Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня магістра

студента *Абрамова Іллі Сергійовича*

академічної групи *125м-22-2*

спеціальності *125 Кібербезпека*

спеціалізації[1]

за освітньо-професійною програмою *Кібербезпека*

на тему *Розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга*

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	д.т.н., проф. Корнієнко В.І.			
розділів:				
спеціальний	д.т.н., проф. Корнієнко В.І.			
економічний	к.е.н., доц. Пілова Д.П.			
Рецензент				
Нормоконтролер	ст. викл. Мєшков В.І.			

Дніпро
2023

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
_____ д.т.н., проф. Корнієнко В.І.

«_____»_____20___року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня магістра

студенту *Абрамову Іллі Сергійовичу* **академічної групи** *125м-22-2*
(прізвище ім'я по-батькові) (шифр)

спеціальності *125 Кібербезпека*

за освітньо-професійною програмою *Кібербезпека*

на тему *Розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга*

затверджену наказом ректора НТУ «Дніпровська політехніка» від _____ № ____

Розділ	Зміст	Термін виконання
Розділ 1	Стан питання. Постановка задачі	02.11.2023
Розділ 2	Спеціальна частина	16.11.2023
Розділ 3	Економічна частина	30.11.2023

Завдання видано _____ Корнієнко В.І.
(підпис керівника) (прізвище, ініціали)

Дата видачі: _____

Дата подання до екзаменаційної комісії: _____

Прийнято до виконання _____ Абрамов І.С.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка складається зі 113 сторінок, 32 рис., 19 табл., 6 додатків, 12 джерел.

Мета кваліфікаційної роботи: розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга для підвищення захисту від автоматизованого виконання зловмисних дій WEB-роботами, що можуть загрожувати конфіденційності, доступності чи цілісності інформації.

У розділі «Стан питання. Постановка задачі» була розглянута класифікація атаки на WEB-сервери, проведений поглиблений аналіз засобів протидії автоматизованому виконанню зловмисних дій на WEB-серверах.

У спеціальній частині був проведений аналіз загроз WEB-серверів та обраний профіль захищеності, узагальнено переваги та недоліки кожного з видів зворотного тесту Тюрінга та розглянуті можливі способи їх обходу. Виконана розробка системи тестування користувачів, яка ґрунтується на зворотному тесті Тюрінга.

В економічній частині виконано розрахунок витрат на розробку програмного забезпечення. Розраховано передбачений рівень збитків від атак на WEB-сервери.

Наукова новизна полягає у вирішенні проблеми автоматизованого виконання зловмисних дій за допомогою зворотного тесту Тюрінга, шляхом створення вдосконаленої системи тестування WEB-користувачів.

ЗВОРОТНИЙ ТЕСТ ТЮРІНГА, CAPTCHA, АВТОМАТИЗАЦІЯ, WEB-СЕРВЕР, ЗАГРОЗИ, WEB-РОБОТИ, СИСТЕМА ТЕСТУВАННЯ.

THE ABSTRACT

Explanatory notes consist of 113 pages, 32 pictures, 19 tables, 6 appendices, 12 sources.

The purpose of qualification work is to develop the system for testing users of WEB servers using the reverse Turing test in order to increase defense from automated malicious activities from WEB-robots that can sufficiently threaten the confidentiality, availability and integrity of information.

The section «Question status. Statement of a task» includes the consideration of classification of attacks on WEB-servers, and in-depth analysis of main features to counteract automated malicious activities.

The special part consists of analysis of threats for WEB-servers and selection of the security profile. This section summarizes the advantages and disadvantages for each type of the reverse Turing test, including possible ways to avoid those disadvantages. The system to test users, which is based on a reverse Turing test, was developed.

The economic section includes calculation of the total amount of costs to create software and provided amount of damage from attacks on the WEB-server.

Scientific novelty lies in addressing the problem of automated execution of malicious acts by reverse Turing test, by an improved testing system users.

REVERSE TURING TEST, CAPTCHA, AUTOMATION, WEB-SERVER, THREATS, WEB-ROBOTS, TESTING SYSTEM.

СПИСОК УМОВНИХ СКОРОЧЕНЬ

ПЗ – програмне забезпечення;

СКБД – система керування базами даних;

ОС – операційна система;

НД – нормативний документ;

ТЗІ – технічний захист інформації;

АС – автоматизована система;

КС – комп'ютерна система;

КЗЗ – комплекс засобів захисту;

ПЗП – постійний запам'ятовуючий пристрій;

КСЗІ – комплексна система захисту інформації;

ЕОМ – електрона обчислювальна машина;

ВДТ – відеотермінал;

ЕПТ – електронно-променева трубка;

АСКП – автоматизована система керування підприємством;

CAPTCHA – Completely Automatic Public Turing Test to Tell Computers and Humans Apart;

WEB – World Wide Web;

URL – Uniform Resource Locator;

HTTP – HyperText Transfer Protocol;

E-Mail – Electronic Mail;

LDAP – Lightweight Directory Access Protocol;

SQL – Structured Query Language;

ISO – International Organization for Standardization;

ANSI – American National Standards Institute;

SSI – Server-site Includes;

XPath – XML Path Language;

XML – eXtensible Markup Language;

XQuery – XML Query Language;

CGI – Common Gateway Interface;

PHP – Hypertext Preprocessor;

ASP – Active Server Pages;

GD – GD Graphics Library;

OCR – Optical Character Recognition;

IP – Internet Protocol;

GIF – Graphics Interchange Format;

CSS – Cascading Style Sheets;

XHTML – Extensible Hypertext Markup Language;

DoS – Denial of Service;

JS – JavaScript;

phpBB – PHP Bulleting Board;

ID – identifier;

W3C – World Wide Web Consortium.

ЗМІСТ

ВСТУП

Швидкий ріст популярності електронних засобів комунікації, файлообмінних сервісів, у тому числі електронної пошти, форумів, блогів, соціальних мереж і засобів обміну миттєвими повідомленнями, привів до потоку несанкціонованих масових розсилок, що з кожним роком збільшується, захопленню облікових записів користувачів на різних ресурсах тощо.

Для багатьох систем, що припускають зовнішній доступ користувачів, існує ряд загроз з боку різного роду WEB-роботів, що здійснюють автоматичну і, як правило, інтенсивну маніпуляцію даними в корисливих цілях зловмисника. Перелік загроз включає:

– інформаційна розвідка – несанкціонований збір великого набору інформації, викачування баз даних;

– розповсюдження незапрошеної інформації;

– автоматичне голосування в системах опитування соціальної думки;

– спотворення статистичної інформації при збиранні її сервером;

– підбір паролів;

– реєстрацію облікових записів з метою перепродажу;

– здійснення безлічі автоматичних дій після підбору пароля та інші.

Нині найбільш популярним рішенням є використання графічного представлення тексту при реєстрації або при додаванні коментарів, при скачуванні файлів з файлообмінних сервісів чи при відновленні втрачених паролів облікових записів. Це найбільш ефективний засіб перевірки користувача на відповідність тим діям, що він виконує.

Слід зауважити, що цей тип візуальної і текстової перевірки доставляє масу незручностей для користувачів з поганим зором або нездібних до читання. Природно, це зображення не має ніякого супровідного тексту,

оскільки цей текст розпізнала б будь-яка автоматизована система. З багатьох причин такий тип перевірки перешкоджає користувачам з фізичними вадами створювати облікові записи, писати коментарі або робити покупки на сайтах, що використовують засоби тестування користувачів. Це означає що засіб тестування не може впізнати користувачів з обмеженими можливостями як людей, а не як програм.

Метою даної кваліфікаційної роботи є розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга для підвищення захисту від автоматизованого виконання зловмисних дій WEB-роботами, що можуть загрожувати конфіденційності, доступності чи цілісності інформації.

РОЗДІЛ 1. СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ

1.1 Актуальність обраної теми

Актуальністю обраної теми є те, що автоматизоване виконання зловмисних дій на WEB-серверах є основою для інформаційної розвідки в великих масштабах, розповсюдження незапрошеної інформації, захопленню облікових записів користувачів на різних WEB-ресурсах, реєстрації облікових записів з метою перепродажу тощо.

Ця проблема є суттєвою, так як кількість незапрошеної інформації з кожним роком стає все більшою, а її використання зловмисниками може бути не тільки з метою розповсюдження рекламної інформації, а також з розповсюдженням шкідливого ПЗ. Захоплення облікових записів користувачів призводить до порушення властивостей інформації, таких як конфіденційність чи доступність. Зловмисна модифікації даних в базах знать, наприклад у Wikipedia, призводить до порушення цілісності інформації. Отже, обмеження можливостей WEB-роботів призведе до зменшення загроз, яку вони становлять.

1.1.2 Мета дослідження

Розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга для підвищення захисту від автоматизованого виконання зловмисних дій WEB-роботами, що можуть загрожувати конфіденційності, доступності чи цілісності інформації.

1.1.3 Об'єкт дослідження

Об'єктом дослідження є система тестування користувачів WEB-серверів.

1.1.4 Предмет дослідження

Предметом дослідження є засоби захисту WEB-серверів від автоматизованого виконання зловмисних дій на основі зворотного тесту Тюрінга.

1.1.5 Задачі дослідження

1) Аналіз загроз WEB-серверів;
2) Аналіз засобів захисту від WEB-роботів;
3) Огляд способів обходу зворотного тесту Тюрінга;
4) Вибір профілю захищеності;
5) Розробка системи тестування користувачів WEB-серверів.

1.2 Класифікація атак на WEB-сервери

Класифікація атак на WEB-сервери має ієрархічну структуру та розділяється на шість основних класів.

1) Атаки на засоби автентифікації;
2) Атаки на засоби авторизації;
3) Атаки на клієнтів;
4) Атаки, направлені на виконання коду;
5) Атаки, направлені на розголошення інформації;
6) Логічні атаки.

1.2.1 Атаки на засоби автентифікації

Атаки цього класу направлені на обхід чи експлуатацію вразливостей в механізмах реалізації автентифікації WEB-серверів.

1.2.1.1 Підбір

Підбір представляє собою автоматизований процес спроб на похибки, основною метою якого є вгадування імені користувача, пароля, номера кредитної картки, ключа шифрування і так далі. Багато систем дозволяють використовувати слабкі паролі або ключі шифрування, і користувачі часто вибирають легко вгадувані або такі, що містяться в словниках парольні фрази. Користувачі навмисно вибирають прості паролі, оскільки складні окрім часу введення, незручні ще і тим, що легко забуваються. Скориставшись цією ситуацією, зловмисник може застосувати електронний словник і спробувати використовувати усю потужність комбінацій символів, що містяться в словнику, як пароль.

Подібна техніка спроб і помилок може бути з успіхом використана для підбору ключів шифрування. У разі використання сервером ключів недостатньої довжини зловмисник може отримати використовуваний ключ, протестувавши усі можливі комбінації. Існує два види підбору: прямий і зворотний. При прямому підборі використовуються різні варіанти пароля для одного імені користувача.

При зворотному перебираються різні імена користувачів, а пароль залишається незмінним. У системах з мільйонами облікових записів вірогідність використання різними користувачами одного пароля досить висока. Незважаючи на популярність і високу ефективність, підбір може займати декілька годин, днів або років. Цей вид атак широко використовується переважно там, де відсутнє блокування у разі невірного поєднання. На рисунку 1.1 зображений механізм підбору.

Рисунок 1.1 – Підбір

1.2.1.2 Недостатня автентифікація

Ця уразливість виникає тоді, коли WEB-сервер дозволяє зловмиснику діставати доступ до важливої інформації або функцій сервера без належної автентифікації. Атаки подібного роду дуже часто реалізуються за допомогою інтерфейсу адміністрування через WEB. Щоб не

використовувати автентифікацію, деякі ресурси по дефолту використовують певну адресу, яка не вказана на основних сторінках сервера або інших загальнодоступних ресурсах. Необхідний URL може бути знайдений шляхом перебору типових файлів і директорій (таких, як /admin/) з використанням повідомлень про помилки журналів перехресних посилань або шляхом простого читання документації. Подібні ресурси мають бути захищені адекватно важливості їх вмісту і функціональних можливостей.

1.2.1.3 Небезпечне відновлення паролів

Ця вразливість реалізується завдяки тому, що WEB-сервер дозволяє зловмиснику несанкціоновано отримувати, модифікувати або відновлювати паролі інших користувачів. Часто автентифікація на WEB-сервері вимагає від користувача запам'ятовування пароля або парольної фрази. Строга політика безпеки передбачає що тільки користувач повинен знати пароль, причому пам'ятати його чітко. Прикладом реалізації функції відновлення паролю є використання «секретного питання», відповідь на який вказується в процесі реєстрації. Питання або вибирається із списку, або вводиться самим користувачем. Ще один механізм дозволяє користувачеві вказати «підказку», яка допоможе йому згадати пароль. Інші способи вимагають від користувача вказати частину персональних даних - таких, як номер паспорта, домашня адреса, поштовий індекс і так далі, - які потім використовуватимуться для встановлення особи. Після того як користувач доведе свою ідентичність, система відобразить новий пароль або перешле його поштою. Уразливості, пов'язані з недостатньою перевіркою при відновленні пароля, виникають, коли зловмисник отримує дані, які використовуються механізмом відновлення. Це трапляється, коли інформацію для перевірки користувача легко вгадати або сам процес підтвердження можна обійти. Система відновлення пароля може бути скомпрометована шляхом використання підбору вразливостей системи або із-за легковгадуваної відповіді на секретне питання.

1.2.2 Атаки на засоби авторизації

Процес авторизації стоїть в основі атак, спрямованих на методи, які використовуються WEB-сервером для визначення того, чи має користувач, служба необхідні для здійснення дії дозволи. Багато WEB-ресурсів дозволяють доступ до деякого вмісту тільки певним користувачам. Доступ для інших має бути обмежений. Використовуючи різну техніку, зловмисник може підвищити свої привілеї і дістати доступ до захищених ресурсів.

1.2.2.1 Передбачуване значення ідентифікатора сесії

Передбачуване значення ідентифікатора сесії дозволяє перехоплювати сесії інших користувачів. Подібні атаки виконуються шляхом припущення або вгадування унікального ідентифікатора сесії користувача. Ця атака, також як і перехоплення сесії у разі успіху, дозволяє зловмисникові послати запит WEB-серверу з правами скомпрометованого користувача. Автентифікація користувачів при першому звернені використовується на багатьох WEB-ресурсах, та в подальшому виконується відстеження сесії користувача. Для цього користувач вказує комбінацію імені і пароля. Замість повторної передачі імені користувача і пароля при кожній транзакції WEB-сервер генерує унікальний ідентифікатор, який привласнюється сесії користувача. Наступні запити користувача до сервера містять ідентифікатор сесії як доказ того, що автентифікація була успішно пройдена. Якщо зловмисник зможе передбачити або вгадати значення ідентифікатора іншого користувача, це може бути використано для проведення атаки.

1.2.2.2 Недостатня авторизація

Недостатня авторизація виникає, коли WEB-сервер дозволяє зловмиснику діставати доступ до важливої інформації або функцій, доступ до яких має бути обмежений. Окрім автентифікації, має бути реалізоване розмежування доступу до WEB-ресурсу. Процедура авторизації визначає, які дії може здійснювати користувач. Правильно побудовані правила

доступу повинні обмежувати дії користувача, згідно з політикою безпеки. Доступ до важливих ресурсів сайту має бути дозволений тільки адміністраторам.

Деякі сервери після автентифікації зберігають в cookie або прихованих полях ідентифікатор «ролі» користувача ПЗ. Якщо розмежування доступу ґрунтується на перевірці цього параметра без верифікації приналежності до ролі при кожному запиті, зловмисник може підвищити свої привілеї, модифікувавши значення cookie.

1.2.2.3 Відсутність тайм-ауту сесії

Ця вразливість виникає у разі, якщо для ідентифікатора сесії або облікових даних не передбачений тайм-аут або його значення занадто 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 має бути забезпечена, щоб запобігти множинному доступу користувачів з одним обліковим записом.

Викрадений ідентифікатор може використовуватися для доступу до даних користувача або здійснення шахрайських транзакцій. Відсутність тайм-ауту сесії збільшує вірогідність успіху різних атак. Велике значення тайм-ауту збільшує шанси підбору діючого ідентифікатора. Крім того, збільшення цього параметра веде до збільшення одночасно відкритих сесій, що ще більше підвищує вірогідність успішного підбору. Якщо функція виходу з системи просто перенаправляє на основну сторінку WEB-сервера, а не завершує сесію, сторінки, відвідані користувачем, можуть бути проглянуті зловмисником. Оскільки ідентифікатор сесії не був відмічений як недійсний, зловмисник може отримати доступ до сторінок сервера без повторної автентифікації.

1.2.2.4 Фіксація сесії

Використовуючи цю атаку, зловмисник вказує ідентифікатору сесії користувача задане значення. Залежно від функціональних можливостей сервера, існує декілька способів зафіксувати значення ідентифікатора сесії. Для цього можуть використовуватися атаки типу «міжсайтове виконання сценаріїв» або підготовка сайту за допомогою попереднього HTTP-запиту. Після фіксації значення ідентифікатора сесії, зловмисник чекає моменту, коли користувач увійде до системи.

Після входу користувача зловмисник використовує ідентифікатор сесії для отримання доступу до системи від імені користувача.

Існують два механізми управління сесіями на основі ідентифікаторів. Перший (дозволяючий) - заснований на тому, що конкретний ідентифікатор сесії привласнюється Інтернет-браузером. Другий (строгий) - функціонує з ідентифікаторами, які згенеровані сервером. У разі дозволяючих механізмів зловмисник може вибрати будь-який ідентифікатор сесії. За умови відсутності спецзахисту від фіксації сесії, ця атака може бути використана проти будь-якого сервера, який автентифікує користувачів за допомогою ідентифікатора сесії. Більшість WEB-серверів зберігають ID в cookie, але це значення також може бути присутнім в URL або прихованому полі форми. Системи, які використовують cookie, є найбільш уразливими. Більшість відомих на даний момент варіантів фіксації сесії спрямовані саме на запис значення в cookie.

1.2.3 Атаки на клієнтів

Під час відвідування сайту між користувачем і сервером встановлюються довірчі «стосунки», як в технологічному, так і в психологічному аспектах. Експлуатуючи цю довіру, зловмисник може використовувати різні методи для проведення атак на клієнтів сервера.

1.2.3.1 Підміна вмісту

На відміну від дефейсу (грубої підміни вмісту) наступний спосіб підміни не переслідує мети збентежити відвідувачів сайту нецензурними виразами. Використовуючи наступну техніку підміни вмісту, зловмисник примушує користувача повірити, що сторінки згенеровані WEB-сервером, а не передані із зовнішнього джерела.

В результаті при використанні зловмисником цієї підміни, у рядку адреси Інтернет-браузера користувача буде відображатися легитивна адреса сайту, але на сторінці будуть знаходитися зовнішні дані передані з серверу зловмисника, які замасковані під легальний контент. Користувач може і не підозрювати, що вводить секретні дані, що стають надбанням того, хто уміло підробив оригінальну сторінку. Завдання зловмисника зводиться до створення копії сайту та переходу користувача по спеціально створеному посиланню.

1.2.3.2 Міжсайтове виконання сценаріїв

Наявність цієї вразливості дозволяє зловмиснику передати серверу виконуваний код, який буде перенаправлений Інтернет-браузеру користувача. Для написання подібного коду зазвичай використовуються HTML/JavaScript, але можуть бути застосовані і VBScript, ActiveX, Java, Flash та ін. Переданий код виконується в контексті безпеки (чи зоні безпеки) уразливого сервера. Використовуючи поточні привілеї, код дістає можливість читати, модифікувати або передавати важливі дані, доступні за допомогою Інтернет-браузеру. При цьому виді атаки у атакованого користувача може бути скомпрометований акаунт (крадіжка cookie), його Інтернет-браузер може бути перенаправлений на інший сервер або здійснена підміна вмісту сервера. В результаті ретельно спланованої атаки зловмисник може використовувати Інтернет-браузер жертви для перегляду сторінок сайту від імені користувача, що атакується. Передача коду в таких випадках здійснюється через URL, в заголовках HTTP-запиту (cookie, user-agent, refferer), значеннях полів форм і так далі. Існує два типи атак, що призводять до міжсайтового виконання сценаріїв: постійні (збережені) і

непостійні (відбиті). Основною відмінністю між ними є те, що у відбитому варіанті передача коду серверу і повернення його клієнтові здійснюється у рамках одного HTTP-запиту, а в збереженому - в різних. Здійснення непостійної атаки вимагає, щоб користувач перейшов по посиланню, яке сформоване зловмисником (посилання може бути передане по e-mail тощо). В процесі завантаження сайту код, впроваджений в URL або заголовки запиту, буде переданий клієнтові і виконаний в його Інтернет-браузері. Збережений різновид уразливості виникає, коли код передається серверу і зберігається на ньому на деякий проміжок часу. Найбільш популярними цілями атак в цьому випадку є форуми, пошта з WEB - інтерфейсом і чати. На рисунку 1.2 зображений механізм міжсайтового виконання сценаріїв.

Рисунок 1.2 – Міжсайтове виконання сценаріїв

1.2.3.3 Розщеплення та підміна HTTP-запиту

Для експлуатації цієї вразливості зловмисник повинен надіслати 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. Можливість здійснення атаки виникає, коли сервер повертає дані, надані користувачем в заголовках HTTP-відповіді. Зазвичай

це відбувається при перенаправленні користувача на іншу сторінку, чи коли дані, отримані від користувача, зберігаються в cookie. У першій ситуації URL, на який відбувається перенаправлення, 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) так, щоб сформувати дві HTTP-транзакції, тоді як реально відбуватиметься тільки одна. Переведення рядка використовується для того, щоб закрити першу (стандартну) транзакцію і сформувати другу пару питання/відповідь, повністю контрольовану зловмисником і абсолютно непередбачену логікою додатка. В результаті успішної реалізації цієї атаки зловмисник може виконати наступні дії:

1 Міжсайтове виконання сценаріїв;

2 Модифікація даних кеша сервера-посередника.

