Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____комп'ютерних наук_____
(повна назва)

Кафедра _____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

_____другий (магістерський)_____
(рівень вищої освіти)

ДОСЛІДЖЕННЯ АЛГОРІТМІЧНИХ ЗАСОБІВ ЗАБЕЗПЕЧЕННЯ МЕРЕЖЕВОЇ БЕЗПЕКИ ТА ВЕБ-САЙТІВ
(тема)

Виконав: студент 2 курсу, групи ІПЗм-18-4__
спеціальності _121 – Інженерія програмного забезпечення_____
(код і повна назва спеціальності)
освітньо-наукової програми _Інженерія програмного_забезпечення_ _____
(повна назва освітньої програми)

_____Кочегура Д.В._____
(прізвище, ініціали)

Керівник _____проф. Шостак І.В._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

2020 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук_____

Кафедра програмної інженерії_____

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність _121 – Інженерія програмного забезпечення_____
(код і повна назва)

Освітньо-наукова програма Інженерія програмного забезпечення _____
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

Студентові _Кочегурі Дмитру Володимировичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження алгоритмічних засобів забезпечення мережевої безпеки та веб-сайтів____

затверджена наказом по університету від «__»_____ __._.2020 р № ____

2. Термін подання студентом роботи до екзаменаційної комісії «11» травня 2020 р.

3. Вихідні дані до роботи *Алгоритми обробки даних мережі, алгоритми захисту даних, методи створення між мережевих екранів, використовувати ОС Windows, середовище об'єктно-орієнтованого проектування.*_____
_____
_____

4. Перелік питань, що потрібно опрацювати в роботі _*мета роботи, аналіз проблемної галузі і постановка задачі, методи пошуку корисних даних, опис об'єктних моделей, використовувані. методи та алгоритми, архітектура програмної системи, опис розробленої програмної системи. , результати тестування програмної системи*_____
_____

5. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	проф. Шостак І.В.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Аналіз предметної галузі	25 березня 2020 р.	
2.	Огляд існуючих методів	31 березня 2020 р.	
3.	Методи створення та аналізу енергозберігаючого ПЗ	15 квітня 2020 р.	
4.	Підготовка пояснювальної записки	20 квітня 2020 р.	
5.	Спецчастина	28 квітня 2020 р.	
6.	Підготовка презентації та доповіді	03 травня 2020 р.	
7.	Попередній захист	05 травня 2020 р.	
8.	Нормоконтроль, рецензування	07 травня 2020 р.	
9.	Занесення роботи в електронний архів	08 травня 2020 р.	
10.	Допуск до захисту в зав. кафедри	10 травня 2020 р.	

Дата видачі завдання _ «____»_ _____2020 р.

Студент _____
(підпис)

Керівник роботи _____ __проф. Шостак І.В._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ ABSTRACT

Пояснювальна записка на роботу: 103 с., 8 табл., 35 рис., 3 дод., 31 джерел.

ПРОДУКТИВНІСТЬ, МЕРЕЖЕВІ ЕКРАНИ, ІНФОРМАЦІЙНІ ВЕБ-САЙТИ, ПОЛІТИКА БЕЗПЕКИ, ОПТИМІЗАЦІЯ

Об'єктом дослідження є процес забезпечення мережної безпеки програмними засобами на веб-сайтах.

Метою дослідження є розробка алгоритмів поліпшення продуктивності програмних засобів, зокрема мережевих екранів, що використовуються для захисту даних інформаційних веб-сайтів.

Методи дослідження: методи планування експерименту, статистичний аналіз результатів експерименту.

В результаті отримано емпіричні формули, що характеризують вплив прийнятої політики безпеки на продуктивність, що можуть бути впроваджені в структуру забезпечення безпеки інформаційних веб-сайтів.

PERFORMANCE, INTERNET NETWORKS, ВЕБ-SITES, SECURITY POLICIES, OPTIMIZATION

The object of research of this work is the process of providing network security with software on websites.

The subject of the study is models and methods for providing network security with software on web-sites.

Methods of research: methods of experiment planning, statistical analysis of experimental results.

The results of the thesis can be implemented in the structure of the security of information web-sites.

ЗМІСТ

Вступ ……6

1 Аналіз стану проблеми і постановка завдань дослідження……...…..8

1.1 Аналіз програмних засобів забезпечення мережної безпеки ……..…...8
1.2 Мережеві екрани й виконувані ними завдання ……....10
1.3 Методи та алгоритми керування доступом до інформації …..……13
1.4 Опис алгоритмів роботи користувачів ……..…….19
1.5 Використання програмних екранів …….21
1.6 Мета і завдання дослідження …….....24

2 Опис проведених теоретичних досліджень …….....26

2.1 Опис алгоритмів ідентифікації та вибору рівнів факторів …….....26
2.2 Класифікація міжмережевих екранів …..……...30
2.3 Алгоритми конфігурування міжмережевих екранів ……36
2.4 Аналіз методів проектування ……..41

3 Опис проектування алгоритмів системи …….....45

3.1 Опис структури мережі моделі веб-сайту ……..45
3.2 Опис використовуваних метрик …...……..50

4 Опис розробленого програмного забезпечення.....53

4.1 Опис протоколу взаємодії ……...53
4.2 Загальний алгоритм роботи програми ……55
4.3 Алгоритм роботи клієнтської частини ……...59
4.4 Загальна архітектура програми ……...60

5 Опис можливості використання отриманих результатів……75

Висновки ……...79

Перелік джерел посилання …….....81

Додаток А Програмний код …….....84

Додаток Б Слайди презентації …….....89

Додаток В Апробація результатів роботи…….……102

ВСТУП

У зв'язку зі швидким розвитком мережі Інтернет і пов'язаних з нею сервісів усе більшу актуальність знаходять проблеми забезпечення мережної інформаційної безпеки. Веб-сайти самої різної спрямованості також вимагають уваги в плані безпеки, що підтверджується періодичними атаками з мережі на сайти таких великих компаній як Microsoft і Yahoo. Одним з найпоширеніших типів веб-сайтів є інформаційні веб-портали, які є загальнодоступними сховищами інформації самої різної тематики. Для таких порталів характерним є велика кількість користувачів, і відповідно висока інтенсивність трафіка. При цьому основний засіб захисту – система мережевого екранування може стати основною причиною зменшення швидкості відповіді порталу на запити користувачів.

Через те, що весь трафік проходить через мережевий екран (МЕ) розуміння характеристик його продуктивності є основним елементом прогнозування й забезпечення продуктивності всієї системи. Порівняння характеристик продуктивності МЕ з різною архітектурою, таких як класичні й розподілені екрани презентовано у [3]. Також багато виробників МЕ публікують спеціальні рекомендації із забезпечення високої продуктивності своїх продуктів. Прикладом може служити досить докладне керівництво, випущене компанією Microsoft для свого МЕ ISA Server.

Однак багато компаній-власників інформаційних веб-порталів так чи інакше зустрічаються із цією проблемою, а більшість пропонованих рішень вимагають значних фінансових витрат на поліпшення апаратного забезпечення. У такий спосіб дослідження продуктивності саме програмних МЕ є актуальною й недостатньо розкритою темою. Особливо це стосується ступеня впливу тих або інших факторів на розглянуту проблему й можливості мінімізації їх дії.

Об'єктом дослідження даної роботи є процес забезпечення мережної безпеки програмними засобами на веб-сайтах.

Метою дослідження, проведеного в даній роботі, є розробка рекомендацій з поліпшення продуктивності програмних засобів, зокрема мережевих екранів, використовуваних для захисту даних інформаційних веб-сайтів.

Для досягнення поставленої мети планується розв'язати наступні завдання:

- провести аналіз стану проблеми забезпечення мережної безпеки програмними засобами, зокрема мережевими екранами, на інформаційних веб-сайтах;
- провести планування експерименту по визначенню продуктивності системи мережевих екранів, включених у мережу за типовою схемою захисту даних інформаційних веб-сайтів;
- розробити прототип програмного забезпечення для проведення експерименту.
- одержати емпіричні формули залежності продуктивності досліджуваних мережевих екранів від використовуваних правил безпеки;
- проаналізувати отримані емпіричні формули й розробити рекомендації з підвищення продуктивності мережевих екранів у заданих умовах;

Методи дослідження: методи планування експерименту, статистичний аналіз результатів експерименту.

Результати магістерської роботи можуть бути впроваджені в структуру забезпечення безпеки інформаційних веб-сайтів.

1 АНАЛІЗ СТАНУ ПРОБЛЕМИ І ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Аналіз програмних засобів забезпечення мережної безпеки

Розвиток мережних технологій, швидкий якісний і кількісний ріст мереж по усьому світу, розширення глобальної мережі інтернет роблять усе більш актуальними питання безпечної роботи в мережі. Схоронність і конфіденційність даних, що перебувають на комп'ютерах, підключених до великих мереж, а також безпека загальнодоступних веб-сайтів, зазнає великої небезпеки через реальну можливість атаки з метою крадіжки або знищення інформації. По даним news.ua 63% веб-сайтів мають критичні уразливості, а в 93 випадків з 100 у програмнім забезпеченні (ПЗ) веб-додатки втримуються уразливості середнього ступеня ризику.

Серед найпоширеніших видів вразливостей можна виділити:

- відсутність перевірки параметрів в HTTP-запитах. У результаті, використовуючи особливі параметри, зловмисник може одержати доступ до ресурсів сервера через веб-додаток;
- недотримання політики керування доступом до ресурсів дозволяє зловмисникові використовувати закриті ресурси або одержувати доступ до облікових записів інших користувачів;
- недотримання правил керування обліковими записами й користувацькими сесіями. Ця вразливість зв'язана, насамперед, з відсутністю надійного захисту користувацьких даних і ідентифікаторів сесій, таких як файли cookie – це дозволяє хакерам перехоплювати дані інших користувачів і користуватися системою від їхнього імені [4];
- вразливості, пов'язані з помилками в механізмі Cross-Site Scripting, використовуваному для перенаправлення користувача на інші сайти. У цьому випадку атака може привести до одержання зловмисником доступу до користувацьких даних або злому локального комп'ютера;

- помилки переповнення буфера, наявні в багатьох програмних продуктах (ПП) – від скриптів і драйверів до операційних систем і серверного ПЗ. Відсутність перевірки деяких параметрів може приводити до переповнення буфера, а зловмисник при цьому захоплює керування комп'ютером, Повідомлення про виявлення помилок переповнення з'являються надзвичайно часто;
- відсутність належного контролю над параметрами, переданими комп'ютерами при доступі до зовнішніх ресурсів. Якщо хакер зуміє ввести в ці параметри свої команди, наслідки можуть бути дуже важкими;
- невдале використання криптографії. У Часто інструменти для шифрування інформації мають власні діри, через що застосування сильної криптографії втрачає всякий зміст;
- відсутність належного захисту підсистем вилученого адміністрування. І хоча наявність веб-інтерфейсу зручно, оскільки дозволяє адміністраторові управляти системою з будь-якого підключеного до інтернету комп'ютера, при відсутності надійного захисту теж саме може робити й зловмисник [5].

Спеціалізовані програмні засоби (ПС) забезпечення мережної інформаційної безпеки здатні досить надійно відгородити веб-сайт від погроз, викликаних цими й іншими типами уразливостей. Однак це можливо тільки при правильнім настроюванні й обслуговуванні таких засобів: своєчасній установці новітніх відновлень, відстеженні підозрілої мережної активності і т.ін.

Основні типи ПС, що забезпечують мережну інформаційну безпеку це мережні монітори, аналізатори мережних протоколів і МЕ.

Мережні монітори служать для постійного моніторингу корпоративної мережі на предмет виявлення атак, тобто, призначені для "активної" захисту. Дані системи виявляють атаки на вузли корпоративної мережі й реагують на них заданим адміністратором безпеки алгоритмом. Сучасні монітори підтримують безліч інших функцій крім своїх основних по визначенню. Наприклад, ведення мережної статистики, у яку входять такі дані як коефіцієнт використання сегмента, рівень колізій, рівень помилок і рівень широкомовного трафіку, визначення швидкості поширення сигналу. Сюди ж ставиться перевірка

легітимності мережних адаптерів, якщо раптом з'являється «підозрілий». Монітори бувають як програмні, так і апаратні. Однак апаратні монітори служать в основному для статистичних функцій і не забезпечують гідного захисту інформації.

Аналізатори протоколів є окремим класом ПЗ, хоча вони, по суті, є частиною мережних моніторів. У кожний монітор вбудоване як мінімум кілька аналізаторів протоколів. Вони застосовуються окремо від мережних моніторів, коли недоцільно застосовувати потужний і, отже, більш дорогий монітор. Аналізатори протоколів, переводять мережний адаптер у безладний режим і збирають увесь трафік мережі для наступного аналізу. Аналізатор протоколів є досить корисним інструментом, який допомагає знаходити й усувати несправності, позбуватися від вузьких місць, що знижують пропускну здатність мережі, і вчасно виявляти проникнення в неї комп'ютерних зломщиків.

1.2 Мережеві екрани й виконувані ними завдання

Міжмережеві екрани є найпоширенішими засобами мережного захисту (МЕ є стандартним елементом операційної системи Windows). Вони широко використовуються як для захисту домашніх комп'ютерів, так і для комплексного екранування більших корпоративних мереж.

МЕ являє собою систему або комбінацію систем, що дозволяють розділити мережа на дві або більш частин і реалізувати набір правил, що визначають умови проходження пакетів з однієї частини в іншу. МЕ, пропускає через себе весь трафік. Для кожного минаючого пакета він ухвалює рішення пропускати його або відкинути. Існує набір правил, по яких МЕ ухвалює рішення щодо пропуску пакета.

Усі МЕ можна розділити на три типи: пакетні фільтри, сервера прикладного рівня, сервера рівня з’єднання.

МЕ з пакетними фільтрами ухвалюють рішення щодо того, пропускати пакет або відкинути, переглядаючи Ір-адреси, прапори або номери TCP портів у заголовку цього пакета. Для опису правил проходження пакетів складаються спеціальні таблиці. МЕ із серверами прикладного рівня використовують сервера конкретних сервісів – TELNET, FTP і т.д., що запускаються на екрані й проникні через себе весь трафік, що відноситься до даного сервісу. Таким чином, між клієнтом і сервером утворюються два з'єднання: від клієнта до МЕ й від МЕ до місця призначення. Використання серверів прикладного рівня дозволяє розв'язати важливе завдання – сховати від зовнішніх користувачів структуру локальної мережі, включаючи інформацію в заголовках поштових пакетів або служби доменних імен (DNS).

МЕ використовуються повсюдно при захисті локальних мереж малих підприємств, великих корпоративних мереж, веб-сервісів, домашніх комп'ютерів, приєднаних до глобальних мереж. І якщо при використанні в малих мережах можна не обертати великої уваги на продуктивність, то при захисті великих корпоративних мереж або популярних веб-порталів, підключених до інтернету через високошвидкісні з'єднання, саме МЕ стає головним фактором, що знижують швидкість доступу в Інтернет.

При розгортанні МЕ в мережах з високими вимогами до продуктивності їх конфігурують таким чином, щоб звести джерела формування вузьких місць до апаратних ресурсів комп'ютера. У випадку з одним комп'ютером початкового рівня потенційним вузьким місцем є не процесор, а пропускна здатність Інтернет-каналу.

З погляду витрат, ефективніше всього проектувати систему з акцентом на ресурси центрального процесора (ЦП), тому що це самий дорогою у відновленні ресурс. Компенсувати нестачу інших ресурсів помітно дешевше – досить установити інший диск або новий мережний адаптер, розширити пам'ять. Настроїти апаратне забезпечення рекомендується таким чином, щоб добитися максимально ефективного споживання ресурсів ЦП.

Як і у випадку з більшістю додатків, продуктивність МЕ підвищується зі збільшенням швидкодії ЦП. У той же час, це збільшення не приводить до лінійного підвищення продуктивності. Через частий і об'ємний доступ до пам'яті збільшення швидкодії ЦП може привести до того, що певна частина циклів ЦП буде простоювати чекаючи відповіді від пам'яті. При споживанні не більш 60% ресурсів ЦП знизити навантаження на процесор допомагає технологія hyper-threading. При більш високому рівні споживання ресурсів її включення не дасть ефекту.

Також при обробці більших обсягів даних доступ до пам'яті здійснюється часто. Кеши рівня 2 і 3 збільшують продуктивність при великих вибірках пам'яті. Тому що робота МЕ пов'язана із проходженням через мережні обладнання, пам'ять і ЦП значних обсягів даних, на його продуктивність впливає не тільки процесор, але й характеристики інших елементів системи. Загальна пропускна здатність підвищується при збільшенні швидкодії зовнішньої шини пам'яті й шин вводу-виводу.

Продуктивність МЕ також можна підвищити шляхом відповідного його настроювання. Одна зі складових такого настроювання включає настроювання фільтрів рівня додатків. Наприклад, для фільтра HTTP може бути включена функція перегляду корисних даних HTTP у пошуку певних сигнатур. При включенні цієї функції обсяг обробки збільшується, а виходить, комп'ютер, на якім установлений МЕ, починає споживати більше ресурсів. Тому при можливості слід заміняти фільтри рівня додатків на просту фільтрацію мережного й транспортного рівня.

При роботі з HTTP трафіком також широко застосовується кешування, однак воно можливо тільки при використанні веб-проксі, що саме по собі трохи знижує продуктивність. Однак якщо основна маса трафіка, що проходить через МЕ, є HTTP трафіком, то застосування правильне настроєного веб-прокси й кешування дає дуже великий виграш у продуктивності й МЕ й веб-серверів, зменшуючи споживання пропускної здатності Інтернет-каналу.

Правила фільтрації також слід розташовувати в списку правил у порядку відповідному найбільшої продуктивності. Оскільки більша частина МЕ працює за принципом "першої відповідності" при ухваленні рішення про передачу або відхилення пакета, те найбільше часто використовувані правила слід розміщати якнайближче до вершини списку. Таким чином, якщо 90% трафіка відповідає першому правилу в списку, то для цього трафіку буде виконуватися тільки одна перевірка.

Продуктивність також можна підвищити, настроївши протоколювання подій МЕ. Багато екранів надають можливість відключення протоколювання деяких категорій подій, які не мають великого значення для адміністрування в заданих умовах, або зміни методу ведення протоколювання на менш ресурсномісткий. Наприклад, ISA Server надає можливість вибору методу ведення журналу активності МЕ: ведення журналу засобами MSDE або ведення журналу у вигляді файлу. У випадку вибору методу ведення журналу у вигляді файлу ефективність споживання ресурсів процесора збільшується на 10-20% за рахунок зменшення функціональності.

Незважаючи на те, що виробники МЕ часто дають рекомендації з підвищення їх продуктивності, до остаточного дозволу цієї проблеми ще дуже далеко.

