ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Пояснювальна записка

до магістерської роботи
на ступінь вищої освіти магістр

на тему: «**РОЗРОБКА АВТОМАТИЗОВАНОЇ ПЕРСОНАЛЬНОЇ ХМАРНОЇ СИСТЕМИ З ВИКОРИСТАННЯМ МІКРОКОМП'ЮТЕРА RASPBERRY PI ДЛЯ IOT** »

Виконав: студент 7 курсу, групи ППЗМ-71
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

Чорний М.В.
(прізвище та ініціали)

Керівник Трінтіна Н.В.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Київ –2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти -«Магістр»
Спеціальність підготовки – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Негоденко О.В.
“____” _____2021 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

ЧОРНОМУ МАКСИМУ ВОЛОДИМИРОВИЧУ

(прізвище, ім’я, по батькові)

1. Тема роботи: «Розробка автоматизованої персональної хмарної системи з використанням мікрокомп'ютера Raspberry PI для IoT»

Керівник роботи: Трінтіна Н.А., к.т.н., доцент кафедри ІПЗ
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від_____
2. Строк подання студентом роботи_____
3. Вхідні дані до роботи
3.1. Raspberry Pi
3.2. Персональна хмарна система
4. Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).
4.1. Вступ.
4.2. Аналітичний огляд і постановка завдання.
4.3. Висновок.

5. Перелік демонстраційного матеріалу (назва основних слайдів)
5.1. Інтерфейс хмарного сховища.
5.2. Термінал операційної системи Raspbian.
5.3. Схеми роботи технологій.

6. Дата видачі завдання –«19» квітня.2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Аналіз літератури та джерел за темою дипломного проекту.	19.04.2021	Виконано
2	Розроблення та затвердження плану дипломного проекту.	21.04.2021	Виконано
3	Проведення консультації з науковим керівником щодо створення першого розділу.	22.04.2021	Виконано
4	Розробка розділу 1	23.04.2021	Виконано
5	Розробка розділу 2	29.04.2021	Виконано
6	Розробка розділу 3	05.05.2021	Виконано
7	Висновки та оформлення пояснювальної записки дипломного проекту.	10.05.2021	Виконано
8	Підписання необхідних документів у встановленому порядку.		
9	Підготовка до захисту та попередній захист дипломного проекту на випусковій кафедрі дипломного проекту		

Студент __ _____ Чорний М.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Трінтіна Н.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту роботи «Розробка автоматизованої персональної хмарної системи з використанням мікрокомп'ютера Raspberry PI для IoT» викладена на 73с., містить 14 рис., табл.2, 8 літературних джерел.

Ключові слова: Cloud, PHP, Raspberry Pi, OpenSSL, Linux, Хмарні технології, Raspberry Pi OS, SSH, API, OwnCloud, NAS, Open Source, SSL, IoT.

Предмет дослідження: Персональна автоматизована хмарна система

***Об'єкт дослідження*:** Хмарне зберігання файлів використовуючи відкрите програмне забезпечення OwnCloud та міні комп’ютер Raspberry Pi.

Методи дослідження: Методи дослідження: анліз ринку відкритого ПЗ хмарних сховищ і вибір оптимального софта для його подальшого розгортання на Raspberry PI.

Мета роботи: Розробка автоматизованого сховища після вибору необхідного софта на ринку ПЗ з відкритим вихідним кодом.

Отримані результати: Розроблено автоматизовану хмарну систему з використанням OwnCloud і мікрокомп'ютера Raspberry PI

Результати дипломної роботи можуть використовуватися для автоматизованого створення хмарного сховища.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних;

ОС – операційна система;

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер;

RPi – система захисту інформації

SSL – сертифікат безпеки;

DNS – Domain Name System;

SSH – Secure Shell;

NTFS – New Technology File System;

IoT – Internet of Things.

WP-CLI – Command-line interface for WordPress

Bash – Bourne again shell

PHP – Hypertext Preprocessor

ВСТУП

Хмарні технології дають нам можливість отримувати дані де і коли завгодно. Багато компаній використовують готові хмарні сховища, інші, в основному з міркувань безпеки, запускають свої, так і люди, одні - довіряють публічним хмарним сховищ, інші - бояться що їх файли можуть потрапити в чужі руки, а хто то просто вважає що занадто дорого платити за великий обсяг файлів і дешевше просто розгорнути персональне сховище.
У цій дипломній роботі ми розглянемо як організувати персональне хмарне сховище використовуючи передові технології автоматизації деплоя такі як Docker і Ansible, при цьому використовуючи багатофункціональний мікрокомп'ютер Raspberry PI.

1 ХМАРНЕ СХОВИЩЕ: ПОНЯТТЯ ТА ПРИЗНАЧЕННЯ

1.1 Хмарне сховище: Загальне визначення

Хмарне сховище (англ. cloud storage, або ще хмара, backup) — це модель збереження даних у комп'ютері, в якій цифрові дані накопичуються в логічні пули, а фізичне зберігання охоплює кілька серверів (зазвичай у кількох місцях). Фізичне середовище, як правило, належить хостинговим компаніям, які ним керують. Ці постачальники хмарних систем зберігання даних відповідають за утримання наявної інформації та доступ до неї, а також за роботу фізичного середовища. Користувачі можуть купувати у постачальників послуг хмарного сховища можливість накопичувати там дані.

До хмарного сховища отримати доступ можна через спеціальні сервіси доступу:

1. WEB сервіси
2. API
3. Спеціальне ПЗ

Підіб’ємо підсумки, хмарне сховище:

1. Об’єднані між собою вузли.
2. Завдяки надлишковому розподіленню даних досягається висока відмовотійкість.
3. Завдяки версіонуванню досягається висока надійність.

Питання персональних хмарних систем найчастіше виникає через недовіру до постачальників послуг, бо ніхто не хоче щоб його конфіденціальна інформація була використана кимось іншим. Далі ми розберемо три найпопулярніші сервіси які себе зарекомендували як надійні та надають послуги хмарного зберігання файлів.

Таблиця 1.1 Найвідоміші сервіси для хмарного зберігання файлів

№п/п	Компанія	Назва продукту	Ємність безкоштовного плану
1	Microsoft	OneDrive	5GB
2	Google	Google Drive	15GB
3	Dropbox	Dropbox	2GB

Як ми бачимо із таблиці вище здебільшого компанії безкоштовно багато простору не дають, хоча при необхідності у них є тарифи які за платну підписку дають вам змогу розширити своє сховище, але якщо ви забудете заплатити тоді деякі сервіси можуть видалити ваші дані.

1.2 Хмарне сховище: Принцип роботи

Для роботи хмарного сховища потрібен принаймні один сервер данних. Зазвичай користувач відправляє дані через WEB, або через спеціальній додаток і так само потім отримує їх.

Зазвичай у хмарних системах використовується реплікація даних для забезпечення високої доступності даних їх розміщують на великій кількості серверів, іноді географічно розділених.

Тоді як реплікація у загальнодоступних хмарах задля забезпечення високої доступності сервери розділяються географічно то основне розташування зазвичай знаходиться максимально близько до компанії для високої швидкості обробки даних.

Також в останні роки збільшилась тенденція до росту хмарних обчислень і serverless технологій тому велика кількість компаній вже перейшла, або ще переходить на хмарні обчислення, а найбільш популярні послуги це:

- Надання хмарного серверу
- Зберігання даних
- Центри обробки даних

- Мережеві сервери

Схематичне зображення хмарного сховища (рис. 1.1)

