присвоїти отримане значення ідентифікатора cookie / прихованого полю форми / URL.

#### 1.2.2.2. Недостатня авторизація (Insufficient Authorization)

Недостатня авторизація виникає, коли Web-сервер дозволяє атакуючому отримувати доступ до важливої інформації або функцій, доступ до яких повинен бути обмежений. Правильно побудовані правила доступу повинні обмежувати дії користувача відповідно до політики безпеки. Доступ до важливих ресурсів сайту повинен бути дозволений тільки адміністраторам.

Приклад

У минулому багато Web-серверів зберігали важливі ресурси в "прихованих" директоріях, таких як "/ admin" або "/ log". Якщо атакуючий запитував ці ресурси напряму, він отримував до них доступ і міг переналаштувати сервер, отримати доступ до важливої інформації або повністю скомпрометувати систему.

Деякі сервери, після аутентифікації, зберігають у cookie або прихованих полях ідентифікатор "ролі" користувача в рамках Web-додатків. Якщо розмежування доступу грунтується на перевірці даного параметра без

верифікації приналежності до ролі при кожному запиті, зловмисник може підвищити свої привілеї, просто модифікувавши значення cookie.

Наприклад, значення cookie SessionId=12345678; Role=User

замінюється на SessionId=12345678; Role=Admin

#### 1.2.2.3 Відсутність таймауту сесії (Insufficient Session Expiration)

У випадку якщо для ідентифікатора сесії або облікових даних не передбачений таймаут або має значення дуже велике, зловмисник може скористатися старими даними для авторизації. Це підвищує уразливість сервера для атак, пов'язаних з крадіжкою ідентифікаційних даних. Оскільки протокол HTTP не передбачає контроль сесії, Web-сервери зазвичай використовують ідентифікатори сесії для визначення запитів користувача. Таким чином, конфіденційність кожного ідентифікатора повинна бути забезпечена, щоб запобігти багаторазовий доступ користувачів з одним профілем.

Викрадений ідентифікатор може використовуватися для доступу до даних користувача або здійснення шахрайських транзакцій. Відсутність таймауту сесії збільшує ймовірність успіху різних атак. Приміром, зловмисник може отримати ідентифікатор сесії, використовуючи мережевий аналізатор або вразливість типу міжсайтового виконання сценаріїв.

В іншій ситуації, якщо користувач отримує доступ до сервера з публічного комп'ютера (бібліотека, Internet-кафе і т.д.) відсутність таймауту сесії може дозволити зловмисникові скористатися історією браузера для перегляду сторінок користувача. Велике значення таймауту збільшує шанси підбору чинного ідентифікатора. Крім того, збільшення цього параметра веде до збільшення одночасно відкритих сесій, що ще більше підвищує ймовірність успішного підбору.

#### 1.2.2.4. Фіксація сесії (Session Fixation)

Використовуючи даний клас атак, зловмисник присвоює ідентифікатору сесії користувача задане значення. Залежно від функціональних можливостей сервера, існує декілька способів "зафіксувати" значення ідентифікатора сесії. Для цього можуть використовуватися атаки типу міжсайтового виконання сценаріїв або підготовка сайту з допомогою попереднього HTTP запиту. Після фіксації значення ідентифікатора сесії атакуючий очікує моменту, коли користувач увійде в систему. Після входу користувача, зловмисник використовує ідентифікатор сесії для отримання доступу до системи від імені користувача.

Можна виділити два типи систем управління сесіями на основі ідентифікаторів. Перший з них, "дозволяючий", дає змогу браузеру вказувати будь-який ідентифікатор. Системи другого "суворого" типу обробляють тільки ідентифікатори, згенеровані сервером. Якщо використовуються "дозволяючі" системи, зловмисник може вибрати будь-який ідентифікатор сесії. У випадку із "суворими" серверами зловмисникові доводиться підтримувати "сесію-заглушку" і періодично з'єднуватися з сервером для уникнення закриття сесії за таймаутом.

Без наявності активного захисту від фіксації сесії, ця атака може бути використана проти будь-якого сервера, аутентифікуючи користувачів за допомогою ідентифікатора сесії. Більшість Web-серверів зберігає ID в cookie, але це значення так само може бути присутнім в URL або прихованому полі форми. Системи, що використовують cookie, є найбільш уразливими. Більшість відомих на даний момент варіантів фіксації сесії спрямовані саме на значення cookie. На відміну від крадіжки ідентифікатора, фіксація сесії надає зловмисникові набагато більше можливостей. Це пов'язано з тим, що активна фаза атаки відбувається до входу користувача в систему.

Приклад

Атаки, спрямовані на фіксацію сесії зазвичай проходять у три етапи.

- Встановлення сесії

Зловмисник встановлює сесію-заглушку на атакуючому сервер і отримує від сервера ідентифікатор або вибирає довільний ідентифікатор. У деяких випадках сесія-заглушка повинна підтримуватися в активному стані шляхом періодичних звернень до сервера.

- Фіксація сесії

Зловмисник передає значення ідентифікатора сесії-заглушки браузеру користувача та фіксує його ідентифікатор сесії. Це можна зробити, наприклад, встановивши значення cookie в браузері за допомогою XSS.

- Підключення до сесії

Атакуючий очікує аутентифікації користувача на сервері. Після того, як користувач зайшов на сайт, зловмисник підключається до сервера, використовуючи зафіксований ідентифікатор, і отримує доступ до сесії користувача.

### 1.2.3. Атаки на клієнтів (Client-side Attacks)

Цей розділ описує атаки на користувачів Web-сервера. Користувач очікує, що сайт надасть йому легітимний вміст. Крім того, користувач не очікує атак з боку сайту. Експлуатуючи цю довіру, зловмисник може використовувати різні методи для проведення атак на клієнтів сервера.

#### 1.2.3.1. Підміна вмісту (Content Spoofing)

Використовуючи цю техніку, зловмисник змушує користувача повірити, що сторінки згенеровані Web-сервером, а не передані з зовнішнього джерела. Деякі Web-сторінки створюються з використанням динамічних джерел HTML- коду.

Якщо атакуючий спровокував користувача на перехід по спеціально створеному посиланню, у користувача може скластися враження, що він переглядає дані з сервера, в той час як частина їх була згенерована зловмисником. Ця атака так само може використовуватися для створення помилкових сторінок, таких як форми введення пароля, прес-релізи і т.д.

#### 1.2.3.2. Міжсайтове виконання сценаріїв (Cross-site Scripting, XSS)

Наявність уразливості Cross-site Scripting дозволяє атакуючому передати серверу код, який буде виконуватися перенаправлено браузеру користувача. Цей код зазвичай створюється на мовах HTML / JavaScript, але можуть бути використані VBScript, ActiveX, Java, Flash, або інші мови які підтримує браузер.

Переданий код виконується в контексті безпечності (або зоні безпеки) уразливого сервера. Використовуючи ці привілеї, код отримує можливість читати, модифікувати або передавати важливі дані, доступні за допомогою браузера. У атакованого користувача може бути скомпрометований аккакунт (крадіжка cookie), його браузер може бути перенаправлений на інший сервер або здійснена підміна вмісту сервера. У результаті ретельно спланованої атаки зловмисник може використовувати браузер жертви для перегляду сторінок сайту від імені атакуємого користувача. Код може передаватися зловмисником в URL, в заголовках HTTP запиту (cookie, user-agent, refferer), значеннях полів форм і т.д.

Існує два типи атак, що приводять до міжсайтового виконання сценаріїв: постійні (збережені) і непостійні (відображені). Основною відмінністю між ними є те, що у другому варіанті передача коду сервером та повернення його клієнту здійснюється в рамках одного HTTP-запиту, а в першому - в різних.

#### 1.2.3.3. Розщеплення HTTP-запиту (HTTP Response Splitting)

При використанні даної уразливості зловмисник посилає серверу спеціальним чином сформований запит, відповідь на який інтерпретується

метою атаки як дві різні відповіді. Друга відповідь повністю контролюється зловмисником, що дає йому можливість підробити відповідь сервера.

У реалізації атак з розщепленням HTTP-запиту беруть участь як мінімум три сторони:

- Web-сервер, який містить подібну уразливість.
- Мета атаки, що взаємодіють з Web-сервером під управлінням зловмисника.
- Типово в якості мети атаки виступає кешуючий сервер-посередник або кеш браузера.
- Атакуючий, який ініціює атаку.

Можливість здійснення атаки виникає, коли сервер повертає дані, надані користувачем в заголовках HTTP відповіді. Зазвичай це відбувається при перенаправлені користувача на іншу сторінку (коди HTTP 3xx) або коли дані, отримані від користувача, зберігаються в cookie.

У першій ситуації URL, на який відбувається перенаправленя, є частиною заголовка Location HTTP відповіді, а в другому випадку значення cookie передається в заголовку Set-Cookie. Основою розщеплення HTTP-запиту є впровадження символів переведення рядка (CR і LF) таким чином, щоб сформувати дві HTTP транзакції, в той час як реально буде відбуватися тільки одна. Переклад рядка використовується для того, що б закрити першу (стандартну) транзакцію, і сформувати другу пару питання / відповідь, повністю контрольовану зловмисником і абсолютно непередбачувану логікою програми.

У результаті успішної реалізації цієї атаки зловмисник може виконати наступні дії:

- Міжсайтового виконання сценаріїв.

- Модифікація даних кеша сервера-посередника. Деякі кешуючий сервери-посередники (Squid 2.4, NetCache 5.2, Apache Proxy 2.0 і ряд інших), зберігають підроблений зловмисником відповідь на жорсткому диску і на подальші запити користувачів за цією адресою повертають кешовані дані. Це призводить до заміни сторінок сервера на стороні клієнта. Крім цього, зловмисник може переправити собі значення Cookie користувача або присвоїти їм певне значення. Так само ця атака може бути спрямована на індивідуальний кеш браузера користувача.
- Міжкористувацька атака (один користувач, одна сторінка, тимчасова підміна сторінки). При реалізації цієї атаки зловмисник не посилає додатковий запит. Замість цього використовується той факт, що деякі сервери-посередники поділяють одне TCP-з'єднання до сервера між декількома користувачами. У результаті другий користувач отримує у відповідь сторінку, сформовану зловмисником. Крім підміни сторінки зловмисник може також виконати різні операції з cookie користувача.
- Перехоплення сторінок, що містять дані користувача. У цьому випадку зловмисник отримує відповідь сервера замість самого користувача. Таким чином, він може отримати доступ до важливої або конфіденційної інформації.

### 1.2.4. Виконання коду (Command Execution)

Ця секція описує атаки, спрямовані на виконання коду на Web-сервері. Всі сервери використовують дані, віддані користувачем при обробці запитів. Часто ці дані використовуються при складанні команд, що застосовуються для генерації динамічного вмісту. Якщо при розробці не враховуються вимоги безпеки, зловмисник отримує можливість модифікувати виконавчі команди.

#### 1.2.4.1. Переповнення буфера (Buffer Overflow)

Переповнення буфера є найбільш поширеною причиною помилок у програмах. Воно виникає, коли обсяг даних перевищує розмір виділеної під них

буфера. Коли буфер переповнюється, дані переписують інші області пам'яті, що призводить до виникнення помилки. Якщо зловмисник має можливість управляти процесом переповнення, це може викликати ряд серйозних проблем.

Переповнення буфера може викликати відмови в обслуговуванні, приводячи до пошкодження пам'яті і викликаючи помилки в програмах. Більш серйозні ситуації дозволяють змінити шлях виконання програми і виконати в її контексті різні дії. Це може відбуватися в кількох випадках. Використовуючи переповнення буферу, можна перезаписувати службові області пам'яті, наприклад, адресу повернення з функцій у стеці. Також, при переповненні можуть бути переписані значення змінних у програмі.

Переповнення буфера є найбільш поширеною проблемою в безпеці і нерідко зачіпає Web-сервери. Проте атаки, що експлуатують цю уразливість, використовуються проти Web-додатків не дуже часто. Причина цього криється в тому, що атакуючому, як правило, необхідно проаналізувати вихідний код або образ програми. Оскільки атакуючому доводиться експлуатувати нестандартну програму на віддаленому сервері, йому доводиться атакувати "всліпу", що знижує шанси на успіх.

#### 1.2.4.2. Атака на функції форматування рядків (Format String Attack)

При використанні цих атак шлях виконання програми модифікується методом перезапису областей пам'яті за допомогою функцій форматування символьних змінних. Вразливість виникає, коли дані користувача застосовуються в якості аргументів функцій форматування рядків, таких як fprintf, printf, sprintf, setproctitle, syslog і т.д. Якщо атакуючий передає додатком рядок, що містить символи форматування ("% f", "% p", "% n" і т.д.), то в нього з'являється можливість:

- Виконати довільний код на сервері;
- Зчитувати значення з стека;
- Викликати помилки в програмі / відмова в обслуговуванні.

Приклад

Припустимо, Web-додаток зберігає параметр emailAddress для кожного користувача. Це значення використовується в якості аргументу функції printf: printf (emailAddress);

Якщо значення змінної emailAddress містить символи форматування, функція printf буде обробляти їх згідно закладеної в неї логіки. Оскільки додаткових значень цієї функції не передано, будуть використані значення стека, що зберігають інші дані.

Можливі наступні методи експлуатації атак на функції форматування рядків:

- Читання даних з стека: Якщо вивід функції printf передається атакуючому, він отримує можливість читання даних з стека, використовуючи символ форматування "% x".
- Читання рядків з пам'яті процесу: Якщо вивід функції printf передається атакуючому, він може отримувати рядки з пам'яті процесу, передаючи в параметрах символ "% s".
- Запис цілочисельних значень в пам'ять процесу: Використовуючи символ форматування "% n", зловмисник може зберігати цілочисельні значення в пам'яті процесу. Таким чином можна переписати важливі значення, наприклад права управління доступом або адресу повернення.

### 1.2.4.3 Впровадження операторів LDAP (LDAP Injection)

Атаки цього типу спрямовані на Web-сервери, що створюють запити до служби LDAP на основі даних, які вводяться користувачем. Спрощений протокол доступу до служби каталогу (Lightweight Directory Access Protocol, LDAP) - відкритий протокол для створення запитів і управління службами каталогу сумісними зі стандартом X.500.

Протокол LDAP працює поверх транспортних протоколів Internet (TCP / UDP). Web-додаток може використовувати дані, надані користувачем для

створення запитів по протоколу LDAP при генерації динамічних Web-сторінок. Якщо інформація, отримана від клієнта, належним чином не верифікуються, атакуючий отримує можливість модифікувати LDAP-запит.

Запит буде виконуватися з тим же рівнем привілеїв, з яким працює компонент програми, що виконує запит (сервер СУБД, Web-сервер і т.д). Якщо цей компонент має права на читання або модифікацію даних у структурі каталогу, зловмисник отримує ті ж можливості.

#### 1.2.4.4. Виконання команд ОС (OS Commanding)

Атаки цього класу спрямовані на виконання команд операційної системи на Web- сервері шляхом маніпуляції вхідними даними. Якщо інформація, отримана від клієнта, належним чином не верифікуються, атакуючий отримує можливість виконати команди ОС. Вони будуть виконуватися з тим же рівнем привілеїв, з яким працює компонент програми, що виконує запит (сервер СУБД, Web-сервер і т.д).

#### 1.2.4.5 Впровадження операторів SQL (SQL Injection)

Ці атаки спрямовані на Web-сервери, що створюють SQL запити до серверів СУБД на основі даних, які вводяться користувачем. Більшість серверів підтримують цю мову у варіантах, стандартизованих ISO та ANSI. У більшості сучасних СУБД присутні розширення діалекту SQL, специфічні для даної реалізації (T-SQL в Microsoft SQL Server, - PL SQL в Oracle і т.д.). Багато Web-додатки використовують дані, передані користувачем, для створення динамічних Web-сторінок. Якщо інформація, отримана від клієнта, належним чином не верифікуються, атакуючий отримує можливість модифікувати запит до SQL-серверу, що відправляється додатком. Запит буде виконуватися з тим же рівнем привілеїв, з яким працює компонент програми, що виконує запит (сервер СУБД, Web-сервер і т.д). У результаті зловмисник може отримати повний контроль на сервером СУБД і навіть його операційною системою. З точки зору експлуатації SQL Injection дуже схожа на LDAP Injection.

Приклад

Припустимо, аутентифікація в Web-додаток здійснюється за допомогою Web- форми, оброблюваної наступним кодом:

SQLQuery = "SELECT Username FROM Users WHERE Username = '"& strUsername &"' AND Password = '"  & StrPassword & "'" strAuthCheck = GetQueryResult (SQLQuery)

У цьому випадку розробники безпосередньо використовує передані користувачами значення strUsername і strPassword для створення SQL-запиту. Припустимо, зловмисник передасть наступні значення параметрів:

Login: 'OR''='

Password: 'OR''='

У результаті сервера буде переданий наступний SQL-запит:

SELECT Username FROM Users WHERE Username ="OR "=" AND Password ="OR"="

Замість порівняння ім'я користувача та пароль з записами в таблиці Users, цей запит порівнює порожню рядок з пустим рядком. Природно, результат подібного запиту завжди буде дорівнювати True, і зловмисник увійде в систему від імені першого користувача в таблиці.

Зазвичай виділяють два методи експлуатації впровадження операторів SQL: звичайна атака, і атака наосліп (Blind SQL Injection). У першому випадку зловмисник підбирає параметри запиту, використовуючи інформацію про помилки, що генерується Web-додатком.

Приклад

Додаючи оператор union до запиту зловмисник перевіряє доступність бази даних:

http://example/article.asp?ID=2+union+all+select+name+from+sysobjects

Сервер генерує повідомлення, аналогічне наступного: Microsoft OLE DB Provider for ODBC Drivers error '80040e14 ' [Microsoft] [ODBC SQL Server Driver] [SQL Server] All queries in an SQL statement containing a UNION operator must have an equal number of expressions in their target lists.

З цього випливає, що оператор union був переданий серверу, і тепер зловмисникові необхідно підібрати використовуване у вихідному виразі select кількість параметрів.

Впровадження SQL коду наосліп

У цьому випадку стандартні повідомлення про помилки модифіковані, і сервер повертає зрозумілу для користувача інформацію про неправильне введення. Здійснення SQL Injection може бути здійснене і в цій ситуації, проте виявлення уразливості утруднене. Найбільш поширений метод перевірки наявності проблеми – додавання виразів, повертають істинне і помилкове значення.

Виконання подібного запиту до сервера: http://example/article.asp?ID=2+and+1=1 має повернути ту ж сторінку, що й запит: http://example/article.asp?ID=2 оскільки вираження 'and 1 = 1' завжди правдиве. Якщо в запит додається вираз, повертає значення «неправда»: http://example/article.asp?ID=2+and+1=0 користувачеві буде повернуто повідомлення про помилки або сторінка не буде згенерована. У разі якщо факт наявність уразливості підтверджений, експлуатація нічим не відрізняється від звичайного варіанта.

### 1.2.4.6. Впровадження серверних розширень (SSI Injection)

Атаки даного класу дозволяють зловмисникові передати виконуваний код, який надалі буде виконаний на Web-сервері. Уразливості, що приводять до

можливості здійснення даних атак, зазвичай полягають у відсутності перевірки даних, наданих користувачем, перед збереженням.

Перед генерацією HTML сторінки сервер може виконувати сценарії, наприклад Server-site Includes (SSI). У деяких ситуаціях вихідний код сторінок генерується на основі даних, наданих користувачем.

Якщо атакуючий передає серверу оператори SSI, він може отримати можливість виконання команд операційної системи або включити до неї заборонений вміст при наступному відображенні.

Приклад

Наступний вираз буде інтерпретовано як команда, для перегляду вміст каталогу сервера в Unix системах.

<! - # Exec cmd = "/ bin / ls /" ->

Наступний вираз дозволяє отримати рядок з'єднання з базою даних та іншу чутливу інформацію, розташовану у файлі конфігурації програми. NET.

<!--# INCLUDE VIRTUAL = "/ web.config" ->

Інші можливості для атаки виникають, коли Web-сервер використовує в URL ім'я підключаємого файлу сценаріїв, але належним чином його не веріфікує. У цьому випадку зловмисник може створити на сервері файл і підключити його до виконуваного сценарію, або вказати в якості імені сценарію URL своєму сервері.

### 1.2.4.7. Впровадження операторів XPath (XPath Injection)

Ці атаки спрямовані на Web-сервери, що створюють запити на мові XPath на основі даних, які вводяться користувачем.

Мова XPath 1.0 розроблений для надання можливості звернення до частин документа на мові XML. Він може бути використаний безпосередньо або в

якості складової частини XSLT-перетворення XML-документів або виконання запитів XQuery.

Синтаксис XPath близький до мови SQL запитів. Припустимо, що існує документ XML, що містить елементи, відповідні іменам користувачів, кожен з яких містить три елементи - ім'я, пароль та номер рахунку. Наступний вираз мовою XPath дозволяє визначити номер рахунку користувача "jsmith" з паролем "Demo1234": string (/ / user [name / text () = 'jsmith' and password / text () = 'Demo1234'] / account / text ())

Якщо запити XPath генеруються під час виконання на основі користувальницького введення, у атакуючого з'являється можливість модифікувати запит з метою обходу логіки роботи програми.

Приклад

Припустимо, що Web-додаток використовує XPath для запитів до даного документу XML для отримання номерів рахунки користувачів, чиє ім'я і пароль було передано клієнтом. Якщо це додаток впроваджує дані користувача безпосередньо до запиту, це призводить до виникнення уразливості.

### 1.2.5. Розголошення інформації (Information Disclosure)

Атаки даного класу направлені на отримання додаткової інформації про Web-додатку. Використовуючи ці уразливості, зловмисник може визначити використовувані дистрибутиви ПЗ, номери версій клієнта і сервера і встановлені оновлення. В інших випадках, в витікаючій інформації може міститися розташування тимчасових файлів або резервних копій. У багатьох випадках ці дані не потрібні для роботи користувача.

Більшість серверів надають доступ до надмірного обсягом даних, однак необхідно мінімізувати обсяг службової інформації. Чим більшими знаннями про програму буде знати зловмисник, тим легше йому буде скомпрометувати систему.

#### 1.2.5.1. Індексування директорій (Directory Indexing)

Надання списку файлів в директорії являє собою нормальну поведінку Web-сервера, якщо сторінка, яка відображається за умовчанням (index.html / home.html /default.htm) відсутня.

Коли користувач запитує основну сторінку сайту, він зазвичай вказуєте доменне ім'я сервера без імені конкретного файлу (http://www.example). Сервер переглядає основну папку, знаходить у ній файл, який використовується за умовчанням, і на його основі генерує відповідь. Якщо такий файл відсутній, як відповідь може повернутися список файлів в директорії сервера.

Ця ситуація аналогічна виконання команди "ls" (Unix) або "dir" (Windows) на сервері і форматування результатів у вигляді HTML. У цій ситуації зловмисник може отримати доступ до даних, не призначеним для вільного доступу. Досить часто адміністратори покладаються на "безпека через приховування", припускаючи, що раз гіперпосилання на документ відсутній, то він недоступний. Сучасні сканери вразливостей можуть динамічно додавати файли і папки до списку сканованих залежно від результатів запитів. Використовуючи інформацію із / robots.txt або отриманого списку директорій сканер може знайти прихований вміст або інші файли.

Таким чином, зовні безпечне індексування директорій може призвести до витоку важливої інформації, яка в подальшому буде використана для проведення атак на систему.

Приклад:

Використовуючи індексування можна дістати доступ до таких даних:

- Резервні копії (. Bak,. Old or. Orig);
- Тимчасові файли. Такі файли повинні вилучатися сервером автоматично, але іноді залишаються доступними.
- Заховані файли, назва яких починається з символу ".";

- Угода про імена. Ця інформація може допомогти передбачити імена файлів чи директорій (admin або Admin, back-up або backup).
- Список користувачів сервера. Досить часто для кожного з користувачів створюється папка з ім'ям, заснованому на назві профілю.
- Імена файлів конфігурації (. Conf,. Cfg or. Config)
- Вміст серверних сценаріїв або виконуваних файлів у разі невірно зазначених розширень чи дозволів.

Можуть використовуватися три основні сценарії отримання списку файлів:

- Помилки конфігурації. Подібні проблеми виникають, коли адміністратор помилково вказує в конфігурації сервера цю опцію. Подібні ситуації часто виникають під час налаштування складних конфігурацій, де деякі папки повинні бути доступні для перегляду. З точки зору зловмисника запит не відрізняється від зазначеного раніше. Він просто звертається до директорії і аналізує результат. Його не турбує, чому сервер веде себе подібним чином.
- Деякі компоненти Web-сервера дозволяють отримувати список файлів, навіть якщо це не дозволено у конфігураційних файлах. Зазвичай це виникає в результаті помилок реалізації, коли сервер генерує список файлів при отриманні певного запиту.
- Бази даних пошукових машин (Google, Wayback machine) можуть містити кеш старих варіантів сервера, включаючи списки файлів.

#### 1.2.5.2. Ідентифікація додатків (Web Server/Application Fingerprinting)

Визначення версій додатків використовується зловмисником для отримання інформації про використовувану сервером і клієнтом операційних системах, Web-серверах і браузерах. Також ця атака може бути спрямована на інші компоненти Web-додатків, наприклад, службу каталогу чи сервер баз даних.

Зазвичай подібні атаки здійснюються шляхом аналізу різної інформації, що надається Web-сервером, наприклад:

- Особливості реалізації протоколу HTTP;
- Заголовки HTTP-відповідей;
- Використовувані сервером розширення файлів (. Asp або. Jsp);
- Значення Cookie (ASPSESSION і т.д.);
- Повідомлення про помилки;
- Структура каталогів і використовуване угоду про імена (Windows / Unix);
- Інтерфейси підтримки розробки Web-додатків (Frontpage / WebPublisher);
- Інтерфейси адміністрування сервера (iPlanet / Comanche);
- Визначення версій операційної системи.

Для визначення версій клієнтських додатків зазвичай використовується аналіз HTTP-запитів (порядок слідування заголовків, значення User-agent і т.д.). Проте, для цих цілей можуть застосовуватись і інші техніки. Так, наприклад, аналіз заголовків листів, створених за допомогою клієнта Microsoft Outlook, дозволяє визначити версію встановленого на комп'ютері браузера Internet Explorer.

Наявність детальної та точної інформації про використовуваних додатках дуже важливо для зловмисника, оскільки реалізація багатьох атак (наприклад, переповнювання буфера) специфічна для кожного варіанту операційної системи або програми. Крім того, детальна інформація про інфраструктуру дозволяє зменшити кількість помилок, і як наслідок - загальний «шум", що виробляється атакуючим. Даний факт відзначено в HTTP RFC 2068, які рекомендують щоб значення заголовка Server HTTP відповіді був налаштованим параметром.

#### 1.2.5.3. Витік інформації (Information Leakage)

Ці вразливості виникають у ситуаціях, коли сервер публікує важливу інформацію, наприклад коментарі розробників або повідомлення про помилки, яка може бути використана для компрометації системи. Цінні з точки зору зловмисника дані можуть міститися в коментарях HTML, повідомленнях про помилки або просто бути присутнім у відкритому вигляді. Існує величезна кількість ситуацій, в яких може статися витік інформації. З витоком важливої інформації можуть виникати ризики різного ступеня, тому необхідно мінімізувати кількість службової інформації, доступної на клієнтській стороні.

Аналіз доступної інформації дозволяє зловмисникові зробити розвідку і отримати уявлення про структуру директорій сервера, що використовуються у SQL запитах, назвах ключових процесів і програм сервера. Часто розробники залишають коментарі у HTML сторінках і коді сценаріїв для полегшення пошуку помилок і підтримки програми. Ця інформація може варіюватися від простих описів деталей функціонування програми до, в гірших випадках, імен користувачів і паролів, які використовуються при налагодженні. Витік інформації може відноситися і до конфіденційних даних, оброблюваних сервером. Це можуть бути ідентифікатори користувача (ІПН, номери водійських посвідчень, паспортів і т.д.), а також поточна інформація (баланс особового рахунку або історія платежів).

