МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»

Освітня програма: «Безпека комп'ютерних систем і мереж»

Група:4КБ-01

Дипломний проект

здобувача освіти денної форми навчання
КБ.01.18.000.ДП

ШМИГОЛЯ
ОЛЕКСАНДРА ІВАНОВИЧА

м. Одеса
2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітньо-професійна програма: ***«Безпека комп'ютерних систем і мереж»***

Група: **4КБ-01**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проєкту на тему:

Проектування системи керування доступом до розподілених веб-ресурсів

Проектний матеріал складається з пояснювальної записки на 63 сторінках та графічного (перзентаційного) матеріалу на 12 аркушах (слайдах).

Дипломник _____ (Шмиголь О.І.)

Керівник _____ (Кірєєв І.А.)

Консультанти:

з економічного розділу _____ (Іванченков В. С.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н. І.)

з нормоконтролю _____ (Петрашова В. І.)

старший консультант _____ (Кривченко Ю. В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю. В.)

Завідувач відділення _____ (Скорнякова О. В.)

Захист «21» 06 2024 р. Протокол ЕК № 5

Оцінка ЕК 4 (добре) / 75 б.

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *ком'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 – «Комп'ютерна інженерія»*

Освітня програма *«Безпека комп'ютерних мереж»*

ЗАТВЕРДЖУЮ

Заст. дир. з НВР *Беркань І. В.*

« 15 » 01 *2024* року

ЗАВДАННЯ

на дипломний проєкт (роботу)

Шмиголю Олександру Івановичу

1. Тема проєкту (роботи) *Проектування системи керування доступом до розподілених веб-ресурсів*

Затверджена наказом по коледжу від « 02 » 11 *2024 р.*, наказ № 244-АД-ОО

2 Термін здачі закінченого проєкту (роботи) 10,06.24

3. Вихідні дані до проєкту (роботи)

вимоги до системи керування доступом до розподілених веб-сервісів; вимоги до системи керування доступом до розподілених веб-сервісів; аутентифікацію; використовувати наступні методи: евристичний аналіз, сигнатурний аналіз, системний аналіз, методи програмної інженерії, клієнт-серверна архітектура, UML- проектування

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Вступ. Основний розділ: аналіз технологій та процесів розподілених веб-ресурсів; механізми керування доступом до розподілених веб-ресурсів, методи та засоби керування доступом до розподілених веб-ресурсів; розробка системи керування доступом . Економічний розділ Розділ охорони праці та техніки безпеки. 5. Перелік використаних інформаційних джерел.

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Презентація Power Point – 10 слайдів

Схема учасників обміну веб-ресурсами; Схема передачі інформації в інформаційній истемі; Переповнення буферу оперативної пам'яті веб-серверу; Процес шифрування даних публічним ключом; приклади програмних кодів..

6. Консультанти по проєкту, із зазначенням розділів проєкту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Кірєєв І.А.*		
Економічний розділ	*Іванченков В. С.*		
Розділ охорони праці	*Чорновол Н. І.*		
Нормоконтроль	*Петрашова В. І.*		
Старший консультант	*Кривченко Ю. В.*		

7. Дата видачі завдання 15.01.24

Керівник *Кірєєв І.А.* (підпис)

Завдання прийняв до виконання *Шмиголь О.І.* (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проєкту (роботи)	Термін виконання етапів дипломного проєкту (роботи)	Відмітка про виконання
1	*Вступ. Постановка мети та задач проєктування*	*20.05.2024 р.*	виконано
2	*Провести аналіз технологій та процесів розподілених веб-ресурсів*	*23.05.2024 р.*	виконано
3	*Провести дослідження механізмів керування*	*25.05.2024 р.*	виконано
	розподіленими веб-ресурсами	*28.05.2024 р.*	виконано
4	*Провести дослідження інструментальних засобів*	*01.06.2024 р.*	виконано
	керування доступом до розподілених веб-ресурсів		
5	*Розробити систему керування доступом до*	*03.06.2024 р.*	виконано
	розподілених веб-ресурсів		
6	*Економічний розділ*	*07.06.2024 р.*	виконано
8	*Розділ охорони праці та техніки безпеки*	*09.06.2024 р.*	виконано
9	*Висновки. Перелік використаних інформаційних джерел*	*10.06.2024 р.*	виконано
10	*Оформлення пояснювальної записки*	*11.06.2024 р.*	виконано
11	*Оформлення графічної (презентаційної) частини*	*12.06.2024 р.*	виконано
12	*Підготовка доповіді для захисту*	*13.06.2024 р.*	виконано
13	*Малий захист дипломного проєкту*	*17.06.2024 р.*	виконано

Дипломник (підпис)

Керівник (підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КБ 01. 18 000. 00 ДП	Дипломний проєкт		
А4			КБ 01. 18 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КБ 07. 18 000. 00 ДП ПЗ		
Розробив		Кірсєєв І.В.		10.06	Проектування системи керування доступом до розподілених веб-ресурсів	Літ.	Аркуш 4 / Аркушів 75
Перевірив		Шувалова І.В.		10.06		НДП	
Н. Контр.		Петрашова В.І.		10.06		ВСП «ОТФК ОНТУ» гр.4КБ-07	
Затверд.		Кривченко Ю.В.		10.06			

ЗМІСТ

ВСТУП

Актуальність теми дипломного проектування полягає у тому, що проблеми пошуку та застосування засобів для керування доступом до інтернет-ресурсів

Зі зростанням інтенсивності використання Інтернету, перенесенням інформаційних просторів у онлайн та розширенням каналів зв'язку, проблеми пошуку та впровадження засобів керування доступом до інтернет-ресурсів стають одними з ключових. Кількість користувачів Інтернету продовжує зростати, що стимулює створення нових методів і технологій доступу. Збільшений попит на інформаційні ресурси породжує відповідну пропозицію, що призводить до зростання кількості доступних веб-ресурсів, нових веб-сайтів та сервісів.

Цей стрімкий розвиток викликає потребу у створенні нових інженерних інструментів для полегшення розгортання веб-серверів, оптимізації потоків даних, їх маршрутизації та керування доступом до веб-ресурсів. Збільшення навантаження та зростання кількості шкідливого програмного забезпечення (ПЗ) у мережі є очевидними наслідками цих процесів.

Для вирішення цих проблем багато світових компаній, таких як AVZ, KasperskyLabs, ESET, створюють та підтримують антивірусне ПЗ, яке задовольняє потреби користувачів у захисті від інтернет-загроз. Переважна більшість їхніх механізмів спрямована на виявлення загроз після потрапляння на кінцеві машини користувачів, використовуючи апаратні ресурси користувачів для аналізу. Інші підходи передбачають аналіз інтернет-загроз на етапі до їх потрапляння на комп'ютер користувача.

Отже, обрана тема дипломного проекту є актуальною з точки зору сучасних тенденцій, оскільки вимагає дослідження у сфері керування та аналізу веб-трафіку, що потребує значних професійних навичок у мережевому та системному програмуванні.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Поняття веб-ресурсу

Веб-ресурси є інформацією, доступною для завантаження з Інтернету за допомогою обладнання та програмного забезпечення, і мають унікальний локатор ресурсу (URL). Ці ресурси можуть бути текстовими, мультимедійними, графічними, веб-сторінками або документами. Передача веб-ресурсів відбувається через мережу за правилами протоколу HTTP, таких як HTTP/1.0, HTTP/1.1, HTTP/2.0, які описані у документах RFC1945, RFC7231, RFC7540.

Зазвичай веб-ресурси створюються під час розробки веб-додатків або веб-сайтів відповідно до їх функціональних вимог. Кожен веб-ресурс має унікальний ідентифікатор ресурсу (URI) в межах домену, URL для доступу до нього та назву та тип.

Робота з веб-ресурсами в програмній системі включає такі учасники, як канал зв’язку з Інтернетом, клієнтська програма, протоколи обміну даними (HTTP), веб-сервер, DNS-сервер для перекладу домену у IP-адресу, а також проксі-сервери, мережеві фільтри та екрани для управління доступом, шифрування даних, стиснення трафіку та аналізу запитів (за потреби).

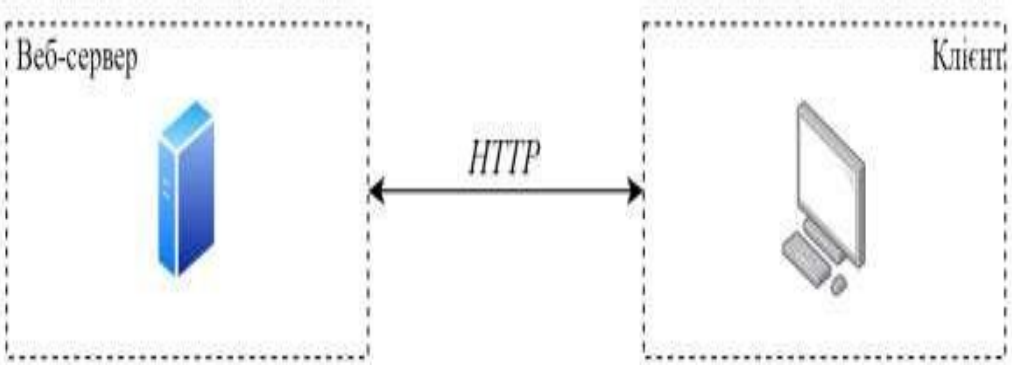


Рисунок.1.1 Схема обміну веб-ресурсами

За рахунок широкого використання вебу та стандартизації протоколу HTTP, операційні системи та мови програмування автоматизують більшість складних процесів роботи з веб-ресурсами, перетворюючи їх на прості високорівневі операції. Це дозволяє приховати деталі, такі як встановлення TCP-з'єднань з веб-серверами, перетворення доменних імен за допомогою DNS-серверів, створення обробників HTTP-запитів та інші низькорівневі процеси, від користувача.

Проблематика та напрямки роботи з веб-трафіком:

1. Протоколи HTTP/HTTPS.
2. Шифрування веб-трафіку.
3. Стиснення даних.
4. Захист від інтернет-загроз.
5. Захист від DDoS-атак.

Основні аспекти дипломного проектування:

1. Розробка системи керування доступом до розподілених веб-ресурсів.
2. Дослідження можливих керівних впливів на цей процес.
3. Аналіз технологій доступу до веб-ресурсів.
4. Дослідження форматів пересилання даних.
5. Розуміння роботи протоколів HTTP/HTTPS та TCP.
6. Аналіз веб-трафіку на наявність інтернет-загроз.
7. Оптимізація швидкодії створюваної системи.
8. Навички проектування та програмування серверних додатків.

1.2 Механізми взаємодії з веб-ресурсами

Запит веб-ресурсу - це спеціальне повідомлення, яке сформоване відповідно до правил протоколу HTTP і призначене для отримання даних конкретного веб-ресурсу. Кожний запит включає наступні складові:

1. Метод запиту, URI ресурсу та версію HTTP протоколу, які використовуються.
2. HTTP-заголовки, які містять додаткову інформацію.
3. Тіло запиту, якщо воно є необхідним.

Основні методи запиту включають GET, POST, PUT, HEAD, CONNECT, OPTIONS та інші. Метод визначає спосіб взаємодії з ресурсом, а не його тип. Коли клієнт формує HTTP-запит, він вказує цільовий URI в одній із таких форм:

1. origin-form
2. absolute-form
3. authority-form
4. asterisk-form

Origin-form використовується для прямих запитів до цільового серверу, за винятком CONNECT та OPTIONS. Клієнт повинен вказати лише абсолютний шлях та, за потреби, додаткові параметри запиту. Також він повинен надіслати заголовок Host.

Absolute-form використовується для запитів до проксі. Клієнт вказує цільовий URI в абсолютній формі.

Authority-form використовується тільки для CONNECT запитів, де клієнт надсилає тільки компонент цільового URI, який відповідає авторизації.

Asterisk-form використовується тільки для OPTIONS запитів, коли клієнт хоче отримати інформацію про весь сервер, а не лише конкретний ресурс.

Проксі-сервер може використовуватися для обробки запиту клієнта, надавши відповідь з кеш-пам'яті або передавши запит на інший сервер.

Ці різноманітні форми дозволяють клієнту ефективно взаємодіяти з веб-ресурсами відповідно до їх специфікацій.

HTTP-заголовки відіграють критичну роль у взаємодії між клієнтом і сервером під час передачі даних через протокол HTTP. Їх використання дозволяє передавати різноманітну метаінформацію про запит, відповідь або самі дані.

Однак, як ви зазначили, необмеженість кількості заголовків може створювати потенційні проблеми з безпекою, такі як атаки на переповнення буфера, коли зловмисники намагаються надіслати велику кількість HTTP-заголовків, які перевищують можливості обробки сервера.

Для запобігання таким атакам важливо використовувати належні заходи безпеки, такі як обмеження на кількість допустимих заголовків, перевірка на коректність та обмеження на розмір кожного заголовка.

Додатково, регулярні оновлення серверного програмного забезпечення для виправлення виявлених уразливостей також є важливою практикою з точки зору кібербезпеки (рисунок 1.2).

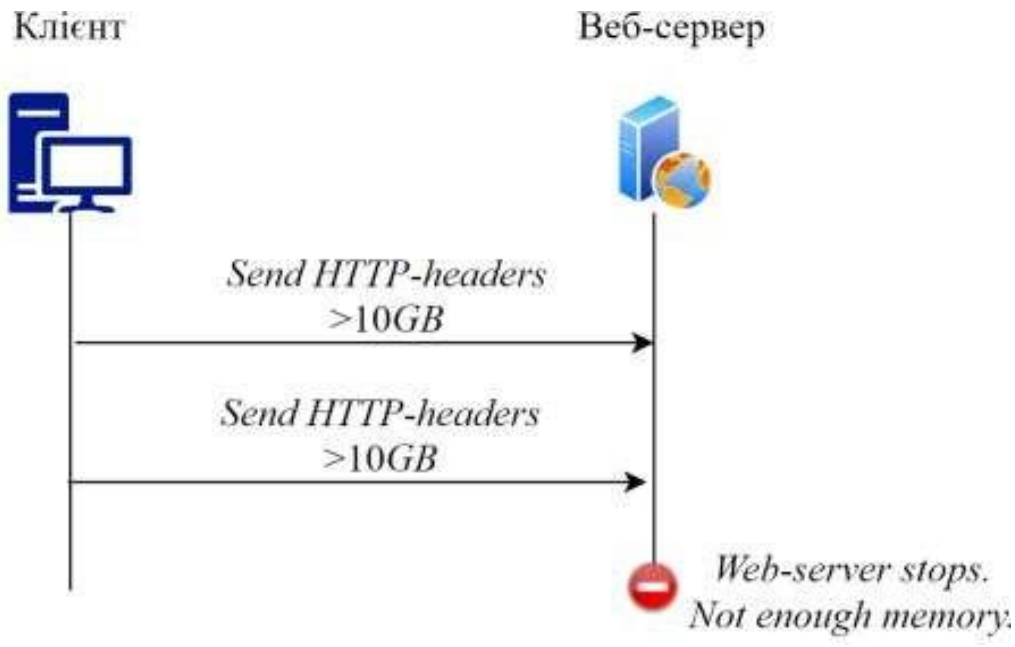


Рисунок.1.2. Переповнення буферу оперативної пам’яті веб-серверу

Розглянемо типи заголовків HTTP:

1. Content-Type: Цей заголовок вказує клієнту на тип медіаресурсу, що повертається. Неправильне використання може призвести до конфліктів в інтерпретації даних або до вразливостей. Клієнти зазвичай намагаються встановити тип ресурсу самостійно у відсутність цього заголовка, що може

призвести до проблем. Використання цього заголовка допомагає у визначенні типу даних, що повертаються.