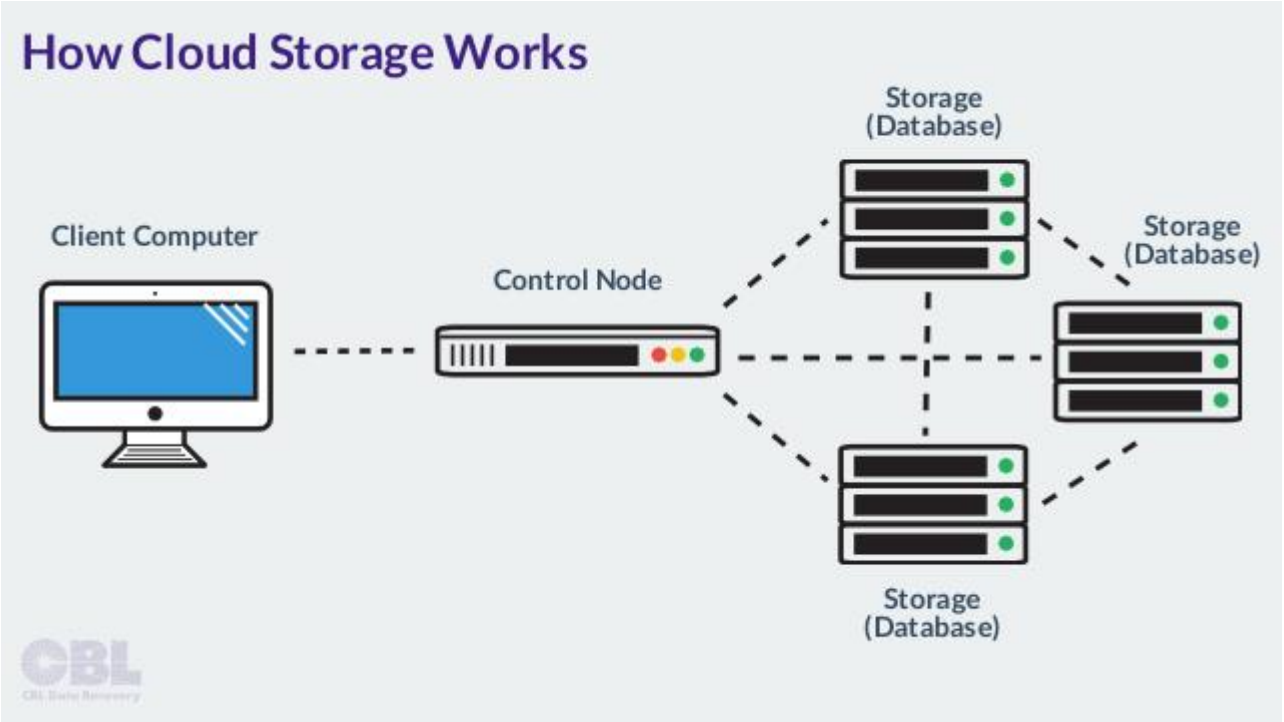

Рис.1.1 – Приблизна схема роботи хмарного сховища

Доступ до хмарних ресурсів може бути тарифікований багатьма способами, але найпопулярніші з них:

- Оплата за транзакцію.

- Фіксована плата за фіксовану кількість ресурсів.

- Динамічна тарифікація за динамічну кількість ресурсів

Ціни можуть відрізнятися в залежності від постачальника послуги, а також необхідної кількості ресурсів.

1.3 Хмарне сховище: Види

Існує чотири основних типи хмарного сховища:

1. Персональне хмарне сховище
2. Приватне хмарне сховище

3. Публічне хмарне сховище
4. Гібридне хмарне сховище.

Персональне хмарне сховище – користувачі зберігають свої дані на персональних пристроях під'єднаних до мережі інтернет. При цьому зазвичай користувач має повний доступ до пристрою

Приватне хмарне сховище – користувачі здебільшого компанії стурбовані безпекою, або їм потрібна більша гнучкість. Зазвичай у таких компаній свої сервери для обробки даних і вони мають повний контроль над системою і даними які у ній зберігаються.

Публічне хмарне сховище – такий тип сховищ здебільшого належить конкретним компаніям (наприклад Google, Amazon, etc.) і вони продають їх як послугу. Цей тип хмарних сховищ контролює, підтримує і обслуговує компанія власник

Гібридне хмарне сховище – поєднання публічного и приватного сховища, наприклад компанія може використовувати свої власні ресурси для роботи програмного забезпечення розробленого іншою компанією

Між приватними та публічними хмарами є істотні відмінності:

- Контроль – приватна інфраструктура хмари належить і контролюється підприємством, яким воно використовується, тоді як за публічну хмару відповідає постачальник послуги.

- Оновлення – приватну хмару оновлювати потрібно своїми силами, а публічну оновлює постачальник

- Ресурси – для приватної хмари потрібна своя інфраструктура і ресурси, а у публічних за ресурси відповідає постачальник.

- Безпека – компаніям з чутливою інформацію краще обійтися приватними хмарами;

- Відмовостійкість – публічні хмари досягають більшої відмовостійкості через надлишковість ресурсів і їх географічного розділення.

ВИСНОВОК ДО РОЗДІЛУ 1

Аналіз проведений у цьому розділі дає нам розуміння того, що таке хмарні сховища, які вони бувають, їх основні переваги та недоліки.

Хмарні сховища відкривають безліч можливостей таких як зберігання файлів, можливість швидко ними поділитися і бувають різних типів тому кожен може обрати хмарне сховище, яке буденайкраще задовільняти потреби користувача, але скільки б не було видів хмарних сховищ це завжди вибір балансу між надійністю, безпекою та підтримкою.

2 ХМАРНЕ СХОВИЩЕ: ЗАСОБИ ДЛЯ РЕАЛІЗАЦІЇ ПХС

2.1 Інтернет речей (Internet of Things)

За останні кілька років «Інтернет речей» (IoT) став однією з найважливіших технологій 21 століття.

Інтернет речей допомагає людям жити і працювати ефективніше, а також отримувати повний контроль над своїм життям. IoT є важливим для бізнесу, та надає компаніям можливість у реальному часі вивчити, як насправді працюють їх системи, надаючи уявлення про все - від продуктивності машин до ланцюгів поставок та логістичних операцій.

IoT дозволяє компаніям автоматизувати процеси та зменшити витрати на робочу силу. Це також скорочує витрати та покращує послуги доставки, роблячи виробництво та доставку товарів дешевшими, а також забезпечує прозорість транзакцій клієнтів.

Таким чином, IoT є однією з найважливіших і найперспективніших технологій у повсякденному житті, і вона буде продовжувати набирати силу, і створювати сотні нових продуктів і призводити до появи нових компаній на ринку, які диктують свої умови IT-гігантам.

Взаємодія споживачів із розумними пристроями має реляційний характер, і дослідження підтвердили "метафору взаємовідносин" як плідний спосіб зрозуміти реакцію споживачів на об'єкти споживання. Але розумні пристрої ставлять унікальні проблеми для розгляду виникнення відносин між споживачем та об'єктом, оскільки їх ступінь свободи влади, автономії та повноважень надає їм власні унікальні можливості взаємодії. Ми представляємо нову структуру взаємовідносин споживач – об’єкт, засновану на ексцемплексній моделі міжособистісної взаємодоповнюваності та розташовану в теорії збірки та об’єктно-орієнтованій онтології. Стилі взаємовідносин між споживачем і об'єктом визначаються з точки зору двох основних вимірів поведінки, свободи волі та спілкування на основі виразних ролей, які виконують споживач та об'єкт.

Накладання теорії асамблеї забезпечує концептуально багате розуміння простору стилів відносин між господарем-слугою, партнером та нестабільними відносинами, а також їх супутнього позитивного (сприяючого) проти негативного (обмежуючого) споживчого досвіду. Основна геометрія моделі підтримує велику емпіричну роботу та забезпечує потужну управлінську основу для вимірювання та відстеження взаємозв'язків споживач – об'єкт та подорожей, які вони здійснюють з часом.

Незважаючи на феноменальний прогрес у обчислювальній потужності та функціональності електронних систем, взаємодія людина-машина здебільшого обмежувалася простими панелями управління, клавіатурою, мишею та дисплеєм. Отже, ці системи або критично покладаються на безпосереднє керівництво людьми, або працюють майже незалежно від користувача. Зразковою технологією, яка міцно інтегрується у наше життя, є смартфон. Однак термін "розумний" є неправильним терміном, оскільки він принципово не має інтелекту, щоб зрозуміти свого користувача. Користувачам все ще потрібно набирати текст, торкатися або говорити (до певної міри), щоб висловити свої наміри у формі, доступній для телефону. Отже, розумне прийняття рішень досі майже повністю є людським завданням. Досвід, що змінює життя, можна досягти, перетворивши машини з пасивних інструментів на агенти, здатні зрозуміти фізіологію людини та те, що хоче їх користувач. Це може вдосконалити людські можливості немислимими шляхами шляхом побудови симбіотичних відносин для спільного вирішення проблем реального світу. Однією з найважливіших областей застосування цього підходу є допоміжні технології Інтернету речей (IoT) для людей з обмеженими фізичними можливостями. Щорічний світовий звіт про інвалідність показує, що 15% світового населення живе з інвалідністю, тоді як 110-190 мільйонів із цих людей мають труднощі у функціонуванні. Якість життя для цього населення може значно покращитися, якщо ми надамо доступ до розумних пристроїв, які будуть допомагати у повсякденних завданнях. Ця робота демонструє, що розумні пристрої IoT відкривають можливість полегшити навантаження на користувача, оснащуючи повсякденні предмети, такі як інвалідне крісло, можливостями прийняття рішень.