1.3 Методи та алгоритми керування доступом до інформації

Ідентифікацію та аутентифікацію можна вважати основою програмно-технічних засобів безпеки, оскільки інші сервіси розраховані на обслуговування іменованих суб'єктів. Ідентифікація і аутентифікація – це перша лінія оборони, "прохідна" інформаційного простору організації. Ідентифікація дозволяє суб'єкту – користувачу або процесу, діє від імені певного користувача, назвати себе, повідомивши своє ім'я. За допомогою аутентифікації друга сторона

пересвідчується, що суб'єкт дійсно той, за кого себе видає. Як синонім слова "аутентифікація" іноді використовують поєднання "перевірка автентичності". Суб'єкт може підтвердити свою автентичність, якщо пред'явить принаймні одну з таких сутностей:

- щось, що він знає: пароль, особистий ідентифікаційний номер, криптографічний ключ і т.п.;
- щось, чим він володіє: особисту картку або інший пристрій аналогічного призначення;
- щось, що є частиною його самого: голос, відбитки пальців тощо, тобто свої біометричні характеристики;
- щось, асоційоване з ним, наприклад координати. Надійна ідентифікація і аутентифікація утруднена по ряду принципових причин.

По-перше, комп'ютерна система ґрунтується на інформації в тому вигляді, в якому вона була отримана; строго кажучи, джерело інформації залишається невідомим. Наприклад, зловмисник міг відтворити раніше перехоплені дані. Отже, необхідно вжити заходів для безпечного введення і передачі ідентифікаційної і аутентифікаційної інформації; в мережевому середовищі це пов'язане з особливими труднощами. По-друге, майже всі аутентифікаційні сутності можна дізнатися, вкрасти або підробити. По-третє, є протиріччя між надійністю аутентифікації з одного боку, і зручностями користувача та системного адміністратора з іншого. Так, з міркувань безпеки необхідно з певною частотою просити користувача повторно вводити аутентификационную інформацію (адже на його місце міг сісти інша людина), а це підвищує ймовірність підглядання за введенням. По-четверте, чим надійніше засіб захисту, тим воно дорожче. Необхідно шукати компроміс між надійністю, доступністю за ціною і зручністю використання і адміністрування засобів ідентифікації та аутентифікації. Зазвичай компроміс досягається за рахунок комбінування двох перших з перерахованих базових механізмів автентифікації. Найбільш поширеним засобом аутентифікації є паролі.

Система порівнює введений і раніше заданий для даного користувача, пароль; у разі збігу справжність користувача вважається доведеною. Інший засіб, поступово набирає популярність і забезпечує найбільшу ефективність, – секретні криптографічні ключі користувачів. Головне достоїнство парольного аутентифікації – простота і звичність. Паролі давно вбудовані в операційні системи та інші сервіси. При правильному використанні паролі можуть забезпечити прийнятний для багатьох організацій рівень безпеки. Тим не менше за сукупністю характеристик їх слід визнати самим слабким засобом перевірки автентичності. Надійність паролів ґрунтується на здатності пам'ятати їх і зберігати в таємниці. Пароль можна підглянути. Пароль можна вгадати методом грубої сили, використовуючи, бути може, словник. Якщо файл паролів зашифрований, але доступний на читання, його можна перекачати собі на комп'ютер і спробувати підібрати пароль, запрограмувавши повний перебір.

Засоби управління доступом дозволяють специфікувати і контролювати дії, які суб'єкти – користувачі і процеси можуть виконувати над об'єктами – інформацією та іншими комп'ютерними ресурсами. Мова йде про логічне управління доступом, яка реалізується програмними засобами. Логічне керування доступом – це основний механізм багатокористувацьких систем, покликаний забезпечити конфіденційність і цілісність об'єктів і, до деякої міри, їх доступність шляхом заборони обслуговування неавторизованих користувачів. Завдання логічного керування доступом полягає в тому, щоб для кожної пари (суб'єкт, об'єкт) визначити безліч допустимих операцій, залежне від деяких додаткових умов, і контролювати виконання встановленого порядку.

Контроль прав доступу проводиться різними компонентами програмного середовища – ядром операційної системи, додатковими засобами безпеки, системою управління базами даних, посередницьким програмним забезпеченням (таким як монітор транзакцій) і т.д.

При прийнятті рішення про надання доступу зазвичай аналізується наступна інформація.

- ідентифікатор суб'єкта (ідентифікатор користувача, мережеву адресу комп'ютера тощо). – подібні ідентифікатори є основою добровільного управління доступом;
- атрибути суб'єкта (мітка безпеки, група тощо), причому мітки безпеки – основа примусового управління доступом;
- місце дії (системна консоль, надійний вузол мережі тощо);
- час дії (більшість дій доцільно вирішувати тільки в робочий час);
- внутрішні обмеження сервісу (число користувачів згідно ліцензії на програмний продукт тощо).

Зручною надбудовою над засобами логічного управління доступом є обмежуючий інтерфейс, коли користувача позбавляють можливості спробувати зробити несанкціоновані дії, включивши в число видимих йому об'єктів тільки ті, до яких він має доступ.

Під протоколюванням розуміється збір та накопичення інформації про події, що відбуваються в інформаційній системі підприємства. У кожного сервісу свій набір можливих подій, але в будь-якому випадку їх можна поділити на зовнішні – викликані діями інших сервісів, внутрішні – викликані діями самого сервісу, і клієнтські – викликані діями користувачів і адміністраторів.

Аудит – це аналіз накопиченої інформації, проводиться оперативно, майже в реальному часі, або періодично.

Реалізація протоколювання і аудиту переслідує наступні основні цілі:

- забезпечення підзвітності користувачів та адміністраторів;
- забезпечення можливості реконструкції послідовності подій;
- виявлення спроб порушень інформаційної безпеки;
- надання інформації для виявлення і аналізу проблем.

Забезпечення підзвітності важливо в першу чергу як засіб стримування. Якщо користувачі і адміністратори знають, що всі їх дії фіксуються, вони, можливо, утримаються від незаконних операцій. Якщо є підстави підозрювати будь-якого користувача в нечесності, можна реєструвати його дії особливо детально, аж до кожного натискання клавіші. При цьому забезпечується не тільки

можливість розслідування випадків порушення режиму безпеки, але і відкат некоректних змін. Тим самим забезпечується цілісність інформації.

Реконструкція послідовності подій дозволяє виявити слабкості в захисті сервісів, знайти винуватця вторгнення, оцінити масштаби завданої шкоди і повернутися до нормальної роботи.

Виявлення та аналіз проблем дозволяють допомогти поліпшити такий параметр безпеки, як доступність. Виявивши вузькі місця, можна спробувати переконфігурувати або переналаштувати систему, знову виміряти продуктивність і т.д.

Одним з найбільш потужних засобів забезпечення конфіденційності і контролю цілісності інформації є криптографія. У багатьох відношеннях вона займає центральне місце серед програмно-технічних регуляторів безпеки, будучи основою реалізації багатьох з них і, в той же час, останнім захисним кордоном.

Розрізняють два основних методи шифрування, що називаються симетричними і асиметричними. У першому з них один і той же ключ використовується і для шифрування, і для дешифрування повідомлень. Існують досить ефективні методи симетричного шифрування. Є стандарт на подібні методи – ГОСТ 28147-89 «Системи обробки інформації. Криптографічний захист. Алгоритм криптографічного перетворення».

Основним недоліком симетричного шифрування є те, що секретний ключ має бути відомий і відправнику, і одержувачу. З одного боку, це ставить нову проблему розсилки ключів. З іншого боку, одержувач, який має зашифроване і розшифроване повідомлення, не може довести, що він отримав його від конкретного відправника, оскільки таке ж повідомлення він міг згенерувати і сам.

В асиметричних методах застосовуються два ключа. Один з них, несекретний, використовується для шифрування і може публікуватися разом з адресою користувача, інший – секретний, застосовується для розшифровки та відомий тільки одержувачу. Найпопулярнішим з асиметричних є метод RSA, заснований на операціях з великими (100-значними) простими числами та їх творами.

Асиметричні методи шифрування дозволяють реалізувати так звану електронну підпис, або електронне засвідчення повідомлення. Ідея полягає в тому, що відправник надсилає два примірники повідомлення – відкрите і дешифроване його секретним ключем (природно, дешифрування незашифрованого повідомлення насправді є форма шифрування). Одержувач може зашифрувати за допомогою відкритого ключа відправника дешифрований примірник і порівняти з відкритим. Якщо вони співпадуть, особу та підпис відправника можна вважати встановленими.

Істотним недоліком асиметричних методів є їх низька швидкодія, тому їх доводиться поєднувати з симетричними, при цьому слід враховувати, що асиметричні методи на 3 – 4 порядки повільніше симетричних. Так, для вирішення завдання розсилки ключів повідомлення спочатку симетрично шифрують випадковим ключем, потім цей ключ шифрують відкритим асиметричним ключем одержувача, після чого повідомлення і ключ відправляються по мережі.

При використанні асиметричних методів необхідно мати гарантію автентичності пари (ім'я, відкритий ключ) адресата. Для вирішення цієї задачі вводиться поняття сертифікаційного центру, який посвідчує довідник імен/ключів своїм підписом.

Послуги, характерні для асиметричного шифрування, можна реалізувати і з допомогою симетричних методів, якщо є надійний третя сторона, яка знає секретні ключі своїх клієнтів. Ця ідея покладена, наприклад, в основу сервера аутентифікації Kerberos.

Криптографічні методи дозволяють надійно контролювати цілісність інформації. На відміну від традиційних методів контрольного підсумовування, здатних протистояти тільки випадковим помилкам, криптографічна контрольна сума (імітовставка), обчислена із застосуванням секретного ключа практично виключає всі можливості непомітного зміни даних.

Останнім часом набула поширення різновид симетричного шифрування, заснована на використанні складових ключів. Ідея полягає в тому, що секретний

ключ ділиться на дві частини, зберігаються окремо. Кожна частина сама по собі не дозволяє виконати розшифровку. Якщо у правоохоронних органів з'являються підозри щодо особи, яка використовує певний ключ, вони можуть отримати половинки ключа і далі діяти звичайним для симетричної розшифровки чином.

1.4 Опис алгоритмів роботи користувачів

Всі користувачі мережі були задумані як рівні співробітники однієї системи. Але деякі з них мають певні додаткові права. Привілеї-відрізняють таких користувачів від інших співробітників. Від типу мережної операційної системи залежить, які саме привілеї можна встановлювати в своїй мережі. Зазвичай права доступу розповсюджуються на цілі каталоги, хоча можливо встановити і спеціальний доступ до деяких файлів і груп файлів. При цьому використовується спеціалізоване ім'я файлу.

У більшості мереж права доступу встановлюються на весь каталог цілком і поширюються на всі підкаталоги, якщо на які-небудь з останніх не накладені спеціальні права.

Головною відмінністю атрибутів від прав доступу до мережевих системах являється те, що значення атрибута поширюється на всіх користувачів, охочих працювати з файлом. У той же час права доступу користувачів різні; тоді як один з них має право тільки читати файл, інший може користуватися необмеженим доступом до цієї інформації. Зрозуміло, що, як мінімум, одна людина в мережі повинен мати необмежений доступ до всієї інформації, що зберігається в мережі і до всіх мережних ресурсів. Така людина називається контролером мережі, або адміністратором. Він несе відповідальність за встановлення та роботу системи захисту. Ось чому на цього користувача не накладаються ніякі захисні обмеження.

У багатьох мережах адміністраторський вхід відкривається автоматично при встановлення системи. Ідентифікатор користувача та пароль, які використовуються в цьому вході, повинні бути відображені в мережевої документації. Вони однакові для будь-якої системи даного типу. Необхідно змінити пароль на такому вході. Інакше кожен користувач, який знає стандартні ідентифікатор і пароль, що встановлюються системою на вході адміністратора, зможе працювати в мережі з необмеженими можливостями доступу до будь-яких компонентів системи.

Кожен серверний комп'ютер в мережі повинен мати свій власний список користувальницьких входів. Якщо встановлена мережа з п'яти машин, причому кожна з них працює і як сервер, і як робоча станція, то необхідно створити п'ять різних списків: по одному на кожен сервер. Списками потрібно правильно управляти, інакше вони можуть вийти з-під контролю.

Необхідно стежити за тим, щоб ідентифікатор конкретного користувача був однаковим на всіх серверах. Нå обов'язково, щоб кожен користувач мав доступ до всіх серверів. В цілях безпеки системи краще, якщо користувачеві буде надано доступ тільки до тих серверних комп'ютерів, які потрібні безпосередньо для роботи.

У деяких мережах існує можливість копіювання користувальницьких списків з одного сервера на інший. Це дозволяє легше і ефективніше управляти мережею. Після того як складено один із списків, його можна скопіювати його на всі інші серверні машини. Якщо потрібно внести зміни в список, достатньо змінити лише одну копію, а потім просто записати її на всі мережеві сервери.

У деяких мережах, передбачена можливість використання одного загального списку для всіх серверних комп'ютерів. Можна користуватися віддаленими входами (remote accounts), які дозволяють обмежитися зберіганням користувацького списку усього на одному сервері. Інші серверні машини в разі потреби звертаються за інформацією до сервера, на якому знаходиться список.

Розглянуті способи захисту, надаються мережевим програмним забезпеченням. Але існує багато інших можливостей захистити мережеву

інформацію від стороннього вторгнення. Ось кілька варіантів подібної захисту. Всі комп'ютери мережі мають бути розташовані в надійних і безпечних місцях.

Необхідно дотримуватися обережності при роботі з принтером. Якщо відправляється на друк якась конфіденційна інформація, необхідно забезпечити, щоб вона роздруковувалася без присутності сторонніх осіб.

Якщо в мережі встановлений модем, що дозволяє користувачеві отримувати доступ до системи з віддаленого комп'ютера, то сторонніх вторгнень можна очікувати і з боку модему. В даному випадку необхідно, щоб кожен ідентифікатор користувача був захищений паролем.

1.5 Використання програмних екранів

Використання програмних екранів – це засіб розмежування доступу клієнтів з однієї множини серверів з іншою множиною. Міжмережевий екран виконує свої функції, контролюючи всі інформаційні потоки між цими двома системами.

У найпростішому випадку екран складається з двох механізмів, один з яких обмежує переміщення даних, а другий, навпаки, йому сприяє. У більш загальному випадку екран або напівпроникну оболонку зручно уявляти собі як послідовність фільтрів. Кожен з них може затримати дані, а може і відразу "перекинути" їх "на іншу сторону". Крім того, допускаються передача порції даних на наступний фільтр для продовження аналізу чи опрацювання даних від імені адресата і повернення результату відправнику.

Крім функцій розмежування доступу екрани здійснюють також протоколювання інформаційних обмінів. Зазвичай екран не є симетричним, для нього визначено поняття "усередині" і "зовні". При цьому завдання екранування формулюється як захист внутрішньої області від потенційно ворожої зовнішньої. Так, міжмережеві екрани встановлюють для захисту локальної мережі організації, що має вихід у відкриту середу, подібну Internet. Інший приклад екрану –

пристрій захисту порту, що контролює доступ до комунікаційного порту комп'ютера до і після незалежно від всіх інших системних захисних засобів.

Екранування дозволяє підтримувати доступність сервісів внутрішньої області, зменшуючи або взагалі ліквідуючи навантаження, індуковану зовнішньою активністю. Зменшується вразливість внутрішніх сервісів безпеки, оскільки спочатку сторонній зловмисник повинен подолати екран, де захисні механізми сконфігуровані особливо ретельно і жорстко. Крім того, екранує система, на відміну від універсальної, може бути влаштована більш простим і, отже, більш безпечним чином.

Екранування дає можливість контролювати також інформаційні потоки, спрямовані в зовнішню область, що сприяє підтримці режиму конфіденційності.

Важливим поняттям екранування є зона ризику, яка визначається як множина систем, які стають доступними зловмиснику після подолання екрану або будь-якого з його компонентів. Для підвищення надійності захисту, екран реалізують як сукупність елементів, так що "злам" одного з них ще не відкриває доступ до всієї внутрішньої мережі. Екранування і з точки зору поєднання з іншими сервісами безпеки, і з точки зору внутрішньої організації використовує ідею багаторівневого захисту, за рахунок чого внутрішня мережа виявляється в межах зони ризику тільки у разі подолання зловмисником декількох, по-різному організованих захисних рубежів. Екранування може використовуватися як сервіс безпеки не тільки в мережевий, але і в будь-який інший середовищі, де відбувається обмін повідомленнями.

Невеликими мережами користуються в основному невеликі організації, де всі співробітники знають один одного і довіряють один одному. Однак, навіть у цьому випадку мережа повинна забезпечувати хоча б мінімальні засоби захисту інформації своїх користувачів.

У будь-якій організації знайдуться документи і відомості, які не обов'язково знати всім користувачам місцевої мережі. Така інформація повинна зберігатися в спеціальному каталозі, доступ до якого мають лише уповноважені особи.

Частіше цікавість, ніж злий умисел співробітників змушують їх читати чужі файли. Далеко не кожен користувач мережі настільки сильний і в інших комп'ютерних системах, щоб мати необмежений доступ до мережевих дисків. Одна необережна команда може знищити весь каталог мережевих файлів. Одна з причин, по якій в мережах встановлюють систему захисту, полягає в тому, щоб уберегти мережеву інформацію від необдуманих дій користувачів.

Перший крок по встановленню системи захисту полягає у створенні спеціальних користувальницьких входів, що надають доступ до мережі тільки певного складу користувачів. Якщо користувач не має свого входу, він не зможе увійти в мережу.

Кожен вхід пов'язаний з ідентифікатором користувача, який вводиться при вході в мережу. Крім користувальницького коду, вхід містить також іншу інформацію про свого власника: пароль, повне ім'я та права доступу, які визначають, які дії і мережні команди дозволено використовувати в роботі цього співробітника, а які ні.

Іноді система встановлена таким чином, що певна група користувачів може працювати в мережі тільки в певний період часу.

Можливість створення спеціалізованих входів значно полегшує роботу, так як можна надати рівні права користування мережею деякій групі співробітників. Однак, справа в тому, що користувачі спеціалізованого входу працюють з одним і тим же паролем. Це значно послаблює систему захисту мережі, оскільки вона діє ефективніше, якщо кожен користувач має свій особистий пароль і зберігає його в найсуворішому секреті.

Якщо є потреба надати однакові права доступу певної групи співробітників, краще користуватися не спеціалізованими, а груповими входами. У цьому випадку кожен користувач входу має як би окремий подвход з власним ідентифікатором та паролем, проте всім абонентам групового входу надаються рівні права при роботі з мережевою системою. Такий підхід набагато надійніше, оскільки кожен співробітник має свій особистий мережевий пароль.

Одним з найважливіших аспектів системи мережевого захисту є система особистих паролів співробітників.

Іноді встановлюють також час дії пароля. Наприклад, 30 днів. Після закінчення цього терміну користувач повинен змінити пароль. Це не дуже зручно, проте скорочує ризик того, що хто-небудь дізнається пароль і захоче ним скористатися трохи пізніше. Користувальницькі входи і паролі – це перша лінія оборони системи захисту. Після того, як користувач отримав доступ до мережі, ввівши правильний ідентифікатор і пароль, він переходить до другої лінії, надається системою захисту: мережа визначає привілеї, які має даний користувач.