Деякі сервери-посередники, що керують, (Squid 2.4, NetCache 5.2, Apache Proxy 2.0 і ряд інших) зберігають підроблену зловмисником відповідь на жорсткому диску і на наступні запити користувачів за цією адресою повертають кешовані дані. Це призводить до заміни сторінок сервера на клієнтській стороні. Окрім цього зловмисник може переправити собі значення cookie користувача або присвоїти їм певне значення. Ця атака може бути також спрямована на індивідуальний кеш Інтернет-браузера користувача.

3 Міжкористувальницька 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 розділяють одне TCP-з'єднання з сервером між декількома користувачами.

В результаті другий користувач отримує у відповідь сторінку, сформовану зловмисником. Окрім підміни сторінки зловмисник може також виконати різні операції з cookie користувача.

4 Перехоплення сторінок, що містять призначені для користувача дані. В цьому випадку зловмисник отримує відповідь сервера замість самого користувача. Таким чином, він може дістати доступ до важливої або конфіденційної інформації. На рисунку 1.3 зображений механізм розщеплення та підміни HTTP-запиту.

Рисунок 1.3 – Розщеплення та підміна HTTP-запиту

1.2.4 Виконання коду

Усі сервери використовують дані, передані користувачем при обробці запитів. Часто ці дані використовуються при складанні команд, вживаних для генерації динамічного вмісту. Якщо при розробці не враховуються вимоги безпеки, зловмисник дістає можливість модифікувати виконувані команди.

1.2.4.1 Переповнення буфера

Переповнення буфера на сьогодні є найпоширенішою уразливістю в області безпеки ПЗ. Експлуатація переповнювання буфера дозволяє зловмисникові змінити шлях виконання програми шляхом перезапису даних в пам'яті системи. Переповнення виникає, коли об'єм даних перевищує розмір виділеного під них буфера. Коли буфер переповнюється, дані

переписують інші області пам'яті, що призводить до виникнення помилки. Переповнення буфера може викликати відмови в обслуговуванні, призводячи до ушкодження пам'яті і викликаючи помилки в програмах.

Використовуючи переповнення буфера, можна перезаписувати службові області пам'яті. При переповненні можуть бути переписані значення змінних в програмі.

1.2.4.2 Атака на функції форматування рядків

При використанні цих атак шлях виконання програми модифікується методом перезапису областей пам'яті за допомогою функцій форматування символьних змінних. Уразливість виникає, коли призначені для користувача дані застосовуються як аргументи функцій форматування рядків - таких, як fprintf, printf, sprintf, setproctitle, syslog і так далі. Якщо зловмисник передає додатку рядок, що містить символи форматування ("%f", "%p", "%n" і так далі), то у нього з'являється можливість:

- виконати довільний код на сервері;
- прочитати значення із стека;
- викликати помилки в програмі/відмову в обслуговуванні.

1.2.4.3 Впровадження операторів LDAP

Атаки цього типу спрямовані на WEB-сервери, які створюють запити до служби LDAP на основі даних, що вводяться користувачем. Протокол LDAP працює поверх транспортних протоколів Internet (TCP/UDP). WEB забезпечення може використовувати дані, надані користувачем для створення запитів по протоколу LDAP при генерації динамічних WEB-сторінок. Якщо інформація отримана від клієнта, належним чином не верифікується, то зловмисник дістає можливість модифікувати LDAP-запит. Слід зауважити, що запит виконуватиметься з тим же рівнем привілеїв, з яким працює компонент ПЗ, виконуючий запит (сервер СКБД, WEB-сервер і т. д). Якщо цей компонент має права на читання або модифікацію даних в структурі каталогу, зловмисник дістає ті ж можливості.

1.2.4.4 Виконання команд ОС

Атаки цього класу спрямовані на виконання команд операційної системи на WEB-сервері шляхом маніпуляції вхідними даними. Якщо інформація, отримана від клієнта, належним чином не верифікується, то зловмисник дістає можливість виконати команди ОС. Вони виконуватимуться з тим же рівнем привілеїв, з яким працює компонент ПЗ, виконуючий запит (сервер СКБД, WEB-сервер і т. д).

Програмні WEB забезпечення часто використовують параметри, які вказують на те, який файл відображувати або використовувати як шаблон. Якщо цей параметр не перевіряється досить ретельно, то зловмисник може підставити свої команди ОС до запиту.

Більшість мов сценаріїв дозволяють запускати команди ОС під час виконання, використовуючи варіанти функції exec. Якщо дані, отримані від користувача передаються цій функції без перевірки, зловмисник може виконати команди ОС на відстані.

1.2.4.5 Впровадження операторів SQL

Ці атаки спрямовані на WEB-сервери, які створюють SQL-запити до серверів СКБД на основі даних, що вводяться користувачем. Мова запитів SQL є спеціалізованою мовою програмування, що дозволяє створювати запити до серверів СКБД. Більшість серверів підтримують цю мову у варіантах, стандартизованих ISO і ANSI. У більшості сучасних СКБД присутні розширення діалекту SQL, специфічні для цієї реалізації (T-SQL в Microsoft SQL Server, -PL SQL в Oracle і т. д.). Багато програмного WEB забезпечення використовує дані, передані користувачем, для створення динамічних WEB-сторінок. Якщо інформація, отримана від клієнта, належним чином не верифікується, то зловмисник дістає можливість модифікувати запит до SQL-серверу, що відправляється ПЗ. Запит виконуватиметься з тим же рівнем привілегій, з яким працює компонент ПЗ, виконуючий запит (сервер СКБД, WEB-сервер і т. д). В результаті

зловмисник може отримати повний контроль над сервером СКБД і навіть його операційною системою.

Зазвичай виділяють два методи експлуатації впровадження операторів SQL: звичайна атака і атака всліпу. У першому випадку зловмисник підбирає параметри запиту, використовуючи інформацію про помилки, які згенеровані програмним WEB забезпеченням. У другому випадку стандартні повідомлення про помилки модифіковані, і сервер повертає зрозумілу для користувача інформацію про неправильне введення. Здійснення SQL Injection можливо і в цій ситуації, проте виявлення уразливості ускладнене. Найбільш поширений метод перевірки наявності проблеми - додавання виразів, що повертають істинне і помилкове значення. На рисунку 1.4 зображений механізм впровадження операторів SQL.

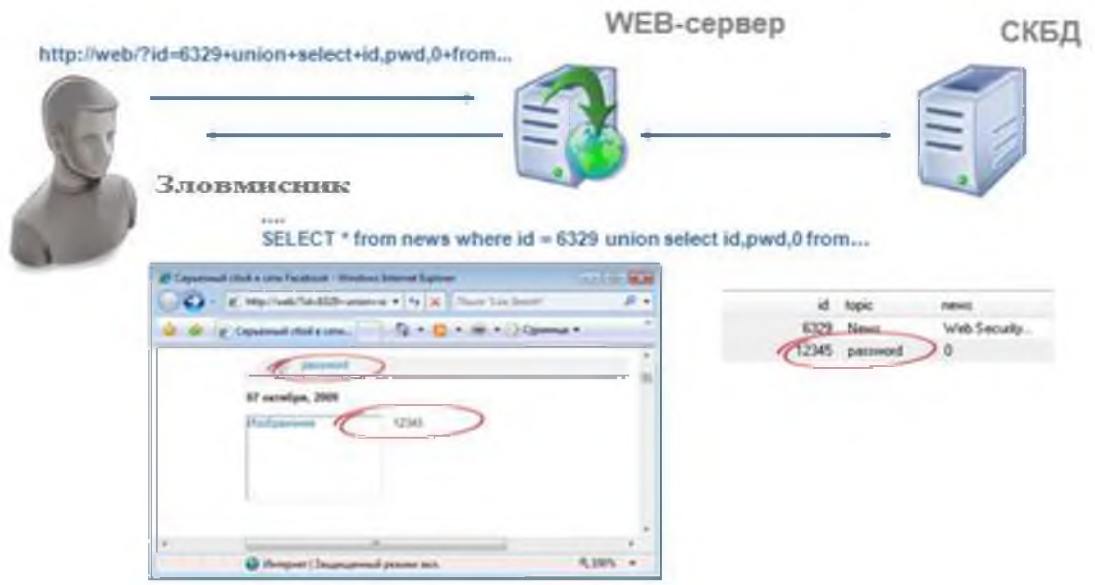

Рисунок 1.4 – Впровадження операторів SQL

1.2.4.6 Впровадження серверних розширень

Атаки цього класу дозволяють зловмисникові передати виконавчий код, який надалі буде виконаний на WEB-сервері. Вразливості, що призводять до можливості здійснення цих атак, полягають у відсутності перевірки даних, наданих користувачем перед збереженням їх у файлі, що

інтерпретується сервером. Перед генерацією HTML-сторінки сервер може виконувати сценарії, наприклад SSI. У деяких ситуаціях початковий код сторінок генерується на основі даних, наданих користувачем. Якщо зловмисник передає серверу оператори SSI, він може дістати можливість виконання команд операційної системи або включити в неї заборонений вміст при наступному відображенні. Інші можливості для атаки виникають, коли WEB-сервер використовує в URL ім'я файлу сценаріїв, що підключається, але належним чином його не верифікує. В цьому випадку зловмисник може створити на сервері файл і підключити його до виконуваного сценарію.

1.2.4.7 Впровадження операторів XPath

Ці атаки спрямовані на WEB-сервера, які створюють запити на мові XPath на основі даних, що вводяться користувачем. Мова XPath 1.0 розроблена для надання можливості звернення до частин документу на мові XML. Він може бути використаний безпосередньо або як складова частина XSLT-перетворення XML-документів, або як виконання запитів XQuery. Синтаксис XPath близький до мови SQL-запитів.

Якщо запити XPath генеруються під час виконання на основі введених даних користувача, то у зловмисника з’являється можливість модифікувати запит з метою обходу логіки роботи програми.

1.2.5 Розголошування інформації

Атаки цього класу спрямовані на отримання додаткової інформації програмного WEB забезпечення. Використовуючи ці вразливості, зловмисник може визначити ПО, що використовується, номери версій клієнта і сервера і встановлені оновлення. У інших випадках може бути отримана інформація, котра може містити шлях розташування тимчасових файлів або резервних копій. У багатьох випадках ці дані не вимагаються для роботи користувачів. Більшість серверів надають доступ до надмірного об'єму даних, проте необхідно мінімізувати об'єм службової інформації.

1.2.5.1 Індексування директорій

Наданням списку файлів в директорії є нормальною поведінкою WEB-сервера, якщо сторінка, що відображується за умовчанням (index.html/home.html/default.htm), відсутня. Коли користувач запрошує основну сторінку WEB-сайту, він зазвичай вказує доменне ім'я сервера без імені конкретного файлу. Сервер переглядає основну директорію, знаходить в ній файл, використовуваний за умовчанням, і на його основі генерує відповідь. Якщо такий файл відсутній, як відповідь може повернутися список файлів в директорії сервера. Ця ситуація аналогічна виконанню команди «ls» (Unix) або «dir» (Windows) на сервері і форматуванню результатів у вигляді HTML. В цьому випадку зловмисник може дістати доступ до даних, не призначених для вільного доступу.

Досить часто адміністратори покладаються на «безпеку через приховання» припускаючи, що раз гіперпосилання на документ відсутнє, то він недоступний необізнаним. Сучасні сканери вразливостей, такі як Nikto, можуть динамічно додавати файли і теки до списку сканованих, залежно від результатів запитів. Використовуючи вміст /robots.txt або отриманого списку директорій, сканер може знайти захований вміст або інші файли. Таким чином зовні безпечне індексування директорій може привести до витоку важливої інформації, яка надалі буде використана для проведення атак на систему.

Використовуючи індексування директорій, можна дістати доступ до наступних даних:

– резервні копії (.bak, .old, .orig);

– тимчасові файли. Такі файли повинні видалятися сервером автоматично, але іноді залишаються доступними;

– приховані файли, назва яких починається з символу «.»;

– угода про імена. Ця інформація може допомогти передбачити імена файлів або директорій (admin або Admin, back - up або backup);

– перелік користувачів сервера. Дуже часто для кожного з користувачів створюється директорія з ім'ям, заснованим на назві облікового запису;

– імена файлів конфігурації (.conf, .cfg, .config);

– вміст серверних сценаріїв або виконуваних файлів у разі невірно вказаних розширень або дозволів.

Можуть бути використані три основні сценарії отримання списку файлів WEB-сервера:

1 Помилки конфігурації. Подібні проблеми виникають, коли адміністратор помилково вказує в конфігурації сервера цю опцію. Подібні ситуації часто виникають при налаштуванні складних конфігурацій, де деякі директорії мають бути доступні для перегляду;

2 Деякі компоненти WEB-сервера дозволяють отримувати список файлів, навіть якщо це не дозволено в конфігураційних файлах. Зазвичай це виникає в результаті помилок реалізації, коли сервер генерує список файлів при отриманні певного запиту;

3 Бази цих пошукових машин (Google, Wayback machine) можуть містити кеш старих варіантів сервера, включаючи списки файлів.

1.2.5.2 Ідентифікація програмного забезпечення

Визначення версій програмного забезпечення використовується зловмисником для отримання інформації про використовуваних сервером і клієнтом операційних системах, WEB-серверах та Інтернет-браузерах. Також ця атака може бути спрямована на інші компоненти програмного WEB-забезпечення, наприклад службу каталогу або сервер баз даних або використовувані технології програмування. Зазвичай подібні атаки здійснюються шляхом аналізу різної інформації, що надається WEB-сервером.

Для визначення версій клієнтського програмного забезпечення зазвичай використовується аналіз HTTP-запитів (порядок дотримання

заголовків, значення User-agent і так далі). Проте для цих цілей може застосовуватися і інша техніка. Так, наприклад, аналіз заголовків поштових повідомлень, створених за допомогою клієнта Microsoft Outlook, дозволяє визначити версію встановленого на комп'ютері Інтернет-браузеру Internet Explorer. Наявність детальної і точної інформації про використовувані програмного забезпечення дуже важлива для зловмисника, оскільки реалізація багатьох атак (наприклад переповнювання буфера) специфічно для кожного варіанту операційної системи або програмного забезпечення. Крім того, детальна інформація про інфраструктуру дозволяє понизити кількість помилок.

1.2.5.3 Просочування інформації

Ці вразливості виникають в 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 просочування інформації. Вона не обов'язково призводить до виникнення вразливості, але часто дає зловмиснику інформацію до подальшої побудови атаки. З просочуванням важливої інформації можуть виникати ризики різної міри, тому необхідно мінімізувати кількість службової інформації, доступної на клієнтській стороні.

Аналіз доступної інформації дозволяє зловмисникові провадити розвідку і отримати уявлення про структуру директорій сервера, використовуваних SQL- запитах, назвах ключових процесів і програм сервера. Часто розробники залишають коментарі в HTML-сторінках і коді сценаріїв для полегшення пошуку помилок і підтримки програмного забезпечення. Ця інформація може варіюватися від простих описів деталей функціонування програми до (у гірших випадках) імен користувачів і паролів, використовуваних при відладці. Просочування інформації може

відноситися і до конфіденційних даних, оброблюваним сервером. Це можуть бути ідентифікатори користувача (ІНН, номери водійських посвідчень, паспортів і т.д.), а також поточна інформація (баланс особового рахунку або історія платежів). Багато атак цієї категорії виходять за рамки захисту програмного WEB-забезпечення і переходять в область фізичної безпеки. Просочування інформації в цьому випадку часто виникає коли, в Інтернет-браузері відображується інформація, яка не повинна виводитися у відкритому виді навіть користувачеві.

1.2.5.4 Зворотний шлях в директоріях

Ця техніка атак спрямована на отримання доступу до файлів, директорій і команд, що знаходяться поза основною директорією WEB-сервера. Зловмисник може маніпулювати параметрами URL з метою отримати доступ до файлів або виконати команди, що розташовуються у файловій системі WEB-сервера. Для подібних атак потенційно уразливий будь-який пристрій, що має WEB - інтерфейс. Багато WEB-серверів обмежують доступ користувача певною частиною файлової системи, зазвичай званої WEB document root або CGI root. Ці директорії містять файли, призначені для користувача, і програми, необхідні для отримання доступу до функцій WEB-забезпечення. Більшість базових атак, що експлуатують зворотний шлях, засновані на впровадженні в URL символів «./» для того щоб змінити розташування ресурсу, який оброблятиметься 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 слешу («.\») в Windows-серверах, символів URLEncode («%2e% 2e% 2f») або подвійного кодування URLEncode («.%255c»). Навіть якщо WEB-сервер обмежує доступ до файлів певним каталогом, ця уразливість може виникати в сценаріях або CGI-програмах. Можливість використання зворотного шляху

в каталогах досить часто виникає в додатках, що використовують механізми шаблонів чи завантажують текст їх сторінок з файлів на сервері. У цьому варіанті атаки зловмисник модифікує ім'я файлу, що передається як параметр CGI - програми або серверного сценарію. В результаті зловмисник може отримати початковий код сценаріїв. Досить часто до імені файлу, що запитується, додаються спеціальні символи - такі, як «%00» - з метою обходу фільтрів. На рисунку 1.5 зображений механізм отримання зворотного шляху в директоріях.

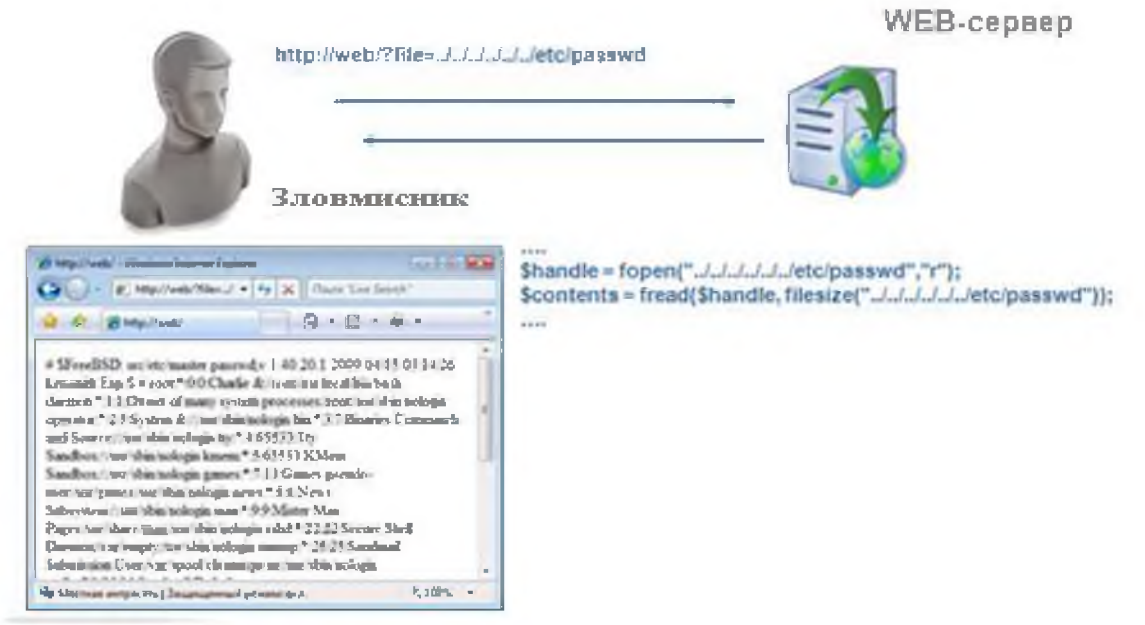

Рисунок 1.5 – Зворотних шлях в директоріях

1.2.5.5 Передбачуване розташування ресурсів

Передбачуване розташування ресурсів дозволяє зловмисникові дістати доступ до прихованих даних або функціональних можливостей. Шляхом підбору зловмисник може дістати доступ до вмісту, не призначеного для публічного перегляду. Тимчасові файли, файли резервних копій, файли конфігурації або стандартні приклади часто є метою подібних атак. В більшості випадків перебір може бути оптимізований шляхом використання стандартної угоди про імена файлів і директорій сервера. Отримувані зловмисником файли можуть містити інформацію про дизайн додатка, інформацію з баз даних, імена машин або паролі, шляхи до директорій.

Приховані файли також можуть містити вразливості, відсутні в основному застосуванні.

1.2.6 Логічні атаки

Атаки цього класу спрямовані на експлуатацію функцій ПЗ або логіки його функціонування. Логіка ПЗ є очікуваним процесом функціонування програми при виконанні певних дій. Як приклади можна привести відновлення паролів, реєстрацію облікових записів, аукціонні торги, транзакції в системах електронної комерції. ПЗ може вимагати від користувача коректного виконання декількох послідовних дій для отримання певного результату. Зловмисник може обійти ці механізми або використовувати їх у своїх цілях.

1.2.6.1 Зловживання функціональними можливостями

Ця атака спрямована на використання функцій програмного WEB-забезпечення з метою обходу механізмів розмежування доступу. Деякі механізми програмного WEB-забезпечення включаючи функції забезпечення безпеки можуть бути використані для цих цілей. Наявність вразливості в одному з другорядних компонентів ПЗ може привести до компрометації усього ПЗ. Рівень ризику і потенційні можливості зловмисника у разі проведення атаки дуже сильно залежать від конкретного ПЗ. Зловживання функціональними можливостями дуже часто використовується спільно з іншими атаками - такими, як зворотний шлях в директоріях і так далі. Приклади зловживання функціональними можливостями включають:

– 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;

– реалізацію відмови в обслуговуванні шляхом використання функції блокування облікового запису при багаторазовому введенні неправильного пароля.