2. Content-Encoding: Цей заголовок вказує на метод кодування, застосований до тіла повідомлення для передачі через мережу. Наприклад, це може бути gzip або deflate. Приймач повинен декодувати тіло повідомлення перед його подальшою обробкою.

3. Content-Length: Цей заголовок вказує на довжину тіла повідомлення в байтах. Надто велике значення цього заголовка може призвести до перевантаження сервера. Недостатнє значення може призвести до неповної передачі даних.

4. Cache-Control: Цей заголовок контролює кешування клієнта або проксі-сервера. Він містить директиви, які вказують, чи може бути відповідь закешована і як довго вона може бути збережена. Це важливо для оптимізації швидкодії та зменшення навантаження на сервер.

5. Connection: Цей заголовок вказує, чи слід підтримувати з'єднання після завершення поточного запиту. Значення "keep-alive" забезпечує повторне використання з'єднання, що допомагає покращити швидкодію мережі шляхом уникнення зайвих з'єднань.

Ці заголовки грають ключову роль у взаємодії між клієнтом і сервером, забезпечуючи правильну передачу та інтерпретацію даних.

Ви абсолютно праві. Заголовок "Connection" визначає, як треба керувати з'єднанням після завершення обробки поточного запиту. Дозволяючи "keep-alive", сервер підтримує з'єднання відкритим для подальших запитів від того ж клієнта, що може значно скоротити час очікування клієнта на відповідь та зменшити навантаження на сервер шляхом уникнення постійного встановлення нових з'єднань. Навпаки, значення "close" вказує, що після завершення поточного запиту з'єднання має бути закрите.

Відкрите з'єднання зазвичай ефективніше з точки зору продуктивності та ресурсів, оскільки воно уникне постійного встановлення нових з'єднань для

кожного запиту. Така практика зменшує затримки та споживання ресурсів мережі та сервера.

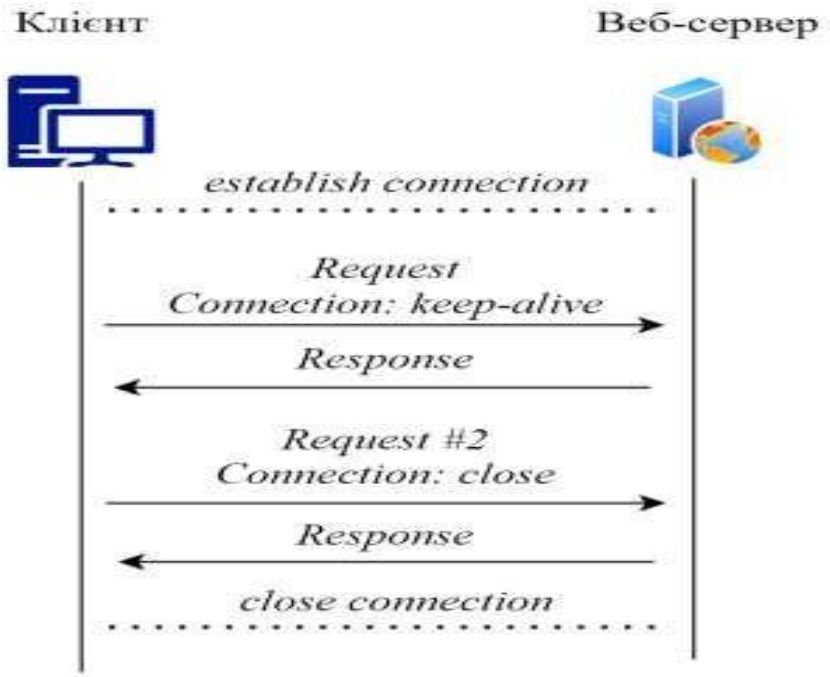


Рисунок 1.3 Процес встановлення з'єднання та обміну даними

Процес встановлення з'єднання та обміну даними між клієнтом і сервером в HTTP можна узагальнити наступним чином:

1. Встановлення з'єднання:

Клієнт встановлює з'єднання з сервером, надсилаючи запит на певну ресурс або URL.

Це зазвичай відбувається через TCP (Transmission Control Protocol), хоча можуть використовуватися інші протоколи, такі як TLS (Transport Layer Security) для забезпечення безпеки.

Сервер відповідає на запит клієнта, підтверджуючи успішне встановлення з'єднання.

2. Передача запиту:

Після встановлення з'єднання клієнт надсилає HTTP-запит серверу. Цей запит містить метод (GET, POST, PUT, DELETE тощо), заголовки та, можливо, тіло запиту з додатковими даними.

3. Обробка запиту:

Сервер отримує запит від клієнта та аналізує його. Він виконує необхідні дії згідно з отриманим запитом, такі як доступ до бази даних, обробка даних або генерація відповіді.

4. Генерація відповіді:

Після обробки запиту сервер генерує HTTP-відповідь.

Ця відповідь містить статусний код, заголовки та, можливо, тіло відповіді з даними або ресурсами.

5. Передача відповіді: Сервер надсилає HTTP-відповідь клієнту через встановлене з'єднання Клієнт отримує відповідь та обробляє її відповідно до власних потреб.

6. Закриття з'єднання:

Після завершення обміну даними клієнт або сервер може закрити з'єднання, освободжуючи ресурси.

Це може відбутися або автоматично після завершення обміну даними, або вручну, якщо одна зі сторін вирішить закрити з'єднання.

Цей процес є основою взаємодії між клієнтом і сервером у протоколі HTTP. Через цей механізм відбувається передача даних та виконання запитів та відповідей.

Керування розподіленими веб-ресурсами передбачає аналіз інформації на рівні мережевого оточення операційної системи (ОС). При цьому сама система керування може використовувати техніки та технології з рівня прикладного програмного забезпечення (ПЗ), наприклад – СУБД. Мета керування розподіленими веб-ресурсами, що розглядається у рамках

дипломної роботи, – здійснення керуючих впливів на процес обміну веб-ресурсами для блокування визначених веб-ресурсів, виявлення інтернет-загроз у веб-трафіку та постійного моніторингу HTTP-з’єднань

1.3 Процедури шифрування даних під час взаємодії з веб-ресурсами.

Одним з найважливіших аспектів процесу обміну даними у веб-середовищі є створення довіреної платформи, де користувачі можуть впевнено здійснювати дії зі своїми особистими даними. Сертифікати SSL виступають основою довіри, забезпечуючи безпечне з'єднання. Використання протоколу HTTPS, зокрема протоколів SSL та його наступника TLS, гарантує шифрування та захист даних користувачів під час їх передачі через мережу під час веб-сесій.

Раніше дані передавалися у відкритому вигляді, що робило їх доступними для перехоплення ким завгодно. Наприклад, при відвідуванні сайтів для покупок та введенні інформації про кредитну картку, номери карток відправлялися у відкритому вигляді. Це сталося за часів протоколу HTTP, який не забезпечував захист даних.

HTTPS був розроблений для вирішення цих проблем безпеки. Він впроваджує заходи захисту конфіденційності користувачів. Протокол HTTPS, протягом свого існування, використовував різні протоколи шифрування даних, такі як SSLv2, SSLv3, TLSv1, TLSv1.1, TLSv1.2, TLSv1.3.

Протоколи SSL та TLS забезпечують такі функції:

1. Аутентифікація та верифікація: Сертифікат TLS/SSL містить інформацію, що підтверджує автентичність користувача, компанії або веб-сайту.

2. Шифрування даних: Сертифікат TLS/SSL забезпечує механізм шифрування даних.

Сертифікат TLS/SSL є свого роду цифровим паспортом, який видається спеціальними центрами сертифікації (CA). Процес видачі сертифікату підпорядковується певним процедурам, щоб підтвердити, що запит на видання сертифікату надійшов від справжнього власника домену. До таких процедур відносяться створення DNS-записів та завантаження файлів CA на веб-сервер.

Протоколи SSL та TLS використовують асиметричну криптографію, що базується на використанні двох ключів: публічного та приватного. Ця криптографія дозволяє шифрувати та розшифровувати дані з використанням різних ключів, що гарантує безпеку обміну даними.

Існують різні типи сертифікатів, такі як однодоменні, мультидоменні та сертифікати з підтримкою ділених доменів, які застосовуються в залежності від потреб користувача.

Таким чином веб-сервер зберігає в таємниці приватний ключ, а публічні віддає клієнтам (рис. 1.4). А процес обміну ключами та встановлення правил криптографії називається рукостисканням (*handshake*).

Рисунок 1.4 Шифрування даних публічним ключом

Протоколи TLS/SSL використовують асиметричну криптографію на початку сесії зв'язку, але лише для процесу рукостискання та встановлення спільного симетричного ключа сеансу. Цей спільний ключ потім використовується для симетричного шифрування та дешифрування даних під час подальшої взаємодії.

Симетрична криптографія, яка базується на використанні одного ключа як для шифрування, так і для дешифрування даних, вимагає менше обчислювальних ресурсів порівняно з асиметричною криптографією. Отже, після встановлення безпечного з'єднання за допомогою асиметричної криптографії для обміну ключами, подальші дані шифруються та дешифруються симетрично з використанням спільного ключа сеансу. Це забезпечує ефективність та безпеку обміну даними в рамках веб-сесії.

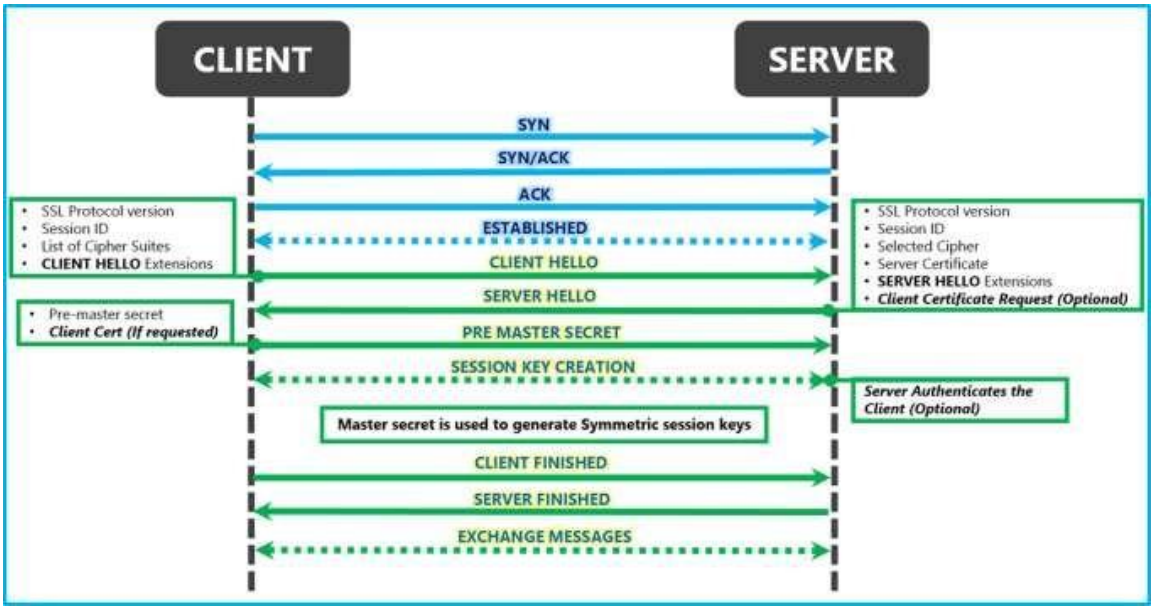


Рисунок 1.5. Процес *handshake*

Розглянемо опис процесу рукостискання та встановлення безпечного з'єднання між клієнтом і веб-сервером. Ось короткий опис кожного кроку:

1. Створення TCP з'єднання: Клієнт встановлює TCP з'єднання з веб-сервером і запитує процес рукостискання, розпочинаючи процес безпечного з'єднання.

2. Надсилання списку підтримуваних шифрів: Веб-сервер відповідає клієнту списком підтримуваних наборів шифрів.

3. Вибір шифру: Клієнт порівнює отриманий список шифрів зі своїм та обирає один для використання в процесі шифрування та дешифрування.

4. Перевірка сертифікату: Веб-сервер надсилає цифровий сертифікат, який клієнт перевіряє на його справжність.

5. Генерація ключа сеансу: Клієнт генерує ключ сеансу та використовує відкритий ключ веб-сервера для його шифрування.

6. Надсилання та дешифрування ключа сеансу: Зашифрований ключ сеансу надсилається веб-серверу, який у свою чергу дешифрує його. Після цього ключ сеансу використовується для шифрування та дешифрування даних протягом усієї сесії.

Ключ сеансу є тимчасовим і діє лише протягом одного з'єднання. У випадку переривання з'єднання процес рукостискання повторюється, і використовується новий ключ сеансу для подальшої комунікації.

Механізми стиснення даних при обміні веб-ресурсами грають важливу роль у зменшенні обсягу передаваних даних та підвищенні швидкості завантаження сторінок. Для цього використовуються різні алгоритми стиснення, які базуються на різних принципах оптимізації даних.

В протоколі HTTP використовуються наступні параметри алгоритмів стиснення:

1. gzip: Використовує кодування Лемпеля-Зіва (LZ77) з 32-бітним CRC.

2. compress: Використовує Лемпеля-Зіва-Велча (LZW).

3. deflate: Використовує структуру zlib з алгоритмом стиснення deflate.

4. identity: Позначає відсутність стискання.

5. br: Використовує алгоритм Brotli.

Ці алгоритми мають забезпечувати зменшення об'єму даних та мінімальний вплив на продуктивність системи. Для вказання використаного алгоритму стиснення у HTTP-заголовку Content-Encoding використовується такий формат:

```

*Content-Encoding: <algorithm>*

```

Крім того, клієнт може вказати підтримувані ним алгоритми стиснення у заголовку Accept-Encoding:

```

Accept-Encoding: <list of supported algorithms>

```

Найчастіше для стиснення даних використовується алгоритм gzip, оскільки він є досить ефективним і широко підтримується браузерами та веб-серверами.

1.4 Механізми управління доступом до веб-ресурсів, що розподілені

Керування доступом до розподілених веб-ресурсів має на меті забезпечити ефективну та безпечну роботу з цими ресурсами для користувачів і систем. Основні мети включають:

1. Безпека: Забезпечення конфіденційності, цілісності та доступності розподілених веб-ресурсів шляхом контролю доступу до них та застосування відповідних аутентифікаційних та авторизаційних механізмів.

2. Ефективність: Забезпечення швидкого та надійного доступу до ресурсів, управління пропускною здатністю та оптимізація використання мережних ресурсів для покращення продуктивності.

