Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2024 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS”

КБПЗ_2024

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-23М
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Федюк Я.В.
«____» _____ 2024 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Дрєєв О.М.
«____» _____ 2024 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Рівень вищої освіти *магістр*
Галузь знань _12 _____*"Інформаційні технології"*_____
Спеціальність 123 *"Комп'ютерна інженерія"*
Освітньо-професійна (освітньо-наукова) програма _*"Комп'ютерна інженерія"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 6 » вересня 2024 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Федюку Ярославу В'ячеславовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS*

2. Керівник роботи *Дрєєв Олександр Миколайович, канд. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 19-13 від 07.08.2024 року

3. Строк подання студентом роботи до захисту *2.12.2024 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.	*6. Наукова новизна.*
2.Перегляд аналогічних існуючих систем.	*7. Маркетингове та економічне обґрунтування ІТ-проєкту.*
3. Опис і обґрунтування проектних рішень.	*8. Заходи з охорони праці та техніки безпеки.*
4. Етапи програмування системи.	*9. Висновки.*
5. Впровадження системи в промислову експлуатацію	

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Показники економічної ефективності	*1 аркуш*

КОПІЯ

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Доренська А.О.	05.10.2024	14.11.2024
Охорона праці	Марченко К.М., к.т.н., доцент	06.10.2024	16.11.2024

7. Дата видачі завдання « 6 » вересня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2024 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2024 р.	
3.	Розробка моделі компонента	20.10.2024 р.	
4.	Розробка структур даних	25.10.2024 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2024 р.	
6.	Програмування алгоритмів	10.11.2024 р.	
7.	Розрахунок економічної ефективності	13.11.2024 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2024 р.	
9.	Оформлення ПЗ	17.11.2024 р.	
10.	Попередній захист роботи	2.12.2024 р.	

КсПЗ_2024

Дата видачі завдання
« 6 » вересня 2024 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 6 » вересня 2024 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

АНОТАЦІЯ

Федюк Я.В. Дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2024.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи віртуального виділеного серверу з використанням технології VDS/VPS.

Метою розробки є дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

Об'єктом дослідження є процес віртуального виділеного серверу з використанням технології VDS/VPS.

Предметом дослідження є методи віртуального виділеного серверу з використанням технології VDS/VPS.

Методи дослідження базуються на методах теорії віртуалізації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерна інженерія, віртуальний сервер, VDS/VPS

КБПЗ_2024

ABSTRACT

Fedyuk Ya.V. Research and software implementation of a virtual dedicated server system using VDS/VPS technology. 123 Computer engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this graduation thesis for the second (master's) level of higher education, software is developed, which is intended for a virtual dedicated server system using VDS/VPS technology.

The goal of development is research and software implementation of a virtual dedicated server system using VDS/VPS technology.

The object of research is the process of a virtual dedicated server using VDS/VPS technology.

The subject of research is the methods of a virtual dedicated server using VDS/VPS technology.

Research methods are based on virtualization theory methods, mathematical statistics methods, and software development methods.

The result of the work is the software implementation of the virtual dedicated server system using VDS/VPS technology.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program can be used on a PC with Windows OS 10/11.

The program was developed in the Python environment.

Keywords: computer engineering, virtual server, VDS/VPS

КБПЗ_2024

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ 3
ВСТУП 4
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ 6
1.1 Призначення системи 6
1.2 Область застосування 7
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ 9
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти 9
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування 12
2.3 Розгорнута постановка завдання 17
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ 19
3.1 Опис функціонування системи 19
3.2 Розробка структурної схеми 22
3.3 Розробка функціональної схеми 27
3.4 Розробка діаграми процесів 37
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ 39
4.1 Розробка блок-схем та опис алгоритмів функціонування системи 39
4.2 Захист розробленого програмного забезпечення 49
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ 52
6 НАУКОВА НОВИЗНА 59

КБПЗ_2024

7 МАРКЕТИНГОВЕ ТА ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ ІТ-ПРОЄКТУ 60
7.1 Визначення цільової аудиторії кінцевого готового продукту 60
7.2 Оцінка привабливості шляхом застосування методів експертних оцінок ... 61
7.3 Вибір методу оцінки вартості ПЗ 63
7.4 Розрахунок економічної ефективності від впровадження реалізованого ПЗ як фактору його привабливості 64
7.5 Пропозиція алгоритму просування проєкту розробки ПЗ 65
7.6 Оптимізація каналів збуту та шляхів реалізації ПЗ 66
7.7 Визначення ключових факторів успіху конкретного проєкту 67
8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ 69
8.1 Вступ 69
8.2 Аналіз умов праці 70
8.3 Техніка безпеки та протипожежна профілактика 74
8.4 Розробка заходів з охорони праці 76
8.5 Висновки до розділу 77
9 ОСНОВНІ ВИСНОВКИ 78
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 80

К6ПЗ_2024

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ЕОМ	–	електрона обчислювальна машина
НСД	–	несанкціонований доступ
ПЗ	–	програмне забезпечення
СУБД	–	система управління базами даних
СВВ	–	система виявлення вторгнень

К6ПЗ_2024

ВСТУП

Актуальність теми. Віртуальний приватний сервер (VPS) – це тип хостингу, який дозволяє створити віртуальний сервер із виділеною операційною системою та ресурсами. Це означає, що VPS працюватиме з власною операційною системою, матиме власну IP-адресу та буде повністю ізольований від інших VPS на тому ж фізичному сервері.

Віртуальний виділений сервер (VDS) – це інший тип хостингу, який пропонує більше контролю над апаратними ресурсами, ніж VPS. На відміну від VPS, які розподіляються як часові інтервали на базовому фізичному сервері, ви можете виділити певні апаратні ресурси для свого VDS.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем віртуального виділеного серверу з використанням технології VDS/VPS.

– Дослідження системи віртуального виділеного серверу з використанням технології VDS/VPS.

– Програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

Об'єктом дослідження є процес віртуального виділеного серверу з використанням технології VDS/VPS.

Предметом дослідження є методи віртуального виділеного серверу з використанням технології VDS/VPS.

Методи дослідження базуються на методах теорії віртуалізації, методах математичної статистики, методах розробки програмного забезпечення.

КБПЗ_2024

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод віртуального виділеного серверу з використанням технології VDS/VPS.

– Розроблено вітчизняний продукт віртуального виділеного серверу з використанням технології VDS/VPS, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі віртуального виділеного серверу з використанням технології VDS/VPS.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Робота апробована на LVII Науково-технічній конференції здобувачів вищої освіти LV науково-технічної конференції «Наука в ЦНТУ: основні досягнення та перспективи розвитку» (2024 р.), основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №15.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

КБПЗ_2024

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Однією з головних переваг хостингу VDS/VPS є можливість інсталювати та налаштовувати спеціальне програмне забезпечення відповідно до ваших потреб. Завдяки кореневому доступу до сервера ви маєте повний контроль над програмним середовищем.

Хостинг VDS або віртуального виділеного сервера (також відомого як віртуальний приватний сервер) є чимось середнім між спільним і виділеним хостингом. Хоча вам все одно доведеться використовувати частину фізичних ресурсів сервера, їх кількість є солідною.

Уявіть, що ваш віртуальний сервер розгорнуто на фізичному сервері з 8 ГБ оперативної пам’яті. Якщо ваш тарифний план обіцяє 1 ГБ оперативної пам’яті, у вашому розпорядженні буде саме стільки. Ваші «сусіди» не зможуть використовувати виділені вам ресурси.

Щоб зробити це можливим, хостинг-провайдери використовують спеціальне програмне забезпечення, яке називається гіпервізорами. З їх допомогою на сервері для кожного клієнта створюються віртуальні машини із заданим набором ресурсів. Ця віртуальна машина діє як окремий сервер, де ви можете встановити власну операційну систему та інше необхідне програмне забезпечення, наприклад панель керування або стек LAMP/LEMP. Завдяки віртуалізації VDS/VPS можна налаштувати дуже швидко.

Крім того, користувач не повинен бути особливо вправним адміністратором. Хоча VDS/VPS вимагає мінімальних навичок адміністрування, більшість дій, таких як встановлення ОС і додатків, автоматизовані. І якщо вам усе ще потрібна більш точна настройка, ви завжди можете скористатися кореневим доступом, який пропонують більшість провайдерів.

КБПЗ_2024

Віртуальний сервер також легко масштабувати. Якщо ви відчуваєте, що поточний набір ресурсів проект уже досяг своїх меж, ви можете подати запит на оновлення, залишаючись на тій самій віртуальній машині.

1.2 Область застосування

Виділений сервер – це найбільш гнучкий, потужний і надійний спосіб розміщення веб-сайту. Як випливає з назви, у вас буде цілий фізичний сервер із повним набором доступних вам ресурсів. Тут вам більше не доведеться ділити простір з іншими клієнтами, що помітно відрізняє виділений сервер від спільного хостингу та VDS/VPS.

Існує велика різноманітність конфігурацій виділених серверів, тож ви можете вибрати той, який підходить саме вам як за потужністю, так і за ціною. У той же час постачальники часто дозволяють клієнтам повністю налаштувати програмне забезпечення на сервері, включаючи ОС, програми, RAID або стеки технологій, а також апаратне забезпечення. Наприклад, якщо ви помітили, що на сервері бракує пам’яті, ви можете встановити додатковий жорсткий диск або SSD.

Однак, щоб повною мірою використовувати можливості виділеного сервера, вам потрібні серйозні навички адміністрування. Найчастіше сервери постачаються або з попередньо встановленою ОС, або навіть без неї (голі сервери). Тому для безперебійної роботи потрібен спеціаліст, який буде відповідати за адміністрування.

Недоліки використання VDS/VPS сервера проти виділеного сервера:

– Загальна дискова система й канал зв'язку: всі віртуальні сервери, що працюють на одній хост-машині, за умови поділу основних ресурсів використовують одну на всіх дискову систему й загальний високошвидкісний канал зв'язку. Це означає, що деякі користувачі можуть зловживати використанням ресурсів і заважати іншим клієнтам.

КБПЗ_2024

Ми використовуємо автоматичну систему, що знаходить таких користувачів. До порушників застосовуються міри від попередження, до вимикання VDS.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

К6ПЗ_2024

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

При замовленні віртуального сервера, клієнт вибирає операційну систему, що буде встановлена. Крім операційної системи, можна вибрати Рецепт – набір програмного забезпечення, що буде встановлений поверх операційної системи.

– ОС без рецепта: установиться «чиста» операційна система. Всі необхідні програми й утиліти прийде встановлювати самостійно.

– ОС із рецептом: відразу після запуску сервера на нього встановиться обране програмне забезпечення.

Передвстановлене програмне забезпечення (ПЗ) займає загальний дисковий простір віртуального сервера. Установлене на сервер ПЗ не обновляється нашими фахівцями, щоб не порушити сумісність із користувальницькими додатками. Клієнт установлює необхідні відновлення самостійно.

Рецепти

Рецепт – набір програмного забезпечення, що буде встановлений на сервер поверх операційної системи. Це можливість швидко підготувати сервер до роботи – наприклад, установити LAMP за два кличі. Скористатися рецептом можна:

– Під час покупки нового сервера в Особистому кабінеті.

– При переустановці ОС на наявному сервері в панелі VMmanager.

КБПЗ-2024

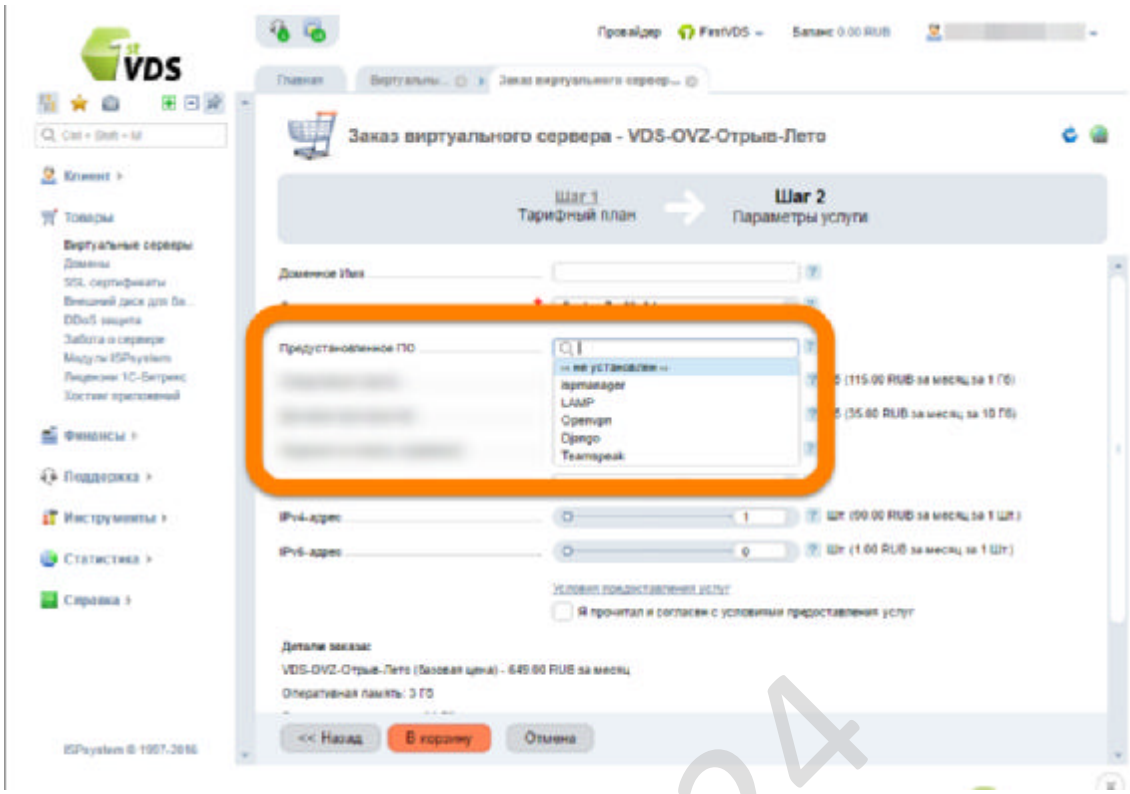


Рисунок 2.1 – Вибір рецепта в Особистому кабінеті при покупці сервера

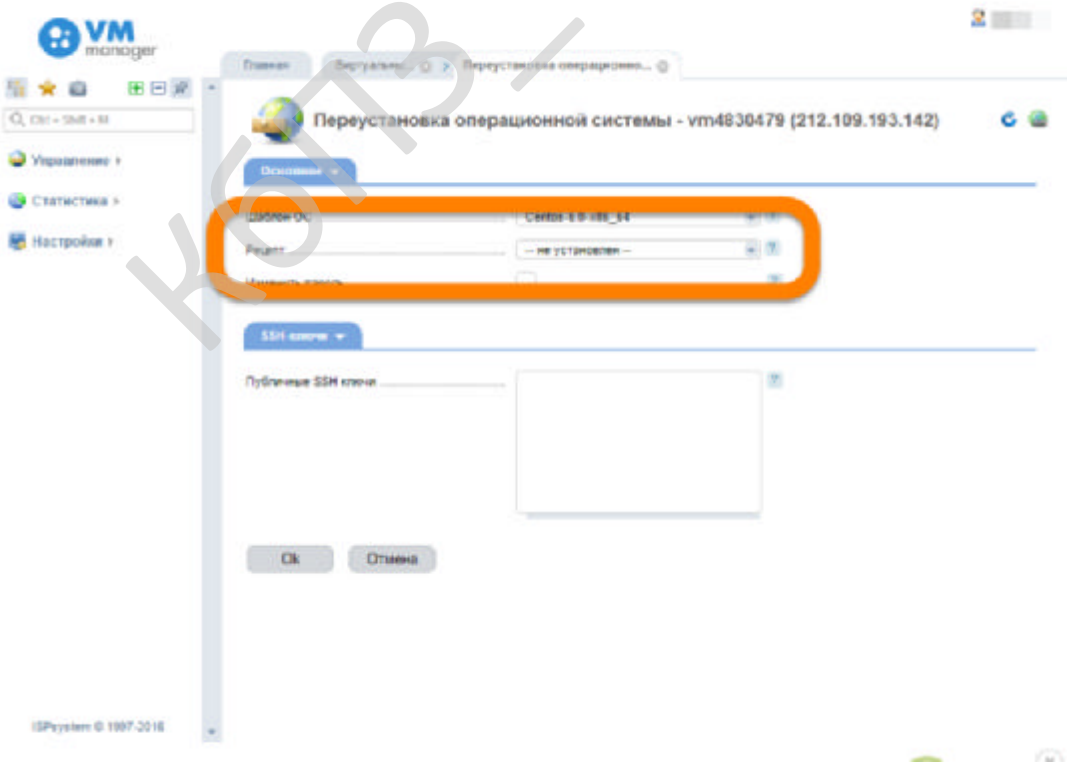


Рисунок 2.2 – Вибір рецепта в VMmanager при переустановці ОС

КПТЗ_2024

Доступні рецепти:

– LAMP – пакет для розробки Apache + MySQL + PHP.

– OpenVPN – вільна реалізація технології VPN (працює на віртуалізації KVM).

– Django – фреймворк для розробки веб-додатків мовою Python.

– TeamSpeak – голосовий чат через VoIP.

– ISPmanager – панель керування сервером ISPmanager 5 Lite.

Особливості роботи рецептів:

– Пакет установиться поверх обраної операційної системи – автоматично, відразу після активації сервера.

– Працюють із операційними системами Linux. Для Windows рецептів немає.

– За раз установлюється тільки один рецепт.

– Можна не встановлювати рецепт – одержите «чисту» операційну систему без додаткового ПЗ.

– Рецепти абсолютно безкоштовні.

Панель керування ispmanager 5 Lite

Панель керування ISPmanager 5 Lite більше не встановлюється разом операційною системою, а встановлюється окремо у вигляді рецепта. Установка рецепта з панеллю безкоштовна, однак, щоб використовувати повну версію панелі, як і раніше необхідно купити ліцензію. Разом з панеллю керування встановлюється весь набір програм, необхідний для веб-розробки – LAMP.

При виборі шаблона без панелі керування ISPmanager керування записами DNS здійснюється прямо з панелі DNSmanager.

Перемінити операційну систему

Змінити ОС можна в будь-який момент, переустановивши шаблон: у панелі VMmanager у розділі Керування → Віртуальні машини кнопка Переустановити. Зміна шаблона рівносильна переустановці ОС із форматуванням

КБПЗ-2024

диска – всі дані й налаштування втратяться! Заздалегідь подбайте про збереження інформації на стороннім сховищі.

Установити свою операційну систему

Ви можете встановити свій дистрибутив на віртуалізації KVM. Для цього в панелі VMmanager у розділі ISO-образи натисніть кнопку Створити й додайте свій ISO-образ системи.

Зверніть увагу, що установка ОС сімейства Windows зі своїх образів неможлива через обмеження ліцензійної угоди між нашою компанією й Microsoft.

Що таке LAMP

Це набір програмного забезпечення, звичайно використовуваного для веб-розробки й запуску сайту на сервері. У набір входять:

– Linux – операційна система Linux.

– Apache – веб-сервер.

– MySQL – система керування базами даних.

– PHP – мова програмування.

Для установки LAMP можна вибрати відповідний рецепт або вибрати рецепт ISPmanager – у нього також входить цей набір ПЗ.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – це об’єктноорієнтована мова програмування високого рівня загального призначення з відкритим кодом. Це визначення може бути важким для новачків, тому розглянемо кожну характеристику окремо, щоб зрозуміти, що вона означає:

– Відкритий вихідний код: це безкоштовно та доступно для подальших покращень, таких як додавання корисних функцій або виправлення помилок.

– Об’єктноорієнтована: заснована не на функціях, але в об’єктах з певними атрибутами й методами.

КБПЗ_2024

– Високий рівень: зручний для людини, а не для комп’ютера.

– Загальне призначення: можна використовувати для створення будь-яких програм.

Ця мова використовується в будь-якому програмному забезпеченні, про яке ви тільки можете подумати. Ви можете використовувати його для створення вебсайтів, штучного інтелекту, серверів, програмного забезпечення для бізнесу та багато іншого. Також застосовується в науці про дані, аналізі даних, машинному навчанні, інженерії даних, веброзробці, розробці програмного забезпечення та інших галузях.

Переваги та недоліки Python

Переваги:

– Її легко читати, вчити та писати. Це мова програмування високого рівня з англійським синтаксисом. Це полегшує читання та розуміння коду. Її дійсно легко зрозуміти і вивчити, тому багато людей рекомендують Python новачкам. Вам потрібно менше рядків коду для виконання того ж завдання в порівнянні з іншими основними мовами, такими як C/C++ та Java.