#### 1.2.5.4. Зворотний шлях в директоріях (Path Traversal)

Дана техніка атак спрямована на отримання доступу до файлів, теки та командам, які знаходяться поза основною директорії Web-сервера. Зловмисник може маніпулювати параметрами URL з метою отримати доступ до файлів або виконати команди, розташовувані у файловій системі Web-сервера. Для подібних атак потенційно вразливий будь-який пристрій, що має Web-інтерфейс.

Багато Web-серверів обмежують доступ користувача певною частиною файлової системи, звичайно званої "web document root" або "CGI root". Ці директорії містять файли, призначені для користувача і програми, необхідні для отримання доступу до функцій Web-додатків.

Більшість базових атак, що експлуатують зворотний шлях, засновані на впровадженні в URL символів "../", для того, щоб змінити розташування ресурсу, який буде оброблятися сервером. Оскільки більшість Web-серверів фільтрують цю послідовність, зловмисник може скористатися альтернативними кодуваннями для представлення символів переходу по директоріях. Популярні прийоми включають використання альтернативних кодувань, наприклад Unicode ("..% u2216 "або" ..% c0% af "), використання зворотного слеша (" .. \ ") в Windows-серверах, символів URLEncode ("% 2e% 2e % 2f ") чи подвійна кодування URLEncode ("..% 255c").

Навіть якщо Web-сервер обмежує доступ до файлів певним каталогом, ця вразливість може виникати в сценаріях або CGI-програмах. Можливість використання зворотного шляху в каталогах досить часто виникає в додатках, що використовують механізми шаблонів або завантажують їх текст сторінок з файлів на сервері. У цьому варіанті атаки зловмисник модифікує ім'я файлу, що передається як параметр CGI-програми або серверного сценарію. У результаті зловмисник може отримати вихідний код сценаріїв. Досить часто до імені запитуваного файлу додаються спеціальні символи, такі як "% 00", з метою обходу фільтрів.

Приклади

Зворотний шлях в каталогах Web-сервера

http://example/../../../../../some/file
http://example/ ..% 255c ..% 255c ..% 255csome/file
http://example/ ..% u2216 ..% u2216some/file

Зворотний шлях в каталогах Web-додатки

Вихідний URL: http://example/foo.cgi?home=index.htm

Атака: http://example/foo.cgi?home=foo.cgi

У наведеному сценарії Web-додаток генерує сторінку, що містить вихідний код сценарію foo.cgi, оскільки значення змінної home використовується як ім'я завантаження. Зверніть увагу, що в даному випадку зловмисник не використовує спеціальних символів, оскільки метою є файл в тій же директорії, в якій розташовується файл index.htm.

Зворотний шлях в каталогах Web-додатки з використанням спеціальних символів:

Вихідний URL: http://example/scripts/foo.cgi?page=menu.txt

Атака: http://example/scripts/foo.cgi?page=../scripts/foo.cgi% 00txt

У наведеному прикладі Web-програма завантажує вихідний текст сценарію foo.cgi. Атакуючий використовує символи "../" для переходу на рівень вище по дереву каталогів та переходу в директорію /scripts. Символ "% 00" використовується для обходу перевірки розширення файлу (додаток дозволяє звертатися тільки до файлів. Txt) і для того, щоб розширення не використовувалося при завантаженні файлу.

### 1.2.5.5. Передбачуване розміщення ресурсів (Predictable Resource Location)

Передбачуване розміщення ресурсів дозволяє зловмисникові отримати доступ до прихованих даних або функціональним можливостям. Шляхом підбору зловмисник може отримати доступ до вмісту, не призначеному для публічного перегляду. Тимчасові файли, файли резервних копій, файли конфігурації або стандартні приклади часто є метою подібних атак. У більшості випадків перебір може бути оптимізовано шляхом використання стандартної

угоди про імена файлів і директорій сервера. Отримувані зловмисником файли можуть містити інформацію про дизайн програми, інформацію з баз даних, імена машин або паролі, шляхи до директорій. Також «приховані» файли можуть містити уразливості, відсутні в основному додатку. На цю атаку часто посилаються як на перерахування файлів і директорій (Forced Browsing, File Enumeration, Directory Enumeration).

Приклад

Атакуючий може створити запит до будь-якого файлу або папці на сервері. Наявність або відсутність ресурсу визначається за кодом помилки (наприклад, 404 у разі відсутності папки або 403 в разі її наявності на сервері). Нижче наведені варіанти подібних запитів.

Сліпий пошук популярних назв директорій:

/admin/
/backup/
/logs/
/vulnerable_file.cgi
Зміна розширень існуючого файлу: (/test.asp)
/test.asp.bak
/test.bak
/test

### 1.2.6. Логічні атаки (Logical Attacks)

Атаки даного класу спрямовані на експлуатацію функцій програми або логіки його функціонування. Логіка програми представляє собою очікуваний процес функціонування програми при виконанні певних дій. В якості прикладів можна навести відновлення паролей, реєстрацію облікових записів, аукціонні торги, транзакції в системах електронної комерції. Додаток може вимагати від користувача коректного виконання кількох послідовних дій для виконання

певного завдання. Зловмисник може обійти або використовувати ці механізми в своїх цілях.

#### 1.2.6.1. Зловживання функціональними можливостями (Abuse of Functionality)

Дані атаки спрямовані на використання функцій Web-додатки з метою обходу механізмів розмежування доступу. Деякі механізми Web-додатки, включаючи функції забезпечення безпеки, можуть бути використані для цих цілей. Наявність уразливості в одному з, можливо, другорядних компонентів системи може призвести до компрометації всього додатку. Рівень ризику та потенційні можливості зловмисника в разі проведення атаки дуже сильно залежать від конкретного додатка.

Зловживання функціональними можливостями дуже часто використовується спільно з іншими атаками, такими як зворотний шлях в директоріях і т.д. Приміром, за наявності уразливості типу міжсайтового виконання сценаріїв в HTML-чаті зловмисник може використовувати функції чату для розсилки URL, який експлуатує уразливість, всім поточним користувачам.

З глобальної точки зору, всі атаки на комп'ютерні системи є зловживаннями функціональними можливостями. Особливо це стосується до атак, спрямованим на Web-додатки, які не вимагають модифікації функцій програми.

Приклади зловживання функціональними можливостями включають в себе:

- Використання функцій пошуку для отримання доступу до файлів за межами кореневої директорії Web-сервера;
- Використання функції завантаження файлів на сервер для перезапису файлів конфігурації або впровадження серверних сценаріїв;

- Реалізація відмови в обслуговуванні шляхом використання функції блокування облікового запису при багаторазовому введенні неправильного пароля.

#### 1.2.6.2. Відмова в обслуговуванні (Denial of Service)

Даний клас атак спрямований на порушення доступності Web-сервера. Зазвичай атаки, спрямовані на відмову в обслуговуванні реалізуються на мережевому рівні, проте вони можуть бути спрямовані і на прикладний рівень. Використовуючи функції Web-додатки, зловмисник може вичерпати критичні ресурси системи, або скористатися уразливістю, що приводить до припинення функціонування системи.

Зазвичай DoS атаки спрямовані на вичерпання критичних системних ресурсів, таких як обчислювальну потужністі, оперативна пам'ять, дисковий простір або пропускна спроможність каналів зв'язку. Якщо якийсь із ресурсів досягне максимального завантаження, додаток цілком буде недоступний. Атаки можуть бути спрямовані на будь-який з компонентів Web-додатків, наприклад, такі як сервер СУБД, сервер аутентифікації і т.д. На відміну від атак на мережевому рівні, що вимагають значних ресурсів зловмисника, атаки на прикладному рівні зазвичай легше реалізувати.

Приклади

Припустимо, що сервер Health Care-генерує звіти про клінічну історії користувачів. Для генерації кожного звіту сервер запитує всі записи, що відповідають певним номером соціального страхування. Оскільки в базі містяться сотні мільйонів записів, користувачеві доводиться чекати результату декілька хвилин. У цей час завантаження процесора сервера СУБД досягає 60%. Зловмисник може послати десять одночасних запитів на отримання звітів, що з високою ймовірністю призведе до відмови в обслуговуванні, оскільки завантаження процесора сервера баз даних досягне максимального значення.

На час обробки запитів зловмисника нормальна робота сервера буде неможлива.

DoS на інший сервер

Зловмисник може розмістити на популярному Web-форумі посилання (наприклад, у вигляді зображення в повідомленні) на інший ресурс. При заході на форум, користувачі будуть автоматично завантажувати дані з атакуємого серверу вказаний ресурс, використовуючи його ресурси. Якщо на атакуючому сервері використовується система запобігання атак з функцією блокування IP-адреси атакуючого, у посиланні може використовуватися сигнатура атаки (наприклад ../../../../../etc/passwd), що призведе до блокування користувачів, що зайшли на форум.

Атаки на сервер СУБД

Зловмисник може скористатися впровадженням коду SQL для видалення даних з таблиць, що призведе до відмови в обслуговуванні програми.

### 1.2.6.3. Недостатня протидія автоматизації (Insufficient Anti-automation)

Недостатня протидія автоматизації виникає, коли сервер дозволяє автоматично виконувати операції, які повинні проводитися вручну. Для деяких функцій програми необхідно реалізовувати захист від автоматичних атак.

Автоматизовані програми можуть варіюватися від нешкідливих роботів пошукових систем до систем автоматизованого пошуку вразливостей і реєстрації облікових записів. Подібні роботи генерують тисячі запитів в хвилину, що може призвести до падіння продуктивності всього додатку.

Протидія автоматизації полягає в обмеженні можливостей подібних утиліт. Наприклад, файл robots може запобігати індексування деяких частин сервера, а додаткові затрати ідентифікації запобігати автоматичну реєстрацію сотень облікових записів системи електронної пошти.

### 1.2.6.4. Недостатня перевірка процесу (Insufficient Process Validation)

Уразливості цього класу виникають, коли сервер не достатньо перевіряє послідовність виконання операцій програми. Якщо стан сесії користувачів та програми належним чином не контролюється, додаток може бути уразливе для шахрайських дій.

У процесі доступу до деяких функцій програми очікується, що користувач виконає ряд дій в певному порядку. Якщо деякі дії виконуються невірно або у неправильному порядку, виникає помилка, що приводить до порушення цілісності. Прикладами подібних функцій виступають переклади, відновлення паролів, підтвердження покупки, створення облікового запису і т.д. У більшості випадків ці процеси складаються з ряду послідовних дій, здійснюваних у чіткому порядку.

Для забезпечення коректної роботи подібних функцій Web-додаток повинен чітко відслідковувати стан сесії користувачів та відслідковувати її відповідність поточним операціям. У більшості випадків це здійснюється шляхом збереження стану сесії в cookie або прихованому полі форми HTML. Але оскільки ці значення можуть бути модифіковані користувачем, обов'язково має проводитися перевірка цих значень на сервері. Якщо цього не відбувається, зловмисник отримує можливість обійти послідовність дій, і як наслідок - логіку програми.

Приклад

Система електронної торгівлі може пропонувати знижку на продукт В, у випадку купівлі продукту А. Користувач, який не хоче купувати продукт А, може спробувати придбати продукт В зі знижкою. Заповнивши замовлення на купівлю обох продуктів, користувач отримає знижку. Потім користувач повертається до форми підтвердження замовлення і видаляє продукт А, шляхом модифікації значень у формі. Якщо сервер повторно не перевірить можливість

покупки продукту B за вказаною ціною без продукту A, буде здійснено закупівлю за низькою ціною.

## 1.3. Ієрархія захисту Web-серверів

Узагальнимо основні причини уразливості веб-серверів.

Більшість зростаючих підприємств регулярно міняє конфігурацію своїх мереж, додаючи нові робочі станції (іноді й сервера), забуваючи при цьому тестувати ЛВС на безпеку.

Більшість веб-майстрів мають кореневой або адміністраторський доступ до сервера. Розумніше прописати кожному користувачеві свою політику доступу, що обмежує його права прямими обов'язками. Наприклад, співробітник працює тільки з одним каталогом сервера, але має доступ на всі інші. Тим самим він ставить під загрозу не лише свій сектор робіт, але і всі дані сервера.

Програмне забезпечення веб-серверів. Суміш з піратських, ліцензійних, shareware-ifreeware-програм робить систему вразливою. Найбезпечніший підхід - сумісне програмне забезпечення від одного виробника.

Як бачите, безпека веб-серверів зводиться до управління ризиками. Але не кожна компанія має потребу у вищій ступені захисту своєї інформації. Питання рівня безпеки - це питання використання ресурсів мережі.

Тому прийнято розділяти рівні захисту веб-серверів.

### 1.3.1. Рівні захисту Web-серверів

Захист мережі можна розділити на шість рівнів складності. Перший – найбільш елементарний, і обов'язковий. Тут головний інструмент захисту - firewall. Firewall повинен лімітувати використання сервісів, що надаються користувачам. Також firewall повинен стежити за всіма з'єднаннями, як з одного, так і з іншого боку.

Другий рівень безпеки на увазі конфігурацію операційної системи, під чиїм керуванням працює веб-сервер. Кожна операційна система дозволяє створювати контрольні листи безпеки (security checklist). Ці установки повинні бути узгоджені з операційними системами вендорів, які співпрацюють з компанією.

Третій рівень орієнтований вже на мережу. Необхідно оснастити датчиками атаки мережного обладнання та програмного забезпечення провайдера, що забезпечує хостинг. Головне, щоб надійшов сигнал про небезпеку був правильно оброблений і нейтралізований.

Четвертий рівень безпеки - установка програмного забезпечення на рівні хостингу. Це значно складніше завдання. По-перше, тут можна зіткнутися з запереченнями самої хостинг-компанії. А по-друге, таке програмне забезпечення набагато складніше простих датчиків. З цієї причини й рівень безпеки більш високий.

Рівень 5 має два підрівня - А і Б. Рівень 5А - це встановлення спеціального програмного забезпечення, що виконує роль прошарку між операційною системою веб-сервера і усіма додатками. Такий буфер дозволяє запобігти атаці хакерів на уразливі програми, які беруть під контроль усю операційну систему під час свого виконання. Рівень 5Б є установкою орієнтованих на конкретні програми firewall або проксі-серверів. Вони орієнтовані на HTTP-протокол і дозволяють запобігти атакам перш, ніж потенційні зловмисники зуміють дістатися до запуску додатків, встановлених на веб-сервері.

Шостий рівень - своєрідна вершина безпеки. Тут допускається використання тільки довірчих операційних систем і працюють під їхнім управлінням додатків. Іншими словами, всі функціонуючі програми та операційні системи повинні бути або максимально адаптовані, або розроблені спеціально для специфічних потреб компанії. Це найдорожчий, але і найефективніший спосіб захисту.

## 1.4. Загальна інформація про засоби пошуку вразливостей

Такі засоби можуть функціонувати на мережевому рівні (network-based), рівні операційної системи (host-based) і рівні додатку (application-based). Найбільшого поширення набули засоби аналізу захищеності мережевих сервісів і протоколів. Пов'язано це, в першу чергу, з універсальністю використовуваних протоколів. Вивченість і повсюдне використання таких протоколів, як IP, TCP, HTTP, FTP, SMTP і т.п. дозволяють з високим ступенем ефективності перевіряти захищеність інформаційної системи, що працює в даному мережевому оточенні. Другими за поширеністю є засоби аналізу захищеності операційних систем (ОС). Пов'язано це також з універсальністю і поширеністю деяких операційних систем (наприклад, UNIX і Windows NT). Однак через те, що кожен виробник вносить в операційну систему свої зміни (яскравим прикладом є безліч різновидів ОС UNIX), засоби аналізу захищеності ОС аналізують в першу чергу параметри, характерні для всього сімейства однієї ОС. І лише для деяких систем аналізуються специфічні для неї параметри. Засобів аналізу захищеності додатків на сьогоднішній день не так багато, як цього хотілося б. Такі засоби існують тільки для широко поширених прикладних систем, типу Web-браузери, СУБД і т.п.

Крім виявлення вразливостей, за допомогою засобів аналізу захищеності можна швидко визначити всі вузли корпоративної мережі, доступні в момент проведення тестування, виявити всі використовувані в ній сервіси та протоколи, їх налаштування і можливості для несанкціонованого впливу (як зсередини корпоративної мережі, так і зовні). Також ці засоби виробляють рекомендації і покрокові заходи, що дозволяють усунути виявлені недоліки.

Оскільки найбільшого поширення набули засоби, що функціонують на рівні мережі то основна увага буде приділена саме їм.

### 1.4.1. Механізми роботи

Існує два основних механізми, за допомогою яких сканер перевіряє наявність уразливості - сканування (scan) і зондування (probe).

Сканування - механізм пасивного аналізу, за допомогою якого сканер намагається визначити наявність уразливості без фактичного підтвердження її наявності - за непрямими ознаками. Цей метод є найбільш швидким і простим для реалізації. У термінах компанії ISS даний метод отримав назву "логічний висновок" (inference). Згідно компанії Cisco цей процес ідентифікує відкриті порти, знайдені на кожному мережевому пристрої, і збирає пов'язані з портами заголовки (banner), знайдені при скануванні кожного порту. Кожен отриманий заголовок порівнюється з таблицею правил визначення мережевих пристроїв, операційних систем і потенційних вразливостей. На основі проведеного порівняння робиться висновок про наявність чи відсутність уразливості.

Зондування - механізм активного аналізу, який дозволяє переконатися, присутня чи ні на уже згадуваному вузлі вразливість. Зондування виконується шляхом імітації атаки, що використовує вразливість яка перевіряється. Цей метод більш повільний, ніж "сканування", але майже завжди набагато більш точний. У термінах компанії ISS даний метод отримав назву "підтвердження" (verification). Згідно компанії Cisco цей процес використовує інформацію, отриману в процесі сканування ( "логічного висновку"), для детального аналізу кожного мережевого пристрою. Цей процес також використовує відомі методи реалізації атак для того, щоб повністю підтвердити передбачувані уразливості і виявити інші уразливості, які не можуть бути виявлені пасивними методами, наприклад схильність атакам типу "відмова в обслуговуванні" ( "denial of service").

На практиці зазначені механізми реалізуються наступними кількома методами.

#### 1.4.1.1."Перевірка заголовків" (banner check)

Зазначений механізм являє собою ряд перевірок типу "сканування" і дозволяє робити висновок про уразливість, спираючись на інформацію в заголовку відповіді на запит сканера. Типовий приклад такої перевірки - аналіз заголовків програми Sendmail або FTP-сервера, що дозволяє дізнатися їхню версію і на основі цієї інформації зробити висновок про наявність в них уразливості.

Найбільш швидкий і простий для реалізації метод перевірки присутності на сканованому вузлі уразливості. Однак за цією простотою ховається чимало проблем.

Ефективність перевірок заголовків досить ефемерна. І ось чому. По-перше, ви можете змінити текст заголовка, завбачливо видаливши з нього номер версії або іншу інформацію, на підставі якої сканер будує свої висновки. І хоча такі випадки надзвичайно рідкісні, нехтувати ними не варто. Особливо в тому випадку, якщо у вас працюють фахівці в області безпеки, які розуміють всю небезпеку заголовків "за замовчуванням". По-друге, найчастіше, версія, що вказується в заголовку відповіді на запит, не завжди говорить про уразливість програмного забезпечення. Особливо це стосується програмного забезпечення, яке розповсюджується разом з вихідними текстами (наприклад, в рамках проекту GNU). Ви можете самостійно усунути уразливість шляхом модифікації вихідного тексту, при цьому забувши змінити номер версії в заголовку. І по-третє, усунення вразливості в одній версії ще не означає, що в наступних версіях ця вразливість відсутня.

Процес, описаний вище, є першим і дуже важливим кроком при скануванні мережі. Він не призводить до порушення функціонування сервісів або вузлів мережі. Однак не варто забувати, що адміністратор може змінити текст заголовків, які повертаються на зовнішні запити.

#### 1.4.1.2. "Активні зондуючі перевірки" (active probing check)

Також відносяться до механізму "сканування". Однак вони засновані не на перевірках версій програмного забезпечення в заголовках, а на порівнянні "цифрового зліпка" (fingerprint) фрагмента програмного забезпечення зі зліпком відомої вразливості. Аналогічним чином роблять антивірусні системи, порівнюючи фрагменти програмного забезпечення, що сканується, з сигнатурами вірусів, що зберігаються в спеціалізованій базі даних. Різновидом цього методу є перевірки контрольних сум або дати програмного забезпечення, що сканується, які реалізуються в сканерах, які працюють на рівні операційної системи.

Спеціалізована база даних містить інформацію про уразливості й способи їх використання (атаки). Ці дані доповнюються відомостями про заходи їх усунення, що дозволяють знизити ризик безпеки в разі їх виявлення. Найчастіше ця база даних використовується і системою аналізу захищеності і системою виявлення атак.

Цей метод також досить швидкий, але реалізується важче, ніж "перевірка заголовків".

#### 1.4.1.3. "Імітація атак" (exploit check)

Дані перевірки відносяться до механізму "зондування" і засновані на експлуатації різних дефектів в програмному забезпеченні.

Деякі уразливості не виявляють себе, поки ви не "підштовхнете" їх. Для цього проти підозрілого сервісу або вузла запускаються реальні атаки. Перевірки заголовків здійснюють первинний огляд мережі, а метод "exploit check", відкидаючи інформацію в заголовках, дозволяє імітувати реальні атаки, тим самим з більшою ефективністю (але меншою швидкістю) виявляючи уразливості на сканованих вузлах. Імітація атак є більш надійним способом

аналізу захищеності, ніж перевірки заголовків, і зазвичай більш надійна, ніж активні зондуючі перевірки.

Однак існують випадки, коли імітація атак не завжди може бути реалізована. Такі випадки можна розділити на дві категорії: ситуації, в яких тест призводить до "відмови в обслуговуванні" аналізованого вузла або мережі, і ситуації, при яких уразливість в принципі не придатна для реалізації атаки на мережу.

Багато проблем захисту не можуть бути виявлені без блокування або порушення функціонування сервісу або комп'ютера в процесі сканування. У деяких випадках небажано використовувати імітацію атак (наприклад, для аналізу захищеності важливих серверів), тому що це може привести до великих витрат (матеріальних і часових) на відновлення працездатності виведених з ладу елементів корпоративної мережі. У цих випадках бажано застосувати інші перевірки, наприклад, активне зондування або, в крайньому випадку, перевірки заголовків.

Однак, є деякі вразливості (наприклад, перевірка підтвердженості до атак типу "Packet Storm"), які просто не можуть бути протестовані без можливого виведення з ладу сервісу або комп'ютера. В цьому випадку розробники надходять у такий спосіб, - за замовчуванням такі перевірки вимкнені і користувач може сам включити їх, якщо бажає.

### 1.4.2. Етапи сканування

Практично будь-який сканер проводить аналіз захищеності в кілька етапів:

1. Збір інформації про мережу. На даному етапі ідентифікуються всі активні пристрої в мережі і визначаються запущені на них сервіси та демони. У разі використання систем аналізу захищеності на рівні операційної системи даний етап пропускається, оскільки на кожному аналізованому вузлі встановлені відповідні агенти системного сканера.

2. Виявлення потенційних вразливостей. Сканер використовує описану вище базу даних для порівняння зібраних даних з відомими вразливостями за допомогою перевірки заголовків або активних зондуючих перевірок. У деяких системах всі вразливості ранжуються за ступенем ризику. Уразливі місця (наприклад, впливаючі на маршрутизатори) вважаються серйознішими в порівнянні з уразливостями, характерними тільки для робочих станцій.
3. Підтвердження обраних вразливостей. Сканер використовує спеціальні методи і моделює (імітує) певні атаки для підтвердження факту наявності вразливостей на обраних вузлах мережі.
4. Генерація звітів. На основі зібраної інформації система аналізу захищеності створює звіти, що описують виявлені вразливості. У деяких системах звіти створюються для різних категорій користувачів, починаючи від адміністраторів мережі і закінчуючи керівництвом компанії. Якщо перших цікавлять технічні деталі, то для керівництва компанії необхідно подати, красиво оформлені, із застосуванням графіків і діаграм, звіти з мінімумом подробиць. Важливим аспектом є наявність рекомендацій щодо усунення виявлених проблем. У багатьох випадках звіти також містять посилання на FTP- або Web-сервера, що містять patch'і і hotfix'и, що усувають виявлені вразливості.
5. Автоматичне усунення вразливостей. Цей етап дуже рідко реалізується в мережевих сканерах, але широко застосовується в системних сканерах.

У будь-якому випадку у адміністратора, який здійснює пошук вразливостей, є кілька варіантів використання системи аналізу захищеності:

Запуск сканування тільки з перевірками на потенціальні вразливості (етапи 1,2 і 4). Це дає попереднє ознайомлення з системами в мережі. Цей метод є набагато менш руйнівним в порівнянні з іншими і також є найшвидшим.

- Запуск сканування з перевірками на потенційні і підтверджені уразливості. Цей метод може викликати порушення роботи вузлів мережі під час реалізації перевірок типу "exploit check".
- Запуск сканування з вашими призначеними для користувача правилами для знаходження конкретної проблеми.
- Все з вищезгаданого.

Розглянемо можливості та роботу деяких відомих сканерів.

## 1.5. XSpider 7 full version

XSpider - платна (з версії 7, раніше була безкоштовною) пропріетарна програма-сканер вразливостей. Випускається російською фірмою Positive Technologies.

XSpider сильно відрізняється від примітивного сканера мережевої безпеки, який просто виконує сканування і показує результати.

Багатовіконний інтерфейс XSpider і вбудовані засоби автоматизації орієнтовані на те, щоб організувати логічний і структурований процес моніторингу безпеки, що вимагає мінімального втручання в процес роботи програми.

Центральної концепцією XSpider є "Завдання". Вона включає в себе перш за все набір перевірених хостів. В Завдання має сенс об'єднувати хости, які слід перевіряти подібним чином (ніщо не заважає мати в Завданні і всього один хост). Як тільки Завдання сформоване, йому можна присвоїти Профіль - набір налаштувань, які визначають нюанси сканування. Якщо ви цього не зробите то буде використовуватися профіль за замовчуванням. Виконання Завдання можна автоматизувати, тобто присвоїти йому розклад, за яким воно буде виконуватися без вашого втручання. Крім того, для кожного Завдання зберігається повна історія всіх сканувань. Результати будь-якого з них ви можете завантажити і

працювати з ними, як зі "свіжими". Це зручно і для аналізу розвитку ситуації, і для того, щоб випадково не втратити якісь результати роботи.

Коли створюється робоче вікно XSpider - це вікно поточного Завдання. Створення нового вікна створює нове Завдання зі стандартним ім'ям, яке при збереженні Завдання краще замінити на щось більш виразне. Завдання, як файли, можна відкривати, зберігати і т.п. Кожному Завданню відповідає файл на диску, що знаходиться за замовчуванням в стандартному каталозі XSpider (Tasks).

Одночасно Xspider може обробляти багато Завдань, кожна з яких може містити багато хостів. Єдине, про що вам при цьому варто турбуватися - пропускна здатність каналу, який зв'язує XSpider з комп'ютерами, що перевіряються. Швидкість і надійність перевірки може сильно падати, якщо канал перевантажений. З огляду на те, що трафік, створюваний XSpider на один хост, невеликий, то перевантаження каналу можливе або при дуже великому (сотні) числі одночасно сканованих хостів, або, якщо канал дуже вузький. Регулювати максимальне число хостів, що перевіряються, на одне Завдання можна через настройки. Тобто, навіть якщо в Завданні, скажімо, 100 хостів, ви можете вказати, що одночасно повинні скануватися 50. При цьому інші будуть стояти в черзі і перевірятися послідовно.

За результатом кожного сканування XSpider може згенерувати звіт, причому в автоматичному режимі вони можуть доставлятися вам по email (або викладатися на доступний мережевий диск). Якщо один раз витратити час на налаштування режиму автоматичної роботи, то потім досить буде тільки регулярно перевіряти свою поштову скриньку, щоб відстежувати безпеку всієї мережі (або мереж).