3. Масштабованість: Здатність системи ефективно працювати при збільшенні обсягів даних та навантаження, забезпечуючи стабільну роботу в умовах зростаючої кількості користувачів.

4. Управління ресурсами: Ефективне розподілення та управління ресурсами мережі для забезпечення рівноваги навантаження, оптимізації використання та підтримки високої доступності.

Методологія обробки даних в сучасних інформаційних системах базується на принципі багаторівневості, що означає розділення функціональності системи на рівні, які взаємодіють між собою. Це дозволяє покращити модульність, масштабованість та підтримуваність системи, а також спрощує розробку та управління.

Розроблювана система, що керує доступом до розподілених веб-ресурсів, є частиною інформаційної системи. Утворення зв'язків у цій системі буде відбуватися з урахуванням архітектурної моделі інформаційної системи.

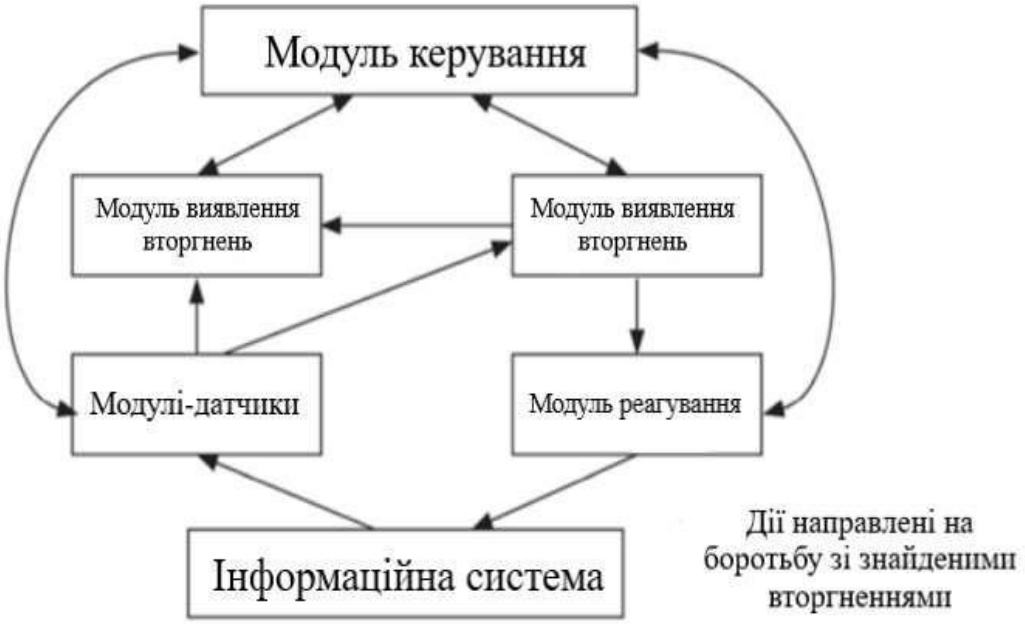


Рисунок 1.6 Приклад моделі інформаційної системи

1.5 Аналіз впливів, що керують системою

Можливість доступу до веб-ресурсів через інтернет визначається можливістю завантажити вміст за допомогою URL-посилання. Цей процес може потребувати авторизації та/або автентифікації. Ключові етапи цього процесу включають:

1. Введення URL користувачем та надсилання запиту.

2. Трансляція доменного імені у відповідну IP-адресу через DNS-сервери або кеш ОС (необов'язково).

3. Встановлення TCP-з’єднання між клієнтом і веб-сервером.

4. Надсилання HTTP-запиту веб-серверу від клієнта.

5. Отримання та обробка HTTP-відповіді веб-сервером.

6. Передача HTTP-відповіді назад клієнту.

7. Відображення інформації користувачем клієнтським програмним забезпеченням.

Для мінімізації втручання у роботу системи можна застосовувати керівний вплив на етапі встановлення з’єднання та передачі даних між клієнтом і сервером. Розглянемо можливі керівні впливи:

1. Створення та розривання HTTP-сесій: Цей вплив дозволяє контролювати процес передачі даних між вузлами мережі в будь-який момент часу.

2.Аналіз та редагування вмісту HTTP-повідомлень: Цей вплив дозволяє маніпулювати інформацією перед її передачею.

3. Редагування формату даних HTTP-повідомлень: Цей вплив спрощує процеси аналізу інформації.

4.Блокування та перенаправлення HTTP-трафіку: Цей вплив дозволяє обмежувати доступ до певних веб-ресурсів.

Кожен з цих впливів має свої переваги та можливі наслідки.

Наприклад, блокування та перенаправлення трафіку можуть бути використані для захисту від шкідливих веб-сайтів, але можуть також обмежити доступ користувачів до певної інформації.

Таким чином, вибір конкретного керівного впливу повинен залежати від потреб системи та цілей власника ресурсу.

Відповідно до проведеного дослідження, керування доступом до розподілених веб-ресурсів може базуватися на різноманітних темах та застосовуватися з різною метою. У дипломному проекті будуть досліджені такі керівні впливи та теми:

1. Аналіз вмісту тіла HTTP-повідомлень на наявність певних інтернет-загроз: Дослідження цієї теми спрямоване на виявлення та відсіювання потенційно небезпечного веб-трафіку, що може містити шкідливі програми або шкідливі

веб-сайти.

2. Блокування HTTP-трафіку для визначених веб-ресурсів: Ця тема передбачає розробку механізмів для блокування доступу до певних веб-ресурсів з метою заборони доступу до небажаних чи шкідливих вмістів.

3. Редагування формату даних HTTP-повідомлень для підготовки до подальшого аналізу: Цей вплив спрямований на обробку та оптимізацію HTTP-трафіку шляхом зміни формату даних з метою полегшення подальшого аналізу та обробки.

4. Створення та розривання HTTP-сесій для з’єднання клієнтів та веб-серверів: Ця тема охоплює розробку механізмів керування HTTP-сесіями з метою забезпечення безпеки та оптимізації процесу передачі даних між клієнтами та серверами.

Ці теми є ключовими у розгляді керівних впливів на доступ до розподілених веб-ресурсів та дозволять виявити методи та механізми для ефективного керування доступом у веб-середовищі.

1.6 Аналіз інтернет - загроз.

Відповідно до дослідження, аналіз інтернет-загроз є важливим аспектом захисту інформаційних систем. У зв'язку з постійним зростанням складності та масштабів кібератак, необхідно виявляти та запобігати різноманітним загрозам.

Для цього в дипломній роботі будуть досліджені такі типи інтернет-загроз:

1. Шкідлива програма (трояни, програми-вимагачі): Ці загрози спрямовані на інфікування комп'ютерів користувачів з метою отримання несанкціонованого доступу або вимагання викупу.

2. Фішинг: Зловмисники намагаються виманити конфіденційну інформацію (наприклад, логіни та паролі) шляхом видачі себе за довірену сторону.

3. Спам: Надсилання небажаних повідомлень з рекламою або шкідливими посиланнями.

4. Прихований майнінг: Використання ресурсів комп'ютера для добування

криптовалют без належного дозволу власника.

5. Крадіжка особистої інформації: Незаконне здобуття конфіденційних даних користувачів для подальшого використання або продажу.

6. Бекдор: Створення та використання задніх дверей у програмному забезпеченні для отримання несанкціонованого доступу.

7. Комп'ютерний вірус: Шкідливе програмне забезпечення, яке внедрюється у систему та розповсюджується, завдаючи шкоду.

8. Експлойт: Використання вразливостей у програмному забезпеченні чи апаратному забезпеченні для здійснення атаки.

9. Шпигунські програми: Загрози, які використовуються для збору конфіденційної інформації без належного дозволу.

10. Мережевий хробак: Саморозповсюджующийся код, який використовується для інфікування мереж та систем.

11. Рекламне ПЗ: Програмне забезпечення, яке відображає нав'язливу рекламу та може містити шкідливий код.

12. Соціальна інженерія: Використання маніпуляційних технік для отримання конфіденційної інформації від користувачів.

Мною було розглянуто лише ті типи інтернет-загроз, які можуть бути виявлені за допомогою аналізу веб-трафіку та мають чіткі сигнатури.

Для цього використовувуться спеціалізовані системи, такі як SNORT та SURICATA, які здатні виявляти та блокувати потенційно небезпечний трафік у реальному часі.

1.7 Проблеми та мета захисту комп’ютерних мереж

Розглянемо проблеми та мету захисту комп'ютерних мереж, що стають все більш актуальними з розвитком Інтернету та зростанням загроз кібербезпеці. Тут наведено кілька ключових моментів, які варто врахувати:

1. Збільшення загроз: Розвиток технологій та доступу до Інтернету

призводить до зростання загроз для користувачів. З кожним днем з'являється все більше користувачів, що створює нові потенційні цілі для кіберзлочинців.

2. Дослідження загроз: Вчені вже давно вивчають загрози комп'ютерних мереж і розробляють методи їх виявлення та захисту. Ваша робота надає огляд різних підходів, від теоретичних моделей до практичних методів аналізу та виявлення загроз.

3. Необхідність адаптації: Запобігання загрозам вимагає постійного адаптування та удосконалення методів захисту. Особливо важливою є реакція на нові стандарти та вимоги, такі як GDPR, які регулюють обробку особистих даних та встановлюють високі стандарти приватності.

4. Мета захисту: Основна мета захисту комп'ютерних мереж полягає в своєчасному виявленні, аналізі та блокуванні можливих загроз. Реагування на загрози на ранніх етапах дозволяє уникнути серйозних наслідків та забезпечити стабільність системи.

5. Виклики для систем захисту: Розробка ефективних систем захисту потребує збалансованості між ефективністю та мінімізацією впливу на швидкодію мережі.

Крім того, важливо постійно вдосконалювати методи аналізу та реагування на загрози.

Загалом, мною було ретельно розглянуто проблематику кібербезпеки та висвітлено важливі аспекти захисту комп'ютерних мереж у сучасному інтернет-середовищі.

Компанії змушені витрачати величезні кошти на:

1. Мережеву безпеку.

2. Підвищення кваліфікації працівникам IT-сектору у сфері комп’ютерної безпеки.

3. Захист та мінімізацію ризиків вірусних вторгнень.

4. Впровадження політик виявлення та запобігання вторгнень.

5. Резерви даних, правил та процесів їх відновлень.

6. Визначення політик та процедур безпеки.

Безпосередній несанкціонований доступ зазвичай тримується через ряд причин:

1. Слабкість в контрольованій системі з неефективними елементами управління.

2. Помилки в ПЗ та відсутність своєчасного виправлення цих помилок.

3. Необачність працівників системи.

Операційні системи піддаються несанкціонованому доступу при умові запуску на ній сторонніх додатків з шкідливим кодом. Такими додатками можуть бути різноманітні файли, як виконуючі, так і ні. У деяких випадках сторонні додатки виступають у вигляді окремих плагінів для інших програм, наприклад плагіни для *Microsoft Word*, *Microsoft Excel* тощо. Через загрозу вірусів розробник вищезазначених програм передав відповідальність за можливі несанкціоновані дії накористувача – просто надавши окрему опцію для вмикання плагінів, а за замовченнямвони вимкнені.

Тобто компанії з такими ресурсами як *Microsoft* не мають змоги повноцінно боротись з вірусами, що розповсюджуються через створене ними ж ПЗ, а основна реакція на це – лише ненадійна латка та перекидання відповідальності у плагіні.

Рисунок 1.7. Приклад попередження *Microsoft Excel* про можливі віруси

Несанкціонований доступ може бути отриманий як до ресурсів самої ОС, так і до ресурсів окремих програм. Причини виникнення вищезазначених дій та зловживань описані у роботах авторів [3, 4], а необхідні умови для цього наступні:

1. Відкритий код ядра ОС (*opensource*).
2. Розповсюдженість та популярність ОС.
3. Велика кількість багів та помилок у ядрі ОС (як відомих, так і ні).

Отримання несанкціонованого доступу до ОС – це наслідок відсутності або ж слабкого захисту мережевого оточення комп'ютеру, на якій встановлена ця ОС. Томує сенс не вирішувати наслідки, а знайти першопричину та боротися в першу чергу з нею.

Безпека будь-якої системи залежить від набору програмних та архітектурних рішень по запобіганню отримання доступу до її керуючого ядра та файлів.

Часто керівні вузли мають параметри та налаштування, які для зручності та доступності зберігаються прямо в постійній пам'яті у вигляді файлів налаштувань. Хоча такий спосіб є загальноприйнятним з точки зору зручності користування будь-якою системою він привносить доволі небезпечні механізми, за допомогою яких частини системи, або ж уся система у цілому, може бути скомпрометована.

Тому використання загальнодоступних файлів налаштувань є можливим рішенням при розробці системи, але використання подібного методу на доступній та працюючій системі з великою кількістю користувачів є неприйнятним. Потрібно пам'ятати, що отримати доступ до динамічної оперативної пам'яті набагато важче, ніж до статичної.

Методи виявлення інтернет-загроз у комп'ютерних мережах є важливою складовою заходів забезпечення кібербезпеки. Ось деякі з найпоширеніших методів:

1. Системи виявлення вторгнень (IDS): IDS аналізують мережевий трафік для виявлення ненормальних чи підозрілих активностей, які можуть

вказувати на потенційні атаки або вторгнення. Вони можуть бути базованими на підписах (виявлення відомих атак) або аналізувати аномалії (незвичайні патерни в трафіку).

2. Системи запобігання вторгненням (IPS): IPS доповнюють роль IDS, не лише виявляючи потенційні загрози, але й автоматично вживаючи заходів для їх блокування або виходу на стійку оборону.

3. Аналіз журналів подій (логів): Моніторинг та аналіз журналів подій може допомогти виявити незвичайні дії або активності, які можуть вказувати на атаки або несанкціонований доступ.

4. Аналіз поведінки користувачів (UBA): Цей метод використовується для виявлення аномальної поведінки користувачів, яка може свідчити про компрометацію облікових записів або інші кіберзагрози.

5. Системи виявлення зловмисного програмного забезпечення (Malware Detection Systems): Ці системи виявляють наявність шкідливого програмного забезпечення, такого як віруси, троянці, шпигунське програмне забезпечення тощо, у мережевому трафіку чи на окремих комп'ютерах.

6. Моніторинг мережевої активності: Цей метод включає постійний моніторинг мережевого трафіку для виявлення незвичайних або підозрілих патернів активності.

7. Використання сигнатур із відомих загроз: Цей метод базується на використанні відомих сигнатур вірусів та інших типів загроз для виявлення їх у мережевому трафіку або на комп'ютерах.

Ці методи часто використовуються в комбінації для максимального забезпечення безпеки комп'ютерних мереж від інтернет-загроз.

Методи знешкодження вірусів також починають бути неефективними.

У результаті комерціалізації сфери шкідливого ПЗ з’явилися нові техніки, якізабезпечують:

1. Приховування шкідливого коду серед звичайного.
2. Мутуючий код, який при запуску з звичайного перетворюється на шкідливий.

3. Перешкоджання видалення шляхом клонування та вимкнення деяких системних функцій.
4. Контроль цілісності вірусної системи.

Ці техніки негативно впливають на результати розпізнання та видалення вторгнень.