1.2.6.2 Відмова в обслуговуванні

Цей клас атак спрямований на порушення доступності WEB-сервера. Атаки, спрямовані на відмову в обслуговуванні, реалізуються на мережевому рівні, проте вони можуть бути спрямовані і на прикладний рівень. Використовуючи функції програмного WEB забезпечення зловмисник може вичерпати критичні ресурси системи або скористатися вразливістю, що призводить до припинення функціонування системи. Зазвичай DoS-атаки спрямовані на вичерпання критичних системних ресурсів - таких, як обчислювальні потужності, оперативна пам'ять, дисковий простір або пропускна спроможність каналів зв'язку. Якщо якийсь з ресурсів досягне максимального завантаження, ПЗ цілком буде недоступне. Атаки можуть бути спрямовані на будь-який з компонентів WEB-забезпечення, наприклад, такі як сервер СКБД, сервер автентифікації і т. д.

1.2.6.3 Недостатня протидія автоматизації

Недостатня протидія автоматизації виникає, коли сервер дозволяє автоматично виконувати операції, які повинні проводитися вручну. Для деяких функцій програмного забезпечення 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 привести до падіння продуктивності усього WEB-серверу. Протидія автоматизації полягає в обмеженні можливостей подібних програмних засобів.

1.2.6.4 Недостатня перевірка процесу

Вразливості цього класу виникають, коли сервер недостатньо перевіряє послідовність виконання операцій ПЗ. Якщо стан сесії користувача і ПЗ належним чином не контролюється, ПЗ може бути вразливий для шахрайських дій. В процесі доступу до деяких функцій ПЗ очікується, що користувач виконає ряд дій в певному порядку. Якщо деякі дії виконуються невірно або в неправильному порядку, виникає помилка, що призводить до порушення цілісності. Прикладами подібних функцій виступають переведення, відновлення паролів, підтвердження купівлі, створення облікового запису і т.д. В більшості випадків ці процеси складаються з ряду послідовних дій, здійснюваних в чіткому порядку. Для забезпечення коректної роботи подібних функцій WEB-забезпечення повинно чітко відстежувати стан сесії користувача і її відповідність поточним операціям. В 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 повинна проводитися перевірка цих значень на сервері. Якщо цього не відбувається, зловмисник дістає можливість обійти послідовність дій, тобто, логіку ПЗ.

1.3 Загальні відомості про автоматизацію

Під автоматизацією розуміють один з напрямів науково-технічного прогресу, застосування саморегулюючих технічних засобів, економіко-математичних методів і систем управління, що звільняють людину від участі в процесах отримання, перетворення передачі і використання енергії, матеріалів або інформації, що істотно зменшують міру цієї участі або трудомісткість виконуваних операцій.

Автоматизуються:

- виробничі процеси;
- проєктування;
- організація, планування і управління;
- наукові дослідження;

– бізнес-процеси та інше.

Мета автоматизації – підвищення продуктивності праці, поліпшення якості продукції, оптимізація управління, усунення людини від виробництв, небезпечних для здоров'я.

1.4 Аналіз засобів захисту від автоматизованого виконання зловмисних дій

Засоби захисту від автоматизованого виконання зловмисних дій розділяються на 2 види:

– засоби, які вимагають дії від користувача;

– засоби, які не вимагають від користувача ніяких дій.

1.4.1 Засоби, які вимагають дії від користувача

1.4.1.1 Зворотний тест Тюрінга

Зворотним тестом Тюрінга є CAPTCHA. CAPTCHA – це акронім від англійських слів «**C**ompletely **A**utomatic **P**ublic **T**uring Test to Tell **C**omputers and **H**umans **A**part» – повністю автоматичний зворотний тест Тюрінга для розрізнення комп'ютерів і людей. Це завдання, яке легко вирішує людина, але яке неможливо (чи украй важко) навчити вирішувати комп'ютер.

1.4.1.2 Класичний зворотний тест Тюрінга

Класичний зворотний тест Тюрінга полягає в тому, що на початку перевірки сервером створюється тест-зображення з набором цифр та букв і пропозиція ввести цей набір в відповідне поле. Мета цієї операції – запобігти атакам WEB-роботів на WEB-сервер. Обґрунтуванням подібної операції є те, що доки не існує програм досить потужних для того, щоб розпізнати всі тест-зображення без виключення і точно відтворити текст з них, тому вважається, що система, яка змогла це зробити, з високою вірогідністю може вважатися людиною.

1.4.1.3 Створення тест-зображення в класичному зворотному тесті Тюрінга

Створення відбувається завдяки використанню скрипту, написаного за допомогою однієї з серверної скрипкової мови програмування (PHP, ASP,

Python, Perl), на стороні WEB-сервері. Слід зазначити, що для виконання дій з зображенням на WEB-сервері що використовує мову програмування PHP, повинна бути підтримка бібліотеки GD (програмна бібліотека для динамічної роботи з зображенням) або інших бібліотек, які працюють з зображенням, наприклад: ImageMagick.

При створенні тест-зображення використовують наступні параметри:

- розмір тест-зображення;
- колір фону;
- шрифт символів;
- колір символів;
- силу деформації;
- інтенсивність зашумлення;
- та інші, з урахуванням необхідного функціоналу.

1.4.1.4 Алгоритм створення тест-зображення в класичному зворотному тесті Тюрінга

Порядок етапів алгоритму може змінюватися, але в цілому налічується 5 етапів:

Таблиця 1.1 – Загальний вид алгоритму створення тест-зображення

	Етапи	Результат
	Створення фону тест-зображення	
	Ініціалізація набору символів	SX3HX
	Об’єднання набору символів з створеним фоном тест-зображення	
	Зашумлення тест-зображення	
	Деформування тест-зображення	

В таблиці 1.1 зображений загальний вид алгоритму створення тест-зображення класичного зворотного тесту Тюрінга та приклад створення. Пункти 4 та 5 можуть мінятися позиціями.

Створення фону тест-зображення

Для створення фону тест-зображення у більшості випадків використовуються 2 параметри: колір, розмір. Рекомендується задавати колір фону, а в подальшому колір набору символів в близькому кольоровому діапазоні.

Ініціалізація набору символів

При ініціалізації набору символів в основному використовується випадковий вибір із заданої кількості символів. Можуть використовуватися як літери, так і цифри. Використання спеціальних знаків не рекомендується, тому що після виконання деформації та зашумлення зображення, може виникнути ситуація, при якій ці символи не будуть прочитані людиною.

Об'єднання набору символів з створеним фоном тест-зображенням

Об'єднання набору символів з зображенням потрібно для того, щоб обробляти тест-зображення в подальшому як єдине ціле. Тобто деформація зображення та зашумлення будуть впливати на все зображення в цілому.

Зашумлення тест-зображення

Зашумлення зображення, як і деформація використовується для ускладнення розпізнавання символів WEB-роботами, які використовують OCR. Зашумлення може бути різноманітним – від розсипу крапок, до інших менш виділених символів.

Деформація тест-зображення

Деформація зображення виконується для ускладнення розпізнання символів WEB-роботами, які використовують OCR. Для деформації зображення використовують таки алгоритми як, наприклад:

– алгоритм MultiSwirl;
– алгоритм MultiWave.

Алгоритм створення тест-зображення може відрізнятися та налічувати додаткові етапи. Слід зазначити, що також можливим є алгоритм створення тест-зображення без деформації зображення. В такому випадку можливі два варіанти:

- використання нестандартних шрифтів;
- використання декількох шрифтів;
- випадковий поворот символів;
- розміщення кожного символу окремо на зображенні у відповідності до координат XY.

1.4.1.5 Алгоритм перевірки користувача класичного зворотного тесту Тюрінга

Алгоритм перевірки користувача може відрізнятися та налічувати додаткові етапи. Так для захисту від підміни сесії може додаватися етап перевірки номеру створеної сесії.

Алгоритм перевірки користувача класичного зворотного тесту Тюрінга, зображений на рисунку 1.6.

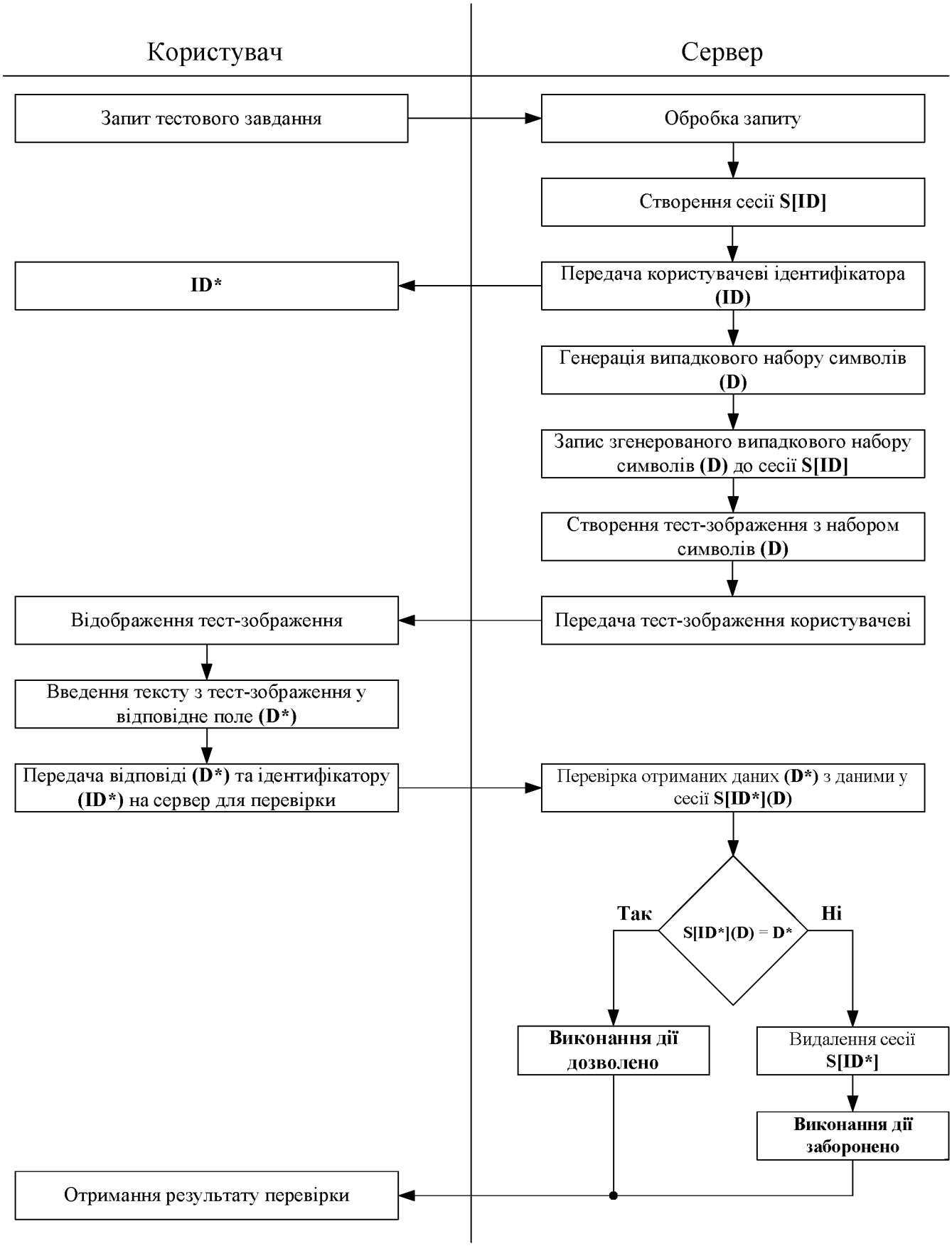

Рисунок 1.6 – Алгоритм перевірки

1) Запит тестового завдання користувачем;

2) Створення сесії (S) з ідентифікатором [ID]. Сесія створюється за допомогою скрипта створення тест-зображення;

3) Передача файлу ідентифікатора ID користувачеві. Ідентифікатор використовується для прив'язки користувача до відповідної сесії на сервері;

4) Генерація випадкового набору символів (D);

5) Запис згенерованого набору символів (D) до сесії S[ID] користувача на сервері. Такі дії дозволять тимчасово зберегти дані для перевірки в подальшому;

6) Створення тест-зображення з набором символів (D). Зображення для перевірки створюється завдяки скрипту, розташованому на сервері;

7) Передача тест-зображення користувачеві;

8) Відображення тест-зображення;

9) Введення тексту з тест-зображення у відповідне поле;

10) Передача відповіді (D*) та ідентифікатору сесії (ID*) на сервер для перевірки;

11) Перевірка отриманих даних (D*) з даними у сесії S[ID*](D);

12) В результаті повного збігу, користувачеві передається результат перевірки. У випадку повного збігу перевірених даних, користувач допускається до виконання дій, передбачених на сервері для нього. У випадку, якщо дані в сесії та отримані від користувача не співпадають, сесія видаляється та користувачеві забороняється виконання дій на сервері.

1.4.1.6 Переваги та недоліки класичного зворотного тесту Тюрінга

Переваги та недоліки класичного зворотного тесту Тюрінга відображені в таблиці 1.2.

Таблиця 1.2 – Переваги та недоліки класичного зворотного тесту Тюрінга

	Переваги	Недоліки
	Простота в реалізації	Використання сторонніх сервісів для розпізнання тест-зображення людьми
	Простота інтеграції до WEB-сайту	Використання OCR
		Незручності для людей з порушенням зору

1.4.1.7 Готові рішення класичного зворотного тесту Тюрінга

Існує велика кількість готових рішень зворотного тесту Тюрінга. Кожний WEB-ресурс може мати свій, але найбільш розповсюдженими є такі:

reCAPTCHA

reCAPTCHA – проєкт, що використовує в ролі робочого елемента для відповідей користувачів на CAPTCHA-запит нерозбірливе для OCR слово, що є одним з безлічі спотворених фрагментів книг, що сканують, на додаток до слова, згенерованого комп'ютером. Цей сервіс враховує прийоми використання і можливості програм оцифрування тексту книг. Для надійності одне й те ж слово пропонується декільком користувачам різних сайтів. Коли різні користувачі однаково відповіли на CAPTCHA-запит, передбачається, що вони ввели правильне слово. Приклад reCAPTCHA показаний на рисунку 1.7.

Рисунок 1.7 – reCAPTCHA

clCAPTCHA

clCAPTCHA – безкоштовний сервіс, що дозволяє стороннім WEB-розробникам використовувати різноманітні стандартні і індивідуальні CAPTCHA. Можливе повне налаштування зовнішнього вигляду, використання фонів, різного зашумления, вказівку переліку символів з підтримкою UNICODE. Запит CAPTCHA здійснюється як звернення до зображення за допомогою тега <img>. Перевірка здійснюється зверненням до сервера CatLair по протоколу HTTP. Приклад clCAPTCHA показаний на рисунку 1.8.

Рисунок 1.8 – clCAPTCHA

KCAPTCHA

Проєкт KCAPTCHA – це готове рішення, написане на мові PHP, яке є безкоштовним та знаходиться у вільному розповсюджені. Проєкт KCAPTCHA ставить перед собою мету запропонувати WEB-програмістові рішення з одного боку дуже захищене, з іншого – максимально маловимогливе до ресурсів і конфігурації хостингу.

Принцип дії: скрипт заводить сесію і записує в неї під ім'ям $_SESSION['capt-cha_keystring'] випадковим чином згенерований набір символів, після чого видає зображення, що містить цей самий набір символів у зашумленном вигляді. Для перевірки достатньо порівняти кодовий набір символів з тим, що ввів користувач.

Системні вимоги: PHP версії 4.0.6 і вище з підтримкою GD версії 2. Не потребує ні бібліотеки для роботи з шрифтами (Libttf та ін.), ні ImageMagick. Приклад KCAPTCHA показаній на рисунку 1.9.

Рисунок 1.9 – KCAPTCHA

1.4.1.8 Альтернативні види зворотного тесту Тюрінга

Існує багато видів CAPTCHA-захисту, що відрізняються від класичного варіанта «Введіть код, зображений на картинці». Їх можна розділити на дві основні частини: завдання, що звертаються до «рефлексів» користувача (на пізнавання-розпізнавання), і завдання, що звертаються до його логіки (питання, завдання). Перші приємніші для користувача, так як не примушують його думати.

Аудіо-CAPTCHA

Аудіо-CAPTCHA пропонує користувачеві прослухати деяку фразу і потім ввести її. Зазвичай фраза складається з цифр, що промовляються, як правило, з варійованою тональністю, паузами і фоновими шумами.

Аудіо-CAPTCHA застосовується досить рідко і тільки як альтернатива для користувачів з порушеннями зору, рисунок 1.10. Переваги та недоліки аудіо- CAPTCHA наведені в таблиці 1.3.

Таблиця 1.3 – Переваги та недоліки аудіо-CAPTCHA

	Переваги	Недоліки
	Можливість відповісти на тест-питання користувачеві з порушенням зору	Користувач повинен мати устаткування для відтворення звуку
		Наявність бази фрагментів звукової інтерпретації символів
		Реалізація звукових спотворень досить вимоглива до кваліфікації програміста і ресурсів сервера

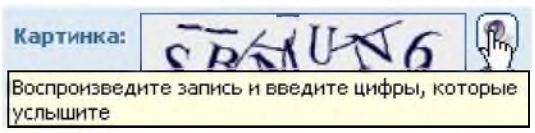

Рисунок 1.10 - Аудіо-CAPTCHA

Математична CAPTCHA

Використання питань типу: «скільки буде 44+78», не дозволяє забезпечити належною захист. Це рішення може забезпечити деякий захист тільки через свою малу поширеність. Як відомо, складати і віднімати числа комп'ютер уміє набагато краще за людину, для самої людини ж проведення в думці математичних дій (особливо, якщо цифр в числах не 1-2, а більше) буде стомливим і складним. Приклад математичної CAPTCHA зображений на рисунку 1.11. Переваги та недоліки наведені в таблиці 1.14

Рисунок 1.11 – Математична CAPTCHA

Таблиця 1.4 – Переваги та недоліки математичної CAPTCHA

	Переваги	Недоліки
	Невелике розповсюдження	Подразливість від проходження тесту
	Простота в реалізації	Користувач повинен добре володіти арифметикою
	Простота в інтеграції до WEB-сайту	Використання спеціальних знаків

Текстові завдання

Суть цього методу полягає в тому, що людині задається питання або загадка, на який вона повинна дати відповідь. Відповідь треба або вибрати зі списку, або ввести в поле. При виборі зі списку вірогідність дати правильну відповідь навмання досить велика (1/n де n – кількість варіантів), зазвичай користувача примушують відповісти на низку запитань, бо вірогідність вгадати правильні відповіді на декілька питань буде добутком вірогідності відповісти на кожне з них. Переваги та недоліки тестових завдань наведені в таблиці 1.5.

Таблиця 1.5 – Переваги та недоліки текстових завдань

	Переваги	Недоліки
	Можливість надання відповіді на тест-питання користувачеві з порушенням зору	Користувач має бути добре знайомий з мовою, на якій ставляться питання
	Можливість надання відповіді користувачам, які використовують Інтернет-браузери, що не відображають зображення	Кінцева кількість бази питань

Можна намагатися комбінувати текстове завдання із завданням на розпізнавання, приміром «введіть першу, третю і четверту букви із запропонованого напису», але це екстенсивний шлях, що веде до уявного збільшення складності проходження тесту WEB-роботом.

Тест на розпізнавання предметів

Суть методу полягає в розпізнанні користувачем зображених на картинці предметів (людей, тварин). Йому або показують предмет і просять ввести його назву (чи вибрати його зі списку), або, навпаки, пишуть назву, а з декількох запропонованих предметів просять вибрати запрошений (запрошені).

Реалізація, де саме картинки є відповідями на питання (наприклад: «вкажіть усіх кішок») в протилежність «що зображене на картинці», представляється перспективнішою, так як, менше напружує користувача та не вимагає роздумувати над синонімами при введенні відповіді вручну і не вимагає проводити виснажливий пошук серед текстових варіантів відповіді.

В інших реалізаціях цього виду зворотного тесту Тюрінга можуть використовуватися тривимірне зображення з вимогою вказати таке саме, але зображене під іншим кутом. Приклад тесту на розпізнання предметів рисунок 1.12 та 1.13.

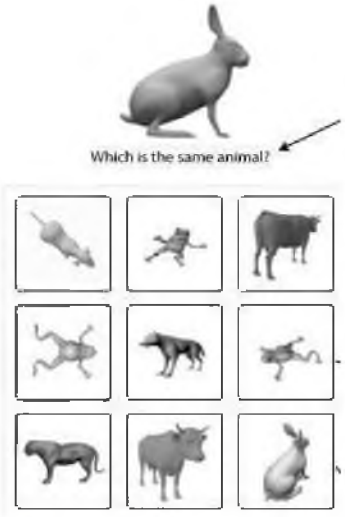

Рисунок 1.12 - CAPTCHA с розпізнання тривимірних зображень

Рисунок 1.13 – CAPTCHA з розпізнання предметів

Переваги та недоліки тесту на розпізнання предметів відображені в таблиці 1.6.