– Підвищує продуктивність. Це дуже продуктивна мова. Завдяки її простоті розробники можуть зосередитися на розв’язанні проблеми. Їм не потрібно витрачати багато часу на розуміння синтаксису або поведінку мови програмування. Ви пишете менше коду та виконуєте більше завдань.

– Інтерпретована мова. Python мова, що інтерпретується, а це означає, що вона безпосередньо виконує код по рядку. Якщо сталася помилка, вона зупиняє подальше виконання та повідомляє про її виникнення. Вона показує лише одну помилку, навіть якщо у програмі їх кілька. Це спрощує налагодження.

– Динамічно типізована. Python не визначає тип змінної, доки ми не запустимо код. Вона автоматично надає тип даних, коли відбувається процес виконання. Фахівець може не турбуватися про оголошення змінних та типи даних.

КБПЗ_2024

– Безкоштовна та з відкритим вихідним кодом. Ця мова постачається під схваленою OSI ліцензією з відкритим вихідним кодом. Це робить його безкоштовним для використання та розповсюдження. Ви можете завантажити вихідний код, змінити його та навіть розповсюджувати свою версію. Це корисно для організацій, які хочуть використати свою версію для розробки.

– Підтримка великих бібліотек. Стандартна бібліотека Python є величезною, ви можете знайти майже всі функції, необхідні для вашого завдання. Таким чином ви не залежите від зовнішніх бібліотек.

– Портативність .У багатьох мовах, таких як C/C++, потрібно змінити свій код, щоб запустити програму на різних платформах. З Python все інакше. Ви тільки пишете один раз і запускаєте її будь-де.

Недоліки:

– Низька швидкість. Вище ми обговорювали, що це інтерпретована мова з динамічною типізацією. Порядкове виконання коду часто призводить до повільного виконання. Динамічна природа Python також є причиною її низької швидкості, оскільки їй доводиться виконувати додаткову роботу при виконанні коду. Тому вона не підходить для цілей, де швидкість важливий аспект проєкту.

– Неефективна для пам’яті. Ця мова програмування використовує великий обсяг пам’яті, це може бути недоліком при створенні програм, коли віддають перевагу оптимізації пам’яті.

– Слабка у мобільних обчисленнях. Python зазвичай використовується у серверному програмуванні. Ми не бачимо – її на стороні клієнта або в мобільних програмах з таких причин: вона не заощаджує пам’ять і має повільну обчислювальну потужність у порівнянні з іншими мовами.

– Доступ до бази даних. Програмувати на цій мові легко, але коли ми взаємодіємо з базою даних, її не вистачає. Рівень доступу до бази даних у Python примітивний та недостатньо розвинений у порівнянні з іншими популярними технологіями.

КБПЗ_2024

– Помилки виконання. Це мова з динамічною типізацією, тому тип даних змінної може змінюватись у будь-який час. Змінна, що містить ціле число, у майбутньому може містити рядок, що може призвести до помилок виконання.

Застосування Python:

– Для аналізу даних. Дані стали цінним активом у будь-якій сучасній галузі, і більшість компаній зацікавлені у збиранні, обробці та аналізі релевантних даних, щоб витягти з них цінну інформацію для бізнесу. І тут Python виходить за межі будь-якої конкуренції. Python особливо цінна тим, що крім великої стандартної бібліотеки надає величезний набір додаткових модулів, розроблених спеціально для аналітичних цілей. Найвідоміші бібліотеки Python для аналізу даних – це pandas і NumPy . Ці інструменти дозволяють робити з вашими даними майже все, наприклад, очищати і аналізувати їх, вивчати статистику або візуалізувати приховані тенденції у ваших даних.

– Для візуалізації даних. Візуалізація даних – це окрема частина аналізу даних, яка допомагає нам подавати інформацію, необроблену чи очищену, у більш змістовній формі. Тут Python знову входить у гру, пропонуючи широкий спектр інструментів візуалізації даних. Найпопулярніші з них – matplotlib і заснований на ній seaborn. Використовуючи їх, ми можемо створювати буквально всі види візуалізації: від найпростіших до складніших.

– Для машинного навчання. Машинне навчання (ML) є основою більшості завдань науки даних. Він є областю штучного інтелекту, пов’язаною з використанням алгоритмів, що дозволяють машинам вивчати закономірності та тенденції на основі історичних даних, щоб робити прогнози на основі невідомих даних. – Використовуючи методи ML, ми можемо створювати моделі, які можуть точно передбачити швидкість відтоку клієнтів компанії, оцінити ризик виникнення у людини певного захворювання, визначити оптимальне розташування автомобілів таксі й т.д. За допомогою Python ми можемо побудувати модель ML, використовуючи лише три рядки коду.

КБПЗ_2024

– Для розробки програмного забезпечення. Крім свого багатостороннього застосування в галузях науки про дані, Python використовується на кожному етапі розробки програмного забезпечення, включаючи контроль складання, автоматичну безперервну компіляцію, прототипування, відстеження помилок, тестування та обслуговування програмного забезпечення. За допомогою цієї мови можемо створювати аудіо- або відеопрограми на основі методів штучного інтелекту, машинного навчання, API (інтерфейсів прикладного програмування), GUI (графічних інтерфейсів) або будь-якого іншого типу програмного забезпечення.

– Для веброзробки. У той час як для створення візуальної частини вебсайту ми переважно будемо використовувати такі мови, як HTML, CSS та JavaScript, для його невидимої частини ми часто вибираємо Python. Серед масштабних вебсайтів та програм, створених за допомогою цієї мови, варто згадати Google, Facebook, Instagram, YouTube, Dropbox та Reddit.

– Для автоматизації задач/скриптингу. Це відмінний інструмент для написання програм для автоматизації різних завдань, що повторюються. Цей процес називається скриптингом. Зокрема, можна робити скрипти для роботи з файлами та папками. Наприклад, можна створювати, перейменовувати, перетворювати, розділяти, об’єднувати або видаляти файли, перевіряти їх на наявність помилок. Ви також можете використовувати автоматизацію Python для пошуку та завантаження інформації з Інтернету, заповнення та надсилання онлайн-форм та надсилання регулярних повідомлень або електронних листів.

Яким фахівцям потрібно володіти Python:

– Фахівець з даних.

– Аналітик даних.

– Інженер даних.

– Інженер з машинного навчання.

– Журналіст даних.

– Архітектор даних.

КБПЗ_2024

– Повний стек веброзробника.

– Backend-розробник.

– DevOps-інженер.

– Інженер-програміст

Можемо зробити висновок, що Python ще довго буде популярною мовою, хоч і має низку недоліків. Цю мову використовують для створення вебсайтів, штучного інтелекту, серверів, програмного забезпечення для бізнесу, аналізу даних, машинного навчання, інженерії даних та для багатьох інших областей. Це перспективна і затребувана навичка, яка необхідна у всіх галузях.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи віртуального виділеного серверу з використанням технології VDS/VPS.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

КБПЗ_2024

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

К6ПЗ_2024

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

У даному розділі проведемо детальне порівняння віртуальних та виділених серверів.

1. Розподіл ресурсів

Віртуальний сервер (VDS/VPS) використовує частину ресурсів фізичного сервера, на якому працює віртуальна машина. Ці ресурси повністю у вашому розпорядженні і не можуть бути використані вашими «сусідами» на сервері.

Виділений сервер (DS) використовує 100% ресурсів, оскільки є повноцінною фізичною машиною. Це означає, що вам не потрібно турбуватися про інших користувачів, ви повністю контролюєте ресурси сервера.

Якщо ваш проект вимагає постійної високої продуктивності, вибирайте виділений сервер, оскільки він надає доступ до повної його потужності.

2. Безпека

Віртуальний сервер (VDS/VPS) є досить безпечним рішенням. На нього можна встановити будь-яке програмне забезпечення, необхідне для захисту, але наявність «сусідів» на одному фізичному сервері робить його більш вразливим. Якщо одного з користувачів вашого фізичного сервера зламано, ваш VDS/VPS також може бути скомпрометований.

Виділений сервер (DS) – це фізично ізольоване середовище. Тут немає інших користувачів, ви можете встановити необхідне програмне забезпечення, налаштувати захист самостійно та повністю контролювати пом’якшення інцидентів. Виділений сервер є більш безпечним варіантом. Хоча VDS/VPS сам по собі забезпечує високий рівень безпеки, виділений сервер дає вам повний контроль над усією системою безпеки.

КБПЗ_2024

3. Продуктивність

Віртуальний сервер (VDS/VPS) – це збалансоване місце серед варіантів розміщення веб-сайтів. Хоча загальна швидкість завантаження і продуктивність сайту на VDS/VPS нижче, ніж на виділеному сервері, цього цілком достатньо для невеликих комерційних або великих приватних сайтів, які потребують більше ресурсів, ніж може надати віртуальний хостинг. Такий тип хостингу допомагає прискорити завантаження розміщених сайтів і уникнути простоїв. Це також чудовий інструмент для боротьби з раптовими скачками трафіку.

Виділений сервер (DS) є найпотужнішим варіантом розміщення веб-сайтів на ринку. Такі сервери зазвичай використовують найсучасніше обладнання та забезпечують найвищу пропускну здатність. Усе це допомагає витримувати раптові сплески трафіку та постійно підтримувати високу швидкість навантаження.

Завдяки своїй сирій потужності виділений сервер забезпечує високу швидкість і продуктивність розміщеного веб-сайту. VDS/VPS в цьому плані втрачає небагато, але може сповільнитися, якщо кілька сайтів на сервері матимуть сплеск трафіку.

4. Конфігурація та налаштування

Віртуальний сервер (VDS/VPS) не вимагає високих навичок адміністрування сервера, щоб почати роботу. Оскільки це лише частина фізичного сервера, провайдер створює віртуальну машину та надсилає вам необхідні дані доступу, щоб ви могли почати працювати зі своїм VDS/VPS. Крім того, ви можете налаштувати сервер відповідно до ваших потреб за допомогою автоматичних інструментів або кореневого доступу. Це дозволяє налаштовувати VDS/VPS на льоту.

Виділений сервер (DS) також надає інструменти автоматичного налаштування та кореневий доступ. Однак у цьому випадку керування фізичним сервером є обов’язком клієнта. Тому для керування сервером може знадобитися виділений адміністратор. Крім того, деякі провайдери пропонують послугу

КБПЗ_2024

віддаленого адміністрування.

Віртуальним сервером легше керувати, і він є більш гнучким у налаштуванні, тоді як виділений сервер вимагає глибоких навичок адміністрування, але дозволяє змінювати апаратне забезпечення.

5. Масштабованість

Віртуальний сервер (VDS/VPS) є дуже гнучким рішенням. Завдяки віртуалізації ви можете легко змінити обмеження ресурсів для конкретного VDS/VPS. Якщо, наприклад, кількість регулярних відвідувачів веб-сайту зростає, а виділених ресурсів стає недостатньо для стабільної роботи, ви можете оновити свій тарифний план і не хвилюватися про перенесення сайту на інший сервер.

Виділений сервер (DS) є більш статичним рішенням. Маючи більше ресурсів і високу продуктивність, фізичний сервер часто досягає бетонної стелі у вигляді власного обладнання. Щоб змінити конфігурацію, потрібно або замінити обладнання в дата-центрі, або переміститися на новий сервер.

З точки зору масштабованості віртуальний сервер перевершує виділений. Завдяки гнучким планам обслуговування та можливості швидко їх змінювати ви можете швидко пристосуватися до потреб веб-сайту, що розвивається. Виділений сервер, у свою чергу, відразу надає багато ресурсів, але його набагато складніше оновити.

6. Ціна

Віртуальний сервер (VDS/VPS) є варіантом хостингу середнього рівня. Вартість оренди фізичного сервера розподіляється між усіма «сусідами», тому це забезпечує відмінне співвідношення ціни та якості. Це повноцінний сервер, який можна налаштувати набагато дешевше, ніж виділений сервер.

Виділений сервер (DS) є найдорожчим варіантом розміщення сайту. Висока продуктивність, швидкість і надійність досягаються за допомогою дорогого сучасного обладнання. Крім того, в цьому випадку ви орендуєте всю машину, тому загальна сума ні з ким не ділиться.

КБПЗ_2024

3.2 Розробка структурної схеми

Надаючи привабливі можливості хостингу за прийнятну ціну, VPS поступово перетворився в хмарний сервіс, що забезпечує гнучке використання серверних ресурсів.

Хостинг розвивався разом з Web: цей вид послуг був найбільше широко затребуваний саме для розміщення Web-сайтів у ЦОДах провайдерів ще в той час, коли про хмарні сервіси ніхто не чув, але системи віртуалізації для платформи x86, такі як Virtuozzo, уже з'явилися. Цей продукт, випущений в 2001 році й забезпечуючий контейнерну віртуалізацію для Linux, швидко став популярним у хостинг-провайдерів завдяки високій щільності й гарній ізоляції віртуальних середовищ укупі з низькими накладними витратами. П'ять років через ці наробітки стали доступні у вигляді Open Source у рамках проекту OpenVZ.

В Virtuozzo компанії SWsoft (пізніше перейменованої в Parallels) віртуалізується рівень операційної системи. Віртуальні приватні сервери (Virtual Private Servers, VPS) являють собою гостьові ОС, які запускаються на фізичному сервері (що працює під керуванням Linux, а пізніше й під Windows) в ізольованих віртуальних середовищах (контейнерах). При цьому кожна така копія – «віртуальна машина» – виглядає для користувача як окремий сервер.

За допомогою віртуалізації вдалося домогтися досить ефективного використання фізичних IT-ресурсів. Звичайно постачальники послуг хостингу встановлювали на свої сервери доступні дистрибутиви Linux. Завдяки прийнятній ціні послуг VPS користувачі всі частіше стали прибігати до цієї пропозиції для рішення своїх завдань.

Віртуалізація програмна й апаратна

Хостингові послуги VPS розвивалися слідом за програмними засобами віртуалізації, яким стало з'являтися усе більше: VMware ESX, Microsoft Hyper-V, XenSource, Xen, KVM, HyperVM, FreeVPS, FreeBSD Jail, VDSManager, Solaris

КБПЗ-2024

Zones. У цей час існують дві основні технології віртуалізації для хостингу: програмна й повна (апаратна).

У системах програмної віртуалізації (до цієї групи можна віднести, наприклад, OpenVZ, Virtuozzo, VDSmanager) віртуальні машини (ВМ) використовують загальне ядро ОС. У випадку ОС Linux користувачеві VPS доступний дистрибутив Linux (CentOS, Debian, Ubuntu і ін.), а для ОС Windows надається сервер, оснащений тією же версією Windows.

Апаратна віртуалізація (до цієї категорії можна віднести гіпервізори Xen, KVM, Microsoft Hyper-V) припускає, що ресурси фізичного сервера розділяються на трохи віртуальних, ізольованих один від іншого серверів, причому с будь-якими операційними системами – Linux, Windows або FreeBSD.

Таким чином, у результаті апаратної віртуалізації створюється повністю ізольований аналог фізичного сервера, у якого є власне ядро ОС. До того ж для нього виділяються певна область операційної пам'яті, місце на диску, а також мережна карта. Однак додавати ці ресурси (у тому числі збільшувати число процесорних ядер) замовник звичайно не може.

Віртуальний приватний (виділений) сервер

Власник віртуального сервера може встановлювати на ньому додатка, розгортати Web-сайти й працювати з файлами у вилученому режимі точно так само, як при використанні фізичного сервера. Завдяки наявності індивідуального IP-адреси (або декількох IP-адрес), вона функціонує ізольовано від інших віртуальних машин, запущених на цьому ж сервері. Але в чому ж тоді відмінність VPS, віртуального приватного сервісу, від VDS – віртуального виділеного сервісу?

І той і інший емулюють роботу фізичного сервера. Звичайно вважається, що VDS і VPS розрізняються типом віртуалізації. І хоча у випадку VDS виділяється фіксований обсяг більш жорстко розмежованих фізичних ресурсів, обидва терміни фактично означають те саме.

КБПЗ_2024

В обох випадках замовник одержує у своє розпорядження частина ресурсів фізичного сервера (процесор, пам'ять, ємність підсистеми зберігання даних, частку пропускної здатності мережі), а крім того, він має право самостійно встановлювати ОС і програмне забезпечення.

У порівнянні з віртуальним хостингом VPS-хостинг дає власникові повний контроль над своїм віртуальним сервером при більше високій продуктивності. Разом з тим, він обходиться дешевше, ніж фізичного сервера.

Віртуальний хостинг

Можливостей налаштування й керування при віртуальному хостингу менше, ніж в VPS (див. рис. 3.1). Користувачеві виділяються ресурси на заздалегідь підготовленому сервері, але він може змінювати тільки інформаційне наповнення виділених йому ресурсів, не маючи права не коректувати налаштування, не встановлювати додаткове ПЗ. Такий хостинг призначений для тих випадків, коли не потрібні значні потужності або гарантовано високий рівень безпеки, а відвідуваність і навантаження невеликі.

При віртуальному хостингу ресурси фізичного сервера розділяються між його користувачами, тому може виникати дефіцит ресурсів під час пікових навантажень

Оскільки твердого поділу ресурсів ні, конкретний сайт може «гальмувати», тому що сусідній споживає занадто багато. Втім, адміністратори намагаються не допускати подібних ситуацій.

Тим часом VPS-хостинг звичайно дорожче віртуального й припускає наявність певних технічних знань, особливо у випадку не керованого провайдером сервісу. Щоб освоїти VPS, навчитися моніторингу й керуванню сервером, домогтися його стабільної й надійної роботи, знадобляться час і зусилля. На віртуальному виділеному сервері можна розміщати будь-яка кількість сайтів, баз даних, доменних зон та ін. Доступні фізичні ресурси обмежені, але VPS їх свідомо одержить, у той час як на віртуальному хостингу такої гарантії немає.

КБПЗ_2024

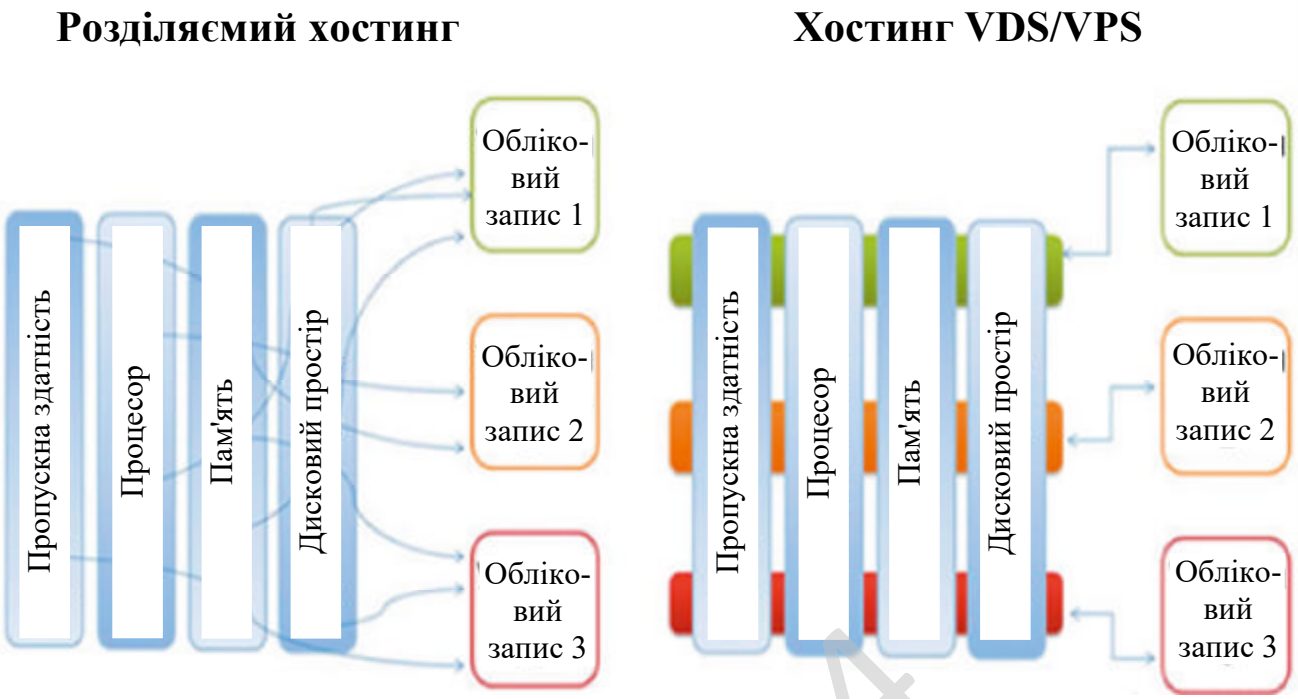


Рисунок 3.1 – Структурна схема системи

Головна відмінність віртуального хостингу від VPS складається в способі поділу ресурсів фізичного сервера. У випадку VPS вони виділяються кожному віртуальному серверу, за який відповідає його власник.