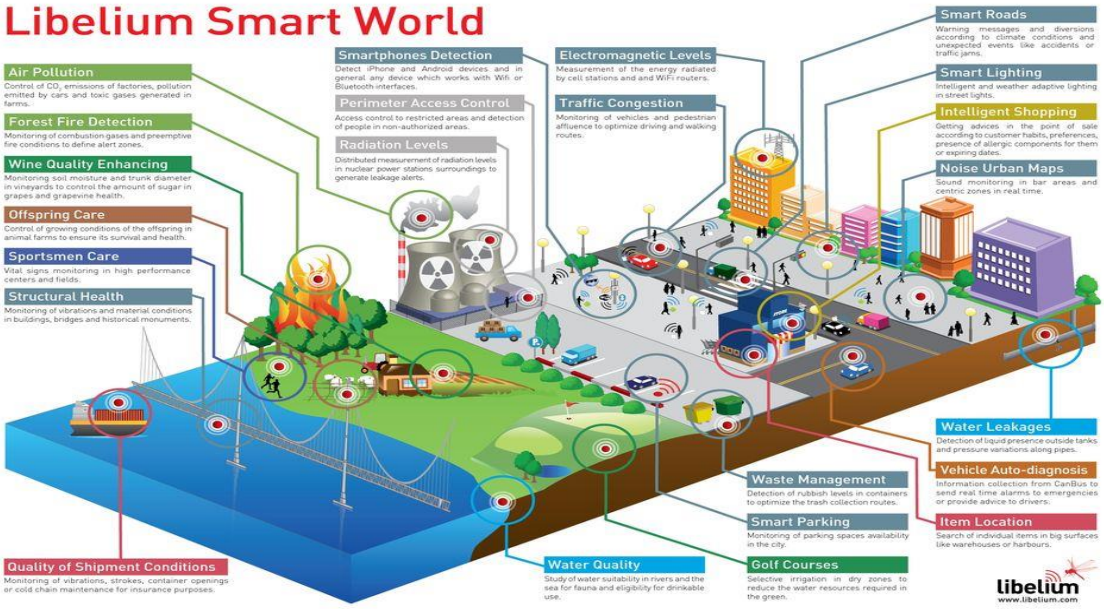

Рис. 2.1 – Приклад розумного міста Libelium

Інтернет речей (IoT) змінює наше життя. IoT є скрізь, хоча ми не завжди бачимо його або знаємо, що пристрій є частиною IoT. IoT перетворює фізичні об'єкти на екосистему інформації, яку обмінюються пристроями, які можна носити, переносити, навіть імплантувати, що робить наше життя технологією з купою даних. IoT-додатків для бізнесу багато. Розумні машини змінюються, коли, де і як виконується робота практично у кожній галузі; але, що це означає для реального життя? IoT - це безпрецедентна мережа, що з'єднує машини, людей, дані та процеси, і тепер фільтрується до реального життя, формуючи те, як ми рухаємось у своєму повсякденному житті. Деякі реальні приклади IoT - це носяться фітнес та трекери (наприклад, Fitbits) та додатки охорони здоров'я IoT, голосові помічники (Siri та Alexa), розумні машини (Tesla) та розумні прилади (iRobot). Завдяки швидкому розгортанню IoT, щоденного контакту з декількома пристроями IoT, неминуче стане невдовзі.

2.2 Raspberry Pi

Raspberry Pi - це недорогий комп’ютер розміром з кредитну картку, який підключається до монітора комп’ютера або телевізора та використовує стандартну

клавіатуру та мишу. Це здатний маленький пристрій, який дозволяє людям різного віку досліджувати обчислення та навчатись програмуванню на таких мовах, як Scratch та Python. Він здатний робити все, що ви очікуєте від настільного комп'ютера, починаючи від перегляду Інтернету та відтворення відео високої чіткості, до створення електронних таблиць, обробки текстів та ігор.

Більше того, Raspberry Pi має можливість взаємодіяти із зовнішнім світом і використовувався в широкому спектрі проектів цифрових виробників, починаючи від музичних машин та батьківських детекторів, закінчуючи метеостанціями та шпаківнями з інфрачервоними камерами. Компанія Raspberry Pi має на меті побачити як цей невеличкий комп'ютер використовується дітьми у всьому світі, щоб навчитися програмувати та зрозуміти, як вони працюють.

Щоразу, коли хтось запитує мене, як перетворити їх дивний технологічний проект на реальність, мій найближчий інстинкт - рекомендувати Raspberry Pi. Цей комп'ютер за 35 доларів, є настільки ж потужним, наскільки дешевим. За допомогою лише трохи ноу-хау та цікавості ви можете використовувати його для створення ретро-ігрової консолі, мозкового робота, сенсора розумного будинку або навіть повністю функціональної динаміки, сумісної з Alexa.

Pi - це одноплатний комп'ютер, що означає, що мікропроцесор, пам'ять, бездротові радіостанції та порти знаходяться на одній друкованій платі.

Сьогодні існують такі моделі:

Таблиця 2.1 Технічні характеристиики різних моделей Raspberry Pi

Product	**SoC**	**Speed MHz**	**RAM MB**	**USB Ports**	**Ether net**	**Wire-less**	**Blue-tooth**
Raspberry Pi Model A+	BCM2835	700	512	1	No	No	No
Raspberry Pi Model B+	BCM2835	700	512	4	100M	No	No

Product	SoC	Speed MHz	RAM MB	USB Ports	Ether net	Wire-less	Blue-tooth
Raspberry Pi 2 Model B	BCM2836/7	900	1024	4	100M	No	No
Raspberry Pi 3 Model B	BCM2837A0/B0	1200	1024	4	100M	802.11n	4.1
Raspberry Pi 3 Model A+	BCM2837B0	1400	512	1	No	802.11ac/n	4.2
Raspberry Pi 3 Model B+	BCM2837B0	1400	1024	4	1000M	802.11ac/n	4.2
Raspberry Pi 4 Model B	BCM2711	1500	2048	2xUSB2, 2xUSB3	1000M	802.11ac/n	5.0
Raspberry Pi 4 Model B	BCM2711	1500	4096	2xUSB2, 2xUSB3	1000M	802.11ac/n	5.0
Raspberry Pi 4 Model B	BCM2711	1500	8192	2xUSB2, 2xUSB3	1000M	802.11ac/n	5.0
Raspberry Pi Zero	BCM2835	1000	512	1	No	No	No
Raspberry Pi Zero W	BCM2835	1000	512	1	No	802.11n	4.1
Raspberry Pi Zero WH	BCM2835	1000	512	1	No	802.11n	4.1

Product	SoC	Speed MHz	RAM MB	USB Ports	Ether net	Wire-less	Blue-tooth
Raspberry Pi 400	BCM2711	1800 MHz	4GB	1xUSB2, 2xUSB3	1000Base-T	802.11 ac/n	5.0

Як виглядають різні моделі Raspberry Pi можна ознайомитись на малюнку 2.2 представленому нижче:

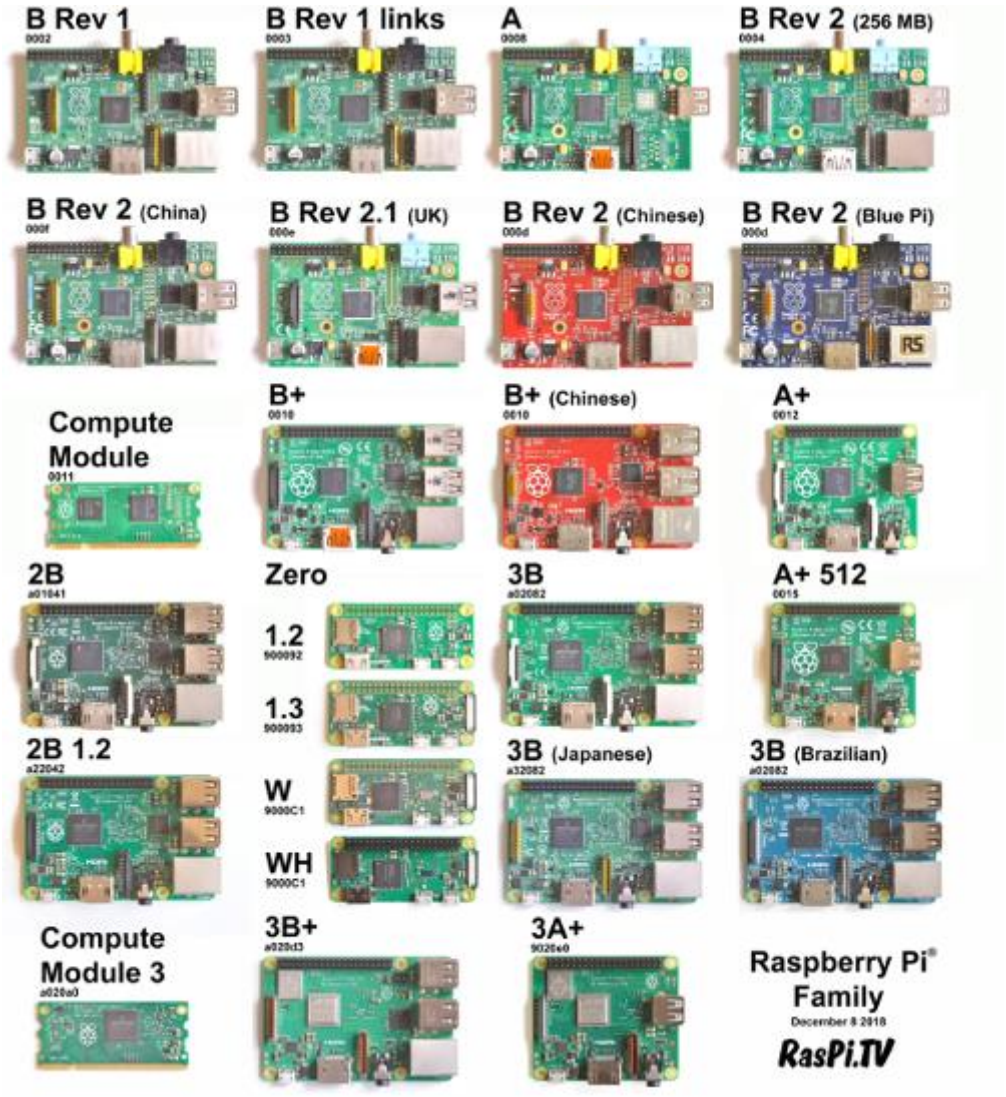

Рис. 2.2 Raspberry PI моделі

Фонд Raspberry Pi - це британська освітня благодійна організація, яка працює над передачею потужності обчислювальної техніки та цифрових технологій в руки людей у всьому світі. Вони хочуть, щоб більше людей мали можливість використовувати цифрові технології для роботи, вирішувати важливі для них проблеми та творчо висловлюватися. Фонд розробив навчальний контент та програми, які допомагають мільйонам людей засвоювати нові знання та навички.

Через Code Club та CoderDojo вони підтримують найбільші у світі мережі безкоштовних обчислювальних клубів для молоді. Десятки тисяч викладачів пройшли їх онлайн-курси, і мільйони людей користуються їхніми безкоштовними онлайн-навчальними ресурсами. Вони є частиною консорціуму, який отримав 82 мільйони фунтів стерлінгів для державного фінансування підготовки вчителів обчислювальної техніки в кожній школі Англії.

Фонд Raspberry Pi забезпечує Raspberry Pi OS (раніше називану Raspbian), дистрибутив Linux на базі Debian (32-розрядний) для завантаження, а також сторонні Ubuntu, Windows 10 IoT Core, RISC OS та LibreELEC (спеціалізований медіа-центр розподіл). Він просуває Python і Scratch як основні мови програмування, підтримуючи багато інших мов. Прошивка за замовчуванням - із закритим кодом, а неофіційна з відкритим кодом - доступна. Багато інших операційних систем також можуть працювати на Raspberry Pi. Сторонні операційні системи, доступні на офіційному веб-сайті, включають Ubuntu MATE, Windows 10 IoT Core, ОС RISC та спеціалізовані дистрибутиви для медіа-центру Kodi та управління класом. Також підтримується офіційно перевірене мікроядро seL4.

2.3 Операційна система Raspberry Pi OS

Raspberry PI OS є рекомендованою OS для встановлення на Raspberry PI. Raspberi PI OS постачається з фірмовим інсталятором для встановлення підготовки Raspberry PI до використання NOOBS (New Out Of Box Software).

NOOBS - це програма встановлення ОС, створена Raspberry Pi. Він поставляється з попередньо завантаженими операційними системами, які можуть працювати на Raspberry Pi.

Як працює NOOBS?

Програмний пакет NOOBS завантажується на ваш комп'ютер і встановлюється на карту MicroSD. Коли ви завантажуєте SD-карту на Raspberry Pi, інтерфейс завантажується з добіркою операційних систем. Ви можете використовувати програму NOOBS, щоб вибрати ОС, завантажити додаткові з Інтернету та встановити їх безпосередньо на свій Pi.

NOOBS економить ваші проблеми з відстеженням окремих пакетів ОС в Інтернеті. Це як універсальний магазин для налаштування ОС на вашому Pi.

NOOBS включає у себе можливість встановлення наступних операційних систем:

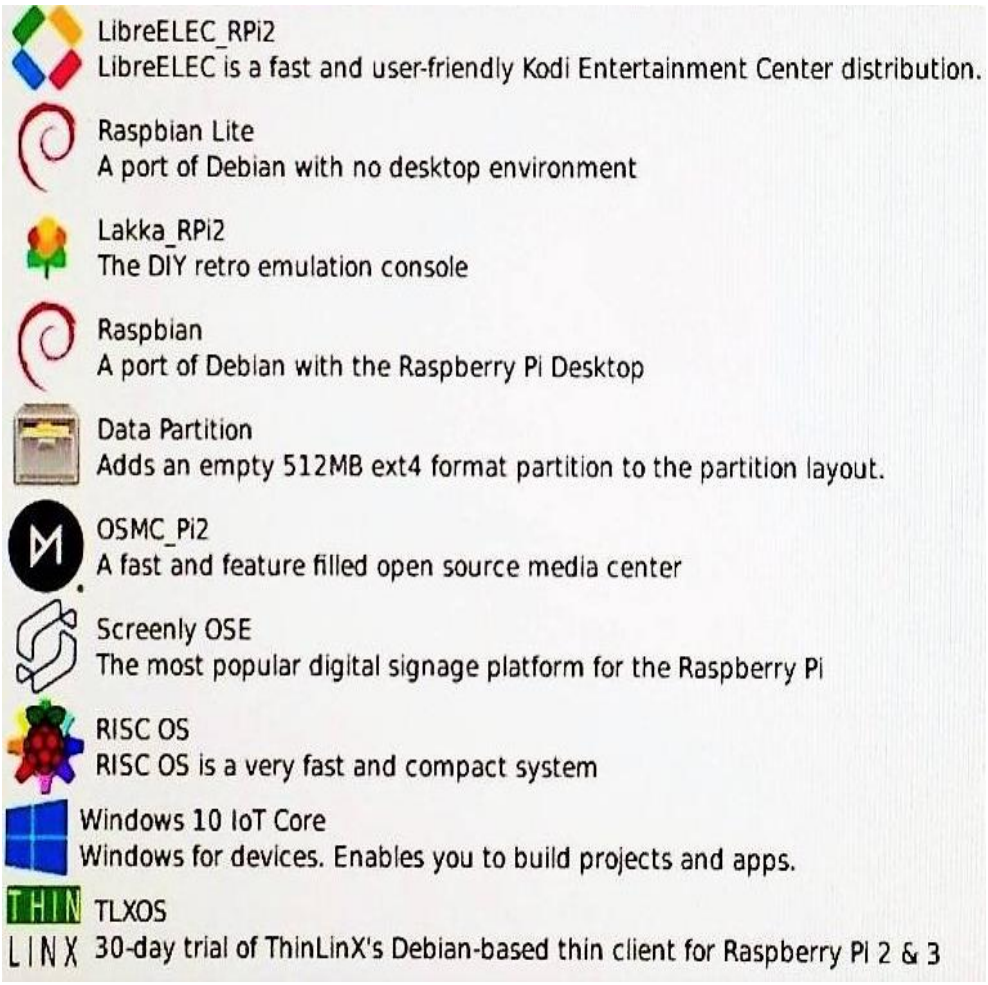

Рис.2.3 Операційні системи Noobs

2.3.1 Підготовка до встановлення операційної системи

Raspberry Pi розробили графічний інструмент для запису на SD-карту, який працює на Mac OS, Ubuntu 18.04 та Windows, і є найпростішим варіантом для більшості користувачів, оскільки він завантажить зображення та автоматично встановить його на SD-карту.

Спочатку вставте карту microSD у ваш комп'ютер.