Таблиця 1.6 – Переваги та недоліки тесту на розпізнання предметів

	Переваги	Недоліки
	Легше розрізнення образів людиною, наприклад (кішки від собаки), ніж спотворених та зашумлених букв «N» і «H»	Користувач має бути ознайомленим з об'єктами та їх назвами
		Кінцева кількість бази картинок з об’єктами
		Розмір тестового завдання
		Підвищення трафіку
		Трудомісткий процес створення та упорядковування бази з зображенням

Тривимірна-CAPTCHA

Являє собою набір символів зображений в проєкції (ізометрична, аксонометрична та ін.). Цей вид CAPTCHA можна вважати класичним-модифікованим. Приклади зображені на рисунку 1.15 та 1.16. Переваги та недоліки тривимірної- CAPTCHA відображені в таблиці 1.8.

Рисунок 1.15 – Тривимірна-CAPTCHA

Рисунок 1.16 – Тривимірна-CAPTCHA

Таблиця 1.8 – Переваги та недоліки тривимірної-CAPTCHA

	Переваги	Недоліки
	Неможливість використання OCR без попередньої підготовки зображення	З використанням засобів зашумлення фону може бути важко зрозуміла користувачеві
		Використання сторонніх сервісів для розпізнання тест-зображення людьми

Тест на почуття прекрасного

Людина відрізняється не лише від комп'ютера, але навіть від тварин тим, що має «почуття прекрасного» і притаманне йому «почуття справедливості». На цій основі можна спробувати будувати тест підтвердження «людяності». Приміром це може бути список зображень людей, серед яких потрібно вибрати найкрасивіших, рисунок 1.14. Або це будуть питання, що мають на увазі моральний вибір. Або музичні фрази, гармонійні або ні (якщо, звичайно, «перевірка алгеброю гармонії» неможлива). Переваги та недоліки даного тесту наведені в таблиці 1.7.

Рисунок 1.14 – Тест на почуття прекрасного

Таблиця 1.7 – Переваги та недоліки тесту на почуття прекрасного

	Переваги	Недоліки
	Базування рішень на «персоніфікації»	Суб’єктивність оцінки «прекрасності»
	Невелике розповсюдження	Складність генерації завдань за співвідношеннями або наявність кінцевої бази зображень

Рухома CAPTCHA

Рухому CAPTCHA можна розділити на два види:

1 Відео-CAPTCHA;

2 Рухома GIF-CAPTCHA.

1 Відео-CAPTCHA, на прикладі NuCaptcha, являє собою рухомий по кривій рядок букв, з яких людині пропонується розпізнати останні три, виконані в червоному кольорі. Фон, по якому рухаються букви, настроюється. Кількість і колір ключових букв - ні; ці параметри контролюються в NuCaptcha і можуть бути у будь-який момент змінені, якщо розробники вирішать, що так треба для зміцнення захисту від WEB-роботів, а також людей, що розпізнають зображення, за плату. Переваги та недоліки відео- CAPTCHA наведені в таблиці 1.19.

Втім, існує деякий додатковий захист який аналізує частоту введення відповідей на відео-CAPTCHA з одного IP-адресу, в результаті біг рядка символів на наступних CAPTCHA сповільнюватиметься все більше і більше. Відповідно затримуватиметься і поява на екрані ключових червоних букв, так що у професійного CAPTCHA-розпізнавачеві знадобиться по 15 секунд на кожну відео-CAPTCHA. Це зробить роботу по розпізнаванню відео-CAPTCHA економічно невигідною.

Таблиця 1.9 – Переваги та недоліки відео-CAPTCHA

	Переваги	Недоліки
	Використання рухомих об’єктів	Використання в рухомому рядку тільки англійських літер
	Існування додаткової системи захисту	Слабка додаткова система захисту
		Використання Adobe Flash
		Складність реалізації

Слабкість додаткової системи захисту полягає в тому, що WEB-роботи в більшості випадків використовують велику кількість PROXY-серверів, що робить цю систему малоефективною;

2 Рухома GIF-CAPTCHA – є «складним» зображенням, яке складається з декількох простих. Кількість простих зображень, які використовуються при створенні CAPTCHA може варіюватися. Цей вид CAPTCHA, як і відео-CAPTCHA, являє собою рухомий рядок символів або в інших реалізаціях статичний рядок з символами які почергово з'являються та зникають. Переваги та недоліки наведені в таблиці 1.10.

Таблиця 1.10 – Переваги та недоліки GIF-CAPTCHA

	Переваги	Недоліки
	Використання рухомих об'єктів	Розділення «складного» зображення типу GIF на більш прості кадри
		Розпізнання символів є таким самим, як і в класичному зворотному тесті Тюрінга, за винятком розпізнання окремих кадрів

1.4.2 Засоби, які не вимагають від користувача ніяких дії

Ці засоби захисту являються більш доброзичливими до користувачів WEB-сайту, ніж засоби, які вимагають виконання додаткових дій, і тому при їх застосуванні зручність в користуванні WEB-сайтом для відвідувача росте. Система може як обмежувати кількість запитів, так і намагатися відрізнити людину від WEB-робота за непрямими ознаками в «поведінці». Непрямі ознаки не дають гарантії, а тільки збільшують вірогідність того, що WEB-робот буде зупинений.

1.4.2.1 Блокування за часом завантаження форми

Захистом може бути блокування при обробці повідомлень на певний час, що пройшов між завантаженням форми і її відправкою, - людині, на

відміну від WEB-робота, знадобиться деякий час на введення даних (який, як правило більше, 1-2 секунд).

1.4.2.2 Зміна імен полів, які беруть участь в передачі даних

Більшість WEB-роботів шукають на сторінці поля із стандартними іменами, наприклад, «name», «email», «mail» і тому подібне. Щоб доставити боту деяку незручність, краще називати поля нестандартно, наприклад, поле «Електронна пошта» назвати «name», а «ім'я» – «email», з розрахунку на те, що WEB-робот прийме рішення про суть полів по атрибуту name тега <input>.

Можливе використання системи генерації імені полів із випадкового набору символів, при цьому імена цих полів будуть для кожного користувача різні і зберігатися в їх сесіях та видалятися при виході чи за певним інтервалом часу.

1.4.2.3 Обмеження кількості запитів

Обмеження кількості запитів з одного IP-адреса – це досить простий спосіб, проте малоефективний, при достатній підготовленості зловмисника. Для обходу такого обмеження зловмисникові досить використовувати «анонімні» PROXY. Це дозволить представитися на сервері від імені іншого комп'ютера в мережі, а точніше, від іншого IP-адреси. Зловмисник може використовувати велику кількість цих PROXY, створюючи ілюзію того, що до сайту звертаються різні люди. Обмеження через виставляння відвідувачеві cookie також малоефективне, їх легко можна стерти або зовсім ігнорувати.

Обмеження кількості запитів від одного користувача має сенс, коли на WEB-сайті зловмисникові потрібно робити багато запитів, приміром, реєструвати поштові скриньки та розсилати спам. Якщо ж це форум або гостьова книга, то для них характерна інша ситуація – багато спамерів залишають кожний по невеликій кількості повідомлень. У результаті

загальна кількість запитів WEB-роботів досить велика, але кожен з них ніяких лімітів не перевищує.

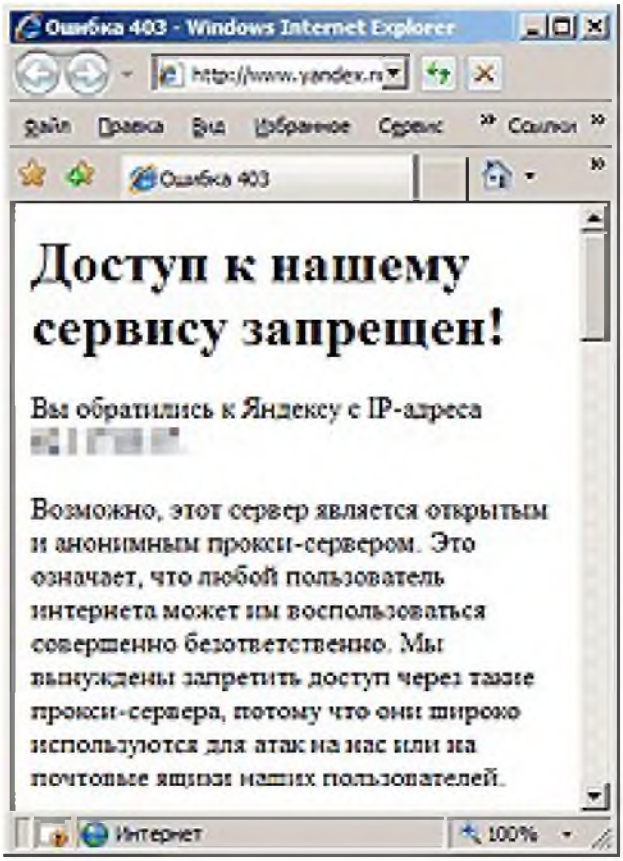

Рисунок 1.17 – Обмеження по IP

1.4.2.4 Використання полів-приманок

Виходячи з того, що метою WEB-робота є заповнення полів, можна створити приховане поле <input> та скрити його за допомогою CSS. У цьому випадку відвідувач не бачить поле і не заповнює його. WEB-робот заповнить це поле, так як вважатиме його справжнім. Форма оброблятиметься тільки тоді, коли створене поле-приманка буде порожнє.

1.4.2.5 Блокування за розміром екрану

Цей захист працює за простою схемою: якщо при програмному визначенні у відвідувача немає розмірів (ширина, висоти) екрану, то форму намагається відправити бот, тому її не слід обробляти.

Слід зазначити, що непрямі методи за визначенням менш ефективні, ніж зворотний тест Тюрінга, тому переважна більшість великих сайтів, які борються з використанням їх сервісів WEB-роботами, використовує тестові

задачі. Оскільки непрямі методи захисту розраховані на те, що WEB-робот тим або іншим чином виявить свою суть, а не на те, щоб поставити перед ним принципово складне для нього завдання.

1.5 Постановка задачі

В даному розділі було розглянуто питання захисту від автоматизованого виконання зловмисних дій на WEB-серверах. Захист від автоматизації полягає в обмеженні можливостей WEB-роботів, які й виконують автоматизовану інтенсивну маніпуляцію з даними на WEB-серверах.

Засоби захисту від автоматизованого виконання зловмисних дій поділяються на 2 види:

– засоби, які вимагають дії від користувача;

– засоби, які не вимагають від користувача ніяких дій.

Огляд та аналіз існуючих засобів захисту дав можливість визначити переваги та недоліки зворотного тесту Тюрінга різних видів, які відносяться до засобу захисту від WEB-роботів, що вимагає дії від користувача.

Засоби, які не вимагають від користувача ніяких дій, в більшій мірі відносяться до додаткового до зворотного тесту Тюрінга, захисту від WEB-роботів.

Отже, розробка нової системи тестування повинна підвищити рівень захисту WEB-серверів від автоматизованого виконання зловмисних дій шляхом виключення можливості використання нині існуючих способів обходу та надати можливість проходження тесту користувачам з порушенням зору.

РОЗДІЛ 2. СПЕЦІАЛЬНА ЧАСТИНА

2.1 Вибір профілю захищеності WEB-серверів

Згідно з рекомендаціями НД ТЗІ 2.5-010-03 та з урахуванням особливостей надання доступу до інформації WEB-сторінки, типових характеристик середовища функціонування та особливостей технологічних процесів оброблення інформації, а також зважаючи на те, що в АС класу «3» розміщується у оператора, а робочі станції – на іншій території, взаємодія яких з WEB-сервером здійснюється з використанням мереж передачі даних (технологія Т2), був обраний наступний профіль захищеності:

{ КА-2, КВ-1, КО-1,
ЦА-1, ЦО-1, ЦВ-1,
ДВ-1, ДР-1,
НР-2, НИ-2, НК-1, НО-1, НЦ-1, НТ-1, НВ-1 }

Таблиця 2.2 – Профіль захищеності WEB-серверів

Критерії	Опис критеріїв
Критерій конфіденційності	
КА-2	Базова адміністративна конфіденційність
КВ-1	Конфіденційність при обміні
КО-1	Повторне використання об’єктів
Критерій цілісності	
ЦА-1	Мінімальна адміністративна цілісність
ЦО-1	Відкат
ЦВ-1	Цілісність при обміні

Продовження таблиці 2.2

Критерій доступності	
ДР-1	Використання ресурсів
ДВ-1	Відновлення після збоїв
Критерій спостереженості	
НР-2	Реєстрація
НИ-2	Ідентифікація і автентифікація
НК-1	Достовірний канал
НО-1	Розподіл обов’язків
НЦ-1	Цілісність комплексу засобів захисту
НТ-1	Самотестування
НВ-1	Ідентифікація і автентифікація при обміні

2.1.1 Критерій конфіденційності

КЗЗ оцінюваної КС надає послуги з захисту об'єктів від несанкціонованого ознайомлення з їх змістом. Конфіденційність забезпечується такими послугами: базова адміністративна конфіденційність, конфіденційність при обміні, повторне використання об'єктів.

КА-2. Базова адміністративна конфіденційність

Ця послуга дозволяє адміністратору безпеки керувати потоками інформації від захищених об'єктів до користувачів.

Політика адміністративної конфіденційності стосується: користувачів усіх категорій, крім, визначених згідно з 6.3.1 "а" НД ТЗІ 2.5-010-03, об'єктів, що містять технологічну інформацію КСЗІ та технологічну інформацію щодо управління АС; системного та функціонального програмного забезпечення, що використовується для актуалізації, захисту загальнодоступної інформації та супроводження WEB-сторінки; доступу користувачів до окремих видів периферійних пристроїв (принтерів, накопичувачів інформації тощо), використання яких передбачено технологією обробки інформації.

КЗЗ здійснює розмежування доступу на підставі атрибутів доступу користувача і захищеного об’єкта. Доступ до загальнодоступної інформації встановлюється для користувачів усіх категорій. Призначення атрибутів доступу користувачам і процесам до захищених об'єктів здійснюється адміністратором безпеки на основі аналізу функціональних та службових обов'язків окремих користувачів.

Права доступу до кожного захищеного об'єкта, визначеного політикою безпеки послуги, встановлюються в момент його створення або ініціалізації.

КВ-1. Конфіденційність при обміні

Ця послуга дозволяє забезпечити захист об’єктів від несанкціонованого ознайомлення з інформацією, що міститься в них, під час їх експорту/імпорту через незахищене середовище.

Політика мінімальної конфіденційності при обміні стосується: користувачів, яким надано право супроводження КСЗІ та управління АС; об’єктів, які містять технологічну інформацію КСЗІ та технологічну інформацію щодо управління АС під час її передавання між віддаленими компонентами АС.

КЗЗ забезпечує захист від безпосереднього ознайомлення з інформацією, що міститься в об’єкті, який передається. КЗЗ забезпечує можливість реєстрації подій, які призвели або можуть призвести до порушення конфіденційності інформації, що міститься в об’єктах, які передаються.

Ця послуга забезпечується використанням протоколу HTTPS, що є модифікацією базового протоку передачі даних HTTP, який використовує криптографічні протоколи TSL и SSL та TCP-порт 443

КО-1. Повторне використання об’єктів

Ця послуга дозволяє забезпечити коректність повторного використання розділюваних об'єктів, гарантуючи, що в разі, якщо розділюваний об'єкт виділяється новому користувачу або процесу, то він не містить інформації, яка залишилась від попереднього користувача або процесу.

Політика повторного використання об'єктів, що реалізується КЗЗ, відноситься до всіх об'єктів КС. Перш ніж користувач або процес зможе одержати в своє розпорядження звільнений іншим користувачем або процесом об'єкт, встановлені для попереднього користувача або процесу права доступу до даного об'єкта будуть скасовані. Перш ніж користувач або процес зможе одержати в своє розпорядження звільнений іншим користувачем або процесом об'єкт, вся інформація, що міститься в даному об'єкті, стає недосяжною.

Ця послуга реалізується завдяки вилученню створених сесій відразу після кінцевої стадії її використання передбаченою КЗЗ.

2.1.2 Критерій цілісності

Цілісність забезпечується дотриманням вимог політики безпеки щодо переміщення інформації до об’єкта з боку користувача або процесу. Правильне (допустиме) переміщення визначається як переміщення інформації до об’єкта від авторизованого користувача або процесу.

ЦА-1. Мінімальна адміністративна цілісність.

Ця послуга дозволяє керувати потоками інформації від користувачів до захищених об’єктів WEB-сторінки. Політика мінімальної адміністративної цілісності стосується: користувачів усіх категорій; загальнодоступної інформації WEB-сторінки; файлової системи та функціонального ПЗ, що використовується для актуалізації, захисту загальнодоступної інформації та супроводження WEB-сторінки; створеної в процесі супроводження WEB-сторінки технологічної інформації КСЗІ та технологічної інформації щодо управління АС.

КЗЗ здійснює розмежування доступу на підставі атрибутів доступу користувачів і захищених об’єктів. Розмежування доступу здійснюється на рівні надання (встановлення заборони) користувачеві прав модифікувати об’єкт.

Право визначати множину об'єктів АС, цілісність яких забезпечується КЗЗ, надається адміністратору безпеки.

Права доступу до захищених об’єктів WEB-сторінки встановлюються в момент їх створення або ініціалізації.

ЦО-1. Відкат

Ця послуга забезпечує можливість відмінити окрему операцію або послідовність операцій і повернути захищений об'єкт після внесення до нього змін до попереднього наперед визначеного стану.

Політика обмеженого відкату стосується користувачів, яким надано право супроводження КСЗІ та управління АС; об’єктів, які містять публічну інформацію; функціонального програмного забезпечення, що використовується для актуалізації, захисту публічної інформації та супроводження WEB-сторінки; створеної в процесі супроводження WEB-сторінки технологічної інформації КСЗІ та технологічної інформації щодо управління АС. Якщо стосовно якогось з об’єктів зазначених категорій в процесі обробки не передбачається можливості його модифікації, політика послуги на нього не розповсюджується.

Ця послуга забезпечується завдяки використанню резервних копій даних.

ЦВ-1. Цілісність при обміні

Ця послуга дозволяє забезпечити захист WEB-сторінки від несанкціонованої модифікації інформації, яка передається між WEB-сервером та робочими станціями у разі використання технології Т2, під час експорту/імпорту інформації через незахищене середовище. Політика послуги стосується всіх об’єктів, що передаються.

КЗЗ забезпечує контроль за цілісністю інформації в повідомленнях, які передаються, а також виявляє факти їх несанкціонованого видалення або дублювання. КЗЗ забезпечує можливість реєстрації подій, які призвели до порушення цілісності повідомлень, їх несанкціонованого видалення або дублювання.

Ця послуга забезпечується використанням протоколу HTTPS, що є модифікацією базового протоку передачі даних HTTP, який використовує криптографічні протоколи TSL и SSL та TCP-порт 443.

2.1.3 Критерій доступності

КЗЗ оцінюваної КС надає послуги щодо забезпечення можливості використання КС в цілому, окремих функцій або оброблюваної інформації на певному проміжку часу і гарантує спроможність КС функціонувати у випадку відмови її компонентів. Доступність забезпечується в КС такими послугами: використання ресурсів, відновлення після збоїв.

ДР-1. Використання ресурсів

Ця послуга дозволяє керувати використанням користувачами послуг та ресурсів.

Політика використання ресурсів, що реалізується КЗЗ, стосується: користувачів загальнодоступної інформації; адміністратора безпеки та користувачів, яким надано повноваження щодо управління АС; файлової системи; системного та функціонального програмного забезпечення; технологічної інформації щодо управління АС; окремих периферійних пристроїв (принтерів, накопичувачів інформації та ін.); обчислювальних ресурсів АС і передбачає можливість встановлення обмежень на їх використання.

Обмеження щодо використання окремим користувачем або процесом обсягів обчислювальних ресурсів АС або кількості об'єктів встановлюються адміністратором безпеки або користувачами, яким надано повноваження щодо управління АС. Запити на зміну встановлених обмежень повинні оброблятися КЗЗ тільки в тому випадку, якщо вони надходять від зазначених користувачів.

Ця послуга забезпечується завдяки використанню системи тестування користувачів на основі зворотного тесту Тюрінга.

ДВ-1. Відновлення після збоїв

Політика відновлення після збоїв, що реалізується КЗЗ, стосується: системного та функціонального програмного забезпечення; засобів захисту інформації та засобів управління КСЗІ; засобів адміністрування та управління обчислювальною системою АС – і гарантує повернення АС у відомий захищений стан після відмов або переривання обслуговування, спричинених помилковими діями користувачів, неврахованою функціональною недостатністю програмного та апаратного забезпечення (наприклад, можливою наявністю не виявлених під час проєктування незадекларованих функцій), іншими непередбачуваними ситуаціями.

Політика відновлення, яка реалізується КЗЗ, визначає множину типів відмов WEB-сторінки і переривань обслуговування, після яких можливе повернення у відомий захищений стан без порушення політики безпеки. Для кожної з відмов чітко визначені і задокументовані рівні відмов, у разі перевищення яких необхідна повторна інсталяція WEB-сторінки.

Ця послуга реалізується за допомогою включення до АС додаткових резервних WEB-серверів, що є копіями основних WEB-серверів, за допомогою яких можна відновити нормальну роботу системи.

2.1.4 Критерій спостереженості

КЗЗ надає послуги щодо забезпечення відповідальності користувача за свої дії і щодо підтримки спроможності КЗЗ виконувати свої функції. Спостереженість забезпечується в КС такими послугами: реєстрація, ідентифікація і автентифікація, достовірний канал, розподіл обов'язків, цілісність КЗЗ, самотестування, ідентифікація і автентифікація при обміні, автентифікація відправника, автентифікація отримувача.

НР-2. Реєстрація

Ця послуга дозволяє контролювати небезпечні відповідно до політики безпеки WEB-сторінки дії користувачів всіх категорій із захищеними об’єктами.