1.6 Мета і завдання дослідження

Наразі 63% веб-сайтів украй уразливі для атак з мережі. Найпоширенішими вразливостями є відсутність перевірки параметрів в HTTP-запитах, недотримання політик керування доступом до ресурсів, недотримання правил керування обліковими записами й користувацькими сесіями, уразливості, пов'язані з помилками в механізмі Cross-Site Scripting помилки переповнення буфера, відсутність належного контролю над параметрами, переданими комп'ютерами при доступі до зовнішніх ресурсів, невдале використання криптографії й відсутність належного захисту підсистем вилученого адміністрування.

Існуючі ПС забезпечення мережної інформаційної безпеки при грамотній експлуатації здатні усунути більшу частину вразливостей.

Найпоширенішими типами ПС забезпечення мережної інформаційної безпеки є мережні монітори, аналізатори протоколів і мережеві екрани.

Найпоширенішим засобом забезпечення мережної інформаційної безпеки є мережеві екрани, які здатні відгородити, під мережу, що захищається, від небажаного трафіку шляхом його аналізу на предмет відповідності заданій політиці безпеки.

Існують як апаратні, так і програмні методи підвищення продуктивності мережевих екранів. Також виробники дають подібні рекомендації для своїх продуктів. Однак проблема забезпечення високої продуктивності даного класу ПС ще далека від дозволу.

Метою дослідження, проведеного в атестаційній роботі, є розробка рекомендацій з поліпшення продуктивності програмних засобів, зокрема мережевих екранів, використовуваних для захисту даних інформаційних веб-сайтів.

Для досягнення поставленої мети планується розв'язати наступні завдання:

- провести аналіз стану проблеми забезпечення мережної безпеки програмними засобами, зокрема мережевими екранами, на інформаційних веб-сайтах;
- провести планування експерименту по визначенню продуктивності системи мережевих екранів, включених у мережу за типовою схемою захисту даних інформаційних веб-сайтів;
- розробити прототип програмного забезпечення для проведення експерименту;
- одержати емпіричні формули залежності продуктивності досліджуваних мережевих екранів від використовуваних правил безпеки;
- проаналізувати отримані емпіричні формули й розробити рекомендації з підвищення продуктивності мережевих екранів у заданих умовах.

У ході роботи планується перевірити гіпотезу про те, що час, що витрачений на перевірку пакетів на відповідність прийнятій політиці безпеки, можна оптимізувати шляхом перерозподілу порядку правил безпеки в списках мережевого екрана таким чином, щоб перевірки, що найбільше повільно виконуються, проводилися як можна рідше.

2 ОПИС ПРОВЕДЕНИХ ТЕОРЕТИЧНИХ ДОСЛІДЖЕНЬ

2.1 Опис алгоритмів ідентифікації та вибору рівнів факторів

У загальному випадку МЕ не є одним обладнанням або програмою. Як правило, краще представляти його як систему компонентів, які призначені для керування доступом до внутрішньої локальної й зовнішньої мереж на основі обраної політики безпеки. На функціонування цієї системи компонентів впливає безліч факторів. Найбільш важливі з них представлені нижче.

Не у всіх сегментах мережі прийнятий однаковий розмір кадрів даних. Оскільки Internet являє собою об'єднання безлічі мереж, розмір ІртПакетів, переданих у який-небудь сегмент, може виявитися більше припустимого в ньому [13]. Протокол IP дозволяє розбивати пакет на трохи більш дрібних пакетів, які знову збираються після доставки до місця призначення. Сильно фрагментовані пакети здатні сповільнити роботу МЕ, оскільки фрагментація ускладнює процес аналізу пакетів на предмет їх відповідності правилам, заданим для МЕ. У реальному веб-порталі фрагментація пакетів має випадковий характер. При проведенні експерименту тестова програма моделює фрагментацію пакетів, але цей фактор є некерованим.

Як уже згадувалося, МЕ є системою компонентів. Основними компонентами в цій системі служать пакетні фільтри й proxy-сервера. Ці компоненти реалізуються різними способами й забезпечують різний рівень захисту. Існують типові схеми включення компонентів у мережу з метою формування МЕ. Найбільш простим з них є установка пакетного фільтра на границі, що захищається підмережи. Пакетний фільтр просто відкидає пакети, що прийшли із завданих адміністратором вузлів, і не пропускає їх в, що захищається підмережа. Аналогічно пакети з, що захищається підмережи не відповідні до правил не пропускаються в зовнішню підмережа. Аналіз пакетів робиться тільки на основі вмісту заголовка IP-пакету. Однак заголовок містить в основному адреси й інші дані, необхідні протоколу для доставки пакета до місця

призначення. У заголовок пакета не включені деталі, які могли б допомогти фільтру розв'язати, чи випливає пропускати пакет через екран. Наприклад, пакетний фільтр може визначити, що пакет якогось протоколу, скажемо, FTP [13]. Але він не в змозі розпізнати, який це запит – get (одержати) або put (передати). При цьому не виключене, що запити одного типу виявляться цілком припустимими, а запити іншого типу для вузла повинні будуть блокуватися. Для цілей аналізу вмісту IP-пакету служать proxy-сервера (або шлюзи рівня додатків). Можна побудувати МЕ простим включенням такого proxy-сервера на границі, що захищається підмережі. Однак більш ефективні методи припускають комбінацію пакетних фільтрів і proxy-серверів. Прикладом такої комбінації може служити схема включення мережевого екрана в мережу у вигляді екранованого вузла. У цьому випадку МЕ використовує захисні можливості й пакетного фільтра, і proxy-сервера. При цьому для підключення до зовнішньої мережі служить пакетний фільтр, а для забезпечення безпеки різних сервісів – proxy-сервер. Схема з екранованою підмережею є подальшим розвитком мережі з екранованим вузлом. За даною схемою на границі, що захищається підмережі встановлюються два пакетні фільтри, між якими включаються один або трохи proxy-серверів, що забезпечують роботу одного або декількох мережних сервісів [14].

Таким чином, остання схема мінімум у два рази збільшує кількість перевірок, які потрібно пройти пакету, щоб потрапити із зовнішньої мережі в, що захищається підмережа у порівнянні зі схемою з єдиним пакетним фільтром на границі підмережи. Час проходження пакета через систему МЕ також збільшується, але остання схема, при правильнім настроюванні, значно надійніше першої.

При проведенні експерименту використовується фіксована схема включення системи МЕ в мережу, відповідна до типової схеми, використовуваного при організації захисту веб-порталу.

У результаті конкуренції серед виробників МЕ і їх спроб удосконалити свій продукт брандмауери наділялися новими властивостями. Оскільки МЕ перебуває

на границі підмережи, він повинен виконувати безліч завдань, у тому числі й не пов'язаних із забезпеченням безпеки:

– кэшування (caching). Ця властивість особлива характерно для мереж, що містять веб-сервери з більшим обсягом інформації, доступної з Internet. Завдяки локальному зберіганню часто запитуваних даних кэшируючий сервер може поліпшити час реакції на запит користувача й заощадити смугу пропущення, яка потрібна була б для повторного завантаження даних;

– трансляція адреси (address translation). Настроєний відповідним чином МЕ дозволяє застосовувати для внутрішньої мережі будь-які IpeАдреси. При цьому зовні видний тільки адреса МЕ [3];

– фільтрація вмісту (content restriction). Усе більше число продуктів забезпечує обмеження інформації, одержуваної користувачами з Internet, шляхом блокування доступу до адрес URL, що містять небажаний уміст, або пошуку заданих ключових слів у прихожих IP-пакетах [3];

– переадресація (address vectoring). Ця функція надає брандмауерові можливість змінювати запити так, щоб вони направлялися серверу не із зазначеним у пакеті запиту Ip-адресою, а з іншим. Таким способом удасться розподіляти навантаження між декількома серверами, які для зовнішнього користувача виглядають як одиночний сервер.

При проведенні експериментів додаткові функції системи МЕ відключені й не впливають на продуктивність. Політика безпеки, реалізована міжмережевим екраном

Проходячи через систему мережевого екрана, Ip-пакети перевіряються на їхню відповідність заданим правилам безпеки. Наявність великої кількості таких правил на пакетних фільтрах і proxy-серверах МЕ збільшує кількість перевірок і відповідно збільшує час проходження пакета. Таким чином, набір правил політики безпеки також впливає на швидкість роботи МЕ.

Правила й групи правил політики безпеки використовуються при проведенні експериментів у якості керованих факторів. Ці фактори представляються у вигляді плану експерименту в додатку А. Найпоширенішим

принципом формування політики безпеки є принцип "заборонене все, що не дозволене", згідно з яким користувачам надаються тільки ті права, які мінімально необхідні їм для виконання своїх функцій.

Для розробки такої політики й відповідних їй правил безпеки необхідно виділити категорії користувачів веб-порталу й визначити права для кожної категорії. Інформаційний веб-портал як правило має наступні категорії користувачів:

Публічні користувачі – найбільш численна категорія користувачів, на обслуговування інформаційних потреб якої властиво й орієнтований веб-портал.

Небажані користувачі, замічені в поширенні спама або інших зловмисних діях. Використання веб-порталу такими користувачами може бути обмежене.

Адміністратор безпеки, у функції якого входить контроль і керування системою безпеки порталу, а також реагування на спроби порушення безпеки порталу.

Адміністратор бази даних, у функції якого входить підтримка в актуальному робочому стані повного обсягу оперативної інформації, що й накопичується, а також захист інформації від несанкціонованого доступу.

Адміністратор інформаційного наповнення визначають політики, атрибути й правила використання різних типів інформаційного наповнення веб-порталу.

Визначимо IP адреси й порти, які будуть відповідати перерахованим вище групам у моделі мережі інформаційного порталу:

192.168.0.1 порти 5000-65535 – публічні користувачі;
192.168.0.1 порт 4444 – адміністратор бази даних;
192.168.0.1 порт 5555 – адміністратор безпеки;
192.168.0.1 порт 6666 – адміністратор інформаційного наповнення;
192.168.0.2 порт 80 – веб-сервер порталу;
192.168.0.2 порт 25 – поштовий сервер;
192.168.0.4 порт 2222 – сервер баз даних;
192.168.0.4 порт 2224 – сервер додатків;
192.168.0.4 порт 2225 – сервер аутентифікації й авторизації;
192.168.0.4 порт 1111 – сервер центру керування безпекою.

Представлений набір правил фільтрації пакетів на мережному й транспортному рівнях. Перші два правила визначають дозволу для вхідного й вихідного трафіка для певного набору протоколів, по яких користувачі можуть взаємодіяти з порталом. Третє правило встановлює дозволу тільки для вхідного трафіка по протоколах msa і odmr. Ці протоколи застосовуються поштовим сервером, використовуваним у моделі. Наступні два правила визначають дозвіл на взаємодію користувачів із зовнішньої мережі прямо із серверами сегмента керування й сегмента службових серверів по певних протоколах. Ці правила необхідні для забезпечення вилученого адміністрування порталу.

Правила фільтрації внутрішнього МЕ. Внутрішній екран, на відміну від зовнішнього, робить фільтрацію тільки на транспортному й мережному рівнях. Перші два правила дозволяють роботу із серверами сегмента службових серверів із зовнішньої мережі. Наступні два правила дозволяють роботу із серверами сегмента керування із зовнішньої мережі. Таким чином, екран підтримує вилучене керування веб-порталом. Наступні два правила визначають дозвіл на роботу із сервером додатків у сегменті службових серверів для серверів загальнодоступного сегмента. Останнє правило є правилом за замовчуванням і згідно зі стратегією «заборонене все, що явно недозволене» забороняє весь інший трафік між сегментами керування й службових серверів порталу й іншою частиною мережі.

2.2 Класифікація міжмережевих екранів

Міжмережевий екран як пристрій контролю доступу в мережу, призначений для блокування всього трафіку, за винятком дозволених даних. Цим воно відрізняється від маршрутизатора, функцією якого є доставка трафіку в пункт призначення в максимально короткий термін.

Існує думка, що маршрутизатор також може відігравати роль міжмережевого екрану. Однак між цими пристроями існує одне принципове розходження: маршрутизатор призначений для швидкої маршрутизації трафіку, а не для його блокування. Міжмережевий екран являє собою засіб захисту, що пропускає певний трафік з потоку даних, а маршрутизатор є мережевим пристроєм, який можна настроїти на блокування певного трафіку.

Крім того, міжмережеві екрани, як правило, мають більший набір настроювань. Проходження трафіка на міжмережевому екрані можна надбудовувати по службах, IP-адресам відправника й одержувача, по ідентифікаторах користувачів, що запитують службу. Міжмережеві екрани дозволяють здійснювати централізоване керування безпекою. В одній конфігурації адміністратор може настроїти дозволений вхідний трафік для всіх внутрішніх систем організації. Це не усуває потребу у відновленні та настроюванні систем, але дозволяє знизити ймовірність невірного конфігурування однієї або декількох систем, у результаті якого ці системи можуть піддатися атакам на некоректно настроєну службу.

Існують два основних типи екранів: міжмережеві екрани прикладного рівня й міжмережеві екрани з пакетною фільтрацією. У їхній основі лежать різні принципи роботи, але при правильному настроюванні обидва типи програм забезпечують правильне виконання функцій безпеки, що полягають у блокуванні забороненого трафіка.

Міжмережеві екрани прикладного рівня, або проксі-екрани, являють собою програмні пакети, що базуються на операційних системах загального призначення (таких як Windows та Unix) або на апаратній платформі. Міжмережевий екран має декілька інтерфейсів, по одному на кожну з мереж, до яких він підключений. Набір правил політики визначає, яким чином трафік передається з однієї мережі в іншу. Якщо в правилі відсутнє явний дозвіл на пропуск трафіка, міжмережевий екран відхиляє або анулює пакети.

Правила політики безпеки підсилюються за допомогою використання модулів доступу. У міжмережевому екрані прикладного рівня кожному

розв'язуваню протоколу, повинен відповідати свій власний модуль доступу. Кращими модулями доступу вважаються ті, що побудовані спеціально для протоколу, що дозволяє доступ. Наприклад, модуль доступу FTP призначений для протоколу FTP і може визначати, чи відповідає минаючий трафік цьому протоколу й чи дозволений цей трафік правилами політики безпеки.

При використанні міжмережевого екрана прикладного рівня усі з'єднання проходять через нього. Як показано на рисунку 2.1, з'єднання починається на системі-клієнті й надходить на внутрішній інтерфейс міжмережевого екрана. Міжмережевий екран приймає з'єднання, аналізує вміст пакета й використаний протокол і визначає, чи відповідає даний трафік правилам політики безпеки. Якщо це так, то міжмережевий екран ініціює нове з'єднання між своїм зовнішнім інтерфейсом і системою-сервером.

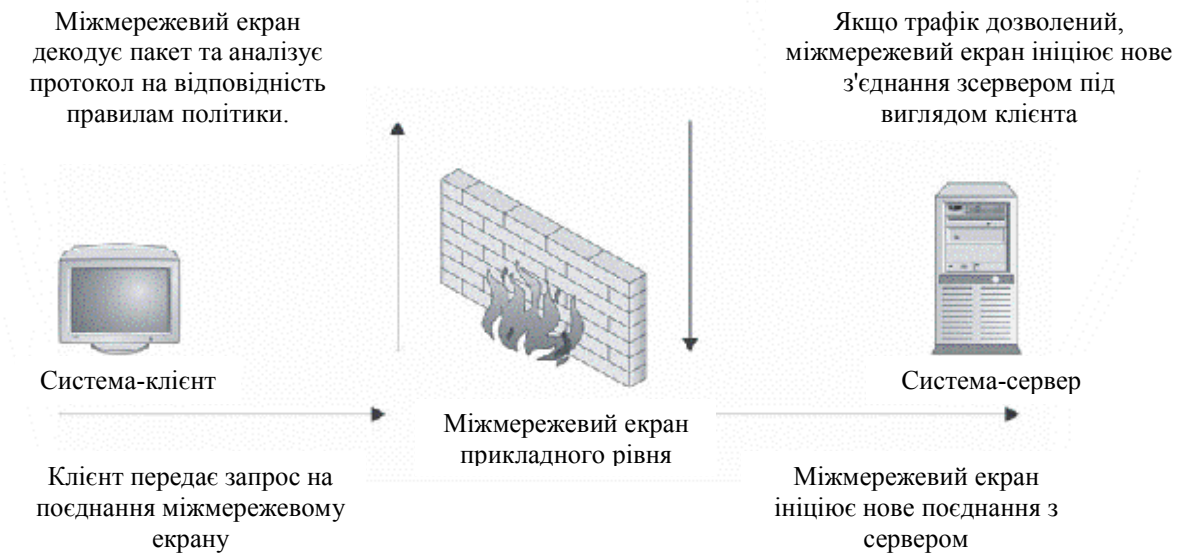

Рисунок 2.1 – З'єднання модуля доступу міжмережевого екрана прикладного рівня

Міжмережеві екрани прикладного рівня використовують модулі доступу для вхідних підключень. Модуль доступу в міжмережевому екрані приймає вхідне підключення й обробляє команди перед відправленням трафіку одержувачеві. Таким чином, міжмережевий екран захищає системи від атак, що виконуються за допомогою програмних додатків.

Тут мається на увазі, що модуль доступу на міжмережевому екрані сам по собі невразливий для атаки. Якщо ж програмне забезпечення розроблене недостатньо ретельно, це може бути й помилковим твердженням.

Додатковою перевагою архітектури даного типу є те, що при її використанні дуже складно, якщо не неможливо, «сховати» трафік усередині інших служб. Наприклад, деякі програми контролю над системою, такі як NetBus й Back Orifice, можуть бути настроєні на використання будь-якого порту, що використований користувачем. Отже, їх можна настроїти на використання порту 80 (HTTP). При використанні вірно настроєного міжмережевого екрана прикладного рівня модуль доступу не зможе розпізнавати команди, що надходять через з'єднання, і з'єднання, швидше за все, не буде встановлено.

Міжмережеві екрани прикладного рівня містять модулі доступу для найбільше часто використовуваних протоколів, таких як HTTP, SMTP, FTP й telnet. Деякі модулі доступу можуть бути відсутні. Якщо модуль доступу відсутній, то конкретний протокол не може використовуватися для з'єднання через міжмережевий екран.

Міжмережевий екран також приховує адреси систем, що розташовані по іншу сторону від нього. Тому що усі з'єднання ініціюються й завершуються на інтерфейсах міжмережевого екрана, внутрішні системи мережі не видні прямо ззовні, що дозволяє сховати схему внутрішньої адресації мережі.

Більша частина протоколів прикладного рівня забезпечує механізми маршрутизації до конкретних систем для трафіку, що спрямований через певні порти. Наприклад, якщо весь трафік, що надходить через порт 80, повинен направлятися на веб-сервер, це досягається відповідним настроюванням міжмережевого екрана.