Тепер вам потрібно встановити відповідний Raspberry Pi Imager для вашої операційної системи

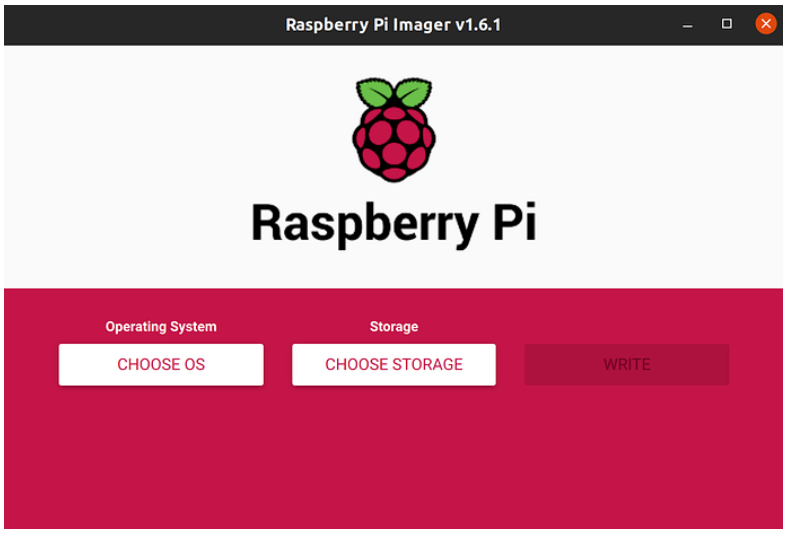

Рис.2.4 – Програма Raspberry PI imager

Натисніть Choose OS

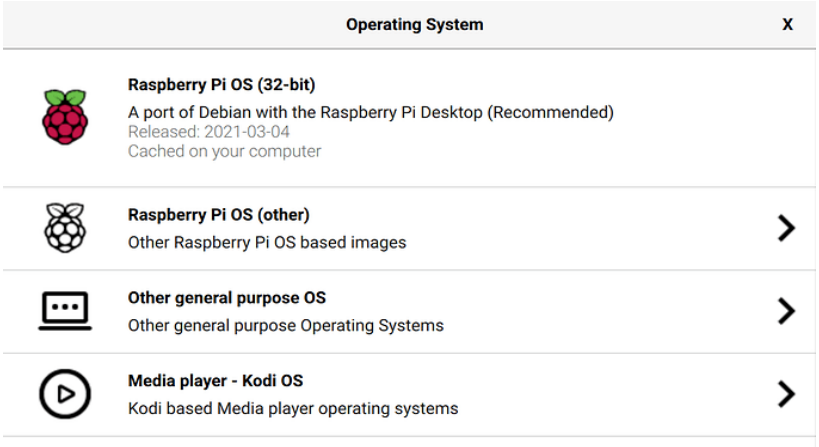

Рис.2.5 – Програма Raspberry PI imager

Оберіть Raspberry PI OS

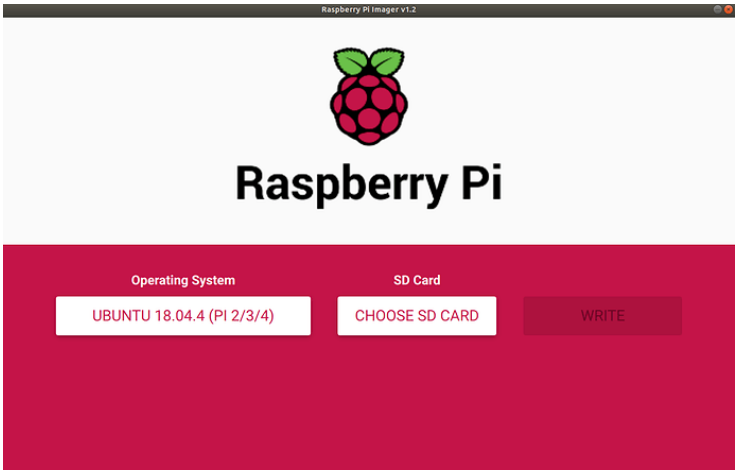

Рис.2.6 – Програма Raspberry PI imager

Оберіть SD карту і натисніть Write

Дочекайтесь закінчення процесу копіювання файлів і встановіть SD карту у Raspberry PI у спеціальний слот який знаходиться на нижньому боці Raspberry PI, потім підключіть Raspberry до живлення і увімкніть. Нижче представлена схема розміщення модулів на платі.

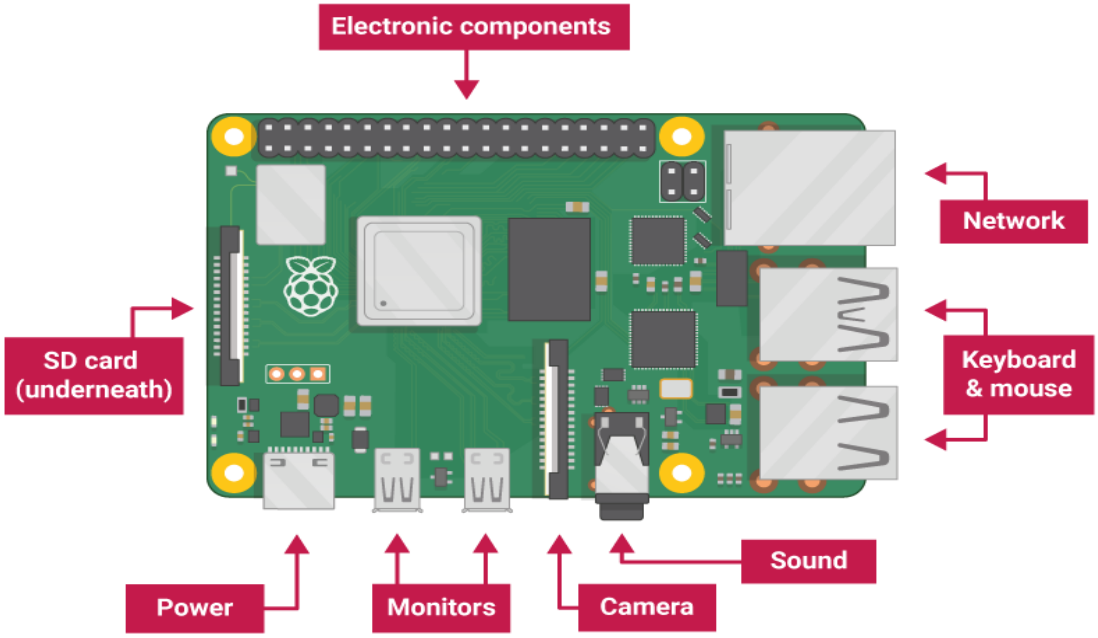

Рис.2.7– Розміщення модулів на платі

2.4 SSH технологія

SSH або Secure Shell - це протокол віддаленого адміністрування, який дозволяє користувачам контролювати та змінювати свої віддалені сервери через Інтернет. Послуга була створена як безпечна заміна незашифрованого Telnet і використовує криптографічні прийоми для забезпечення того, щоб усі зв'язки з віддаленим сервером відбувалися в зашифрованому вигляді. Він забезпечує механізм автентифікації віддаленого користувача, передачі вхідних даних від клієнта на хост і передачі вихідних даних клієнту.

Будь-який користувач Linux або macOS може передавати SSH на свій віддалений сервер безпосередньо з вікна терміналу. Користувачі Windows можуть скористатися перевагами клієнтів SSH, таких як Putty. Ви можете виконувати команди оболонки так само, як і в тому випадку, якщо ви фізично керували віддаленим комп'ютером. Якщо ви використовуєте Linux або Mac, то використовувати SSH дуже просто. Якщо ви використовуєте Windows, вам потрібно буде використовувати клієнт SSH, щоб відкрити з'єднання SSH.

Для користувачів Mac та Linux перейдіть до програми терміналу та виконайте наведену нижче процедуру:

Команда SSH складається з 3 різних частин:

ssh {user}@{host}

Команда ключа SSH вказує вашій системі, що ви хочете відкрити зашифроване з'єднання Secure Shell. {user} представляє обліковий запис, до якого ви хочете отримати доступ. Наприклад, ви можете отримати доступ до кореневого користувача, який в основному є синонімом системного адміністратора з повними правами змінювати що-небудь у системі. {host} - це комп’ютер, до якого ви хочете отримати доступ.

Коли ви натиснете Enter, вам буде запропоновано ввести пароль для запитуваного облікового запису. Коли ви вводите його, на екрані нічого не відображатиметься, але ваш пароль фактично передається. Закінчивши вводити, натисніть клавішу Enter ще раз. Якщо ваш пароль правильний, вас зустріне вікно

віддаленого терміналу.