Політика реєстрації стосується: користувачів усіх категорій; публічної інформації WEB-сторінки; системного та функціонального програмного забезпечення, що використовується для актуалізації, захисту публічної інформації та супроводження WEB-сторінки; створеної в процесі супроводження WEB-сторінки технологічної інформації КСЗІ та технологічної інформації щодо управління АС. КЗЗ повинен забезпечувати реєстрацію всіх подій, які мають безпосереднє відношення до безпеки.

Реєстрація всіх подій, що мають безпосереднє відношення до безпеки, здійснюється в журналі реєстрації, який містить інформацію стосовно дати, часу, місця, типу і наслідків зареєстрованої події (успішність/неуспішність), ім'я (IP-адресу) та ідентифікатор причетного до цієї події користувача. Реєстраційна інформація повинна бути достатньою для однозначної ідентифікації користувача, процесу або об'єкта, що мали відношення до кожної зареєстрованої події.

НИ-2. Ідентифікація і автентифікація

Ідентифікація і автентифікація дозволяють КЗЗ визначити і перевірити особу суб'єкта, що намагається одержати доступ до захищених об'єктів WEB-сторінки.

Політика ідентифікації і автентифікації стосується: всіх користувачів WEB-сторінки, які намагаються одержати доступ до системного та функціонального програмного забезпечення, що використовується для актуалізації, захисту публічної інформації та супроводження WEB-сторінки; створеної в процесі супроводження WEB-сторінки технологічної інформації КСЗІ та технологічної інформації щодо управління АС; задіяного для цього периферійного обладнання.

КЗЗ однозначно ідентифікує категорії користувачів WEB-сторінки і за атрибутами кожної з цих категорій визначати послуги, що їм доступні. Ідентифікація здійснюється на підставі ідентифікатору сесії користувача.

НО-1. Розподіл обов'язків

Ця послуга дозволяє розмежувати повноважень користувачів, визначивши категорії користувачів з певними і притаманними для кожної з категорій функціями (ролі). Послуга призначена для зменшення потенційних збитків від навмисних або помилкових дій користувачів і обмеження авторитарності керування АС.

Політика розподілу обов'язків, що реалізується КЗЗ, стосується користувачів усіх категорій і визначає такі ролі:

– адміністратор безпеки;

– користувач, якому надано право доступу до певних видів інформації (публічної, технологічної, системного та функціонального ПЗ).

Кількість користувачів, які мають доступ до технологічної інформації та системного і функціонального ПЗ мінімізована, щоб обмежити їх коло тільки тими, кому необхідний такий доступ для виконання функціональних обов'язків, що передбачаються експлуатаційною та розпорядчою документацією на WEB-сторінку.

Розмежування повноважень користувачів здійснюється на підставі атрибутів повноважень і захищених об'єктів.

НЦ-1. Цілісність комплексу засобів захисту

Ця послуга визначає міру здатності КЗЗ WEB-сторінки захищати себе і гарантувати свою спроможність керувати захищеними об'єктами.

Політика цілісності КЗЗ визначає склад КЗЗ, механізми контролю цілісності його компонентів та порядок їх використання.

Політика цілісності КЗЗ стосується: адміністратора безпеки; окремих компонентів системного та функціонального програмного забезпечення, які задіяні для реалізації механізмів КЗЗ; засобів захисту інформації, а також технологічної інформації КСЗІ – і забезпечує взаємодію зазначених об'єктів.

Політика реалізації послуги гарантує, що всі послуги безпеки доступні тільки через інтерфейс КЗЗ і всі запити на доступ до захищених об'єктів

контролюються КЗЗ. Обмеження, недотримання яких може призвести до надання послуг в обхід інтерфейсу КЗЗ і порушення цілісності КЗЗ, описуються і документуються.

НТ-1. Самотестування

Самотестування дозволяє КЗЗ перевірити і на підставі цього гарантувати правильність функціонування і цілісність певної множини функцій захисту WEB-сторінки.

Політика самотестування поширюється на адміністратора безпеки, компоненти системного та функціонального програмного забезпечення, які задіяні для реалізації механізмів КЗЗ, засоби захисту інформації.

До складу КЗЗ входить множина тестових процедур, яка враховує особливості функціонування компонентів конкретної WEB-сторінки і достатня для оцінки правильності виконання всіх критичних для безпеки публічної та технологічної інформації КСЗІ функцій, а сам КЗЗ здатен контролювати їх виконання.

Ця послуга реалізується завдяки ПЗ Xspider, Nmap, Nikto, які мають у своїй базі даних велику кількість тестових запитів для тестування WEB-сервера через користувача.

НВ-1. Ідентифікація і автентифікація при обміні

Ця послуга дозволяє компонентам КЗЗ WEB-сервера і віддаленої робочої станції здійснити взаємну ідентифікацію, перш ніж розпочати взаємодію.

КЗЗ надає доступ до процесів, що забезпечують ініціалізацію обмін даними, тільки адміністратору безпеки і користувачам, яким надано повноваження щодо супроводження WEB-сторінки.

Обмін інформацією між компонентами КЗЗ здійснюється тільки після ідентифікації і автентифікації КЗЗ-відправником КЗЗ-отримувача інформації. Результати процедури ідентифікації і автентифікації є дійсними протягом всього сеансу обміну (незалежно від кількості об’єктів, що експортуються) і втрачають свою силу після його закінчення.

Процедура ідентифікації і автентифікація компонентів КЗЗ здійснюється на підставі ідентифікатору сесії.

Підтвердження ідентичності виконується на підставі затвердженого в АС протоколу автентифікації.

2.2 Аналіз загроз WEB-серверів

По відношенню до інформації, що обробляється на WEB-серверах, та до користувачів WEB-серверів існує ряд загроз відображених в таблиці 2.3.

Таблиця 2.3 – Загрози безпеки інформації WEB-серверів

Загроза	Вплив
1	2
Антропогенні загрози	
Порушення властивостей цілісності інформації в наслідок несанкціонованої модифікації даних зловмисником	На інформацію, до якої отримає доступ зловмисник
Порушення властивостей конфіденційності інформації в наслідок захоплення облікових записів користувачів WEB-серверу	На інформацію, до якої отримає доступ зловмисник
Порушення властивостей доступності інформації в наслідок підвищення навантаження на WEB-сервер	На інформацію, яка призначена до розповсюдження
Розповсюдження незапрошеної інформації	На користувачів WEB-серверу
Техногенні загрози	
Відмови і збої обладнання, яке приймає, обробляє, зберігає й передає інформацію	На інформацію, яка обробляється обладнанням, що відмовило
Відмови і збої активного мережевого обладнання	На інформацію, яка передавалася обладнанням, що відмовило

Продовження таблиці 2.3

1	2
Саботаж роботи інформаційної системи за допомогою програмних засобів (віруси, шкідливі скрипти та ін.)	На інформацію, до якої вдасться отримати доступ шкідливій програмі
Стихійні загрози	
Руйнування технічних засобів, що обробляють інформацію, внаслідок стихійних впливів (землетруси, пожежі, електромагнітні поля та блискавки)	На інформацію, що обробляється засобами, які піддаються зовнішнім впливам

2.3 Переваги та недоліки видів зворотного тесту Тюрінга

В таблиці 2.4 наведені характерні для зазначених видів зворотного тесту Тюрінга переваги та недоліки.

Таблиця 2.4 – Переваги та недоліки видів зворотного тесту Тюрінга

Переваги	Недоліки
1	2
Класична тест Тюрінга	
1 Простота в реалізації; 2 Простота інтеграції до WEB-сайту.	1 Використання сторонніх сервісів для розпізнання тест-зображення людьми; 2 Використання OCR; 3 Незручності для людей з порушенням зору.
Аудіо-CAPTCHA	
1 Можливість відповісти на тест-питання користувачеві з порушенням зору.	1 Користувач повинен мати устаткування для відтворення звуку; 2 Наявність бази фрагментів звукової інтерпретації символів; 3 Реалізація звукових спотворень досить вимоглива до кваліфікації програміста і ресурсів сервера.

Продовження таблиці 2.4

1	2
Математична CAPTCHA	
1 Невелике розповсюдження; 2 Простота в реалізації; 3 Простота в інтеграції до WEB-сайту.	1 Подразливість від проходження тесту; 2 Користувач повинен добре володіти арифметикою; 3 Використання спеціальних знаків.
Текстові завдання	
1 Можливість надання відповіді на тест-питання користувачеві з порушенням зору; 2 Можливість надання відповіді користувачам, які використовують Інтернет-браузери, що не відображають зображення.	1 Користувач має бути добре знайомий з мовою, на якій задаються питання; 2 Кінцева кількість бази питань.
Тест на розпізнання предметів	
1 Легше розрізнення образів людиною, наприклад (кішки від собаки), ніж спотворених та зашумлених букв «N» і «H»	1 Користувач має бути ознайомленим з об'єктами та їх назвами; 2 Кінцева кількість бази картинок з об’єктами.
Тест на почуття прекрасного	
1 Базування рішень на «персонофікації»; 2 Невелике розповсюдження.	1 Суб’єктивність оцінки «прекрасності»; 2 Складність генерації завдань за співвідношеннями або наявність кінцевої бази зображень.
Тривимірна-CAPTCHA	
1 Неможливість використання OCR без попередньої підготовки зображення.	1 З використанням засобів зашумлення фону може бути важко зрозуміла користувачеві;

Продовження таблиці 2.4

1	2
	2 Використання сторонніх сервісів для розпізнання тест-зображення людьми.
Рухома відео-CAPTCHA	
1 Використання рухомих об'єктів; 2 Існування додаткової системи захисту.	1 Використання в рухомому рядку тільки англійських літер; 2 Слабка додаткова система захисту; 3 Використання Adobe Flash; 4 Складність реалізації.
Рухома GIF-CAPTCHA	
1 Використання рухомих об'єктів.	1 Можливість розділення «складного» зображення типу GIF на більш прості кадри; 2 Розпізнання символів є таким самим, як і в класичному зворотному тесті Тюрінга, за винятком розпізнання окремих кадрів.

2.4 Способи обходу зворотного тесту Тюрінга

Для кожного з видів зворотного тесту Тюрінга існують способи обходу, таблиця 2.5.

Таблиця 2.5 – Алгоритм створення набору цифр для тестового завдання

Види зворотного тесту Тюрінга	Способи обходу
1 Класичний зворотний тест Тюрінга; 2 Математична CAPTCHA; 3 Тривимірна-CAPTCHA; 4 Рухома відео-CAPTCHA; 5 Рухома GIF-CAPTCHA.	1 Використання OCR; 2 Передача задачі розпізнання спеціальному сервісу.
1 Текстові завдання; 2 Тест на розпізнавання предметів; 3 Тест на почуття прекрасного.	1 Використання бази вірних відповідей.
1 Аудіо-CAPTCHA.	1 «Накладання маски» аудіо-фрагменту.

2.4.1 Використання OCR

Використання OCR, тобто «чесне» розпізнання тексту, можливе лише при використанні зворотного тесту Тюрінга, який оснований на введені символів з тест-зображення. Для обходу зворотного тесту Тюрінга за допомогою OCR, як правило, не використовується універсальне ПЗ розпізнання відсканованого тексту типу FineReader чи MS Office Document Imaging. Слід розуміти, якщо CAPTCHA не піддається розпізнанню цим ПЗ, це не означає, що символи на тест-зображенні не можуть бути легко розпізнані ПЗ, розробленим саме під неї.

Розпізнання ділиться на 2 основних етапи:

1 Визначення місцезнаходження та границь кожного символу;

2 Безпосереднє розпізнання символу.

Якщо символи завжди знаходяться на одних й тих же місцях (як наприклад, у форумі Invision Power Board – рисунок 2.1), з цих двох етапів залишається лише другий. Тому, як мінімум, для забезпечення захисту CAPTCHA, необхідно варіювати координати символів.

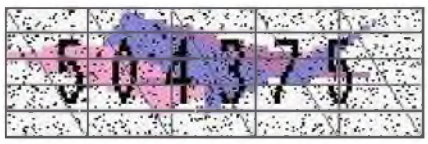

Рисунок 2.1 – CAPTCHA форуму Invision Power Board

Якщо місця символів не фіксовані, наступний шлях по їх виділенню - порівняння по контрасту з фоном. Якщо колір символів дуже відрізняється від кольору фону (як наприклад, у форумі phpBB – рисунок 2.2), виділення окремих символів в такому разі не є складною задачею, тому що попіксельне порівняння надає змогу отримати контури символу, а значить й сам символ.

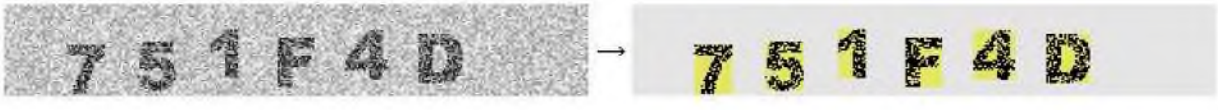

Рисунок 2.2 – CAPTCHA форуму phpBB

Способи розпізнання діляться на два види:

1 Накладання «маски»;

2 Розпізнання символів по їх характерним ознаками.

Суть «накладання маски» полягає в тому, що в розпізнавальному ПЗ знаходиться еталонний шрифт, з котрим порівнюється символи відображені на тест-зображенні. Той символ, у якого більше всього збігів пікселів, що й в еталоні, й вважається вірним для відповідного символу тестового зображення.

Порівнянню з маскою схильна CAPTCHA, в якій не використовується злиття символів. Якщо CAPTCHA має геометричні спотворення символів, то попередньо до накладання маски виконується геометрична корекція символів.

Розпізнання символів по їх характерним ознаками є більш складними алгоритмами розпізнання. В цих алгоритмах символи, розпізнаються по таким признакам як: кількість розгалужень, замкнутих областей, їх взаємне розташування. Ці алгоритми використовують нейронні мережі, які в першу чергу треба навчити, даючи тестові-образи та відповіді на них. Після етапу навчання цей алгоритм зможе давати правильні відповіді сам. Однак слід зазначити, що процес навчання досить довгий та трудомісткий.

2.4.2 Передача задачі розпізнання спеціальному сервісу

Суть даного способу полягає в передачі тестового зображення на спеціальні сервіси розпізнання. Ці сервіси використовують працю інших людей, які за кожну вірно розпізнане тест-зображення отримують кошти.

В основному ці сервіси працюють за наступним алгоритмом:

1 Отримання тест-зображення від користувача сервісу;

2 Видача ID замовлення на розпізнання;

3 Розпізнання тест-зображення;

4 Видача в текстовому вигляді результату розпізнання користувачеві з відповідним ID.

Можна виділити наступні сервіси розпізнавання CAPTCHA:

– http://anti-captcha.com/ – сервіс з великою кількістю операторів з розпізнавання. Процес розпізнання на даному сервісу досить швидкий, і в середньому займає менше хвилини. Вартість 1000 правильно розпізнаних тест-зображень 1$. Якість розпізнання 90-95%.

– http://captchabot.com/ – сервіс для розпізнання тестового зображення в автоматичному режимі. Сервіс працює з усіма платформами, які вимагають розпізнання тестового зображення. Є підтримка багатозадачності. Якість розпізнання на рівні 80%. Є можливість інтеграції з будь-яким ПЗ, який так чи інакше пов'язаний з визначенням символів на тест-зображенні. Швидкість розпізнання – до 30 секунд.

2.4.3 Використання бази вірних відповідей

Суть даного способу полягає в формуванні всіх вірних відповідей на тестові завдання. В наступному ця база відповідей надається ПЗ розробленому для проходження зворотного тесту Тюрінга, який використовує базу тестових завдання. Використання великої бази тестових завдань може дещо ускладнити процес формування вірних відповідей зловмиснику, так як він повинен на початку використання WEB-роботу пройти всі можливі тестові завдання.

2.4.4 Накладання «маски» аудіо-фрагменту

Суть накладання маски аудіо-фрагменту є такою самою, як і в накладанні «маски» на тест-зображення. Однак для порівняння аудіо-CAPTCHA використовуються еталонні аудіо-фрагменти символів з ПЗ розробленого зловмисником.

На рисунку 2.3 зображений фрагмент звукового файлу, що відповідає тест-зображенню – рисунок 2.4 (новинний WEB-сайт http://digg.com).

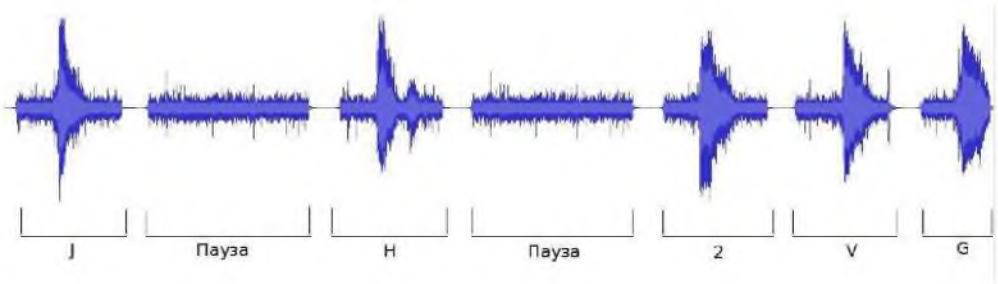

Рисунок 2.3 – Звуковий спектр аудіо-CAPTCHA

Рисунок 2.4 – Тест-зображення

Створення бази аудіо-фрагментів дозволить зловмиснику порівнювати їх з фрагментами аудіо файлу, що надається користувачеві для прослуховування.

2.5 Розробка система тестування користувачів на основі зворотного тесту Тюрінга

Розроблена система тестування користувачів на основі зворотного тесту Тюрінга відноситься до засобу протидій автоматизованому виконанню зловмисних дій, який вимагає від користувача виконання дії щодо сортування наданого йому набору цифр.

2.5.1 Опис системи тестування

Ідея створеної системи тестування полягає в тому, що користувачеві надається набір цифр, та пропонується відсортувати його за зростанням. При цьому не використовуються засоби зашумлення чи деформації. Відмова від засобів зашумлення та деформації обумовлена тим що, процес тестування не основується на розпізнанні образів. Це дозволяє знизити подразливість при проходженні тестового завдання людиною, прискорити процес тестування, та надати можливість виконання тесту користувачам з порушенням зору.

На рисунку 2.1 зображений набір цифр, який використовується в системі тестування:

Рисунок 2.5 – Набір цифр системи тестування користувачів

Сортування наданого набору цифр виконується з використанням маніпулятору типу «миш» чи тачпад або за допомогою пальців рук при використанні користувачем пристроїв з сенсорним екраном, наприклад: смартфони, неттопи, планшетні ПК.

2.5.2 Програмні засоби для створення системи тестування

Для створення системи тестування користувачів використовується найбільш розповсюджена серверна скриптова мова програмування – PHP v5.3.6; мова розмітки XHTML 1.0 Strict; скрипкова мова JavaScript з бібліотекою drag_n_drop.js.

Для використання системи тестування в якості WEB-серверу виступає – Apache Web Server 2.2.19.

Для створення зображень набору символів використовується ПЗ для роботи з зображенням GIMP v2.6.11.

2.5.3 Алгоритм створення набору цифр для тестового завдання

Алгоритм створення набору цифр для тестового завдання складається з наступних етапів, наведених в таблиці 2.6.

Таблиця 2.6 – Алгоритм створення набору цифр для тестового завдання

	Етап	Результат
	Ініціалізація одномірного масиву з цифрами від 0 до 9	Array ([0] => **0** [1] => **1** [2] => **2** [3] => **3** [4] => 4 [5] => **5** [6] => **6** [7] => **7** [8] => **8** [9] => **9**)
	Перемішування масиву. Функція PHP – shuffle()	Array ([0] => **7** [1] => **5** [2] => **0** [3] => **6** [4] => **4** [5] => **1** [6] => **9** [7] => **2** [8] => **8** [9] => **3**)
	Визначення початкового розташування елементів тестового завдання	Array ([0] => **7** [1] => **5** [2] => **0** [3] => **6**)
	Отримання вірного розташування елементів тестового завдання. Функція PHP – asort()	Array ([2] => **0** [1] => **5** [3] => **6** [0] => **7**)
	Визначення вірних позицій для елементів, що є кодом перевірки. Функція PHP – array_keys()	Array ([0] => **2** [1] => **1** [2] => **3** [3] => **0**)

Примітка. [ ] – номер елементу в масиві.

1 На етапі 1 виконується ініціалізація одномірного масиву (М), розмірністю [10] з цифрами від 0 до 9;

2 На етапі 2 виконується перемішування масиву (М);

3 На етапі 3 виконується вибір перших 4-х елементів масиву (М) та запис їх до масиву (К);

4 На етапі 4 виконується сортування масиву (К) з збереженням ключів – отримання вірного розташування елементів тестового завдання (Р);

5 На етапі 5 виконується вибір ключів масиву (P), що визначає вірні позицій для елементів масиву (K), що і є правильною відповіддю (D).

В результаті завершення алгоритму буде використовуватися:

1 В якості початкового стану тесту – порядок цифр з етапу 3 рисунок 2.9;

2 В якості вірної відповіді на тест – порядок цифр з етапу 4 рисунок 2.7.

Рисунок 2.6 – Початковий стан тесту

Рисунок 2.7 – Вірна відповідь на тест

В таблиці 2.7 показане відношення порядку цифр до позицій в початковому стані тесту та в результаті правильної відповіді.

Таблиця 2.7 – Відношення порядку цифр до їх позицій

	Порядок				Позиція			
Початковий стан тестового завдання.								
Вірна відповідь на тестове завдання.								