XSpider навіть в ранніх, некомерційних версіях відрізнявся високою якістю пошуку та ідентифікації вразливостей. Тут досить складно вдаватися в технічні подробиці, які пояснюють, як XSpider домагається максимально точної роботи,

тим більше, що багато механізмів є конфіденційною інформацією. Так чи інакше, всі реальні уразливості знаходяться в переважній більшості випадків, а помилкові спрацьовування буваю вкрай рідко.

Хотілося б згадати тільки евристичні алгоритми, що використовуються XSpider. Він не тільки займається простим перебором вразливостей з бази, а й виконує додатковий аналіз по ходу роботи, виходячи з особливостей поточної ситуації. Завдяки цьому, Xspider може іноді виявити специфічну вразливість, інформація про яку ще не була опублікована.

База вразливостей постійно поповнюється фахівцями Positive Technologies, що в сумі з автоматичним оновленням баз і модулів програми постійно підтримує актуальність версії XSpider.

Також XSpider виводить до звіту про результати перевірки не тільки інформацію про знайдену уразливість, а й посилання, наприклад, на статті на сайті Microsoft, які описують виявлену XSpider вразливість і дають рекомендації по її усуненню.

Переваги даного програмного продукту:

- Контроль змін на сканованих вузлах дозволяє отримати повну картину захищеності в динаміці.
- Повна ідентифікація сервісів на випадкових портах для виявлення вразливостей серверів з нестандартною конфігурацією.
- Евристичний метод визначення типів і імен серверів (HTTP, FTP, SMTP, POP3, DNS, SSH і ін.) для визначення справжнього імені сервера і коректної роботи перевірок.
- Обробка RPC-сервісів (Windows і * nix) з повною ідентифікацією, включаючи визначення детальної конфігурації комп'ютера.
- Перевірка слабкості парольного захисту: оптимізований підбір паролів практично у всіх сервісах, що вимагають аутентифікації.

- Глибокий аналіз контенту веб-сайтів, включаючи виявлення вразливостей в скриптах: SQLi, XSS, запуск довільних програм і ін.
- Аналіз структури HTTP-серверів для пошук слабких місць в конфігурації.
- Розширена перевірка вузлів під управлінням Windows.
- Проведення перевірок на нестандартні DoS-атаки.

### 1.5.1. Робота зі сканером

Після встановлення і запуску XSpider на екран буде виведено головне вікно програми (рис. 1.1).

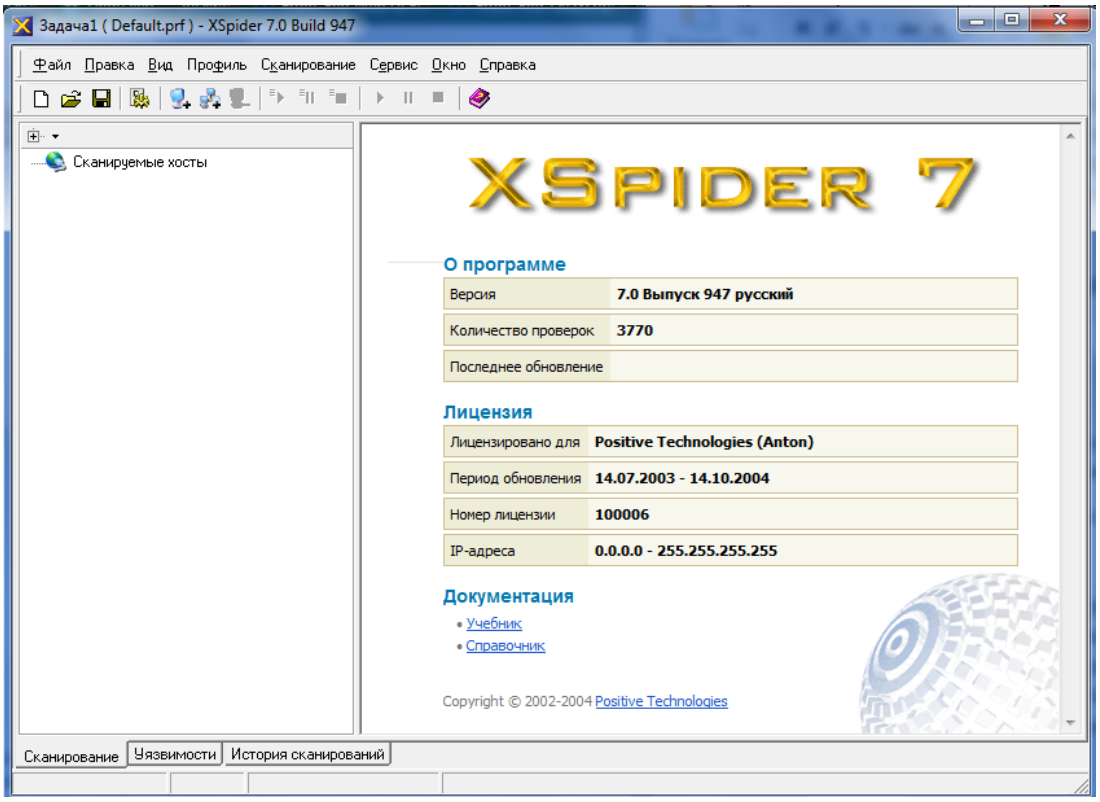



Рисунок 1.1 – Головне вікно програми

У список сканованих хостів потрібно додати адреси для сканування. Можна додати відразу діапазон адрес. Після того, як список хостів сформований, потрібно вибрати профіль перевірки або з існуючих, або створити новий профіль виходячи з власних уподобань. Існуючі профілі (рис. 1.2).

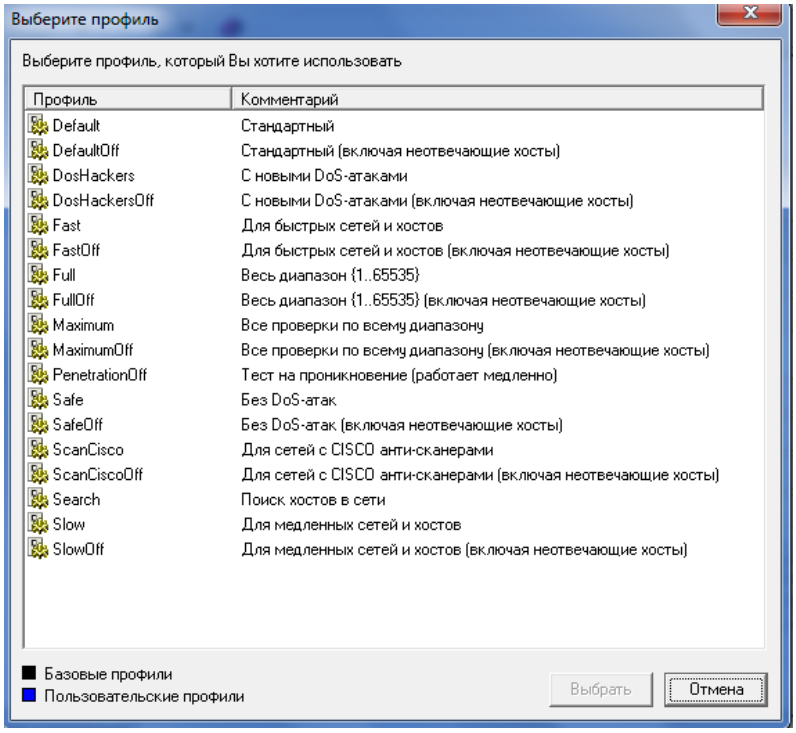



Рисунок 1.2 – Профілі сканувань

Будь-який з існуючих профілів можна налаштувати або створити новий профіль (рис. 1.3).

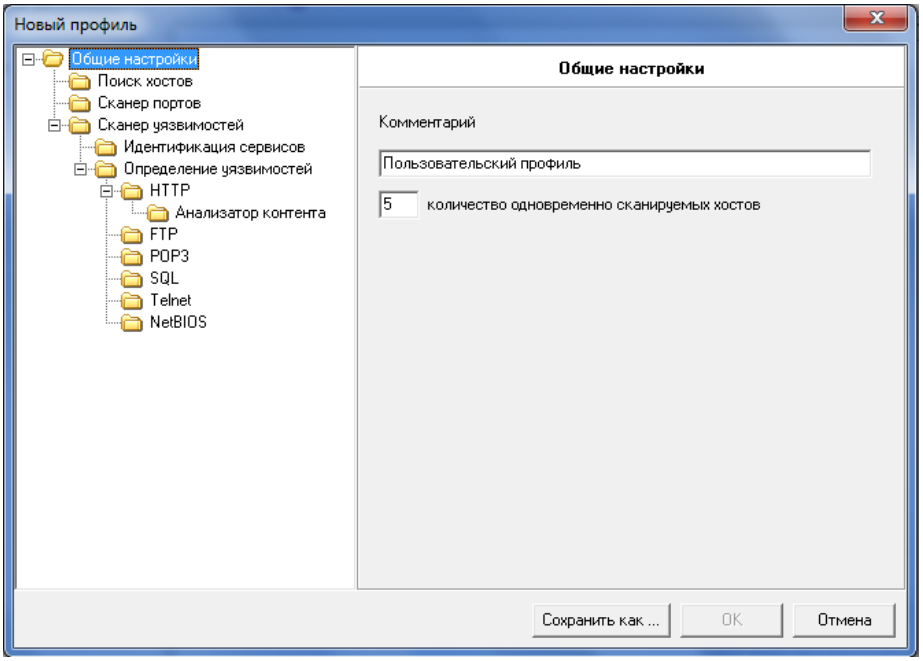



Рисунок 1.3 – Налаштування власного профілю

Після того, як завдання налаштоване, його можна зберегти, щоб надалі не доводилося налаштовувати все заново.

Запуск створених і збережених завдань можна запланувати, налаштувавши Планувальник (рис. 1.4).

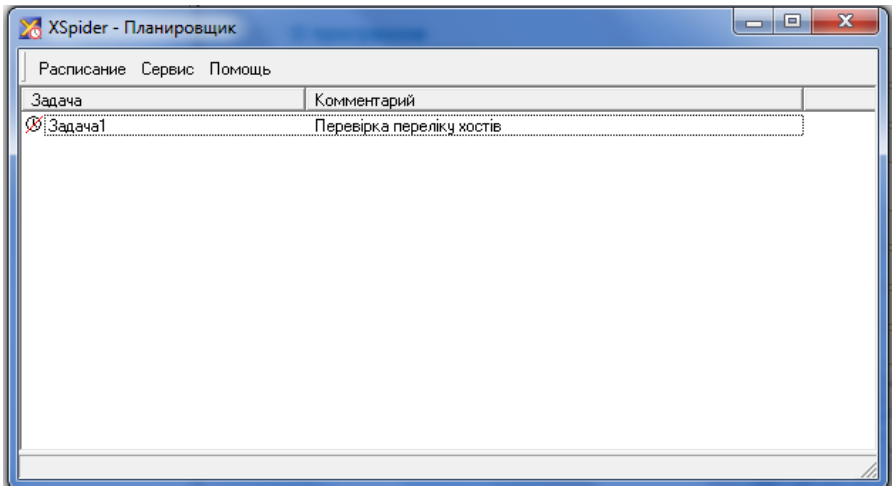



Рисунок 1.4 – Загальний вид Планувальника

Планувальник XSpider налаштовується досить просто, по аналогії з планувальником Windows. Уваги заслуговує останній етап налаштування завдання (рис. 1.5).

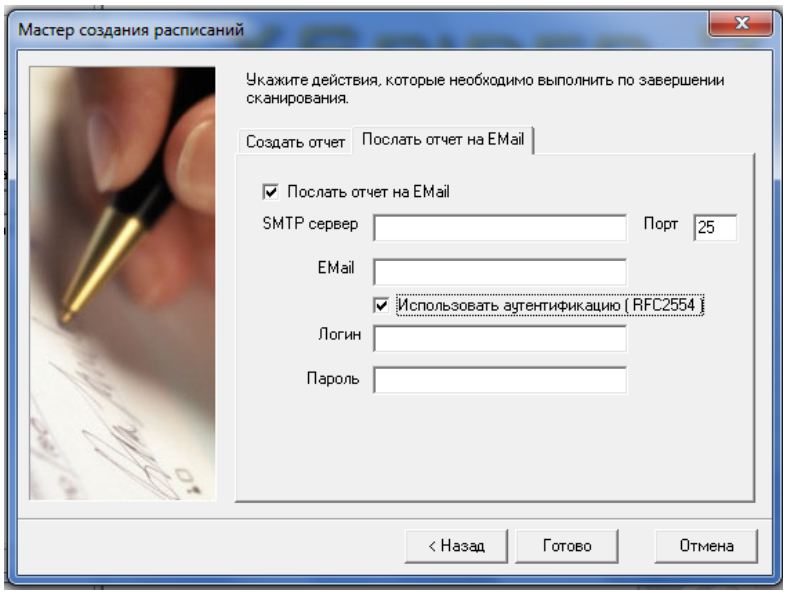



Рисунок 1.5 – Останній етап налаштування завдання

На цій вкладці майстра створення нового завдання планувальника XSpider можна налаштувати відправку звіту на поштову адресу або зберегти його в певній папці.

Кожне налаштоване завдання зберігається в файлі. Якщо запланований запуск завдання по планувальнику, то результат його виконання буде збережений у файлі .tsk, який може бути відкритий в будь-який час за допомогою XSpider. У файлі зберігається результат не тільки останньої перевірки хоста або хостів, а вся історія перевірок. Таким чином, можна контролювати зміни рівня безпеки після ліквідації виявлених раніше вразливостей і оновлення операційних систем та сервісів.

Приклад завантаженого завдання (рис. 1.6).

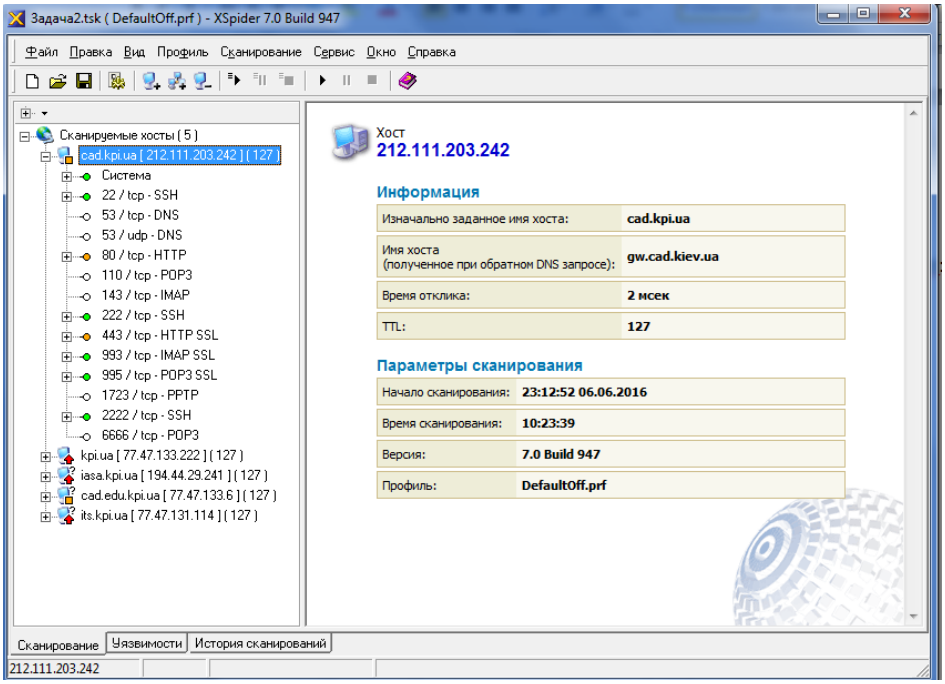



Рисунок 1.6 – Приклад завантаженого завдання

У результатах перевірки, крім інформації про виявлені вразливості, були наведені посилання на опис уразливості на сайтах, що спеціалізуються на

безпеці і надані посилання на завантаження оновлених версій програмного забезпечення (рис. 1.7).

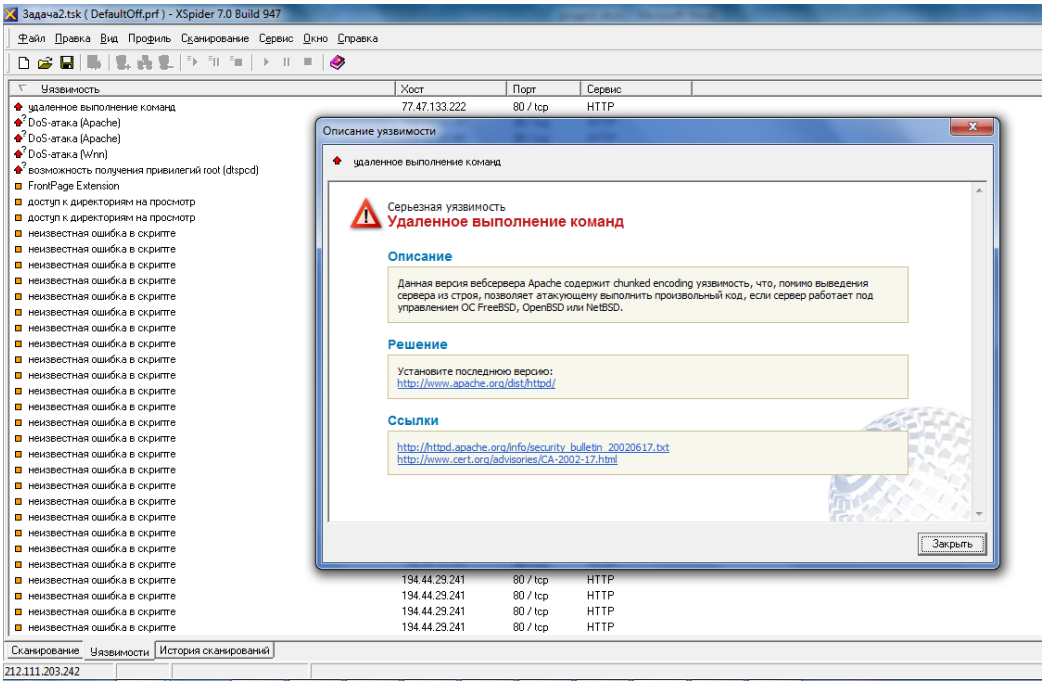



Рисунок 1.7 – Опис знайденої вразливості

XSpider пропонує на вибір кілька стандартних типів звітів про результати перевірки (рис. 1.8).

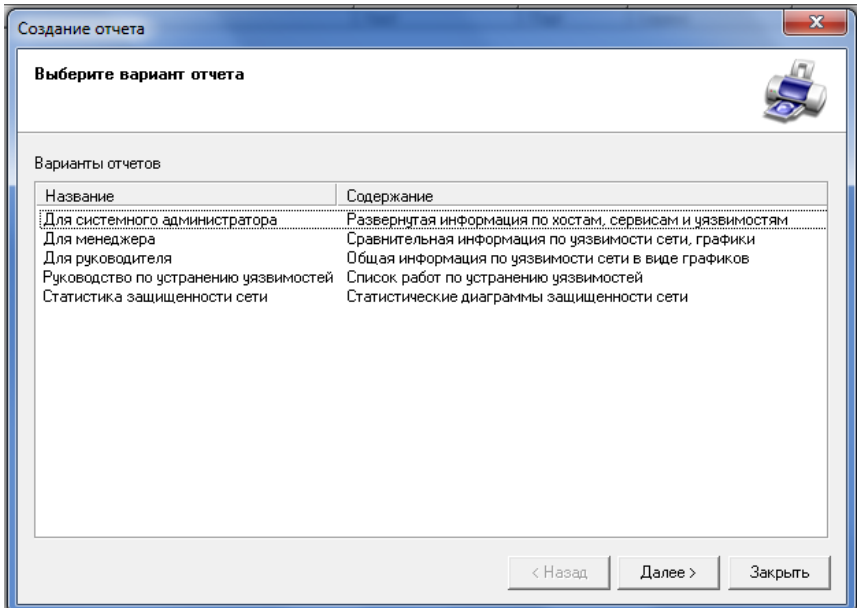



Рисунок 1.8 – Типи звітів

### 1.5.2. Робота з XSpider 7.7 Demo version

Для порівняння розглянемо можливості більш новішої версії програми, яка доступна лише в демонстраційному режимі (Версія 7.7 Випуск 3100. Кількість перевірок 7237).

Обмеження демонстраційної версії:

- Відсутня система щоденного онлайнового поновлення;
- Відсутні потенційно небезпечні перевірки на DoS-уразливості;
- Перевірки вмісту веб серверів на предмет SQL ін'єкцій, ін'єкцій коду, отримання файлів і т.д. не містять деталі;
- Відсутня цілий ряд перевірок, які використовують оригінальні евристичні механізми;
- Відсутні перевірки, пов'язані з використанням різних словників;
- Планувальник завдань має повноцінний інтефейс, але не зберігає створені розклади;
- Звіти, що генеруються містять тільки резюме по реальному скануванню, в тілі звіту - стандартний демонстраційний фрагмент;
- Історія сканувань доступна тільки для загального перегляду, старі результати не можна використовувати для подальшої роботи.

В версії 7.7 більше видів звітів, а також є можливість змінювати формат звіту (для демоверсії результати в звіті не відображаються). Типи звітів демоверсії (рис. 1.9).

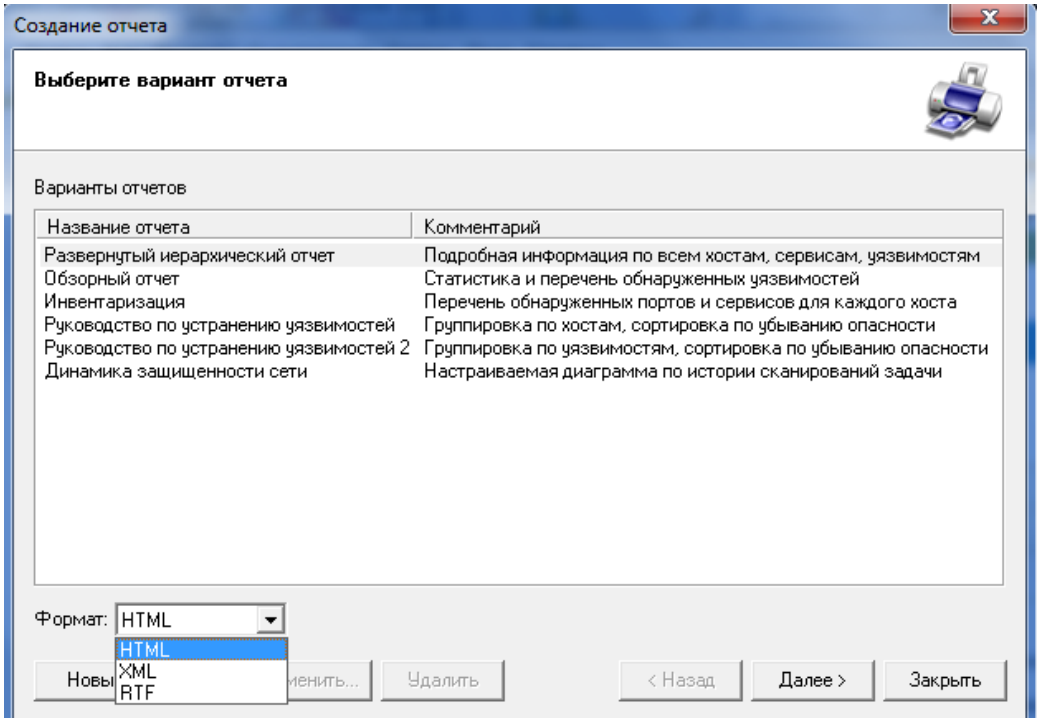



Рисунок 1.9 – Типи звітів демоверсії

Також є можливість створювати власний шаблон для звіту (рис. 1.10).

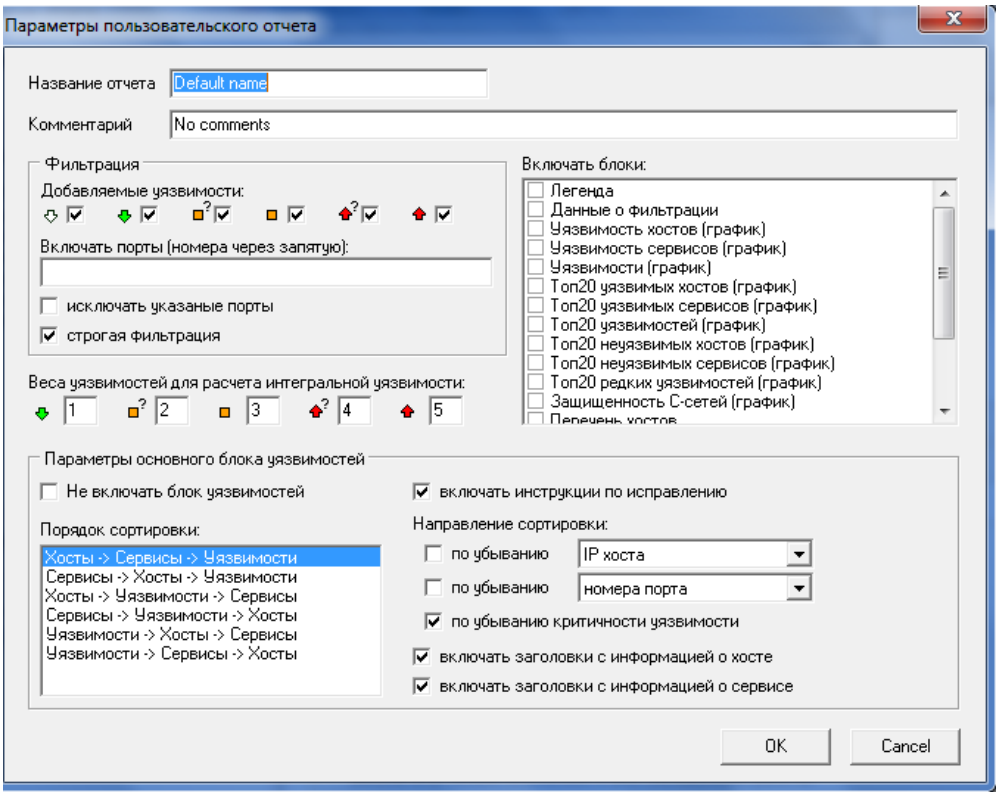



Рисунок 1.10 – Створення власного шаблону для звіту

## 1.6. Acunetix Web Vulnerability Scanner 10.0 (Consultant edition)

Acunetix Web Vulnerability Scanner автоматизує задачу контролю безпеки Web додатків і дозволяє виявити вразливі місця в захисті web-сайту до того, як їх виявить і використає зловмисник.

Як працює Acunetix Web Vulnerability Scanner:

- Acunetix WVS досліджує і формує структуру сайту, обробляючи всі знайдені посилання і збираючи інформацію про всі виявлені файли;
- Потім програма тестує всі web-сторінки з елементами для введення даних, моделюючи введення даних з використанням усіх можливих комбінацій і аналізуючи отримані результати;
- Виявивши вразливість, Acunetix WVS видає відповідне попередження, яке містить опис уразливості і рекомендації по її усуненню;
- Підсумковий звіт WVS може бути записаний в файл чи базу даних для подальшого аналізу і порівняння з результатами попередніх перевірок.

Які уразливості виявляє Acunetix Web Vulnerability Scanner:

- Cross site scripting (виконання шкідливого сценарію в браузері користувача при зверненні та у контексті безпеки довіреного сайту);
- SQL injection (виконання SQL-запитів з браузера для отримання несанкціонованого доступу до даних);
- База даних GHDB (Google hacking database) - перелік типових запитів, що використовуються хакерами для отримання несанкціонованого доступу до web-додатків і сайтів.
- Виконання коду:
- Обхід каталогу;
- Вставка файлів (File inclusion);
- Розкриття вихідного тексту сценарію;
- CRLF injection
- Cross frame scripting;

- Загальнодоступні резервні копії файлів і папок;
- Файли і папки, що містять важливу інформацію;
- Файли, які можуть містити інформацію, необхідну для проведення атак (системні логи, журнали трасування додатків і т.д.);
- Файли, що містять списки папок;
- Папки з низьким рівнем захисту, що дозволяють створювати, модифікувати або видаляти файли.