Споживання системних ресурсів також впливає на систему. Ефективність будь-якого захисту від інтернет-загроз залежить від моніторингу веб-трафіку у режимі реального часу, що вимагає активної роботи з системними механізмами та подіями для фільтрації потоків можливих загроз. Як показує практика, активне використання системних функцій призводить до сповільнення роботи програм. Тому більшість користувачів, з причини відсутності необхідних високошвидкісних елементівпостійної пам'яті (*SSD*) та потужних *CPU*, вимикають будь-який захист ОС таким чином ставлячи себе під ризик зовнішніх загроз.

Також існує проблема несумісності різних антивірусних програм через конфлікти під час перехоплення та фільтрації системних подій. Загалом – це несуттєва проблема, проте певне антивірусне ПЗ добре захищає від одного типу загроз, а інше – від іншого, і як результат неможливо створити єдиний захисний екранв межах однієї антивірусної програми.

Мета системи виявлення загроз – це своєчасно аналізувати, виявляти та блокувати можливі загрози. Ранній етап виявлення можливої загрози забезпечує стабільність системи та зменшує негативні наслідки. Тому властивість виявлення загрози на ранніх стадіях відноситься до найважливіших у системах виявлення загроз.

При низькому значенню цієї властивості загальна користь від такої системи падає у геометричній прогресії. Підвищення складності програмного забезпечення (ПЗ) призводить до аналогічного ускладнення вірусного ПЗ. Вірусне ПЗ еволюціонує та його програмні елементи можливо знайти на чорних ринках ПЗ.

Ці можливості завдають шкоди звичайним користувачам, корпораціям, установам, а іноді – економіці та стабільності цілих країн (приклад – вірус Petya). Програмні елементи вірусного ПЗ складаються з невеликих програмних блоків вбудованих до виконуваних файлів звичайних програм, з цілих систем взаємопов'язаних компонентів для виконання конкретних завдань: віруси-маскувальники, інсталятори тощо. Основне джерело вірусного ПЗ – це мережа Інтернет.

Способи та методи виявлення загроз у комп'ютерних мережах – різноманітні та багатогранні. Тим не менш, оцінка якості виконання їх прямих обов'язків не завжди достатня для загального оцінювання. Будь-який додатковий елемент у системі може призвести до непропорціонального росту її складності. Наряду з виконанням своїх прямих обов'язків такі системи також повинн

1.8 Методи та засоби керування доступом до розподілених веб-ресурсів

Методи керування доступом:

1. Аутентифікація та авторизація**:**

Використання паролів, двофакторної аутентифікації (2FA), біометричних даних.

Рольова модель доступу (RBAC), атрибутивна модель доступу (ABAC).

OAuth, OpenID Connect для управління доступом до ресурсів сторонніх сервісів.

2. Шифрування:

SSL/TLS для захисту даних під час передачі між клієнтом і сервером.

Шифрування даних на рівні зберігання за допомогою AES, RSA.

3. VPN та проксі-сервери:

Віртуальні приватні мережі (VPN) для захищеного доступу до внутрішніх ресурсів організації.

Використання проксі-серверів для фільтрації та контролю доступу до веб-ресурсів.

4. Контроль доступу на рівні брандмауера:

Брандмауери для моніторингу та контролю мережевого трафіку на основі заздалегідь визначених правил. Використання веб-брандмауерів (WAF) для захисту веб-додатків від атак.

5. Захист від атак:

Використання системи запобігання вторгненням (IPS) та виявлення вторгнень (IDS).

Засоби захисту від DDoS-атак, такі як Cloudflare, Akamai.

Засоби керування доступом:

1. Системи управління ідентифікацією (IAM):

Azure Active Directory, AWS Identity and Access Management (IAM), Okta.

2. Платформи управління доступом:

Auth0, Ping Identity, OneLogin.

3. Антивірусне та антишкідливе програмне забезпечення:

AVZ, KasperskyLabs, ESET для виявлення та блокування шкідливих програм.

4. Інструменти моніторингу та аналізу трафіку:

Wireshark, Splunk для аналізу та виявлення підозрілого трафіку.

Zabbix, Nagios для моніторингу мережевої інфраструктури.

5. Веб-брандмауери (WAF):

ModSecurity, F5 BIG-IP для захисту веб-додатків від поширених загроз, таких як SQL-ін'єкції, XSS-атаки.

6. Системи захисту від DDoS-атак:

Cloudflare, Akamai для забезпечення безперервності роботи веб-ресурсів під час масових атак.

Ці методи та засоби забезпечують багаторівневий підхід до захисту та керування доступом до розподілених веб-ресурсів, що дозволяє ефективно захищати дані та сервіси від різноманітних загроз.

Створення системи керування доступом до розподілених веб-ресурсів вимагає високих інженерних навичок та знань у сфері проектування та розробки програмного забезпечення, системного та мережевого програмування. Перед початком програмування системи необхідно визначити наступні аспекти:

1. Формат роботи системи та її топологію.
2. Складові частини (класи, об’єкти, компоненти).
3. Взаємозв’язки між складовими частинами.
4. Процеси передачі даних у системі.
5. Механізми зберігання даних та читання конфігураційних файлів.

Крім системних вимог, необхідно описати процеси та механізми реалізації функціональних вимог:

1. Реалізацію виявлення інтернет-загроз у веб-трафіку в режимі реального часу: база даних сигнатур, алгоритм перевірки веб-трафіку, взаємопов’язані процеси, алгоритми дій у випадку виявлення інтернет-загроз.
2. Механізм блокування доступу користувачів до визначених веб-ресурсів: процес створення конфігурації, сам механізм блокування, виведення повідомлення про блокування користувачу.
3. Процеси моніторингу мережевого трафіку: запис основних даних до бази даних.

На рис. 1.8 представлена загальна топологія та розташування вузлів у системі. Розроблювана система повинна бути розміщена між хостами мережі для ефективного виявлення інтернет-загроз у веб-трафіку.

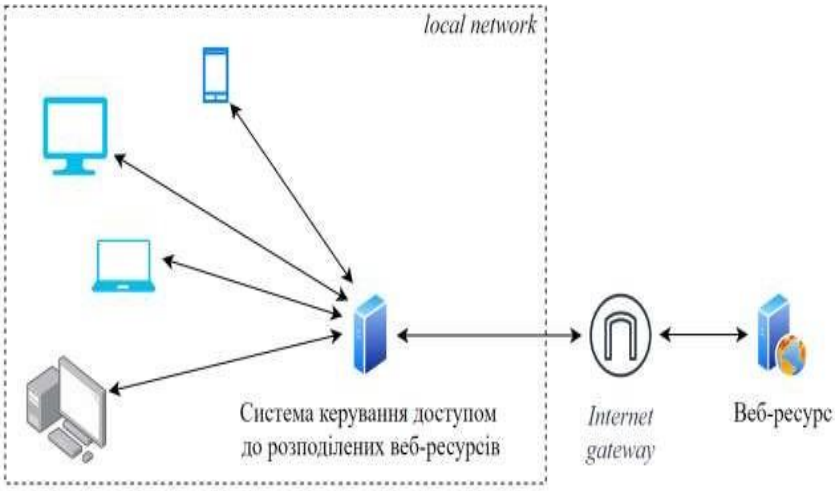


Рисунок 1.8. Приклад схеми топології системи

Я розглянув технологій та процесів розподілених веб-ресурсів і дійшов висновку: поняття веб-ресурсу, описані механізми обміну веб-ресурсами, розглянуті особливості шифрування та стиснення даних при обміні веб-ресурсами. З'ясовано, що веб-ресурс – це узагальнене поняття одиниці корисної інформації, що може бути представлена у текстовому чи медіа форматі. Доступ до веб-ресурсів здійснюється за допомогою мережі та веб-ресурс ідентифікується у ній через унікальний ідентифікатор ресурсу – URL. Особлива увага приділив механізмам обміну веб-ресурсами, оскільки для створення системи керування доступом до розподілених веб-ресурсів необхідно повністю розуміти усі механізми та правила обміну ними.

Встановив, що основний протокол для здійснення вищезазначених операцій – HTTP. Детально розглянуті деякі процеси, що використовуються протоколом HTTP. Вони будуть використані у подальшому під час розробки системи керування доступом до розподілених веб-ресурсів. Додатково проаналізував шифрування та стиснення веб-трафіку перед їх передачею по мережі.

Встановлено, що використання зазначених механізмів обґрунтовується збільшенням об'єму та розмірів веб-трафіку. Шифрування веб-трафіку обумовлюється створенням нових методів для несанкціонованого доступу до персональних даних користувачів під час обміну даними між клієнтами та веб-серверами.

1.9 Практична реалізація системи керування доступом до розподілених веб-ресурсів

При розробці системи керування доступом до розподілених веб-ресурсів мені необхідно врахувати та вирішити наступні задачі:

1.Передача даних.

2.Аналіз та обробка даних.

4. Зберігання даних.

Відповідно до вищеописаних задач розроблювана система повинна реалізовувати методи та механізми, що:

1.Реалізують механізми передачі даних.

2.Реалізують методи аналізу даних з метою подальшого прийняття керуючих рішень.

3Зберігають результати перевірок та ведуть журнал дій.

Сформуємо функціональні вимоги.

Необхідно спроектувати та реалізувати систему керування доступом до розподілених веб-ресурсів, що виконуватиме наступні завдання:

1.Аналіз вхідного мережевого трафіку на наявність інтернет-загроз (вірусів, хробаків, троянів тощо) методом сигнатурного аналізу

2. Блокування доступу до веб-ресурсів з можливістю ручного налаштування списку таких веб-ресурсів.

3. Здійснювати постійний моніторинг веб-трафіку та вести записи по основним мережевим характеристикам кожного вузла у окремий журнал (кількість вузлів, кількість вхідного/вихідного мережевого трафіку по кожному вузлу, виявлені інтернет-загрози).

Процес розробки системи вимагає:

1. Формування специфікації ПЗ.

2.Проектування ПЗ.

3. Програмування та тестування ПЗ.

5. Розгортання ПЗ.

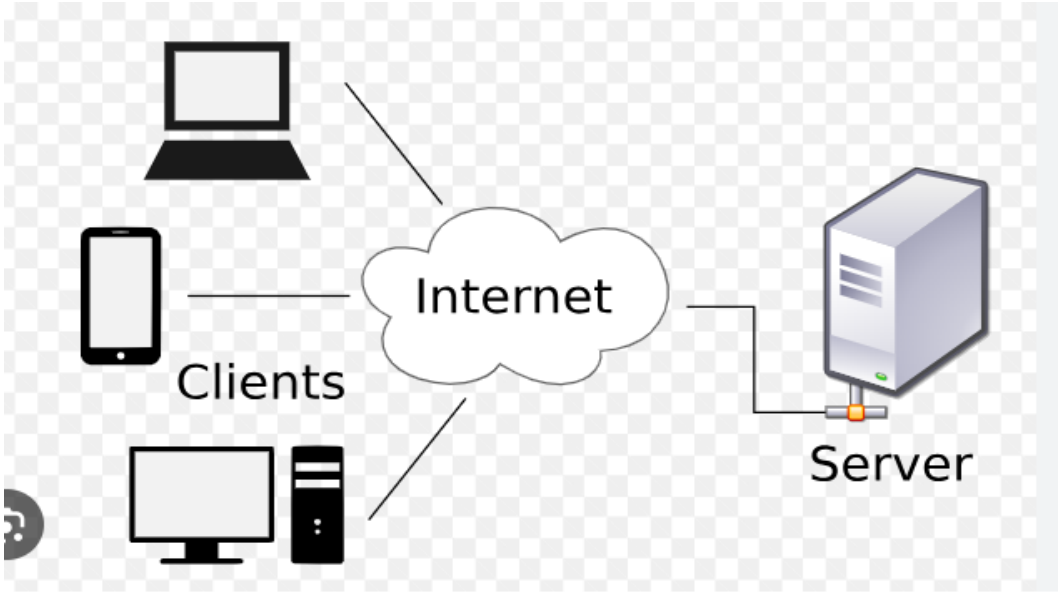


Рисунок 1.9.Приклад архітектури клієнт-сервер.

1. Створюю просту розподілену інформаційну систему, де сервер обробляє запити від клієнта. Клієнт надсилає до сервера текстове повідомлення, а сервер повертає його в зміненому вигляді.

Підготовка середовища

Переконуюсь, що на момукомп'ютері встановлено Node.js. Це можна зробити, запустивши команду `node -v` у терміналі. Якщо Node.js не становлено, завантажуємо його з офіційного сайту nodejs.org.

2. Створення сервера

Створюю файл `server.js` і додаю до нього наступний код на javascript:

```
const express = require('express');
const app = express();
const port = 3000;
app.get('/', (req, res) => {
  res.send(' ');
});
    app.listen(port, () => {
```

```
  console.log(`Сервер запущено на http://localhost:${port}`);
});
```

Код наразі не обробляє можливі помилки, які можуть виникнути під час читання або запису даних. Мною було додано обробник подій error до об’єктів req та res для обробки таких ситуацій.

Перевірка типу контенту:

Якщо я очікую, що запит буде містити текстові дані, я додав перевірку Content-Type заголовка запиту, щоб переконатися, що він відповідає очікуваному типу (наприклад, text/plain).

Далі створюємо клієнта, який надає до сервера запити. Код на javascript:

```
const express = require('express');
const app = express();
const port = 3000;
app.get('/', (req, res) => {
  res.send('Привіт Світ!');
});
app.listen(port, () => {
  console.log(`Сервер запущено на http://localhost:${port}`);
});
```

Ліміт розміру тіла запиту:

Щоб захистити сервер від атаки типу “відмова в обслуговуванні” (DoS), мною було встановлено максимальний розмір тіла запиту і припинення читання даних, якщо цей ліміт перевіщено:

Код `server.js`

server.js

```
const http = require('http');
const url = require('url');
```

```
const fs = require('fs');

 const hostname = '127.0.0.1';

const port = 3000;

const MAX_BODY_SIZE = 1e6; // 1 мегабайт

let blockedUrls = [];

 // Список заблокованих URL let trafficLog = {};

// Журнал трафіку // Завантаження списку заблокованих URL з файлу const loadBlockedUrls = () => { try { const data = fs.readFileSync('blockedUrls.json', 'utf8');

blockedUrls = JSON.parse(data);

} catch (err) { console.error('Помилка завантаження заблокованих URL:', err);

 } };

// Збереження списку заблокованих URL у файл const saveBlockedUrls = () => { try { fs.writeFileSync('blockedUrls.json', JSON.stringify(blockedUrls), 'utf8');

} catch (err) { console.error('Помилка збереження заблокованих URL:', err);

} }; // Аналіз та блокування вхідного трафіку

 const analyzeTraffic = (req) => { const parsedUrl = url.parse(req.url, true);

if (blockedUrls.includes (parsedUrl.hostname)) { return true; }

return false;

 }; // Оновлення журналу трафіку

const updateTrafficLog = (req, res) => { const clientIp =
req.connection.remoteAddress;

if (!trafficLog[clientIp]) { trafficLog[clientIp] = { in: 0, out: 0, threats: [] };

} trafficLog[clientIp].in += parseInt(req.headers['content-length'] '0');
```

```
trafficLog[clientIp].out += Buffer.byteLength(res.body ");
 };
//
```

Цей код додає обробку помилок, перевірку Content-Type заголовка та ліміт розміру тіла запиту.