2.5.4 Алгоритм перевірки користувача системи тестування

Алгоритм перевірки користувача налічує два основних етапи:

– етап створення тестового завдання;

– етап перевірки результату проходження користувачем тестового завдання.

Етап створення тестового завдання включає в створення сесії для зберігання вірної відповіді та створення набору цифр для тестового завдання, алгоритм створення описаний в розділі 2.6.3.

Етап перевірки результату проходження користувачем тестового завдання включає перевірку наявності відповідної до ідентифікатору користувача сесії на WEB-серверів та безпосередню перевірку відповіді

користувача та еталонної відповіді на тестове завдання яка знаходиться в створеній сесії з відповідним ідентифікатором.

Повний алгоритм перевірки користувача відображений на рисунку 2.8.

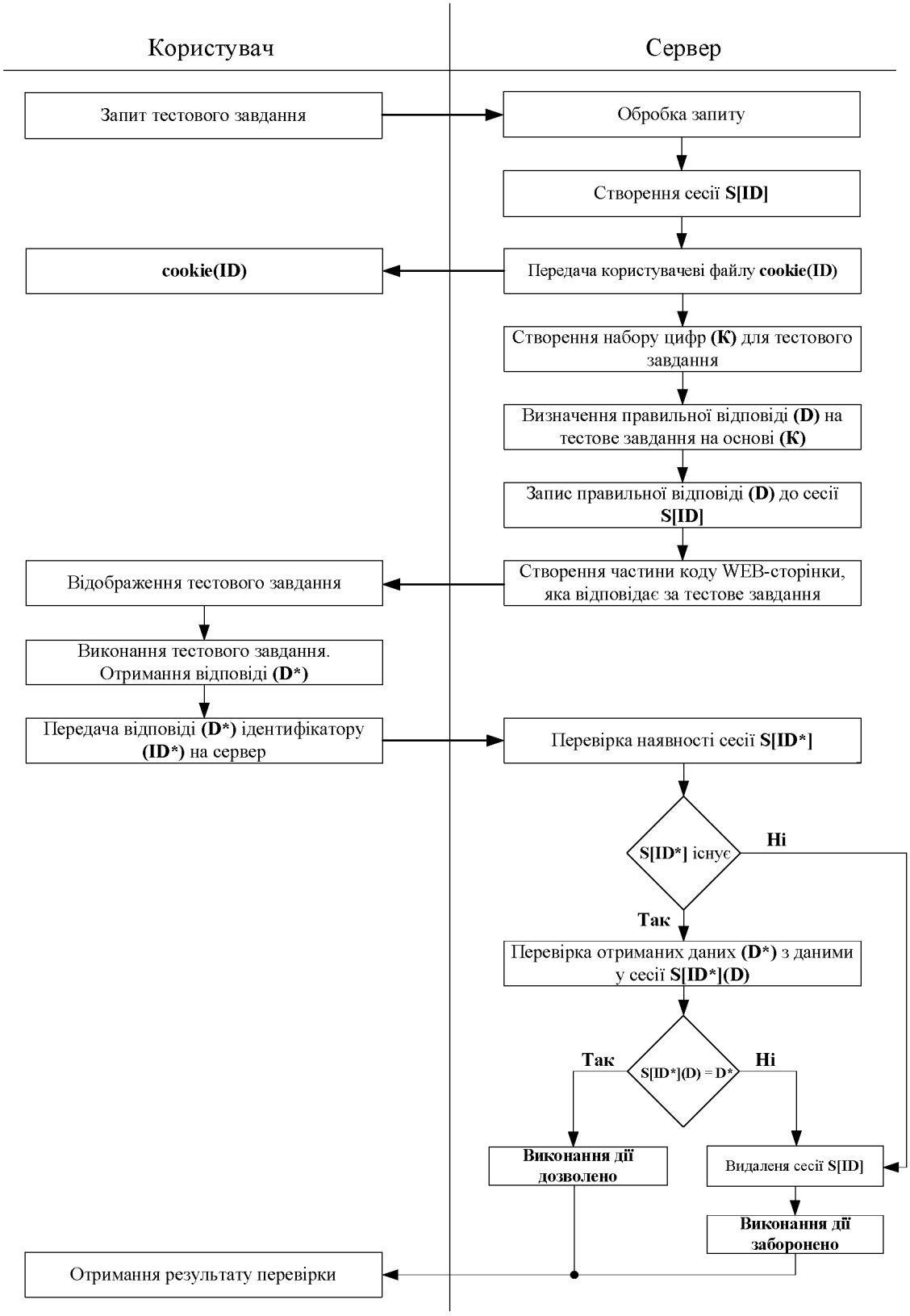

Рисунок 2.8 – Алгоритм перевірки

2.5.5 Файлова структура системи тестування

На рисунку 2.9 зображена файлова структура системи тестування користувачів.

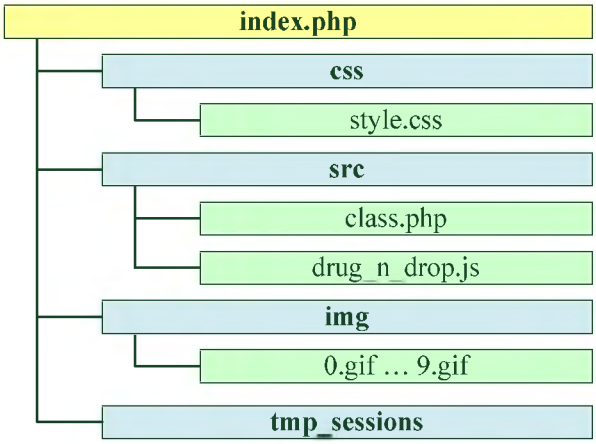

Рисунок 2.9 – Файлова структура системи тестування користувачів

1 index.php – головна сторінка, на якій безпосередньо виконується тестування користувача. Вона розташована в корні каталогу «/»;

2 style.css – файл стилю, який відповідає за відображення тесту та сторінки index.php в цілому. Розташування – директорія «css»;

3 class.php – файл безпосереднього створення тесту, перевірки правильності його проходження та видалення сесії. Розташування – директорія «src»;

4 drug_n_drop.js – бібліотека організації перенесення елементів на сторінці. Розташування – директорія «src»;

5 0.gif … 9.gif – файли зображення елементів, які беруть безпосередню участь в тестовому завданні. Розташування – директорія «img»;

6 Директорія tmp_sessions призначена для тимчасового зберігання сесій користувачів.

На рисунку 2.10 відображена взаємодія файлів системи тестування.

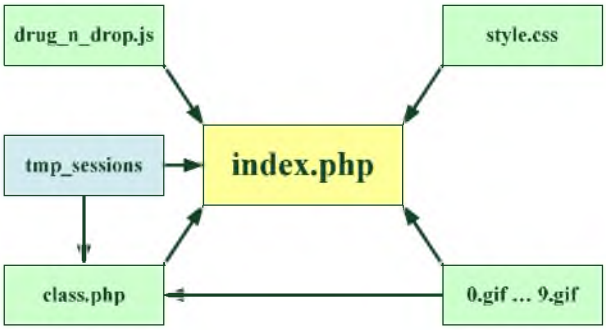

Рисунок 2.10 – Взаємодія файлів системи тестування користувачів

2.6 Загальний вигляд системи тестування

В результаті виконання скрипту index.php, який містить в собі посилання на інші файли системи тестування, початковий код яких приведений у Додатку Б, початковий код сторінки index.php буде мати вигляд:

Лістингу 2.1 – Початковий код сторінки index.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
 <title>Система тестування користувачів WEB-серверу</title>

 <link rel="stylesheet" href="css/style.css" type="text/css" />

 <script type="text/javascript" src="src/drag_n_drop.js"></script>

 <script type="text/javascript">
 <!--
  function preloadCaptcha()
  {
   new DragObject(document.getElementById('c_n1'));
   new DragObject(document.getElementById('c_n2'));
   new DragObject(document.getElementById('c_n3'));
   new DragObject(document.getElementById('c_n4'));
   new DropTarget(document.getElementById('captcha_field'));
   var temp;

   for(var i=1;i<=4;i++)
   {
    temp=(ObjectPosition(document.getElementById('captcha_field'))[0]+(i-1)*35+5);
    document.getElementById('c_n'+i).style.left=temp+'px';

document.getElementById('c_n'+i).style.top=(ObjectPosition(document.getElementById('captcha_fi
eld'))[1]+10)+'px';
```

Продовження лістингу 2.1

```
document.getElementById('coord'+i).value=temp;
  }
 }
-->
</script>
</head>

<body onload="preloadCaptcha();">
 <div id="form">
  <form action="src/class.php" method="POST">

   <h1>Ім'я користувача:</h1> <input id="inp" type="text" name="name"/>
   <div id="clear"></div>

 <h2>Розташуйте цифри в порядку зростання, перетягуючи їх!</h2>
   <div id="clear"></div>

<div id='captcha_field'>
  <img src='img/8.gif' id='c_n1' style='position:absolute;'/>
  <img src='img/3.gif' id='c_n2' style='position:absolute;'/>
  <img src='img/0.gif' id='c_n3' style='position:absolute;'/>
  <img src='img/7.gif' id='c_n4' style='position:absolute;'/>
 </div>

 <input type='hidden' name='coords[]' id='coord1'>
 <input type='hidden' name='coords[]' id='coord2'>
 <input type='hidden' name='coords[]' id='coord3'>
 <input type='hidden' name='coords[]' id='coord4'>

   <div id="clear"></div>

   <input id="button" type="submit" name="verification" value="Перевірити"/>
  </form>
 </div>