Переваги SSH

1) Шалено потужний. Просто немає можливості порівняти PHP-скрипт із тим, що ви можете досягти за допомогою командного рядка.

2) Ефективні резервні копії та міграції. Оскільки всі резервні копії та міграція можуть відбуватися безпосередньо на веб-сервері, все відбувається блискавично. Це повністю руйнує будь-які резервні копії чи міграції, запущені з плагіна WordPress або сторонньої служби.

3) Прямий доступ до сайту. Це означає, що більшість інструментів управління продовжуватимуть працювати, навіть якщо веб-сайт, який працює в режимі офлайн, закінчується через завершення терміну дії домену або перехід клієнта на інший хост / платформу. Доменне ім'я може навіть змінитися на щось нове, не потребуючи повторного підключення сайту до інструментів управління.

4) Прямий доступ до дивовижних інструментів командного рядка. Сюди входить WP-CLI, який має можливість спілкуватися з WordPress, навіть якщо є проблеми з темою або плагіном, що впливають на серверну інформацію WordPress.

5) Автоматизація. Будь-що можна автоматизувати за допомогою сценаріїв BASH. Хочете встановити плагін на 1000 сайтів? Напишіть сценарій. Хочете застосувати конкретні конфігурації серверної бази? Напишіть сценарій.

6) Менший слід на WordPress. Немає необхідності у супутньому плагіні WordPress або навіть до серверного доступу. Інструменти управління спілкуються безпосередньо з веб-сайтом через SSH.

Недоліки SSH

1) Додаткова попередня робота. Кожен доданий сайт потребує ключа SSH, доданого через SFTP або вручну через SSH.

2) Немає власного графічного інтерфейсу. Використання графічного інтерфейсу додає додатковий рівень, що означає, що дуже прості речі, такі як управління плагінами/темами, можуть зайняти більше часу. Це також означає, що вам потрібно створити графічний інтерфейс, якщо ви хочете використовувати

щось інше, ніж командний рядок.

3) Потрібно більше технічних знань. SSH - це в першу чергу інструмент для веб-розробників, а не щось для звичайного користувача.

4) Не скрізь доступний.

2.4.1 Підключення до Raspberry Pi за допомогою SSH

З міркувань безпеки Secure Shell не ввімкнено за замовчуванням у Raspbian. На своєму Raspberry Pi виберіть Меню> Налаштування> Конфігурація Raspberry Pi. Клацніть на Interfaces (Інтерфейси) та встановіть для SSH значення Enabled (Увімкнено). Клацніть OK. Вам не потрібно перезавантажувати Raspberry Pi, і SSH буде ввімкнено кожного разу, коли ви будете використовувати цю інсталяцію Raspbian з цього моменту (обов'язково оновіть свій пароль за замовчуванням, тобто „raspberry“).

Linux і macOS підтримують SSH нестандартно; перейдіть до кроку 3, якщо ви використовуєте одну з цих операційних систем.

Windows 10 підтримує SSH, але вам потрібно його активувати. Клацніть на Пошук і знайдіть «Управління додатковими функціями». Клацніть на ньому в Пошуку, щоб відкрити вікно Налаштування.

Клацніть на «Додати функцію» та зачекайте, поки завантажиться список Необов'язкових функцій. Прокрутіть список вниз до Відкрити клієнт SSH (бета-версія). Натисніть Встановити.

Підключіть Raspberry Pi до локальної мережі. Використовуйте бездротову локальну мережу або підключіть Raspberry Pi безпосередньо до маршрутизатора за допомогою кабелю Ethernet. Відкрийте вікно терміналу та введіть таку команду:

hostname

Ви повинні побачити:

Raspberrypi

У деяких випадках ви можете використовувати це ім'я хосту, що зручно, якщо у вас немає IP-адреси, але надійніше використовувати IP-адресу (протокол Інтернету). Введіть це, щоб отримати свою IP-адресу:

hostname -I

Це поверне чотири числа, розділені крапками. Наприклад, наш IP:

192.168.1.71

Запишіть це число. Вам це знадобиться незабаром.

Відкрийте командний рядок на ПК з Windows або вікні терміналу в Linux або macOS.

Введіть цю команду:

ssh pi@[IP]

Замініть [IP] на IP-адресу Raspberry Pi. У нашому випадку ми входимо

ssh pi@192.168.1.71

Він запитає: "Ви впевнені, що хочете продовжувати з’єднання?"

Введіть так і натисніть RETURN. Вам буде запропоновано ввести пароль для вашого Raspberry Pi.

2.5 Nginx

Nginx - це програмне забезпечення веб-сервера з відкритим кодом, яке також виконує послуги зворотного проксі-сервера, балансування навантаження, проксі-сервера електронної пошти та кешування HTTP. Програмне забезпечення спочатку було створено Ігорем Сисоєвим як відповідь на проблему обробки 10000 одночасних підключень користувачів.

Nginx забезпечує високу продуктивність веб-серверів із розширеним масштабуванням. Nginx здатний працювати на високих швидкостях при більших навантаженнях. Функція зворотного проксі-сервера дозволяє одному веб-сайту представляти агреговані джерела інформації так, ніби всі вони походять з однієї сторінки. Його балансування дозволяє розподіляти навантаження між різними ресурсами, такими як сервери.

Багато відомих компаній використовують Nginx для управління сторінками з високим трафіком, включаючи Autodesk, Facebook, Atlassian, LinkedIn, Twitter, Apple, Citrix Systems, Intuit, T-Mobile, GitLab, DuckDuckGo, Target, Intel, Microsoft, IBM, Google і Cisco.

Nginx масштабується настільки ефективно і працює швидше, ніж інше програмне забезпечення веб-сервера. На відміну від збірки Apache, Nginx не створює процес для кожного користувача. Натомість Nginx використовує структуру головного та робочого процесів. Головний процес керує робочими процесами, які виконують обчислення.

Nginx важливий, оскільки він був спеціально створений для екстремальних навантажень та ефективності. Програмне забезпечення веб-сервера допомагає ряду аспектів розміщення програм веб-сайтів та служб доставки контенту. Nginx - це друге за популярністю програмне забезпечення веб-сервера після Apache.

В даний час F5 Networks володіє Nginx, придбавши його за 670 мільйонів доларів у березні 2019 року. Програмне забезпечення розповсюджується з BSD-подібною ліцензією. Nginx безкоштовний, але пропонується також як Nginx Plus з платною підтримкою.

2.5.1 Зворотній проксі сервер Nginx

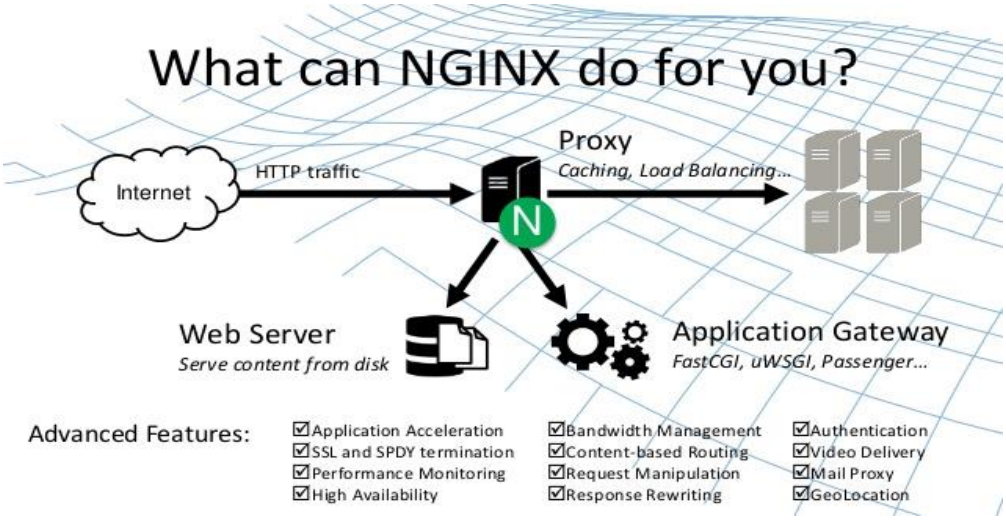

Рис.2.8–Nginx зворотній проксі

Зворотний проксі-сервер Nginx HTTPS - це посередницька проксі-служба, яка приймає запит клієнта, передає його одному або декільком серверам і надає відповідь сервера назад клієнту. Хоча більшість поширених додатків можуть працювати як веб-сервер самостійно, веб-сервер Nginx може забезпечити низку вдосконалених функцій, таких як балансування навантаження, можливості TLS / SSL та прискорення, яких бракує у більшості спеціалізованих додатків. Використовуючи зворотний проксі Nginx, всі програми можуть скористатися цими функціями.