Створюю «клієнта»:

Створюємо модіфікований код для файлу `client.js` і додаю до нього наступний код:

Код client.js

```
client.js
const http = require('http');
const options = {
hostname: '127.0.0.1',
port: 3000, path: '/message',
method: 'POST',
headers: {
 'Content-Type': 'text/plain',
 },
};
const req = http.request(options, res => {
if (res.statusCode !== 200) {
console.error (Помилка: отримано статус ${res.statusCode});
return;
} console.log(Статус: ${res.statusCode});
res.on('data', d => { process.stdout.write(Відповідь від сервера: ${d}\n);
 });
```

Цей код додає обробку помилок та перевірку статусу відповіді до нашого оригінального коду.

4. Тестування:

1. Відкрийте два термінали.

2. У першому терміналі запускаємо сервер командою `node server.js`, на рисунку 1.10.

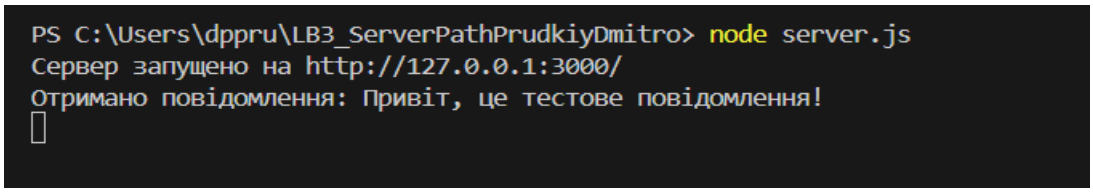


Рисунок. 1.10. Скріншот повідомлення від сервера

4. У другому терміналі запустіть клієнта командою `node client.

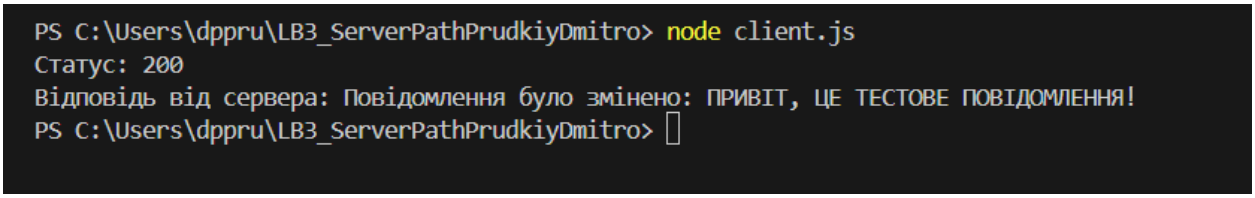


Рисунок 1.11. Скріншот повідомлення від «клієнта»

Мною було розроблено Сервер успішно запущено і він коректно обробляє POST-запити від клієнта. Клієнт відправляє повідомлення “Привіт, це тестове повідомлення!” на сервер, а сервер відповідає, перетворюючи це повідомлення на верхній регістр: “ПРИВІТ, ЦЕ ТЕСТОВЕ ПОВІДОМЛЕННЯ!”.

Це відповідає очікуваному поведінку, описаному в ваших вимогах.

Далі реалізую системи керування доступом до розподілених веб-ресурсів, згідно до постановлених вимог мені потрібно урахувати, що при розробці системи керування доступом до розподілених веб-ресурсів мені необхідно врахувати та вирішити наступні задачі:

1.Передача даних.

2.Аналіз та обробка даних.

4. Зберігання даних.

Відповідно до вищеописаних задач розроблювана система повинна реалізовувати методи та механізми, що:

1.Реалізують механізми передачі даних.

2.Реалізують методи аналізу даних з метою подальшого прийняття керуючих рішень.

3.Зберігають результати перевірок та ведуть журнал дій.

Сформуємо функціональні вимоги.

Необхідно спроектувати та реалізувати систему керування доступом до розподілених веб-ресурсів, що виконуватиме наступні завдання:

1.Аналіз вхідного мережевого трафіку на наявність інтернет-загроз (вірусів, хробаків, троянів тощо) методом сигнатурного аналізу.

2. Блокування доступу до веб-ресурсів з можливістю ручного налаштування списку таких веб-ресурсів.

3. Здійснювати постійний моніторинг веб-трафіку та вести записи по основним мережевим характеристикам кожного вузла у окремий журнал (кількість вузлів, кількість вхідного/вихідного мережевого трафіку по кожному вузлу, виявлені інтернет-загрози).

Процес розробки системи вимагає:

1. Формування специфікації ПЗ.
2.Проектування ПЗ.
3. Програмування та тестування ПЗ.
4. Розгортання ПЗ.

Модифіковані коди:

Модифікований програмний код «*CLIENT*»

client.js

```
const http = require('http');
const options = {
  hostname: '127.0.0.1',
  port: 3000,
```

```
  path: '/message',
  method: 'POST',

  headers: {
    'Content-Type': 'text/plain',
  },
};

const req = http.request(options, res => {
  if (res.statusCode !== 200) {
    console.error(Помилка: отримано статус ${res.statusCode});
    return;
  }
  console.log(Статус: ${res.statusCode});
  res.on('data', d => {
    process.stdout.write(Відповідь від сервера: ${d}\n);
  });
});

req.on('error', error => {
  console.error(Помилка: ${error});
});

// Надсилання повідомлення на сервер
req.write('Привіт, це тестове повідомлення!');
req.end();

// Функція для блокування URL
const blockUrl = (urlToBlock) => {
  const blockOptions = {
```

```
    hostname: '127.0.0.1',
    port: 3000,
    path: '/block',
    method: 'POST',
    headers: {
      'Content-Type': 'text/plain',
    },
  };
  const blockReq = http.request(blockOptions, res => {
    if (res.statusCode !== 200) {
      console.error(Помилка блокування: отримано статус ${res.statusCode});
      return;

    }
    console.log(URL успішно заблоковано);
    res.on('data', d => {
      process.stdout.write(Відповідь від сервера: ${d}\n);
    });
  });
  blockReq.on('error', error => {
    console.error(Помилка: ${error});
  });
  blockReq.write(urlToBlock);
  blockReq.end();
};
// Використання функції для блокування URL
blockUrl('http://example.com');
```

Модифікований програмний код «Server»:

server.js

```
const http = require('http');
const url = require('url');
const fs = require('fs');
const hostname = '127.0.0.1';
const port = 3000;
const MAX_BODY_SIZE = 1e6; // 1 мегабайт
let blockedUrls = []; // Список заблокованих URL
let trafficLog = {}; // Журнал трафіку
// Завантаження списку заблокованих URL з файлу
const loadBlockedUrls = () => {
  try {
    const data = fs.readFileSync('blockedUrls.json', 'utf8');
    blockedUrls = JSON.parse(data);
  } catch (err) {
    console.error('Помилка завантаження заблокованих URL:', err);
  }
};

// Збереження списку заблокованих URL у файл
const saveBlockedUrls = () => {
  try {
    fs.writeFileSync('blockedUrls.json', JSON.stringify(blockedUrls), 'utf8');
  } catch (err) {
    console.error('Помилка збереження заблокованих URL:', err);
  }
};

// Аналіз та блокування вхідного трафіку
```

```
const analyzeTraffic = (req) => {
  const parsedUrl = url.parse(req.url, true);
  if (blockedUrls.includes(parsedUrl.hostname)) {
    return true;
  }

  return false;
};

// Оновлення журналу трафіку
const updateTrafficLog = (req, res) => {
  const clientIp = req.connection.remoteAddress;
  if (!trafficLog[clientIp]) {
    trafficLog[clientIp] = { in: 0, out: 0, threats: [] };
  }
  trafficLog[clientIp].in += parseInt(req.headers['content-length'] '0');
  trafficLog[clientIp].out += Buffer.byteLength(res.body  '');
};

// Збереження журналу трафіку у файл
const saveTrafficLog = () => {
  try {
    fs.writeFileSync('trafficLog.json', JSON.stringify(trafficLog, null, 2), 'utf8');
  } catch (err) {
    console.error('Помилка збереження журналу трафіку:', err);
  }
};

// Завантаження заблокованих URL при старті сервера
loadBlockedUrls();
```

```
const server = http.createServer((req, res) => {
  if (req.method === 'POST' && req.url === '/message' && req.headers['content-type']
=== 'text/plain') {
    let body = '';
    req.on('data', chunk => {
      body += chunk.toString();
      if (body.length > MAX_BODY_SIZE) {
        res.statusCode = 413;
        res.end('Request Entity Too Large');
        req.connection.destroy();
      }
    });
    req.on('end', () => {
      if (analyzeTraffic(req)) {
        res.statusCode = 403;
        res.end('Access to this resource is blocked');
        return;
      }

  console.log(Отримано повідомлення: ${body});
      res.statusCode = 200;

      res.setHeader('Content-Type', 'text/plain');
      res.body = Повідомлення було змінено: ${body.toUpperCase()};
      res.end(res.body);
      updateTrafficLog(req, res);
      saveTrafficLog();
    });
    req.on('error', err => {
      console.error(err);
```

```
      res.statusCode = 500;
      res.end('Internal Server Error');
    });
  } else if (req.method === 'POST' && req.url.startsWith('/block')) {
    let urlToBlock = '';
    req.on('data', chunk => {
      urlToBlock += chunk.toString();
    });
    req.on('end', () => {
      blockedUrls.push(urlToBlock);
      saveBlockedUrls();
      res.statusCode = 200;
      res.end(URL заблоковано: ${urlToBlock});
    });
  } else {
    res.statusCode = 404;
    res.end('Not Found');
  }
});

server.listen(port, hostname, () => {
  console.log(Сервер запущено на http://${hostname}:${port}/);
});
```

При виконанні завдання було виконано:

Аналіз вхідного трафіку: Додана функція analyzeTraffic, яка перевіряє вхідний трафік на наявність заблокованих URL.

Блокування доступу: Додана можливість блокувати доступ до певних URL через POST-запит на /block.

Моніторинг і журналювання: Додана функція updateTrafficLog, яка веде облік трафіку по кожному клієнту.

Завантаження та збереження заблокованих URL: Додані функції loadBlockedUrls та saveBlockedUrls для управління списком заблокованих URL.

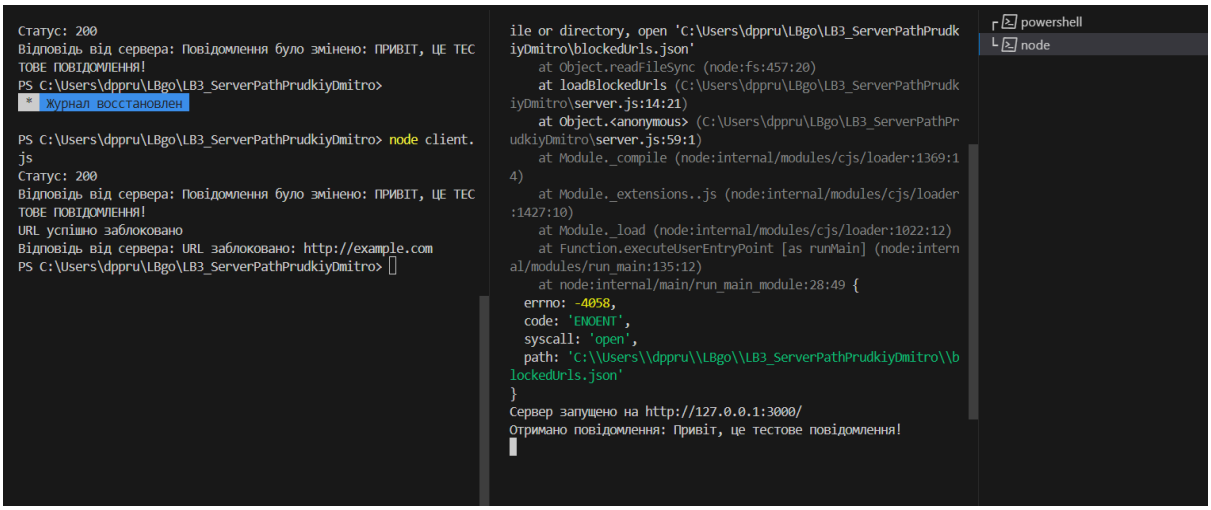


Рис. 1.12. Скриншот запуска сервера.

Рис. 1.13. Скриншот запуска сервера(trafficlog.json.)

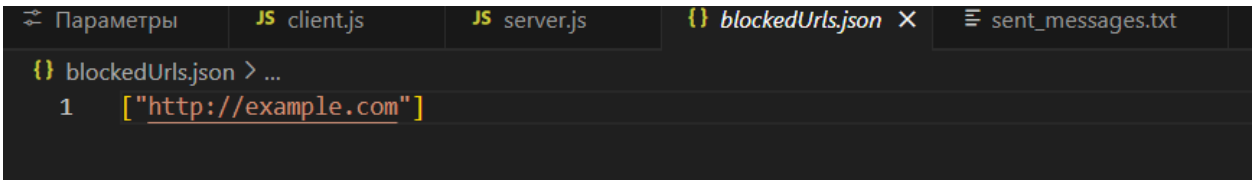


Рис. 1 14. Скриншот запуск сервера(blockedUrls.json).

Далі, згідно поставленому завданню виконаємо:

1. Аналіз вхідного мережевого трафіку на наявність інтернет-загроз (вірусів, хробаків, троянів тощо) методом сигнатурного аналізу

Реалізація:

У моєму коді це представлено простою перевіркою наявності URL у списку заблокованих.

Справжній сигнатурний аналіз вимагає інтеграції з антивірусними бібліотеками або API для перевірки вмісту трафіку на наявність загроз.

Код:

```
const analyzeTraffic = (req) => {
  const parsedUrl = url.parse(req.url, true);
  if (blockedUrls.includes(parsedUrl.hostname)) {
    return true;
  }
  return false;
};
```

Ця функція перевіряє, чи запитаний URL знаходиться у списку заблокованих.

2. Блокування доступу до веб-ресурсів з можливістю ручного налаштування списку таких веб-ресурсів

Реалізація:

Сервер дозволяє додавати URL до списку заблокованих через запити до /block.

Код:

```
else if (req.method === 'POST' && req.url.startsWith('/block')) {
  let urlToBlock = '';
  req.on('data', chunk => {
    urlToBlock += chunk.toString();
  });
```

```
  req.on('end', () => {
    blockedUrls.push(urlToBlock);
    saveBlockedUrls();
    res.statusCode = 200;
    res.end(`URL заблоковано: ${urlToBlock}`);
  });
}
```

Цей блок коду обробляє запити для блокування URL і додає їх до списку заблокованих.

3. Здійснюваємо постійний моніторинг веб-трафіку та вести записи по основним мережевим характеристикам кожного вузла у окремий журнал (кількість вузлів, кількість вхідного/вихідного мережевого трафіку по кожному вузлу, виявлені інтернет-загрози)

Реалізація:

Сервер зберігає інформацію про трафік і зберігає її у файл trafficLog.json.