Повну інформацію по програмному продукту та його можливостях можна отримати за посиланням: http://www.acunetix.com/resources/wvsmanual.pdf

### 1.6.1. Робота зі сканером

Інтерфейс програми (рис. 1.11).

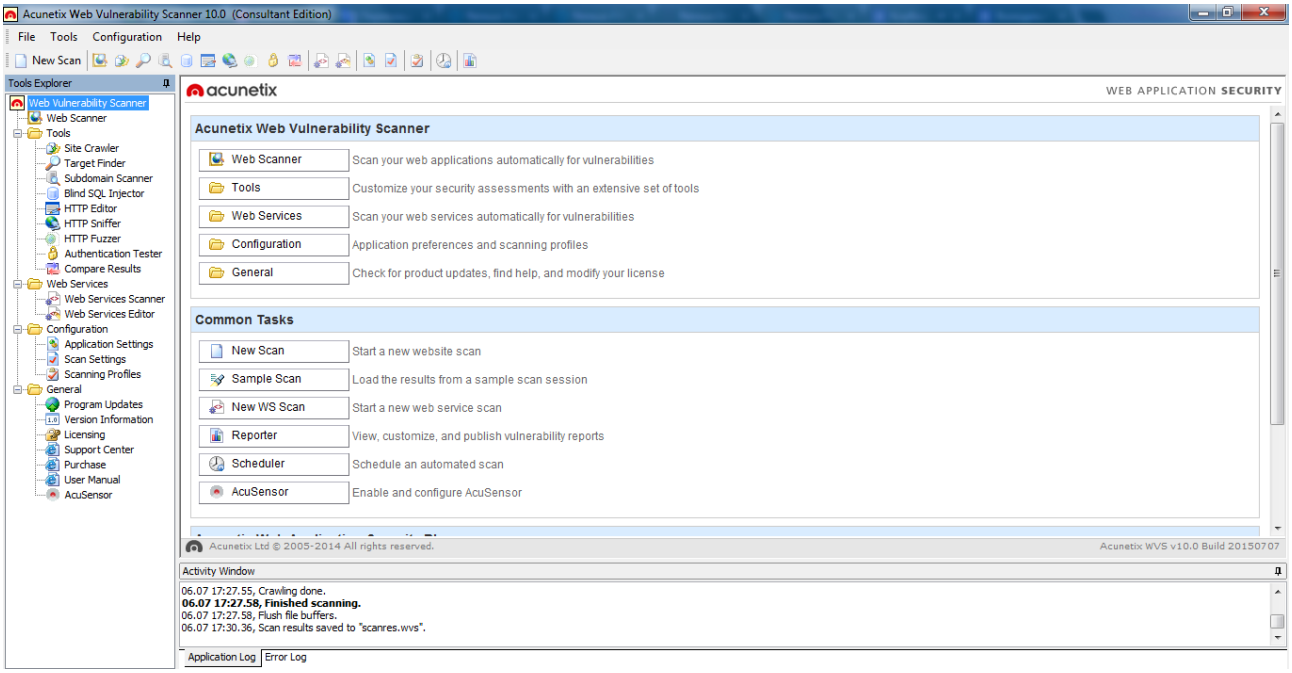



Рисунок 1.11 – Інтерфейс програми

Короткий опис інструментів(Tools)

Site Crawler - Створює карту веб-сайту шляхом відстеження всіх посилань і збирання інформації про кожен виявлений файл на сайті.

Target Finder - Виявляє будь-які HTTP / S сервери на IP-діапазоні, з можливістю сканування виявлених серверів на наявність вразливостей.

Subdomain Scanner – Визначає нові і незареєстровані піддомени домену вищого рівня, з можливістю сканування виявлених об'єктів на наявність вразливостей.

Blind SQL Injector - Автоматично використовує SQL ін'єкції для отримання даних з бази даних, використовуваної на сервері.

HTTP Editor – Дозволяє створення користувацьких HTTP/S запитів для аналізу відповіді сервера.

HTTP Sniffer – HTTP проксі для реєстрації, захоплення і модифікації всього перехопленого HTTP/HTTPS трафіка.

HTTP Fuzzer – Тести на такі вразливості як переповнення буфера і перевірка вхідних данних запитів з нечіткими заголовками і параметрами.

Authentication Tester – Перевіряє надійність пароля, виконуючи словникові атаки використовуючи HTTP, NTLM методи чи методи аутентифікації на основі форм.

Compare Results – Порівняння збережених результатів сканувань і відображення їх різниці.

Acunetix AcuSensor – підвищує ефективність сканування за рахунок поліпшення crawling, виявлення і повідомлення про знаходження вразливостей. AcuSensor може бути використаний на .NET і PHP веб-додатках.

Сканування відбувається в 2 етапи – обхід і власне сканування.

Створимо нове скануваня (File-New-Web Site Scan). Після чого з’явиться вікно Майстера сканування (рис. 1.12).

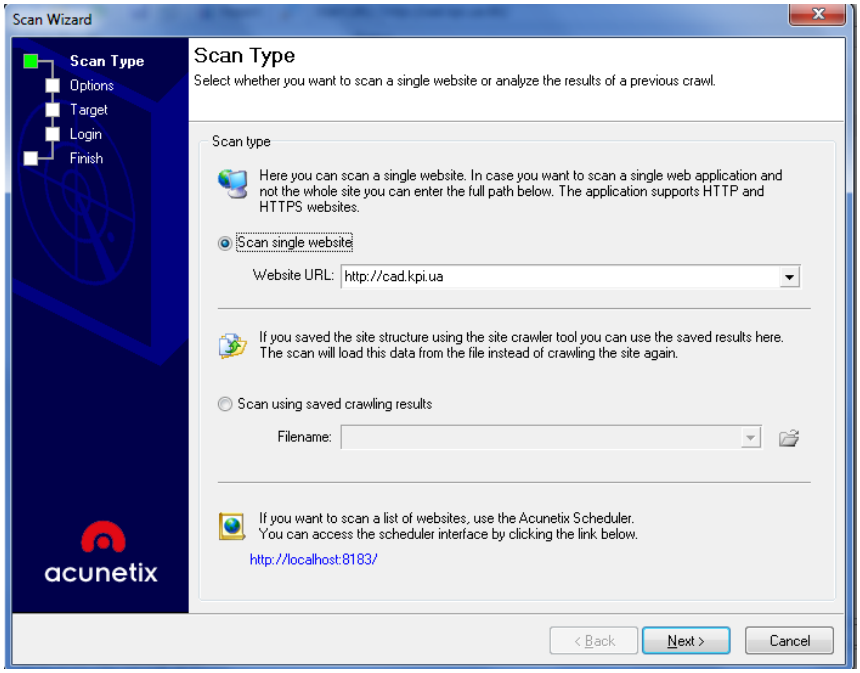



Рисунок 1.12 – Майстер сканувань

На першій сторінці Майстра розміщені три кнопки:

- Сканувати один сайт (HTTP або HTTPS). Введіть URL або IP адреса сайту, який необхідно просканувати;
- Сканування за допомогою збережених результатів обходу сайту (обхід використовується для побудови структури сайту; в процесі обходу перераховуються всі файли сайту для подальшого їх сканування) – обравши цей режим ми завантажуємо готовий результат роботи пошукового агенту сайту(результати обходу), замість того щоб робити обхід знову;
- Сканувати перелік сайтів використовуючи Планувальник.

Розглянемо Sheduler(планувальник) детальніше (рис. 1.13).


Рисунок 1.13. - Планувальник

Ми можемо додати нове сканування і обрати базові й розширені параметри для нього, а також обрати налаштування для результатів сканування і звіту. Налаштування параметрів планувальника (рис. 1.14 – рис. 1.17).

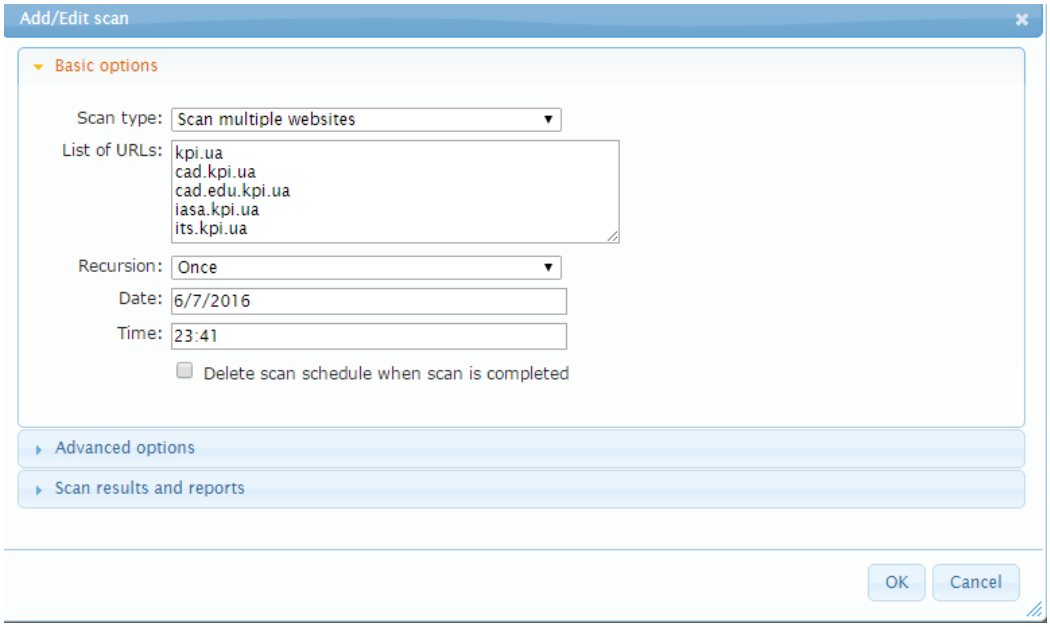



Рисунок 1.14

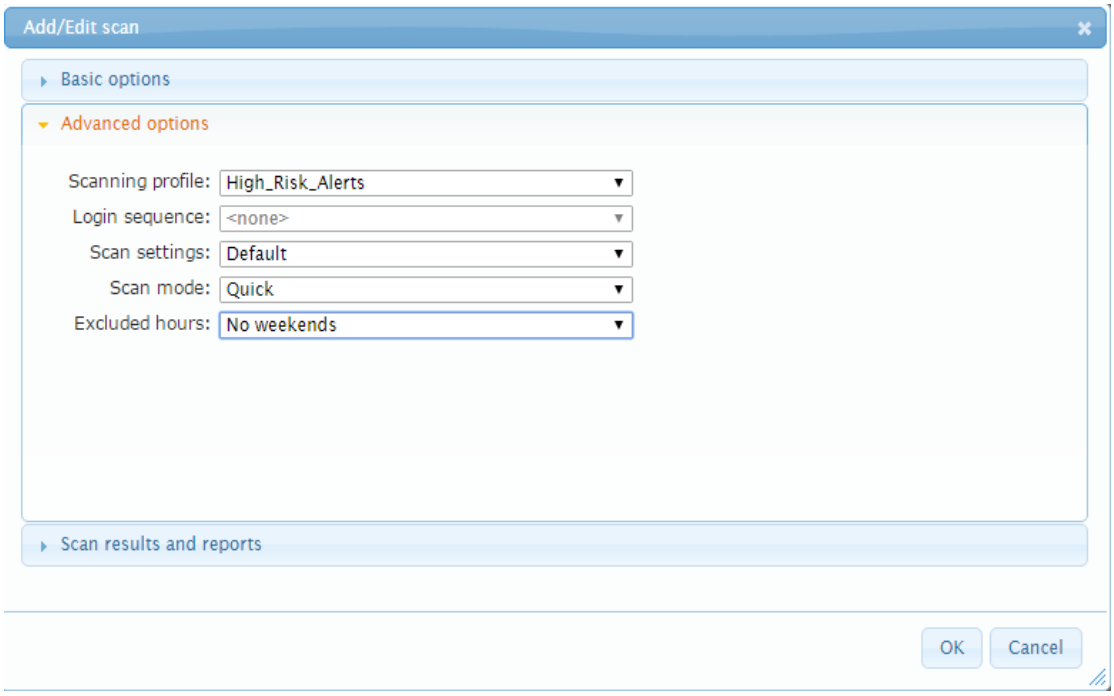



Рисунок 1.15

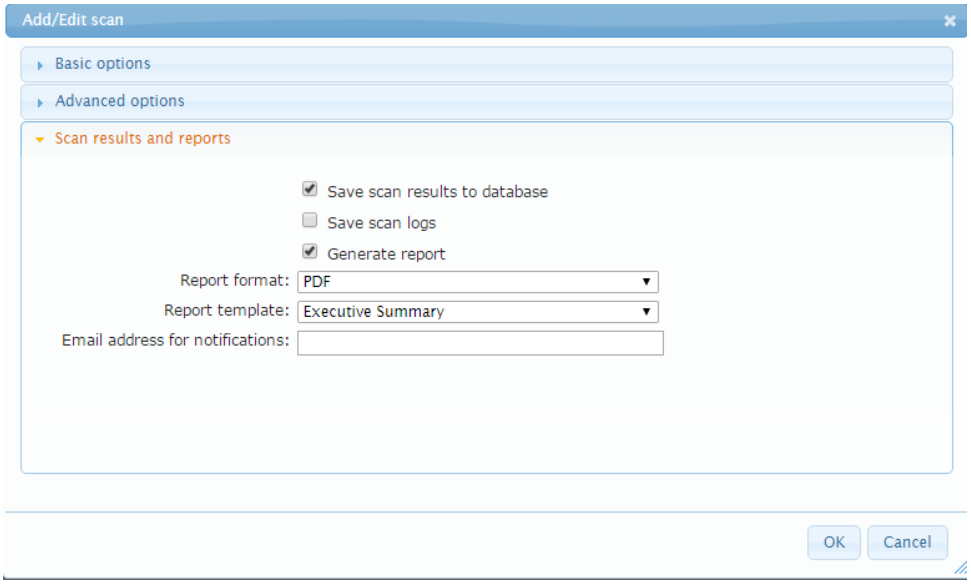



Рисунок 1.16

Після чого отримаємо наступне:

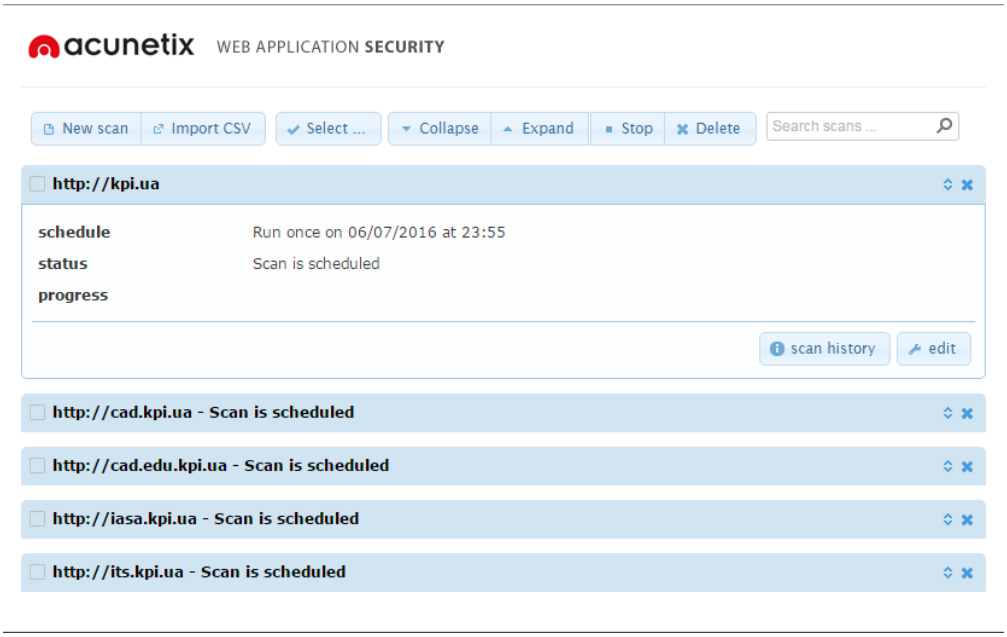



Рисунок 1.17

Після сканування можна буде завантажити результати по кожному з сайтів у вигляді архівів, які міститимуть дані (рис. 1.18).

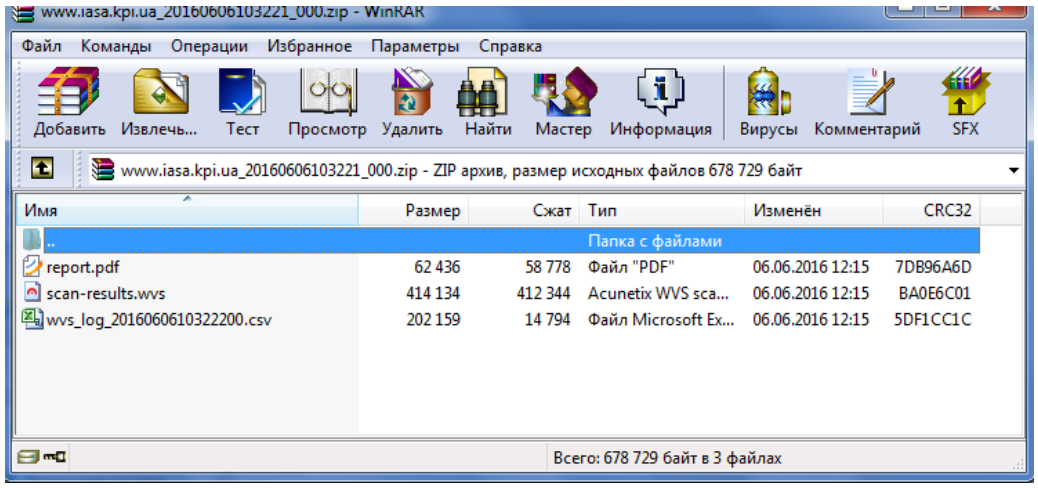



Рисунок 1.18 – Завантаження результатів

Повернемось до вікна майстра сканування і оберем перший варіант. Тепер можна обрати налаштування та профіль сканування. Налаштування майстра сканування (рис. 1.19 – рис. 1.22).

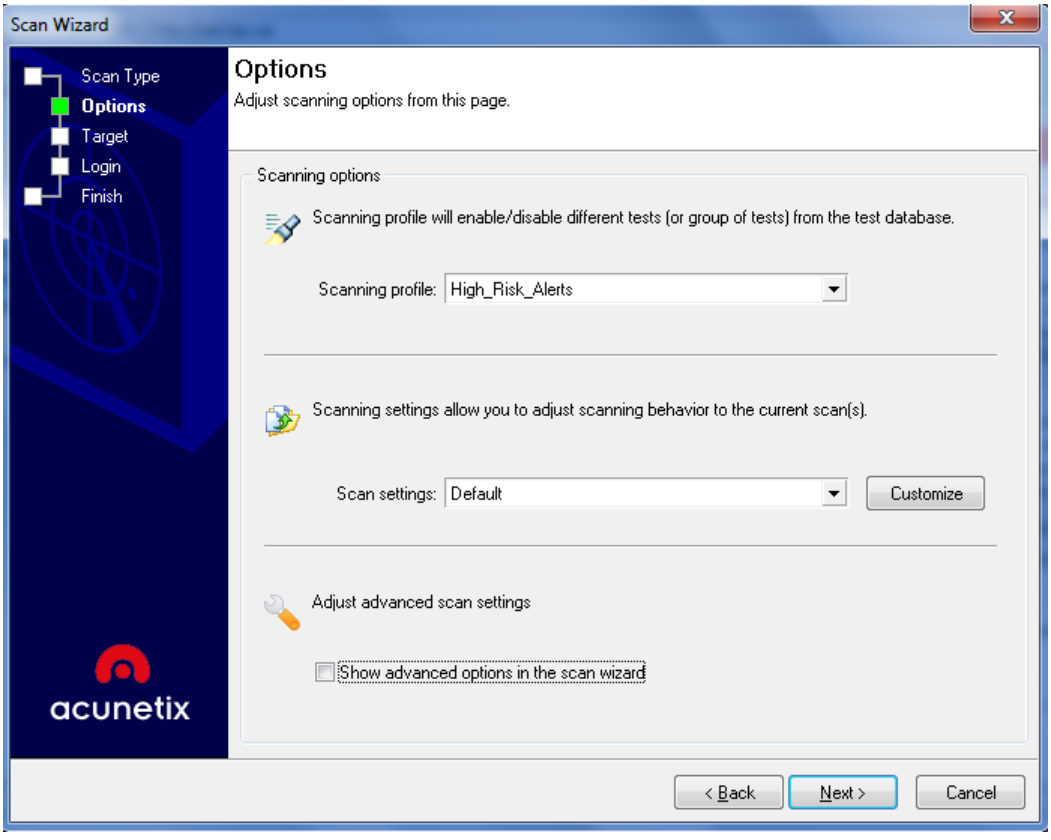



Рисунок 1.19

В наступному вікні можна додати деталі для об’єкта сканування. Це зменшить час сканування.

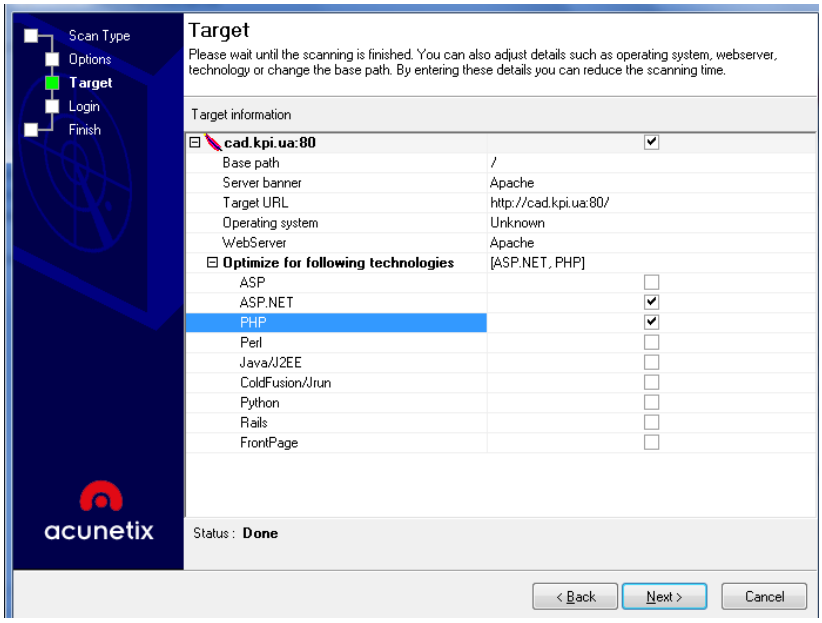



Рисунок 1.20

На наступному етапі можна додати дані для аутентифікації на сканованому сайті (якщо потрібно).

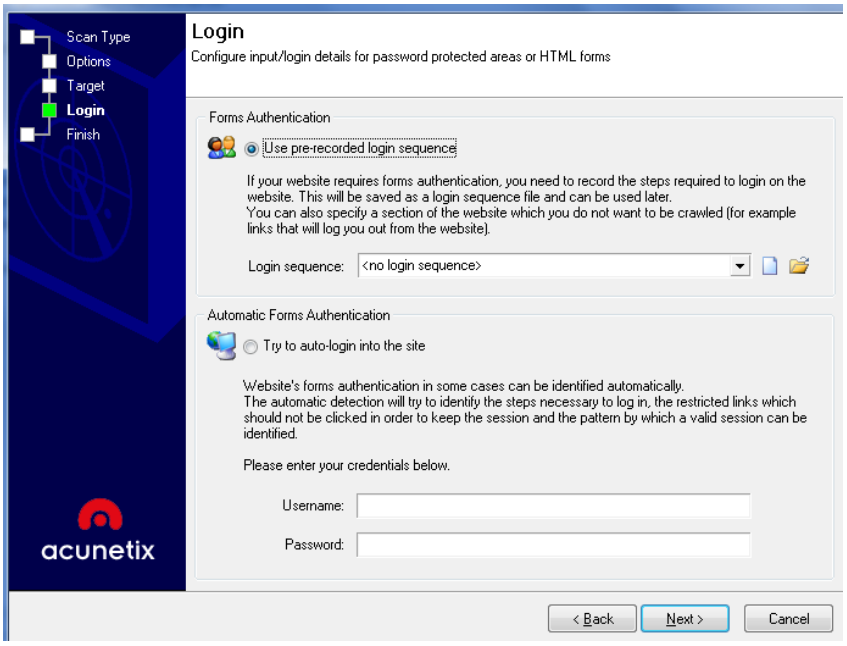



Рисунок 1.21

В останньому вікні нам надають список рекомендацій для поточного сканування, виходячи з аналізу відповідей веб-сайту. Також в цьому вікні можна зберегти поточні налаштування, як новий шаблон для можливості подальшого використання.

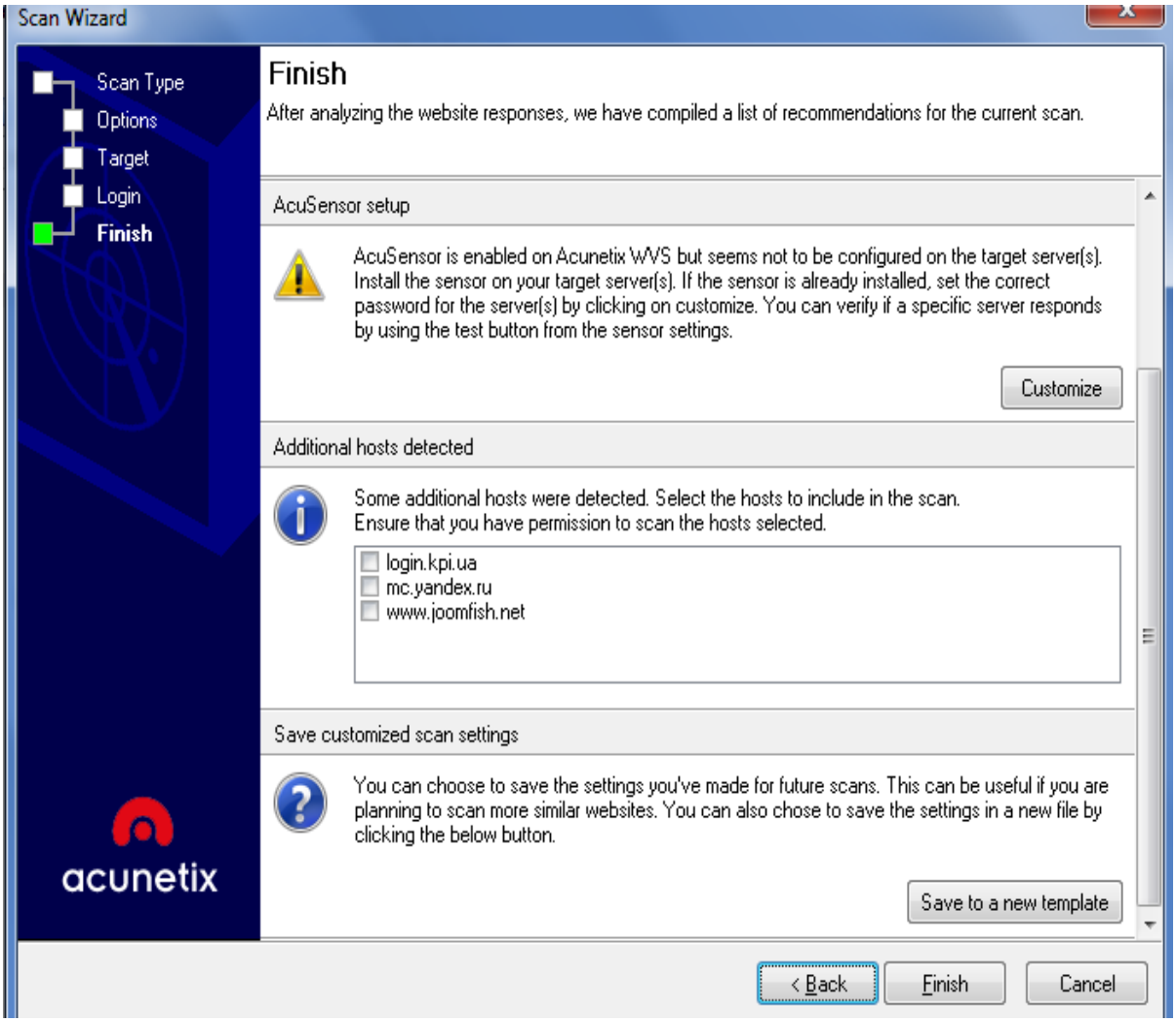



Рисунок 1.22

Після цього ми нарешті можемо розпочати сканування.