Як ми вже відзначали, власник VPS має повний доступ до системи (права користувача root) і може встановлювати кожне ПЗ, тоді як при віртуальному хостингу це може робити тільки провайдер.

Для віртуального сервера VPS/VDS звичайно передбачаються захист від DDoS-атак, можливість резервного копіювання даних і окремий IP-адреса (при необхідності можна одержати й ще трохи). У випадку віртуального хостингу найчастіше для декількох сайтів виділяється один IP-адреса.

Ще можливий варіант – хмарний хостинг.

Хмарний хостинг

Хмарний VPS (Cloud VPS) відрізняється гнучкістю: користувач може самостійно вибрати обсяг необхідних йому ресурсів, додати їх або тимчасово

КБПЗ_2024

відмовитися, а також створити другий сервер. При цьому оплачуються тільки реально спожиті ресурси.

При виборі VPS або Cloud VPS варто оцінити масштаби майбутнього проекту й спрогнозувати зміна потреби в ресурсах, тому що тарифи звичайного хостингу VPS можуть виявитися більше вигідними. Якщо навантаження на сервер нерівномірна, частина часу доби сервер буде працювати вхолосту. При хмарному хостингу застосовується погодинна оплата, причому тариф можна міняти як у ручному, так і в автоматичному режимі.

У числі провайдерів хмарного хостингу – Active Cloud, Amazon EC2, IPS Server, IX Webhosting, HostPro, Clodo, Cloud4y, Cloudes.ru, Cloudone Parking.ru, RuVDS, Scalaxy, Selectel, Slidebar.

Фізичний хостинг

Компанія, що надає послуги хостингу (хостер), забезпечує надійність і якість підключення серверів до мережі, надає послуги техпідтримки, здійснює обслуговування, модернізацію встаткування й резервне копіювання даних, гарантує безпека й т.д. Подібний сервіс досить затребуваний, але іноді необхідно більше – наприклад, установка спеціального ПЗ, керування їм або можливість повністю розпоряджатися своїм сервером.

У таких випадках додатка й дані розміщаються на виділених фізичних серверах: на орендованому (Dedicated) або на встаткуванні замовника, установленому в провайдеру (Colocation). Звичайно виділені сервери використовуються при значних вимогах до ресурсів – для великих сайтів з високою відвідуваністю, онлайн-ігор, торговельних площадок та ін. Власникові такого сервера доступні всі його ресурси й налаштування – аж до установки кожного ПЗ й заміни операційної системи.

Цей вид сервісу пропонують багато великих провайдерів і власники центрів обробки даних. Вони забезпечують безперебійне енергопостачання, надають можливість вибрати необхідне встаткування, системи охолодження, моніторингу й безпеки.

КБПЗ_2024

Вартість використання фізичного виділеного сервера значно вище, ніж віртуального, хоча, крім набору надаваних ресурсів, VPS майже нічим йому не уступає. Виключення становить лише використання додатків, яким потрібен прямий доступ до встаткування.

Керування серверами: web-консолі

Власникам віртуальних і фізичних серверів потрібний інструментарій для вилученого керування серверами. Для цієї мети використовуються панелі керування – Web-Консолі. З їхньою допомогою адміністратори й користувачі можуть управляти різними сервісами сервера з Web-Браузера. У більшості панелей керування реалізований однаковий набір типових функцій, включаючи керування службою доменних імен, системою електронної пошти, сервісом FTP і доступом до файлової системи, базами даних, резервним копіюванням, користувачами, журналами й звітами.

Деякі пропонують і засобу корпоративного рівня, такі як керування декількома серверами, моніторинг сервісів і оповіщення, підтримку протоколу IPv6. Можливості панелей керування різноманітні. Одні краще підійдуть для віртуального хостингу, інші – для хмарного. Проте, незалежно від того, чи працює сайт на базі виділених або віртуальних серверів, багато замовників оцінять можливість управляти файлами, електронною поштою, додатками й іншими засобами використовуючи єдиний інтуїтивно зрозумілий інтерфейс.

КБПЗ_2024

3.3 Розробка функціональної схеми

Як було зазначено вище, на відміну від звичайного віртуального (shared) хостингу, на віртуальному сервері доступні технології захисту. Тому, у зв’язку з тим, що однією з основних вимог до системи віртуального виділеного серверу з використанням технології VDS/VPS є забезпечення безпеки, то функціонал системи в першу чергу буде спрямований на вирішення саме цієї задачі.

Функціонально архітектура системи віртуального виділеного серверу з використанням технології VDS/VPS включає (рисунок 3.2):

– модуль взаємодії з базою даних моніторингу системи віртуального виділеного серверу з використанням технології VDS/VPS;

– підсистему, призначену для збору подій, пов'язаних з безпекою системи віртуального виділеного серверу з використанням технології VDS/VPS;

– підсистему аналізу, призначену для виявлення атак і підозрілих дій на основі даних сенсорів системи віртуального виділеного серверу з використанням технології VDS/VPS;

– сховище, що забезпечує нагромадження первинних подій і результатів аналізу;

– модуль обробки результатів;

– консоль управління, що дозволяє конфігурувати системи віртуального виділеного серверу з використанням технології VDS/VPS, спостерігати за станом системи віртуального виділеного серверу з використанням технології VDS/VPS, і системи віртуального виділеного серверу з використанням технології VDS/VPS, переглядати виявлені підсистемою аналізу інциденти.

Для реалізації функціональної схеми опишемо структуру методу динамічного виявлення аномалій в інтенсивностях потоків телекомунікаційних даних, що включає наступні основні етапи:

1. Збір даних.

2. «Ручна» класифікація частини даних на предмет присутності нормального (типового) або аномального трафіку.

3. Попередній дисперсійний аналіз типового трафіку.

4. Виконання вейвлет-розкладання типового трафіку за обраними функціями для оцінки його спектрального-часового подання.

5. Обробка вейвлет-подання аномального трафіку.

а) визначення моментів різкої зміни характеру поводження рядів телекомунікаційних даних по рівневим і частотних характеристиках;

б) формування баз шаблонів вейвлет-образів:

– бази вейвлет-образів для характеру розвитку трафіку на тлі критичних ситуацій;

КБПЗ_2024

– бази вейвлет-образів для характеру розвитку трафіку до критичних моментів (до початку зареєстрованих проблем у роботі мережі);

– збереження відповідностей між даними баз;

в) визначення частотних показників для провокуючих розвиток аномалії подій, у тому числі аналіз енергетичних спектрів послідовностей до критичних моментів (спектра потужності й скалограми) на предмет перерозподілу енергії між частотами;

6. Верифікація прогнозу характеристик трафіку з реальними значеннями (на основі попередньо зібраних даних поза реальним режимом часу).

7. При надходженні нових даних:

а) зіставлення їх вейвлет-подань із шаблонами сформованих баз даних за допомогою кореляції й формування рішення про наявність передкритичної або критичної ситуації.

б) виявлення зв'язків між аномаліями, що відбулися в різних точках спостереження.

У результаті виконання етапів з першого по шостий включно маємо:

– алгоритм розпізнавання позаштатної ситуації;

– початкові бази вейвлет-шаблонів для позаштатних ситуацій і попередніх за часом інтервалів;

– модель прогнозування характеру розвитку трафіку і як наслідок апріорне визначення можливих позаштатних ситуацій і їхня класифікація.

Застосування спектральних і спектрально-часових алгоритмів, є основними інструментами в розв'язуваному завданні динамічного виявлення аномалій у часових рядах інтенсивностей телекомунікаційних даних:

– аналіз внесків частот визначає формування моделей типового трафіку й окремих ділянок з особливостями;

– картини вейвлет-коефіцієнтів аномалій і попередніх їм ділянок сигналу формують бази даних шаблонів для характеру розвитку трафіку на тлі критичних ситуацій і до критичних моментів;

КБПЗ – 2024

– побудовані на основі обчислених коефіцієнтів енергетичні спектри й інші залежності дозволяють відслідковувати розподіл (або перерозподіл) енергії між частотами й визначати сховану природу окремих ділянок сигналу.

Переваги запропонованого методу:

– висока швидкість обчислень;

– лінійна залежність швидкості обчислень від розміру «вікна».

Недоліки запропонованого методу:

– наявність швидко зростаючої погрішності обчислень ПХ після другої ітерації алгоритму перерахунку спектра.

Для усунення недоліку запропонованого методу необхідно:

– застосування розширених форматів подання даних;

– при досягненні критичного значення погрішності зробити перерахування спектра по стандартному алгоритмі.

Наведемо деякі з атак на комп’ютерну мережу системи віртуального виділеного серверу з використанням технології VDS/VPS.

Електронна пошта

Найбільший потік спаму поширюється через електронну пошту (e-mail). В наш час частка вірусів і спаму в загальному трафіку електронної пошти становить за різними оцінками від 85 до 95 відсотків.

Спамери збирають e-mail адреси за допомогою спеціального робота або вручну (рідко), використовуючи веб-сторінки, конференції Usenet, списки розсилок, електронні дошки оголошень, гостьові книги, чати тощо.

Такі програми-роботи здатні зібрати за годину тисячі адрес і створити з них базу даних для подальшого розсилання на них спаму.

Деякі компанії займаються тільки збором адрес, а бази потім продають. Деякі компанії продають спамерам e-mail адреси своїх клієнтів, що замовили в них товари чи послуги електронною поштою.

Для розсилання спаму використовуються підключені до Інтернет погано захищені комп'ютери. Це можуть бути:

КБПЗ_2024

– Сервери, які помилково відконфігуровані так, що дозволяють вільне пересилання пошти (open relay, open proxy).

– Webmail сервіси, які дозволяють анонімний доступ чи доступ з простою реєстрацією нових користувачів (яку можуть виконати спеціальні програми-роботи).

– Комп'ютер-зомбі. Деякі спамери використовують відомі уразливості в програмному забезпеченні чи комп'ютерні віруси для того, щоб захопити керування великою кількістю комп'ютерів, підключених до Інтернету і використовувати їх для розсилання спаму.

Для ускладнення автоматичної фільтрації спаму повідомлення часто спотворюються – замість букв використовуються схожі цифри, латинські букви замість кирилиці, у випадкових місцях додаються пропуски тощо. Застосовуються різні хитрощі для того, щоб переконатися, що повідомлення отримане й прочитане. Серед них:

– Підтвердження про доставку.

– Деякі поштові клієнти можуть відправляти його автоматично.

– Листи з зображеннями, які завантажуються із сайтів, контрольованих спамерами.

– Посилання на веб-сторінки, на яких пропонується одержати додаткову інформацію.

– Пропозиція відмінити підписку на цю розсилку, пославши листа на вказану адресу.

Якщо спамери одержують підтвердження, що поштова адреса дійсно використовується, то потік спаму може багаторазово збільшитися.

Миттєві повідомлення

З розвитком служб миттєвих повідомлень, таких як ICQ, AI і ін., спамери почали використовувати їх для своїх цілей.

Багато з цих служб надають список користувачів, який можна використати для розсилання спаму.

КБПЗ_2024

Блоги, Вікі

Останнім часом з'явилися веб-сайти, які можна вільно редагувати – блоги й вікі. Наприклад, Вікіпедія створена за цією технологією. Оскільки ці сторінки відкриті для вільного редагування, на них може розміщуватися спам.

SMS-повідомлення

Спам може поширюється не тільки через Інтернет. Рекламні SMS-повідомлення, які надходять на мобільні телефони особливо неприємні тим, що від них важче захиститися, і одержувач іноді повинен платити за кожне повідомлення.

Це може бути досить велика сума, особливо якщо абонент використовує роумінг.

Спамерам розсилання практично нічого не коштують, проте дорого обходяться одержувачу спаму, якому доводиться оплачувати своєму провайдеру час, витрачений на одержання непрошеної кореспонденції з поштового сервера.

А також через масовість поштових розсилань вони ускладнюють роботу інформаційних систем і ресурсів, створюючи для них непотрібне навантаження. Користувачі інтернету, крім того, змушені щодня витрачати час на фільтрацію рекламних повідомлень.

Для того щоб обмежити цей час, користувачі змушені використовувати протиспамові фільтри, які можуть стерти й важливе повідомлення, вважаючи його спамом. Втім, і людина, яка змушена переглядати десятки рекламних повідомлень у день, теж легко може пропустити серед них потрібне.

Боротьба зі спамом

Найнадійніший засіб боротьби зі спамом – не дозволити спамерам довідатися вашу електронну адресу. Це важке завдання, але деякі запобіжні заходи все ж варто вжити.

– Не варто без необхідності публікувати адресу електронної пошти на веб-сайтах чи в групах Usenet.

– Не потрібно реєструватися на підозрілих сайтах. Якщо якийсь корисний сайт вимагає реєстрації, можна вказати спеціально для цього створену адресу.

КОПЗ_2024

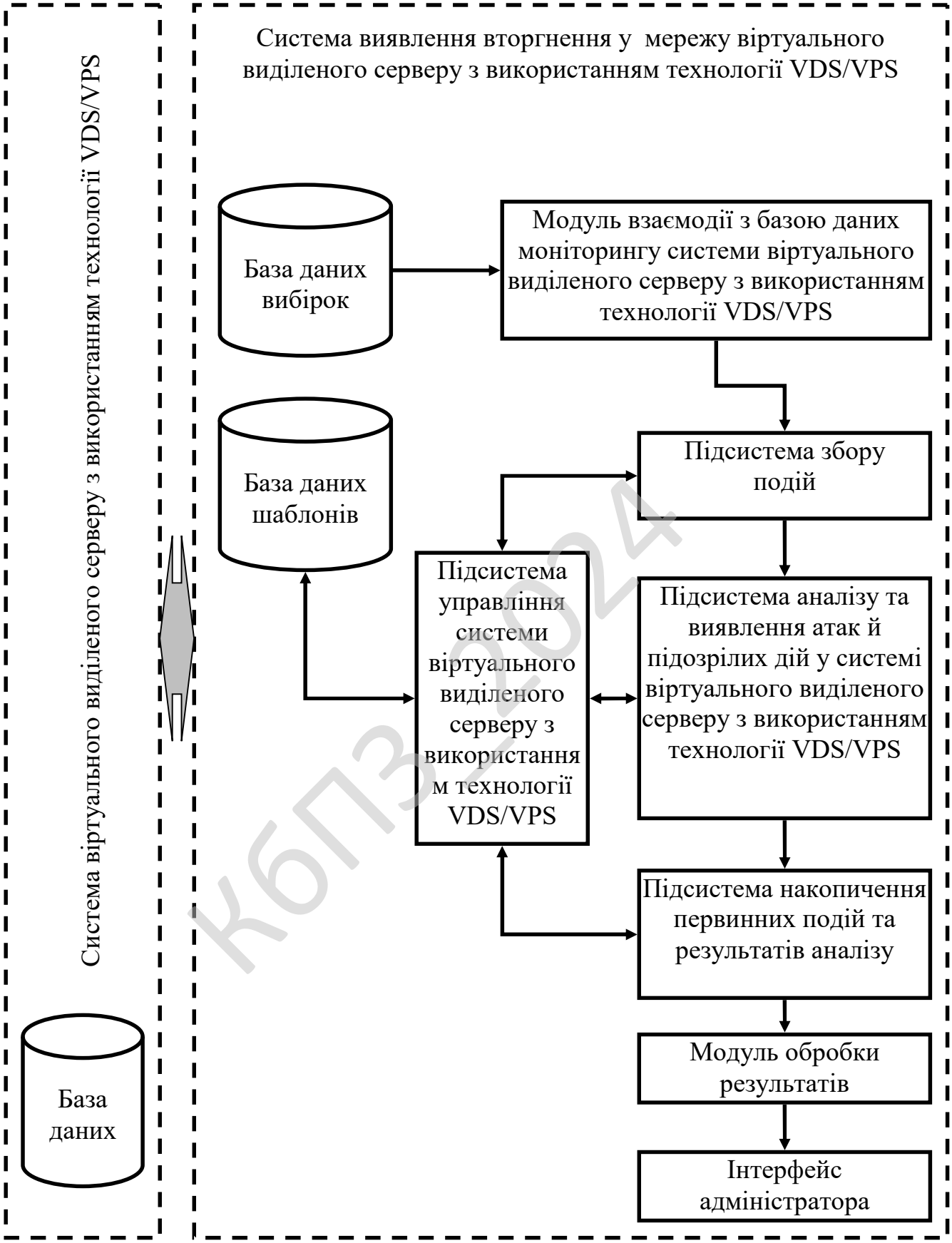
– Ніколи не відповідати на спам і не переходити за посиланнями, які містяться в ньому. Цим ви тільки підтвердите, що користуєтеся своєю електронною адресою й будете одержувати ще більше спаму.

– Вибираючи собі ім'я електронної пошти варто, за можливості, обирати довге й незручне для вгадування ім'я.

Деякі користувачі час від часу змінюють свою адресу, але це зв'язано з очевидними труднощами: потрібно повідомити нову адресу людям, від яких ви хочете одержувати пошту. Існує програмне забезпечення (ПЗ) для автоматичного визначення спаму (т.зв. фільтри). Воно може застосовуватися кінцевими користувачами або на серверах. Це ПЗ має два основні підходи. Перший полягає в аналізі змісту листа на основі чого робиться висновок, спам це чи ні. Якщо лист класифікований як спам, він може бути позначений, переміщений в іншу папку або навіть вилучений. Таке ПЗ може працювати як на сервері, так і на комп'ютері клієнта. При такому підході ви не бачите відфільтрованого спаму, але продовжуєте повністю платити витрати, пов'язані з його прийомом, тому що антиспамне ПЗ в будь-якому випадку одержує кожен спамерський лист (затрачаючи ваші гроші), а тільки потім вирішує показувати його чи ні. Другий підхід базується на класифікації відправника як спамера, не заглядаючи в текст листа.

Для визначення застосовуються різні методи. Це ПЗ може працювати тільки на сервері, який безпосередньо приймає пошту. При такому підході можна зменшити витрати – гроші витрачаються тільки на спілкування зі спамерскими поштовими програмами (тобто на відмову приймати листи) і звертання до інших серверів (якщо такі потрібні) при перевірці. Виграш, однак, не такий великий, як можна було б очікувати. Якщо одержувач відмовляється прийняти лист, спамерська програма намагається обійти захист і відправити його іншим способом. Кожну таку спробу доводитися відбивати окремо, що збільшує навантаження на сервер.

КБПЗ_2024



КБПЗ-2024

Рисунок 3.2 – Функціональна схема системи

Місце встановлення антиспамного ПЗ (комп'ютер кінцевого користувача чи поштовий сервер, наприклад, провайдера) визначає те, хто буде платити витрати, пов'язані з фільтрацією спаму. Якщо спам фільтрує кінцевий користувач, то він і буде платити витрати (а можливо й провайдер, якщо пошта «безкоштовна»), тому що буде змушений одержувати всі повідомлення, включно зі спамом. Якщо спам фільтрує сервер, то користувач не несе витрат, тому що одержує тільки корисну кореспонденцію, а усі витрати лягають на власника сервера. В даний час використовується кілька методів фільтрації електронної пошти.

Авторизація поштових серверів

Були запропоновані різні способи для підтвердження того, що комп'ютер, що відправляє лист, дійсно має на це право (Sender ID, SPF, Caller ID, Yahoo DomainKeys), але вони поки не розповсюджені.

Статистичні методи фільтрації спаму

Ці методи використовують статистичний аналіз змісту листа для прийняття рішення, чи є він спамом. Найбільшого успіху удалося досягти за допомогою алгоритмів, заснованих на теоремі Байеса. Для роботи цих методів потрібно «навчання» фільтрів, тобто потрібно використовувати розсортовані вручну листи для виявлення статистичних особливостей нормальних листів і спаму. Після навчання на досить великій вибірці, вдається розпізнати до 95-97 % спаму.

Сірі списки

Метод сірих списків базується на тому, що «поведінка» програмного забезпечення, призначеного для розсилання спаму відрізняється від поведінки звичайних поштових серверів, а саме, спамерські програми не намагаються повторно відправити лист при виникненні тимчасової помилки, як того вимагає протокол SMTP. Спочатку всі невідомі сервери заносяться в "сірий список" і листи від них не приймаються. Серверові відправника повертається код тимчасової помилки, тому, звичайні листи (не спам) не втрачаються, а тільки затримується їхня доставка (вони залишаються в черзі на сервері відправника і

КОПЗ 2024

доставляються при наступній спробі). Якщо сервер поводиться так, як очікувалося, він автоматично переноситься в білий список і наступні листи приймаються без затримки. Цей метод в наш час дозволяє відсіяти до 90 % спаму, практично без ризику втратити важливі листи. Однак він теж не бездоганний.