Код:

```
const updateTrafficLog = (req, res) => {
  const clientIp = req.connection.remoteAddress;
  if (!trafficLog[clientIp]) {
    trafficLog[clientIp] = { in: 0, out: 0, threats: [] };
  }
  trafficLog[clientIp].in += parseInt(req.headers['content-length'] || '0');
  trafficLog[clientIp].out += Buffer.byteLength(res.body || '');
};
const saveTrafficLog = () => {
  try {
    fs.writeFileSync('trafficLog.json', JSON.stringify(trafficLog, null, 2), 'utf8');
  } catch (err)
{
    console.error('Помилка збереження журналу трафіку:', err);
  }
};
```

- updateTrafficLog оновлює журнал трафіку для кожного клієнта.
- saveTrafficLog зберігає оновлений журнал у файл.

Додаткові функції для забезпечення безпеки

Перевірка розміру тіла запиту:

```
if (body.length > MAX_BODY_SIZE) {
  res.statusCode = 413;
  res.end('Request Entity Too Large');
  req.connection.destroy();
}
```

Цей блок коду гарантує, що розмір тіла запиту не перевищує допустимий ліміт, захищаючи сервер від атаки типу DoS.

Обробка помилок:

Код:

```
req.on('error', err => {
  console.error(err);
  res.statusCode = 500;
  res.end('Internal Server Error');
});
```

Цей блок коду обробляє помилки, які можуть виникнути під час обробки запиту, забезпечуючи стабільну роботу сервера.

Підведемо підсумок виконаного завдання:

В ході виконання поставленого завдання в коді були виконано:

Обробка помилок: код наразі не обробляє можливі помилки, які можуть виникнути під час відправки запиту або отримання відповіді. Мною було додано обробник подій error до об’єкта req для обробки таких ситуацій.

Перевірка статусу відповіді: Мною було додано перевірку статусу відповіді, щоб переконатися, що запит було успішно оброблено сервером.

Сервер виконує наступні функції відповідно до вимог:

1. Аналізує трафік на наявність заблокованих URL у функції analyzeTraffic.

2. Блокує доступ до веб-ресурсів через обробку запитів до /block.

3. Моніторить веб-трафік та веде журнал у функціях updateTrafficLog та saveTrafficLog.

4.Забезпечує безпеку:

6. Перевірка розміру тіла запиту.

7. Обробка помилок.

Це забезпечує відповідність серверного коду вашим вимогам до системи керування доступом до розподілених веб-ресурсів.

Процес запуску сервера пройшов успішно, і він ефективно обробляє POST-запити від клієнта. Підтверджено, що сервер коректно відповідає на тестове повідомлення від клієнта, конвертуючи його в верхній регістр. Це узгоджується з очікуваною функціональністю, визначеною у вимогах.

2.ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

В даному дипломному проекті була спроектована система керування доступом до розподілених веб-ресурсів.

Ефективність кожного програмного продукту визначається його якістю та ефективністю процесу розробки. Якість ПП визначається наступними складовими: з точки зору користувача; з позиції використання ресурсів; виконання вимог до програмного забезпечення.

Оцінка якості програмного продукту з точки зору користувача визначається необхідним на стадії функціонування розміром оперативної пам'яті, витратами машинного часу, пропускною спроможністю каналів передачі даних. Оцінка якості програмного продукту включає визначення трудомісткості і вартості його створення.

2.2. Визначення трудомісткості розробки програмного забезпечення.

Тривалість розробки програмного продукту залежить від його обсягу, трудомісткості розробки, кваліфікації виконавців, а також планових термінів, визначених умовами ринку. Методом структурної аналогії по відповідних каталогах аналогів програмного забезпечення визначається обсяг програмних засобів, у тисячах умовних машинних команд програми аналога.

Таблиця 2.1.- Каталог аналогів

Найменування ПП	Обсяг функції ПП – V_o, усл. машинних командах.
1. ПП автоматизації засобів по каталогу	680 – 7000
2. ПП автоматизованих розрахунків	1300 – 8600
3. ПП імітаційного моделювання	7800 – 8800

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких,у більшому або меншому ступені, виконує розроблений програмний продукт. Для в умовних машинних командах, трудомісткості визначати на основі табл.2.2 нашого варіанта виділено сірим кольором. Вибравши аналог ПП, що містить Vo

Таблиця.2.2

Обсяг ПП, тис.умов.машинних команд	Норма часу, люд/год
1.00	229
2.00	244
3.00	262

На підставі отриманого значення, по довіднику, визначається укрупнена норма часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розробки ПП, тобто в умовах комп'ютера, Кк=0,7÷0,8): Тар = 229 х 0,7 = 160,3 (люд/годин). Трудомісткість програмного продукту визначається по кожному етапу розробки окремо на підставі трудомісткості аналога з урахуванням складності розробки, ступеня новизни і ступеня використання в розробці стандартних модулів на підставі формул:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \quad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \quad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \quad (2.3)$$

Для розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розробки (див. табл. 2.3.);

Кн – поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.4.);

Кт – поправочний коефіцієнт, що враховує ступінь використання в розробці типових програм (див. табл. 2.5.).

Таблиця 2.3.-Значення питомих коефіцієнтів трудомісткості стадії в загальній трудомісткості розробки ПП.

Код стадії	Ступінь новизни		
	А	Б	В
ТЗ (L_1)	0,15	0,12	0,12
ТП (L_2)	0,16	0,15	0,11
РП (L_3)	0,55	0,58	0,61

Для нашого варіанта виділено сірим кольором.

Таблиця 2.4. Значення поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	Ступінь новизни	Значення $К_н$
А	Принципово нові ПП	1,75 – 1,2
Б	ПО – розвиток визначеного параметричного ряду	1,0 – 0,8
В	ПО маючий аналог	0,7

Для нашого варіанта виділено сірим кольором.

Таблиця 2.5. Значення коефіцієнта ступеня використання в розробці типових програм

Ступінь охоплення реалізованих функцій розроблювального ПО типовими програмами, %	Значення $К_т$
60 і вище	0,6
40-60	0,7
20-40	0,8
До 20	0,9

Для нашого варіанта виділено сірим кольором.

Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$Т_{тз}=Та*L_1*К_н=160,3 *0,12*0,8= 15,38 \text{ (люд/годин)} \quad (2.1)$$

Трудомісткість розробки технічного проекту

$$T_{тп}=Ta*L_2*K_н=160,3\ *0,11*0,8 = 14,11 \text{ (люд/годин)} \quad (2.2)$$

Трудомісткість розробки робочого проекту

$$T_{рп}=Ta*L_3*K_н*K_т=160,3\ *0,61*0,8*0,8= 62,58 \text{ (люд/годин)} \quad (2.3)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап: технічне завдання $N_{тз}$=3 (стр), розробка ТП $N_{тп}$=10(стр), розробка робочого проекту $N_{рп}$=14 (стр), пояснювальна записка відповідно $N_{пз}$=36 (стр)

Розрахунок зведений у таблицю 2.6.

Таблиця 2.6.- Розрахунок трудомісткості ПП

Найменування етапів	Розрахунок, годин.		
1.ТЗ	$T_{Pтз}$=15,38	$T_{кк}=0,7*N_{тз}$= 0,7*3=2,1	$T_{нк}=0,15*N_{тз}=0,15*3=0,45$
2.Розробка ТП	$T_{Pтп}$=14,11	$T_{кк}=0,7*N_{тп}=0,7*10=7,0$	$T_{нк}=0,15*N_{тп}=0,15*10=1.5$
3.Розробка РП	$T_{Pрп}$= 62,58	$T_{кк}=0,7*N_{рп}=0,7*14=9,8$	$T_{нк}=0,15*N_{рп}=0,15*14=2,1$
4.Розробка ПЗ	$T_{пз}=1,5**N_{пз}$= 1,5*36 =54	$T_{кк}=0,7*N_{тз}=0,7*36=25,2$	$T_{нк}=0,15*N_{пз}=0,15*36 =5,4$
Усього, в т.ч.:	199,7		
- на розробку	$\Sigma T_р$=146.1		
- контроль керівника		$\Sigma T_{кк}$=44,1	
- нормоконтроль			$\Sigma T_{нк}$=9,5

2.3. Розрахунок ціни програмного продукту.

У цьому розділі для визначення ціни розраховуємо основну заробітну плату виконавців, матеріальні витрати, вартість машино – години і витрати на розробку ПО. Розрахунок основної заробітної плати виконавців приведений у таблиці 6.7. Відповідно до статті 8 «Закону про Державний бюджет України на 2024» встановлено мінімальну заробітну плату у місячному розмірі з 1 квітня 2024 року - 8000 гривень; мінімальну погодинну тарифну ставку – 46.00 грн.

Таблиця 2.7. - Розрахунок основної заробітної плати виконавців.

Найменування робіт	Трудомісткість робіт, години	Погодинна тарифна ставка, грн.	Розрахунок, грн.
1.Розробка ПП	146,1	46,00	6720,60
2.Контроль керівника	44,1	70,00	3087,00
3.Нормоконтроль	9,5	70,00	665,00
Усього	-	-	ΣЗо= 10472,60

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок зведемо в таблицю 2.8

Таблиця 2.8.- Розрахунок матеріальних витрат на розробку ПП

Найменування матеріальних витрат	Тип, модель	Кількість	Ціна одиниці, грн.	Вартість, грн.
Папір	Лист А4	65	4.0	260,0
Разом	-	-	-	$В_{м\,і}$ =260,0
Транспортно – заготівельні Витрати (10%)				$Втр\_з = 0{,}1 \times Вм1$ = 0,1*260=26,00
Усього				$В_м = В_{мі} + В_{тр\_з=}$ 286.00

На підставі отриманих даних по окремих статтях витрат складена калькуляція планової собівартості в цілому ПП за формою, приведеною в таблиці 2.9.

Таблиця 2.9. - Розрахунок статей витрат планової собівартості

Стаття витрат	Значення, грн.	Формула розрахунку
1. Матеріали	286.00	$В_м$ (див. табл. 2.8)
2. Основна заробітна плата	10472,60	$З_о$ (див. табл. 2.7.)
3.Додаткова заробітна плата	1047,26	$Зд = 0{,}1 \times Зо$ = 10472,60*0,1
4.Відрахування до єдиного фонду соціального внеску	2534,36	$Вє.с.в. = 0{,}22 \times (Зо + Зд)$ =0,22* (10472,60+1047,26)
5. Накладні витрати	4188,80	$Внак. = 0{,}4 \times Зо$ =0,4 * 10472,60
6. Повна собівартість	18529,02	$С_{пов}$= $В_м$ +$З_о$ + $З_д$ +Вє.с.в. + Внак.= 286.00+10472,60+1047,26+2534,36+4188,80

Розмір прибутку, що включається в ціну, визначаємо по наступній формулі:

$$П=(С_п*Р)/100=(18529,02*10)/100=1852,90 \text{ грн} \quad (2.4)$$

Де р – плановий рівень рентабельності (10-15%).

Оптова ціна (кошторисна вартість) визначається по формулі:

$$Ц_о = С_п + П = 18529,02+1852,90= 20381,92 \text{ грн;} \quad (2.5)$$

Виходячи з отриманих даних, ціна реалізації розробленого програмного продукту на основі наступної формули, становитиме:

$$Ц_р = Ц_о + ПДВ = 20381,92 + 20381,92 * 0.2 = 24458,30 \text{ грн;} \quad (2.6)$$

3. РОЗДІЛ ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

3.1 Аналіз та безпека умов праці працівника на робочому місці

Охорона праці на виробництві завжди була надзвичайно важливою, тому завдяки дотриманню рекомендацій з охорони праці персонал підприємства створює алгоритм виконання робочих завдань з чітким дотриманням цих рекомендацій. Основна мета охорони праці полягає у створенні та проведенні заходів, спрямованих на захист життя, працездатності та здоров'я працівників під час трудової діяльності.

3.2 Розробка заходів з охорони праці

При роботі з комп'ютером, як і в багатьох інших галузях, слід враховувати нормативи освітлення, температури, відносної вологості та сили вібрації. Найважливішим при роботі в приміщенні з комп'ютерами є дотримання правил пожежної безпеки, рівня звукового шуму та характеристик електромагнітних, ультрафіолетових і інфрачервоних полів.

Під час будь-якої роботи за комп'ютером працівник може зазнавати дії небезпечних факторів виробничого середовища, зокрема фізичних та психофізіологічних небезпечних і шкідливих виробничих факторів.

Фізичні небезпечні фактори

Найпоширеніші фізичні небезпечні фактори при роботі з комп'ютерами включають підвищену температуру повітря робочої зони, підвищений рівень шуму та знижену вологість повітря. Робота комп'ютерів підвищує температуру і знижує вологість повітря в приміщеннях.

Крім того, комп'ютери випромінюють електростатичні та електромагнітні поля у діапазоні від 5 Гц до 2 кГц та від 2 до 400 кГц, що спричиняє підвищений рівень електромагнітного випромінювання та статичної електрики.

Недостатня кількість природного освітлення часто компенсується штучним, яке не завжди налаштоване належним чином, що може призводити до недостатньої або надмірної яскравості.

Психофізіологічні небезпечні фактори

Психофізіологічні небезпечні фактори поділяються на фізичні та нервово-психічні перевантаження, з яких при роботі з комп'ютером найбільш поширеними є нервово-психічні перевантаження. Програмісти часто зазнають перевантаження аналізаторів, монотонність праці та інколи розмовне перевантаження, коли потрібно складати технічне завдання разом із клієнтом.

Виробниче освітлення

Штучне освітлення в приміщеннях з робочими місцями, обладнаними ВДТ, має здійснюватися системою загального рівномірного освітлення. У приміщеннях, де переважно працюють з документами, можна застосовувати комбіноване освітлення (загальне та місцеве освітлення).

Мікроклімат

У приміщеннях з великою кількістю комп'ютерів температура влітку може перевищувати 35°С, що негативно впливає на здоров'я. У таких приміщеннях повітря повинно охолоджуватися, а вологість регулюватися спеціальним обладнанням. Відповідно до норм ДСН 3.3.6.042-99, температура повітря в офісі повинна бути 22-25°С, вологість 40-60%, швидкість руху повітря не більше 0,1 м/с. У разі перевищення цих норм, робочий день працівника повинен бути скорочений на 10%.

3.3 Організація робочого місця користувача ПК

Охорона праці означає систему заходів, спрямованих на збереження життя, здоров'я та працездатності людини під час виконання роботи. Покращення методів забезпечення безпеки праці є важливим шляхом підвищення ефективності виробництва, оскільки травматизм є значною причиною

непродуктивних втрат робочої години. Безпека праці також впливає на продуктивність, оскільки високу продуктивність може бути досягнуто лише в умовах безпечного працюючого середовища. В розділі охорони праці дипломного проекту розглядається питання розробки мобільного додатку для виступів. Тому об’єктом дослідження беремо безпеку праці на робочому місці програміста.