Результати сканувань зберігаються в базі даних сканувань. Використовуючи їх можна згенерувати звітню документацію. Для цього відкриємо Reporter(Майстер звітів) (рис. 1.23).

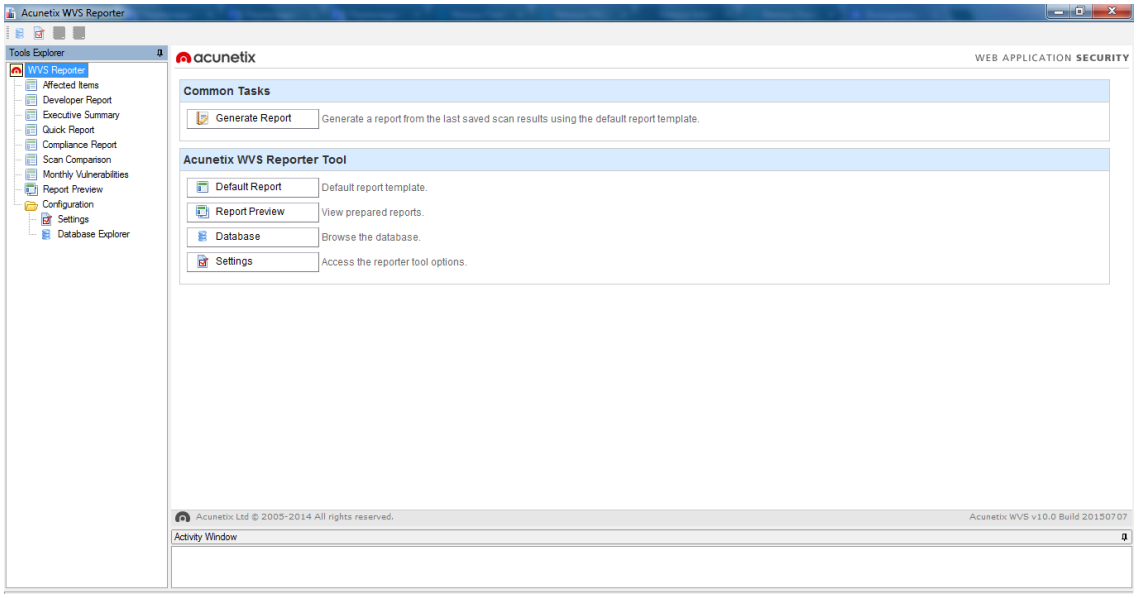



Рисунок 1.23 – Майстер звітів

Оберемо звіт у вигляді Executive Summary. Можна одразу згенерувати звіт. Також можна обрати параметри в Settings і в Report Wizard, що впливатимуть на зміст звіту. Згенеруємо звіт і оберем Report Preview для попереднього перегляду (рис. 1.24). Після чого можна експортувати звіт у PDF або інші доступні формати.

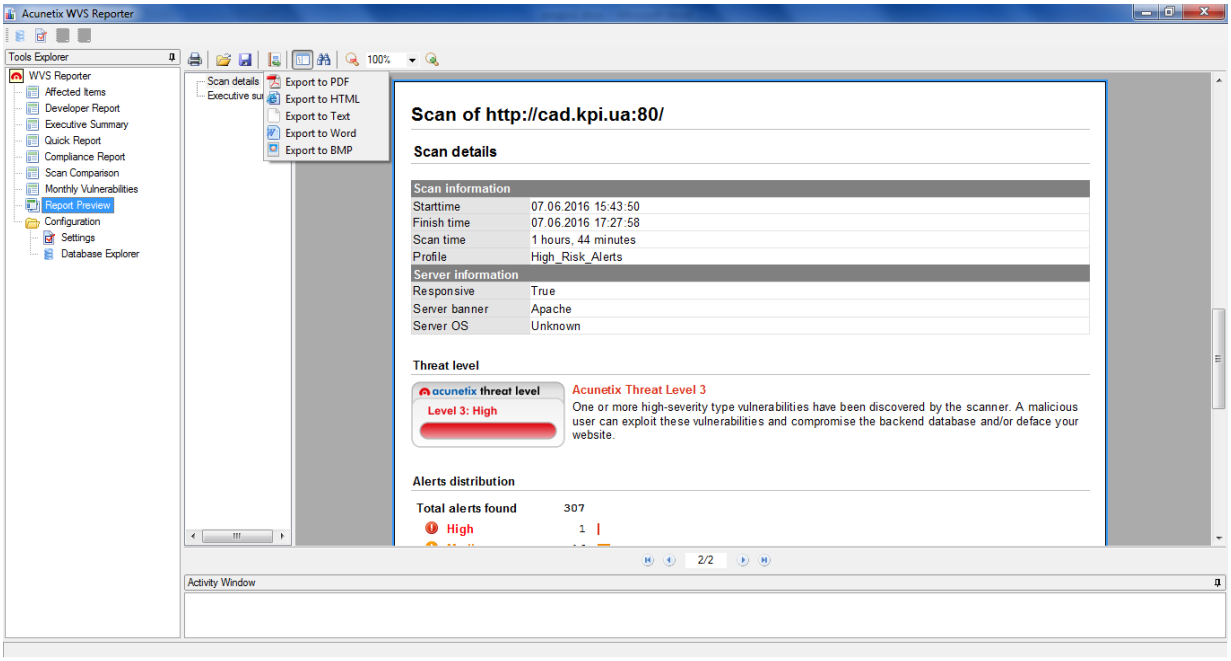



Рисунок 1.24 – Попередній перегляд згенерованого звіту

Даний програмний продукт дозволяє створювати найрізноманітнішу звітню документацію і деталізувати її зміст. Також можливе створення звітної документації згідно міжнародних стандартів.

## 1.7. Shadow Security Scanner v7.303 Build 309

Shadow Security Scanner - сканер мережевої безпеки який завдяки унікальним методам дозволяє надійно перевірити ваш сайт або мережу на наявність дір і дозволить надійно захистити вашу мережу від проникнення хакерів. Якщо ви дійсно хочете постійно утримувати свою мережу в цілковитій безпеці і весь час бути в курсі нових багів в системі безпеки, в цьому вам допоможе повністю автоматизований засіб для сканування ваших мереж.

Shadow Security Scanner (SSS) сканер створений для швидкого і надійного виявлення прогалин у системі безпеки як відомих так і не відомих на момент випуску нової версії продукту. При скануванні системи, SSS виробляє аналіз даних, виявляє уразливі місця, можливі помилки в налаштуванні сервера і запропонує можливі шляхи виправлення недоліків та вразливих місць в системі. SSS має унікальну технологію виявлення можливих дірок в системі, використовуючи систему засновану на інтелектуальному ядрі власної розробки, SSS сканує систему з такою швидкістю і точністю, на яку здатні лише професійні служби безпеки або хакери, які намагаються зламати вашу мережу.

SSS на сьогоднішній день найкращий сканер, що володіє багатими можливостями, ідеальний помічник у перевірці сервера системного адміністратора, незамінний інструмент в роботі служби безпеки вашого підприємства або організації. SSS працюючи під операційною системою Windows, сканує сервера незалежно від того на якій платформі вони побудовані. При скануванні SSS виявляє прогалини в системі безпеки серверів побудованих як на платформі Unix (Linux, FreeBSD, OpenBSD, Net BSD, Solaris and etc) так і на Windows 95/98 / ME / NT / 2000 / XP / .NET. Використовуючи

унікальну технологію, SSS може визначати помилки в обладнанні CISCO, HP та інших.

На даний момент проводитися перевірка наступних основних сервісів: FTP, SSH, Telnet, SMTP, DNS, Finger, HTTP, POP3, IMAP, NetBios, NFS, NNTP, SNMP, Squid (SSS перевіряє проксі сервера на предмет аудитів - більшість сканерів просто визначають наявність порту), LDAP (SSS перевіряє LDAP сервера на предмет аудитів - більшість сканерів просто визначають наявність порту), HTTPS, SSL, TCP / IP, UDP, Registry, Сервіси. Маючи повністю відкриту архітектуру (побудовану на ActiveX) будь-який програміст, який знайомий з VC ++, C ++ Builder, Delphi може без проблем і в найкоротший час розширити можливості SSS. SSS має функції прямого доступу до свого ядра що дозволяє вам через API повністю управляти або змінювати властивості і функції SSS. Також завдяки технології ActiveX фахівець з комп'ютерної безпеки або системний адміністратор - може інтегрувати SSS в будь-який продукт, що підтримує ActiveX.

Використовуючи Редактор Правил та налаштування, користувач може вибрати для сканування тільки ті порти і сервіси, які він вважає за потрібне, і не витрачати час на сканування інших сервісів. Гнучкість налаштування дозволяє адміністратору керувати як повнотою так і глибиною сканування, що дозволяє прискорити сканування мережі не на шкоду якості.

Для підвищення швидкодії введена функція одночасного множинного сканування мережі (до 10 хостів одночасно). Дружній і простий інтерфейс максимально оптимізований для зручності використання SSS. Вся інформація в сканері добре структурована і інтуїтивно зрозуміла. SSS має легке управління усіма параметрами, до всіх елементів програми є спливаючі підказки, які найбільш точно описують призначення елемента управління, будь то кнопка або поле для введення даних.

Майстер Оновлень дозволяє оперативно оновлювати базу даних вразливостей системи та виконавчі модулі, тим самим забезпечуючи міцний захист системи і досить високу стійкість до різних атак зловмисників. База даних оновлюється практично кожен день, що дозволяє Адміністратору не побоюватися появи нових прогалин в системі, знайдених після випуску останньої версії SSS, так як він завжди буде мати повністю захищений сервер, а також оперативно мати методи виправлення всіх нових вразливостей.

### 1.7.1. Робота зі сканером

Загальний вид програми після її запуску (рис. 1.25).

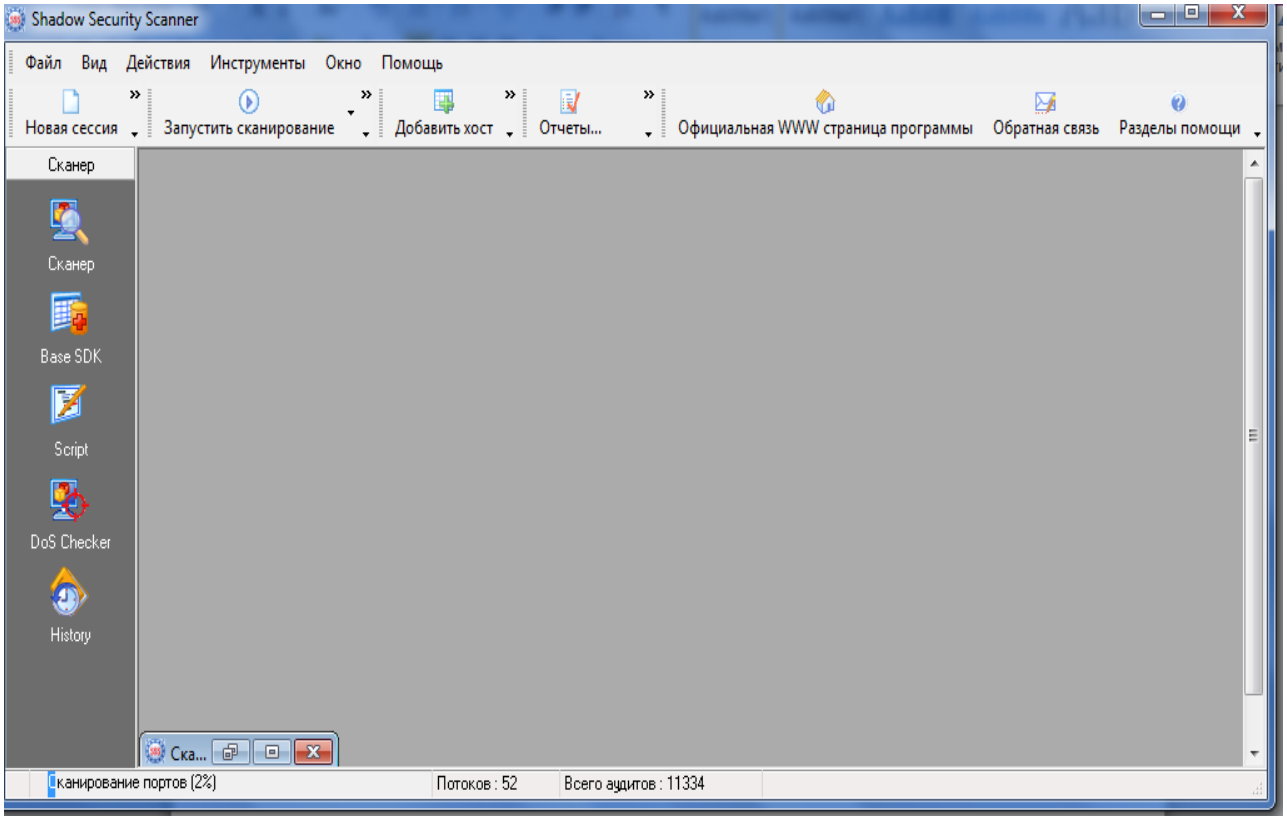



Рисунок 1.25 – Загальний вид програми

В опціях є можливість налаштувати планувальник для запуску сканування по розкладу (рис. 1.26).

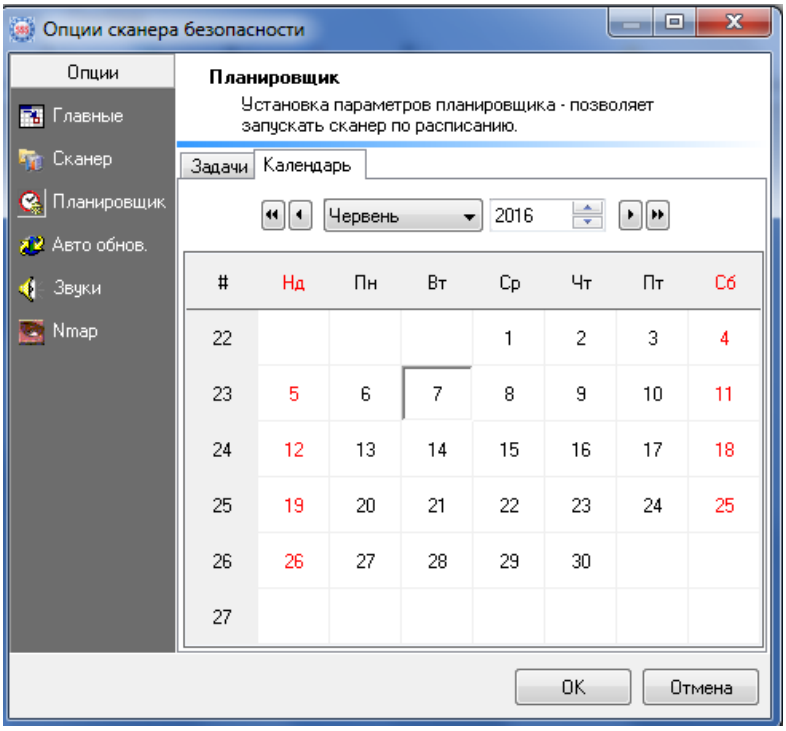



Рисунок 1.26 – Налаштування планувальника

Для початку роботи потрібно створити нову сесію чи відкрити збережену. Вікно створення нової сесії (рис. 1.27).

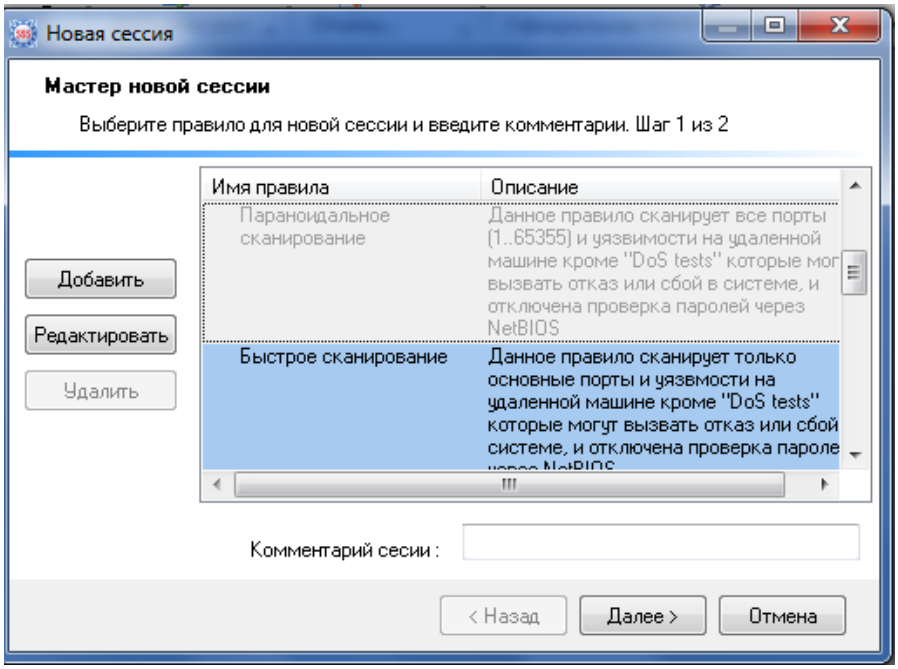



Рисунок 1.27 – Створення нової сесії

Після того як було обрано правило для сканування можна натиснути кнопку «Редагувати» після чого ми отримаємо доступ до більш тонкого налаштування правила сканування (рис. 1.28).

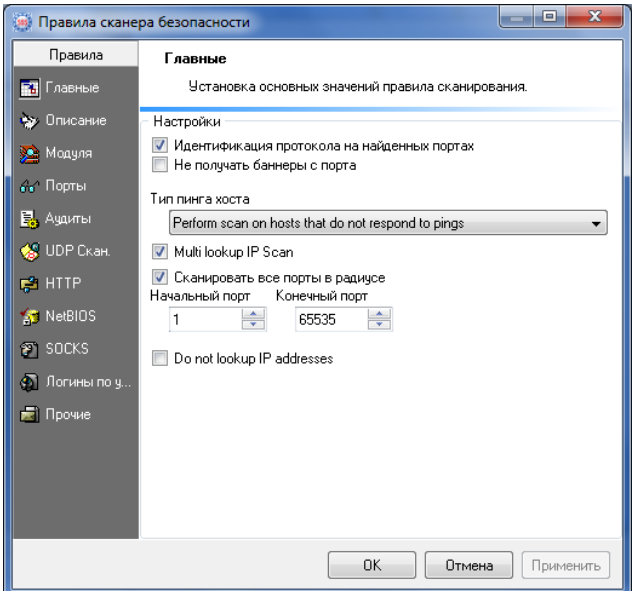



Рисунок 1.28 – Налаштування правил сканування

Далі перейдемо до наступного кроку створення сесії – додавання хоста чи хостів для сканування (рис. 1.29).

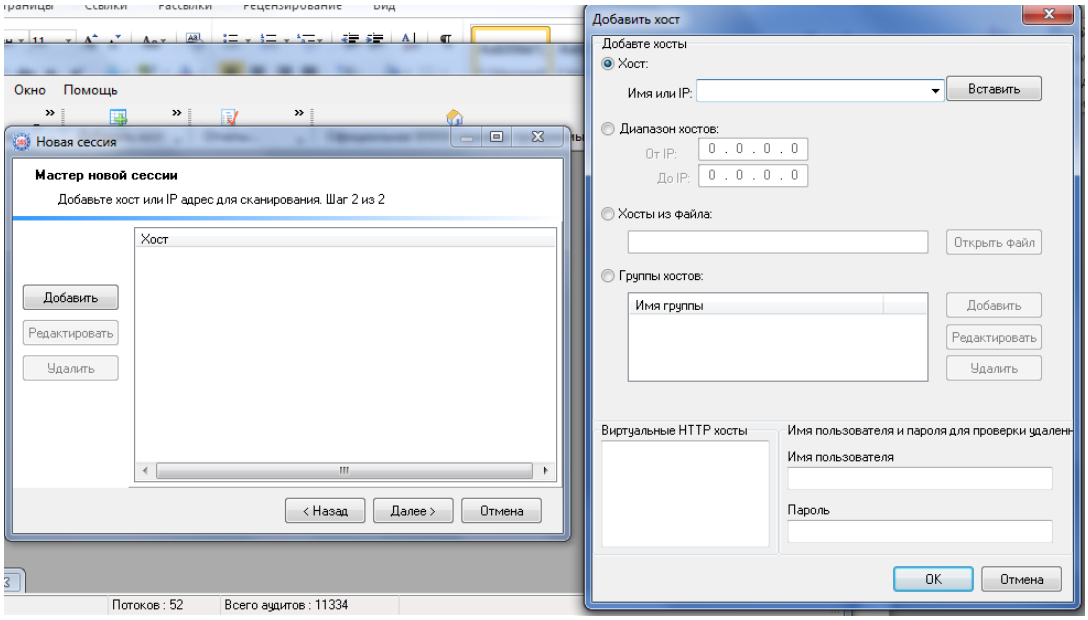



Рисунок 1.29 – Додавання хостів для сканувань

Після цього залишається лише розпочати сканування і чекати на його завершення.

Якщо ж Ви не володієте жодною мовою програмування, але знайшли нову уразливість (і володієте елементарними знаннями мережевих технологій) Ви можете повідомити про це або скористатися BaseSDK, який в режимі помічника проведе вас від початку до кінця додавання нового аудиту. Використання BaseSDK (рис. 1.30) дозволить вам в разі необхідності додати більше95% типів аудитів.

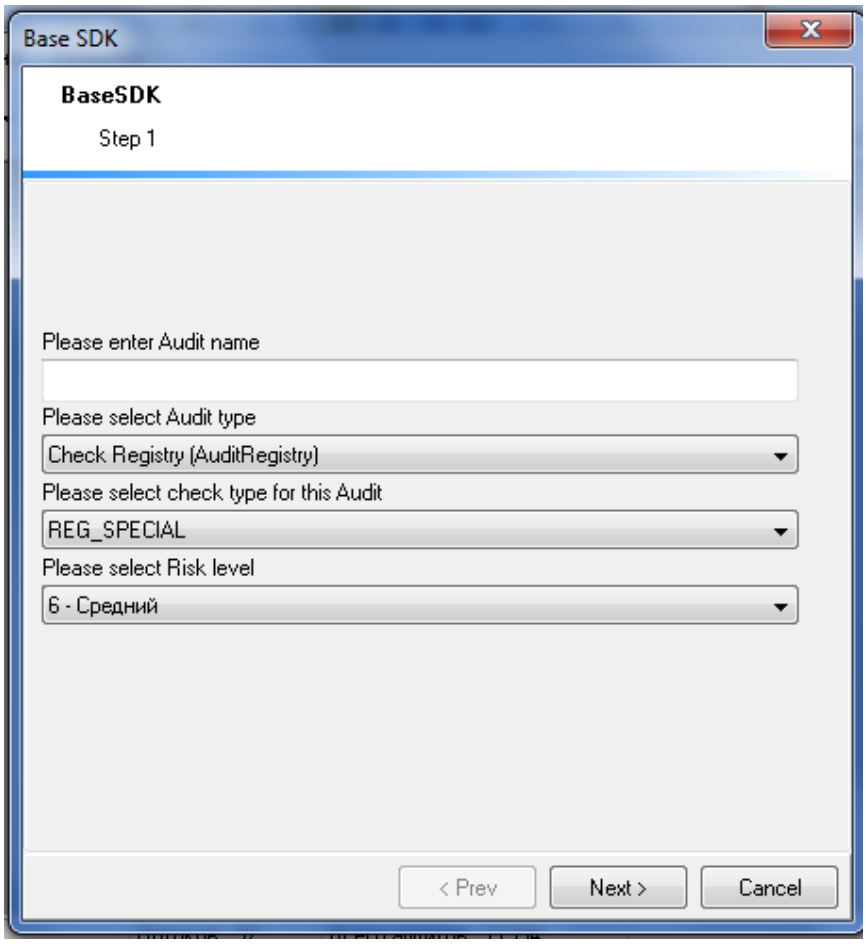



Рисунок 1.30 – Використання Base SDK

Також є можливість перевірки на DoS тести і перегляду історії сканувань. Вікно перевірки на DoS тести (рис. 1.31).

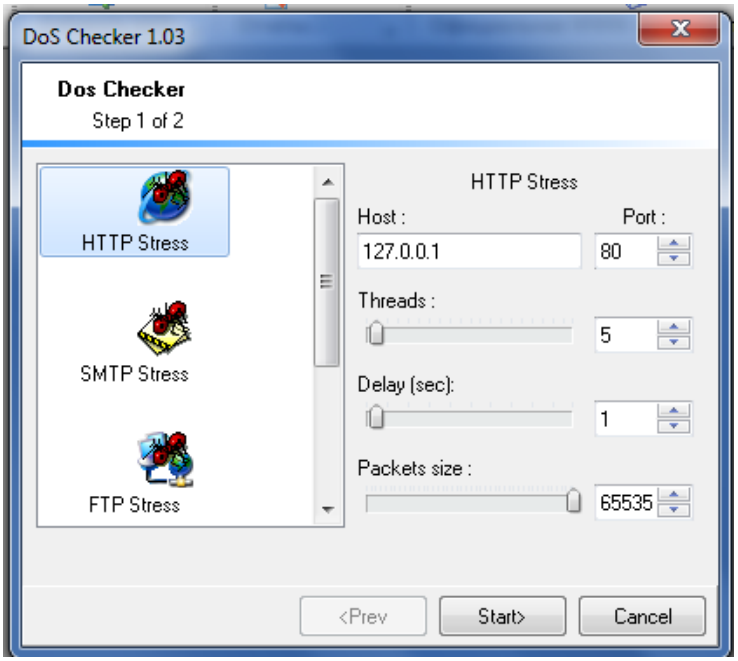



Рисунок 1.31 – Вікно перевірки на DoStести

Загальний вигляд програми під час сканування (рис. 1.32).