Міжмережеві екрани з пакетною фільтрацією можуть також бути програмними пакетами, що базуються на операційних системах загального призначення (таких як Windows або Unix) або на апаратних платформах. Міжмережевий екран має кілька інтерфейсів, по одному на кожну з мереж, до яких підключений екран. Аналогічно міжмережевим екранам прикладного рівня,

доставка трафіка з однієї мережі в іншу визначається набором правил політики. Якщо правило не дозволяє явно певний трафік, то відповідні пакети будуть відхилені або анульовані міжмережевим екраном.

Правила політики підсилюються за допомогою використання фільтрів пакетів. Фільтри вивчають пакети й визначають, чи є трафік дозволеним, відповідно до правил політики й стану протоколу (перевірка з урахуванням стану). Якщо протокол додатка функціонує через TCP, визначити стан відносно просто, тому що TCP сам по собі підтримує стани. Це означає, що коли протокол перебуває в певному стані, дозволена передача тільки певних пакетів. Розглянемо як приклад послідовність встановлення з'єднання. Перший очікуваний пакет – пакет SYN. Міжмережевий екран виявляє цей пакет і переводить з'єднання в стан SYN. У даному стані очікується один із двох пакетів – або SYN ACK (упізнавання пакета й дозвіл з'єднання) або пакет RST (скидання з'єднання через відмову в з'єднанні одержувачем). Якщо в даному з'єднанні з'являться інші пакети, міжмережевий екран анулює або відхилить їх, тому що вони не підходять для даного стану з'єднання, навіть якщо з'єднання дозволене набором правил.

Якщо протоколом з'єднання є UDP, міжмережевий екран з пакетною фільтрацією не може використати стан, що властивий протоколу, замість чого відслідковує стан трафіка UDP. Як правило, міжмережевий екран приймає зовнішній пакет UDP й очікує вхідний пакет від одержувача, що відповідає вихідному пакету за адресою й портом, протягом певного часу. Якщо пакет приймається протягом цього відрізку часу, його передача дозволяється. У противному випадку міжмережевий екран визначає, що трафік UDP не є відповіддю на запит, і анулює його.

При використанні міжмережевого екрану з пакетною фільтрацією з'єднання не перериваються на міжмережевому екрані (рисунку 2.2), а направляються безпосередньо до кінцевої системи. При надходженні пакетів міжмережевий екран з'ясовує, чи дозволені даний пакет і стан з'єднання правилами політики. Якщо це так, пакет передається по своєму маршруту. У противному випадку пакет відхиляється або анулюється.

Рисунок 2.2 – Передача трафіка через міжмережевий екран з фільтрацією пакетів

Міжмережеві екрани з фільтрацією пакетів не використовують модулі доступу для кожного протоколу й тому можуть використовуватися з будь-яким протоколом, що працює через IP. Деякі протоколи вимагають розпізнавання міжмережевим екраном виконуваних ними дій. Наприклад, FTP буде використовувати одне з'єднання для початкового входу й команд, а інше – для передачі файлів. З'єднання, використані для передачі файлів, встановлюються як частина з'єднання FTP, і тому міжмережевий екран повинен ввміти зчитувати трафік і визначати порти, які будуть використовуватися новим з'єднанням. Якщо міжмережевий екран не підтримує цю функцію, передача файлів неможлива.

Як правило, міжмережеві екрани з фільтрацією пакетів мають можливість підтримки більшого обсягу трафіку, тому що в них відсутнє навантаження, що створюване додатковими процедурами настроювання й обчислення, що мають місце в програмних модулях доступу. Різні виробники міжмережевих екранів зіставляють їхню продуктивність різними способами. Історично склалося так, що міжмережеві екрани з пакетною фільтрацією мають можливість обробки більшого обсягу трафіку, ніж міжмережеві екрани прикладного рівня, на платформі того

самого типу. Це порівняння показує різні результати залежно від типу трафіку й числа з'єднань, що мають місце в процесі тестування.

Міжмережеві екрани, що працюють тільки за допомогою фільтрації пакетів, не використовують модулі доступу, і тому трафік передається від клієнта безпосередньо на сервер. Якщо сервер буде атакований через відкриту службу, що дозволена правилами політики міжмережевого екрана, екран ніяк не відреагує на атаку. Міжмережеві екрани з пакетною фільтрацією також дозволяють бачити ззовні внутрішню структуру адресації. Внутрішні адреси приховувати не потрібно, тому що з'єднання не перериваються на міжмережевому екрані.

2.3 Алгоритми конфігурування міжмережевих екранів

Як і багато інших пристроїв, міжмережеві екрани змінюються й удосконалюються із часом, тобто еволюціонують. Виробники міжмережевих екранів прикладного рівня в певний момент прийшли до висновку, що необхідно розробити метод підтримки протоколів, для яких не існує певних модулів доступу. Внаслідок цього побачила світ технологія модуля доступу Generic Services Proxy (GSP). GSP розроблена для підтримки модулями доступу прикладного рівня інших протоколів, необхідних системі безпеки й при роботі мережевих адміністраторів. У дійсності GSP забезпечує роботу міжмережевих екранів прикладного рівня як екрани з пакетною фільтрацією.

Виробники міжмережевих екранів з пакетною фільтрацією також додали деякі модулі доступу у свої продукти для забезпечення більш високого рівня безпеки деяких широко розповсюджених протоколів. На сьогоднішній день багато міжмережевих екранів з пакетною фільтрацією поставляються з модулем доступу SMTP.

У той час як базова функціональність міжмережевих екранів обох типів залишилася колишньої, (що є причиною більшості «слабких місць» цих

пристроїв), сьогодні на ринку присутні гібридні міжмережеві екрани. Практично неможливо знайти міжмережевий екран, функціонування якого побудовано винятково на прикладному рівні або фільтрації пакетів. Це обставина аж ніяк не є недоліком, тому що вона дозволяє адміністраторам, відповідальним за безпеку, надбудовувати пристрій для роботи в конкретних умовах.

Тепер треба розглянути деякі стандартні мережні архітектури й з'ясувати, яким чином варто надбудовувати мережевий екран у тієї або іншій конкретній ситуації. У цій вправі мається на увазі, що в організації присутні зазначені нижче системи, і що ці системи приймають вхідні з'єднання з інтернету:

- веб-сервер, що працює тільки через порт 80;
- поштовий сервер, що працює тільки через порт 25. Він приймає всю вхідну й відправляє всю вихідну пошту. Внутрішній поштовий сервер періодично зв'язується з даною системою для одержання вхідної пошти й відправлення вихідних повідомлень.

Існує внутрішня система DNS, що запитує системи інтернету для перетворення імен в адреси, однак в організації відсутня своя власна головна зовнішня DNS.

Інтернет-політика організації дозволяє внутрішнім користувачам використовувати наступні служби:

- HTTP;
- HTTPS;
- FTP;
- Telnet;
- SSH.

На базі цієї політики можна побудувати правила політики для різних архітектур.

Архітектура 1: системи за межами міжмережевого екрана, доступні з Інтернет. На рисунку 2.3 показане розміщення доступних з інтернету систем між мережним екраном і зовнішнім маршрутизатором. У табл. 2.1 наведені правила міжмережевого екрану.

Рисунок 2.3 – Системи за межами міжмережевого екрана, що доступні з Інтернет

Таблиця 2.1 – Правила міжмережевого екрана для розташованих за межами міжмережевого екрану систем, що доступні з Інтернет

Номер	Вихідний IP	Кінцевий IP	Служба	Дія
1	Внутрішній поштовий сервер	Поштовий сервер	SMTP	Прийняття
2	Внутрішня мережа	Поштовий сервер	Любою HTTP, HTTPS, FTP, telnet, SSH	Прийняття
3	Внутрішня DNS	Будь-яка	DNS	Прийняття
4	Будь-яка	Будь-яка	Будь-яка	Скидання

На маршрутизаторі може бути встановлена фільтрація, що дозволяє тільки зовнішнім http-даним надходити на веб-сервер і передавати на поштовий сервер тільки вступники ззовні дані SMTP. Як видно з наведених правил, незалежно від того, який тип міжмережевого екрану використовується, веб-сервер і поштовий сервер не захищені міжмережевим екраном. У цьому випадку міжмережевий екран лише захищає внутрішню мережу організації.

Архітектура 2: один міжмережевий екран. Друга стандартна архітектура показана на рисунку 2.4. У даній архітектурі використовується один міжмережевий екран для захисту як внутрішньої мережі, так і будь-яких інших систем, що доступні з інтернету. Ці системи розташовуються в окремій мережі.

Рисунок 2.4 – Один міжмережевий екран

Таблиця 2.2 – Правила міжмережевого екрана для архітектури з одним міжмережевим екраном

Номер	Вихідний IP	Кінцевий IP	Служба	Дія
1	Будь-який	Веб-сервер	HTTP	Прийняття
2	Будь-який	Поштовий сервер	SMTP	Прийняття
3	Поштовий сервер	Будь-який	SMTP	Прийняття
4	Внутрішня мережа	Будь-який	HTTP, HTTPS, FTP, telnet, SSH	Прийняття
5	Внутрішня DNS	Будь-який	DNS	Прийняття
6	Будь-який	Будь-який	Будь-яка	Скидання

Як видно з таблиці 2.2, правила практично аналогічні правилам архітектури 1. Міжмережевий екран доповнює правила, які використовувалися в маршрутизаторі в попередній архітектурі. Також ми бачимо, що не існує явного правила, що дозволяє внутрішньому поштовому серверу підключатися до поштового сервера в окремій мережі. Причиною цьому є правило 2, що дозволяє будь-якій системі (внутрішній або зовнішній) підключатися до згаданої системи.

Архітектура 3: подвійні міжмережеві екрани**.** Третя архітектура, про яку піде мова, використовує подвійні міжмережеві екрани. Доступні з інтернету системи розташовуються між міжмережевими екранами, а внутрішня мережа розташована за другим міжмережевим екраном. У таблиці 2.3 наведені правила для міжмережевого екрана 1.

Таблиця 2.3 – Правила міжмережевого екрана 1 в архітектурі із двома міжмережевими екранами

Номер	Вихідний IP	Кінцевий IP	Служба	Дія
1	Будь-який	Веб-сервер	HTTP	Прийняття
2	Будь-який	Поштовий сервер	SMTP	Прийняття
3	Поштовий сервер	Будь-який	SMTP	Прийняття
4	Внутрішня мережа	Будь-який	HTTP, HTTPS, FTP, telnet, SSH	Прийняття
5	Внутрішня DNS	Будь-який	DNS	Прийняття
6	Будь-який	Будь-який	Будь-яка	Скидання

Побудова набору правил міжмережевого екрана. Якісно створений набір правил не менш важливий, ніж апаратна платформа. Більша частина міжмережевих екранів працює за принципом «першої відповідності» при ухваленні рішення про передачу або відхилення пакета. При побудові набору правил відповідно до алгоритму «першої відповідності» найбільш специфічні правила розташовуються у верхній частині набору правил, а найменш специфічні (тобто більш загальні) – у нижній частині набору. Таке розміщення правил гарантує, що загальні правила не перекривають собою більш специфічні. Деякі міжмережеві екрани містять оброблювач набору правил, що перевіряє набір на наявність правил, що перекривають іншими правилами. Оброблювач інформує про цю ситуацію адміністратора міжмережевого екрана перед встановленням правил на міжмережевий екран.

Даний підхід гарний у загальному плані, однак він не вирішує проблему продуктивності міжмережевого екрана. Чим більше правил необхідно перевіряти для кожного пакета, тим більше обчислень повинен проводити міжмережевий

екран. При розробці якісного набору правил варто взяти до уваги це обставини, тому що від нього залежить рівень ефективності роботи міжмережевого екрана.

Для підвищення ефективності роботи екрана варто оцінити очікуване навантаження трафіку на міжмережевий екран й упорядкувати трафік по типах. Як правило, найбільший обсяг займає трафік HTTP. Для підвищення ефективності міжмережевого екрана варто розмістити правила, що стосуються до HTTP, вгорі набору правил. Це означає, що правило, що дозволяє внутрішнім системам використовувати HTTP для підключення до будь-якої системи в інтернеті, і правило, що дозволяє зовнішнім користувачам здійснювати доступ до веб-сайту організації, повинні бути розташовані дуже близько до верхньої межі набору правил. Єдиними правилами, які повинні перебувати вище двох згаданих правил, є специфічні правила відмови в доступі, що відносяться до протоколу HTTP.

2.4 Аналіз методів проектування

Для адекватного проектування мережевих зв’язків у бізнес-структурі потрібно провести аналіз цілей та структури функціонування IT-компанії. Мети функціонування підприємства. Ефективним інструментом установлення цілей системи керування є метод «дерева цілей», що складається із глобальної мети, цілей і підцілей. При побудові «дерева цілей» його проектування йде по методу «від загального до частки». Припинення декомпозиції мети на більш дрібні припиняється в той момент, коли подальший процес є недоцільним у рамках розгляду «головної мети». Правильно побудоване дерево цілей надалі легко може бути перетворене в план-графік або діаграму Ганта [12].

Конкретна функціональна структура керування визначається залежно від сполучення двох основних типів керівництва – лінійного (генеральний директор, рада директорів) і функціонального (спеціалізація керівників по окремих функціях керування) [13]. У складі підприємства доцільно виділити три області

керування: виробничу, що забезпечує й управлінську [14]. Функціональні завдання й підзадачі підприємства представлені у вигляді таблиці (табл. 2.4).

Таблиця 2.4 – Функціональні завдання й підзадачи IT-фірми

Номер і назва функціонального завдання	Номер і зміст функціональної підзадачи
1. Виробнича (надання інформаційних послуг)	1.1 Розробка програмного забезпечення 1.2 Продаж програмного забезпечення 1.3 Надання консультаційних послуг 1.4 Одержання замовлень від клієнтів й оформлення договорів на їхнє виконання
2. Управлінська	2.1 Керування кадрами 2.2 Планування фінансово-економічної діяльності організації 2.3 Аналіз інформації про діяльність підприємства

У таблиці 2.5 наведені можливі засоби й критерії досягнення поставлених цілей.

Набір правил фільтрації поштових повідомлень для зовнішнього МЕ – така фільтрація проводиться за принципом усунення відомих вразливостей і видалення повідомлень із небажаним умістом, тому всі правила заборонні. Перше правило блокує пошту з небажаних адрес, наприклад, спам. Наступні три правила блокують потенційно небезпечні для порталу повідомлення. Фрагментовані, деформовані, зашифровані й повідомлення в темі або змісті яких виявлені слова зі списку заборонених також відкидаються зовнішнім екраном.

Кожне правило безпеки може використовуватися або не використовуватися системою. Таким чином, число рівнів факторів рівно двом.

Загальне число досліджуваних правил безпеки при вимірі продуктивності першого МЕ дорівнює 16, а другого 6. Якщо виділяти кожне правило як окремий фактор полнофакторного експерименту, то загальна кількість вимірів у першому випадку буде рівно 65536, а в другому випадку – 64. Таким чином, має сенс об'єднати подібні за змістом, і, приблизно, що виявляють рівний невеликий вплив

на продуктивність системи правила в один фактор експерименту. У табл. 2.5 представлений набір факторів експерименту й відповідні йому правила безпеки для дослідження продуктивності зовнішнього МЕ.

Таблиця 2.5 – Фактори експерименту по виміру продуктивності зовнішнього МЕ

№ фактора	1	2	3	4	5
№ правил	1,2,3	4,5	13,16,18	12,14,15	17

Перші 3 правила обробляють трафік між демілітаризованою зоною (ДМЗ) і зовнішньої підмережею, причому найбільший вплив на продуктивність мають правила 1 і 2, які перевіряють відповідність трафіку великій кількості мережних протоколів, по яких можлива взаємодія з порталом. Третє правило обробляє трафік, що виконує службові функції по адмініструванню самого МЕ й, приблизно, має мінімальний вплив на продуктивність.

Четверте й п'яте правила обробляють трафік між зовнішньої підмережею й внутрішніми сегментами мережі порталу. Оскільки частка такого трафіку в загальній масі невелика, те, приблизно, вплив цих правил на продуктивність невелике і їх має сенс представити в якості єдиного фактора повно-факторного експерименту.

Правила 13, 16 і 18 обробляють поштовий трафік і мають справу із зовнішніми атрибутами повідомлення, такими як розмір і склад прикріплень. Аналіз цих атрибутів, приблизно, займає відносно небагато часу, тому має сенс також представити ці правила як окремий фактор.

Правила 12, 14 і 15 також мають відношення до поштового трафіка. Приблизно, усі вони є досить ресурсномісткими й можуть розглядатися як один вагомий фактор експерименту. Перше правило зіставляє адреса відправника зі списком небажаних адрес. Оскільки такий список звичайно досить об'ємний, такий аналіз може займати досить багато часу. Чотирнадцяте й п'ятнадцяте

правила дуже схожі тим, що аналізують повідомлення на наявність небезпечних html-тегів, що також є витратним за часом завданням.

Сімнадцяте правило представлене як окремий фактор оскільки, приблизно, виконується найбільший проміжок часу, тому що аналізує як заголовок, так і зміст поштового повідомлення на наявність слів із забороненого списку.

У таблиці 2.6 представлений набір факторів експерименту й відповідні йому правила безпеки для дослідження продуктивності внутрішнього МЕ.

Таблиця 2.6 – Фактори експерименту по виміру продуктивності внутрішнього мережевого екрану

№ фактора	1	2	3
№ правил	6,7	8,9	10,11

Усі пари правил, що представлено у таблиці дозволяють вхідний і вихідний трафік одного типу, тому логічно представити їх як один фактор.

Таким чином, при вимірі продуктивності зовнішнього МЕ аналізується вплив п'яти факторів. Кожний фактор має два рівні значень. Число опитів повнофактрного експерименту в цьому випадку дорівнює 25. При вимірі продуктивності внутрішнього екрана використовуються 3 фактора, а число опитів 8.

3 ОПИС ПРОЕКТУВАННЯ АЛГОРИТМІВ СИСТЕМИ

3.1 Опис структури мережі моделі веб-сайту

Модель веб-сайту при проведенні досліджень буде включати чотири комп'ютери об'єднаних у локальну мережу. Кожний комп'ютер у мережі буде моделювати окремий сегмент або елемент мережі реального порталу.

Перший комп'ютер моделює зовнішню, стосовно порталу підмережа – інтернет. Усі вільні порти цього комп'ютера застосовуються для моделювання адрес користувачів порталу.

Другий комп'ютер моделює сегмент загальнодоступних серверів порталу. На ньому встановлений і відповідним чином настроєний зовнішній МЕ порталу. Крім нього на комп'ютері встановлений поштовий сервер, інтегрований з екраном з метою забезпечення фільтрації пошти на рівні додатків. У моделі також будуть використовуватися два порти цього комп'ютера – один для моделювання поштового сервера, іншої для моделювання HTTP веб-сервера порталу.

На третьому комп'ютері встановлений і сконфігурований відповідним чином внутрішній МЕ, у чиї завдання входить захист сегмента службових серверів і сегмента керування безпекою веб-порталу.