При кольоровому оформленні виробничих і допоміжних приміщень необхідно враховувати орієнтацію їхніх вікон стосовно частин світу і використовувати гармонійне сполучення кольорів. Для стін і робочих поверхонь використовують мало насичені (основні) кольори, для невеликих помешкань або ділянок, що рідко потрапляють у поле зору працюючих, а також для створення контрастності – кольори середньої насиченості (допоміжні), для маленьких по площі поверхонь – насичені (акценти) – як функціональне фарбування. Стелі у всіх приміщеннях повинні бути білими. Поверхні устаткування в приміщеннях повинні бути матовими або напівматовий, для виключення випадку відблисків світла в очі працюючого, а стіни бути пофарбованими фарбами пастельних тонів

Конструкція робочого місця користувача ПК повинна відповідати антропометричним, фізіологічним і психологічним вимогам.

Робочі меблі повинні забезпечувати можливість індивідуального регулювання відповідно до зросту працівника. Дисплей слід розташовувати так, щоб його верхній край був на рівні очей, на відстані 60-90 см. Частота мерехтіння екрана повинна бути не меншою 70 Гц.

3.4 Пожежна безпека

Забезпечення пожежної безпеки на об’єкті є важливою частиною роботи зі створення безпечних умов праці. Проходи до аварійних виходів повинні бути вільні, шириною не менше 1 метра. Використовуйте сміттєзбірники для зберігання горючих відходів. Електроприлади повинні використовуватися за призначенням, а у разі їх пошкодження слід вимкнути живлення.

Первинні засоби пожежогасіння, такі як пожежні кран-комплекти, вогнегасники та пожежний інвентар, розміщуються на пожежних щитах і фарбуються у червоний колір. На кожному поверсі адміністративної будівлі повинно бути не менше двох вогнегасників. Забороняється палити на підприємстві, крім спеціально відведених місць, та зберігати легкозаймисті матеріали біля електрощитів та приладів опалення.

Для запобігання розповсюдження пожежі встановлюються протипожежні системи, які складаються з датчиків, звукових сповіщувачів, аварійних кнопок та приймально-контрольної панелі. Підприємство має укласти договір на обслуговування протипожежної системи з ліцензованою фірмою.

У разі виникнення пожежі:

1. Терміново повідомити пожежну охорону по телефону 101.
2. Організувати евакуацію людей та матеріальних цінностей.
3. Повідомити адміністрацію та чергового.
4. Вимкнути струмоприймачі та вентиляцію.
5. Розпочати гасіння пожежі наявними первинними засобами пожежогасіння.
6. Організувати зустріч підрозділів пожежної охорони та надати їм допомогу.

Пожежна безпека може бути забезпечена заходами пожежної профілактики і активного пожежного захисту. Пожежна профілактика включає комплекс заходів, спрямованих на попередження пожежі або зменшення його наслідків.

Пожежна безпека при роботі з комп'ютером передбачає обережність при обслуговуванні, ремонтних та профілактичних роботах та виконування всіх інструкцій працівником щодо пожежної безпеки. Заземлені конструкції, що знаходяться в приміщеннях, де розміщені робочі місця (батареї опалення, водопровідні труби, кабелі із заземленим відкритим екраном), мають бути надійно захищені діелектричними щитками або сітками з метою недопущення потрапляння працівника під напругу. Приміщення, де розміщені робочі місця, мають бути оснащені системою автоматичної пожежної сигналізації і вогнегасниками відповідно до вимог чинного законодавства України. Проходи до засобів пожежогасіння мають бути вільними

Всі приміщення повинні бути забезпечені первинними засобами пожежогасіння: пожежним водопостачанням (пожежні крани ПК), пожежні щити з набором пожежного інструменту, вуглекислотними або порошковими вогнегасниками. У випадку виникнення пожежі необхідно відключити електроживлення, викликати по телефону 101 пожежну команду, евакуювати людей із приміщення відповідно до плану евакуації і приступити до ліквідації пожежі.

Забезпечення безпечних та здорових умов праці значною мірою залежить від правильної оцінки небезпечних та шкідливих факторів, які можуть впливати на працюючу людину. Різні фактори, такі як виробниче середовище, фізичні та розумові навантаження, стрес і т.д., можуть спричиняти складні зміни у фізичному стані людини. Оператори та програмісти можуть стикатися з різними фізично небезпечними та шкідливими факторами, такими як підвищений рівень шуму, підвищена температура, недостатнє освітлення, електричний струм, статична електрика та інші. На робочому місці програміста необхідно створити умови для безпечної та продуктивної праці, що включає в себе не лише усунення фізичних та ергономічних стресів, але й забезпечення відповідної психологічної атмосфери та можливості регулярних перерв для відпочинку та відновлення. Вимоги Державних санітарних правил і норм 3.3.2.007-98 визначають об'ємно-планувальні рішення будівель та приміщень для роботи з ВДТ. У приміщеннях з робочими місцями важливо забезпечити правильне освітлення, яке сприяє комфортній та продуктивній праці. Зазвичай використовується система загального рівномірного освітлення, що розподіляє світло по всьому приміщенню рівномірно. Якщо діяльність передбачає переважну роботу з документами або інші види робіт, які потребують більшого концентрованого освітлення, можна використовувати систему комбінованого освітлення, яка поєднує загальне освітлення з додатковими світильниками місцевого освітлення, щоб забезпечити більшу якість освітлення на конкретному робочому місці.

ВИСНОВКИ

У результаті виконання дипломного проєкту було проведено дослідження механізмів доступу до розподілених веб-ресурсів та розроблено систему керування доступом до цих ресурсів. Було здійснено аналіз технологій та процесів, пов'язаних із розподіленими веб-ресурсами, що включає поняття "веб-ресурсу", механізми обміну ними, додаткові механізми шифрування та стиснення даних під час обміну веб-ресурсами. Також було досліджено механізми керування доступом до розподілених веб-ресурсів, зокрема мету керування ними, дослідження керівних впливів на процес доступу до веб-ресурсів, аналіз сучасних інтернет-загроз, проблеми та мету захисту комп’ютерних мереж, методи виявлення та інструментальні засоби для виявлення інтернет-загроз у комп’ютерних мережах.

Крім того, були досліджені методи та засоби керування доступом до розподілених веб-ресурсів: описані технології проксі-серверів та веб-фільтрів, проведений порівняльний аналіз проксі-серверів, досліджені протоколи проксі-серверів, розкрито вплив технологій шифрування та стиснення даних на процес аналізу веб-трафіку.

Було розроблено систему керування доступом до розподілених веб-ресурсів: сформовані функціональні вимоги, обґрунтовано використання відповідних технологій, здійснено проектування та програмування системи, описано процес її розгортання та представлено результати розробки. Сервер виконує наступні функції відповідно до вимог:

Аналізує трафік на наявність заблокованих URL у функції analyzeTraffic.

Блокує доступ до веб-ресурсів через обробку запитів до /block.

Моніторить веб-трафік та веде журнал у функціях updateTrafficLog та saveTrafficLog.

Забезпечує безпеку:

Перевірка розміру тіла запиту.

Обробка помилок.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. RFC 1945. Hypertext Transfer Protocol – HTTP/1.0 [Інтернет-ресурс]. – Режимдоступу: https://tools.ietf.org/html/rfc1945 вільний.
2. RFC 7231 - Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content

[Інтернет-ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc7231 вільний.

3. RFC 7540 - Hypertext Transfer Protocol Version 2 (HTTP/2) [Інтернет-ресурс].
4. Denning D. An intrusion-detection model. In Proc. // IEEE Symposium on Securityand Privacy. - 2019. - Vol. 13, No 2. – P. 222-232.
5. Sheyner O. Scenario Graphs and Attack Graphs. / PhD thesis, SCS, Pittsburgh: Carnegie Mellon University. – 2024. – P. 141.
6. Edward G. Intrusion Detection. 1st ed., Intrusion.Net Books / New Jersey: Sparta. – 2019. – P. 218.
7. Eckmann S.T., Vigna G., Kemmerer R. A. STATL: An Attack Language for State- based Intrusion Detection // Dept. of Computer Science, University of California, Santa Barbara. – 2000. – Vol. 12, No 2. – P. 71-103.
8. Vigna G., Kemmerer R. A. NetSTAT: A Network-based Intrusion Detection Approach // Proceedings of the 14th Annual Computer Security Application Conference. – 2000. – P. 73-81.
10. Гамаюнов Д.Ю., Смелянський Р.Л. Модель поведінки мереживих об’єктів в розподеленних обчислювальних системах / Д.Ю. Гамаюнов, Р.Л. Смелянський.
11. Шелухін О.И. Виявлення вторгнень у комп'ютерні мережі (мережеві аномалії)/ О.И. Шелухін, Д.Ж. Сакалема, А.С. Филинова. – М.: Гаряча лінія – Телеком, 2013. – 221

ДОДАТОК А

СЛАЙДИ МУЛЬТИМЕДІЙНОЇ ПРЕЗЕНТАЦІЇ

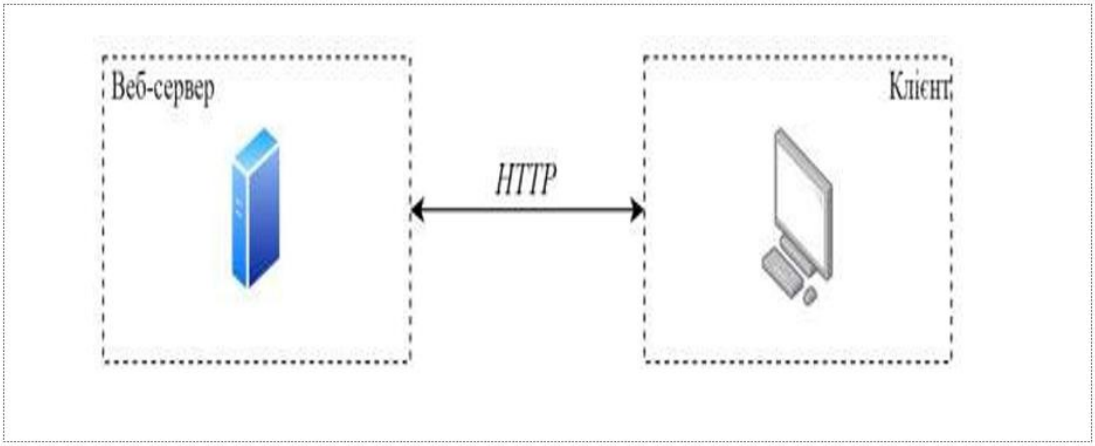
ДИПЛОМНИЙ ПРОЕКТ НА ТЕМУ:

Проектування системи керування доступом до розподілених веб-ресурсів

Виконав студент гр.4КБ-01: Шмиголь О.І.

СЛАЙД 1

Схема обміну веб-ресурсами



СЛАЙД 2

Переповнення буферу оперативної пам’яті веб-серверу

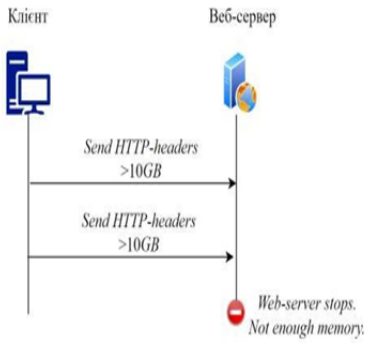
1. Content-Type: Цей заголовок вказує клієнту на тип медіаресурсу, що повертається.

2. Content-Encoding: Цей заголовок вказує на метод кодування, застосований до тіла повідомлення для передачі через мережу.

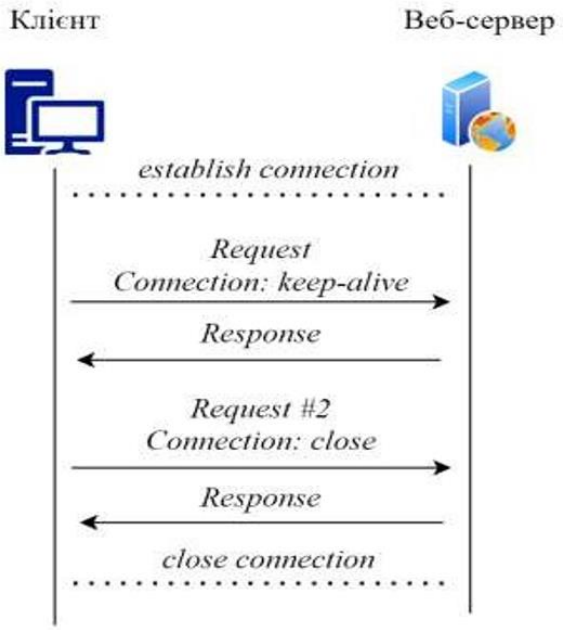
3.Content-Length: Цей заголовок вказує на довжину тіла повідомлення в байтах.

4.Cache-Control: Цей заголовок контролює кешування клієнта або проксі-сервера.

5.Connection: Цей заголовок вказує, чи слід підтримувати з'єднання після завершення поточного запиту.