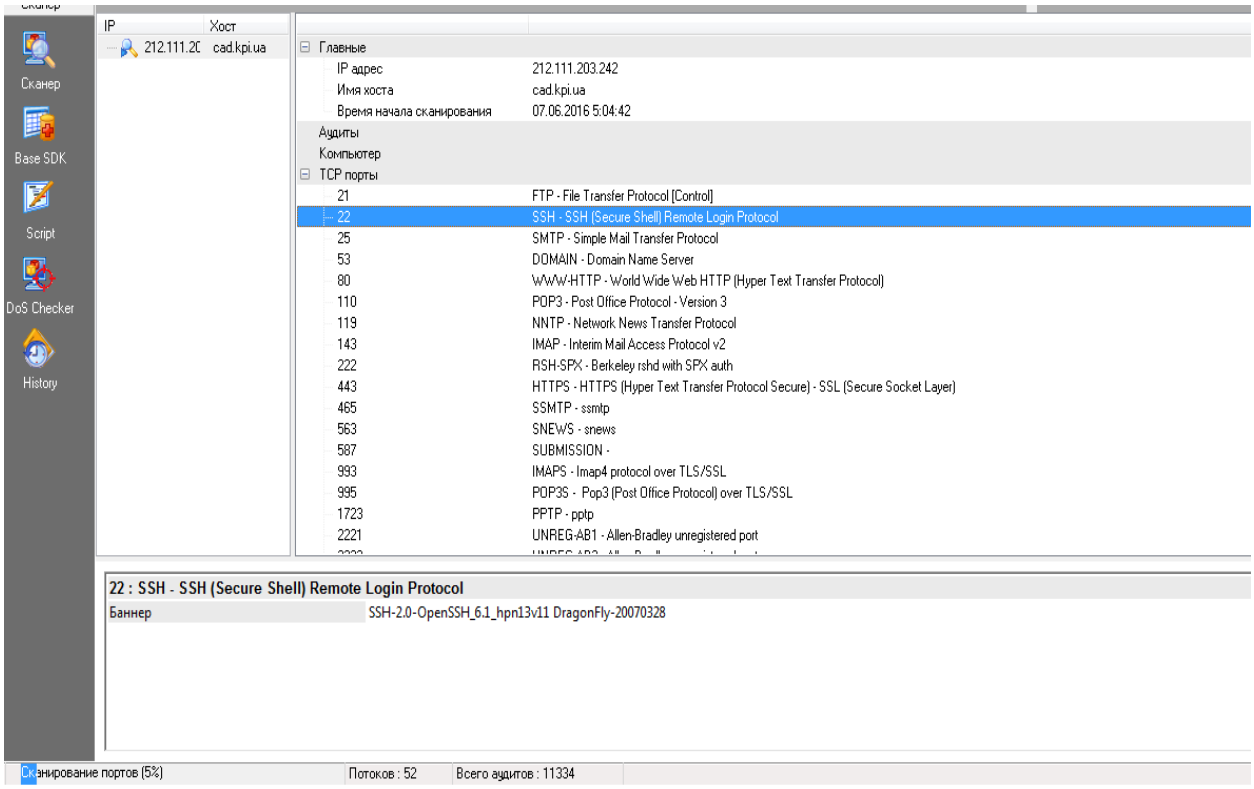



Рисунок 1.32 – Програма під час сканування

Після завершення сканування можна створити звіт за допомогою майстра звітів, вибравши результат сканування, аудити, тип сортування хостів і аудитів, модулі звіту і формат. Майстер звітів (рис. 1.33).

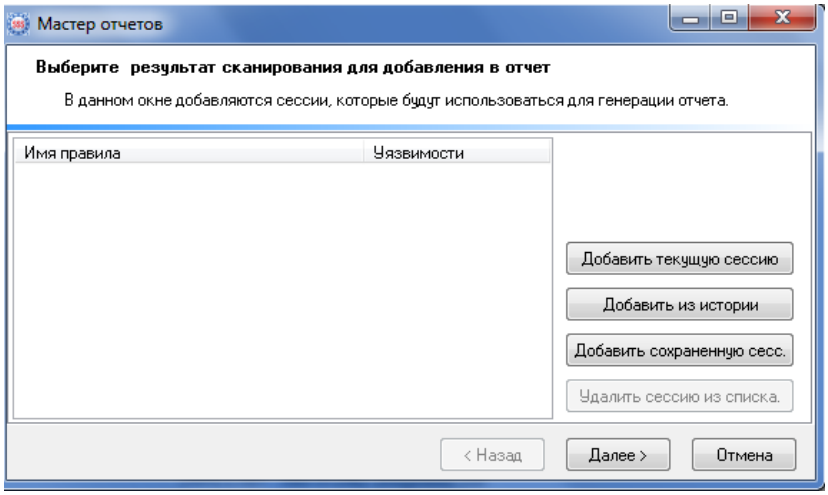



Рисунок 1.33 – Майстер звітів

## 1.8. Порівняння деяких можливостей та функціоналу обраних сканерів.

Розглянемо загальні характеристики обраних сканерів у вигляді таблиці.

Таблиця 1.1 – Загальні характеристики обраних сканерів

| Назва | XSpider | Acunetix | SSS |
|---|---|---|---|
| Інтерфейс | Досить простий | (Багато функціоналу) Складний у порівнянні з іншими | Простий та зрозумілий( кожен елемент має спливаючі підказки) |
| Мова інтерфейсу | Російська | Англійська | Російська або Англійська |
| Можливість планувати сканування | + | + | + |

Таблиця 1.1 – Загальні характеристики обраних сканерів (продовження)

| Назва | XSpider | Acunetix WVS | SSS |
|---|---|---|---|
| Одночасне сканування декількох хостів | + | Залежно від ліцензії(від 1 до 10 і більше) | + |
| Боротьба зі знайденими вразливостями | Опис, спосіб рішення, корисні посилання для вирішення. | Опис, спосіб рішення, корисні посилання для вирішення | Опис, спосіб рішення, корисні посилання для вирішення |
| Звіти | Декілька видів звітів. Звіти генеруються в html (xml, rtf в нових версіях) | Reporter (окрема програма для звітів). Багато різновидів. Детальне налаштування. Експорт у різні формати | Формати: html, xml, chm, pdf, sql, user формат. Вибір аудитів та розділів звіту. |
| Оновлюваність | База вразливостей поповнюється фахівцями Positive Technologies+автоматичне оновлення баз і модулів програми | Можливість завантажувати оновлення та нові збірки через програмний інтерфейс | Майстер оновлень оперативно оновлює базу даних вразливостей системи і виконавчі модулі. Можливість ручного додавання аудитів через Base SDK. |

Розглянемо порівняльну характеристику функціональних можливостей обраних засобів пошуку вразливостей. Порівняємо вразливості, які первіряються та об’єкти, що скануються обраними засобами. Порівняння виконаємо у вигляді таблиці, яку наведено нижче.

Таблиця 1.2 – Характеристика функціоналу обраних засобів

| Назва сканеру | Сканування вразливостей | Об’ єкти, що скануються |
|---|---|---|
| XSpider | Аутентифікація форм, нестандартні DoS атаки, SQL ін’єкції, ін’єкції коду, виконання довільних програм, отримання файлів, міжсайтовій скриптинг (XSS), HTTP Response Splitting, пошук та аналіз директорій доступних для перегляду і запису | Сканує все, незалежно від програмного та апаратного забезпечення вузлів. Обробка RPC-сервісів (Windows і * nix), HTTP, FTP, SMTP, POP3, DNS, SSH, мережеві пристрої CISCO. |
| SSS | Аутентифікація форм, DoS атаки, міжсайтовий скриптинг, SQL ін’єкції, переповнення буферу, виконання коду, вставка файлів і т.д. | FTP, SSH, Telnet, SMTP, DNS, Finger, HTTP, POP3, IMAP, NetBios, NFS, NNTP, SNMP, Squid, LDAP, HTTPS, SSL, TCP / IP, UDP, Registry, Сервіси. Windows/Unix сервери, обладнання CISCO та HP. |

Таблиця 1.2 – Характеристика функціоналу обраних засобів (продовження)

| Назва сканера | Сканування вразливостей | Об’єкти, що скануються |
|---|---|---|
| Acunetix WVS | Аутентифікація форм, Cross site scripting , SQL ін’єкції, база даних GHDB (Google hacking database), виконання коду, обхід каталогу, вставка файлів (File inclusion), розкриття вихідного тексту сценарію, CRLF injection, Cross frame scripting, загальнодоступні резервні копії файлів і папок, файли і папки, що містять важливу інформацію, файли, які можуть містити інформацію, необхідну для проведення атак (системні логи, журнали трасування додатків і т.д.), файли, що містять списки папок, папки з низьким рівнем захисту, що дозволяють створювати, модифікувати або видаляти файли. | Віртуальні вузли, в незалежності від типу веб-сервера і операційної системи. Сканування доступу до FTP, DNS, SMTP, IMAP, POP3, SSA, SNMP, Telnet. |

## 1.9. Висновки до розділу 1

В даному розділі були розглянуті основні вразливості Web-сайтів та серверів, а також наведені їх приклади. Також була розглянута ієрархія та рівні захисту Web-серверів.

Далі були розглянуті основні принципи роботи сканерів вразливості. Після цього були обрані деякі відомі сканери вразливостей, а саме: XSpider, Acunetix WVS та Shadow Security Scanner.

Для обраних сканерів був проведений огляд роботи, описані їх загальні характеристики та функціональні можливості. Після чого було проведене порівняння отриманих характеристик у вигляді таблиць.

Як бачимо, обрані сканери мають досить схожі можливості. Всі вони мають добру оновлюваність. Дані сканери здатні знаходити поширені вразливості та пропонувати для них рішення.

З обраних сканерів, Acunetix WVS має найбільше інструментів для пошуку вразливостей, а також найкращу систему створення звітів.

# 2. РЕЗУЛЬТАТИ СКАНУВАНЬ, ЇХ АНАЛІЗ І ПОРІВНЯННЯ

В цьому розділі буде проведено аналіз та порівняння результатів сканувань обраними засобами. Результати буде отримано шляхом сканування переліку веб-сайтів: kpi.ua, cad.kpi.ua, cad.edu.kpi.ua, iasa.kpi.ua, its.kpi.ua.

## 2.1. Результати роботи Xspider (full version)

В даній версії звіти генеруються в html форматі. По результатам перевірки було згенеровано звіт для системного адміністратора. Даний звіт містить загальну статистику, інформацію по кожному хосту окремо. Для кожного хоста міститься детальна інформація про сервіси і вразливості з описом. Для вразливостей типу «Підозра на помилку» і вище, наведені рішення і потрібні посилання. Далі наведено початок звіту та статистику (рис. 2.1 – рис. 2.2).

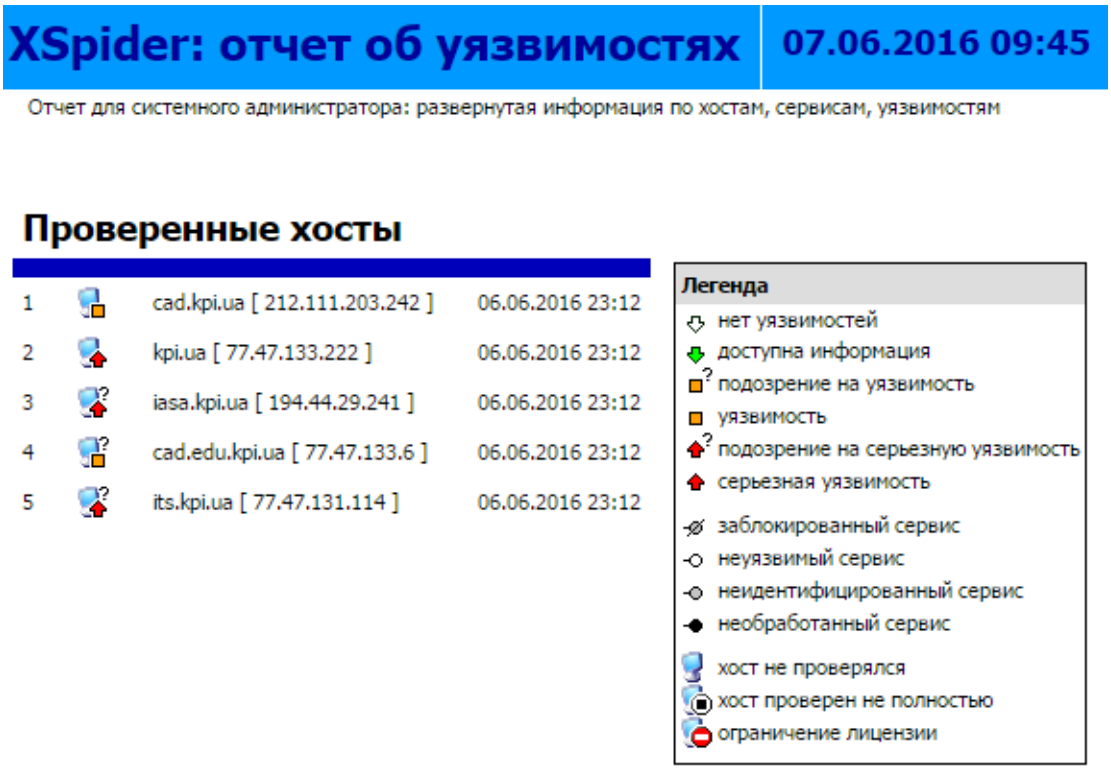



Рисунок 2.1

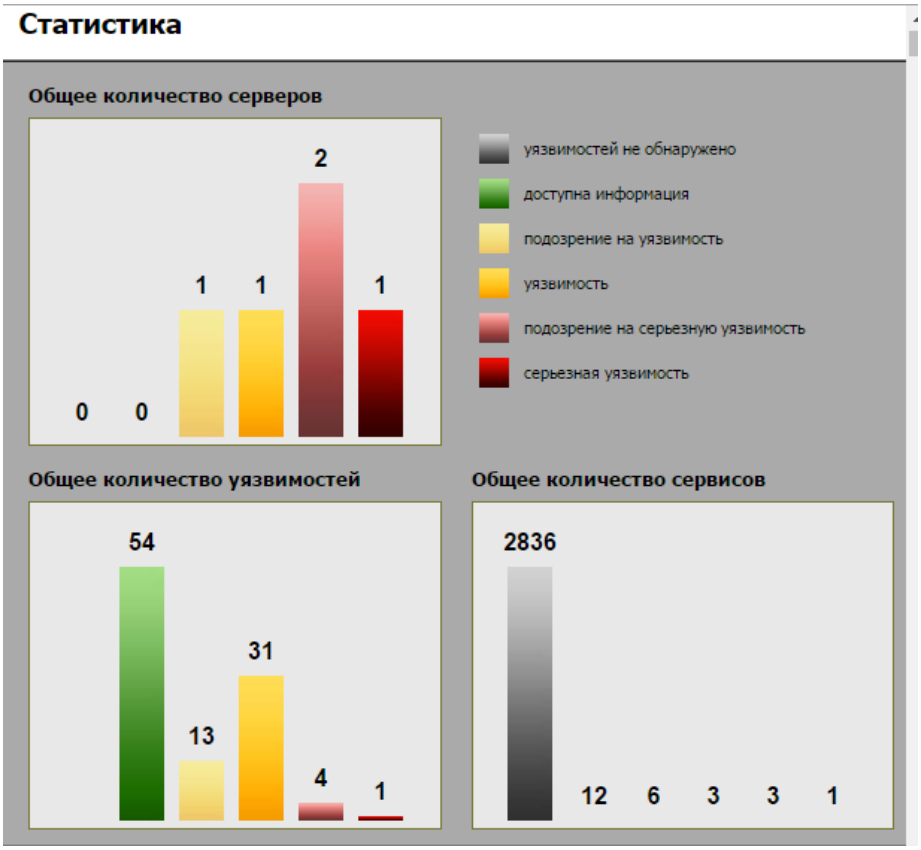



Рисунок 2.2

### 2.1.1. Результати сканування XSpider Demo та порівняння з попередньой версією.

Розглянемо статистику роботи демонстраційної версії програми для обраного переліку хостів. Після чого порівняємо отримані результати з результатами роботи повної версії. Маємно наступні результати:

- kpi.ua (вразливості – 35, доступна інформація – 8);
- cad.kpi.ua (вразливості – 16 , підозра на вразливість – 7, підозра на серйозну вразливість – 2, доступна інформація – 17);
- cad.edu.kpi.ua (доступна інформація – 6);
- iasa.kpi.ua (доступна інформація – 2);
- its.kpi.ua (вразливість – 2, підозра на вразливість – 6, доступна інформація – 17);

Загальна статистика: доступна інформація – 50, підозра на вразливість – 13, вразливість – 53, підозра на серйозну вразливість – 2.

По хостам: доступна інформація – 2 (iasa.kpi.ua, cad.edu.kpi.ua), вразливість – 2 (kpi.ua, its.kpi.ua), підозра на серйозну вразливість – 1 (cad.kpi.ua). А за результатом попередньої версії: підозра на вразливість – 1 (cad.edu), вразливість – 1 (cad), підозра на серйозну вразливість – 2 (iasa, its), серйозна вразливість – 1 (kpi.ua).

Як бачимо результати дещо відрізняються, а саме: у Demo версії більша кількість знайдених вразливостей – 53 ( в попередній версії 31) через новішу базу, але відсутній детальний опис багатьох вразливостей, також на cad.edu.kpi.ua і iasa.kpi.ua взагалі не знайдено серйозних чи звичайних за серйозністю вразливостей чи підозр на них.

Можна зробити висновок що повна стара версія даної програми виявляє вразливості краще за нову демонстраційну.

## 2.2. Результати роботи Acunetix WVS.

Даний сканер дуже потужний і має дуже багато різних перевірок, тому сканування займає дуже багато часу (декілька днів). Крім того в даній версії ліцензії програми не можливе одночасне сканування декількох сайтів, тому проведемо сканування одного з обраних сайтів.

Сканування cad.kpi.ua в режимі High_Risk_Alerts (рис. 2.3). Даний режим сканування спрямований в першу чергу на пошук вразливостей з всоким ступенем ризику. Перевіримо це, провівши сканування та проаналізуємо отримані результати для подальшого їх порівняння з результатами роботи інших сканерів пошуку вразливостей Web-сайтів та серверів.

Отже сканування виглядатиме наступним чином:

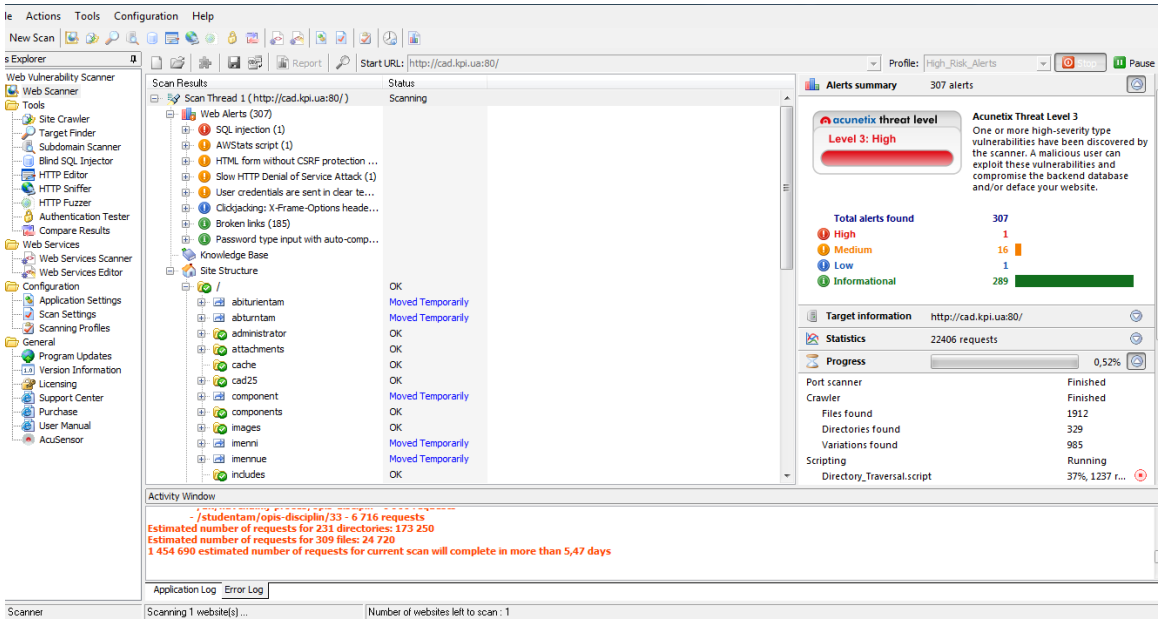



Рисунок 2.3 – Сканування

Основний етап сканування завершився, а як видно у Aplication Log (рис. 2.4) – скриптинг буде тривати понад 5 днів, що нас не влаштовує.

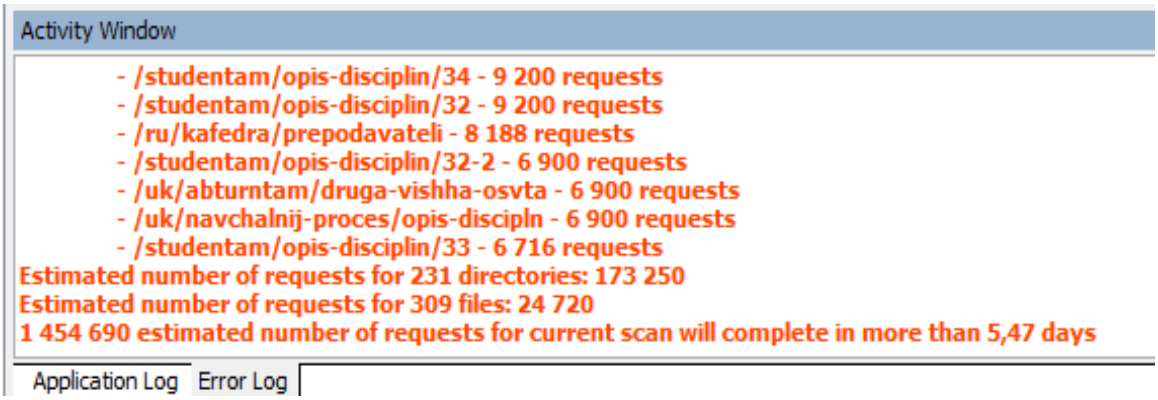
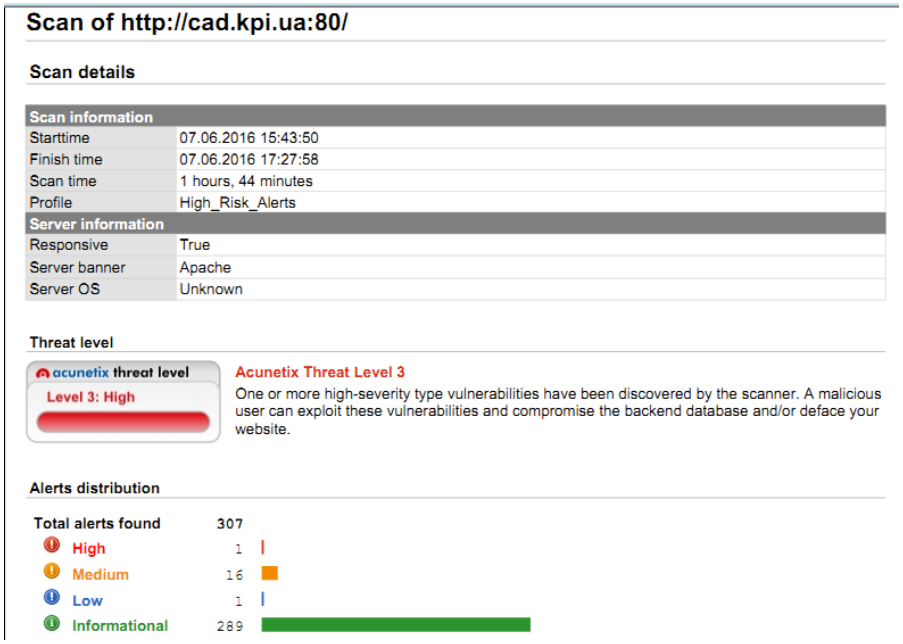



Рисунок 2.4 – Aplication Log

Оскільки було згенеровано більше 20000 запитів і знайдено понад 300 вразливостей – сканування буде примусово зупинене. Після чого результати сканування буде збережено і згенеровано звіт. Оберемо звіт у вигляді Executive Summary. Можна одразу згенерувати звіт. Також можна обрати параметри в

Settings і в Report Wizard. Згенеруємо звіт і оберем Report Preview для попереднього перегляду. Після чого можна експортувати його у PDF.

Наведем результати даного звіту (рис. 2.5 – рис. 2.6):



**Scan of http://cad.kpi.ua:80/**

**Scan details**

| Scan information | |
|---|---|
| Starttime | 07.06.2016 15:43:50 |
| Finish time | 07.06.2016 17:27:58 |
| Scan time | 1 hours, 44 minutes |
| Profile | High_Risk_Alerts |
| **Server information** | |
| Responsive | True |
| Server banner | Apache |
| Server OS | Unknown |

**Threat level**

acunetix threat level
Level 3: High

**Acunetix Threat Level 3**

One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website.

**Alerts distribution**

| | |
|---|---|
| Total alerts found | 307 |
| High | 1 |
| Medium | 16 |
| Low | 1 |
| Informational | 289 |

Рисунок 2.5

**Executive summary**

| Alert group | Severity | Alert count |
|---|---|---|
| SQL injection | High | 1 |
| AWStats script | Medium | 1 |
| HTML form without CSRF protection | Medium | 2 |
| Slow HTTP Denial of Service Attack | Medium | 1 |
| User credentials are sent in clear text | Medium | 12 |
| Clickjacking: X-Frame-Options header missing | Low | 1 |
| Broken links | Informational | 185 |
| Password type input with auto-complete enabled | Informational | 104 |

Рисунок 2.6

В звіті наведена лише статистика та опис вразливостей. Якщо обрати більш детальний звіт то туди будуть додані загальні/детальні описи вразливостей, рішення по даним вразливостям, а також корисні посилання. Статистика наступна: високий рівень – 1, середній рівень – 16, низький рівень – 1, інформаційна вразливість – 289. Всього – 307.

## 2.3. Результати сканування SSS

Було обрано швидке сканування. Це правило сканує тільки основні порти і вразливості на віддаленій машині крім “DoS tests”, які можуть визвати відмову чи збій в системі, і відключена перевірка паролей через NetBIOS. Звіт було згенеровано у html форматі.

kpi.ua (рис. 2.7)

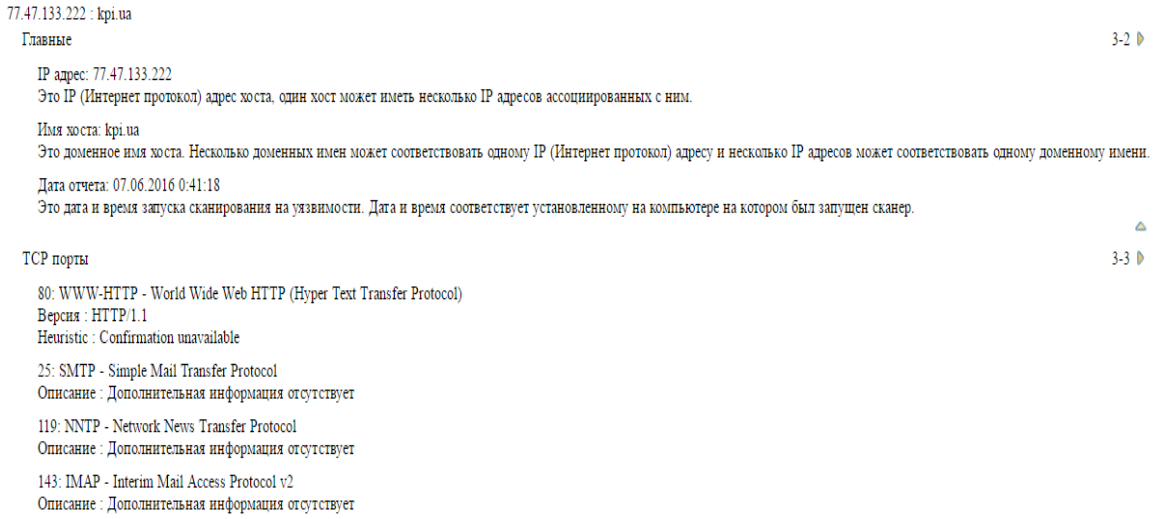



Рисунок 2.7

cad.edu.kpi.ua (рис. 2.8)

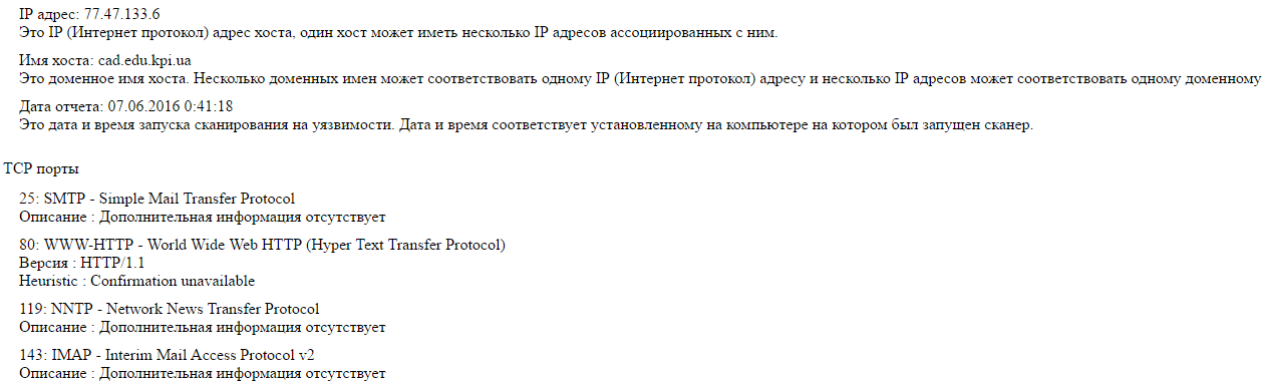



Рисунок 2.8

cad.kpi.ua (рис. 2.9 – рис. 2.10)

212.111.203.242 : cad.kpi.ua 4-1

Главные 4-2

IP адрес: 212.111.203.242
Это IP (Интернет протокол) адрес хоста, один хост может иметь несколько IP адресов ассоциированных с ним.