Є важливі переваги налаштування зворотного проксі-сервера Nginx HTTPS:

1) *Балансування навантаження:* Зворотний проксі-сервер Nginx може виконувати балансування навантаження, що допомагає рівномірно розподіляти запити клієнтів між серверами. Це також покращує навантаження, оскільки якщо один сервер перестає працювати, зворотний проксі просто перенаправляє запити на інший сервер відповідно до політики маршрутизації.

2) *Підвищена безпека*: Зворотний проксі-сервер Nginx також діє як лінія захисту для ваших серверів. Налаштування зворотного проксі-сервера гарантує, що ідентифікація ваших серверних серверів залишатиметься невідомою.

3) *Краща продуктивність*: Відомо, що Nginx працює ефективніше з доставкою файлів статичного вмісту та аналізу URL-адрес

4) *Простота реєстрації та аудиту*: Оскільки існує лише одна точка доступу, коли реалізований зворотний проксі-сервер Nginx, це значно спрощує реєстрацію та аудит.

5) *Зашифроване з'єднання*: Шифруючи з'єднання між клієнтом та зворотним проксі-сервером Nginx за допомогою TLS, користувачі отримують прибуток від зашифрованого та захищеного з'єднання HTTPS, захищаючи свої дані.

2.6 SSL сертифікат

Сертифікат SSL - це цифровий сертифікат, який засвідчує справжність веб-сайту та забезпечує зашифроване з’єднання. SSL означає Secure Sockets Layer,

протокол безпеки, який створює зашифроване посилання між веб-сервером і веб-браузером.

Компанії та організації повинні додавати SSL-сертифікати на свої веб-сайти, щоб захистити транзакції в Інтернеті та зберегти конфіденційність та безпеку інформації про клієнтів.

Коротше кажучи: SSL захищає з’єднання з Інтернетом та заважає злочинцям читати або змінювати інформацію, передану між двома системами. Коли ви бачите піктограму замка поруч із URL-адресою в адресному рядку, це означає, що SSL захищає веб-сайт, який ви відвідуєте.

З часу свого існування близько 25 років тому існувало кілька версій протоколу SSL, які в певний момент зіткнулися з проблемами безпеки. Далі з’явилась оновлена та перейменована версія - TLS (Transport Layer Security), яка використовується і сьогодні. Однак ініціали SSL застрягли, тому нова версія протоколу все ще зазвичай називається старою назвою.

2.6.1 OpenSSL

OpenSSL - це бібліотека програмного забезпечення, що реалізує протоколи веб-безпеки SSL (захищений сокет) і TLS (захист транспортного рівня). SSL і TLS - це методи використання криптографії для забезпечення зв'язку між двома сторонами. Хоча на технічному рівні є деякі важливі відмінності (SSL в основному застарів на користь більш безпечного TLS), вони обидва працюють по суті однаково. (Насправді багато людей просто позначають обидва протоколи як "SSL".)

Після встановлення з'єднання між клієнтом та сервером, клієнт вимагає захищене з’єднання. Він запитує інформацію про те, які типи криптографічної безпеки підтримує клієнт.

Сервер вибирає найбільш безпечний варіант, який підтримують як сервер, так і клієнт, а потім надсилає сертифікат безпеки, підписаний відкритим ключем сервера.

Клієнт перевіряє сертифікат і генерує секретний ключ для надсилання на сервер, зашифрований відкритим ключем.

Клієнт і сервер використовують секретний ключ для створення пари симетричних ключів (або двох пар відкрито-приватних ключів), і спілкування розпочинається надійно.

SSL / TLS не є програмним забезпеченням чи технологією - це протокол, процедура для виконання вищезазначеної серії кроків, разом із конкретними криптографічними алгоритмами. Для реалізації протоколу потрібна частина програмного забезпечення .

OpenSSL - найпопулярніша реалізація SSL / TLS, що використовується зараз.

2.7 PHP

PHP (скорочення від PHP: Hypertext Preprocessor) - це мова сценаріїв, яка зазвичай використовується у веб-розробці на стороні сервера. Для того, щоб все це розібрати, важливо спочатку зрозуміти, що таке мова сценаріїв. Мови сценаріїв (сімейство мов програмування, включаючи PHP, а також такі мови, як JavaScript і Ruby) - це підмножина мов кодування, що використовуються для автоматизації процесів, які в іншому випадку повинні були б виконуватися поетапно в коді сайту кожного разу, коли вони виникають.

Сюди входять такі речі, як діалогові вікна, що відкриваються на екрані у відповідь на дії користувача, чат-боти, що відповідають на визначену поведінку користувача відповідними повідомленнями, або анімацією, яка відбувається, коли користувач прокручує повз певну точку сторінки - будь-які динамічні функції веб-сайту, які мають відбуваютися на екрані без необхідності користувачеві перезавантажувати сайт вручну. Мови сценаріїв, такі як PHP, відрізняються від мов розмітки, таких як HTML і CSS, у тому сенсі, що, хоча HTML і CSS визначають макет та вигляд веб-сторінок, мови сценаріїв вказують статичній веб-сторінці (побудованій за допомогою HTML та CSS) "робити" конкретні дії . Якщо ви

витратили якийсь час на читання про JavaScript, це може здатися звичним. Тож PHP - це ще один спосіб досягнення того, що можна зробити за допомогою JavaScript

2.7.1 Використання PHP

Як згадувалося раніше, PHP зазвичай використовується як мова на стороні сервера (на відміну від мови, такої як JavaScript, яка зазвичай виконується на стороні клієнта). То що це означає? З точки зору програмування, на стороні клієнта мається на увазі діяльність веб-сайту, яка відбувається локально на комп'ютері користувача через веб-браузер користувача. Мови на стороні клієнта, такі як HTML, CSS та JavaScript, дають вказівки, що веб-браузери можуть аналізувати та перетворювати вміст на екран вашого комп’ютера. Зверніть увагу, що в цьому списку є JavaScript (мова сценаріїв, як PHP). Знову ж таки, процеси, за сценарієм JavaScript, відбуваються на стороні клієнта - JS надає інструкції, які можуть бути зрозумілі та виконані у вашому веб-браузері. Клієнтська сторона - це сторона, яку ви бачите, коли користуєтесь Інтернетом.

2.7.2 Переваги та недоліки PHP

Переваги використання PHP Framework

1) Прискорити розробку веб-додатків

У наш час програмістам PHP доводиться писати веб-додатки на основі складних бізнес-вимог. Подібним чином вони повинні дослідити способи зробити веб-додаток більш багатим для користувачів. Інструменти, функції та фрагменти коду, надані платформами PHP, допомагають розробникам пришвидшити розробку власних веб-додатків.

2) Спростити обслуговування веб-додатків

На відміну від інших мов програмування, PHP не робить акцент на читабельності коду. Фреймворки PHP спрощують розробку та обслуговування веб-додатків, підтримуючи архітектуру Model-View-Controller (MVC). Розробники

можуть скористатися архітектурою MVC, щоб розділити веб-додаток на моделі, подання та контролери. Вони можуть використовувати фреймворк MVC для PHP, щоб тримати розділений інтерфейс користувача та рівні бізнес-логіки.

3) Не потрібно писати додатковий код

PHP, на відміну від інших мов програмування, не дозволяє програмістам висловлювати поняття без написання довших рядків коду. Отже, програмістам PHP доводиться писати довгий і складний код, додаючи функції або функціональність веб-сайту. Фреймворки PHP значно скорочують час кодування, забезпечуючи можливість генерації коду. Функції генерації коду, надані певними фреймворками PHP, дозволяють програмістам підтримувати вихідний код веб-додатків чистим.

4) Працюйте з базами даних ефективніше

Більшість фреймворків PHP дозволяють програмістам працювати з низкою широко використовуваних реляційних баз даних. Деякі фреймворки додатково спрощують операції з базами даних, забезпечуючи системи об'єктного реляційного відображення (ORM). Програмісти можуть скористатися перевагами систем ORM для виконання операцій з базами даних без написання тривалого коду SQL. ORM навіть дозволяє програмістам писати об'єктний код безпосередньо мовою програмування PHP.