</body>

</html>
```

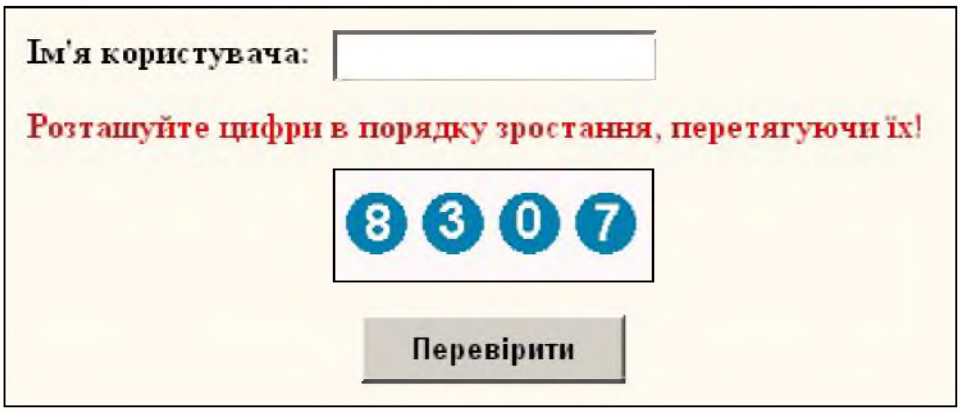

Рисунок 2.11 – Форма тестового завдання

На рисунку 2.11 зображена форма тестового завдання, відображена з використанням Інтернет-браузеру Opera 11.01. Вид форми представлений для ознайомлення і може бути змінений.

В результаті використання системи тестування в директорії tmp_sessions на WEB-серверів створюється тимчасовий файл сесії, рисунок 2.12.

Файл сесії дійсний до моменту закриття вікна Інтернет-браузеру чи до моменту проходження тесту. Тобто, при неправильній відповіді на тестове завдання поточна сесія буде закрита та видалена і створена нова.

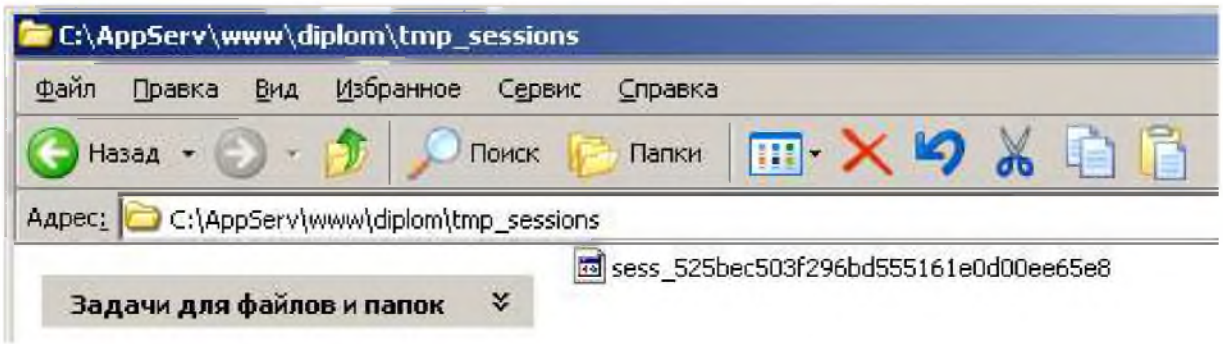

Рисунок 2.12 – Тимчасовий файл сесії

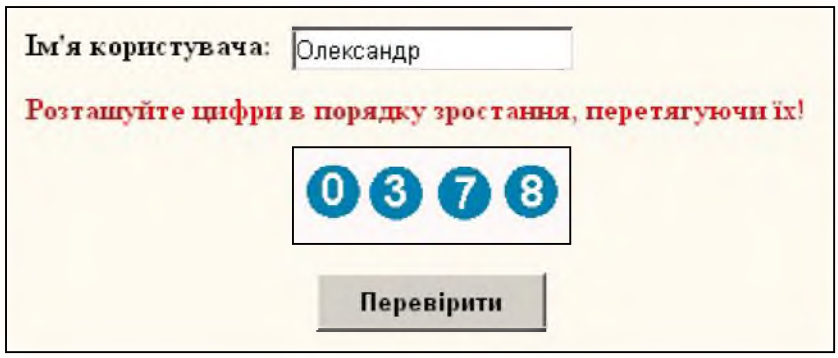

Рисунок 2.13 – Проходження тестового завдання

Внаслідок успішного проходження тестового завдання (рисунок 2.14) може бути виконане створення облікового запису, отриманий дозвіл на завантаження файлу, додання коментарю та інші дії, які передбачені адміністратором WEB-сайту.

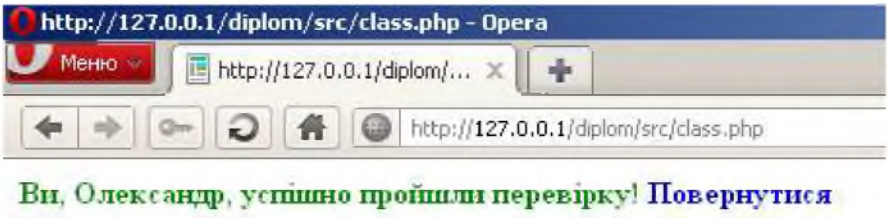

Рисунок 2.14 – Успішне проходження тестового завдання

В прикладі системи тестування після проходження тесту сесія у будь-якому випадку видаляється. В реальних умовах час життя сесії буде змінений, сесія буде існувати, доки не буде закінчена операція, яка передбачає її наявність.

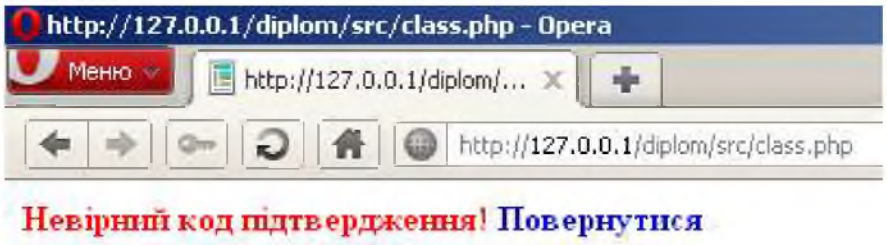

Рисунок 2.15 – Невдале проходження тестового завдання

На рисунку 2.15 зображене невдале проходження тестового завдання.

У Додатку В наведені приклади використання системи тестування в інших Інтернет-браузерах. Створена система тестування користувачів повністю відповідає стандартам W3C.

2.7 Переваги та недоліки створеної системи тестування користувачів

Розроблена система тестування користувачів, яка ґрунтується на зворотному тесті Тюрінга має наступні переваги та недоліки, які наведені в таблиці 2.8.

Таблиця 2.8 – Переваги та недоліки створеної системи тестування

Переваги	Недоліки
Система тестування користувачів	
1 Виключення можливості використання існуючих способів обходу притаманних іншим видам зворотного тесту Тюрінга; 2 Використання рухомих об’єктів; 3 Відносно невеликий розмір; 4 Надання можливості проходження тесту користувачам з порушенням зору.	1 Використання JavaScript.

В порівнянні з існуючими системами тестування користувачів створена система має наступні переваги та недоліки:

Переваги

Не використовуючи тестові зображення, бази з вірними відповідями та аудіо фрагменти, створена система тестування стає захищеною від існуючих способів обходу, яким піддаються системи тестування зазначені в таблиці 2.5, описаних у розділі 2.5.

Використання рухомих об’єктів, а саме, переміщення наданих елементів системи тестування користувачем за допомогою маніпуляторів, дає змогу забезпечити швидше проходження тесту легітимним користувачам, а також збільшити надійність системи тестування завдяки задачі розміщення елементів, а не простому уведенню коду підтвердження.

Невеликий розмір даної системи тестування у порівнянні з зворотним тестом Тюрінга, основаним на розпізнаванні предметів. Розміри порівняні з класичним зворотним тестом Тюрінга.

Використовуючи чіткі символи (рисунок 2.5) та не використовуючи засоби зашумлення, створена система дає змогу вільно використовувати її користувачам з порушенням зору, а також знизити напруження зору для користувача в цілому.

Недоліки

Всі сучасні Інтернет-браузери підтримують JS, однак якщо у користувача створеної системи тестування буде відключене чи примусово заблоковане використання JS, то даний користувач не зможе пройти тест, а отже, не зможе виконати дії, які захищаються даною системою.

2.8 Висновок

Результатом проведеної роботи в даному розділі став вибір профілю захищеності для WEB-серверів, аналіз загроз WEB-серверів, огляд можливих способів обходу зворотного тесту Тюрінга для кожного з видів, створення узагальненої таблиця переваг та недоліків видів зворотного тесту Тюрінга, та розробка удосконалена системи тестування користувачів.

Розроблена система тестування користувачів основана на зворотному тесті Тюрінга. Вона була створена з використанням найбільш розповсюдженої серверною скриптової мова програмування – PHP v5.3.6 та JavaScript бібліотека drag_n_drop.js.

Створена системи тестування користувачів виключає можливості використання існуючих способів обходу, притаманних іншим видам зворотного тесту Тюрінга. Створена система захисту від автоматизованого виконання зловмисних дій, дає змогу користувачам з порушенням зору безперешкодно, уразі правильного проходження тестування, створювати облікові записи на різних WEB-сайтах, так як не використовує засобів спотворення та зашумлення.

РОЗДІЛ 3. ЕКОНОМІЧНА ЧАСТИНА

3.1 Необхідність обґрунтування витрат на реалізацію політики безпеки

Завдяки запропонованій в кваліфікаційній роботі системі тестування користувачів, яка ґрунтується на зворотному тесті Тюрінга вирішується питання захисту від автоматизованого виконання зловмисних дій на WEB-серверах, що є основою для інформаційної розвідки в великих масштабах, розповсюдження незапрошеної інформації, захопленню облікових записів користувачів та інших. Система тестування користувачів являє собою ПЗ без використання технічних засобів.

Метою даного розділу є розрахунки економічної ефективності впровадження цих рекомендацій та підтримки їх виконання в ТОВ «ІнфоАналітика». Запропонована політика інформаційної безпеки включає, але не обмежується розглянутою системою.

Для розрахунку вище вказаного необхідно:

– розрахувати капітальних витрат на реалізацію запропонованих рекомендацій;

– розрахувати річні експлуатаційні витрати на виконання рекомендацій;

– сума річних амортизаційних відрахувань на апаратні засоби, необхідні для виконання рекомендацій;

– показники економічної ефективності впровадження системи захисту на підприємстві.

3.2 Розрахунок капітальних витрат

Капітальні інвестиції – це кошти, призначені для створення і придбання основних фондів і нематеріальних активів, що підлягають амортизації.

Фіксовані витрати на впровадження системи розраховуватимуться за формулою:

$$К = К_{пр} + К_{навч} + К_{н} + К_{зпз}\text{ , грн.} \quad (3.1)$$

де $К_{пр}$ – вартість впровадження, грн.;

$К_{навч}$ – витрати на навчання технічних фахівців і обслуговуючого персоналу, грн.;

$К_{н}$ – витрати на встановлення та налагодження ПЗ (системи тестування користувачів), грн.;

$К_{зпз}$ – вартість закупівель, грн.

Розрахунок капітальних витрат буде проводитися на прикладі впровадження засобів захисту інформації та додаткового програмного забезпечення: резервне копіювання, контроль стану обладнання, інструктаж з ІБ, Firewall Analyzer Standard Edition, ESET Internet Security тощо.

3.2.1 Розрахунок заробітної плати системного адміністратора

Обліком носіїв інформації, резервним копіюванням, встановленням фаєрволу, антивірусу, налагодженням та встановленням ПЗ та обліком носіїв інформації займається системний адміністратор.

Заробітна плата при простій часовій системі оплати праці визначається за формулою:

$$З = ТС* Ф, \quad (3.2)$$

де ТС – тарифна ставка привласненого робітникові кваліфікаційного розряду в одиницю часу (година, день, місяць), грн.;

Ф – фактично відпрацьований час;

Почасова тарифна ставка системного адміністратора складає ТС = 200 грн/год.

Час на налагодження резервного копіювання займає 10 год.:

$$З = ТС* Ф = 200*10=2000 \text{ грн.}$$

Час на розробку системи тестування користувачів займає 15 год.:

$$З = ТС* Ф = 200*15=3000 \text{ грн.}$$

Час на впровадження системи тестування користувачів займає 5 год.:

$$З = ТС* Ф = 200*5 = 1000 \text{ грн.}$$

Час на встановлення фаєрволу займе 2 год.:

З = ТС* Ф = 200*2 = 400 грн.

Час на встановлення антивірусу займе 2 год.:

З = ТС* Ф = 200*2 = 400 грн.

Час на створення журналу обліку носіїв займе 6 год.:

З = ТС* Ф = 200*6 = 1200 грн.

3.2.2 Розрахунок капітальних витрат

В таблиці 3.1 наведена кількісно-вартісна характеристика заходів, що впроваджується на підприємстві.

Таблиця 3.1 – Кількісно-вартісна характеристика заходів

Міри	Характеристика	Вартість
Резервне копіювання	SSD Samsung T7 2TB Shield Blue (MU-PE2T0R) 2022, up to 1050MB/s, www.rozetka.com.ua	11499
Час на розробку системи тестування користувачів	Займає 15 год.	3000
Час на впровадження системи тестування користувачів	Займає 5 год.	1000
Фаєрвол	Firewall Analyzer Standard Edition, hhttps://www.fortsoft.com.ua/	18486
Антивірус	ESET Internet Security www.rozetka.com.ua	1276
Облік носіїв інформації	Створення журналу (6 год., переоблік раз на тиждень)	1200

Фіксовані витрати на проєктування та впровадження заходів захисту інформації складатимуть:

Резервне копіювання:

К = 2000 + 11499 = 13499 грн.

Розробка системи тестування користувачів:

К = 3000 грн.

Впровадження системи тестування користувачів:

К = 1000 грн.

Фаєрвол:

$$К = 400 + 18486 = 18886 \text{ грн.}$$

Антивірус:

$$К = 400 + 1276 = 1676 \text{ грн.}$$

Облік носіїв інформації:

$$К = 1200 \text{ грн.}$$

Загальні затрати складуть:

$$К = 39261 \text{ грн.}$$

3.3 Розрахунок поточних (експлуатаційних) витрат

Експлуатаційні витрати – це поточні витрати на експлуатацію та обслуговування об'єкта проєктування за визначений період (наприклад, рік), що виражені у грошовій формі.

Під «витратами на керування системою» маються на увазі витрати, пов'язані з керуванням й адмініструванням компонентів системи інформаційної безпеки. До цієї статті витрат можна віднести наступні витрати:

- навчання адміністративного персоналу й кінцевих користувачів;
- амортизаційні відрахування від вартості обладнання та ПЗ;
- заробітна плата персоналу;
- навчальні курси й сертифікація обслуговуючого персоналу;
- технічне й організаційне адміністрування й сервіс.

До експлуатаційних витрат віднесено:

- заробітну плату співробітника системного адміністратора;
- витрати на ліцензію антивірусу;
- витрати на резервне копіювання;
- витрати на розробку системи тестування користувачів;
- витрати на ліцензію фаєрволу;
- витрати на впровадження системи тестування користувачів;
- витрати на облік носіїв інформації;

Річні поточні витрати на функціонування системи заходів протидії загрозам інформації визначаються за формулою (3.3):

$$C = C_1 + C_2 + \ldots + C_n, \text{ грн}, \quad (3.3)$$

де С – вартість підтримки заходу протидії загрозам інформації;

n – порядковий номер заходів протидії загрозам інформації.

Обліком носіїв інформації, резервним копіюванням, підтримкою фаєрволу, антивірусу та обліком носіїв інформації займається системний адміністратор.

Заробітна плата системного адміністратора складає $З_{СА} = 200$ грн/год.

Час на резервне копіювання займе 0,5 год/день.

$$С = ТС* Ф = 200*0,5*250 = 25000 \text{ грн.}$$

Час на підтримку фаєрволу займе 1 год/тиждень, затрати:

$$С = ТС* Ф = 200*1*50 = 10000 \text{ грн.}$$

Час на підтримку антивірусу займе 1 год/тиждень, затрати:

$$С = ТС* Ф = 200*1*50 = 10000 \text{ грн.}$$

Час на створення журналу обліку носіїв займе 1 год/тиждень, затрати:

$$С = ТС* Ф = 200*1*50 = 10000 \text{ грн.}$$

Час на коригування системи тестування користувачів займе 1 год/тиждень, затрати:

$$С = ТС* Ф = 200*1*50 = 10000 \text{ грн.}$$

Затрати на продовження ліцензії антивірусу складають 700 грн.

Затрати на продовження ліцензії фаєрволу складають 7200 грн.

Значення загальних річних поточних витрат складає:

$$С = 25000 + 10000 + 10000 + 10000 + 10000 + 700 + 7200 = 72900 \text{ грн.}$$

3.4 Оцінка можливого збитку від порушення інформаційної безпеки

Загалом можливо виділити такі види збитку, що можуть вплинути на ефективність комп'ютерної системи інформаційної безпеки (КСІБ):

- порушення конфіденційності ресурсів КСІБ (тобто неможливість доступу до них неавторизованих суб'єктів або несанкціонованого використання каналів зв'язку);

- порушення доступності ресурсів КСІБ (тобто можливість доступу до них авторизованих суб'єктів (завжди, коли їм це потрібно);

- порушення цілісності ресурсів КСІБ (тобто їхня неушкодженість);

- порушення автентичності ресурсів КСІБ (тобто їхньої дійсності, непідробленості).

3.5 Визначення збитку від поломок обладнання

Запобігти поломкам обладнання практично неможливо. Природньо, первинна передумова наступна: витрати на ремонт або заміну деяких деталей обладнання не повинні перевищувати вартість самого обладнання.

Вихідні дані для підрахунку збитку:

- час простою внаслідок поломки, $t_п$ (в годинах), $t_п$ = 4 год.;
- час відновлення після поломки, $t_в$ (в годинах), $t_в$ = 2 год.;
- час повторного введення втраченої інформації, $t_{ви}$ (в годинах), $t_{ви}$ = 2 год.;
- заробітна плата обслуговуючого персоналу, Z_0 (грн. в місяць з податками), Z_0 = 32000 грн.;
- заробітна плата співробітників, Z_c (грн. в місяць з податками), Z_c = 20000 грн.;
- кількість обслуговуючого персоналу, N_0, N_0 = 2;
- число співробітників, N_c, N_c = 20;
- прибуток , O (грн. на рік), O = 11500000 грн.;

– вартість заміни обладнання та запасних частин, виправлення помилок в роботі системи, $П_{зч}$ (грн.), $П_{зч}$ = 2000 грн;

– число зламаного обладнання, I, I = 2;

– число поломок на рік, n, n = 6.

Вартість втрат від зниження продуктивності співробітників несправного обладнання розраховується за формулою 3.10:

$$П_п = \frac{\sum Z_c}{\frac{N_c}{160}} \cdot t_п, \text{ грн.}, \quad (3.4)$$

де місячний фонд робочого часу при 40-а годинний робочий тиждень 160 годин.

Підставивши вихідні дані отримаємо:

$$П_п = (20*20000/160)*4 = 10000 \text{ грн.}$$

Вартість відновлення зламаного обладнання розраховується за формулою 3.11:

$$П_в = П_{ви} + П_{пв} + П_{зч}, \text{ грн.} \quad (3.5)$$

де $П_{ви}$ – вартість повторного введення інформації (формула 3.12),

$П_{пв}$ – вартість відновлення обладнання (формула 3.13).

$$П_{ви} = \frac{\sum Z_c}{\frac{N_c}{160}} \cdot t_{ви}, \text{ грн.} \quad (3.6)$$

$$П_{пв} = \frac{\sum Z_o}{\frac{N_o}{160}} \cdot t_в, \text{ грн.} \quad (3.7)$$

Отримаємо:

$$П_{ви} = (20*20000/160)*2 = 5000 \text{ грн.}$$

$$П_{пв} = (2*32000/160)*2 = 800 \text{ грн.}$$

Вартість заміни обладнання та запасних частин, виправлення помилок в системі, $П_{зч}$ (грн.)

$П_{зч}$ = 2000 грн.

Підставивши отримані результати в загальну формулу отримаємо:

$П_в$ = 5000 + 800 + 2000 = 7800 грн.

Втрачена вигода від простою зламаного обладнання становить та розраховується за формулою 3.14 й 3.15 відповідно:

$$U = П_п + П_в + V, \text{ грн.} \quad (3.8)$$

$$V = \frac{O}{Fг} \cdot (t_п + t_в + t_{ви}), \text{ грн,} \quad (3.9)$$

де $F_г$ – річний фонд часу роботи організації (52 робочих тижні, 5-ти денний робочий тиждень, 8-ми годинний робочий день) становить 2080 годин.

V = (11500000/2080)*(4+2+2) = 44230,77 грн.

U = 10000 + 7800 + 44230,77 = 62030,77 грн.

Таким чином, загальний збиток від поломки обладнання, повторного введення інформації в системі, виявлення та усунення помилок в системі складе (формула 3.16):

$$ОУ = \sum_n \sum_I U, \text{ грн.} \quad (3.10)$$

ОУ = 6 * 2 * 62030,77 = 744369,24 грн.

3.6 Загальний ефект від впровадження моделі

Загальний ефект від впровадження моделі підтримки прийняття рішень оцінки загроз інформаційній безпеці для компанії визначається за формулою 3.17 з урахуванням ризиків порушення інформаційної безпеки і становить:

$$E = ОУ \cdot R - C, \text{ грн,} \quad (3.11)$$

де $ОУ$– загальний збиток від атаки на вузол або сегмент корпоративної мережі, грн.;

R – очікувана ймовірність атаки на вузол або сегмент корпоративної мережі, частки одиниці;

C – щорічні витрати на експлуатацію моделі підтримки прийняття рішень оцінки загроз інформаційній безпеці для компанії, грн.

Таким чином, загальний ефект від впровадження становить:

$$E = 744369{,}24 * 0{,}4 - 72900 = 224847{,}7 \text{ грн.}$$

3.7 Визначення та аналіз показників економічної ефективності моделі

Оцінка економічної ефективності моделі, розглянутої у спеціальній частині кваліфікаційної роботи, здійснюється на основі визначення та аналізу коефіцієнта повернення інвестицій *ROSI* (Return on Investment for Security) за формулою 3.18 та терміну окупності капітальних інвестицій T_o за формулою 3.19.

$$ROSI = \frac{E}{K}, \text{ частки одиниці,} \qquad (3.12)$$

де E – загальний ефект від впровадження системи захисту, грн.;

K – капітальні інвестиції за варіантами, що забезпечили цей ефект, грн.

Підставивши відповідні значення, маємо:

$$\text{ROSI} = 224847{,}7 / 39261 = 5{,}73$$

Організація здійснює фінансування капітальних інвестицій за рахунок реінвестування власних коштів (частини прибутку та амортизаційних відрахувань).

Проєкт визнається економічно доцільним, якщо розрахунковий коефіцієнт ефективності перевищує річний рівень прибутковості.

Отже, проєкт є економічно доцільним.

Термін окупності капітальних інвестицій T_o показує, за скільки років капітальні інвестиції окупляться за рахунок загального ефекту від впровадження системи безпеки:

$$T_O = \frac{K}{E} = \frac{1}{ROSI}, \text{ років.} \quad (3.14)$$

Підставимо значення:

$T_o = 1 / 5{,}73 = 0{,}17$ року (приблизно 2 місяці).

3.8 Висновок

Розрахувавши збитки від реалізації можливих несправностей, які склали 744369,24 грн., і порівнявши їх з витратами на забезпечення підтримки працездатності системи 72900 грн., та витратами на розробку моделі 39261 грн., можна зробити висновок, що витрати на забезпечення інформаційної безпеки є не значними у співвідношенні до збитків, впровадження системи є економічно доцільним заходом, термін окупності системи безпеки становить 2 місяці. Для подальшого розвитку діяльності підприємства впровадження даних заходів є необхідною умовою для виконання.

ВИСНОВКИ

У даній кваліфікаційній роботі була розглянута класифікація атаки на WEB-сервери. Проведено аналіз загроз WEB-серверів та обрано профіль захищеності відповідно до НД ТЗІ 2.5-010-03. Досліджено існуючі засоби протидії автоматизованому виконанню зловмисних дій на WEB-серверах. Поглиблено проаналізований один з видів засобу протидії, оснований на зворотному тесті Тюрінга, який вимагає від користувача виконання дії щодо проходження тестового завдання.

В результаті дослідження було виявлено, що більшість сучасних видів зворотного тесту Тюрінга мають вразливість основану на використанні сторонніх сервісів, або використовують бази тестових завдань, які мають обмежену кількість. На основі результатів дослідження була розроблена система тестування, яка ґрунтується на зворотному тесті Тюрінга та не має вразливостей, притаманних більшості проаналізованим системам тестування. Розроблена система тестування була вдало випробувана в різних Інтернет-браузерах і відповідає сучасним нормам WEB-програмування.

Було проведено розрахунок витрат на розробку системи тестування, впровадження системи інформаційної безпеки та розрахунок передбачених збитків від атак на WEB-сервери. Ефективність і доцільність запропонованих заходів доведена.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 НД ТЗІ 2.5-010-03. Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу.

2 НД ТЗІ 2.5-005-99. Класифікація автоматизованих систем і стандартні функціональні профілі захищеності оброблюваної інформації від несанкціонованого доступу.

3 НД ТЗІ 2.5-004-99. Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу.

4 Способи захисту сайту від автоматичного заповнення Web-форм / Спосіб доступу: URL: http://mycodes.in.ua/archives/398. – Назва з екрана.

5 Міжнародний стандарт ISO/IEC 27005:2011 «Інформаційна технологія. Методи і засоби забезпечення безпеки. Менеджмент ризику інформаційної безпеки» [Електронний ресурс] - Режим доступу: www/ URL: https://exebit.files.wordpress.com/2013/11/iso-27005-2011-ru-v1.pdf.

6 Закон України «Про інформацію» [Електронний ресурс] / Київ, Верховна Рада України - Режим доступу : www/ URL: http://zakon5.rada.gov.ua/laws/show/2657-12 - 21.05.2015 г. – Назва з екрана.

7 Методи та засоби оцінювання ризиків безпеки інформації в системах електронної комерції [Електронний ресурс] - Режим доступу: www/ URL: http://www.nbuv.gov.ua/old_jrn/natural/Vnulp/ISM/2008_610/03.pdf.

8 Risk Identification and Analysis – GIAC [Електронний ресурс] - Режим доступу: www/ URL: http://www.nap.edu/read/11183/chapter/6.

9 Методика визначення ефективності капітальних вкладень [Електронний ресурс]. – Режим доступу : www/ URL: http://pidruchniki.com/1541010436255/ekonomika/metodika_viznachennya_efektivnosti_kapitalnih_vkladen.

10 Розрахунок річних експлуатаційних витрат [Електронний ресурс]. – Режим доступу : www/ URL: http://studopedia.org/6-128587.html.

11 Зоріна Т.І. Системи виявлення і запобігання атак в комп'ютерних мережах / Т.І. Зоріна // Вісник східноукраїнського національного університету ім. В. Даля. – № 15 (204), ч.1. – 2013.

12 Домарев В.В. Організація захисту електронних документів / Спосіб доступу: URL: http://www.security.ukrnet.net/modules/sections/index.php?op=viewarticle&artid=568 – Назва з екрана.

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

№	Формат	Найменування	Кількість листів	Примітки
Документація				
1	А4	Реферат	2	
2	А4	Список умовних скорочень	2	
3	А4	Зміст	2	
4	А4	Вступ	2	
5	А4	Розділ 1	42	
6	А4	Розділ 2	29	
7	А4	Розділ 3	10	
8	А4	Висновки	1	
9	А4	Перелік посилань	2	
10	А4	Додаток А	1	
11	А4	Додаток Б	10	
12	А4	Додаток В	4	
13	А4	Додаток Г	1	
14	А4	Додаток Ґ	1	
15	А4	Додаток Д	2	

ДОДАТОК Б. Початковий код файлів системи тестування

Лістинг Б.1 – drug_n_drop.js

```
function fixEvent(e) {
	e = e || window.event

	if ( e.pageX == null && e.clientX != null ) {
		var html = document.documentElement
		var body = document.body
		e.pageX = e.clientX + (html && html.scrollLeft || body && body.scrollLeft || 0) -
(html.clientLeft || 0)
		e.pageY = e.clientY + (html && html.scrollTop || body && body.scrollTop || 0) -
(html.clientTop || 0)
	}

	if (!e.which && e.button) {
		e.which = e.button & 1 ? 1 : ( e.button & 2 ? 3 : ( e.button & 4 ? 2 : 0 ) )
	}

	return e
}

function getOffset(elem) {
   if (elem.getBoundingClientRect) {
      return getOffsetRect(elem)
   } else {
      return getOffsetSum(elem)
   }
}

function getOffsetRect(elem) {
   var box = elem.getBoundingClientRect()

   var body = document.body
   var docElem = document.documentElement

   var scrollTop = window.pageYOffset || docElem.scrollTop || body.scrollTop
   var scrollLeft = window.pageXOffset || docElem.scrollLeft || body.scrollLeft
   var clientTop = docElem.clientTop || body.clientTop || 0
   var clientLeft = docElem.clientLeft || body.clientLeft || 0
   var top  = box.top +  scrollTop - clientTop
   var left = box.left + scrollLeft - clientLeft

   return { top: Math.round(top), left: Math.round(left) }
}
```

Продовження лістингу Б.1

```
function getOffsetSum(elem) {
    var top=0, left=0
    while(elem) {
        top = top + parseInt(elem.offsetTop)
        left = left + parseInt(elem.offsetLeft)
        elem = elem.offsetParent
    }

    return {top: top, left: left}
}

function DragObject(element) {
    element.dragObject = this

    dragMaster.makeDraggable(element)
    var rememberPosition
    var mouseOffset

    this.onDragStart = function(offset) {
        var s = element.style
        rememberPosition = {top: s.top, left: s.left, position: s.position}
        s.position = 'absolute'

        mouseOffset = offset
    }

    this.hide = function() {
        element.style.display = 'none'
    }

    this.show = function() {
        element.style.display = ''
    }

    this.onDragMove = function(x, y) {
        element.style.top =  y - mouseOffset.y +'px'
        element.style.left = x - mouseOffset.x +'px'
    }

    this.onDragSuccess = function(dropTarget) { }
    this.onDragFail = function() {
        var s = element.style
        s.top = rememberPosition.top
        s.left = rememberPosition.left
        s.position = rememberPosition.position
    }

    this.toString = function() {
        return element.id
    }
}
```

Продовження лістингу Б.1

```
function DropTarget(element) {

   element.dropTarget = this

   this.canAccept = function(dragObject) {
     return true
   }

   this.accept = function(dragObject) {
     document.getElementById('coord'+dragObject.toString().substr(dragObject.toString().length-
1)).value=ObjectPosition(document.getElementById(dragObject))[0];
   }

   this.onLeave = function(dragObj) {
   }

   this.onEnter = function() {
   }

   this.toString = function() {
      return element.id
   }
}

var dragMaster = (function() {

   var dragObject
   var mouseDownAt

   var currentDropTarget

   function mouseDown(e) {
      e = fixEvent(e)
      if (e.which!=1) return

       mouseDownAt = { x: e.pageX, y: e.pageY, element: this }

      addDocumentEventHandlers()

      return false
   }

function mouseMove(e){
      e = fixEvent(e)

      if (mouseDownAt) {
         if (Math.abs(mouseDownAt.x-e.pageX)<5 && Math.abs(mouseDownAt.y-e.pageY)<5) {
            return false
         }
         var elem  = mouseDownAt.element
         dragObject = elem.dragObject
```

Продовження лістингу Б.1

```
            var mouseOffset = getMouseOffset(elem, mouseDownAt.x, mouseDownAt.y)
            mouseDownAt = null

            dragObject.onDragStart(mouseOffset)

        }

        dragObject.onDragMove(e.pageX, e.pageY)

        var newTarget = getCurrentTarget(e)

        if (currentDropTarget != newTarget) {
            if (currentDropTarget) {
                currentDropTarget.onLeave(dragObject)
            }
            if (newTarget) {
                newTarget.onEnter()
            }
            currentDropTarget = newTarget

        }

        return false
    }

function mouseUp(){
        if (!dragObject) {
            mouseDownAt = null
        } else {
            if (currentDropTarget) {
                currentDropTarget.accept(dragObject)
                dragObject.onDragSuccess(currentDropTarget)
            } else {
                dragObject.onDragFail()
            }

            dragObject = null
        }
        removeDocumentEventHandlers()
    }

    function getMouseOffset(target, x, y) {
        var docPos    = getOffset(target)
        return {x:x - docPos.left, y:y - docPos.top}
    }

    function getCurrentTarget(e) {

        if (navigator.userAgent.match('MSIE') || navigator.userAgent.match('Gecko')) {
            var x=e.clientX, y=e.clientY
```

Продовження лістингу Б.1