Имя хоста: cad.kpi.ua
Это доменное имя хоста. Несколько доменных имен может соответствовать одному IP (Интернет протокол) адресу и несколько IP адресов может соответствовать одному доменному имени.

Дата отчета: 07.06.2016 0:41:18
Это дата и время запуска сканирования на уязвимости. Дата и время соответствует установленному на компьютере на котором был запущен сканер.

Аудиты 4-3

DNS Services: ISC BIND 8 Remote Cache Poisoning Vulnerability
Порт : 53
Описание : BIND 8 is prone to a remote cache-poisoning vulnerability because of weaknesses in its random number generator.An attacker may leverage this issue to manipulate cache data, potentially facilitating man-in-the-middle, site-impersonation, or denial-of-service attacks.Versions of BIND from 8.2.0 through to 8.4.7 are vulnerable to this issue.
Уровень риска : Высокий
Как исправить : Upgrade to the current version of bind.
ISC Bind Security : http://www.isc.org/products/BIND/bind-security.html
CVE : CVE-2007-4019
Bugtraq ID : 25459

TCP порты 4-4

21: FTP - File Transfer Protocol [Control]
Баннер : 220 Welcome to CAD FTP Service

25: SMTP - Simple Mail Transfer Protocol
Описание : Дополнительная информация отсутствует

80: WWW-HTTP - World Wide Web HTTP (Hyper Text Transfer Protocol)
Версия : HTTP/1.1
Сервер : Apache
Heuristic : Confirmation unavailable

Рисунок 2.9

22: SSH - SSH (Secure Shell) Remote Login Protocol
Баннер : SSH-2.0-OpenSSH_6.1_hpn13v11 DragonFly-20070328
Версия протокола : 1.99 2.0

119: NNTP - Network News Transfer Protocol
Описание : Дополнительная информация отсутствует

143: IMAP - Interim Mail Access Protocol v2
Баннер : * OK [CAPABILITY IMAP4rev1 LITERAL+ SASL-IR LOGIN-REFERRALS ID ENABLE IDLE STARTTLS AUTH=PLAIN] Dovecot ready.

53: DOMAIN - Domain Name Server
version.bind : 8.4.6-REL-NOESW

6666: IRC-SERV - irc-serv
Баннер : +OK Ready to serve.

Рисунок 2.10

iasa.kpi.ua (рис. 2.11)

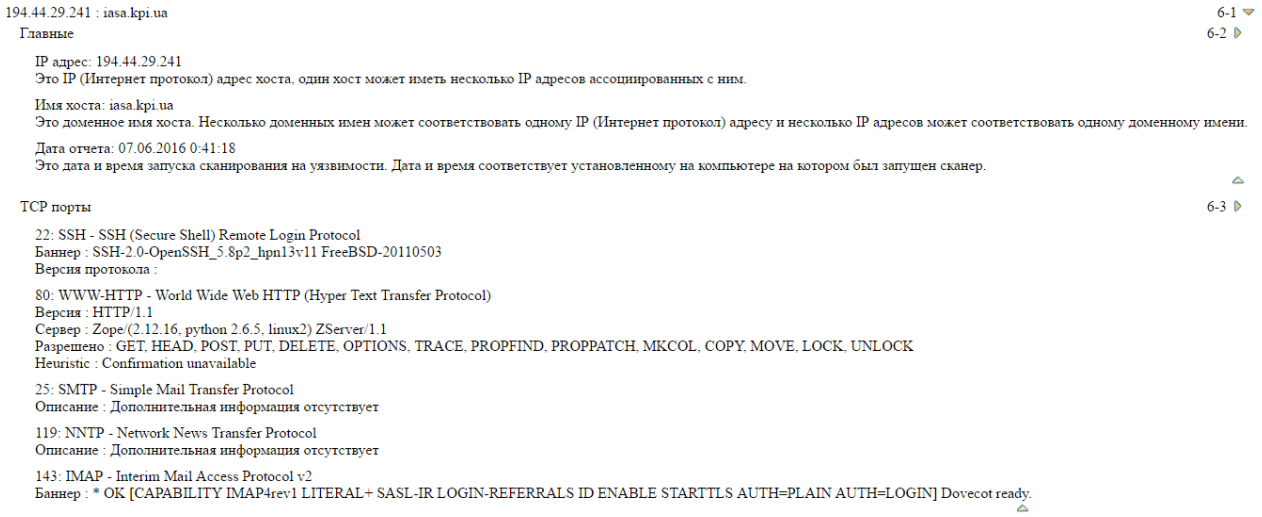



Рисунок 2.11

its.kpi.ua (рис. 2.12)

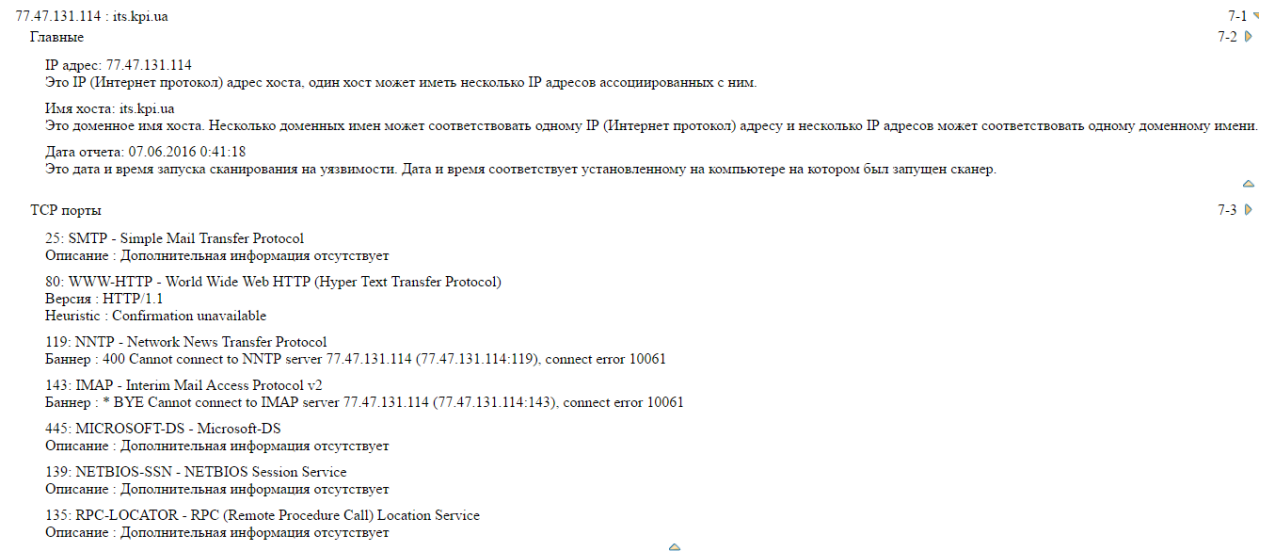


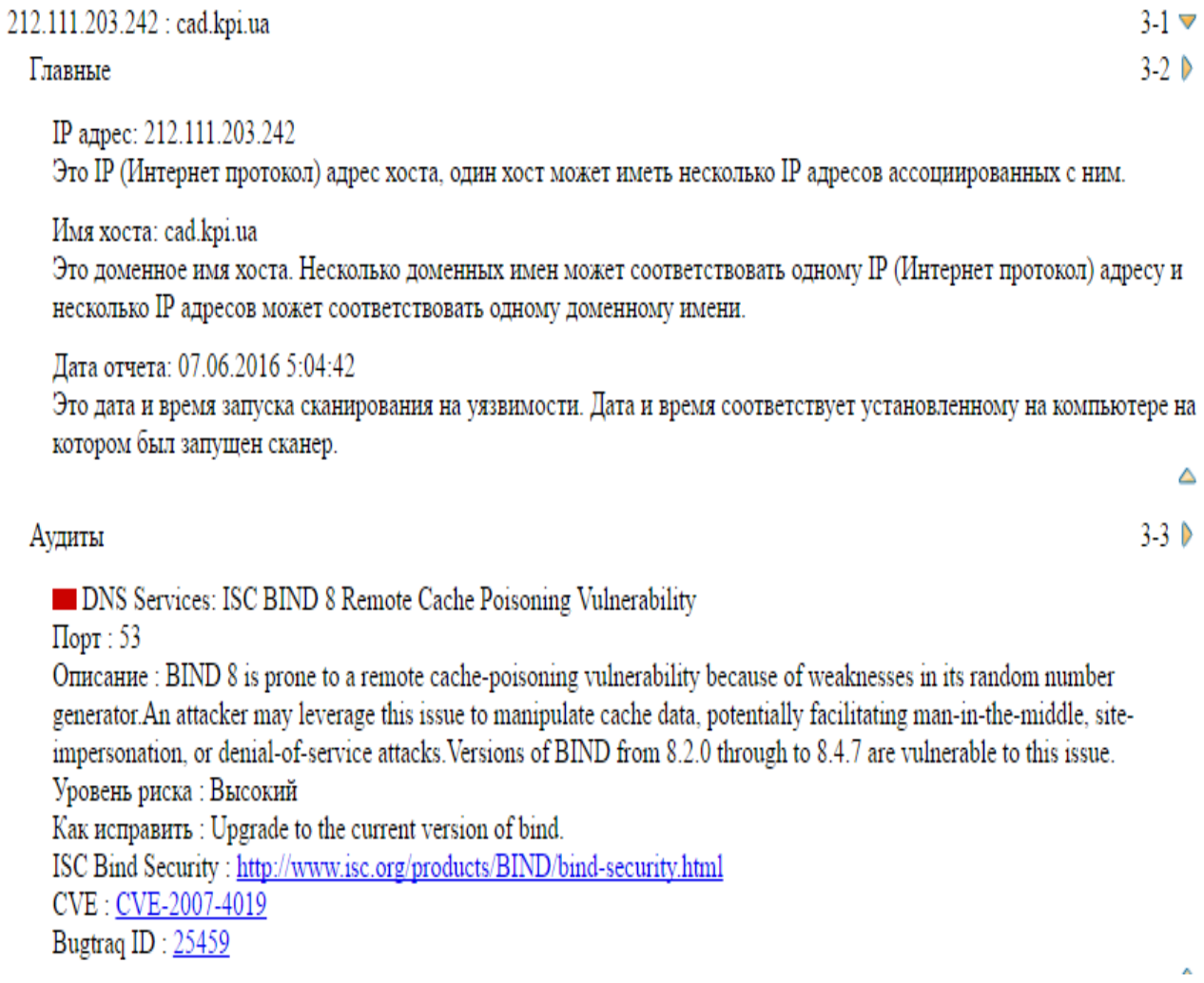

Рисунок 2.12

Як бачимо,вразливість знайшло лише на cad.kpi.ua(1 вразливість з високим ступенем ризику). Для інших ресурсів було відображено лише перелік доступних портів, та загальна інформація.

Проведемо сканування http://cad.kpi.ua/ використовуючи повне сканування. Це правило сканує всі порти (1..65335) і вразливості на віддаленій машині крім “DoS tests”, які можуть визвати відмову чи збій в системі, і відключена перевірка паролей через NetBIOS.

Результати сканування наступні: (звіт згенеровоно в html)

cad.kpi.ua має 1 вразливість з високим ступенем ризику, що співпадає з результатами попереднього сканування. Результати сканування у режимі повного сканування (рис. 2.13 – рис. 2.17).


212.111.203.242 : cad.kpi.ua 3-1

Главные 3-2

IP адрес: 212.111.203.242
Это IP (Интернет протокол) адрес хоста, один хост может иметь несколько IP адресов ассоциированных с ним.

Имя хоста: cad.kpi.ua
Это доменное имя хоста. Несколько доменных имен может соответствовать одному IP (Интернет протокол) адресу и несколько IP адресов может соответствовать одному доменному имени.

Дата отчета: 07.06.2016 5:04:42
Это дата и время запуска сканирования на уязвимости. Дата и время соответствует установленному на компьютере на котором был запущен сканер.

Аудиты 3-3

DNS Services: ISC BIND 8 Remote Cache Poisoning Vulnerability
Порт : 53
Описание : BIND 8 is prone to a remote cache-poisoning vulnerability because of weaknesses in its random number generator.An attacker may leverage this issue to manipulate cache data, potentially facilitating man-in-the-middle, site-impersonation, or denial-of-service attacks.Versions of BIND from 8.2.0 through to 8.4.7 are vulnerable to this issue.
Уровень риска : Высокий
Как исправить : Upgrade to the current version of bind.
ISC Bind Security : http://www.isc.org/products/BIND/bind-security.html
CVE : CVE-2007-4019
Bugtraq ID : 25459

Рисунок 2.13

TCP порты 3-4

21: FTP - File Transfer Protocol [Control]
Баннер : 220 Welcome to CAD FTP Service

22: SSH - SSH (Secure Shell) Remote Login Protocol
Баннер : SSH-2.0-OpenSSH_6.1_hpn13v11 DragonFly-20070328
Версия протокола : 1.99 2.0

25: SMTP - Simple Mail Transfer Protocol
Описание : Дополнительная информация отсутствует

53: DOMAIN - Domain Name Server
version.bind : 8.4.6-REL-NOESW

80: WWW-HTTP - World Wide Web HTTP (Hyper Text Transfer Protocol)
Версия : HTTP/1.1
Сервер : Apache
Heuristic : Confirmation unavailable

110: POP3 - Post Office Protocol - Version 3
Баннер : +OK Dovecot ready.

119: NNTP - Network News Transfer Protocol
Описание : Дополнительная информация отсутствует

143: IMAP - Interim Mail Access Protocol v2
Баннер : * OK [CAPABILITY IMAP4rev1 LITERAL+ SASL-IR LOGIN-REFERRALS ID ENABLE IDLE STARTTLS AUTH=PLAIN] Dovecot ready.

222: RSH-SPX - Berkeley rshd with SPX auth
Баннер : SSH-2.0-OpenSSH_6.2_hpn13v11 FreeBSD-20130515

443: HTTPS - HTTPS (Hyper Text Transfer Protocol Secure) - SSL (Secure Socket Layer)
SSLSubject : C=UA,ST=Kyiv,L=Kyiv,O=CAD,OU=IT,CN=cad.ntu-kpi.kiev.ua,emailAddress=admin@cad.ntu-kpi.kiev.ua
SSLIssuer : C=UA,ST=Kyiv,O=CAD,OU=IT,CN=cad.ntu-kpi.kiev.ua,emailAddress=admin@cad.ntu-kpi.kiev.ua

Рисунок 2.14

SSLVersion : TLSv1
SSlPeerName : cad.ntu-kpi.kiev.ua
SSLCertInfo : Certificate:
Data:
Version: 3 (0x2)
Serial Number:
db:86:43:53:64:e4:45:bd
Signature Algorithm: sha1WithRSAEncryption
Issuer: C=UA, ST=Kyiv, O=CAD, OU=IT, CN=cad.ntu-kpi.kiev.ua/emailAddress=admin@cad.ntu-kpi.kiev.ua
Validity
Not Before: Mar 31 13:07:41 2012 GMT
Not After : Mar 29 13:07:41 2022 GMT
Subject: C=UA, ST=Kyiv, L=Kyiv, O=CAD, OU=IT, CN=cad.ntu-kpi.kiev.ua/emailAddress=admin@cad.ntu-kpi.kiev.ua
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:ac:e7:7b:a0:9d:1b:b3:8c:78:9c:bd:da:4c:4b:
5b:9a:15:02:e1:91:66:fd:6e:1a:1a:10:e0:59:60:
c5:ae:7e:d1:ff:47:b5:ae:1b:35:8e:d0:d7:63:8d:
6c:f4:14:ca:74:b6:93:ad:6f:3e:f1:4b:39:54:97:
05:d3:a1:23:46:36:50:7b:a8:b0:52:20:7e:49:69:
13:7c:fa:20:d9:e3:57:93:35:54:27:05:35:b9:b4:
bf:03:ca:d3:22:aa:9b:f7:48:52:a5:2d:5e:79:ff:
45:33:79:d5:7d:4d:21:f0:74:ab:b7:b1:1f:99:70:
82:06:78:66:1a:a1:4e:f5:03
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Basic Constraints:
CA:FALSE
Netscape Comment:
Certificate issued by CAD Dept. of NTUU KPI

Рисунок 2.15

X509v3 Subject Key Identifier:
C9:E8:E2:A7:77:0A:AA:36:09:7E:32:F8:2A:AF:7C:1F:A5:DD:26:86
X509v3 Authority Key Identifier:
keyid:A3:97:9E:1C:36:13:C4:5F:82:B9:43:D9:81:8C:D1:02:33:9B:A6:A8

X509v3 Subject Alternative Name:
DNS:cad.kiev.ua, DNS:www.cad.kiev.ua, DNS:cad.kpi.ua, DNS:www.cad.kpi.ua, DNS:www.cad.ntu-kpi.kiev.ua, DNS:cad, DNS:cad.cad, DNS:www.cad, DNS:cad.ntu-kpi.kiev.ua
Netscape CA Revocation Url:
http://www.cad.ntu-kpi.kiev.ua/ssl/cacrl.pem
Netscape Base Url:
http://www.cad.ntu-kpi.kiev.ua/ssl/
Netscape Revocation Url:
http://www.cad.ntu-kpi.kiev.ua/ssl/revocation.html
Netscape Renewal Url:
http://www.cad.ntu-kpi.kiev.ua/ssl/renewal.html
Netscape CA Policy Url:
http://www.cad.ntu-kpi.kiev.ua/ssl/policy.html
Signature Algorithm: sha1WithRSAEncryption
a3:d4:8b:71:3e:80:b1:31:36:c7:b5:ab:1d:94:96:33:ea:b3:
cd:0b:7b:d0:3b:39:c5:1f:e7:07:bf:fd:0e:b2:5a:66:b2:39:
1a:30:97:0d:26:55:f4:d7:73:ac:b9:0e:b0:05:c5:07:59:01:
89:5f:ff:ee:75:c9:d5:54:92:4c:ed:c9:26:3c:59:c2:62:b8:
eb:ca:a5:59:74:ac:bd:ec:ef:12:a9:52:94:fa:9c:53:a6:21:
97:8b:ca:45:35:1f:bf:03:99:8a:50:d4:8f:76:c2:c6:b7:6a:
f3:34:61:88:15:94:ce:e1:f9:02:c1:25:cc:37:b3:a7:c2:c6:
6f:0c
SSLChipperName : DHE-RSA-AES256-SHA

465: SSMTP - ssmtp
Описание : Дополнительная информация отсутствует

Рисунок 2.16

563: SNEWS - snews
Описание : Дополнительная информация отсутствует

587: SUBMISSION -
Описание : Дополнительная информация отсутствует

993: IMAPS - Imap4 protocol over TLS/SSL
Описание : Дополнительная информация отсутствует

995: POP3S - Pop3 (Post Office Protocol) over TLS/SSL
Описание : Дополнительная информация отсутствует

1723: PPTP - pptp
Описание : Дополнительная информация отсутствует

2221: UNREG-AB1 - Allen-Bradley unregistered port
Описание : Дополнительная информация отсутствует

2222: UNREG-AB2 - Allen-Bradley unregistered port
Баннер : SSH-2.0-OpenSSH_4.2p1 FreeBSD-19980513

6666: IRC-SERV - irc-serv
Баннер : +OK Ready to serve.

Рисунок 2.17

## 2.4. Порівняння результатів сканувань

Виконаємо порівняння результатів сканування обраними засобами. Для ефективного порівняння використаємо результат сканування cad.kpi.ua. Результати відобразимо у вигляді таблиці.

Таблиця 2.1 – Порівняння результатів сканування cad.kpi.ua обраними засобами

| Назва сканера | Опис результатів |
|---|---|
| XSpider (full) | Вразливості – 3(вразливість сервісів 80/tcp – HTTP та 443/tcp HTTP SSL). Знайдені вразливості дають доступ до переліку директорій на перегляд, а також можливість визначення наявності користувача в системі. Доступна інформація – 19 |
| XSpider (demo) | Вразливості – 16(вразливість сервісів 80/tcp – HTTP та 443/tcp HTTP SSL). Знайдені вразливості пов'язані з міжсайтовим скриптингом та доступом до директорій. Підозра на вразливість – 7(2222/tcp SSH). Розголошення інформації, підвищення привілеїв, підміна даних в log-файлі. Підозра на серйозну вразливість – 2(2222/tcp SSH). Відмова в обслуговуванні та виконання довільного коду. Доступна інформація – 17 |
| Acunetix WVS | високий рівень – 1(SQL ін'єкція), середній рівень – 16(скриптова вразливість, незахищена HTML форма, доступні облікові данні користувача, відмова в обслуговуванні), низький рівень – 1, інформаційна вразливість – 289 |
| SSS | 1 вразливість з високим ступенем ризику (порт 53 – атакуючий має доступ до кеша) |

## 2.5. Створення сторінки Web-сайту з описом роботи обраних сканерів

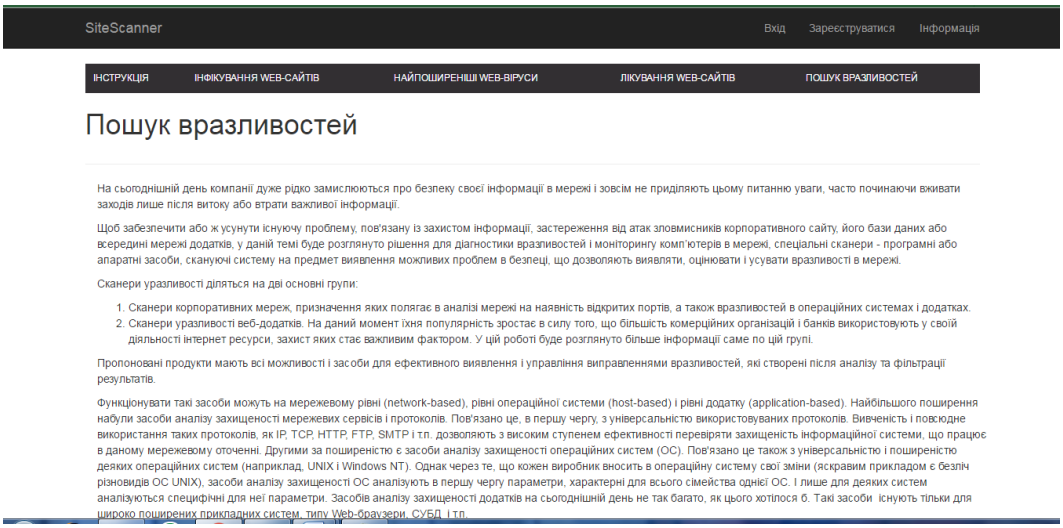
Для збереження опису роботи з обраними сканерами вразливостей мною було створено розділ для сайту bug.kpi.ua. Розділ було створено за допомогою PHP фреймворку Yii 1.1.

Було зроблено записи в  таблицю article (рис. 2.18).

| | | | | | |
|---|---|---|---|---|---|
| Изменить Быстрая правка Копировать Удалить | 37 | Пошук вразливостей | <p>На сьогоднішній день компанії дуже рідко замисл... | 1 | 0 |
| Изменить Быстрая правка Копировать Удалить | 39 | Робота з XSpider | <p>XSpider 7  <st... | 1 | 37 |
| Изменить Быстрая правка Копировать Удалить | 40 | Робота з Acunetix Web Vulnerability Scanner | <p>Acunetix Web Vulnerability Scanner 10.0... | 1 | 37 |
| Изменить Быстрая правка Копировать Удалить | 41 | Робота з Shadow Security Scanner | <p>Shadow Security</stron... | 1 | 37 |

Рисунок 2.18 – Запис в таблицю

Зайшовши на сайт та перейшовши на  сторінку «Інформація» ми потрапляємо на сторінку з інформаційними розділами. Перейдемо до створеного розділу «Пошук вразливостей». Нам буде відображено загальну інформацію про пошук вразливостей (рис. 2.19).




.Рисунок 2.19 – Розділ «Пошук вразливостей»

Даний розділ містить 3 підрозділи (у вигляді випадаючого меню) в кожному з яких описана робота зі сканером вразливостей. Підрозділ вигляда наступним чином (рис. 2.20):

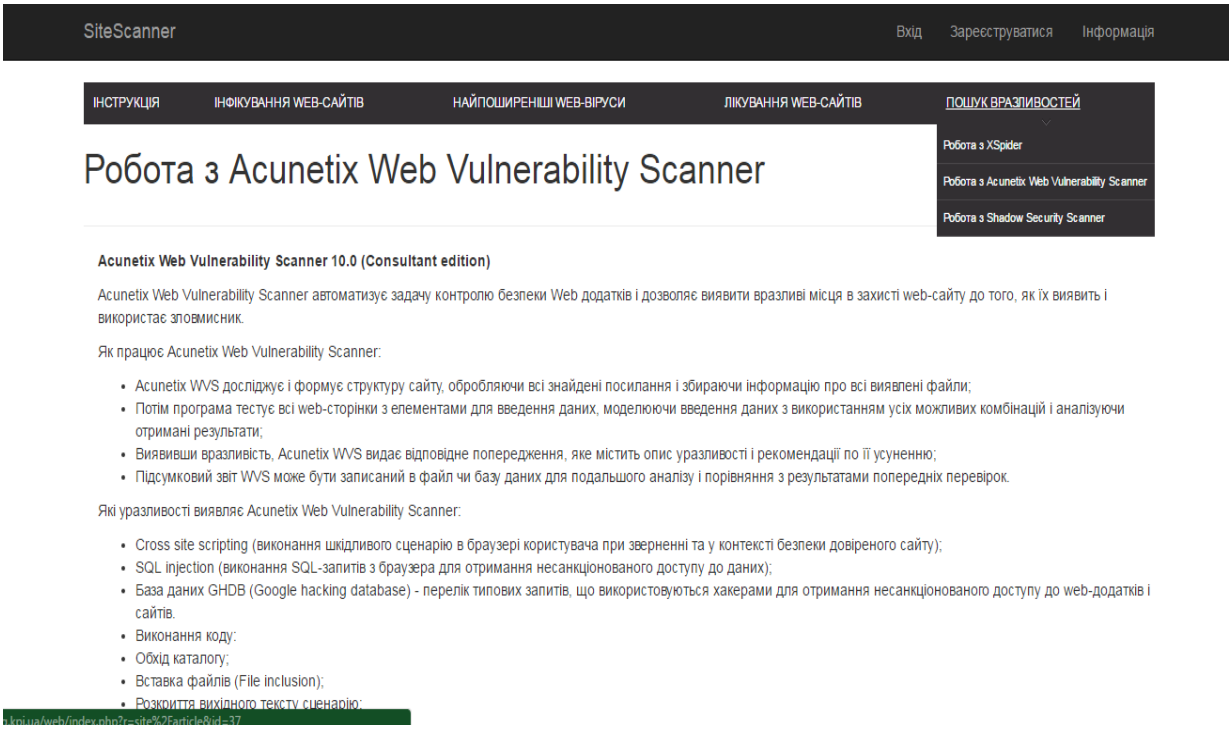



Рисунок 2.20 – Підрозділ

## 2.6. Висновки до розділу 2

В даному розділі було розглянуто результати сканувань для обраних засобів. Після чого на прикладі сканування cad.kpi.ua було проведено порівняння результатів сканувань. Кожен сканер надає інформацію про знайдені вразливості та шляхи боротьби з ними. Через дуже громісткі звіти було наведено лише основні дані.