Четвертий комп'ютер моделює сегмент службових серверів порталу. В експерименті використовуються 3 порту цього комп'ютера для моделювання сервера БД, сервера додатків і сервера аутентифікації й авторизації.

На рисунку 3.1 представлена логічна модель мережі веб-сайту.

У моделі мережі при проведенні експерименту використовуються наступні ПС:

- Kerio Winroute Firewall 6 – кросплатформений МЕ корпоративного рівня. У моделі виконує функції зовнішнього мережевого екрану;
- Mdaemon 10.1 – поштовий сервер;
- Ipfirewall – кросплатформений МЕ базового рівня. У моделі виконує функції внутрішнього мережевого екрана;
- спеціалізована програма для виміру продуктивності мережі.

Рисунок 3.1 – Логічна модель мережі веб-сайту, що використана при проведенні досліджень

При проведенні експериментів навантаження на МЕ буде створюватися шляхом пересилання великої кількості тестових пакетів різного розміру по протоколу TCP. Сервер тестової програми створює з'єднання із клієнтами по іншу сторону МЕ, моделюючи взаємодію користувачів і адміністраторів з порталом. Після установки з'єднання сервер генерує велику кількість трафіку. Час приходу кожного пакета фіксується клієнтами також, як і час початку й закінчення тесту. Після чого ця інформація пересилається на сервер для наступної обробки.

На рисунку 3.2 і рисунку 3.3 представлені схеми розміщення сервера й клієнтів тестової програми в моделі мережі веб-сайту при проведенні експериментів на швидкість обробки вхідного й вихідного трафіка відповідно.

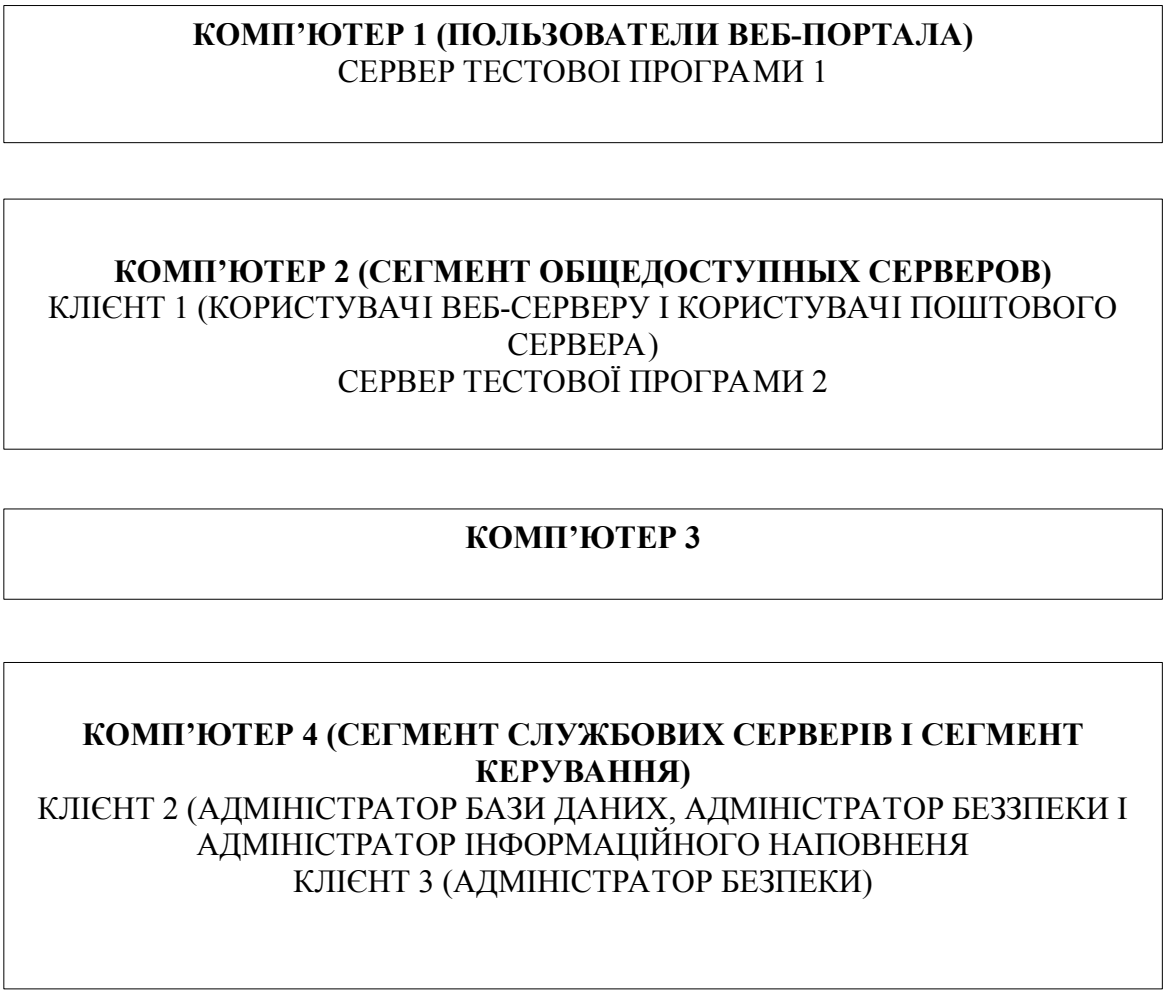

Рисунок 3.2 – Схема розміщення компонентів тестової програми в моделі мережі веб-сайту при проведенні експерименту на швидкість обробки вхідного трафіку

Точність вимірів – характеристика якості виміру, що відбиває ступінь близькості результатів вимірів до дійсного значення вимірюваної величини [14]. Чим менше результат виміру відрізняється від дійсного значення вимірюваної величини, тобто чим менше погрішність, тем вище точність вимірів, незалежно від того чи є погрішність систематичної, випадкової або містить ту або іншу складову.

Вірогідність експериментальних (або вибіркових) оцінок залежить від двох факторів: обсягу вибірок (кількості спостережень) і дисперсії оцінюваної випадкової величини. Очевидно, що для одержання достовірних результатів з певною довірчою ймовірністю p, потрібно провести не менш певної кількості спостережень n.

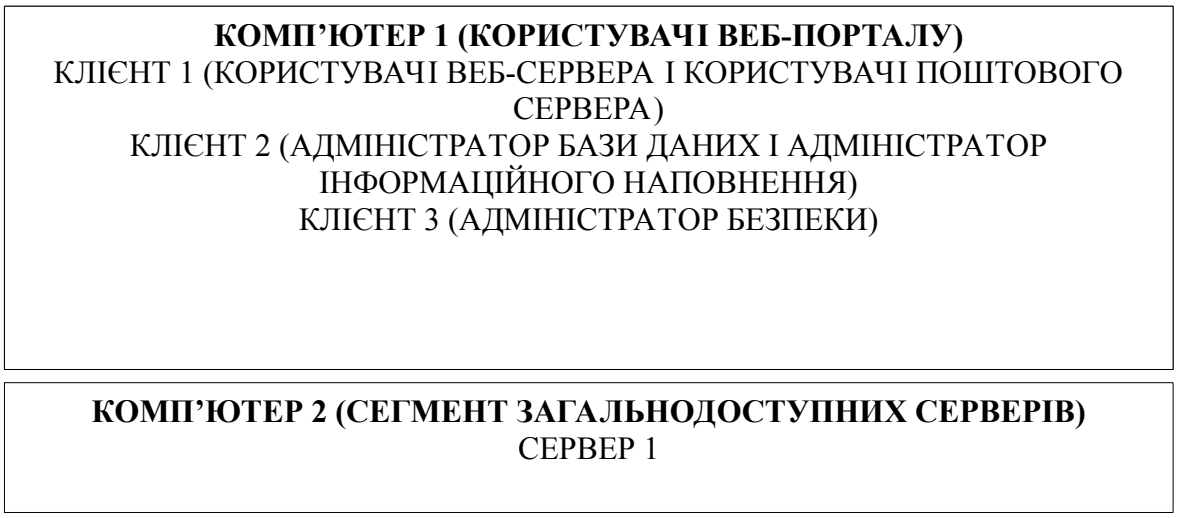

Рисунок 3.3 – Схема розміщення компонентів тестової програми в моделі мережі веб-сайту при проведенні експерименту на швидкість обробки вихідного трафіку

Завдання визначення необхідного числа досвідів n, вирішується з використанням інтервальної оцінки математичного очікування цієї величини і її нормованої форми:

$$u = \frac{\overline{y} - M\{y\}}{\sigma / \sqrt{n}},$$

де y – середнє значення випадкової величини по вибірці.

Інтервальна оцінка для M{y} представлена нерівністю

$$\overline{y} - \frac{u_p \sigma_y}{\sqrt{n}} < M\{y\} < \overline{y} + \frac{u_p \sigma_h}{\sqrt{n}}, \quad (3.1)$$

де Up – табличний квантиль стандартної величини, що відповідає ймовірності p;

M{y} – математичне очікування.

Вибірка має певний розмах значень від лівої границі q_1 до правої границі q_2. Тоді довжина інтервалу значень $L=q_{2-q1}$. Чим більше розмах значень величини, тем менш достовірні й менш точні вибіркові оцінки [25].

Тут знаменник є константою – чим більше інтервал значень L, тем менше точність і більше відносне відхилення ε, тобто значення цієї характеристики назад точності.

Ліву й праву частини будемо розглядати як границі q1 і q2, тоді

$$L = 2\frac{u_p \sigma_y}{\sqrt{n}},$$

а відносна погрішність ε буде $\varepsilon = u / \sqrt{n}$, звідки випливає

$$n \geq \left(\frac{u_p}{\varepsilon} \right)^2$$

Цю формулу можна записати як

$$n \geq \left(\frac{u_p}{u / \sqrt{n}} \right)^2$$

звідки виведено

$$n = (2u_p)^2 \qquad (3.2)$$

Приймемо довірчу ймовірність рівної 0,95, а рівень значимості 0,01. Табличний квантіль U0.95 при цьому рівний 2,629. Тоді відповідно до (3.2) необхідна мінімальна кількість спостережень п складе

$$n = (2 * 2.629)^2 = 28.$$

3.2 Опис використовуваних метрик

У даній роботі експеримент проводиться над моделлю мережі веб-сайту. При цьому для з'ясування характеристик продуктивності досліджуваних типів МЕ використовується вимір по різних прямих метриках:

Середній час проходження IP-пакету до сегмента ДМЗ – демілітаризованої зони по протоколу HTTP:

- середній час проходження IP-пакету до сегмента ДМЗ по поштових протоколах;
- середній час проходження IP-пакету із сегмента ДМЗ у зовнішню мережу по протоколу http;
- середній час проходження IP-пакету із сегмента ДМЗ у зовнішню мережу по поштових протоколах;
- середній час проходження IP-пакету до службового сегмента;
- швидкість обробки вхідного трафіка зовнішнім МЕ являє собою кількість IP-пакетів перевірених на відповідність заданій політиці безпеки за одиницю часу (1 сек.);
- швидкість обробки вихідного трафіка зовнішнім МЕ являє собою кількість IP-пакетів перевірених на відповідність заданій політиці безпеки за одиницю часу (1 сек.);
- швидкість обробки трафіка внутрішнім МЕ являє собою кількість IP-пакетів перевірених на відповідність заданій політиці безпеки за одиницю часу (1 сек.).

На основі результатів вимірів по перерахованих вище метриках для різних категорій користувачів за допомогою методу лінійного регресійного аналізу будуть побудовані емпіричні формули залежності швидкості проходження пакета по протоколу TCP/IP через систему МЕ від використовуваних правил безпеки. Процес підготовки до проведення експериментів починається з підключення

комп'ютерів до тестової мережі, їх настроювання шляхом відключення всіх зайвих служб операційної системи.

Після того як операційні системи комп'ютерів, що приймають участь у тестуванні, і ПЗ, що тестується, підготовлено, за допомогою тестової програми. Створюються два виміри для виміру швидкості вхідного й вихідного трафіка. Завмер для виміру вхідного трафіка містить 153 тестових з'єднання. Вимір для виміру вихідного трафіку включає 150 тестових з'єднань. Їхній опис представлено в таблиці 3.1.

Таблиця 3.1 – Опис тестових з'єднань для алгоритму виміру швидкості роботи із вхідним трафіком

Кількість з'єднань	IP хоста	Порт хоста	Цільовий IP	Цільовий порт
65	192.168.0.1	5000-5064	192.168.0.2	80
65	192.168.0.1	5065-5129	192.168.0.2	8080
10	192.168.0.1	5130-5139	192.168.0.2	25
10	192.168.0.1	5140-5149	192.168.0.2	110
1	192.168.0.1	4444	192.168.0.4	2222
1	192.168.0.1	5555	192.168.0.4	2224
1	192.168.0.1	6666	192.168.0.4	1111

Таблиця 3.2 – Опис тестових з'єднань для алгоритму виміру швидкості роботи з вихідним трафіком

Кількість з'єднань	IP хоста	Порт хоста	Цільовий IP	Цільовий порт
65	192.168.0.2	80	192.168.0.1	5000-5064
65	192.168.0.2	8080	192.168.0.1	5065-5129
10	192.168.0.2	110	192.168.0.1	5140-5149

Після того як тестові виміри створені, проводиться безпосередньо експеримент. При цьому процесам тестової програми привласнюється пріоритет

«Реального часу». Сам процес проведення експерименту полягає в почерговім виконанні вхідних і вихідних тестових вимірів на відповідних комп'ютерах тестової мережі. Правила безпеки тестуємих МЕ попередньо встановлюються відповідно до плану експерименту перед виконанням кожної наступної пари вимірів.

4 ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Опис протоколу взаємодії

При взаємодії клієнта й сервера по протоколу UDP/IP передані повідомлення розбиваються на пакети, які знову збираються на стороні адресата. Формат пакета має такий вигляд: "<index>/<count> <data>", де <index> – номер пакета, <count> – загальна кількість пакетів використовуваних для передачі даних, <data> – частина повідомлення, розміщена в пакеті.

Після одержання пакета адресат відсилає підтвердження у вигляді пакета, що має наступний формат:

"<index> Received", де <index> – номер отриманого пакета.

У випадку якщо частина пакетів не була отримана адресатом, він відправляє один або кілька пакетів наступного формату "<index>/<count> Get" з метою одержання відсутніх для складання всього повідомлення пакетів з даними. Синхронізації часу клієнтів і часу сервера здійснюється по протоколу UDP/IP з використанням наступних команд:

- «Synchronization» – запит сервера на початок синхронізації із клієнтом;
- «Ready» – підтвердження клієнта почати синхронізацію із сервером;
- «<count>» – передача числового значення між клієнтом і сервером (може відбуватися як від клієнта на сервер, так і із сервера на клієнт). Може передаватися кількість циклів синхронізації й показання часу в мілісекундах;
- «<time1>;<time2>» – передача показань годин клієнта (<time1>) при прийманні останнього повідомлення й відправленню поточного повідомлення (<time2>);
- «Synchronized» – підтвердження синхронізації клієнтом.

Підготовка й проведення тестових вимірів швидкості проходження трафіка між сервером і клієнтами здійснюється по протоколу TCP/IP з використанням наступних команд:

- «Gaugetestpreparing» – запит сервера на передачу на клієнт даних про тестові з'єднання;
- «Gaugetestpreparing Ready» – повідомлення про готовність клієнта прийняти дані про тестові з'єднання;
- «<count> <port> <size> <remote port>» – повідомлення, що містить дані про одне тестовім з'єднанні. <count> – кількість тестових пакетів; <port> – номер порту на серверному комп'ютері; <size> – розмір тестового пакета; <remote port> – номер порту на клієнтському комп'ютері;
- «Gaugetestpreparing Portionreceived» – повідомлення клієнта про одержання інформації про одне з'єднанні й готовності прийняти інформацію про наступний;
- «Gaugetestpreparing Datasended» – повідомлення сервера про закінчення передачі інформації про тестові з'єднання;
- «Gaugetestpreparing Datareceived» – підтвердження клієнтом закінчення передачі інформації про тестові з'єднання;
- «Canreceiveconnections» – підтвердження клієнта про готовність установити тестові з'єднання;
- «Measuringrequest» – запит сервера на початок тестування;
- «Measuringready» – підтвердження клієнта про готовність почати тестування;
- «<number>» – повідомлення утримуюче числове значення, наприклад тестовий пакет або результат виміру;
- «Measuringfinished» – підтвердження клієнтом закінчення тестування;
- «Readytogetresults» – повідомлення сервера про готовність прийняти результати вимірів;
- «Measuringcount Portionreceived» – повідомлення серверу про приймання даних про час одержання тестового пакета клієнтом;
- «Sendsumtime Ready» – повідомлення сервера про готовність прийняти загальний час, витрачене на проведення тестового пересилання;

- «Sendsumtime Sending» – підтвердження клієнта переслати загальний час, витрачене на проведення тестового пересилання;
- «Sendsumtime Wait» – повідомлення сервера про перехід у режим очікування пересилання загального часу, витраченого на проведення тестового пересилання;
- «Sendsumtime Received» – підтвердження сервером одержання загального часу проведення тестового пересилання.

У процесі роботи тестової програми, можливе виникнення виняткових ситуацій, викликаних, як правило, порушенням протоколу взаємодії клієнтів і сервера або некоректними діями користувача. Мова програмування C#, з використанням якого розроблена програма має вбудовані засоби обробки виняткових ситуацій, використання яких дозволяє уникнути аварійного завершення програми.

4.2 Загальний алгоритм роботи програми

Як правило, при виникненні виняткової ситуації при порушенні протоколу взаємодії клієнтів і сервера, подальші дії по порушеному протоколу неможливі, тому після перехоплення виключення програма видає користувачеві повідомлення про помилку й припиняє подальше виконання алгоритму, у якім виникла помилка. У результаті користувачеві прийде запустити виконання перерваного алгоритму заново, однак програма уникне аварійного завершення своєї роботи.

При некоректнім користувацькім уведенні, також збуджується виключення, яке перехоплюється за допомогою стандартних засобів мови C#, і користувачеві видається повідомлення, що сповіщає його про некоректне введення, що й пропонує повторити його.