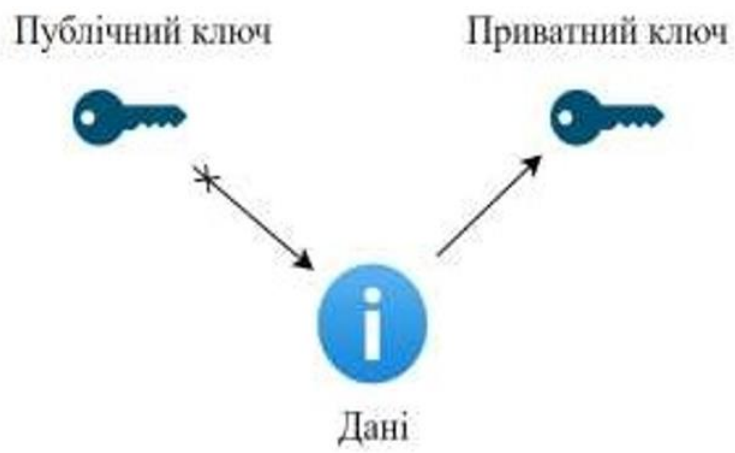


СЛАЙД 3



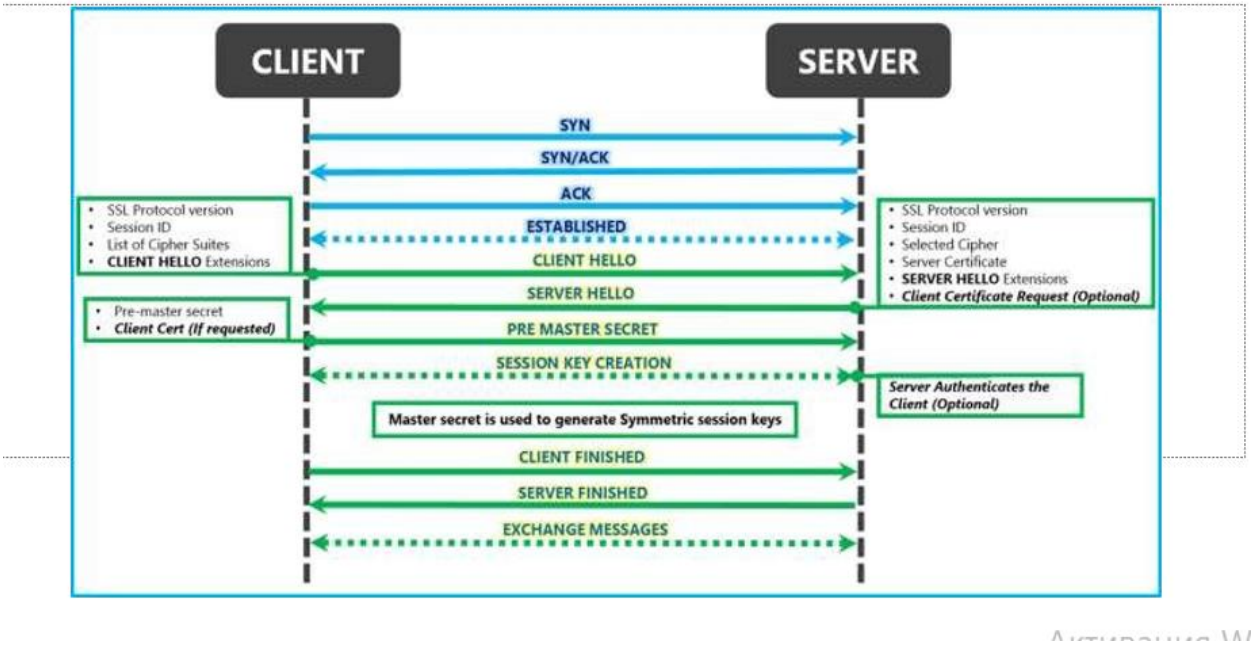
СЛАЙД 4

Шифрування даних публічним ключом



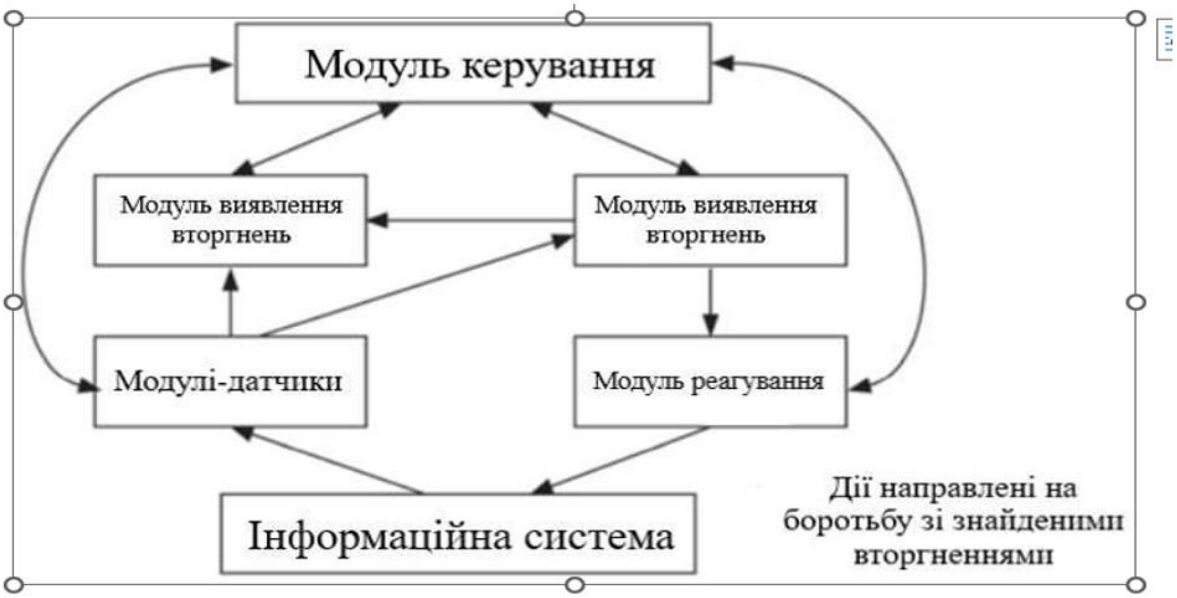
СЛАЙД 5

Процес *handshake*



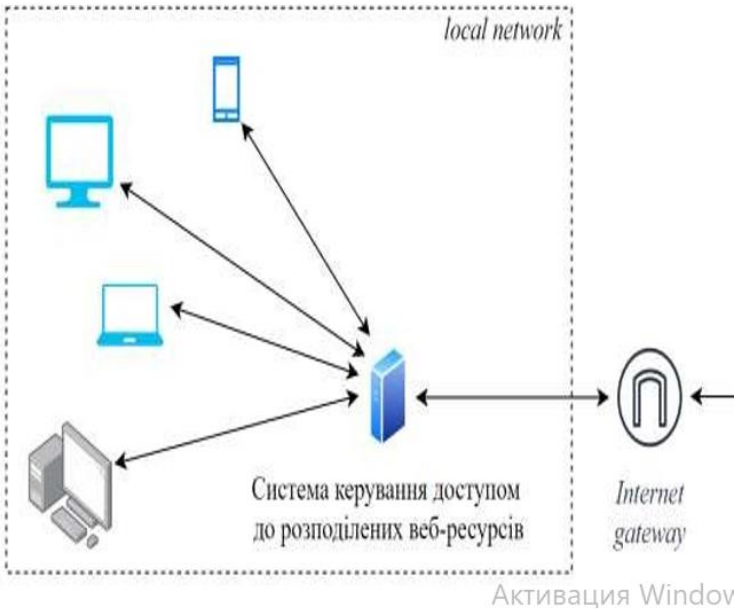
СЛАЙД 6

Приклад моделі інформаційної системи



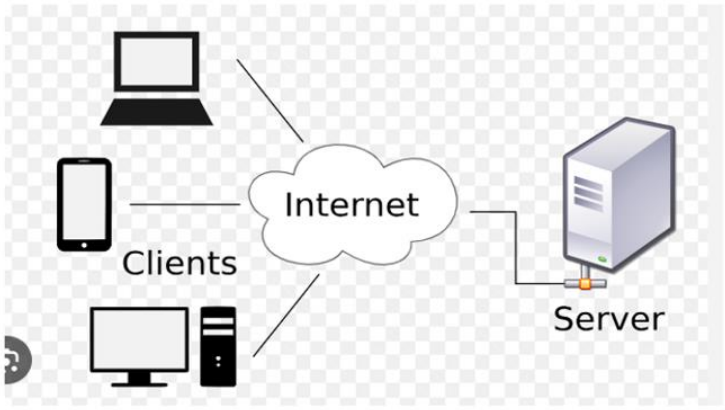
СЛАЙД 7

Приклад схеми топології системи



СЛАЙД 8

Приклад архітектури клієнт-сервер



СЛАЙД 9

Практична реалізація системи керування доступом до розподілених веб-ресурсів

• При розробці системи керування доступом до розподілених веб-ресурсів,згідно поставленому завданню необхідно врахувати та вирішити наступні задачі:

1.Передача даних.

2.Аналіз та обробка даних.

4. Зберігання даних.

• Відповідно до вищеописаних задач розроблювана система повинна реалізовувати методи та механізми, що:

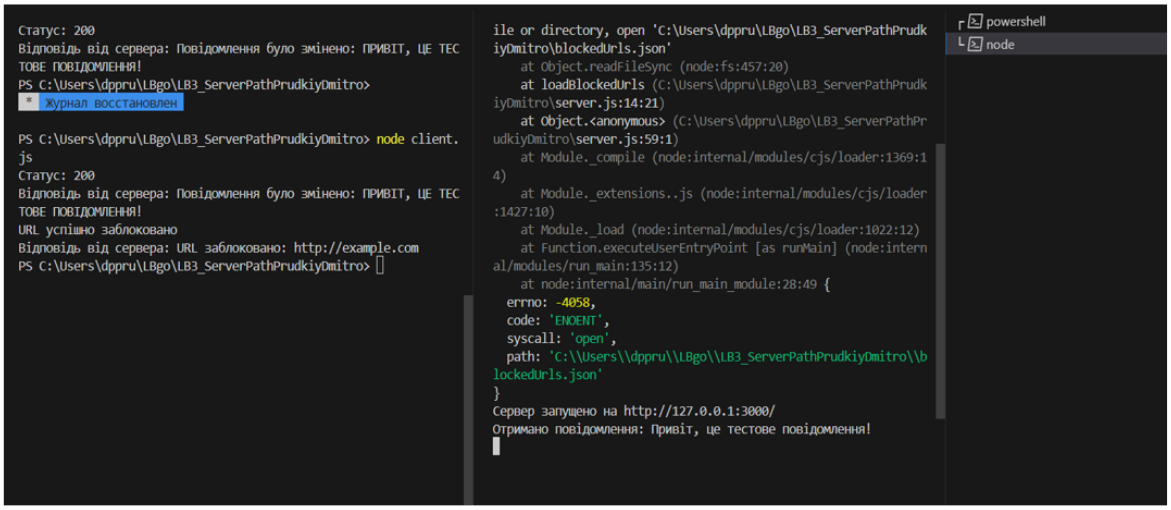
1.Реалізують механізми передачі даних.

2.Реалізують методи аналізу даних з метою подальшого прийняття керуючих рішень.

3Зберігають результати перевірок та ведуть журнал дій.

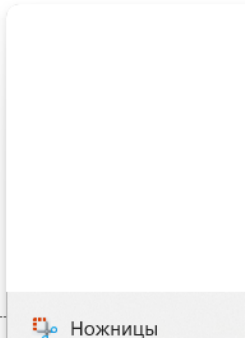
СЛАЙД 10

Скриншот запуску сервера



СЛАЙД 11

ДЯКУЮ ЗА УВАГУ !!!



СЛАЙД 12

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Шмиголя Олександра Івновича

(прізвище, ім'я та по батькові)

Спеціальність: 123 "Комп'ютерна інженерія"

Освітня програма: «Безпека комп'ютерних систем і мереж»

Тема дипломного проекту: Проектування системи керування доступом до розподілених веб-ресурсів

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка містить 63 сторінки. У пояснювальній записці виконано опис етапів розробки системи керування доступом до розподілених а також його програмне забезпечення. Графічна частина складається з 12 слайдів мультимедійної презентації, які також містять креслення, передбачені технічним завданням. Якість виконання пояснювальної записки та графічної частини добра, розробку виконано в повному обсязі.

б) самостійність роботи над проектом: Протягом всього строку дипломного проектування та переддипломної практики здобувач освіти Шмиголь О.І. поступово та послідовно виконував всі етапи розробки. Всі роботи здобувач освіти виконував самостійно, з оглядом на рекомендації керівника

в) теоретична підготовка випускника (випускниці): Здобувач освіти Шмиголь О.І. під час роботи над дипломним проектом вивчив достатню кількість літературних джерел та матеріалів за даною тематикою.

Вважаю, що теоретична підготовка дипломника добра і він готовий до захисту дипломного проекту

г) вміння розв'язувати виробничі та конструкторські питання

Під час дипломного проектування здобувач освіти Шмиголь О.І. мав змогу самостійно приймати окремі рішення з реалізації системи керування доступом до розподілених веб-ресурсів та показав вміння організовано працювати над поставленим завданням, розробляти програмні коди за допомогою сучасних комп'ютерних програмних засобів та мов програмування.

Оцінка розрахункової частини Добре

Оцінка графічної частини Добре

Загальна оцінка Добре

Прізвище, ім'я, по батькові керівника дипломного проекту

к.т.н. Кірєєв Ігор Анатолійович

Місце роботи і посада керівника дипломного проекту

Державний університет інтелектуальних технологій, доцент кафедри інформаційної безпеки та передачі даних

Підпис

«10» 06 2023 р.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект (роботу) здобувача (здобувачки) освіти

відділення комп'ютерних систем

Шмиголя Олександра Івановича

(прізвище, ім`я та по батькові)

Спеціальність **123 Комп'ютерна інженерія**

Освітня програма **«Безпека комп'ютерних систем та мереж»**

Керівник дипломного проекту (роботи) **Кірєєв І.А.**

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) **Проектування системи керування доступом до розподілених веб-ресурсів**

Обсяг розрахунково-пояснювальної записки 60 сторінок

Обсяг графічної (презентаційної) частини 12 аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту (роботи) завданню

Представлений на рецензію робота відповідає затверджений темі та виконаний відповідно технічному завданню. Дипломний проект є актуальним з погляду останніх рекомендацій проектування доступом до розподілених веб-ресурсів

б) характеристика виконання кожного розділу дипломного проекту (роботи)

Пояснювальна записка складається з технологічної частини, розробки структури системи керування доступом до розподілених веб-ресурсів, опису і експлуатації засобів, економічної частини, розділу охорони праці та додатку. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано. Розділ охорони праці містить загальну інформацію та вимоги до техніки безпеки оператора ЕОТ. Економічна частина проекту містить розрахунок затрат на виконання та реалізацію проекту

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту (роботи)

Графічна частина складається з 12 слайдів мультимедійної презентації, виконаної у програмному продукті MS PowerPoint, які містять ілюстративні схеми, скріншоти роботи програмних застосунків, передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання графічної частини проекту та пояснювальної записки висока, розробку виконано у повному обсязі

г) перелік позитивних якостей дипломного проекту (роботи)

У роботі досить обґрунтовано досліджено основні характеристики принципів сучасних систем керування доступом до розподілених веб-ресурсів.

д) основні недоліки дипломного проекту (роботи)

1. Занадто великий обсяг аналітичного матеріалу
2. Немає посилань на використану літературу

Оцінка розрахункової частини Добре

Оцінка графічної частини Добре

Загальна оцінка Добре

Прізвище, ім'я, по батькові рецензента к.т.н. Селіванова Алла Віталіївна

Місце роботи і посада рецензента Одеський національний технологічний університет, декан факультету комп'ютерної інженерії, програмування та кіберзахисту

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ ФАКУЛЬТЕТ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ, ПРОГРАМУВАННЯ ТА КІБЕРЗАХИСТУ

Підпис:

«10» 06 2024 р.

UNICHECK
by Turnitin

Ім'я користувача:
Катерина Григоріївна Краснокутська

ID перевірки:
1016316130

Дата перевірки:
03.06.2024 19:44:54 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
03.06.2024 19:49:34 EEST

ID користувача:
100011688

Назва документа: 4КБ-01_Шмиголь

Кількість сторінок: 51 **Кількість слів:** 8188 **Кількість символів:** 62792 **Розмір файлу:** 462.17 KB **ID файлу:** 1016113699

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

26.8%
Схожість

Найбільша схожість: 15% з Інтернет-джерелом (https://dspace.nau.edu.ua/bitstream/NAU/47219/1/%d0%a4%d0%9a%d...

26.8% Джерела з Інтернету | **397** ..... **Сторінка** 53

Не знайдено джерел з Бібліотеки

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | **24**

Підозріле форматування | **9 сторінок**

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Шмиголь Олександр Іванович
здобувач освіти гр. 4КБ-01, та

Кірєєв Ігор Анатолійович,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до випускної кваліфікаційної роботи фахового молодшого бакалавра на тему:

«Проектування системи керування доступом до розподілених веб-ресурсів» (автор роботи – Шмиголь О.І., керівник роботи – Кірєєв І.А.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2024 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи, і даємо згоду на обробку персональних даних.

Виконавець _____ / Шмиголь О.І./

Керівник _____ / Кірєєв І.А./

«10» 06 2024 р.