– Можуть помилково відсіватися листи з серверів, які не виконують рекомендації протоколу SMTP, наприклад, розсилки з сайтів новин.

– Затримка при доставці листа може досягати півгодини (а іноді й більше), що неприйнятно для термінової кореспонденції. Цей недолік компенсується тим, що затримка вноситься тільки при відправці першого листа з раніше невідомої адреси.

– Великі поштові служби використовують кілька серверів, з різними IP-адресами, більш того, можлива ситуація, коли кілька серверів по-черзі намагаються відправити той самий лист. Це може привести до дуже великих затримок при доставці листів.

– Спамерські програми можуть удосконалюватися. Підтримка повторної посилки повідомлення реалізується досить легко і цілком нівелює даний вид захисту.

Чорні списки

У чорні списки заносяться IP-адреси комп'ютерів, про які відомо, що з них ведеться розсилання спаму. Також широко використовуються списки комп'ютерів, які можна використовувати для розсилання – «відкриті релеї» і «відкриті проксі», а також – списки «діалапів» – клієнтських адрес, на яких не може бути поштових серверів. Можна використовувати локальний список або список який підтримує хтось інший. Завдяки простоті реалізації, широке поширення одержали чорні списки, запит до яких здійснюється через службу DNS. Вони називаються DNSBL (DNS Black List). В наш час цей метод не дуже ефективний. Спамери знаходять нові комп'ютери для своїх цілей швидше, ніж їх встигають заносити в чорні списки. Крім того, кілька комп'ютерів, що відправляють спам, можуть скомпрометувати весь поштовий домен і тисячі законослухняних користувачів на невизначений час будуть позбавлені

КБПЗ_2024

можливості відправляти пошту серверам, що використовують такий чорний список.

Usenet

Багато груп новин Usenet, особливо немодеровані, втратили користувачів і зараз містять переважно рекламу, часто навіть не за темою.

Замість інших були створені модеровані конференції.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування). Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи.

Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється. Діаграма процесів розробленої системи зображена на рисунку 3.3. При детальному її розгляді можна побачити як саме проходе взаємодія у розробленій системі. Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграми потоків даних містять чотири типи елементів:

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

КБПЗ_2024

Рисунок 3.3 – Діаграма взаємодії процесів

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

КБПЗ_2024

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Блок-схеми є основою ПЗ. Тому від точності і детальності проробки блок-схеми залежить результат всієї програми.

При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови програмування і інших технічних засобів, програма буде об’єктно-орієнтована що вимагає оптимізації, також те, що при розробці програми слід надати особливу увагу модулю виділеного серверу з використанням технології VDS/VPS.

Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні блоки можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підпрограми та останньої стадії – перевірки поточного стану та поверненням на початок схеми чи з завершенням роботи розробленого ПЗ.

КБПЗ_2024

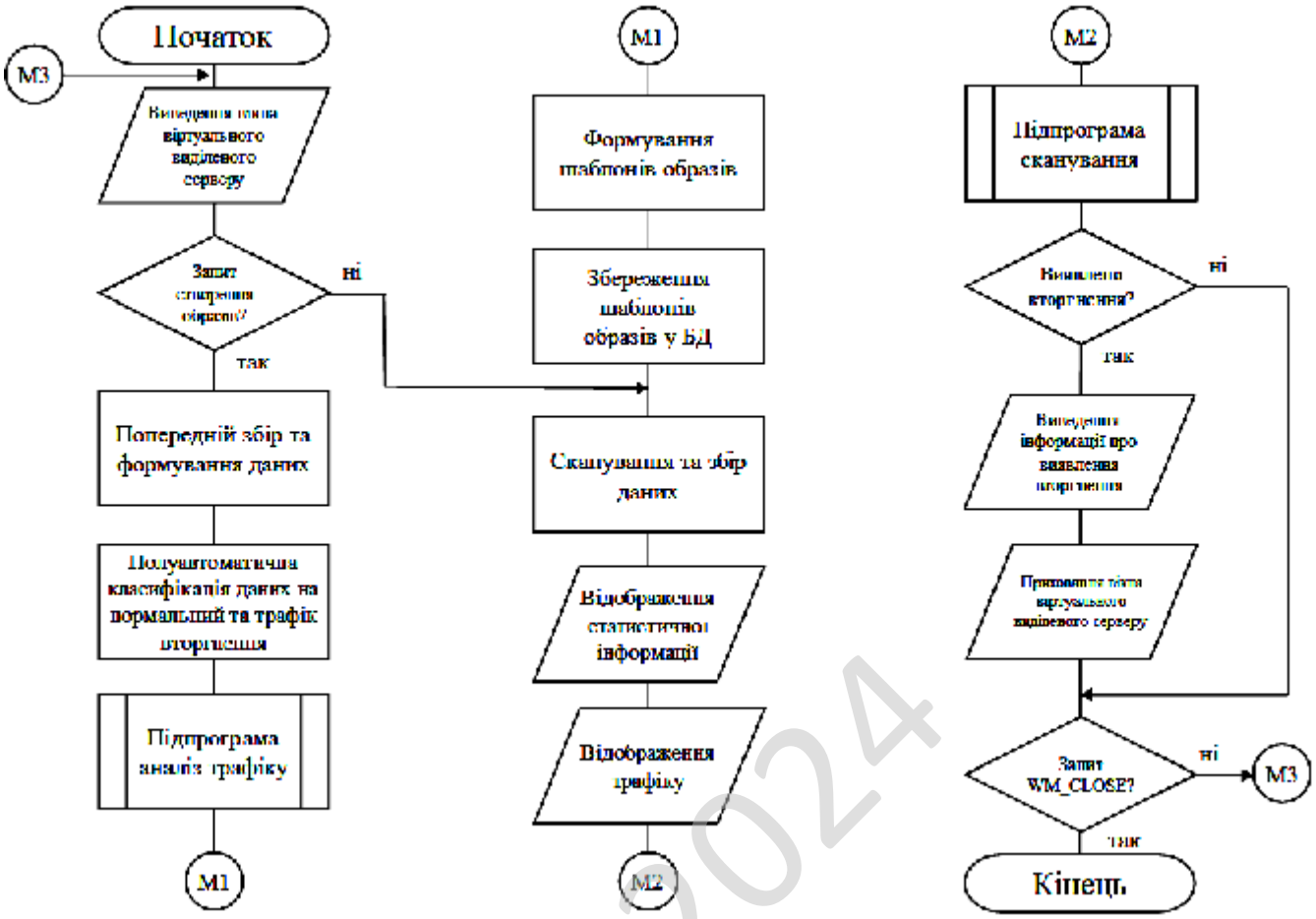


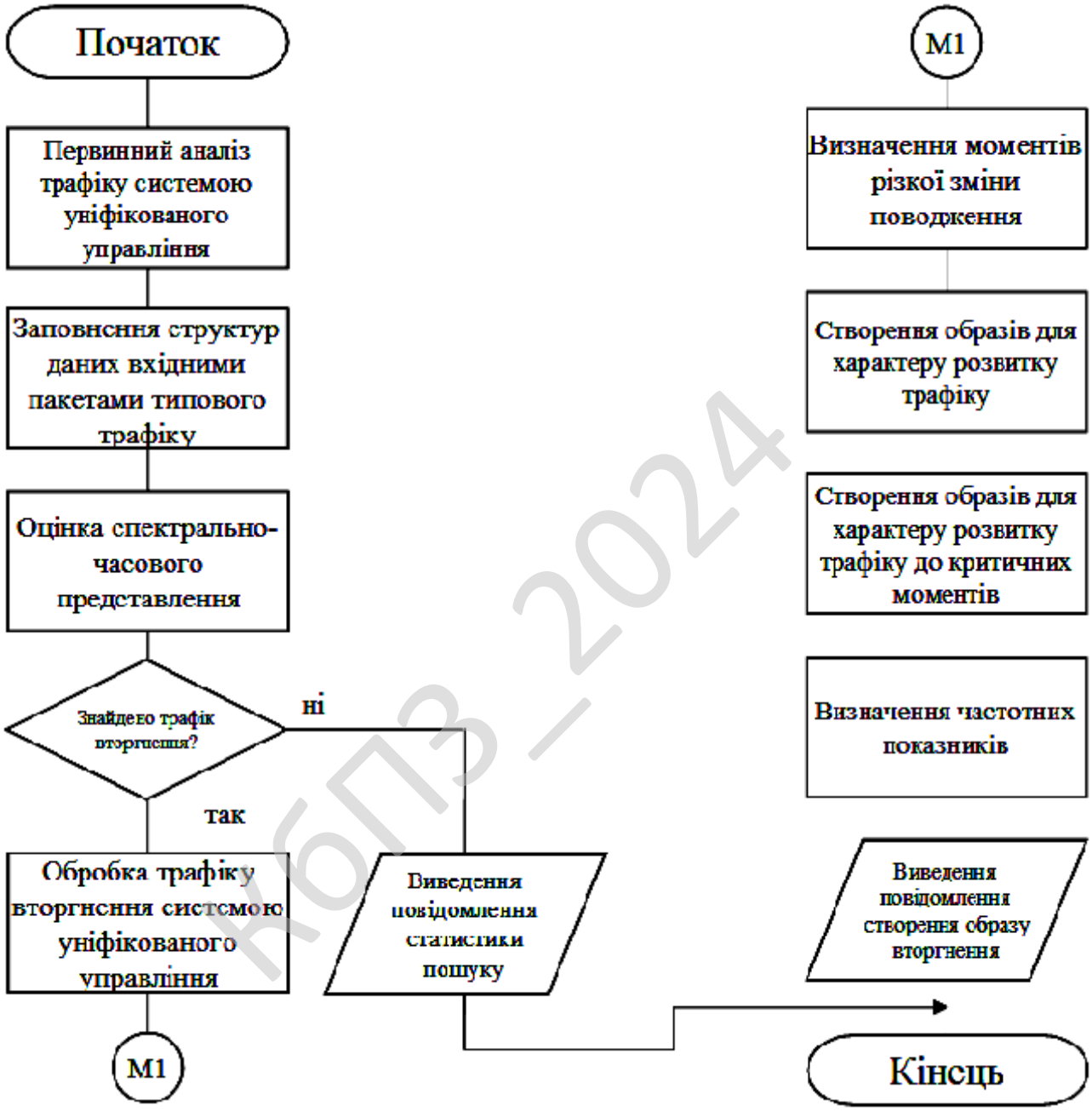
Рисунок 4.1 – Блок-схема основної програми

При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Було використано підходи з використанням UML, це уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю. UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

КБПЗ – 2024

Розглянемо використані технології та їх основні компоненти що підтверджують правильність використаних проектних рішень.



КбПЗ_2024

Рисунок 4.2 – Блок-схема роботи підпрограми

Управління вимогами це процес запису, аналізу, трасування, пріоритезації і узгодження вимог та контролю змін і доведення до їх зацікавлених

сторін. Це безперервний процес протягом всього життя проекту. Вимога – якість, якій мають відповідати результати проекту (продукту або послуги).

Мета управління вимогами полягає в тому, щоб переконатися, що організація відповідає потребам і очікуванням своїх клієнтів, внутрішніх або зовнішніх зацікавлених сторін. Управління вимогами починається з аналізу і виявлення цілей і обмежень організації. Управління вимогами додатково включає в себе підтримку планування вимог, інтеграції вимог і організації роботи з ними (атрибути для вимог).

Управління вимогами передбачає спілкування між членами проектної групи і зацікавленими сторонами, і адаптацію до змін у вимогах протягом всього проекту. Щоб запобігти перетину поля одного класу вимог з іншим, постійні зв'язки між членами команди розробників є критичними. Наприклад, при розробці програмного забезпечення для внутрішнього використання у бізнесу можуть бути настільки сильні потреби, що він може проігнорувати вимоги користувачів, або вважати, що створені сценарії використання покриють також і користувальницькі вимоги.

Відслідковування вимоги фактично означає документування всього життєвого циклу вимоги. Часто необхідно дізнатися першоджерело кожної вимоги. Для цього всі зміни вимог повинні бути задокументовані, щоб досягти стану повного відстеження. Відстежувати треба бути навіть використання реалізованих вимог.

Вимоги мають різні джерела, такі як ділова людина, що замовляє продукт, менеджер зі збуту і фактичний користувач. У всіх цих людей є різні вимоги до продукту. Використовуючи відслідковування вимог, реалізована в системі функція може бути простежена назад до людини або групі, яка замовляла її під час збору вимог. Ця особливість може, наприклад, використовуватися в процесі розробки для пріоритезації вимог, визначаючи, наскільки цінною є дана вимога для певного користувача.

КБПЗ_2024

Відслідковування може також використовуватися після розгортання продукту. Наприклад, коли вивчення використання системи показує, що якась функція не використовується, можна визначити навіщо вона була потрібна спочатку.

Завдання управління вимогами

На кожному етапі процесу розробки існують ключові методи і задачі пов'язані з управлінням вимогами. Для ілюстрації, розглянемо наприклад стандартний процес розробки з п'ятьма фазами: дослідженням, аналізом здійсненності, дизайном, розробкою та тестуванням і випуском.

Дослідження. Під час фази дослідження збираються перші три класи вимог від користувачів, бізнесу і команди розробників. У кожній області задають однакові питання: які цілі, які обмеження, які використовуються процеси та інструменти і так далі. Тільки коли ці вимоги добре зрозумілі, можна приступати до розробки функціональних вимог.

Тут необхідне застереження: незалежно від того, як сильно група намагається це зробити, вимоги не можуть бути повністю визначені на початку проекту. Деякі вимоги змінюються, або тому що вони просто не були знайдені спочатку, або тому що внутрішні чи зовнішні сили торкаються проекту в середині циклу. Таким чином, учасники групи повинні спочатку погодитися, що головна умова успіху – гнучкість у мисленні та діях.

Результатом стадії дослідження є документ – специфікація вимог, схвалений усіма членами проекту. Пізніше, в процесі розробки, цей документ буде важливий для запобігання розповзання меж проекту або непотрібних змін. Оскільки система розвивається, кожна нова функція відкриває світ нових можливостей, таким чином специфікація вимог прив'язує команду до оригінального бачення системи і дозволяє контрольоване обговорення змін.

У той час як багато організацій все ще використовують звичайні документи для керування вимогами, інші управляють своїми базовими вимогами, використовуючи програмні інструменти.

КБПЗ-2024

Ці інструменти керують вимогами використовуючи базу даних, і зазвичай мають функції автоматизації відстеження (наприклад, дозволяючи створювати зв'язки між батьківськими і дочірніми вимогами, або між тестами і вимогами), управління версіями, і управління змінами. Зазвичай такі інструментальні засоби містять функцію експорту, яка дозволяє створювати звичайний документ, експортуючи дані вимог.

Аналіз здійсненності

На стадії аналізу здійсненності визначається вартість вимог. Для користувальницьких вимог поточна вартість роботи порівнюється з майбутньою вартістю встановленої системи. Задаються питання такі як: «Скільки нам зараз варті помилки введення даних?» Або, «Яка вартість втрати даних через помилки оператора пов'язаної з використовуваним інтерфейсом?». Фактично, потреба в новому інструменті часто розпізнається, коли подібні питання потрапляють до уваги людей, що займаються в організації фінансами.

Ділова вартість включає відповіді на такі питання як: «У якого відділу є бюджет на це?» «Який рівень повернення коштів від нового продукту на ринку?» «Який рівень скорочення внутрішніх витрат на навчання і підтримку, якщо ми зробимо нову, більш просту в використанні систему?»

Технічна вартість пов'язана з вартістю розробки програмного забезпечення та апаратною вартістю. «Чи є у нас потрібні люди, щоб створити інструмент?» «Чи потребуємо ми нове устаткування для підтримки нової системи?»

Подібні питання дуже важливі. Група повинна з'ясувати, чи буде новий автоматизований інструмент мати достатню ефективність аби перенести частину тягаря користувачів на систему і зекономити час людей.

Ці питання також вказують на основну суть управління вимогами. Людина і інструмент формують систему, і це розуміння особливо важливе, якщо інструмент – комп'ютер або новий додаток на комп'ютері. Людський розум вкрай ефективний у паралельній обробці та інтерпретації тенденцій з недостатніми даними. Комп'ютерний процесор ефективний у послідовній обробці і точному

КБПЗ-2024

математичному обчисленні. Основна мета управління вимогами для програмного проекту полягала б у тому, щоб гарантувати, що автоматизована робота призначена «правильному» процесору.

Наприклад, «не змушуйте людину пам'ятати, де вона знаходиться в системі. Примусьте інтерфейс завжди повідомляти про місцезнаходження людини в системі ». Або «не змушуйте людини вводити ті ж самі дані в два екрани. Примусьте систему зберігати дані і заповнювати їх де необхідно автоматично». Результатом стадії аналізу здійсненності є бюджет і графік проекту.

Дизайн. Припускаючи, що вартість точно визначена і переваги, які будуть отримані, є досить великими, проект може перейти до стадії проектування.

На стадії дизайну основна діяльність управління вимогами полягає в тому, щоб перевіряти чи відповідають результати дизайну документу вимог, щоб упевнитися, що робота залишається в межах проекту.

І знову, гнучкість є ключем до успіху. Ось класичний приклад змін проекту, які відмінно працювали. Проектувальники Форда на початку 1980-х очікували, що ціни на бензин піднімуться до 3,18 дол за галон до кінця десятиліття. На середині процесу дизайну автомобіля Ford Taurus, ціни встановилися приблизно на рівні 1,50 дол за галон. Колектив дизайнерів вирішив, що вони могли б створити більший, більш зручний, і більш потужний автомобіль, якщо б ціни на бензин залишилися низькими. Таким чином, вони перепроектувати автомобіль. Коли новий автомобіль вийшов, він встановив загальнонаціональні рекорди продажів.

У більшості випадків, однак, відступ від оригінальних вимог до такої міри не працює. Таким чином документ вимог стає ключовим інструментом, який допомагає команді приймати рішення про зміни дизайну.

Розробка та тестування. На стадії розробки і тестування, основна діяльність управління вимогами – це гарантувати, що робота і ціна залишаються в межах графіка і бюджету, і що створюваний інструмент дійсно відповідає вимогам. Основним інструментом, використовуваним на цій стадії, є створення

КБПЗ-2024

прототипу і ітераційне тестування. Для програмного додатка користувацький інтерфейс може бути створений на папері і перевірений з потенційними користувачами, в той час як створюється основа програми. Результати цих тестів записуються в керівництві по дизайну користувацького інтерфейсу і передаються колективу дизайнерів. Це економить їх час і робить їх завдання набагато простіше.

Щоб створити систему віртуального виділеного серверу (VDS/VPS) на Python, можна розглянути архітектуру, яка включає наступні модулі.

1. Модуль управління ресурсами.

Цей модуль відповідальний за динамічне розподілення ресурсів сервера між віртуальними машинами (CPU, пам’ять, диск, мережевий трафік). Він використовує бібліотеки системного рівня для контролю контейнерів або гіпервізорів. Основні інструменти це бібліотеки psutil для моніторингу та subprocess для взаємодії з командним рядком.

2. Модуль контейнеризації або віртуалізації.

Контейнери, створені за допомогою таких інструментів, як Docker, або інші віртуалізаційні рішення (KVM, QEMU), забезпечують ізольоване середовище для кожного віртуального серверу. У Python інтеграція з Docker може здійснюватися за допомогою бібліотеки docker-py.

3. Модуль управління мережею.

Цей модуль відповідає за маршрутизацію, NAT, та балансування мережевих з’єднань для кожної віртуальної машини. Використовує інструменти iptables, firewalld для налаштування брандмауера та мережевих правил.

4. Модуль адміністрування.