Ініціалізація й проведення вимірів швидкості передачі пакетів по тестовій мережі за допомогою розробленої програми проходить по алгоритму, що включає 7 основних етапів:

- синхронізація часу клієнтів і сервера – на цьому етапі відбувається вимір транспортної затримки й різниці в показаннях годин клієнта й сервера. Після чого на клієнт передається сума цих двох величин, ці дії повторюються для кожного клієнта, зазначеного в серверному списку клієнтів;
- підготовка тестування. На цьому етапі встановлюються сервісні з'єднання між клієнтами й сервером і по них на клієнти передається інформація, необхідна для створення й настроювання тестових з'єднань, наприклад, порт тестового з'єднання на сервері й на клієнтові, розмір тестового пакета і т.д.;
- установка тестових з'єднань. Відбувається створення тестових з'єднань на клієнтах і їх переклад у режим очікування підключення з боку сервера. Після того як з'єднання встановлені з кожним із клієнтів виконується наступний етап алгоритму;
- ініціалізація процесу тестування. На цьому етапі для кожного тестового з'єднання відбувається перевірка готовності почати виміру;
- безпосередньо процес тестування. Відбувається передача тестових пакетів по всіх тестових з'єднаннях з фіксацією часу оправлення на стороні сервера й часу одержання пакета на стороні клієнтів;
- обробка результатів. На цьому етапі клієнти пересилають на сервер часи одержання тестових пакетів і загальний час проведення тестування. На сервері ці дані сортуються по групах тестових з'єднань, що імітують різні категорії користувачів, обробляються й зберігаються у файли формату XML;
- розрив тестових з'єднань. На цьому етапі всі тестові з'єднання розриваються, і порти на сервері й клієнтських комп'ютерах звільняються для подальшого використання.

Алгоритм роботи сервера тестової програми є частиною загального алгоритму роботи й включає ті ж етапи що й загальний алгоритм.

На етапі синхронізації часу сервер спочатку обновляє свої настроювання, після чого з настроювань зчитується список клієнтів сервера. Далі відбувається обхід по всіх елементах цього списку й для кожного клієнта відбувається синхронізація часу у відповідність із протоколом синхронізації часу.

Синхронізація з конкретним клієнтом починається з ініціалізації необхідних для взаємодії по протоколу UDP/IP об'єктів і відсилання повідомлення про початок синхронізації. Після одержання відповідного повідомлення клієнтові передається кількість циклів синхронізації й починається з'ясування різниці в показаннях годин клієнта й сервера. У циклі сервер фіксує свої показання годин при відсиланні й прийманні повідомлення від клієнта. Клієнт, у свою чергу фіксує й передає на сервер показання своїх годин при прийманні повідомлення від сервера й відсилання йому нового повідомлення. На основі цих даних сервер розраховує різницю в часі із клієнтом. Після виконання всіх циклів синхронізації із клієнтом сервер розраховує середнє по всіх результатах і посилає його клієнтові як значення, яке клієнт повинен використовувати для корекції часу приймання тестових пакетів при проведенні тестування. Після приймання повідомлення, що підтверджує закінчення синхронізації, сервер розриває з'єднання й звільняє об'єкти, які використовувалися для зв'язку із клієнтом.

Перед тем як проводити тестування необхідно сповістити клієнта про те, скільки тестових з'єднань і на яких портах буде задіяне для вимірів. Для цього в алгоритмі тестування передбачений етап підготовки тестування.

Етап підготовки тестування містить у собі основний цикл, у якім відбувається обхід по всіх елементах списку клієнтів, установка сервісних з'єднань із клієнтами й безпосередньо пересилання даних про тестові з'єднання. Пересилання даних про з'єднання починається з відсилання сервером ініціюючого повідомлення й очікування відповідного підтвердження від клієнта. Після того як підтвердження отримане, інформація про кожне з'єднання відправляється в циклі на клієнт. Інформація про наступне з'єднання відправляється тільки після підтвердження попереднього. Після того як увесь цикл пройдений сервер

відправляє повідомлення про завершення передачі даних і очікує відповідного підтвердження від клієнта.

Перед проведенням безпосередньо вимірів по протоколу TCP/IP потрібно встановити тестові з'єднання із клієнтами. У програмі вони встановлюються в багатопоточному режимі, де кожна пара потоків клієнта й сервера відповідає за установку одного тестового з'єднання.

Алгоритм починається з того що сервер створює й запускає потоки, завданням яких є очікування готовності клієнтами прийняти з'єднання. Після того як усі клієнти підтвердили таку готовність сервер обходить по циклу всі вилучені адреси тестових з'єднань. У цьому циклі виконуються вкладені цикли, що безпосередньо прив'язують тестові сокети до відповідних до портів на сервері, що й установлюють з'єднання з вилученими адресами.

На етапі обробки результатів тестування сервер спочатку одержує інформацію про загальний час виконання тесту від усіх клієнтів і на підставі цієї інформації й даних про кількість тестових пакетів розраховує пропускну здатність для кожного клієнта. Коли ця робота завершена, сервер зберігає її результати в XML файл і приступає до одержання даних про час прибуття кожного тестового пакета на клієнт. Ця інформація використовується для розрахунку мінімального, максимального й середнього часу проходження пакетів для кожного тестового з'єднання й для кожної категорії тестових з'єднань у цілому. Алгоритм завершується формуванням звітів і збереженням їх в XML файли.

Алгоритм виконання тестових вимірів програмою є багатопоточним. Кількість потоків дорівнює кількості тестових з'єднань. Виконання кожного потоку починається з генерації масиву тестових пакетів у циклі. Після цього виконується їхнє відсилання клієнтові з одночасною фіксацією часу відправлення кожного пакета. Після того як усі пакети були передані й від клієнта отримане повідомлення про коректне завершення процесу тестування, тестовий потік завершує свою роботу. Після завершення всіх потоків сервер переходить до виконання наступного етапу загального алгоритму.

4.3 Алгоритм роботи клієнтської частини

Алгоритм роботи клієнта тестової програми є частиною загального алгоритму роботи й включає ті ж етапи що й загальний алгоритм.

Синхронізація із сервером починається з ініціалізації необхідних для взаємодії по протоколу UDP/IP об'єктів і приймання повідомлення про початок синхронізації. Після відсилання відповідного повідомлення серверу, клієнт ухвалює кількість циклів синхронізації й починається з'ясування різниці в показаннях годин клієнта й сервера. У циклі клієнт фіксує свої показання годин при прийманні повідомлення від сервера й відсиланню повідомлення із цими показаннями серверу. Уся обробка цих значень проводиться сервером.

Після виконання всіх циклів синхронізації клієнт ухвалює розраховане сервером значення, яке клієнт повинен використовувати для корекції часу приймання тестових пакетів при проведенні тестування.

Підготовка до тестування на клієнтові починається із приймання ініціюючого повідомлення із сервера, після чого клієнт відсилає повідомлення підтверджувальне готовність ухвалювати інформацію про тестові з'єднання. Ця інформація ухвалюється в циклі. Після приймання кожного повідомлення з даними серверу посилає повідомлення підтверджувальне приймання. У випадку якщо наступне отримане повідомлення є повідомленням про завершення передачі, то клієнт посилає підтверджувальне повідомлення й переходить до наступного етапу алгоритму.

Алгоритм установки тестових з'єднань на клієнтові починається з обходу всіх портів, що приймають участь у тестуванні, і створення робочого потоку для кожного з них. У кожному із цих потоків відбувається переклад відповідного сокету в стан очікування підключень і приймання одного або декількох тестових з'єднань (залежно від того скільки тестових з'єднань прив'язане до кожного порту на клієнтові).

Після того як усі потоки створені клієнт посилає серверу повідомлення про готовність прийняти тестові з'єднання й очікує поки всі потоки, що встановлюють ці з'єднання, завершать своє виконання.

Клієнтська частина алгоритму тестових вимірів починається з фіксації поточних показань часу, необхідних для розрахунку пропускної здатності з'єднання із сервером. Після цього відбувається обхід по всіх елементах списку клієнтських портів, задіяних у тесті. Для кожного з них, у внутрішньому циклі створюються й запускаються потоки, які виконують приймання тестових пакетів для кожного тестового з'єднання. Після того як усі тестові потоки завершили свою роботу показання часу фіксуються повторно й шляхом обчислення різниці часу закінчення й часу початку тестування обчислюється загальний час тестування.

Алгоритм роботи тестового потоку містить у собі цикл приймання тестових пакетів з фіксацією часу їх одержання, цикл корекції зафіксованих показань часу на раніше отримане значення різниці в часі сервера й клієнта й відправлення повідомлення з повідомленням про вдале завершення тестування на сервер.

Завдання клієнта при виконанні етапу обробки результатів тестування полягає в пересиланні результатів тестування на сервер. Алгоритм підрозділяється на 2 етапу – відсилання загального часу виконання тестів і відсилання показань часу під час приймання кожного тестового пакета.

4.4 Загальна архітектура програми

Програма повинна забезпечувати можливість виконання перерахованих нижче функцій:

- паралельна робота з декількома клієнтами, що працюють на різних комп'ютерах тестуємої мережі;
- можливість редагування списку клієнтів (зміна IP-адрес і портів);

- можливість синхронізації показань часу клієнтів із часом сервера перед проведенням вимірів;
- можливість паралельного тестування в багатопочному режимі, де кожний потік відповідає з'єднанню з одного порту сервера до одному або декільком портам одного або декількох клієнтів;
- можливість створення й редагування списку тестових з'єднань для багатопоточного режиму включаючи можливість створення нових крапок виміру по одній або створення групи однотипних крапок з'єднання, що відрізняються номерами портів на сервері, а також розміром і кількістю тестових пакетів, можливість зміни кількості й розміру тестових пакетів для вже існуючих з'єднань;
- можливість зміни порту, на якім працює серверна частина програми;
- можливість збереження у файл тестового виміру, що включає ім'я виміру й список тестових з'єднань, і наступного завантаження виміру з файлу;
- можливість проведення іншого тестового виміру з іншим набором тестових з'єднань без перезапуску програми;
- можливість вибору тестового виміру зі списку доступних вимірів;
- первинна обробка результатів тестування, що включає знаходження середніх, максимальних і мінімальних значень часу проходження пакетів по мережі для конкретного тестового з'єднання, а також для всіх тестових з'єднань із одним клієнтом;
- можливість збереження результатів тестування у файл формату XML.

Архітектура програмного забезпечення для тестування продуктивності мережевих екранів – це базова організація системи, втілена в її компонентах, їх відносинах між собою й з оточенням, а також принципи, що визначають проектування й розвиток системи.

При розробці тестової програми, яка використовується при проведенні експериментального дослідження, застосовуються методи об'єктно-орієнтованого програмування. Використана мова програмування C# є повністю об'єктно-орієнтованим.

Архітектура програми спроектована з використанням парадигми «4 + 1». Ця парадигма описує систему за допомогою чотирьох вистав: фізичне, логічне, вистава процесів системи, вистава сценаріїв використання системи й вистава реалізації. Усі ці вистави можуть бути зображені у вигляді відповідних UML діаграм.

Розроблена програмна система має клієнт серверну архітектуру, яка може бути представлена за допомогою діаграми розгортання. У загальному випадку серверна й клієнтські частини програми запускаються на різних комп'ютерах і працюють під управлінням середовища виконання .NET. Фізичне представлення програмної системи у вигляді діаграми розгортання презентовано на рис. 4.1.

Представлення процесів використовується для відображення процесів системи і їх взаємодії. Основним процесом розробленої програмної системи є процес тестування, який охоплює всі основні функції програми. Основними об'єктами, що приймають участь в цьому процесі, є сервер, клієнти і їх програмні мережні інтерфейси.

Представлення сценаріїв використання системи відбиває загальні вимоги до поведінки системи по взаємодії з користувачем і іншими системами.

Це описує реалізовані функціональні компоненти системи й може бути зображене у вигляді UML-діаграми пакетів.

класи використовують для своєї роботи об'єкти інших класів, то між пакетами можуть існувати залежності – спрямовані відносини, які йдуть від залежного пакета до пакета, який використовується залежним у своїй роботі. Вистава реалізації системи у вигляді діаграми пакетів презентовано на рис. 4.2.

Це представлення може бути зображене з використанням діаграми прецедентів. Представлення сценаріїв використання розробленої програмної системи зображене на рисунку 4.1.

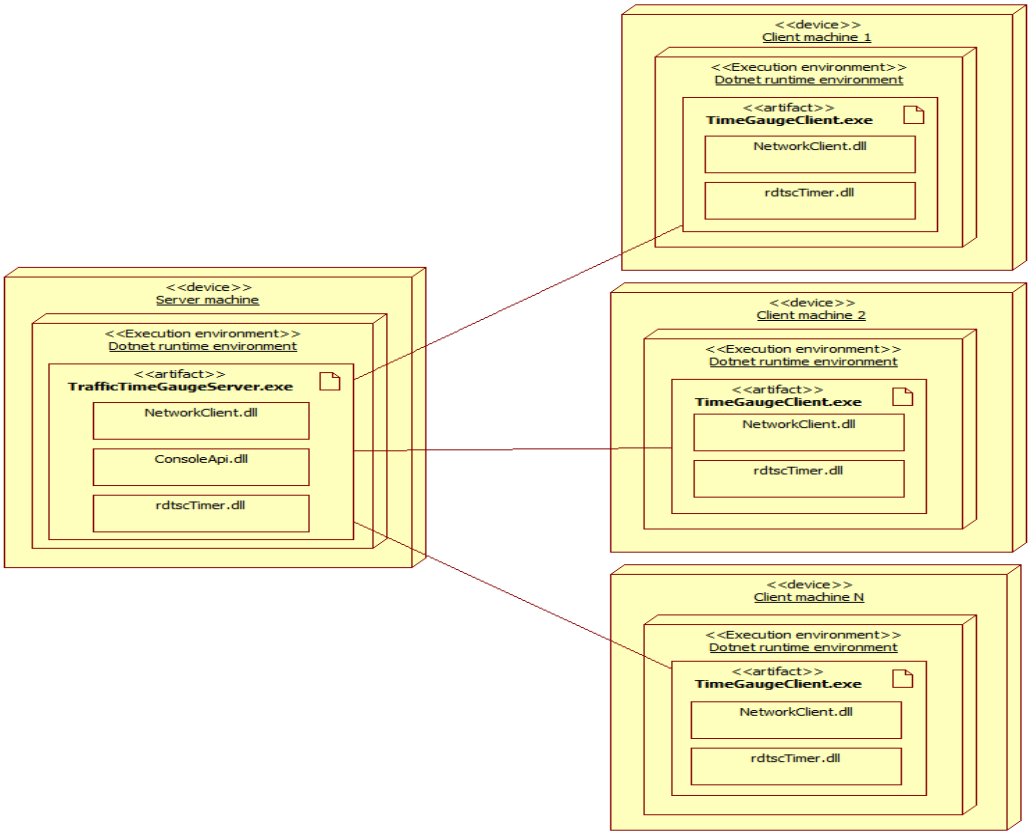

Рисунок 4.1 – Діаграма розгортання програмної системи

Пакети являють собою функціонально відособлені набори класів. Оскільки

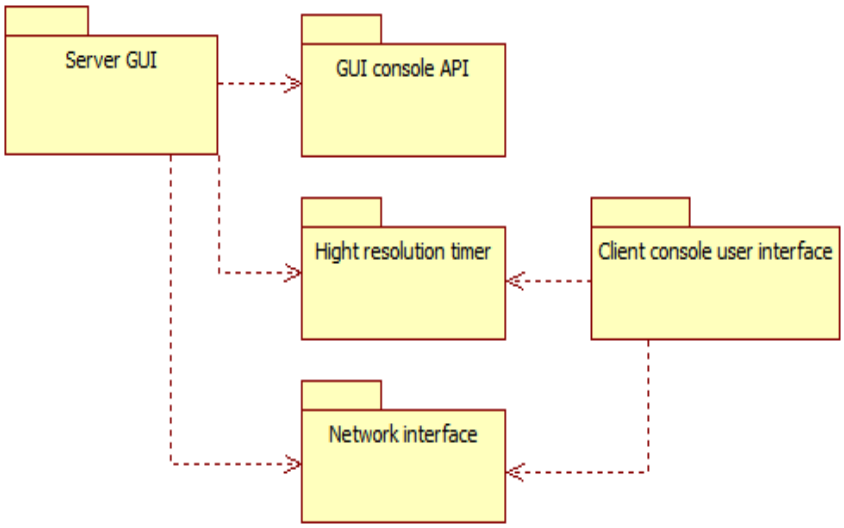

Рисунок 4.2 – Діаграма пакетів програмної системи

Рисунок 4.3 – Діаграма прецедентів програмної системи

Логічне представлення системи визначає зв'язку залежностей між системними елементами й може бути презентовано у вигляді діаграм класів. У цілому програма реалізовано у вигляді трьох модулів, що реалізують функції сервера, клієнта, а також програмного інтерфейсу для мережної взаємодії між ними. Кожний модуль містить свій набір класів (рисунок 4.4).

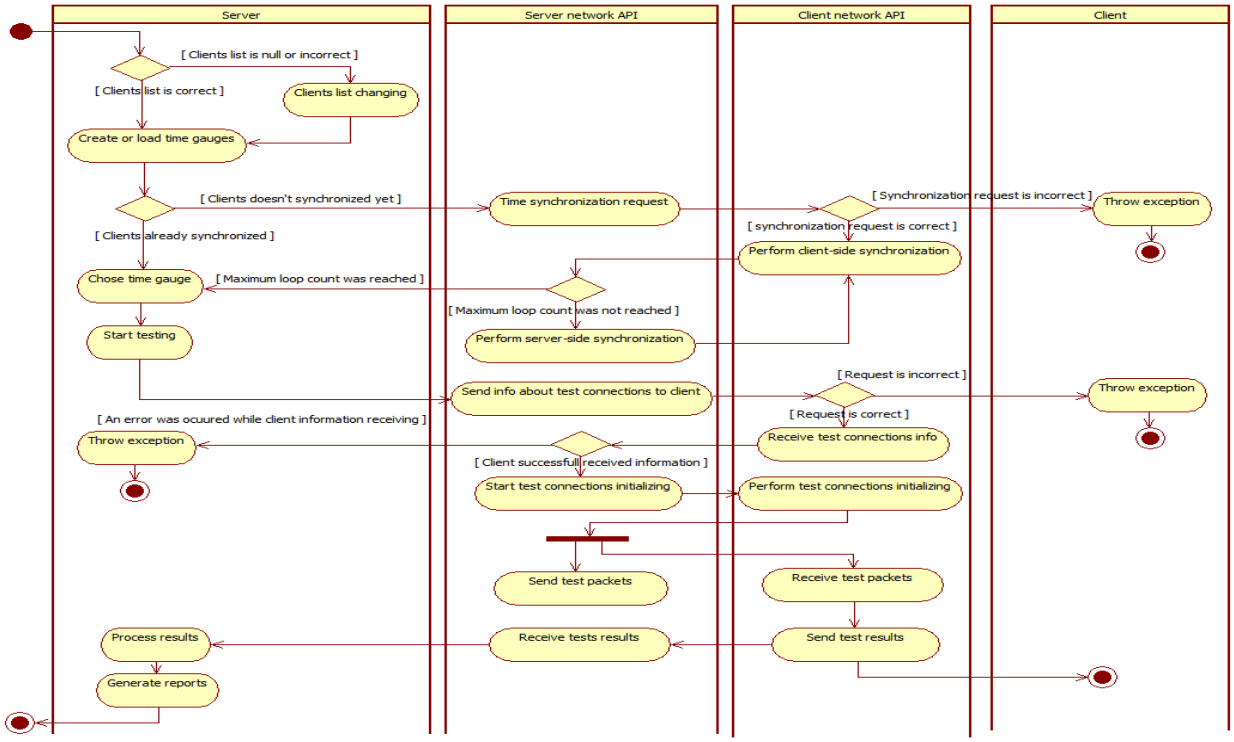

Рисунок 4.4 – Діаграма активностей програмної системи

Модуль, що забезпечує програмний інтерфейс взаємодії клієнтів і сервера реалізований у вигляді Clr-Сумісної бібліотеки класів `Networkclient.dll`, що включає 4 класу, один з яких абстрактний, і 3 перерахування. Діаграма класів цієї бібліотеки представлено на рисунку 4.5.