На основі табл. 2.1 можна зробити висновки про наявність на cad.kpi.ua вразливостей високого та середнього рівнів, які дають можливість проводити атаки на даний сайт.

Найбільше вразливостей було знайдено за допомогою Acunetix WVS, що свідчить про високу якість даного сканера (хоча швидкість сканування досить повільна).

Для опису роботи зі сканерами було створено інформаційний розділ «Пошук вразливостей» Web-сайту bug.kpi.ua. В розділ додано підрозділи що описують роботу зі сканерами.

Отримані результати можна використати для покращення роботи просканованих Web-сайтів.

# 3. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка основних характеристик програмного продукту, який являє собою Web-сайт «Сучасні засоби пошуку вразливостей Web-сайтів та серверів та аналіз їх можливостей і практичного використання». Сайт буде створюватися за допомогою Web-фреймворку.

Фінальний сайт призначений для надання інформації «Сучасні засоби пошуку вразливостей Web-сайтів та серверів та аналіз їх можливостей і практичного використання». Сайт повинен однаково добре відображатися у браузерах, таких як Google Chrome, Mozilla Firefox, Opera та інші.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

– визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам: ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат.

– для кожної функції визначаються повні річні витрати й кількість робочих часів.

– для кожної функції наоснові оцінок попереднього пункту визначається кількісна характеристика джерел витрат.

– після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

## 3.1. Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

– програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;

– забезпечувати цілодобовий доступ клієнтів до інформації «Сучасні засоби пошуку вразливостей Web-сайтів та серверів та аналіз їх можливостей і практичного використання»;

– забезпечувати зручність і простоту взаємодії з користувачем або з адміністратором програмного забезпечення;

– передбачати мінімальні витрати на впровадження програмного продукту.

### 3.1.1. Обґрунтування функцій програмного продукту

Головна функція $F_0$ – обрання програмного продукту для встановлення на фізичний сервер. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

- $F_1$ – вибір Web-фреймворку;
- $F_2$ – вибір типу підключення до бази даних;
- $F_3$ – вибір серверу на який буде встановлюватися Web-фреймворк.
- Кожна з основних функцій може мати декілька варіантів реалізації.

Функція $F_1$:

а) Web-фреймворк Yii 1.1;

б) Web-фреймворк Symfony;

Функція $F_2$:

а) підключення MySQLi;

б) підключення MySQL.

Функція $F_3$:

а) сервер PHP Apache ;

б) сервер PHP Denwer.

### 3.1.2. Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 3.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 3.1).

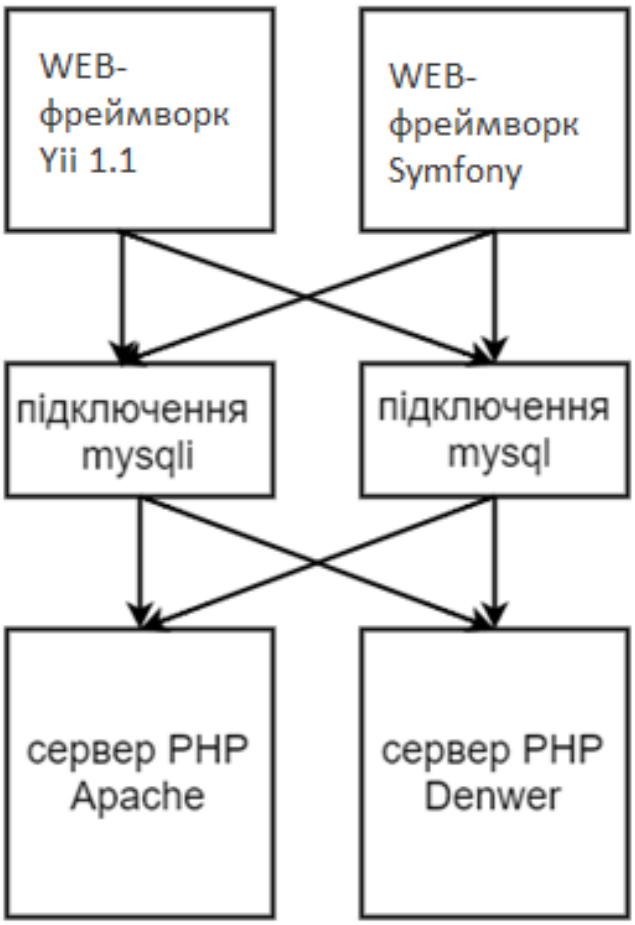



Рисунок 3.1 – Морфологічна карта

Морфологічна карта відображає всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

На основі аналізу позитивно-негативної матриці робимо висновок, що при обрані програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Таблиця 3.1 – Позитивно-негативна матриця

| **Основні функції** | **Варіанти реалізації** | **Переваги** | **Недоліки** |
|---|---|---|---|
| *F1* | *А* | Введення і валідація форм, аутентифікація і авторизація, використання AJAX і інтеграція з jQuery, генерація базового PHP-коду для CRUD-операцій (скаффолдинг), підтримка тем оформлення для їх легкої зміни, можливість підключення сторонніх бібліотек, міграції бази даних. | Не дуже потужний роутинг і наявність великого кількість віджетів, на настройку яких витрачається багато часу. |
| | *Б* | Підтримує безліч баз даних (MySQL,PostgreSQL, SQLite, або будь-яка інша PDO-сумісна СУБД) , вбудовані класи для роботи з email, гнучка система шаблонів у поданні,підтримка французького спонсора Sensio. | Складний в освоєнні, підходить тільки для великих проектів,відсутність російської документації. |
| *F2* | А | Більша оптимізація ніж у аналога, більша швидкість доступу. | Неможливе з’єднання з старими версіями баз даних. |
| | Б | Підтримка старих версій MySQL баз даних та PHP. | Менша швидкість. |
| *F3* | *А* | Можливість використання різних версій PHP. | PHP та MySQL необхідно встановлювати додатково. |
| | *Б* | Повна комплектація з PHP та MySQL. | Неможливість використання інших версій PHP та MySQL. |

Функція *F1:*

Обидва Web-фреймворки доволі розвинені та мають багато розширень та доповнень, тому обидва варіанти А і Б гідні розгляду.

Функція *F2:*

У роботі нам важлива підтримка старих версій PHP, тому обираємо варіант Б.

Функція *F3:*

Так як обрані фреймворки використовують різні версії PHP для своєї роботи, необхідно обрати сервер Apache для більш тонкого налаштування серверу. Отже варіант А.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1а – F2б – F3а
2. F1б – F2б – F3а

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

## 3.2. Обґрунтування системи параметрів ПП

### 3.2.1 Опис параметрів

На підставі даних про основні функції, що повинен мати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– *X1* – час відповіді на запит клієнта;

– *X2* – кількість модулів на сайті;

– *X3* – об’єм даних, що передається через Інтернет;

– *X4* – об’єм оперативної пам’яті, що використовується.

### 3.2.2. Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 3.2

Таблиця 3.2 – Основні параметри ПП

| **Назва Параметра** | **Умовні позначення** | **Одиниці виміру** | **Значення параметра** | | |
|---|---|---|---|---|---|
| | | | **гірші** | **середні** | **Кращі** |
| Час відповіді на запит клієнта | X1 | С | 10 | 4 | 2 |
| Кількість модулів на сайті | X2 | Кількість модулів | 3 | 5 | 10 |
| Об’єм даних, що передається через Інтернет | X3 | Кб | 2000 | 800 | 500 |
| Об’єм оперативної пам’яті, що використовується | X4 | Мб | 150 | 100 | 70 |

За даними таблиці 3.2 будуються графічні характеристики параметрів – рис. 3.2 – рис. 3.5.

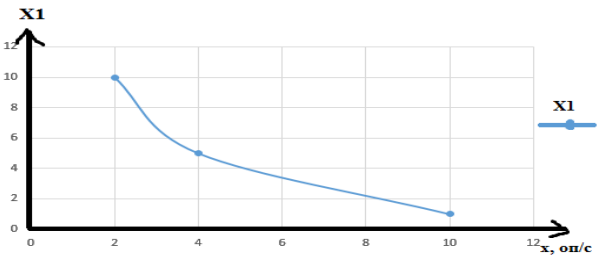



Рисунок 3.2 – X1, завантаженість підключення до Інтернету

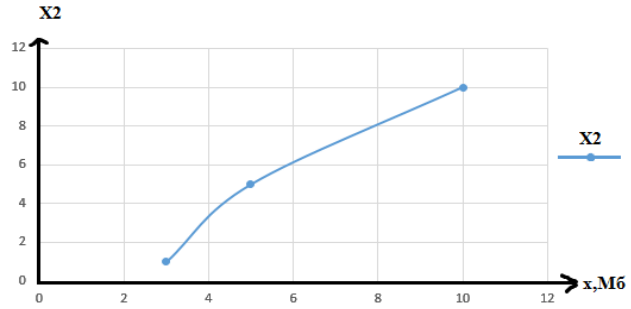



Рисунок 3.3 – X2, об’єм оперативної пам’яті, що використовується

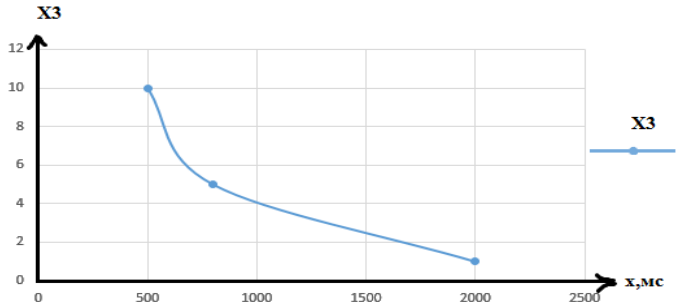



Рисунок 3.4 – X3, час необхідний на підключення до медіа серверу

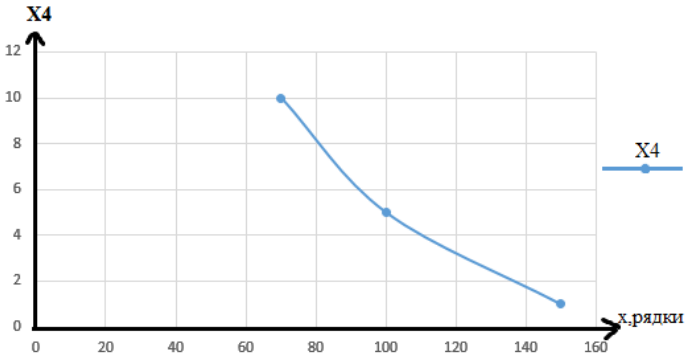



Рисунок 3.5 – X4, кількість функцій, доступних для віддаленого доступу

### 3.2.3. Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – вибір

програмного продукту, який має найбільш зручний та функціональний набір можливостей при низькій необхідності у ресурсах.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

– визначення рівня значимості параметра шляхом присвоєння різних рангів;

– перевірку придатності експертних оцінок для подальшого використання;

– визначення оцінки попарного пріоритету параметрів;

– обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 3.3.

Таблиця 3.3 – Результати ранжування параметрів

| Позначення параметра | Назва параметра | Одиниці виміру | Ранг параметра за оцінкою експерта | | | | | | | Сума рангів $R_i$ | Відхилення $\Delta_i$ | $\Delta_i^2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | |
| *X1* | Час відповіді на запит клієнта | С | 1 | 2 | 1 | 2 | 1 | 1 | 1 | 9 | -8,5 | 72,25 |
| *X2* | Кількість модулів на сайті | Кількість модулів | 4 | 3 | 4 | 3 | 4 | 4 | 3 | 25 | 7,5 | 56,25 |
| *X3* | Об’єм даних, що передається через Інтернет | Кб | 2 | 1 | 2 | 1 | 2 | 3 | 2 | 13 | -4,5 | 20,25 |

Таблиця 3.3 – Результати ранжування параметрів (продовження)

| Позначення параметра | Назва параметра | Одиниці виміру | Ранг параметра за оцінкою експерта | | | | | | | Сума рангів $R_i$ | Відхилення $\Delta_i$ | $\Delta_i^2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | |
| *X4* | Об’єм оперативної пам’яті, що використовується | Мб | 3 | 4 | 3 | 4 | 3 | 2 | 4 | 23 | 5,5 | 30,25 |
| | Разом | | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 70 | 0 | 179 |

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70, \qquad (3.1)$$

де $N$ – число експертів, $n$ – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17{,}5 \qquad (3.2)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T \qquad (3.3)$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 179. \qquad (3.4)$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12 \cdot 179}{7^2(4^3-4)} = 0{,}73 > W_k = 0{,}67 \qquad (3.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 3.4.

Таблиця 3.4 – Попарне порівняння параметрів

| Параметри | Експерти | | | | | | | Кінцева оцінка | Числове значення |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | |
| X1 і X2 | > | > | > | > | > | > | > | > | 1.5 |
| X1 і X3 | > | < | > | < | > | > | > | > | 1,5 |
| X1 і X4 | > | > | > | > | > | > | > | > | 1,5 |
| X2 і X3 | < | < | < | < | < | < | < | < | 0,5 |
| X2 і X4 | < | > | < | > | < | < | > | < | 0.5 |
| X3 і X4 | > | > | > | > | > | < | > | > | 1,5 |

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, $a_{ij}$ визначається по формулі:

$$a_{ij} = \begin{cases} 1{,}5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases} \qquad (3.6)$$

З отриманих числових оцінок переваги складемо матрицю $A = \| a_{ij} \|$. Для кожного параметра зробимо розрахунок вагомості $K_{ві}$ за наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \text{де } b_i = \sum_{i=1}^{N} a_{ij}. \qquad (3.7)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \text{де } b_i' = \sum_{i=1}^{N} a_{ij} b_j. \qquad (3.8)$$

Як видно з таблиці 3.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 3.5 – Розрахунок вагомості параметрів

| Параметри$x_i$ | Параметри $x_j$ | | | | Перша ітер. | | Друга ітер. | | Третя ітер. | |
|---|---|---|---|---|---|---|---|---|---|---|
| | X1 | X2 | X3 | X4 | $b_i$ | $K_{\text{ві}}$ | $b_i^1$ | $K_{\text{ві}}^1$ | $b_i^2$ | $K_{\text{ві}}^2$ |
| X1 | 1,0 | 1,5 | 1,5 | 1,5 | 5,5 | 0,344 | 21,25 | 0,360 | 77,875 | 0,361 |
| X2 | 0,5 | 1,0 | 0,5 | 0,5 | 2,5 | 0,156 | 9,25 | 0,157 | 34,125 | 0,158 |
| X3 | 0,5 | 1,5 | 1,0 | 1,5 | 4,5 | 0,281 | 16,25 | 0,275 | 59,125 | 0,274 |
| X4 | 0,5 | 1,5 | 0,5 | 1,0 | 3,5 | 0,219 | 12,25 | 0,208 | 44,875 | 0,207 |
| Всього: | | | | | 16 | 1 | 59 | 1 | 216 | 1 |

## 3.3. Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X1*(час відповіді на запит клієнта) та *X4* (Об’єм оперативної пам’яті, що використовується) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X2*(Кількість модулів на сайті) буде найкращим у випадку обрання у F2 варіанта Б і становитиме 5, для варіанту А це значення буде 8.

Абсолютне значення параметра X3(Об’єм даних, що передається через Інтернет) буде найкраще при обрані варіанту А 600, а при обрані варіанту Б воно буде середнім 1500.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 3.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j}\,, \qquad (3.9)$$

де $n$ – кількість параметрів;$K_{вi}$– коефіцієнт вагомості $i$–го параметра;$B_i$ – оцінка $i$–го параметра в балах.

Таблиця 3.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

| Основні функції | Варіат реалізації функції | Параметри | Абсолютне значенняпараметра | Бальна оцінка параметра | Коефіцієнт вагомості параметра | Коефіцієнт рівня якості |
|---|---|---|---|---|---|---|
| F1 | А | X2 | 5 | 5 | 0,158 | 0,79 |
| | | X3 | 600 | 7 | 0,274 | 1,918 |
| | Б | X2 | 8 | 8 | 0,158 | 1,264 |
| | | X3 | 1500 | 2,5 | 0,274 | 0,685 |
| F2 | Б | X1 | 3 | 6,5 | 0,361 | 2,35 |
| F3 | А | X4 | 100 | 5 | 0,207 | 1,035 |

За даними з таблиці 3.6 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \qquad (3.10)$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 0{,}79 + 1{,}918 + 2{,}35 + 1{,}035 = 6{,}093$$

$$K_{K2} = 1{,}264 + 0{,}685 + 2{,}35 + 1{,}035 = 5{,}334$$

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

## 3.4. Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Обидва варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Програмна реалізація рішення;

Обидва завдання за ступенем новизни відноситься до групи В. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується нормативно-довідкова інформація, а завдання 2 використовує банк даних.

Проведемо розрахунок норм часу для виконання кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \qquad (3.11)$$

де $T_P$ – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт; $K_{СК}$ – коефіцієнт на складність вхідної інформації; $K_M$ – коефіцієнт рівня мови програмування; $K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру ступеня новизни В та групи складності алгоритму 1, трудомісткість дорівнює: $T_P$ =43 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 0.81. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для обох завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці кожного з завдань використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.8. Тоді, за формулою 3.11, загальна трудомісткість першого завдання дорівнює:

$T_1 = 43 \cdot 0.81 \cdot 0.8 = 27.864$ людино-днів.

Проведемо аналогічні розрахунки для другого завдання.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни В), тобто $T_P$ =12 людино-днів, $K_П$ =0.5, $K_M$ = 1 $K_{СК}$ = 1, $K_{СТ}$ =0.8, $K_{СТ.М}$ = 1.6 (однакові для обох варіантів реалізації):

$T_2 = 12 \cdot 0{,}5 \cdot 0.8 \cdot 1.6 = 7{,}68$ людино-днів.

Складаємо трудомісткість відповідних завдань для отримання загальної трудомісткості реалізації ПП, яка буде однакова для обох варіантів тому, що обидва фреймворки використовують мову програмування PHP і їх вибір впливає лише на зручність роботи розробника, а не на вартість кінцевого ПП:

Т = (27.864+7.68) · 8 = 284.352 людино-годин;

В розробці беруть участь два програмісти з окладом 7000 грн. Визначимо зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} грн., \qquad (3.12)$$

де М – місячний оклад працівників;$T_m$ – кількість робочих днів тиждень;$t$ – кількість робочих годин в день.

$$С_Ч = \frac{7000 + 7000}{2 \cdot 21 \cdot 8} = 41.67 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$$С_{ЗП} = С_Ч \cdot Т_i \cdot К_Д, \qquad (3.13)$$

де $С_Ч$– величина погодинної оплати праці програміста;$Т_i$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробника:

$$С_{ЗП} = 41.67 \cdot 284.352 \cdot 1.2 = 14218.74 \text{ грн.}$$

Відрахування на єдиний соціальний внесок становить 22%:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 14218.74 \cdot 0.22 = 3128.12 \text{ грн.} \qquad (3.14)$$

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 9000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 7000 \cdot 0.2 = 16800 \text{ грн.} \qquad (3.15)$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 16800 \cdot (1 + 0.2) = 20160 \text{ грн.} \qquad (3.16)$$

Відрахування на єдиний соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 20160 \cdot 0.22 = 4435.2 \text{ грн.} \qquad (3.17)$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 12000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 10000 = 2875 \text{ грн.}, \quad (3.18)$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 10000 \cdot 0.05 = 575 \text{ грн.}, \quad (3.19)$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В, \quad (3.20)$$

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

$$Т_{ЕФ} = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4 \text{ годин}$$

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706.4 \cdot 0.5 \cdot 0.2 \cdot 2.0218 = 345.00 \text{ грн.}, \quad (3.21)$$

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнт зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 10000 \cdot 0.67 = 6700 \text{ грн.} \quad (3.22)$$

Тоді річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н \quad (3.23)$$

$$С_{ЕКС} = 20160 + 4435.2 + 2875 + 575 + 345 + 6700 = 35090.2 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 35090.2/1706.4 = 20.56 \text{ грн/час.} \quad (3.24)$$

Оскільки в даному випадку всі роботи, які пов’язані з розробкою програмного продукту, ведуться на ЕОМ, витрати на оплату машинного часу складає:

$$С_М = С_{М\text{-}Г} \cdot Т \qquad (3.25)$$

$$С_М = 20.56 \cdot 284.352 = 5846.27 \text{ грн.};$$

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0.67 \qquad (3.26)$$

$$С_Н = 14218.74 \cdot 0.67 = 9526.56 \text{ грн.};$$

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н \qquad (3.27)$$

$$С_{ПП} = 14218.74 + 3128.12 + 11692.54 + 9526.56 = 38565.96 \text{ грн.};$$

### 3.4.1. Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕРj} = К_{Кj} / С_{Фj}, \qquad (3.28)$$

$$К_{ТЕР1} = 6.093 / 38565.96 = 0.158 \cdot 10^{-3};$$

$$К_{ТЕР2} = 5.334 / 38565.96 = 0.138 \cdot 10^{-3};$$

Оскільки обидва варіанти однаково трудомісткі, найбільш ефективним є перший варіант реалізації програми(через більший рівень якості) з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 0.158 \cdot 10^{-3}$.

## 3.5. Висновки до розділу 3

В даному розділі було проведено функціонально-вартісний аналіз програмного продукту, що буде створено. Процес аналізу складався з двох частин.

У першій проведено дослідження програмного продукту з технічної точки зору: були визначені основні параметри, що повинні бути головними при обранні кращої реалізації. На основі отриманих значень параметрів, оцінок експертів було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

У другій частині виконувалися обрахунки вартості розробки ПП з урахуванням витрат на заробітні плати, електроенергії, накладні витрати. Трудомісткість обох варіантів реалізації вважалась однаковою, адже різниця полягала лише у виборі фреймворку (обидва фреймворки використовують мову програмування PHP).

Після виконання функціонально-вартісного аналізу програмного продукту що розроблюється, можна зробити висновок, що перший варіант є найбільш оптимальним для реалізації (через більше значення коефіцієнту технічного рівня). Його показник техніко-економічного рівня якості $К_{ТЕР1}= 0.158 \cdot 10^{-3}$;

Цей варіант реалізації програмного продукту має такі параметри:

– Web-фреймворк Yii 1.1;

– Підключення MySQL;

– Сервер Apache.

Даний варіант реалізації є найбільш продуктивним за рахунок використання більш оптимального для проекту фреймворку та сервера, а також надає користувачу зручний інтерфейс, непоганий функціонал і швидкодію.

# ВИСНОВКИ

В даній роботі були розглянуті основні вразливості Web-сайтів та серверів і наведено їх приклади. Також було розглянуто ієрархію та рівні захисту Web-серверів.

Далі були розглянуті механізми роботи засобів пошуку вразливостей. Після чого були вибрані поширені сканери вразливостей для подальшого аналізу їх можливостей, а саме:

- XSpider;
- Acunetix WVS;
- Shadow Security Scanner.

Наступним кроком було проведення аналізу основних можливостей обраних сканерів, а також вивчено їх функціонал. Результати порівнянь можливостей та функціоналу обраних сканерів були наведені у вигляді таблиць.

За допомогою обраних засобів була проведена перевірка визначеного переліку хостів на наявність вразливостей, а саме:

- kpi.ua;
- cad.kpi.ua;
- cad.edu.kpi.ua;
- iasa.kpi.ua;
- its.kpi.ua.

Кожен сканер представив детальний опис знайдених вразливостей, а також запропонував можливі рішення та перелік корисних посилань. Також, по результатам роботи обраних сканерів були  згенеровані звіти. Порівняння результатів сканування для обраних засобів було виконано у вигляді таблиці, яка містить в собі статистику та опис знайдених вразливостей .

Найефективніше себе показав Acunetix WVS. Сканування даним засобом займає дуже багато часу, але результати перевершують очікування. Даний програмний засіб має багато вбудованих інструментів для ретельного сканування на найрізноманітніші вразливості. При скануванні Acunetix створює детальну карту сайту.  Карта сайту містить в собі повний опис його структури та вмісту (файлів, посилань тощо.). Крім того Acunetix має дуже гнучку систему для створення звітної документації. За допомогою генератора звітів можна обрати різні види звітів та ступені їх деталізації. Також даний сканер має можливість створювати звіти згідно поширених міжнародних стандартів.

Отримані результати сканувань можна легко використати для покращення роботи просканованих Web-сайтів, так як вони уже містять опис причин вразлвостей та шляхи боротьби з ними. Усунення знайденних вразливостей неодмінно підвищить рівень безпеки перевірених сайтів.

Далі за допомогою PHP фреймворку Yii 1.1 мною було створений інформаційний розділ для сайту bug.kpi.ua – «Пошук вразливостей». В цьому розділі були описані основні механізми роботи сканерів пошуку вразливостей, а також робота з обраними сканерами у вигляді підрозділів.

В останньому розділі було проведено функціонально-вартісний аналіз програмного продукту, що було створено. Процес аналізу складався з двох частин.

У першій проведено дослідження програмного продукту з технічної точки зору, а у другій частині виконувалися обрахунки вартості розробки ПП з урахуванням витрат на заробітні плати, електроенергії, накладні витрати.

На основі отриманих результатів було обрано найбільш продуктивний варіант для реалізації програмного продукту, за рахунок використання найбільш оптимальних параметрів, обраних для створюваного ПП.

# ПЕРЕЛІК ПОСИЛАНЬ


1. Shadow Security Scanner [Електронний ресурс] – Режим доступу до ресурсу: http://www.safety-lab.com/en/products/securityscanner.htm.
2. XSpider [Електронний ресурс] – Режим доступу до ресурсу: http://www.ptsecurity.ru/products/xspider/.
3. Сканер уязвимостей XSpider 7 [Електронний ресурс] – Режим доступу до ресурсу: http://www.ixbt.com/soft/xspider7.shtml#int.
4. Web Vulnerability Scanner v10 Product Manual [Електронний ресурс] – Режим доступу до ресурсу: http://www.acunetix.com/resources/wvsmanual.pdf
5. Acunetix Web Vulnerability Scanner [Електронний ресурс] – Режим доступу до ресурсу: http://www.securitylab.ru/software/266415.php
6. Дослідження вразливостей Web-сайтів та методів їх усунення. [Електронний ресурс] – Режим доступу до ресурсу: http://phone.kpi.ua/wpcontent/uploads/2014/06/4.pdf
7. Как работает сканер безопасности? [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/securities/scaner.shtml
8. Жуков Ю.В. Основы веб-хакинга. Нападение и за- щита / Юрий Викторович Жуков, 2012. – 206 с
9. Захист веб-додатків [Електронний ресурс]. – 2010. – Режим доступу до ресурсу:http://www.ereading.club/bookreader.php/1012355/DJ-Andrey-sXe__Zaschita_veb-prilozheniy.html.
10. Khan Khaled M. Managing Web Service Quality: Measuring Outcomes and Effectiveness. / Khaled M. Khan. – IGI Global, 2008. – 418 p.
11. Menaske D. Productivity of Web-Services. Analysis, Assessment and Planning / D. Menaske, V. Almeyda. – SPb: «DiaSoftJUP» Ltd., 2003. – 408 p.

12. Використання PHP фреймворків в розробці сайту [Електронний ресурс] – Режим доступу до ресурсу: http://ukrbukva.net/page,5,39718-Ispol-zovanie-PHP-freiymvorkov-v-razrabotke-saiyta.html.
13. Полное руководство по Yii [Електронний ресурс] – Режим доступу до ресурсу: http://www.yiiframework.com/doc/guide/1.1/ru/index.
14. В.Є. Богданюк . Методичні вказівки до виконання організаційно-економічного розділу дипломних проектів / В.Є. Богданюк, К.В. Березовський, В.П. Пашін та ін. - К.: НТУУ "КПІ", 1999. — 66 с.
15. Пашин В.П. Функционально-стоимостный анализ конструкторско-технологических решений. / Пашин В.П.: РДЭНТП «Знание» УССР, 1989. - 22 с.