Це API або CLI для адміністрування віртуальних машин: створення, налаштування параметрів ресурсів, зупинка, запуск, перезапуск. Рекомендується використовувати REST API або gRPC для взаємодії з сервером.

```
import psutil
import subprocess
import docker
import os
```

КБПЗ_2024

```
import json

# Ініціалізація Docker-клієнта
client=docker.from_env()

# Функція створення віртуального сервера
def create_vps(container_name, cpu_quota, mem_limit):
    client.containers.run(
        "ubuntu:latest",
        name=container_name,
        detach=True,
        cpu_quota=cpu_quota * 1000, # Розподіл CPU
        mem_limit=f"{mem_limit}m" # Розподіл пам'яті
    )
    print(f"VPS {container_name} успішно створений")

# Функція зупинки віртуального сервера
def stop_vps(container_name):
    container=client.containers.get(container_name)
    container.stop()
    print(f"VPS {container_name} зупинено")

# Функція видалення віртуального сервера
def delete_vps(container_name):
    container=client.containers.get(container_name)
    container.remove()
    print(f"VPS {container_name} видалено")

# Функція моніторингу ресурсів сервера
def monitor_resources():
    cpu_usage=psutil.cpu_percent()
    mem_info=psutil.virtual_memory()
    net_info=psutil.net_io_counters()
    return {
        "cpu_usage": cpu_usage,
        "memory_usage": mem_info.percent,
        "network_sent": net_info.bytes_sent,
        "network_received": net_info.bytes_recv
    }

# Функція розподілу мережевих налаштувань для VPS
def configure_network(container_name, port):
```

КБПЗ_2024

```
        cmd=f"iptables -A INPUT -p tcp --dport {port} -j ACCEPT"
        subprocess.run(cmd, shell=True, check=True)
        print(f"Мережевий порт {port} налаштований для {container_name}")

    # Головна функція запуску системи
    def main():
        # Ініціалізація контейнера з параметрами CPU та пам'яті
        create_vps("vps1", cpu_quota=2, mem_limit=1024)

        # Моніторинг ресурсів
        resources=monitor_resources()
        print("Моніторинг ресурсів", json.dumps(resources, indent=4))

        # Конфігурування мережі
        configure_network("vps1", 8080)

        # Зупинка та видалення VPS після тесту
        stop_vps("vps1")
        delete_vps("vps1")

    if __name__ == "__main__":
        main()
```

Основні розрахунки та параметри.

1. Розподіл CPU.

Розрахунок CPU для кожного контейнера (на основі cpu_quota) визначає відсоток доступного часу CPU. Для обмеження CPU-квоти для процесу використовуються значення у мілісекундах.

2. Розподіл пам'яті.

mem_limit задає максимальний обсяг пам'яті для контейнера, і забезпечує оптимальне управління обсягом оперативної пам’яті.

3. Конфігурація мережі.

Для кожного порту використовуються правила маршрутизації (iptables) для ізоляції мережевих з'єднань між VPS, що забезпечує безпечний доступ до сервісів.

Ця архітектура забезпечує гнучкість і масштабованість для середовища VDS/VPS.

КБПЗ_2024

4.2 Захист розробленого програмного забезпечення

Дані у системі захищаються за допомогою алгоритму обміну ключа Діффі-Хеллмана.

Ціль алгоритму полягає в тому, щоб два учасники могли безпечно обмінятися ключем, що надалі може використовуватися в якому-небудь алгоритмі симетричного шифрування. Сам алгоритм Діффі-Хеллмана може застосовуватися тільки для обміну ключами.

Алгоритм заснований на труднощі обчислень дискретних логарифмів. Дискретний логарифм визначається в такий спосіб. Уводиться поняття примітивного кореня простого числа Q як числа, чиї ступені створюють всі цілі від 1 до Q – 1. Це означає, що якщо A є примітивним коренем простого числа Q, тоді числа:

$$A \bmod Q,\ A^2 \bmod Q,\ \ldots,\ A^{Q-1} \bmod Q,$$

є різними й складаються із цілих від 1 до Q – 1 з деякими перестановками. У цьому випадку для будь-якого цілого $Y < Q$ і примітивного кореня A простого числа Q можна знайти єдину експоненту X, таку, що:

$$Y = A^X \bmod Q,$$

де $0 \le X \le (Q - 1)$.

Експонента X називається дискретним логарифмом, або індексом Y, по підставі A mod Q. Це позначається як $\mathrm{ind}_{,Q}(Y)$.

Тепер опишемо алгоритм обміну ключів Діффі-Хеллмана.

Загальновідомі елементи

Q – просте число.

A – $A < Q$ і A є примітивним коренем Q.

Створення пари ключів користувачем I

Вибір випадкового числа X_i (закритий ключ):

$$X_i < Q.$$

Обчислення числа Y_i (відкритий ключ):

КБПЗ-2024

$$Y_i = A^{Xi} \bmod Q.$$

Створення відкритого ключа користувачем J

Вибір випадкового числа X_j (закритий ключ):

$$X_j < Q.$$

Обчислення випадкового числа Y_j (відкритий ключ):

$$Y_j = A^{Xj} \bmod Q.$$

Створення загального секретного ключа користувачем I

$$K = (Y_j)^{Xi} \bmod Q.$$

Створення загального секретного ключа користувачем J

$$K = (Y_i)^{Xj} \bmod Q.$$

Передбачається, що існують два відомих усім числа: просте число Q і ціле A, що є примітивним коренем Q.

Тепер припустимо, що користувачі I і J хочуть обмінятися ключем для алгоритму симетричного шифрування.

Користувач I вибирає випадкове число $X_i < Q$ і обчислює:

$$Y_i = A^{Xi} \bmod Q.$$

Аналогічно користувач J незалежно вибирає випадкове ціле число $X_j < Q$ і обчислює:

$$Y_j = A^{Xj} \bmod Q.$$

Кожна сторона тримає значення X у секреті й робить значення Y доступним для іншої сторони.

Тепер користувач I обчислює ключ як $K = (Y_j)^{Xi} \bmod Q$, і користувач J обчислює ключ як $K = (Y_i)^{Xj} \bmod Q$. У результаті обоє одержать те саме значення:

$$K = (Y_j)^{Xi} \bmod Q = (A^{Xj} \bmod Q)^{Xi} \bmod Q = (A^{Xj})^{Xi} \bmod Q =$$

за правилами модульної арифметики

$$= A^{Xj\,Xi} \bmod Q = (A^{Xj})^{Xj} \bmod Q = (A^{Xi} \bmod Q)^{Xj} \bmod Q = (Y_i)^{Xj} \bmod Q$$

Таким чином, дві сторони обмінялися секретним ключем. Так як X_i і X_j є закритими, супротивник може одержати тільки наступні значення: Q, A, Y_i і Y_j.

КбПЗ_2024

Для обчислення ключа атакуючий повинен зламати дискретний логарифм, тобто обчислити:

$$X_j = ind_{a,\,q}\,(Y_j).$$

Безпека обміну ключа в алгоритмі Діффі-Хеллмана випливає з того факту, що, хоча відносно легко обчислити експоненти по модулю простого числа, дуже важко обчислити дискретні логарифми. Для великих простих чисел задача вважається нерозв'язною.

Варто помітити, що даний алгоритм уразливий для атак типу "in-the-middle". Якщо супротивник може здійснити активну атаку, тобто має можливість не тільки перехоплювати повідомлення, але й заміняти їх іншими, він може перехопити відкриті ключі учасників Y_i і Y_j , створити свою пару відкритого й закритого ключа ($X_{оп}$, $Y_{оп}$) і послати кожному з учасників свій відкритий ключ. Після цього кожний учасник обчислить ключ, що буде загальним із супротивником, а не з іншим учасником. Якщо немає контролю цілісності, то учасники не зможуть виявити подібну підміну.

КБПЗ_2024

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розглянемо розроблене ПЗ віртуального виділеного серверу з використанням технології VDS/VPS яке зображено на рисунку 5.1. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні функціональні блоки:

– Навігаційне меню: Файл; Дані; Додатки; Налаштування; Довідка.

– Блоку виведення статистичних даних.

– Вікно виведення графічних даних.

– Навігаційного меню яке викликається натисканням правої клавіші маніпулятора миші.

– Функціональних кнопок ПЗ.

Віртуальний сервер працює незалежно від інших, розташованих на одній з ним хост-машині. Це значить, що збій у роботі одного сервера не впливає на функціонування «сусідів».

На відміну від звичайного віртуального (shared) хостингу, на віртуальному сервері доступні технології захисту від DDoS-атак, а схоронність даних гарантується можливістю створення резервної копії.

Переваги використання VDS/VPS у порівнянні з виділеним сервером:

– Невисока вартість: оренда віртуального виділеного сервера обходиться користувачеві значно дешевше оренди порівнянного по потужності фізичного сервера. При цьому продуктивність VDS / VPS може навіть виявитися вище (дивися наше дослідження).

– Проста масштабованість: віртуальному серверу дуже легко додати потужності у вигляді додаткового обсягу оперативної пам'яті, процесорних ядер або обсягу жорсткого диска.

КБПЗ_2024

Недоліки використання VDS/VPS сервера проти виділеного сервера:

– Загальна дискова система й канал зв'язку: всі віртуальні сервери, що працюють на одній хост-машині, за умови поділу основних ресурсів використовують одну на всіх дискову систему й загальний високошвидкісний канал зв'язку. Це означає, що деякі користувачі можуть зловживати використанням ресурсів і заважати іншим клієнтам.

Ми використовуємо автоматичну систему, що знаходить таких користувачів. До порушників застосовуються міри від попередження, до вимикання VDS.

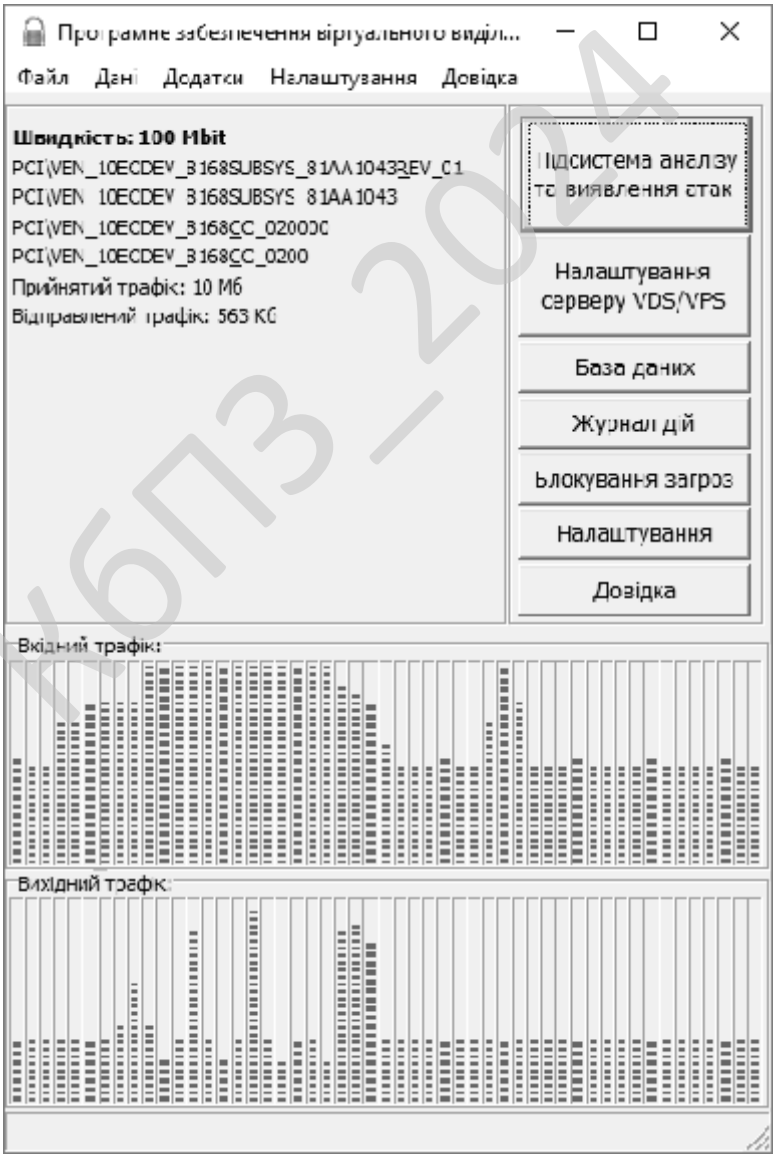


Рисунок 5.1 – Головне вікно ПЗ

ЖБПЗ_2024

Розроблена програма має дуже простий і інтуїтивно зрозумілий інтерфейс з користувачем. Кожен, хто в достатньому обсязі володіє операційним середовищем Windows без особливих складностей освоїть і цю програму, оскільки її інтерфейс інтуїтивно зрозумілий.

Якщо програма не видала ніяких помилок, і працює, то можна використовувати, інакше слід слідувати інструкціям, які пропонує програма.

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення.

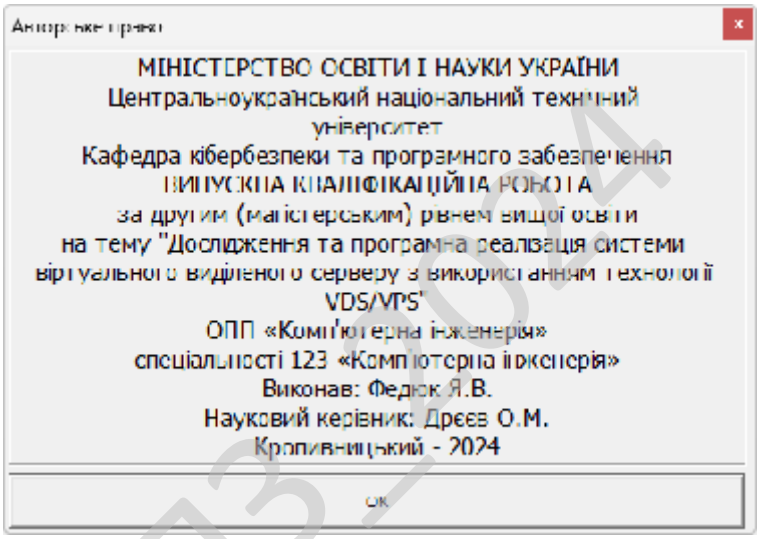


Рисунок 5.2 – Авторське право

Розглянемо процес впровадження програмного забезпечення, це процес налаштування програмного забезпечення під певні умови використання, а також навчання користувачів роботі з програмним продуктом. Впровадження програмного забезпечення це усі дії, що роблять розроблену програмну систему готовою до використання. Даний процес є частинною життєвого циклу програмного забезпечення.

Загалом процес розгортання складається з кількох взаємопов'язаних дій із можливими переходами між ними. Ця активність може відбуватися як з боку виробника так і з боку споживача. Оскільки кожна програмна система є

КБПЗ – 024

унікальною, то усі процеси та процедури під час розгортання важко передбачити. Тому, "розгортання" можна трактувати як загальний процес відповідно до певних вимог та характеристик. Розгортання може здійснюватись програмістом і в процесі розробки програмного забезпечення.

До діяльностей пов'язаних із розгортанням програмного забезпечення відносять:

– Випуск.

– Встановлення та активація.

– Деактивація.

– Адаптація.

– Обновлення.

– Вмонтування.

– Відстежування версій.

– Видалення.

– Вилучення з обігу.

При впровадженні програмного забезпечення потрібно урахувати наступні дії:

– Виділення критичних, з точки зору загального результату, процедур в діяльності організації. Коли набір таких процедур визначений, необхідно в першу чергу використовувати IT рішення для автоматизації операцій усередині саме цих процедур. Таким чином, розроблене IT рішення автоматично стає життєво важливим і затребуваним для організації, а також буде забезпечена публічність процесу впровадження;

– Розширення нормативної бази організації шляхом включення до неї регламентів, що описують порядок виконання процедур автоматизованих процесів. В іншому випадку є небезпека виникнення неузгодженості між автоматизованими процедурами та іншими процесами організації.

– Виконання робіт з загальної стандартизації існуючої діяльності організації, коли виділяються кращі практики виконання процедур і включаються

КБПЗ_2024

в IT рішення за принципом найбільшої корисності для більшості учасників. Відсоток таких процедур щодо загального обсягу автоматизації може бути невеликий, але це надає процесу побудови рішення вагу в організації за рахунок збільшення його необхідності.

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

– відповідності поставленим вимогам;

– правильна відповідь для усіх можливих вхідних даних;

– виконання функцій за прийнятний час;

– практичність;

– сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Проводилось тестування форматом білої скриньки засноване на аналізі керуючої структури програми. Програма вважається повністю перевіреною, якщо проведено вичерпне тестування маршрутів (шляхів) її графа управління.

У цьому випадку формуються тестові варіанти, в яких:

– Гарантується перевірка всіх незалежних маршрутів програми.

– Знаходяться гілки True, False для всіх логічних рішень.

– Виконуються всі цикли (у межах їхніх кордонів та діапазонів).

– Аналізується правильність внутрішніх структур даних.

КПЗ_2024

Недоліки тестування "білої скриньки":

– Кількість незалежних маршрутів може бути дуже велика.

– Повне тестування маршрутів не гарантує відповідності програми вихідним вимогам до неї.

– У програмі можуть бути пропущені деякі маршрути.

– Не можна виявити помилки, поява яких залежить від даних.

Переваги тестування "білої скриньки" пов'язані з тим, що принцип «білої скриньки» дозволяє врахувати особливості програмних помилок:

– Кількість помилок мінімально в «центрі» і максимально на «периферії» програми.

– Попередні припущення про ймовірність потоку керування або даних у програмі часто бувають некоректними. У результаті типовим може стати маршрут, модель обчислень за яким опрацьована слабо.

– При записі алгоритму програмного забезпечення у вигляді тексту на мові програмування можливе внесення типових помилок трансляції (синтаксичних та семантичних).

– Деякі результати в програмі залежать не від вихідних даних, а від внутрішніх станів програми.

Обрано умови розповсюдження – proprietary software.

Програмне забезпечення, на яке зберігаються як немайнові, так і майнові авторські права. Отримавши або придбавши таке програмне забезпечення, користувач отримує обмежені права користування ним: може бути заборонено або закрито доступ до коду (вивчення), внесення змін, тиражування, розповсюдження та перепродаж. Програмне забезпечення вважається власницьким, якщо наявне хоча б одне з перелічених обмежень.

Найчастіше основним методом захисту майнових прав на власницьке ПЗ, поза ліцензійною угодою, власник обирає закриття сирцевого коду, захищаючи свій продукт від модифікації і вбудовуючи системи обмеження користування через авторизацію. Таке програмне забезпечення називається закритим. Проте,

КБПЗ_2024

код власницького продукту може бути і відкритим, але власник може обмежити права користувача умовами користувацької ліцензії.

Власницьке програмне забезпечення та комерційне програмне забезпечення не є синонімами – власницьким може бути і безплатне (тобто, некомерційне) програмне забезпечення.

На противагу власницькому ПЗ існує вільне програмне забезпечення, автори і власники якого дозволяють вивчати, модифікувати і поширювати свій продукт.

Саме визначення власницького програмного забезпечення виникло в результаті діяльності громадського руху вільного програмного забезпечення (представленого Фондом вільного програмного забезпечення та іншими організаціями) і осмислення умов свободи користування програмами. Визначенням власницького програмного забезпечення є не невідповідність хоча б одній з базових умов вільного програмного забезпечення. Сама назва власницьке ПЗ підкреслює визначальне значення власника у способі використання і можливостях розвитку цього програмного забезпечення.

КБПЗ_2024

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи віртуального виділеного серверу з використанням технології VDS/VPS.

Метою розробки є дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

Об'єктом дослідження є процес віртуального виділеного серверу з використанням технології VDS/VPS.

Предметом дослідження є методи віртуального виділеного серверу з використанням технології VDS/VPS.

Методи дослідження базуються на методах теорії віртуалізації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод віртуального виділеного серверу з використанням технології VDS/VPS.

– Розроблено вітчизняний продукт віртуального виділеного серверу з використанням технології VDS/VPS, який має більш широкі можливості, на відміну від існуючих аналогів.

КБПЗ_2024

7 МАРКЕТИНГОВЕ ТА ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ІТ-ПРОЄКТУ

7.1 Визначення цільової аудиторії кінцевого готового продукту

Результати дослідження та програмної реалізації системи віртуального виділеного сервера (VDS/VPS) можуть зацікавити широке коло осіб та організацій (рисунок 7.1).

ІТ-компанії – особливо ті, що займаються наданням послуг хостингу, розробкою хмарних сервісів, та інфраструктурними рішеннями. Вони можуть впровадити вдосконалення або нові підходи для покращення своїх сервісів.
Розробники ПЗ та DevOps-фахівці – для впровадження та адміністрування інфраструктури, яка використовує VDS/VPS технології, особливо для великих проєктів та сервісів, де потрібна масштабованість і висока доступність.
Компанії, що займаються обробкою великих даних або штучним інтелектом – VDS/VPS може допомогти створити віртуальні середовища для обробки даних, тестування моделей, або запуску інтенсивних обчислень.
Стартапи та підприємці – вони можуть використати знання про VDS/VPS для побудови недорогих, масштабованих та гнучких інфраструктур для своїх додатків та сервісів, не витрачаючи значні кошти на фізичні сервери.
Навчальні заклади та наукові установи – для розширення навчальних програм з адміністрування серверів, віртуалізації, та хмарних технологій, а також для власних внутрішніх проєктів, які потребують віртуальних ресурсів.
Фахівці з кібербезпеки – їм цікаві аспекти безпеки VDS/VPS, а також методи захисту даних та управління доступом у віртуалізованих середовищах.
Фінансові та медичні установи – для забезпечення безпеки, масштабованості та розподілу ресурсів у віртуальних середовищах при роботі з чутливими даними.