Клас Worker надає функціональність для взаємодії клієнта й сервера з використанням одного мережного з'єднання. Він є абстрактним і містить наступні методи: Free – метод, що розриває з'єднання й звільняючий використовуваний їм сокет. Connect – установлює з'єднання з вилученим хостом. Цей метод має перевантаження в параметрах якої можна вказати вилучена адреса з яким потрібно встановити з'єднання. При використанні перевантаження без параметрів з'єднання встановлюється з адресою, переданим у якості параметра конструктора одного із класів спадкоємців.

Метод Bind прив'язує сокет, використовуваний об'єктом для мережної взаємодії до порту, зазначеного в параметрах конструктора одного із класів спадкоємців.

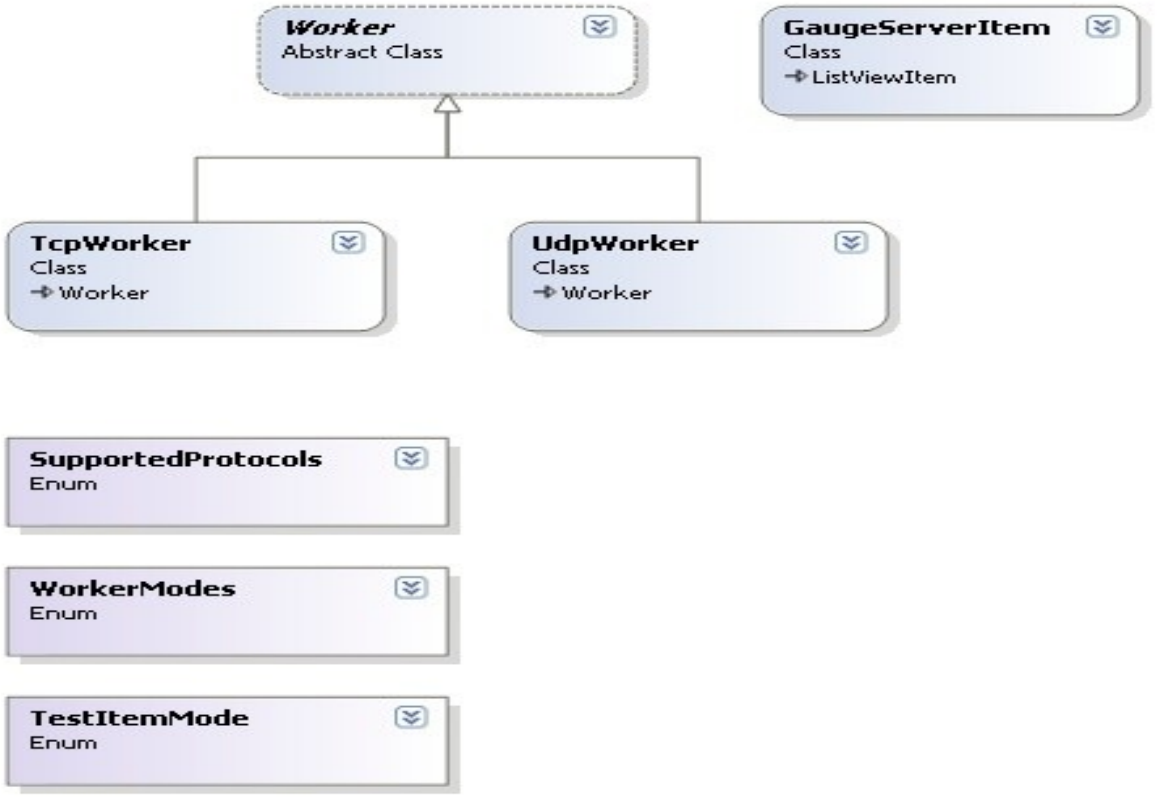

Рисунок 4.5 – Діаграма класів бібліотеки Networkclient.dll

Крім цих методів клас Worker визначає кілька закритих об'єктів, використовуваних класами спадкоємцями.

Клас Udpworker є спадкоємцем класу Worker і, крім успадкованих, визначає також кілька власних відкритих і закритих методів:

- Senddata – відкритий метод, використовуваний для відсилання масиву байтів з використанням UDP сонета;
- Receivedata – відкритий метод, що переводить використовуваний UDP сокет у режим приймання даних, і, у випадку якщо дані отриман, що повертає їх у вигляді масиву байтів у крапку виклику;
- sendpacketconfirm – закритий метод, що ухвалює масив байтів, що містить прийнятий пакет, у якості вхідного параметра, що й відсилає підтвердження одержання цього пакета;

- removeudppacketheader – закритий метод, що ухвалює отриманий пакет як вхідний параметр, що видаляє з пакета заголовну частину, що й повертає, що втримуються в пакеті дані у вигляді масиву байтів;

- sendpacket – закритий метод, що ухвалює масив байтів і індекс пакета як вхідних параметрів, що формує пакет із заданим індексом, що й відсилає його адресатові.

- receivepacket – закритий метод для одержання UDP пакета. Він ухвалює рядок з очікуваним заголовком пакета, а також кількість байт, починаючи з нульового, серед яких цей заголовок слід шукати. У випадку якщо заголовок отриманого пакета не відповідає очікуваному, метод відсилає запит на повторне відправлення необхідного пакету.

Залежно від цілей створення об'єкта Tcpworker він може працювати в чотирьох режимах: тестового клієнта, тестового сервера, клієнтського й серверного сервісних з'єднань. Залежно від режиму ті або інші методи будуть недоступні (їх виклик буде викликати виключення).

Клас `Gaugeserveritem` представляє тестове з'єднання й інкапсулює усі його характеристики, а також Tcpworker використовуваний для взаємодії клієнта й сервера при проведенні тесту. Він є спадкоємцем Listviewitem що дозволяє використовувати його в елементі керування `Listview` для відображення характеристик тестового з'єднання. Для цієї мети використовуються наступні загальнодоступні властивості Listviewitem:

- Localport – містить номер порту, пов'язаного з тестовим з'єднанням на сервері;

- Category – містить назва категорії, до якої належить тестове з'єднання;

- Supportedprotocols – містить назва протоколу, по якім працює тестове з'єднання;

- Remoteip – IP адреса комп'ютера, на якій працює клієнт;

- Remoteport – порт, на якім працює клієнт;

- Packetsize – розмір тестового пакета, використовуваний з'єднанням;

- Packetscount – кількість пакетів, що відсилаються клієнтові під час тестування.

Для опису одного тестового з'єднання необхідно 2 об'єкта класу Listviewitem – один на сервері й один на клієнтові. Відповідно його об'єкти можуть працювати в одному із двох режимів – серверному або клієнтському. Частина нижчеперелічених методів класу Listviewitem доступна тільки в одному з них:

- Setlistener – відкритий метод, що ухвалює об'єкт класу Tcpworker, який буде використовуватися для приймання вхідного запиту на підключення;
- Initinnervariables – закритий метод, який використовується в різних перевантаженнях конструктора для ініціалізації закритих полів класу;
- Connectclientpoint – відкритий метод, що з'єднує тестове з'єднання з вилученим клієнтом. Логічний параметр вказує, чи вимагається прив'язати з'єднання до порту або це вже зроблене;
- Makeserversidetestinit – відкритий метод, що виконує серверну частину роботи з ініціалізації тестового з'єднання перед початком тесту;
- Makeserversidetestactions – відкритий метод виконуючий пересилання тестових пакетів клієнтові про спеціально створений потік.

Для того щоб процедура верифікації була повною, необхідно охопити такі області, що як виконується код клієнтської й серверної частин, а також бібліотеки класів, використовуваної для мережної взаємодії між ними.

Критерієм успішності проходження тестів клієнтської й серверної частин є коректність тестуємих функцій, стійкість до помилок користувача, а також працездатність ПС у заданих умовах. Для бібліотеки класів, що реалізує мережний інтерфейс для клієнта й сервера, такими критеріями є наявність коректної взаємодії між частинами програми по мережі, за умови правильного використання класів бібліотеки, також стійкість програмної системи до фізичних проблем мережі.

Серверна частина додатка забезпечує користувача графічним віконним інтерфейсом і дає йому можливість створити, настроїти й провести тестування. Серверна частина використовує у своїй роботі бібліотеку класів Networkclient.dll, а також власні класи для синхронізації часу із клієнтами, обробки результатів тестування й класи вікон додатка. Вихідний код модуля, що реалізує серверну частину додатка, представлений у додатку А. Усього серверна частина складається з п'ятнадцяти класів і одного перерахування. Діаграма класів серверної частини тестової програми представлено на рисунку 4.6.

Статичний клас Program є головним класом серверної частини додатка й містить єдиний статичний метод Main, що представляє собою крапку входу в програму.

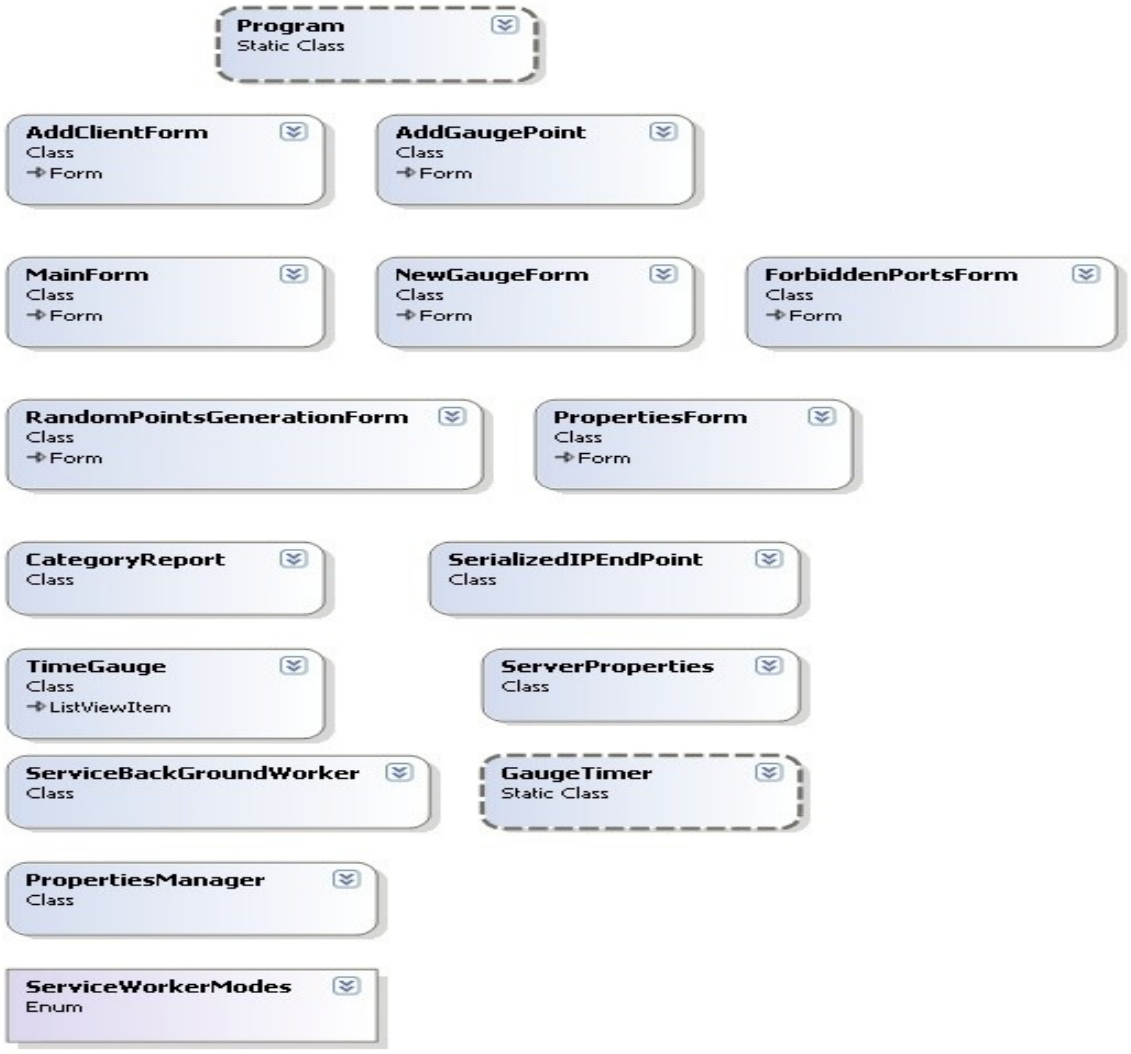

Рисунок 4.6 – Діаграма класів серверної частини тестової програми

Клас Mainform представляє головну форму програми. Він містить оброблювач події Load, у якім відбувається завантаження настроювань програми, а також оброблювачі подій клацання по кнопках tsbtnsyncronize_Click, tsbtnproperties_Click, tsbtnnewgauge_Click, btnstartgauge_Click і закритий метод synchronizeclient, що робить синхронізацію із клієнтом по його IP адресі й сервісному порту. Також клас містить список доступних тестових вимірів і спеціальний об'єкт, що містить настроювання програми як загальнодоступних полів.

Клас Newgaugeform це клас форми для створення нового тесту. Він дозволяє сформувати список тестових з'єднань, а також завантажити тест із файлу або зберегти його у файл. Клас містить наступні методибоброблювачі подій:

- btncancel_Click – оброблювач події Click кнопки btncancel. Скасовує додавання нового тестового виміру, закриває поточну форму й робить активної головну форму додатка;
- btncreategauge_Click – оброблювач події Click кнопки btncreategauge. Створює новий об'єкт тестового виміру, закриває поточну форму й робить активної головну форму додатка. Якщо необхідні для створення виміру дані відсутні на формі, видає повідомлення про помилку й припиняє своє виконання4
- Addgaugepoint_Click – оброблювач події Click кнопки Addgaugepoint. Відкриває форму Addgaugepointform, що дозволяє додати нове тестове з'єднання до списку тестових з'єднань виміру;
- btndeletegaugepoint_Click – оброблювач події Click кнопки btndeletegaugepoint. Видаляє обране в списку тестових з'єднань з'єднання з виміру;
- btngeneraterandompoints_Click – оброблювач події Click кнопки btngeneraterandompoints. Відкриває форму Randompointsgenerationform, що дозволяє згенерувати довільна кількість (але не більше чому 65535) однотипних тестових з'єднань;
- btnforbiddenportschange_Click – оброблювач події Click кнопки btnforbiddenportschange. Відкриває форму Forbiddenportsform, яка дозволяє

редагувати список заборонених портів, які не можна використовувати в тестових з'єднаннях;

- btnsavegauge_Click – оброблювач події Click кнопки btnsavegauge. Відкриває діалог збереження файлу, що дозволяє зберегти тестовий вимір у файл;
- btnloadgauge_Click – оброблювач події Click кнопки btnloadgauge. Відкриває діалог завантаження файлу, що дозволяє завантажити тесовий вимір з файлу.

Для збереження й завантаження тестових вимірів застосовується бінарна серіалізація й десеріалізація об'єкта виміру.

Крім методів оброблювачів клас Newgaugeform також має два закриті методи: fillpointslist і Loadports. Перший заповнює список тестових з'єднань форми інформацією про тестові з'єднання об'єкта виміру, а другий повертає масив заборонених до використання під час тестів портів, завантажений з файлу, назва якого передається методу в якості вхідного параметра.

Клас Addclientform являє собою клас форми, призначеної для додавання інформації про клієнта в настроювання сервера. Він містить у собі два методи оброблювача подій: btnok_Click і tbclientipaddress_Validating. Перший є оброблювачем події Click кнопки btnok, за допомогою якої створюється новий об'єкт, що містить адреса клієнта (, що включає IP адреса й порт), а другий є оброблювачем події Validating текстового поля tbclientipaddress, за допомогою якого перевіряється коректність уведеного користувачем IP адреси. Клас також має одне загальнодоступне поле, що містить повну адресу нового клієнта (, що включає IP адреса й порт).

Клієнтська частина додатка реалізована у вигляді консольного додатка, що дозволяє користувачеві управляти тільки одним параметром – IP адресою сервера.

До складу клієнтської частини входить шість класів, один з яких абстрактний. Діаграма класів клієнтської частини представлено на рисунку 4.7.

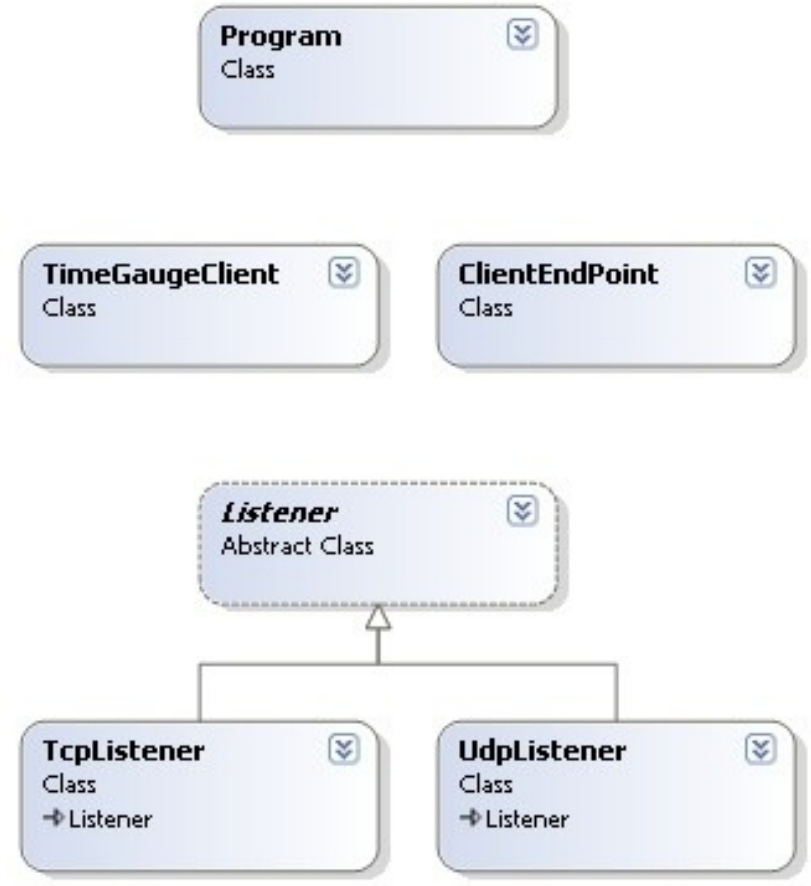

Рисунок 4.7 – Діаграма класів клієнтської частини

Клас Program є головним класом клієнтської частини й містить єдиний статичний метод Main, який є крапкою входу в додаток.