```
        } else {
            var x=e.pageX, y=e.pageY
        }
        dragObject.hide()
        var elem = document.elementFromPoint(x,y)
        dragObject.show()

        while (elem) {
            if (elem.dropTarget && elem.dropTarget.canAccept(dragObject)) {
                return elem.dropTarget
            }
            elem = elem.parentNode
        }

        return null
    }

    function addDocumentEventHandlers() {
        document.onmousemove = mouseMove
        document.onmouseup = mouseUp
        document.ondragstart = document.body.onselectstart = function() {return false}
    }
    function removeDocumentEventHandlers() {
        document.onmousemove = document.onmouseup = document.ondragstart =
document.body.onselectstart = null
    }

    return {

        makeDraggable: function(element){
            element.onmousedown = mouseDown
        }
    }
}())

function ObjectPosition(obj) {
    var curleft = 0;
    var curtop = 0;

    if (obj.offsetParent) {
        do {
            curleft += obj.offsetLeft;
            curtop += obj.offsetTop;
        } while (obj = obj.offsetParent);
    }

    return [curleft,curtop];
}
```

Лістинг Б.2 – Файл style.css

```
*{
    margin: 0;
    padding: 0;
}

#clear{
    clear: both;
    overflow: hidden;
    height: 0;
}

#form{
    border: 1px solid black;
    width: 415px;
    margin: 150px 0 0 250px;
    padding: 10px 10px;
    background: #FFFAF0;
    }
    #form h1{
        font-size: 16px;
        float: left;
        font-weight: bold;
        width: 140px;
    }
    #inp{
        width: 145px;
        height: 20px;
        float: left;
    }
    #form h2{
        font-size: 16px;
        font-weight: bold;
        margin: 10px 0 0 0;
        color: #d00c0c;
    }
    #form #captcha_field{
        width: 145px;
        height: 50px;
        border: 1px solid black;
        margin: 10px 0 0 140px;
        background: #FFFAFA;
    }
    #form #button{
        width: 121px;
        height: 31px;
        font-weight: bold;
        margin: 15px 0 0 153px;
    }
```

Лістинг Б.3 – Файл class.php

```
<?php
if (!session_id()) {
    session_save_path('../tmp_sessions');
    @session_start();
}

class captcha
{
    function create_captcha()
    {
        $nums = array('0', '1', '2', '3', '4', '5', '6', '7', '8', '9');
        shuffle($nums);

        $nums2 = array($nums[0], $nums[1], $nums[2], $nums[3]);

                    asort($nums2);

        $nums2 = array_keys($nums2);

        $_SESSION['captcha_number'] = array($nums2[0], $nums2[1], $nums2[2], $nums2[3]);

        echo "
            <div id='captcha_field'>
                <img src='img/{$nums[0]}.gif' id='c_n1' style='position:absolute;' alt=''/>
                <img src='img/{$nums[1]}.gif' id='c_n2' style='position:absolute;' alt=''/>
                <img src='img/{$nums[2]}.gif' id='c_n3' style='position:absolute;' alt=''/>
                <img src='img/{$nums[3]}.gif' id='c_n4' style='position:absolute;' alt=''/>
            </div>

            <input type='hidden' name='coords[]' id='coord1'/>
            <input type='hidden' name='coords[]' id='coord2'/>
            <input type='hidden' name='coords[]' id='coord3'/>
            <input type='hidden' name='coords[]' id='coord4'/>
        ";
    }

    function check_captcha($coords)
    {
        if (!isset($_SESSION['captcha_number'])) {
            return false;
        }

        asort($coords);
        $coords = array_keys($coords);

        for ($i = 0; $i < 4; $i++) {
            if (!isset($coords[$i]) || !isset($_SESSION['captcha_number'][$i])) {
                return false;
            }
            if ($coords[$i] !== $_SESSION['captcha_number'][$i]) {
```

Продовження лістингу Б.3

```
            return false;
        }

        if ($coords[$i] !== $_SESSION['captcha_number'][$i]) {
            return false;
        }
    }
    return true;
  }

  function unset_captcha()
  {
    $tmp = session_name();
    $_SESSION = array();
    unset($_COOKIE[$tmp]);
    session_destroy();
    return true;
  }
}

$obj = new captcha();

if (isset($_REQUEST['verification']) && $_POST['name'] == '')
{
  $obj->unset_captcha();
  header ('Location: ../index.php');
  exit(0);
}
elseif (isset($_POST['coords']) && is_array($_POST['coords']) && ($_POST['name'] != '') ?
$_POST['coords'] : array()) {

  echo $obj->check_captcha($coords) ? "<font color='green'><b>Ви, $_POST[name], успішно
пройшли перевірку!</b></font> <a style='text-decoration: none; font-weight: bold;'
href='../index.php'>Повернутися</a>" : "<font color='red'><b>Невірний код
підтвердження!</b></font> <a style='text-decoration: none; font-weight: bold;'
href='../index.php'>Повернутися</a>";

  $obj->unset_captcha();}
?>
```

Лістинг Б.4 – index.php

```
<?php
  session_save_path('tmp_sessions');

  @session_start();

  require('src/class.php');
?>
```

Продовження лістингу Б.4 – index.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=windows-1251" />
   <title>Система тестування користувачів WEB-серверу</title>

   <link rel="stylesheet" href="css/style.css" type="text/css" />

   <script type="text/javascript" src="src/drag_n_drop.js"></script>

   <script type="text/javascript">
   <!--
      function preloadCaptcha()
      {
         new DragObject(document.getElementById('c_n1'));
         new DragObject(document.getElementById('c_n2'));
         new DragObject(document.getElementById('c_n3'));
         new DragObject(document.getElementById('c_n4'));
         new DropTarget(document.getElementById('captcha_field'));
         var temp;

         for(var i=1;i<=4;i++)
         {
            temp=(ObjectPosition(document.getElementById('captcha_field'))[0]+(i-1)*35+5);
            document.getElementById('c_n'+i).style.left=temp+'px';

document.getElementById('c_n'+i).style.top=(ObjectPosition(document.getElementById('captcha_fi
eld'))[1]+10)+'px';
            document.getElementById('coord'+i).value=temp;
         }
      }
   -->
   </script>
</head>

<body onload="preloadCaptcha();">

      <div class="form">
         <form action="src/class.php" method="post">
         <div>
            <h1>Ім'я користувача:</h1> <input class="inp" type="text" name="name"/>
            <div class="clear"></div>

             <h2>Розташуйте цифри в порядку зростання, перетягуючи їх!</h2>
            <div class="clear"></div>

            <?php
               $obj->create_captcha();
            ?>
```

Продовження лістингу Б.4

```
            <div class="clear"></div>

            <input class="button" type="submit" name="verification" value="Перевірити"/>
          </div>
          </form>
        </div>

    </body>

</html>
```

ДОДАТОК В. Використання системи тестування в різних Інтернет-браузерах

Internet Explorer 6.0.2900.5512

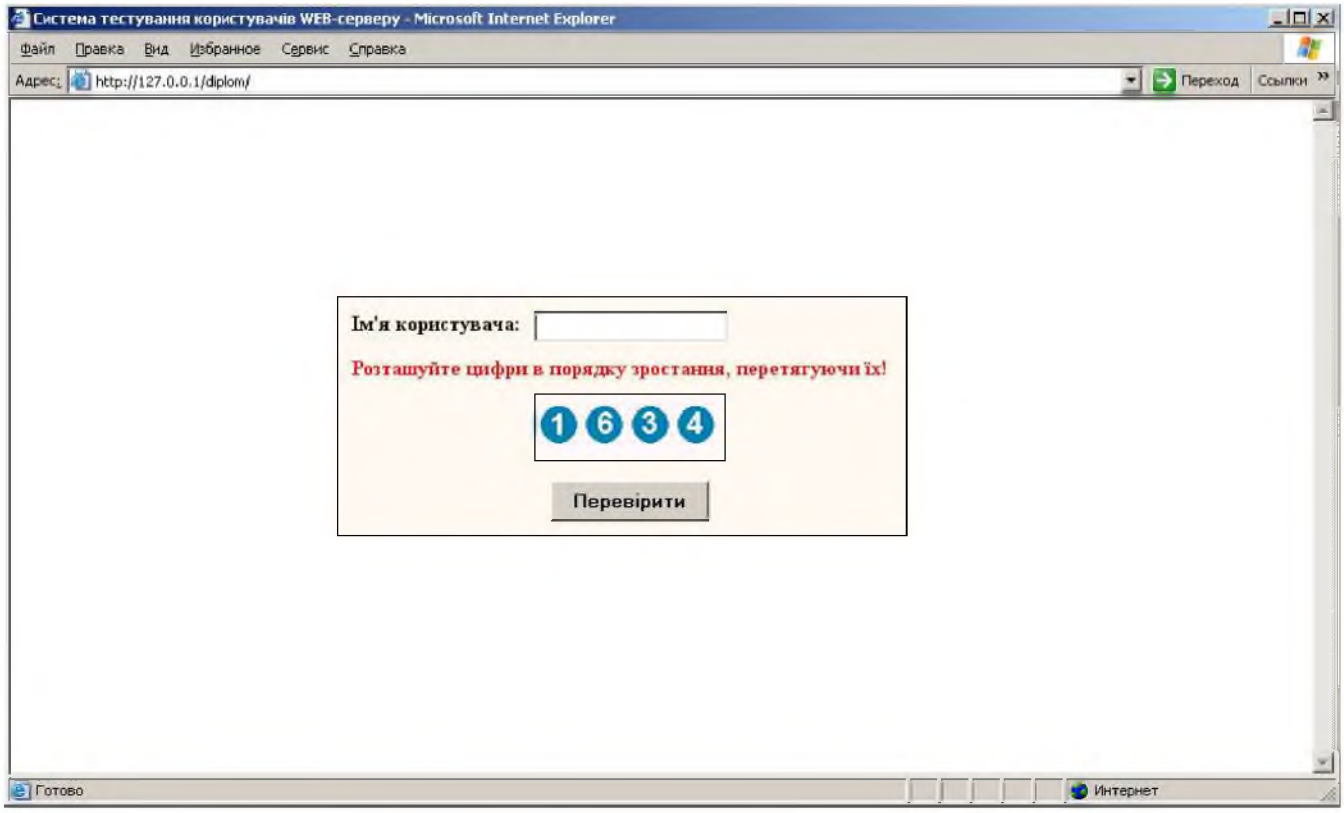

Рисунок В.1 – Форма тестового завдання

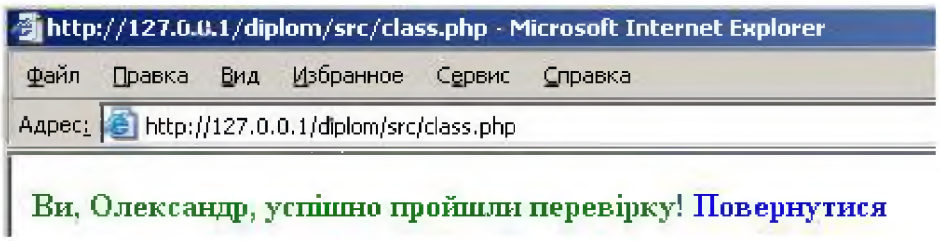

Рисунок В.2 – Успішне проходження тестового завдання

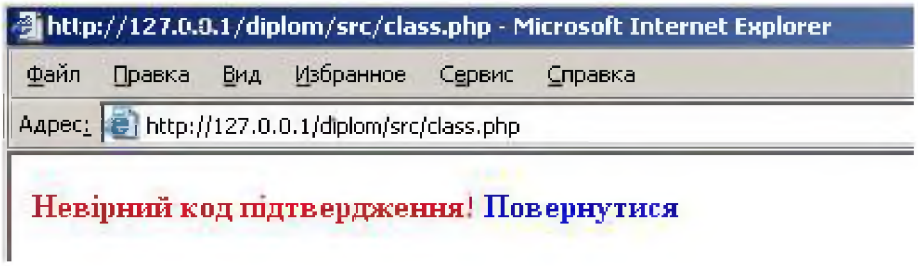

Рисунок В.3 – Невдале проходження тестового завдання

MozillaFirefox 3.6.15

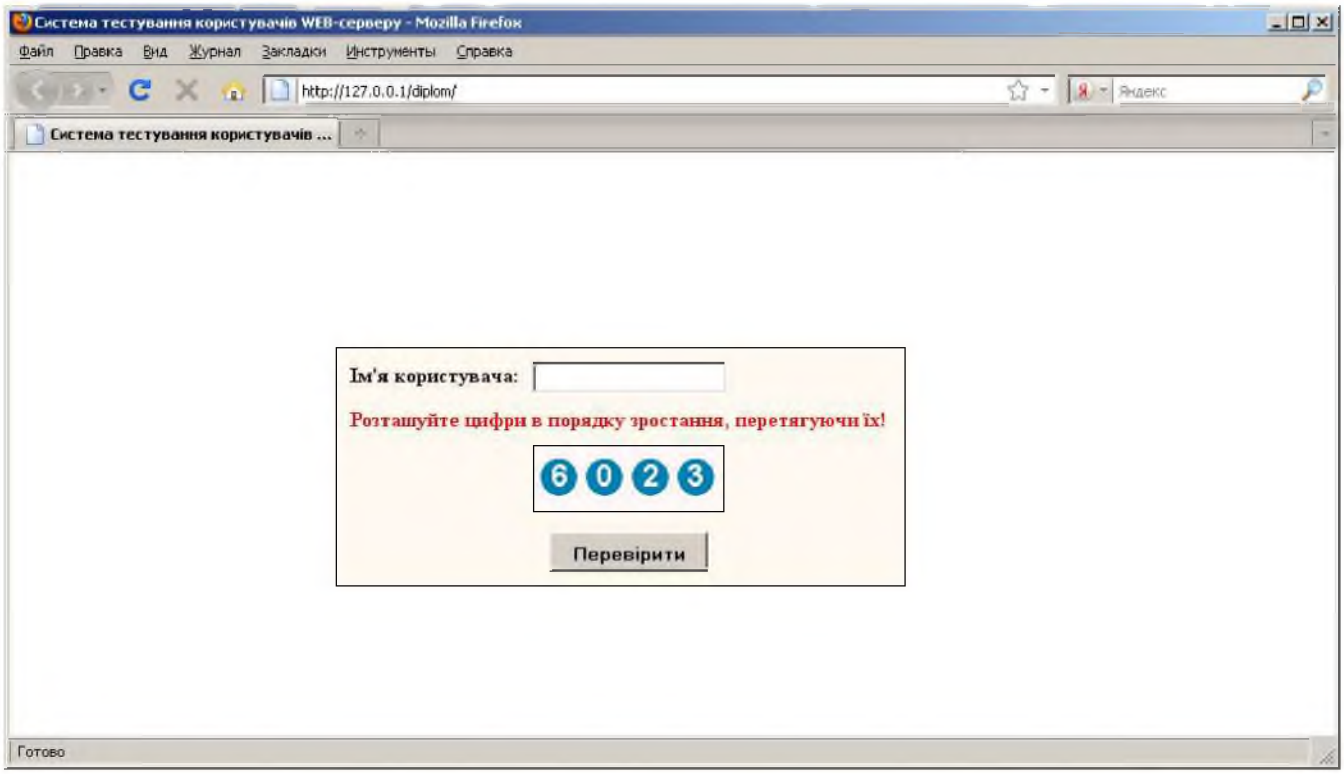

Рисунок В.4 – Форма тестового завдання

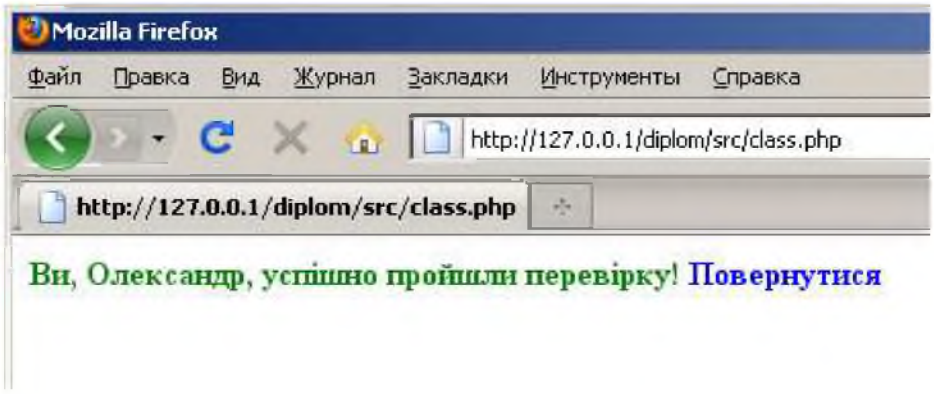

Рисунок В.5 – Успішне проходження тестового завдання

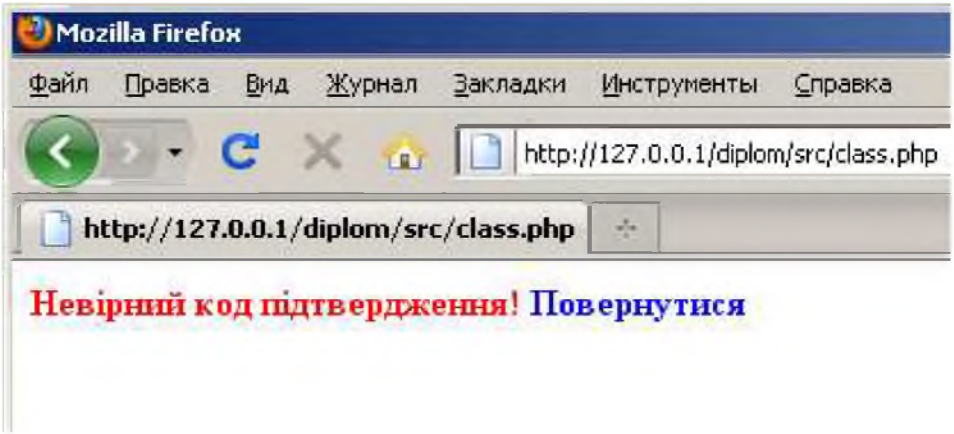

Рисунок В.6 – Невдале проходження тестового завдання

Google Chrome 10.0.648.205

Рисунок В.7 – Форма тестового завдання

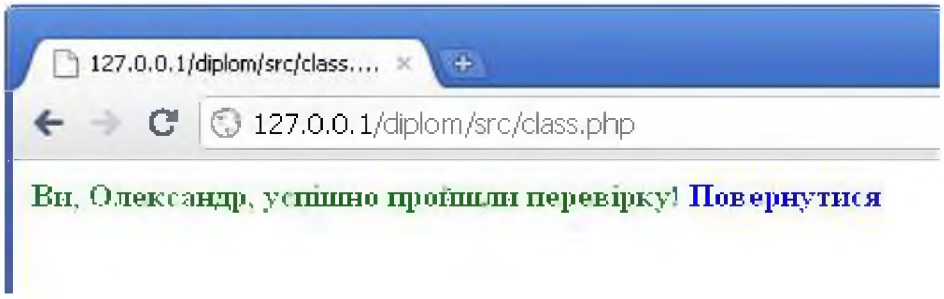

Рисунок В.8 – Успішне проходження тестового завдання

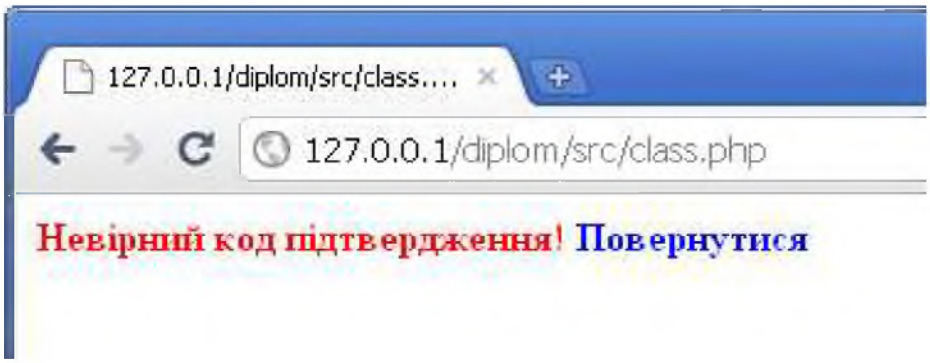

Рисунок В.9 – Невдале проходження тестового завдання

ДОДАТОК Г. Перелік документів на оптичному носії

1 Презентація_Абрамов.ppt

2 Кваліфікаційна робота_Абрамов.doc

ДОДАТОК Ґ. Відгук керівника економічного розділу

Керівник розділу _____ _____

(підпис) (прізвище, ініціали)

ДОДАТОК Д. Відгук керівника кваліфікаційної роботи

В І Д Г У К

на кваліфікаційну роботу студента групи 125м-22-2 Абрамова І.С. на тему: «Розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга»

Кваліфікаційна робота представлена пояснювальною запискою на 113 с., містить 32 рис., 19 табл., 6 додатків, 12 джерел.

Метою даної кваліфікаційної роботи розробка системи тестування користувачів WEB-серверів із використанням зворотного тесту Тюрінга для підвищення захисту від автоматизованого виконання зловмисних дій WEB-роботами, що можуть загрожувати конфіденційності, доступності чи цілісності інформації.

У ході виконання кваліфікаційної роботи були вирішені наступні питання: проаналізовано загрози WEB-серверів, проаналізовано засоби захисту від WEB-роботів, проведено огляд способів обходу зворотного тесту Тюрінга, обрано профіль захищеності відповідно до НД ТЗІ, розроблено систему тестування користувачів WEB-серверів.

В економічному розділі проведено розрахунок витрат на розробку ПЗ та розрахунок передбачених збитків від атак.

Запропонована в роботі система тестування користувачів може бути використана для забезпечення захисту від автоматизованого виконання зловмисних дій на будь-яких WEB-серверах.

До недоліків проєкту слід віднести нечіткість окремих висновків і визначень, окремі невідповідності вимогам при оформленні.

Рівень запозичень у кваліфікаційній роботі відповідає вимогам «Положення про систему виявлення та запобігання плагіату».

В цілому робота задовольняє усім вимогам, а її автор Абрамов І.С. заслуговує на оцінку « » та присвоєння кваліфікації «Магістр з кібербезпеки» за спеціальністю 125 Кібербезпека.

Керівник роботи,
д.т.н., проф. **Корнієнко В.І.**