Рисунок 7.1 – Цільова аудиторія

Направленість на цю цільову аудиторію дозволить мати широку клієнтську базу.

КПІ-2024

7.2 Оцінка привабливості шляхом застосування методів експертних оцінок

Для оцінки привабливості програмної реалізації системи віртуального виділеного сервера (VDS/VPS) можна застосувати метод експертних оцінок з використанням бальної системи. Цей метод передбачає участь групи експертів, які оцінюють різні аспекти розробки на основі певних критеріїв.

Спершу визначимо ключові критерії, за якими буде оцінюватись привабливість системи VDS/VPS (рисунок 7.2).

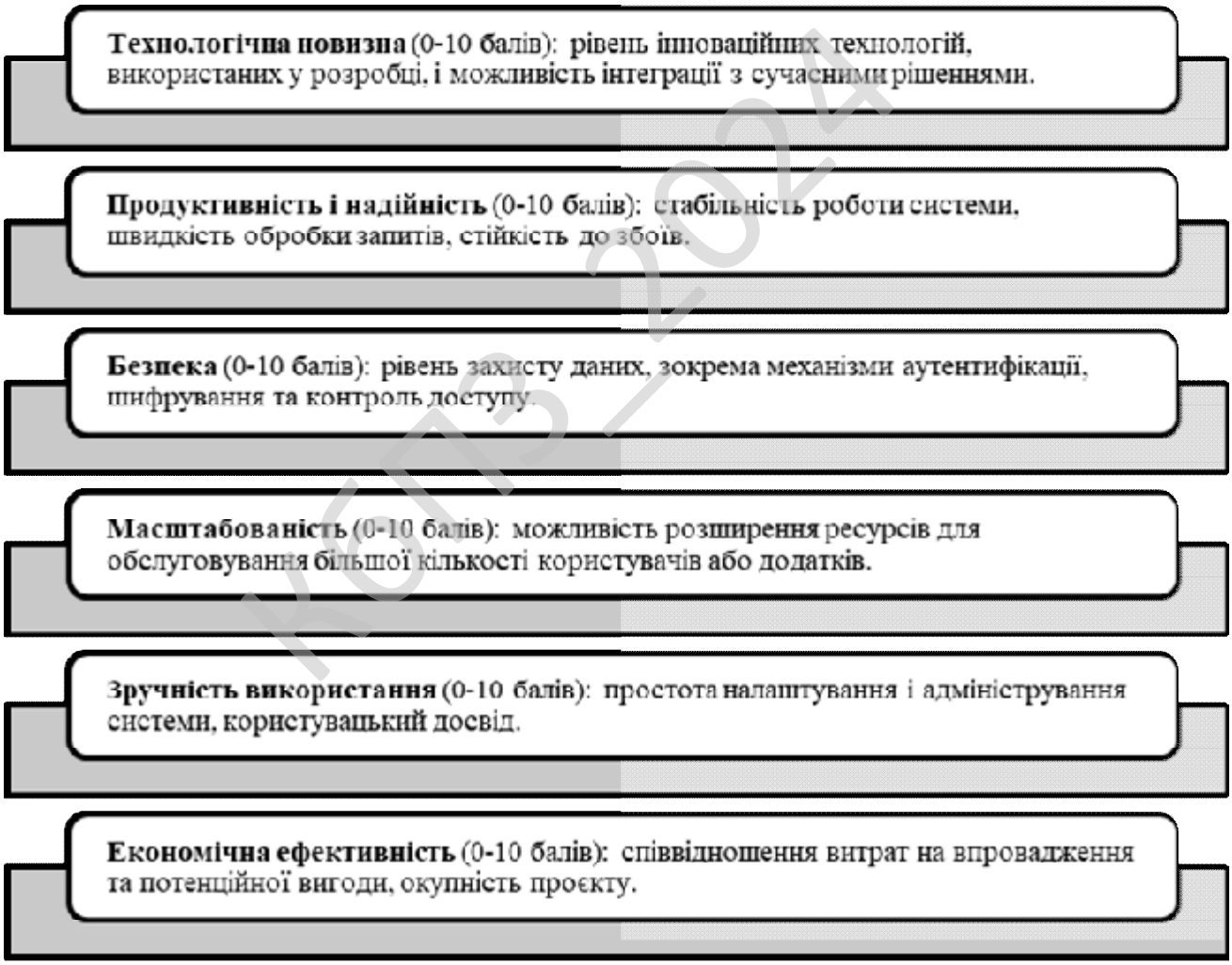


Рисунок 7.2 – Основні критерії оцінки привабливості

КСПЗ 2024

Кожен експерт оцінює проєкт за зазначеними критеріями, виставляючи бали від 0 до 10 для кожного критерію. Результати зводимо в таблицю 7.1.

Таблиця 7.1 – Зведені результати експертних оцінок

Критерій	Експерт 1	Експерт 2	Експерт 3	Експерт 4	Експерт 5	Середній бал
Технологічна новизна	8	7	9	8	7	7.8
Продуктивність і надійність	9	8	9	9	8	8.6
Безпека	8	9	8	8	7	8.0
Масштабованість	7	8	8	7	9	7.8
Зручність використання	6	7	8	7	7	7.0
Економічна ефективність	8	8	9	8	7	8.0

Після підрахунку середніх балів за кожним критерієм, можна вирахувати загальний рейтинг привабливості системи, який є середнім значенням всіх критеріїв: загальний рейтинг=7.8 + 8.6 + 8.0 + 7.8 + 7.0 + 8.06 ≈ 7.87

Загальний рейтинг привабливості 7.87 вказує на те, що система є привабливою для потенційних впроваджень. Найвищі оцінки отримали критерії продуктивності та надійності, а також економічної ефективності, що свідчить про відповідність системи вимогам бізнесу. Можливо, слід додатково попрацювати над зручністю використання для підвищення загального рівня користувацького досвіду. Такий підхід дає комплексну оцінку привабливості проєкту на основі експертної думки, що дозволяє виявити як сильні сторони, так і аспекти, які потребують покращення.

КБПЗ-2024

7.3 Вибір методу оцінки вартості ПЗ

Для оцінки вартості програмної реалізації системи віртуального виділеного сервера з використанням технології VDS/VPS найкраще підійдуть наступні методи:

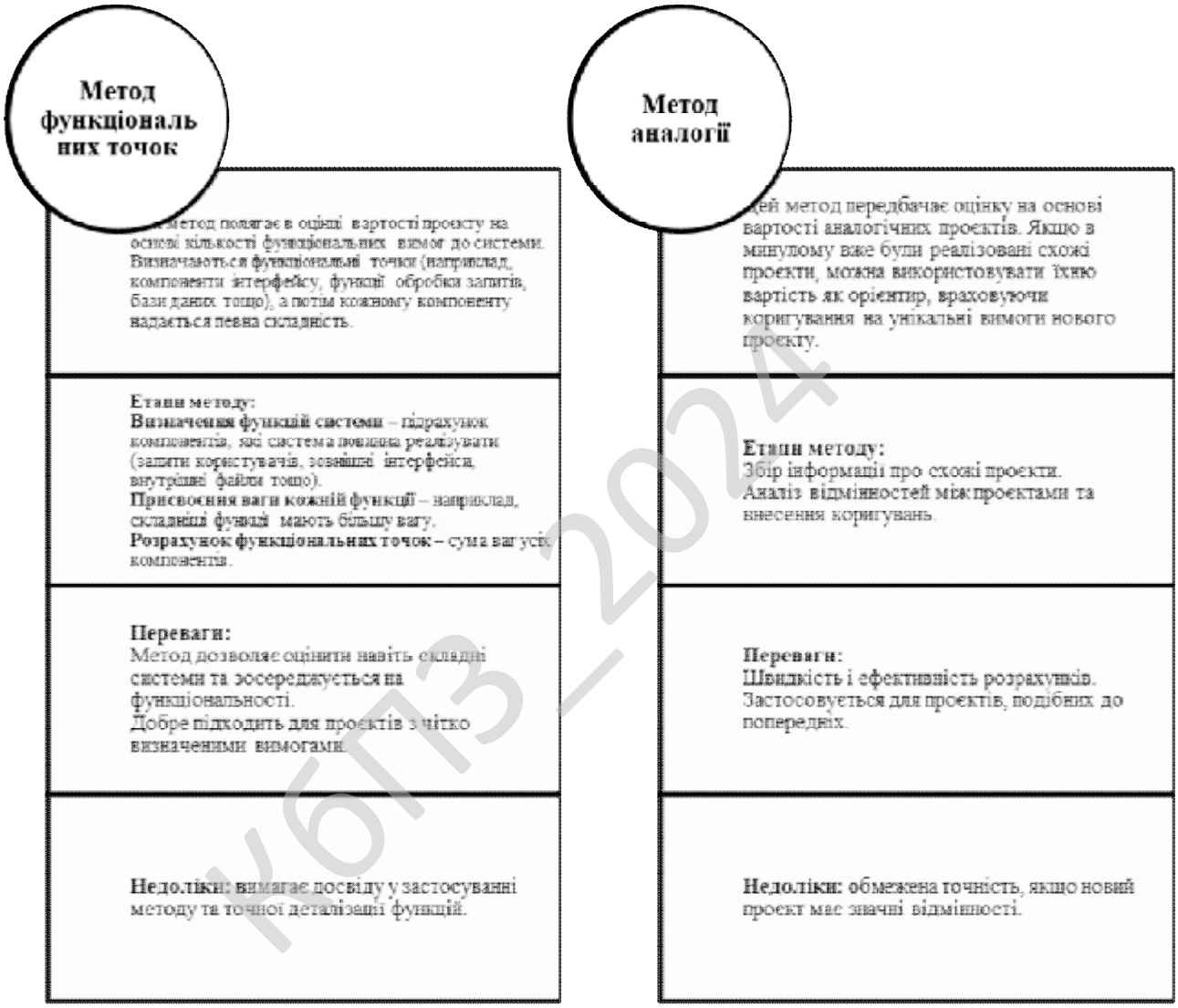


Рисунок 7.3 – Оптимальні методи оцінки вартості ПЗ

Найбільш придатними методами можуть метод функціональних точок, оскільки вони дозволяють розрахувати вартість розробки на основі детальних характеристик і вимог системи. Якщо у вас є схожі проєкти, метод аналогії теж буде ефективним, але з меншим рівнем деталізації.

УКПЗ_2024

7.4 Розрахунок економічної ефективності від впровадження реалізованого ПЗ як фактору його привабливості

Економічна ефективність від впровадження системи віртуального виділеного сервера з використанням технології VDS/VPS може проявлятися у кількох аспектах. Орієнтовні вхідні та розрахункові дані зведено в таблицю.7.2.

Таблиця 7.2 – Основні показники впровадження проєкту

Компанія планує перейти на використання VDS/VPS для свого IT-інфраструктурного проєкту. До впровадження цієї системи компанія використовувала фізичні сервери, які мали такі річні витрати: **Закупівля і обслуговування обладнання**: $50,000 **Оплата за електроенергію** (охолодження та живлення серверів): $20,000 **Оренда приміщень для серверної кімнати**: $10,000 **Технічне обслуговування (зарплата співробітників)**: $15,000 Таким чином, загальні річні витрати до переходу на віртуальні сервери становили **$95,000**.
Витрати після впровадження VDS/VPS Після впровадження технології VDS/VPS компанія переходить на віртуальні сервери, орендуючи їх у хмарному провайдері. Витрати змінюються наступним чином: **Оренда віртуальних серверів (хмарний провайдер)**: $35,000 **Економія на електроенергії**: $0 (немає витрат на електроенергію, адже вони включені в орендну плату) **Зменшення площі офісу**: $0 (немає витрат на оренду приміщень під серверну кімнату) **Технічне обслуговування**: $5,000 (зменшення витрат на персонал, оскільки провайдер обслуговує сервери) Загальні витрати після переходу на VDS/VPS становлять **$40,000** на рік.

КБПЗ_2024

Продовження таблиці 7.2

Розрахунок економічної ефективності Економічна ефективність=Економія коштівВитрати до впровадження×100% **Економія коштів**: 95,000−40,000=55,00095,000 - 40,000 = 55,00095,000−40,000=55,000 **Розрахунок ефективності**: Економічна ефективність=55,00095,000×100%≈57.89% Таким чином, економічна ефективність від впровадження VDS/VPS становить **приблизно 57.89%**.
Додаткові переваги Окрім прямої економії витрат, VDS/VPS надає компанії й інші переваги, які непрямо сприяють економічній ефективності: **Масштабованість**: компанія може легко збільшувати або зменшувати обсяг серверних ресурсів залежно від потреб, зменшуючи витрати на придбання додаткового обладнання. **Підвищена надійність**: VDS/VPS-провайдери часто надають резервне копіювання, системи моніторингу та підтримки, що знижує ризики простоїв та витрати на обслуговування. **Фокус на бізнес-задачах**: зменшення навантаження на IT-команду дозволяє спрямувати ресурси на основні бізнес-процеси.

КБПЗ-2024

Розрахунок демонструє, як перехід на VDS/VPS може значно знизити операційні витрати компанії та підвищити її загальну ефективність.

7.5 Пропозиція алгоритму просування проєкту розробки ПЗ

Ось алгоритм просування проєкту програмної реалізації системи віртуального виділеного сервера (VDS/VPS). Він передбачає покрокові дії для досягнення популяризації проєкту та залучення клієнтів (рисунок 7.4).

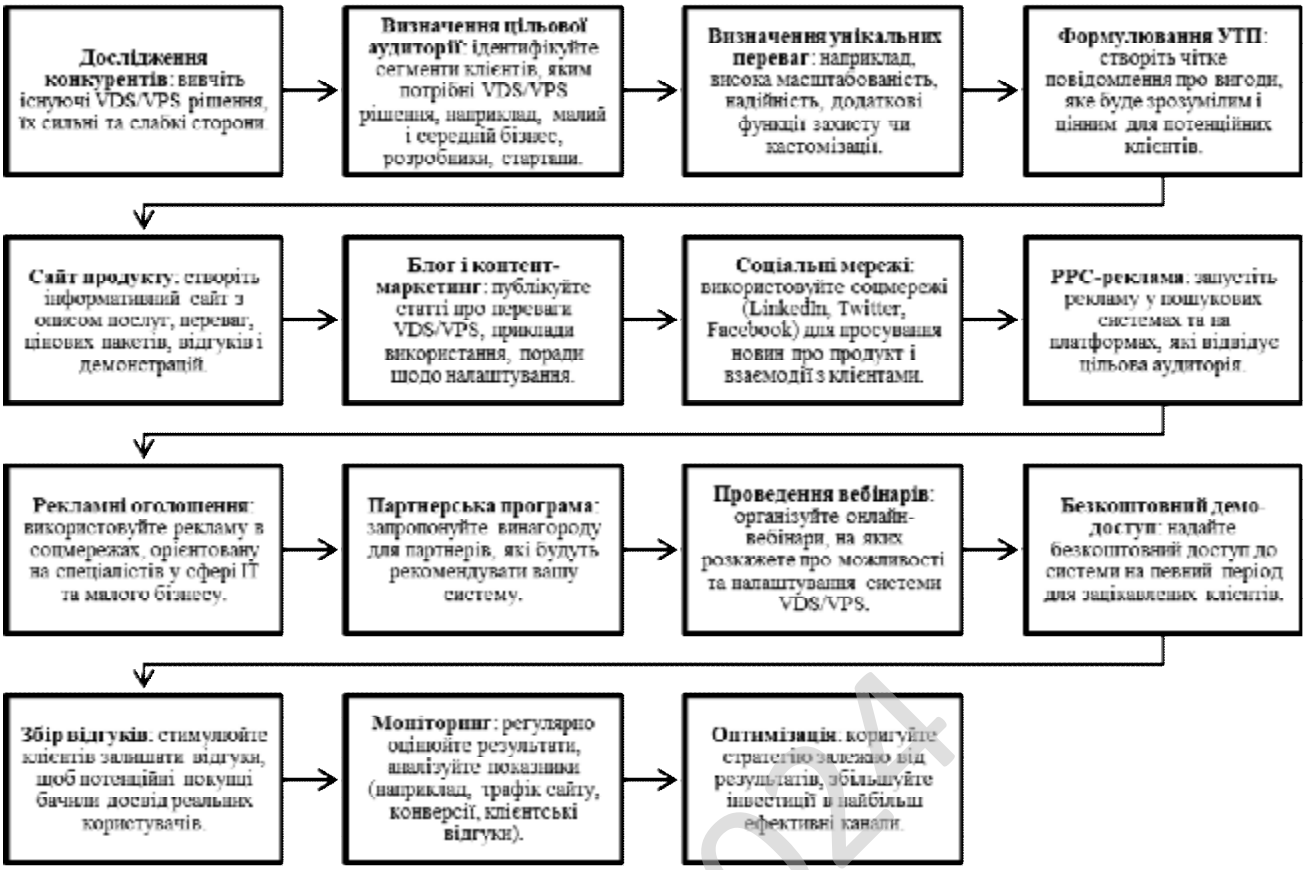


Рисунок 7.4 – Алгоритм просування проєкту

Дотримуючись цього алгоритму, ви зможете не лише просунути проєкт програмної реалізації VDS/VPS, але й налагодити стабільний потік клієнтів і створити стійке конкурентне положення на ринку.

7.6 Оптимізація каналів збуту та шляхів реалізації ПЗ

Для оптимізації каналів збуту та шляхів реалізації проєкту VDS/VPS можна застосувати комплексний підхід, який охоплює як цифрові, так і традиційні канали (рисунок 7.5).

Ці кроки дозволять досягти більшої ефективності, підвищити впізнаваність продукту і розширити охоплення на ринку.

КБПЗ_2024

Онлайн-продажі (Digital Channels): Оптимізуйте сайт і створіть можливість автоматизованого замовлення VDS/VPS послуг, включаючи тарифні плани та додаткові опції. Це полегшить доступ до вашого продукту для широкої аудиторії.

Партнерські програми: Розробіть партнерську систему, де партнери отримують комісійні за рекомендації вашого VDS/VPS. Це може бути вигідним для IT-компаній та фрілансерів, які працюють у сфері хостингу та підтримки серверів.

Сегмент для малого та середнього бізнесу: Створіть спеціальні пакети, орієнтовані на потреби малого бізнесу, з інтеграцією VDS/VPS у популярні інструменти для малого бізнесу (1С, CRM-системи тощо).

Великий бізнес та корпорації: Пропонуйте індивідуальні пакети з розширеними можливостями підтримки, можливістю кастомізації та консультаціями для великих корпоративних клієнтів.

Маркетингова автоматизація (Marketing Automation): Використовуйте інструменти автоматизації для спрощення комунікації з потенційними клієнтами. Впровадьте email-розсилки, спливаючі повідомлення на сайті та ремаркетинг для повернення користувачів.

Інтеграція з хмарними платформами: Співпрацюйте з хмарними провайдерами, які можуть включати ваші VDS/VPS сервіси як додаткові опції для своїх клієнтів.

Пропозиція додаткових послуг: Додайте супутні послуги, такі як кіберзахист, резервне копіювання, антивірусне програмне забезпечення та технічна підтримка.

PPC-кампанії: Запустіть рекламні кампанії в Google Ads і соцмережах, орієнтуючись на спеціалістів, що потребують VDS/VPS-рішень.

SEO для сайту: Оптимізуйте сайт для пошукових запитів, пов'язаних з VDS/VPS, щоб залучити органічний трафік від потенційних клієнтів.

Рисунок 7.5 – Оптимізація каналів збуту та шляхів реалізації

7.7 Визначення ключових факторів успіху конкретного проєкту

Ключові фактори успіху проєкту програмної реалізації системи віртуального виділеного сервера (VDS/VPS) включають низку аспектів, які забезпечують як технічну, так і бізнес-ефективність проєкту (рисунок 7.6).

Ці фактори допоможуть не лише реалізувати технічно надійний продукт, але й зробити його популярним та конкурентоспроможним на ринку.