Клас Timegaugeclient представляє тестовий вимір на стороні клієнта, тобто виконує функції аналогічні серверному класу Timegauge. Він містить як внутрішні поля список тестових вимірів, об'єкт Tcpworker, використовуваний для організації сервісного з'єднання, і трохи внутрішніх змінних для розрахунку загального часу, витраченого на проведення тесту. Крім цього до складу класу входять кілька відкритих і закритих методів:

- Gettickcount – закритий методооболонка Winapi функції Gettickcount;
- starttimemeasuring – закритий метод, що зчитує поточні показання годин клієнта, що й зберігає їх у спеціальному внутрішньому полі;

- getelapsedsecondscount – закритий метод, що зупиняє вимір часу, що й повертає кількість секунд, що пройшли з моменту виклику методу starttimemeasuring у точку виклику;

- Acceptconnections – відкритий метод, що виконує роботу із приймання тестових з'єднань на клієнтові;

- Initializetesting – відкритий метод, що виконує клієнтську частину роботи з ініціалізації тестових з'єднань;

- Maketesting – відкритий метод, що запускає тестові потоки, що проводять тестування, на виконання;

- Makeresultssending – відкритий метод, що передає результати тестування на сервер;

- Jointests – відкритий метод, що зупиняє виконання програми до завершення роботи тестових потоків;

- Disconnecttestpoints – відкритий метод, що розриває всі з'єднання;

- sendconnectionspreparing – закритий метод, що відсилає підтвердження готовності клієнтом прийняти тестові з'єднання;

Клас Clientendpoint представляє адресу тестового з'єднання на клієнтові. Він використовується для установки й керування тестовими з'єднаннями, прив'язаними до певного порту на клієнтові. Крім закритих полів він надає доступ до свого списку тестових з'єднань через відкриту властивість, а також має ознака установки тестових з'єднань у вигляді відкритого логічного поля. Також клас включає кілька відкритих і закритих методів:

- Listen – відкритий метод, що переводить внутрішній об'єкт Tcplistener у стан прослуховування запитів на вхідні підключення;

- Acceptasync – відкритий метод, що запускає внутрішній робітник потік, призначений для приймання тестових з'єднань, на виконання;

- Performtestsinitialization – відкритий метод, що виконує клієнтську частину роботи з ініціалізації тестових з'єднань, пов'язаних з певним портом на клієнтові;

- Performtests – відкритий метод, що виконує роботу із проведення тестування з використанням своїх тестових з'єднань на клієнтові;

- Sendtestingresults – відкритий метод, що передає результати тестування серверу для тестових з'єднань пов'язаних з певним портом на клієнтові;

- Jointests – відкритий метод, що зупиняє виконання програми до завершення роботи тестових потоків, пов'язаних з певним портом на клієнтові;

- startaccepting – закритий метод, що є робочим методом потоку, що ухвалює тестові з'єднання;

- Disconnect – відкритий метод, що розриває тестові з'єднання, пов'язані з певним портом на клієнтові.

Абстрактний клас Listener є батьком класів Tcplistener і Udplistener, призначених для керування вхідними підключеннями. Крім внутрішніх полів, що містять об'єкти для прослуховування й взаємодії через вхідні підключення, він містить два абстрактних методів Creategaugeitems і Listen. Перший з них не має параметрів і повертає список об'єктів тестових з'єднань (Gaugeserveritem), отриманий від сервера по сервіснім з'єднанню, а другий переводить внутрішній об'єкт Tcpworker у стан прослуховування запитів на вхідні підключення.

Клас Udplistener є спадкоємцем класу Listener і, крім успадкованих методів, має також власний метод Synchronize, призначений для виконання роботи із синхронізації часу клієнта й сервера на стороні клієнта.

Клас Tcplistener також успадкований від класу Listener і крім його методів має також ряд власних:

- Waitforconnectionconnection – відкритий метод, що зупиняє виконання програми до одержання запиту на підключення з боку сервера.

- Accept – відкритий метод, що виконує дії аналогічні попередньому, але, що не переводить з'єднання в стан прослуховування перед цим.

- Free – Метод, що розриває внутрішньо з'єднання із сервером.

Крім того, клас Tcplistener надає доступ до внутрішнього об'єкта Tcpworker через відкриту властивість.

5 ОПИС МОЖЛИВОСТІ ВИКОРИСТАННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Для початку роботи в додатку потрібно запустити файл TrafficManager.exe. Додаток рекомендується запускати на сервері TMG, тому що на інших комп'ютерах можливі проблеми із завантаженням службових бібліотек.

Для початку роботи із програмою потрібно запустити файл TrafficManager.exe. На екран виведеться вікно авторизації (рисунок 5.1).

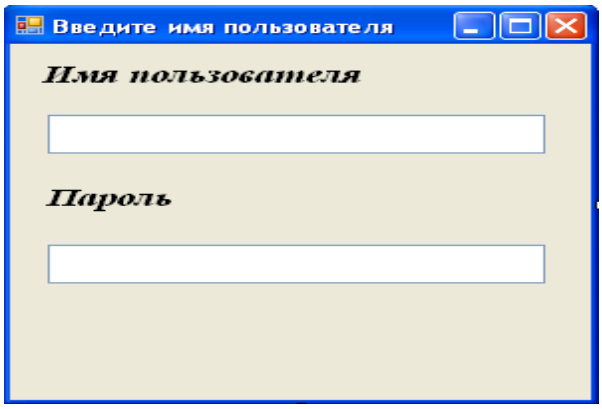

Рисунок 5.1 – Вікно авторизації додатку

Якщо перевірка завершилася успішно, на екран виводиться головна форма додатка, представлена на рисунку 5.2.

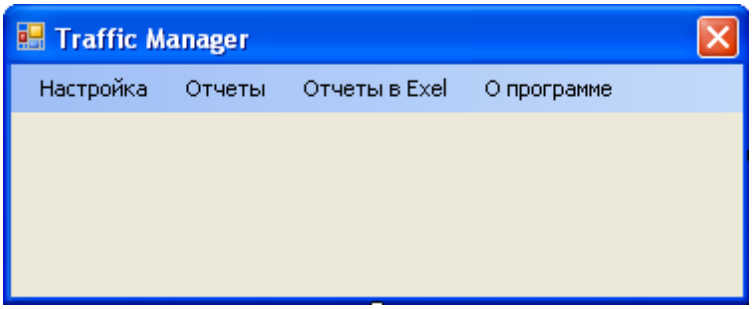

Рисунок 5.2 – Головна форма додатка

Керування правами користувача здійснюється у формі «Керування користувачами» у пункті меню «Настроювання» (рисунок 5.3). Користувач може бути або адміністратором з доступом до розділу з настроюванням міжмережевого екрану, спостерігачем, що може тільки робити моніторинг або звичайним користувачем.

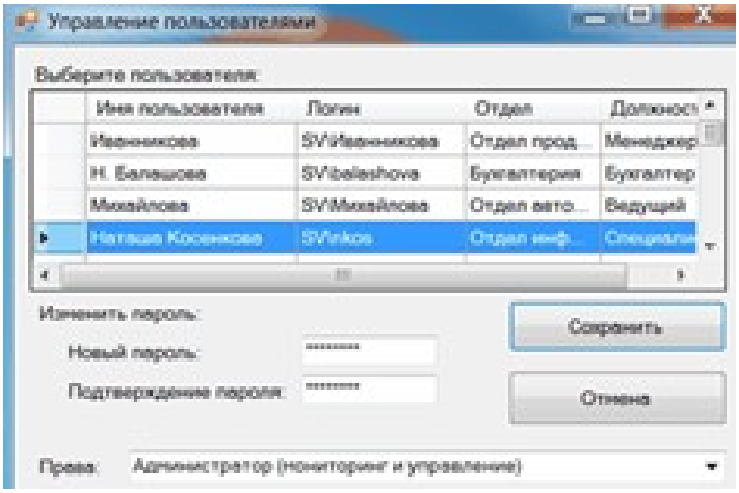

Рисунок 5.3 – Форма додатка для керування користувачами

Параметри з'єднання із сервером баз даних і міжмережевим екраном визначаються у формі «Настроювання з'єднання» пункту меню «Настроювання» (рисунок 5.4).

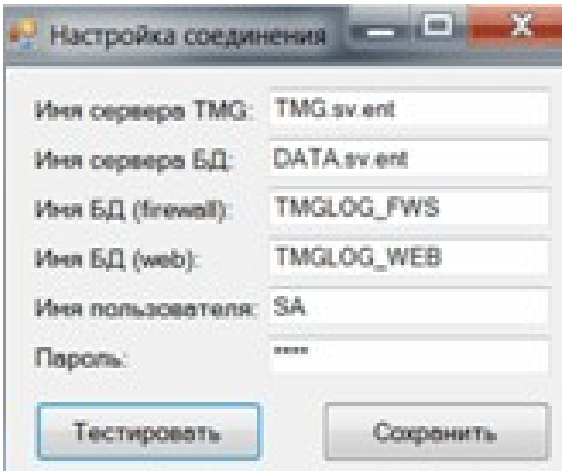

Рисунок 5.4 – Форма додатка для керування користувачами

Особливість програми, що дозволяє збільшити гнучкість створення звітів - формування зведеній таблиці. Для того щоб запустити один з таких звітів, потрібно вибрати «Звіти» – «Використання HTTP-трафіку за годину».

Таблиця 5.1 – Вхідні значення для тестового прикладу

Ім'я	Значення
IP адреса й порт першого клієнта:	192.168.0.2:2000
IP адреса й порт другого клієнта:	192.168.0.3:2000
Кількість тестових з'єднань першого клієнта (192.168.0.2):	10
Кількість тестових з'єднань другого клієнта (192.168.0.3):	3
Вилучений порт для тестових з'єднань першого клієнта:	3000
Вилучений порт для тестових з'єднань другого клієнта:	3000
Порти для тестових з'єднань першого клієнта:	5001-5010
Порти для тестових з'єднань другого клієнта:	5011-5013

Вихідні значення для тестового прикладу представлено в таблиці 5.2.

Таблиця 5.2 – Вихідні значення для тестового прикладу

	Очікувані	Отримані	Результат
Кількість елементів у списку тестових з'єднань серверного вікна «Створення нового виміру» після видалення з'єднання з портом 5013	12	12	Пройдений
Кількість елементів у списку тестових з'єднань серверного вікна «Створення нового виміру»	11	11	Пройдений

Підсумки:

- загальна кількість виконаних тестових прикладів: 5;
- кількість успішна пройдених тестових прикладів: 5;
- кількість неуспішна пройдених тестових прикладів: 0;
- загальна кількість перевірених вихідних значень: 44;
- кількість вихідних значень, у яких очікуване значення не збіглося з реальним: 0.

При проведенні експерименту тестова програма автоматично виконує таку частину первинної обробки даних як одержання мінімальних, максимальних і середніх значень інтервалів часу проходження пакетів для категорій тестових з'єднань, а також показників пропускної здатності мережі.

Загальна структура клієнтських і серверних алгоритмів ідентична, тому що в процесі роботи клієнти й сервер, по суті, будуть виконувати один загальний алгоритм, взаємодіючи по заздалегідь певному протоколу.

Відмінності між структурою клієнтських і серверних алгоритмів в основному обумовлені необхідністю сервера паралельно працювати з декількома клієнтами й обробляти результати вимірів.

Алгоритми клієнтської й серверної частин використовують протоколи TCP/IP і UDP/IP для взаємодії один з одним

Для проведення експерименту потрібна наявність чотирьох ПК, із операційною системою Windows, а також два програмні МЕ, поштовий сервер, тестова програма й ПС, призначене для виміру продуктивності мережі.

Архітектура програми для виміру продуктивності спроектована на основі парадигми «4+1» і визначає сукупну поведінку системи.

Набір програмних модулів включає модулі `Traffictimegaugeserver, Timegaugeclient, Networkclient, Consoleapi і rdtsctimer`.

Реалізований обсяг тестування забезпечує перевірку всіх вимог ТЗ.

На основі результатів обробки експериментальних даних розроблений порядок розміщення правил у списках правил системи МЕ

.

ВИСНОВКИ

У процесі виконання атестаційної роботи виконаний аналіз сучасного стану проблеми забезпечення високої продуктивності МЕ, використовуваних для захисту інформаційних веб-сайтів. Виділені такі найпоширеніші методи підвищення цього показника, як збільшення потужності апаратних платформ шляхом збільшення потужності процесора, збільшення обсягу кеш-пам'яті і т.д., заміна фільтрації рівня додатків на базову, зменшення обсягу протоколіруємої інформації й оптимізація розташування правил у списках. Виділені основні уразливості, найбільш характерні для сучасних веб-сайтів, а також основні фактори, що виявляють істотний вплив на продуктивність.

На основі цієї інформації розроблена модель мережі веб-сайту, з якої вилучені елементи, несуттєві для вивчення проблем продуктивності. Обране ПО необхідне для проведення експериментальних досліджень у складі МЕ Kerio Winroute Firewall і ipfirewall, а також поштового сервера Mdaemon.

Також розроблено метрики, по яких проводилися експериментальні виміри продуктивності. У результаті аналізу типових політик безпеки даних веб-сайтів розроблений план експерименту.

На основі вивчення доступних засобів виміру продуктивності мережі зроблений висновок про їхню непридатність для використання в даній роботі. У результаті ухвалене рішення про розробку спеціалізованого ПЗ, здатного проводити тестування в багатопоточному режимі з декількома клієнтами з використанням великої кількості тестових з'єднань.

У процесі розробки створена архітектура ПЗ задовольняючого цим умовам, протокол мережної взаємодії клієнтів і сервера, спроектовані й реалізовані відповідні модулі. У процесі верифікації й валідації перевірені всі вимоги ТЗ і зроблений висновок про коректність реалізації ПЗ.

Розроблене ПЗ використане для проведення експерименту над моделлю мережі веб-сайту. Результати експерименту зафіксовані й піддані відповідній до первинної обробки.

Отримані емпіричні формули залежності показників продуктивності по раніше розроблених метриках від використовуваних правил безпеки.

У результаті аналізу отриманих формул розроблений рекомендований порядок розміщення правил безпеки в списках системи МЕ в заданих умовах, що свідчить про виконання всіх завдань і досягненні мети роботи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1 Синицын С.В., Налютин Н.Ю. «Верифікація програмного забезпечення. / М.: 2006. – 258 с.

2 Оглтри Т. Firewalls. Практичне застосування мережевих екранів: Пер. с англ. Москва 2011. 400 с.

3 В.А.Галатенко. Стандарти інформаційної безпеки: курс лекцій: навчальний посібник/Друге видання. "Інтернет університет Інформаційних Технологій", 2016. 264 с.

4 Стандарт ISO/IEC 27001 "Системи менеджменту ІБ".

5 Основні методи захисту від вилучених і локальних атак: www.it-seminars.ru/clause/clause/21636/2132/.

6 Посібник з TCP/IP: http://ishodniki.ru/art/ artshow.php?id=416&cat=3&show=веб_protocols.

7 Посібник з Ethernet мережам для початківців: http://ishodniki.ru/art/artshow.php?cat=5&id=1018&show=local.

8 Агулиев Р.М., Шахалиев Р.Г. Основні способи екранування корпоративних мереж: Інститут Інформаційних Технологій Національної Академії Наук Азербайджану. Баку.

9 Проскурін В.Г. Програмно-апаратні засоби забезпечення інформаційної безпеки. Захист в операційних системах. –М.: Радіо й зв'язок, 2000.

10 Журнал "Інформаційні технології моделювання й керування". – М.: Наукова книга, 2006

11 Труфляк Е.В. Методика вибору кількості повторностей при проведенні експериментальних досліджень: Науковий журнал Кубгау, №38(4), 2008.

12 IEEE 1471, Recommended Practice for Architecture Description of Software-Intensive Systems.

13 ДЕРЖСТАНДАРТ Р ІСО/МЕК 10746-3-2001, Керування даними й відкрита розподілена обробка. Архітектура.

14 Зорин В. Архітектура захищеного порталу: www.elvis.ru/files/prot_portal.pdf.

15 В. Сердюк. Як захистити ВебрПортал від інформаційних атак "Журнал мережних рішень LAN", 2004, №9.

16 Лабутина А. Практичний підхід до аналізу продуктивності й моделюванню великомасштабних паралельних додатків: Державний університет ім. Н.І.Лобачевського. 2007.

17 Марков А.С. Розробка політики безпеки організації у світлі новітньої нормативної бази: HTTP://npo-echelon.ru/about/articles/politics_development.php.

18 Шашков В. Б. Обробка експериментальних даних і побудова емпіричних формул. Курс лекцій Оренбург 2005.

19 Симоньян Т.А. Засобу обчислювальної техніки в захищеному варіанті, як частина рішення завдання інформаційної безпеки: журнал "Спеціальна Техніка" №2 2002.

20 Максимов В. Захист мережі: комплексний підхід: www.ellib.gpntb.ru/?journal=ap&year=2006&num=7&art=8.

21 Керівний документ: Засобу обчислювальної техніки. Мережеві екрани. Захист від несанкціонованого доступу до інформації. Показники захищеності від несанкціонованого доступу до інформації. Державна технічна комісія Російської Федерації 1997.

22 OSWAP, The ten most critical веб application security vulnerabilities, 2003

23 Shostak I., Matyushenko I., Romanenkov Yu., Danova M., Kuznetsova Yu. Computer Support for Decision-Making on Defining the Strategy of Green IT Development at the State Level. In book: Green-IT Engineering: Social, Business and

Industrial Applications, V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.), Vol. 171. Berlin, Heidelberg: Springer International Publishing, 533–559 (2018), https://doi.org/10.1007/978-3-030-00253-4

24 Avishai Wool. How to Use Firewall Management Solutions to Improve Firewall Performance and Security: HTTP://www.eweek.com/c/a/Security/how-to-use-firewall-management-solutions-to-improve-firewall-performance-and-security/.

25 Logan Twedt. A Comparison of Firewall Performance in Distributed Systems, Saint Mary's University of Minnesota, 2005.

26 Research and Markets Ltd. Firewalls are critical components of every network – from Small-Office/Home-Office (SOHO) through the largest Enterprises and Service Providers: HTTP://www.download3k.com/Press-firewalls-are-critical-components-of-every.html.

27 Albert Greenberg. Simulation Study of Firewalls to Aid Improved Performance: HTTP://www2.computer.org/portal/веб/csdl/doi/10.1109/ANSS.2006.42.

28 IEEE 1471, Recommended Practice for Architecture Description of Software-Intensive Systems.

29 ДЕРЖСТАНДАРТ Р ИСО/МЕК 10746-3-2001, Керування даними й відкрита розподілена обробка. Архітектура.

30 Microsoft Corporation. Посібник з розробки архітектури порталу органа державної влади, 2003.

31 К. Вигерс. Розробка вимог до програмного забезпечення/Пер. с англ. – 2018. – 576 с.