КБПЗ_2024

Рисунок 7.6 – Ключові фактори успіху проєкту

КБПЗ_2024

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Вступ

Охорона праці – система збереження життя і здоров'я працівників у процесі трудової діяльності, що включає правові, соціально-економічні, організаційні, технічні, санітарно-гігієнічні, лікувально-профілактичні, реабілітаційні та інші заходи. Згідно закону України “Про охорону праці” [1] кожна компанія впроваджує заходи з охорони праці. Реалізується трудові відносини з вживанням необхідних засобів з охорони праці та розробки відповідних документів:

– Інструкцій з охорони праці по кожній професії і загальні.

– Положення про охорону праці.

– Накази з охорони праці.

– Журнали реєстрації та інструктажу.

Законом України “Про охорону праці” [2] регламентуються загальні положення державної політики в галузі охорони праці, а конкретизуються ці положення нормативно-правовими актами про охорону праці, зокрема Наказом Міністерства соціальної політики України 14.02.2018 № 207, який зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [3], яким затверджено нормативно-правовий акт з охорони праці НПАОП 0.00-7.15-18, «Правила охорони праці під час експлуатації електронно-обчислювальних машин», та «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98 [4].

Програмісти у процесі роботи мають негативний вплив на органи зору, а також мають значну розумову напругою і нервово-емоційне навантаження. Руки (суглоби пальців та м'язи рук) при роботі з клавіатурою мають теж істотне

КБПЗ_2024

навантаженням. До шкідливих факторів, які впливають на робітників галузі інформаційних технологій (ІТ) спеціалісти відносять високочастотні електромагнітні коливання (випромінювання) роботи апаратної частини ЕОМ та виділення шкідливих газів.

Ці шкідливі фактори можуть привести до професійних захворювань.

Науково-технічний прогрес суттєво вплинув на умови виробничої діяльності робітників розумової діяльності. Їх праця стала більш інтенсивною, напруженою і вимагає значних витрат розумової, емоційної і фізичної енергії. Це призвело до необхідності у знаходженні комплексного рішення проблем ергономіки, гігієни і організації праці, регламентації режимів праці та відпочинку. Охорона здоров'я робітників, забезпечення безпеки умов праці, ліквідація та профілактика професійних захворювань і виробничого травматизму складає одну з головних турбот людського суспільства.

КБПЗ – 2024

8.2 Аналіз умов праці

У приміщенні розташовано 3 робочих місць з комп'ютерами (далі ПК). Відповідно до норм «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98 [2] площа, що відводиться для робочого місця з комп'ютером повинна бути не менше 6 м^2, об'єм не менше 20 м^3. Розміри даного приміщень складають: довжина – 6 м, ширина – 4,5 м, висота – 3,5 м, тобто загальна фактична площа складає 27 м^2. Необхідна площа на 3 робочих місця із установленими ПК складає 18 м^2, що не перевищує фактичну. Обсяг кабінету на одного працюючого складає 31,5м^3, отже відповідає нормі ДСанПіН 3.3.2-007-98 – не менше 20 м^3.

При роботі з ПК людина може піддатися впливу шкідливих та небезпечних факторів. Під шкідливими виробничими факторами розуміють фактори, тривалий вплив яких викликає розвиток професійних захворювань. Небезпечні виробничі фактори – вплив яких на працюючого викликає травму, тобто пошкодження

організму. Шкідливі і небезпечні чинники, з якими стикається бібліограф при роботі з ПК, приведені в таблиці 8.1.

Таблиця 8.1 – Перелік шкідливих та небезпечних виробничих факторів

Найменування факторів	Можливі джерела їх виникнення	Характер дії
Небезпека ураження електричним струмом	Мережа живлення	Небезпечний
Пожежонебезпечність приміщень	Наявність матеріалів, що згорають і джерел запалення (електроапаратура)	Небезпечний та шкідливий
Іонізація повітря	Статична електрика випромінювання	Шкідливий
Підвищений рівень шуму	Шум створюється перетворювачем напруги ЕОМ, її технічною периферією, а також людьми, що працюють в приміщенні	Шкідливий
Несприятлива освітленість	Недостатнє штучне і природне освітлення	Шкідливий
Незадовільні параметри мікроклімату	Незадовільний стан системи опалення і вентиляції	Шкідливий
Психофізіологічні напруження	Монотонність праці, перенапруженість зорових аналізаторів, розумова напруженість, незручність і статичність пози	Шкідливий

КбПЗ-2024

По категорії вибухо – і пожежонебезпеки, згідно дане приміщення відноситься до категорії В – пожежонебезпечне, тому що присутні тверді матеріали, що горять, такі як дерев'яні столи, папір і інше. Виходячи з категорії пожежонебезпеки і поверховості будинку, ступінь вогнестійкості будівлі II. Згідно з ДБН В 1.1-7-2016 «Пожежна безпека об’єктів будівництва» [3] ЕОМ повинні розташовуватись в будівлі не менше ніж II ступню вогнестійкості.

По ступені небезпеки поразки людей електричним струмом відділ, згідно, класифікується як приміщення з підвищеною небезпекою, тому що не виключена можливість одночасного дотику людини до маючих з'єднання з землею конструкціям будинку, з одного боку, і до металевих корпусів електроустаткування, що можуть виявити під напругою – з іншого.

Для забезпечення вищевказаних оптимальних метеорологічних умов у помешканні передбачена система опалення (загальне парове) в холодному періоді, та вентиляція і кондиціювання в теплий період року, згідно ДБН2.5–67–2013 «Опалення, вентиляція та кондиціювання» [4]. При виконанні замірів параметрів мікроклімату, значення їх відповідали оптимальним та допустимим параметрам відповідно до ДСанПіН3.3.2.007 – 98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно – обчислювальних машин».

Припустимий рівень іонізації повітря помешкання відповідно до СН 21.52-80 повинен складати 1500 – 3000 один./м^3.

Нормування освітлення здійснюється відповідно до ДБН В.2.5 – 28 – 2006 «Природне та штучне освітлення». [5]

Відділ забезпечений комбінованим освітленням. В темний час доби передбачається загальне і/або місцеве рівномірне штучне, а в світлий – бокове одностороннє природне освітлення два віконних прорізи.

Одним з найбільш поширеніших чинників зовнішнього середовища, який несприятливо впливає на людину, є шум. Вплив шуму на організм людини

КБПЗ_2024

залежить від рівня звукового тиску, частотних характеристик, тривалості дії, а також індивідуальних особливостей людини.

При тривалій дії шуму у оператора ЕОМ виявляються симптоми утомленості, нервового збудження, що сприяють погіршенню працездатності і допущенні помилок при роботі. Для уникнення шкідливої дії шуму на організм працюючого, необхідне дотримання нормованих параметрів, які не повинні перевищувати допустимих величин. При роботі на комп'ютері рівень шуму не повинен перевищувати 50 дБА. Приміщення розташоване вікнами у двір і знаходиться далеко від проїжджої частини вулиці. Основними джерелами шуму в приміщенні є устаткування і люди. Розглянута кімната не призначена для прийому відвідувачів і тому в ній не спостерігається великого скупчення людей. Тому основним джерелом шуму є комп'ютерна техніка.

Джерелами шуму при роботі ЕОМ є механічні частини принтера, що рухаються, і вентилятори ($L_{пк}$ = 35 дБА, L_{prn}= 48 дБА) При роботі вентиляційної системи, що забезпечує оптимальний температурний режим електронних блоків ЕОМ і вмонтована в задню панель, створюється аеродинамічний шум. Шум, створюваний працюючим комп'ютером, може бути охарактеризований як широко смуговий постійний з аперіодичним посиленням при роботі принтера. Час роботи ПЕОМ – 6 – 8 год. за добу; принтери працюють не більш 1,5-2 год. за добу.

При наявності великої кількості джерел шуму еквівалентне значення шуму $L_{ЕКВ}$, дБА розраховують по наступній формулі:

$$L_{скв}=10\cdot\lg\left(\frac{1}{T}\sum_{i=1}^{n}\left(t_i\cdot 10^{0.1\cdot L_i}\right)\right) \quad (8.1)$$

де L_i – рівень шуму і-го джерела (пристрою),

t_i – час роботи і-го джерела (пристрою),

T – загальний час роботи,

n – кількість джерел шуму даного типу;

Для даного приміщення необхідні змінні складають:

Загальний час роботи – робочий день, тобто T=8 годин.

КБПЗ_2024

Для фонового шуму (вентиляторів):

L_1 = 35 дБА, T_1 = 8 годин, n_1=15 (5х3);

Для лазерного принтера Lexmark Jet:

L_2 = 48дБА, Т2 = 2 години, n_2=1, для сканера L_3 = 46дБА, Т3 = 2 години.

Підставляємо отримані величини у формулу (8.1):

$$L_{\text{скв}} = 10 \cdot \lg\left(\frac{1}{8} \cdot \left(15 \cdot 8 \cdot 10^{0,1 \cdot 35} + 1 \cdot 2 \cdot 10^{0,1 \cdot 48} + 1 \cdot 2 \cdot 10^{0,1 \cdot 46}\right)\right) = 46{,}3 \text{ дБА}$$

Таким чином, еквівалентний рівень шуму в приміщенні за робітник день Lекв = 46,3 дБА , тобто не перевищує норму 50 дБА.

8.3 Техніка безпеки та протипожежна профілактика

Відповідно ДБН В 1.1-7-2016 «Пожежна безпека об’єктів будівництва» будинок можна віднести до II групи по ступені вогнестійкості й до категорії Д по ступені пожежонебезпеки.

Від розподільного щита по праву й ліву сторони встановлені кондиціонери, зовнішня електропроводка, поміщена в ізольований кабель. Висота проводки становить 2,2 м від рівня підлоги, її кріплення здійснюється за допомогою металевих власників. Біля кожного стола організований розподільний щит, розташований на текстолітовій пластинці, закріпленої на стіні на рівні 1м від підлоги. Усього до складу входять п'ять розеток і дві клеми заземлення. Всі обчислювальні машини з'єднані із клемами заземлення. Чотири з п'яти розеток забезпечують подачу напруги 220 В, а одна, забезпечує подачу напруги в 36 В. Про це є відповідні написи на кожному розподільному щиті.

Робота обслуговуючого персоналу полягає в інсталяції необхідного програмного забезпечення й наступному його використанні в діалоговому режимі роботи з ЕОМ. Іноді може виникати необхідність написання допоміжних програм для поліпшення роботи вузла або для зниження витрат. З погляду забезпечення умов праці й вимог техніки безпеки для роботи програміста необхідно наступне:

КБПЗ-2024

достатнє висвітлення екрана дисплея й робочого місця; повна технічна справність устаткування, його електробезпечність; достатня пожежобезпечність приміщення; оптимальний мікроклімат, що сприяє продуктивній роботі; відповідність робочого місця вимогам ергономіки.

До небезпечних і шкідливих факторів, дії яких піддається програміст, можна віднести: можливість поразки електричним струмом, при електроні справності встаткування, порушенні заземлення або техніки безпеки; робота в мікрокліматі з неприпустимими параметрами; робота при недостатній освітленості екрана дисплея й робочого місця.

Відповідно НПАОП 40.1-1.21-98 “Правил безпечної експлуатації електроустановок споживачів” [6], приміщення можна віднести до приміщень без підвищеної небезпеки, оскільки це приміщення, сухе, з нормальною температурою й ізолюючими підлогами, що не має заземлених металоконструкцій.

Персональні ЕОМ можна віднести до першого класу електротехнічних виробів по способі захисту людини від поразки електричним струмом, оскільки їхні корпуси зроблені з ізолюючої пластмаси й кожен пристрій має заземлення. Відповідно правилам пристрою електроустановок ЕОМ можна віднести до електроустановок з робочою напругою до 1000 В.

Однієї з достовірних причин пожежі в приміщенні з обчислювальною технікою може бути коротке замикання, що спричиняє спалах електропроводки. Для його попередження вся обчислювальна техніка, а також інші електричні пристрої повинні бути обладнані плавкими запобіжниками, а на вході електромережі повинен бути передбачений автомат захисту. Не слід користуватися електричними подовжувачами й трійниками, що не мають сертифікатів відповідності вимогам безпеки.

Необхідно передбачити наявність у межах досяжності первинних засобів гасіння пожежі (вогнегасників) для локалізації вогню власними засобами до приїзду команди пожежної охорони. Повинен бути розроблений план екстреної

КБПЗ_2024

евакуації персоналу при виникненні загоряння. Кількість евакуаційних виходів повинне бути не менш двох. Допускається використання одного евакуаційного виходу, якщо відстань найбільш віддаленого робочого місця до цього виходу не перевищує 25 м.

8.4 Розробка заходів з охорони праці

Перерахуємо проведені заходи щодо забезпечення умов праці на робочому місці програміста.

Для зменшення шуму в приміщенні пропоную використовувати замість матричного принтера, що створює багато шуму, більш тихий – лазерний принтер.

З точки зору забезпечення електробезпеки до цих заходів можна віднести: устаткування розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв; періодична перевірка всіх приладів і пристроїв; щорічна здача іспитів з охорони праці.

З точки зору забезпечення оптимальних умов мікроклімату і освітленості до цих заходів можна віднести: організацію природної вентиляції, за допомогою дефлектора, для забезпечення необхідного повітрообміну в приміщенні вузла; організацію системи центрального опалювання, для підтримки оптимальної температури в холодний період року; організацію штучного загального освітлення, для забезпечення необхідних умов зорової роботи, що відповідають, оформлення паспорта на приміщення вузла, з занесенням в нього вимірювань освітленості, проведених відділом охорони праці.

Для робочої зони виробничих приміщень встановлюються оптимальні та допустимі мікрокліматичні умови з урахуванням важкості виконуваної роботи та періоду року. При одночасному виконанні в робочій зоні робіт різної категорії важкості рівні показників мікроклімату повинні встановлюватись з урахуванням найбільш чисельної групи працівників.

КБПЗ_2024

Як міри по зниженню шуму можна запропонувати:

– облицювання стелі і стін звукопоглинаючим матеріалом (знижують шум на 6-8 до);

– екранування робочого місця (постановкою перегородок, діафрагм);

– установка в комп'ютерних приміщеннях устаткування, що робить мінімальний шум;

– раціональне планування приміщення.

З точки зору забезпечення пожежної безпеки до цих заходів можна віднести наявність схеми евакуації з приміщення вузла, у випадку пожежі, повішену на вхідні двері.

8.5 Висновки до розділу

У даному розділі магістерської роботи були виконано аналіз умов праці користувачів ПК, які працюють у зазначеному приміщенні Проведено перевірка організації робочого місця із відповідними замірами параметрів мікроклімату, освітлення, рівня шуму та розрахунком рівня шуму.

Розроблені заходи щодо поліпшення умов праці дотримання техніки безпеки та проведення протипожежної профілактики дозволить створити умови, які будуть забезпечувати більш комфортну роботу.

КБПЗ_2024

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи віртуального виділеного серверу з використанням технології VDS/VPS.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів віртуального виділеного серверу з використанням технології VDS/VPS.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем віртуального виділеного серверу з використанням технології VDS/VPS.

– Досліджена система віртуального виділеного серверу з використанням технології VDS/VPS.

– На основі отриманих результатів досліджень створена програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання віртуального виділеного серверу з використанням технології VDS/VPS.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

КБПЗ_2024

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм Діффі-Хеллмана.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

Проведено маркетингове та економічне обґрунтування ІТ-проєкту, що дозволило визначити ключові фактори успіху даного проекту.

КБПЗ_2024

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Федюк Я.В. Дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS // Збірник праць молодих науковців ЦНТУ. – Вип. 14. – Кропивницький: ЦНТУ, 2024.

2. Doug Lowe «Networking For Dummies 12th Edition». 2020. – 480 p.

3. Ramon Nastase «Computer Networking: The Beginner's guide for Mastering Computer Networking, the Internet and the OSI Model». 2018. – 186 p.

4. Russ White & Ethan Banks «Computer Networking Problems and Solutions: An Innovative Approach to Building Resilient, Modern Networks». 2017. – 832 p.

5. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

6. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.

7. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

8. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». *Lecture Notes on Data Engineering and Communications Technologies*, 2023, 178, pp. 208–223.

9. Smirnova, T., Gnatyuk, S., Yudin, O., Sydorenko, V., Polozhentsev, A., «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». CEUR Workshop Proceedings Volume 3156, 2022, Pages 390-399.

КБПЗ_2024

10. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». *Communications in Computer and Information Science*, 2021, vol 1486. Springer, Cham. pp 169-184.

11. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

12. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

13. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

14. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

15. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

16. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379.

17. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated

КбПЗ-2024

with the Markov chains», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 633-645.

18. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». *International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019*; Odessa; Ukraine; 9-13 September 2019. P.22-28.

19. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

20. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

21. Smirnov, O., Odarchenko, R., Abakumova, A., Usik, P., Kundyz, M., «QoE optimization technique for media delivery in 5G networks». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019. P.597-601.

22. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

23. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019*/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

24. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in

КБПЗ – 2024

Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

25. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

26. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629.

27. Smirnov A.A., Kuzne tsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

28. Смірнова Т.В., Конопліцька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка.* 2024. №4(24), С. 6-27.

29. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології,* 2024, № 13, с. 28-35.

30. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». *Сучасні інформаційні системи,* 2023, том 7, № 2, С. 49-56.

31. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління,* № 2(70). 2022. С. 28-37.

КБПЗ_2024

32. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98.

33. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

34. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

35. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95

36. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного технологічного університету. Технічні науки.* №4. С. 103-110. 2020.

37. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», *Кібербезпека: освіта, наука, техніка*. № 3(7). С. 43-62. 2020.

38. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в

КБПЗ – 2024

комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

39. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

40. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». *Центральноукраїнський науковий вісник. Технічні науки.* № 2(33). с. 161-172, 2019.

41. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

42. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

43. Смірнов О.А., Дрєєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139

44. Дрєєва Г.М., Смірнов О.А., Дрєєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 173-183, 2019.

45. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дрєєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу

КБПЗ-2024

відновлення та зміцнення поверхонь деталей. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

46. Смірнов О.А., Смірнов С.А., Поліщук Л.І., Смірнова Т.В., Конопліцька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кібербезпека: освіта, наука, техніка. – Том 3 № 3. – Київ: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

47. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

48. Смірнов О.А., Котелянець В.В. Стійкі до колізій стохастичні моделі функціонування безпроводових сенсорних мереж. Вісник інженерної академії України, №3, с. 145-152, 2018

49. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Алгоритми формування безлічі маршрутів передачі метаданих у антивірусні хмарні системи. Збірник наукових праць "Системи обробки інформації". – Випуск 5 (142). – Х.: ХУПС – 2016. – С. 148-152.

50. Смірнов О.А., Смірнов С.А. Дідик А.К., Дрєєв О.М. Моделі системи нейромережевих експертів безпечної маршрутизації у хмарних антивірусних системах. Збірник наукових праць "Системи обробки інформації". – Випуск 3 (140). – Х.: ХУПС – 2016. – С. 36-39.

51. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Спосіб контролю ліній зв'язку телекомунікаційної системи антивірусу. Збірник наукових праць Харківського університету Повітряних Сил. Випуск 2 (47). – Харків: ХУПС. – 2016. – С. 121-127.

52. Смірнов О.А., Смірнов С.А., Дідик А.К. Метод безпечної маршрутизації метаданих у хмарні антивірусні системи. Системи озброєння та військова техніка. – Випуск 2 (46) – Х.: ХУПС – 2016. – С. 146-149.

КБПЗ_2024

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування....2
2 Підстава для розробки....2
3 Мета та призначення розробки....2
4 Джерела розробки....2
5 Технічні вимоги....2
5.1 Вміст проекту....2
5.2 Показники призначення....3
5.3 Вимоги до функціональних характеристик....3
5.4 Вимоги до архітектури....3
5.5 Вимоги до надійності....3
5.6 Умови експлуатації....4
5.7 Вимоги до складу та параметрів технічних засобів....4
5.8 Вимоги до інформаційної і програмної сумісності....4
5.8.1 Обладнання....4
5.8.2 Мова програмування....4
5.8.3 Вхідні дані....5
5.8.4 Вихідні дані....5
6 Вимоги до програмної документації....5
7 Економічні вимоги....5
8 Вимоги щодо охорони праці....5
9 Перелік документів, що розробляються....6
10 Етапи розробки....6
11 Порядок контролю та приймання....6

КБПЗ_2024

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи віртуального виділеного серверу з використанням технології VDS/VPS.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 19-13 від 07.08.2024 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

КБПЗ_2024

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– програмну реалізацію системи віртуального виділеного серверу з використанням технології VDS/VPS;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

КбПЗ_2024

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпечення.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп’ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14” 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Python.

КБПЗ_2024

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати маркетингове та економічне обґрунтування ІТ-проєкту з урахуванням цін на 3 вересня 2024 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти повинні бути розглянуті техніка безпеки та протипожежна профілактика.

КБПЗ_2024

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 86 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти.

10.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Маркетингове та економічне обґрунтування ІТ-проєкту.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на попередній захист 02.12.2024 р.

11.2 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на захист 18.12.2024 р.

КБПЗ_2024

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

_____ Дрєєв О.М.

КБПЗ_2024

Дослідження та програмна реалізація системи віртуального виділеного серверу з використанням технології VDS/VPS

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 19

Літера: РП

Кропивницький – 2024 року

Основна програма

```
import psutil
import subprocess
import docker
import os
import json
import socket
from datetime import datetime

# Ініціалізація Docker-клієнта
client = docker.from_env()

# Функція для створення контейнера з параметрами
def create_vps(container_name, cpu_quota, mem_limit, port):
    container = client.containers.run(
        "ubuntu:latest",
        name=container_name,
        detach=True,
        cpu_quota=cpu_quota * 1000,
        mem_limit=f"{mem_limit}m",
        ports={'80/tcp': port},
        tty=True
    )
    configure_network(container_name, port)
    log_action(f"VPS {container_name} створений з CPU {cpu_quota} та пам'яттю
{mem_limit} MB")
    return container

# Функція для запуску VPS
def start_vps(container_name):
    container = client.containers.get(container_name)
    container.start()
    log_action(f"VPS {container_name} запущено")

# Функція для зупинки VPS
def stop_vps(container_name):
    container = client.containers.get(container_name)
    container.stop()
    log_action(f"VPS {container_name} зупинено")

# Функція для видалення контейнера VPS
def delete_vps(container_name):
    container = client.containers.get(container_name)
    container.remove()
    log_action(f"VPS {container_name} видалено")

# Функція для конфігурації мережі
def configure_network(container_name, port):
    cmd = f"iptables -A INPUT -p tcp --dport {port} -j ACCEPT"
    subprocess.run(cmd, shell=True, check=True)
    log_action(f"Налаштовано мережевий порт {port} для {container_name}")

# Функція моніторингу системних ресурсів
def monitor_resources():
    cpu_usage = psutil.cpu_percent(interval=1)
    mem_info = psutil.virtual_memory()
    net_info = psutil.net_io_counters()
    disk_info = psutil.disk_usage('/')
    resources = {
        "cpu_usage": cpu_usage,
        "memory_usage": mem_info.percent,
        "network_sent": net_info.bytes_sent,
        "network_received": net_info.bytes_recv,
        "disk_usage": disk_info.percent
    }
    log_action("Моніторинг ресурсів системи виконано")
```

КБПЗ_2024

```
    return resources

# Функція для логування дій системи
def log_action(action):
    with open("system_log.txt", "a") as f:
        f.write(f"{datetime.now()} - {action}\n")

# Функція для отримання системної інформації VPS
def get_vps_info(container_name):
    container = client.containers.get(container_name)
    container_stats = container.stats(stream=False)
    info = {
        "cpu_stats": container_stats['cpu_stats'],
        "memory_stats": container_stats['memory_stats'],
        "network_stats": container_stats['networks']
    }
    log_action(f"Отримана інформація для {container_name}")
    return info

# Функція для керування портами в VPS
def setup_port_forwarding(container_name, external_port, internal_port):
    cmd = f"iptables -t nat -A PREROUTING -p tcp --dport {external_port} -j DNAT
--to-destination 127.0.0.1:{internal_port}"
    subprocess.run(cmd, shell=True, check=True)
    log_action(f"Налаштовано перенаправлення портів {external_port} ->
{internal_port} для {container_name}")

# Функція для оновлення ресурсів VPS
def update_vps_resources(container_name, cpu_quota, mem_limit):
    container = client.containers.get(container_name)
    container.update(cpu_quota=cpu_quota * 1000, mem_limit=f"{mem_limit}m")
    log_action(f"Оновлено ресурси VPS {container_name}: CPU {cpu_quota}, пам'ять
{mem_limit} MB")

# Функція для відображення статусу усіх активних VPS
def list_active_vps():
    containers = client.containers.list()
    active_containers = [container.name for container in containers]
    log_action("Список активних VPS отримано")
    return active_containers

# Функція для перевірки стану конкретного VPS
def check_vps_status(container_name):
    container = client.containers.get(container_name)
    status = container.status
    log_action(f"Перевірка стану для {container_name}: {status}")
    return status

# Функція для резервного копіювання даних VPS
def backup_vps_data(container_name, backup_path):
    container = client.containers.get(container_name)
    with open(backup_path, 'wb') as f:
        for chunk in container.export():
            f.write(chunk)
    log_action(f"Резервне копіювання даних {container_name} виконано")

# Функція для відновлення VPS з резервної копії
def restore_vps_data(container_name, backup_path):
    with open(backup_path, 'rb') as f:
        container = client.containers.create(
            "ubuntu:latest",
            name=container_name,
            detach=True,
            tty=True
        )
        container.put_archive('/', f.read())
    log_action(f"Відновлення даних для {container_name} виконано")

# Функція для перевірки доступності порту
```

КСТВ_2024

```
def is_port_available(port):
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        return s.connect_ex(('localhost', port)) != 0

# Функція для ініціалізації VPS з унікальним портом
def init_vps_with_unique_port(container_name, cpu_quota, mem_limit):
    port = 8080
    while not is_port_available(port):
        port += 1
    container = create_vps(container_name, cpu_quota, mem_limit, port)
    log_action(f"Ініціалізовано VPS {container_name} з унікальним портом
{port}")
    return container

# Головна функція
def main():
    vps1 = init_vps_with_unique_port("vps1", cpu_quota=2, mem_limit=1024)
    start_vps("vps1")
    update_vps_resources("vps1", cpu_quota=4, mem_limit=2048)
    resources = monitor_resources()
    print(json.dumps(resources, indent=4))
    stop_vps("vps1")
    delete_vps("vps1")

if __name__ == "__main__":
    main()
```

КБПЗ_2024

Автоматичне масштабування ресурсів на основі навантаження (autoscale.py)

```
# autoscale.py
import psutil
from vps_management import update_vps_resources, monitor_resources, log_action

# Поріг навантаження для збільшення ресурсів
CPU_THRESHOLD = 80
MEMORY_THRESHOLD = 80

# Крок для збільшення ресурсів
CPU_INCREMENT = 2
MEMORY_INCREMENT = 512

def autoscale_vps(container_name):
    resources = monitor_resources()
    cpu_usage = resources['cpu_usage']
    memory_usage = resources['memory_usage']

    if cpu_usage > CPU_THRESHOLD or memory_usage > MEMORY_THRESHOLD:
        new_cpu_quota = min(cpu_usage + CPU_INCREMENT, 8)
        new_memory_limit = min(memory_usage + MEMORY_INCREMENT, 8192)
        update_vps_resources(container_name, new_cpu_quota, new_memory_limit)
        log_action(f"Масштабування VPS {container_name}: новий CPU
{new_cpu_quota}, пам'ять {new_memory_limit} MB")
```

КБПЗ_2024

Інтеграція з системою моніторингу та оповіщення (monitoring_alerts.py)

```
# monitoring_alerts.py
import psutil
import smtplib
from email.mime.text import MIMEText
from vps_management import log_action

EMAIL_SENDER = "alert@example.com"
EMAIL_RECEIVER = "admin@example.com"
EMAIL_SUBJECT = "Системне Оповіщення VPS"
CPU_ALERT_THRESHOLD = 90
MEMORY_ALERT_THRESHOLD = 90

def send_alert(message):
    msg = MIMEText(message)
    msg['Subject'] = EMAIL_SUBJECT
    msg['From'] = EMAIL_SENDER
    msg['To'] = EMAIL_RECEIVER
    with smtplib.SMTP('localhost') as server:
        server.sendmail(EMAIL_SENDER, EMAIL_RECEIVER, msg.as_string())
    log_action("Оповіщення надіслано")

def monitor_and_alert():
    cpu_usage = psutil.cpu_percent(interval=1)
    mem_info = psutil.virtual_memory().percent

    if cpu_usage > CPU_ALERT_THRESHOLD or mem_info > MEMORY_ALERT_THRESHOLD:
        alert_msg = f"CPU використання {cpu_usage}%, Пам'ять {mem_info}%"
        send_alert(alert_msg)
        log_action(f"Оповіщення для CPU {cpu_usage}% і пам'яті {mem_info}%")
```

КбПЗ_2024

Встановлення різних операційних систем у контейнерах (os_installation.py)

```
# os_installation.py
import docker
from vps_management import log_action

client = docker.from_env()

def create_vps_with_os(container_name, cpu_quota, mem_limit, os_image):
    container = client.containers.run(
        os_image,
        name=container_name,
        detach=True,
        cpu_quota=cpu_quota * 1000,
        mem_limit=f"{mem_limit}m",
        tty=True
    )
    log_action(f"VPS {container_name} створений з образом ОС {os_image}")
    return container
```

КБПЗ_2024

Керування через веб-інтерфейс (web_interface.py)

```
# web_interface.py
from flask import Flask, jsonify, request
from vps_management import start_vps, stop_vps, list_active_vps, log_action

app = Flask(__name__)

@app.route('/vps/start', methods=['POST'])
def start_vps_api():
    data = request.json
    container_name = data.get("container_name")
    start_vps(container_name)
    return jsonify({"status": "started", "container": container_name})

@app.route('/vps/stop', methods=['POST'])
def stop_vps_api():
    data = request.json
    container_name = data.get("container_name")
    stop_vps(container_name)
    return jsonify({"status": "stopped", "container": container_name})

@app.route('/vps/list', methods=['GET'])
def list_vps_api():
    containers = list_active_vps()
    return jsonify({"active_vps": containers})

if __name__ == '__main__':
    app.run(debug=True)
```

КБПЗ_2024

Резервне копіювання на хмарні сервіси (cloud_backup.py)

```
# cloud_backup.py
import boto3
from vps_management import log_action

s3_client = boto3.client('s3')
BUCKET_NAME = "vps-backups"

def backup_to_cloud(container_name, backup_path):
    file_name = f"{container_name}_backup.tar"
    s3_client.upload_file(backup_path, BUCKET_NAME, file_name)
    log_action(f"Резервна копія {container_name} завантажена в хмару як
{file_name}")
```

Налаштування періодичних резервних копій (scheduled_backup.py)

```
# scheduled_backup.py
import schedule
import time
from vps_management import backup_vps_data, log_action

BACKUP_PATH = "/backups"

def periodic_backup(container_name):
    backup_vps_data(container_name,
f"{BACKUP_PATH}/{container_name}_backup.tar")
    log_action(f"Періодичне резервне копіювання {container_name} завершено")

def schedule_backups(container_name, interval_minutes):
    schedule.every(interval_minutes).minutes.do(periodic_backup,
container_name=container_name)

    while True:
        schedule.run_pending()
        time.sleep(1)
```

КБПЗ_2024

Інтеграція з API інших хмарних сервісів для створення кластерів (cloud_cluster.py)

```
# cloud_cluster.py
import boto3
from vps_management import log_action

ecs_client = boto3.client('ecs')
CLUSTER_NAME = "vps_cluster"

def create_vps_cluster():
    response = ecs_client.create_cluster(clusterName=CLUSTER_NAME)
    log_action(f"Кластер {CLUSTER_NAME} створено:
{response['cluster']['clusterArn']}")

def add_vps_to_cluster(instance_id):
    response = ecs_client.register_container_instance(
        cluster=CLUSTER_NAME,
        instanceIdentityDocument={"instanceId": instance_id}
    )
    log_action(f"VPS {instance_id} додано до кластера")
```

КБПЗ_2024

Додавання політик безпеки для ізоляції мережевих з'єднань (network_security.py)

```
# network_security.py
import subprocess
from vps_management import log_action

def apply_security_policy(container_name):
    cmd = f"iptables -A INPUT -s 192.168.0.0/16 -j DROP"
    subprocess.run(cmd, shell=True, check=True)
    log_action(f"Політика безпеки застосована до {container_name}")
```

Автоматичне оновлення контейнерів та середовища (auto_update.py)

```
# auto_update.py
from vps_management import log_action
import docker

client = docker.from_env()

def update_container_image(container_name):
    container = client.containers.get(container_name)
    container_image = container.image.tags[0]
    client.images.pull(container_image)
    container.restart()
    log_action(f"Оновлено контейнер {container_name} до останньої версії
{container_image}")
```

КБПЗ_2024

Система управління правами доступу (access_control.py)

```
# access_control.py
import json

USER_ROLES = {"admin": ["start", "stop", "delete", "backup"], "user": ["start",
"stop", "status"]}

def check_permission(role, action):
    if action in USER_ROLES.get(role, []):
        return True
    return False

def perform_action(role, action, container_name):
    if check_permission(role, action):
        # Виконати дію (наприклад, викликати функції vps_management)
        return f"{action} дозволено для {role}"
    return f"{action} заборонено для {role}"
```

КБПЗ_2024

Підтримка VPN-з'єднань для кожного VPS (vpn_support.py)

```
# vpn_support.py
import subprocess
from vps_management import log_action

VPN_SERVER_ADDRESS = "vpn.example.com"
VPN_CREDENTIALS = "/etc/openvpn/credentials.conf"

def setup_vpn(container_name):
    cmd = f"docker exec {container_name} openvpn --config
/etc/openvpn/client.conf --auth-user-pass {VPN_CREDENTIALS}"
    subprocess.run(cmd, shell=True)
    log_action(f"VPN-з'єднання налаштовано для {container_name}")
```

Автоматичне очищення невикористаних контейнерів (cleanup.py)

```
# cleanup.py
import docker
from vps_management import log_action

client = docker.from_env()

def cleanup_inactive_containers():
    containers = client.containers.list(all=True)
    for container in containers:
        if container.status != 'running':
            container.remove()
            log_action(f"Контейнер {container.name} видалено через
бездіяльність")
```

КбПЗ_2024

Тимчасове призупинення та відновлення VPS (pause_resume.py)

```
# pause_resume.py
import docker
from vps_management import log_action

client = docker.from_env()

def pause_vps(container_name):
    container = client.containers.get(container_name)
    container.pause()
    log_action(f"VPS {container_name} призупинено")

def resume_vps(container_name):
    container = client.containers.get(container_name)
    container.unpause()
    log_action(f"VPS {container_name} відновлено")
```

Налаштування автоматичних оновлень ОС у контейнерах (auto_os_update.py)

```
# auto_os_update.py
import docker
from vps_management import log_action

client = docker.from_env()

def enable_auto_os_updates(container_name):
    container = client.containers.get(container_name)
    cmd = "apt-get update && apt-get upgrade -y"
    container.exec_run(cmd, detach=True)
    log_action(f"Автоматичне оновлення ОС налаштовано для {container_name}")
```

Введення тарифікації для VPS з відстеженням використаних ресурсів (billing.py)

```
# billing.py
import time
from vps_management import monitor_resources, log_action

COST_PER_CPU_HOUR = 0.05
COST_PER_GB_RAM_HOUR = 0.01
COST_PER_GB_DISK_HOUR = 0.02

def calculate_billing(container_name, duration_hours):
    resources = monitor_resources()
    cpu_cost = resources['cpu_usage'] * COST_PER_CPU_HOUR * duration_hours
    memory_cost = resources['memory_usage'] * COST_PER_GB_RAM_HOUR *
duration_hours
    disk_cost = resources['disk_usage'] * COST_PER_GB_DISK_HOUR * duration_hours
    total_cost = cpu_cost + memory_cost + disk_cost
    log_action(f"Тарифікація для {container_name}: загальна сума {total_cost}")
    return total_cost
```

КБПЗ_2024

Балансування навантаження між VPS (load_balancer.py)

```
# load_balancer.py
import subprocess
from vps_management import log_action

def setup_load_balancer(vps_list, port=80):
    for idx, container_name in enumerate(vps_list):
        target_port = port + idx
        cmd = f"iptables -t nat -A PREROUTING -p tcp --dport {port} -j DNAT --
to-destination 127.0.0.1:{target_port}"
        subprocess.run(cmd, shell=True)
        log_action(f"Балансування навантаження налаштовано для {container_name}
на порт {target_port}")
```

Інтеграція з інструментами CI/CD для розгортання додатків (cicd_integration.py)

```
# cicd_integration.py
import docker
from vps_management import log_action

client = docker.from_env()

def deploy_app_to_vps(container_name, app_image):
    container = client.containers.get(container_name)
    container.stop()
    container.remove()
    client.containers.run(
        app_image,
        name=container_name,
        detach=True,
        tty=True
    )
    log_action(f"Додаток {app_image} розгорнуто на {container_name}")
```

КБПЗ_2024

Підтримка багатозональної мережевої конфігурації (multi_zone_network.py)

```
# multi_zone_network.py
import subprocess
from vps_management import log_action

def configure_multi_zone_network(container_name, zone="us-west-1"):
    cmd = f"iptables -A INPUT -s {zone} -j ACCEPT"
    subprocess.run(cmd, shell=True)
    log_action(f"Мережева конфігурація багатозональна для {container_name} у
зоні {zone}")
```

Автоматизоване тестування середовища VPS (environment_testing.py)

```
# environment_testing.py
import docker
from vps_management import log_action

client = docker.from_env()

def run_environment_tests(container_name):
    container = client.containers.get(container_name)
    test_cmds = [
        "ls /",
        "uname -a",
        "ping -c 3 8.8.8.8",
    ]
    for cmd in test_cmds:
        result = container.exec_run(cmd)
        log_action(f"Тест {cmd} для {container_name}: {result.output}")
```

КБПЗ_2024

Можливість створення snapshot для швидкого відновлення VPS (snapshot.py)

```
# snapshot.py
import docker
from vps_management import log_action

client = docker.from_env()

def create_snapshot(container_name, snapshot_name):
    container = client.containers.get(container_name)
    snapshot = container.commit(repository=snapshot_name)
    log_action(f"Снапшот {snapshot_name} створено для {container_name}")
    return snapshot.id

def restore_from_snapshot(snapshot_name, new_container_name):
    snapshot_image = client.images.get(snapshot_name)
    new_container = client.containers.run(
        snapshot_image,
        name=new_container_name,
        detach=True,
        tty=True
    )
    log_action(f"Контейнер {new_container_name} відновлено зі снапшоту
{snapshot_name}")
```

КБПЗ_2024

Інтеграція з антивірусним ПЗ для захисту VPS (antivirus_integration.py)

```
# antivirus_integration.py
import subprocess
from vps_management import log_action

def scan_vps_for_viruses(container_name):
    cmd = f"docker exec {container_name} clamscan -r /"
    result = subprocess.run(cmd, shell=True, capture_output=True, text=True)
    log_action(f"Результат антивірусної перевірки для {container_name}:
{result.stdout}")
    return result.stdout
```

Автоматична перевірка портів для унікальності (port_check.py)

```
# port_check.py
import socket
from vps_management import log_action

def is_port_unique(port):
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        result = s.connect_ex(('localhost', port))
        if result == 0:
            log_action(f"Порт {port} вже використовується")
            return False
        else:
            log_action(f"Порт {port} доступний для використання")
            return True
```

Додавання підтримки WebSocket для управління у реальному часі (websocket_control.py)

```
# websocket_control.py
import asyncio
import websockets
import json
from vps_management import start_vps, stop_vps, log_action

async def handle_vps_control(websocket, path):
    async for message in websocket:
        data = json.loads(message)
        action = data.get("action")
        container_name = data.get("container_name")

        if action == "start":
            start_vps(container_name)
            await websocket.send(json.dumps({"status": "started", "container":
container_name}))
        elif action == "stop":
            stop_vps(container_name)
            await websocket.send(json.dumps({"status": "stopped", "container":
container_name}))
        log_action(f"Виконано {action} для {container_name}")

start_server = websockets.serve(handle_vps_control, "localhost", 8765)

asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

Інтеграція з системами керування логами (log_management.py)

```
# log_management.py
import logging
from logging.handlers import RotatingFileHandler
from vps_management import log_action
```

КБПЗ_2024

```
log_file_path = "vps_system.log"
handler = RotatingFileHandler(log_file_path, maxBytes=2000, backupCount=5)
logger = logging.getLogger("vps_system")
logger.setLevel(logging.INFO)
logger.addHandler(handler)

def log_event(event_message):
    logger.info(event_message)
    log_action(f"Лог події: {event_message}")

def retrieve_logs():
    with open(log_file_path, "r") as log_file:
        logs = log_file.read()
    return logs
```

К6ПЗ_2024