5) Автоматизуйте загальні завдання веб-розробки

Створюючи веб-додаток, розробники повинні виконувати ряд завдань на додаток до написання коду. Деякі з цих загальних завдань веб-розробки вимагають від програмістів додаткових витрат часу та зусиль. Функції та інструменти, що надаються платформами PHP, допомагають розробникам автоматизувати загальні завдання веб-розробки, такі як кешування, управління сеансами, автентифікація та відображення URL-адрес.

6) Захистіть веб-сайти від цілеспрямованих атак безпеки

PHP вважається однією з найбільш незахищених мов програмування. Часто програмістам доводиться вивчати способи захисту програм PHP від різних атак безпеки. Вбудовані функції та механізми безпеки, передбачені фреймворком, полегшують розробникам захист веб-сайту від існуючих та нових загроз безпеці.

Крім того, веб-розробники PHP можуть легко запобігти поширеним загрозам безпеки, таким як введення SQL, підробка міжсайтових запитів та підробка даних.

7) Ефективно виконуйте модульні тести

Створюючи власний веб-додаток, розробники повинні регулярно проводити модульне тестування, щоб оцінити його окремі блоки або компоненти. Великий відсоток веб-розробників використовують PHPUnit для швидкого та ефективного виконання модульних тестів. Окрім того, що PHPUnit є об'єктно-орієнтованою структурою модульного тестування для PHP, додатково допомагає розробникам писати та запускати модульні тести, надаючи допомогу в кодуванні. Багато фреймворків PHP підтримують PHPUnit спочатку і дозволяють програмістам плавно виконувати модульне тестування.

8) Не потрібно збільшувати вартість веб-розробки

Як відкрита серверна мова програмування, PHP допомагає користувачам значно скоротити витрати на веб-розробку. Розробники також мають можливість вибрати з декількох веб-фреймворків з відкритим кодом для PHP. Вони навіть можуть скористатися функціями та інструментами, що надаються цими фреймворками з відкритим кодом, пришвидшити розробку власних веб-додатків, не збільшуючи накладні витрати на проект.

Недоліки використання PHP Framework

1) Програмістам потрібно вивчати PHP фреймворки замість PHP

Фреймворки PHP дозволяють програмістам додавати функціональність веб-додатку без написання додаткового коду. Але програмістам доводиться докласти трохи часу та зусиль, щоб вивчити фреймворк PHP. Вони навіть можуть вивчати та використовувати певні фреймворки, не володіючи PHP-кодуванням.

2) Якість PHP-фреймворків відрізняється

Найбільш широко використовувані фреймворки PHP є відкритими та безкоштовними. Отже, веб-розробники можуть скористатися цими фреймворками, не збільшуючи вартість проекту. Але сила спільноти окремих рамок відрізняється. Отже, деякі фреймворки PHP не мають швидкої та адекватної підтримки.

3) Відсутність можливості змінити основну поведінку

Окрім доведення базової структури для розробки веб-додатків, фреймворки PHP ще більше пришвидшують розробку власних веб-додатків. Але розробники все ще не мають можливості вносити зміни в основну поведінку цих фреймворків. Деякі фреймворки навіть вимагають від розробників використання конкретних інструментів або прийняття певної схеми веб-розробки.

4) Впливає на швидкість та продуктивність веб-сайтів

Більшість фреймворків PHP мають і функції та інструменти для пришвидшення розробки. Веб-розробники можуть не потребувати цих розширених функцій під час створення невеликих або простих веб-додатків. Крім того, ці додаткові функції часто негативно впливають на продуктивність та швидкість веб-сайтів.

Загалом, веб-розробники мають можливість вибору з декількох повнофункціональних та мікро-веб-фреймворків для PHP. Але широко використовувані фреймворки PHP відрізняються один від одного категорією функціональності, зручності використання та продуктивності. Крім того, кожен фреймворк PHP має свої плюси і мінуси. Ось чому веб-розробники повинні враховувати точні вимоги проекту, оцінюючи переваги та недоліки використання фреймворків PHP.

2.8 Linux

Кожного разу, коли ви вмикаєте комп’ютер, ви бачите екран, на якому ви можете виконувати різні дії, наприклад писати, переглядати Інтернет або дивитися відео. Що саме змушує апаратне забезпечення комп’ютера працювати так? Звідки процесор на вашому комп’ютері знає, що ви просите його запустити файл mp3?

Ну, це операційна система або ядро, яке виконує цю роботу. Отже, для роботи на комп’ютері потрібна операційна система (ОС). Насправді ви використовуєте такий, коли читаєте його на своєму комп’ютері. Зараз ви, можливо, використовували такі популярні ОС, як Windows, Apple OS X, але тут ми

дізнаємось про введення в операційну систему Linux, огляд Linux та про переваги, які вона пропонує перед іншими виборами ОС.

Linux - це операційна система або ядро, яке проростало як ідея у свідомості молодого та яскравого Лінуса Торвальда, коли він був студентом інформатики. Раніше він працював над ОС UNIX (запатентоване програмне забезпечення) і вважав, що вона потребує вдосконалення.

Однак, коли його пропозиції були відхилені дизайнерами UNIX, він задумав запустити ОС, яка буде сприйнятлива до змін, модифікацій, запропонованих її користувачами.

Тож Лінус розробив ядро під назвою Linux в 1991 році. Хоча для його роботи потрібні були програми, такі як Менеджер файлів, Редактори документів, Аудіо-Відео програми.

З часом він співпрацював з іншими програмістами в таких місцях, як MIT, і програми для Linux почали з'являтися. Отже, приблизно в 1991 р. Була офіційно запущена діюча операційна система Linux з деякими додатками, і це було початком роботи однієї з найулюбленіших та доступних на сьогоднішній день опцій ОС.

Ранні версії ОС Linux не були настільки зручними для користувача, їх використовували комп'ютерні програмісти, і Лінус Торвальдс ніколи не думав комерціалізувати свій продукт.

Це безумовно приборкало популярність Linux, оскільки інші комерційно орієнтовані операційні системи Windows стали відомими. Тим не менше, аспект з відкритим кодом операційної системи Linux зробив її більш надійною.

Головною перевагою Linux було те, що програмісти мали змогу використовувати ядро Linux для розробки власних власних операційних систем. З часом новий спектр зручних для ОС штурмував комп’ютерний світ. Зараз Linux є одним з найпопулярніших і широко використовуваних ядер, і він є основою таких популярних операційних систем, як Debian, Knoppix, Ubuntu та Fedora. Тим не менше, на цьому список не закінчується, оскільки доступні тисячі найкращих версій ОС Linux на базі ядра Linux, які пропонують різноманітні функції для користувачів.

Ядро Linux зазвичай використовується в поєднанні з проектом GNU доктором Річардом Столлманом. Усі сучасні дистрибутиви Linux насправді є дистрибутивами Linux / GNU

ОС Linux зараз користується популярністю у найвищому розквіті, і вона відома серед програмістів, а також серед звичайних користувачів комп'ютерів у всьому світі. Його головні переваги:

- Він пропонує безкоштовну операційну систему. Вам не потрібно витрачати сотні доларів, щоб отримати таку ОС, як Windows!

- Будучи відкритим кодом, кожен, хто володіє знаннями програмування, може його змінити.

- Легко вивчити Linux для початківців

- Операційні системи Linux тепер пропонують на вибір мільйони програм / програм та програмного забезпечення для Linux, більшість з яких безкоштовні!

- Після встановлення Linux вам більше не потрібен антивірус! Linux - це надзвичайно безпечна система. Більше того, існує спільнота світового розвитку, яка постійно шукає шляхи підвищення своєї безпеки.

- З кожним оновленням ОС стає більш безпечною та надійною

- Безкоштовна програма Linux є найкращою ОС для середовищ сервера завдяки своїй стабільності та надійності (такі мегакомпанії, як Amazon, Facebook та Google використовують Linux для своїх серверів). Сервер на базі Linux може працювати без зупинок без перезавантаження протягом багатьох років.

2.8.1 Навіщо використовувати Linux?

Користувачі, які не знайомі з Linux, зазвичай уникають його, помилково вважаючи його складною і технічною ОС для роботи, але, правду кажучи, за останні кілька років операційні системи Linux стали набагато зручнішими для користувачів, ніж їхні аналоги Windows, тому спробувати їх - найкращий спосіб дізнатись, підходить вам Linux чи ні.

На основі ядра Linux доступні тисячі найкращих ОС і програмного забезпечення Linux; більшість із них пропонують найсучасніші засоби безпеки та додатки, і все це безкоштовно!

Саме в цьому полягає Linux, і тепер ми перейдемо до того, як встановити Linux і який дистрибутив вам слід вибрати.

UNIX називають матір'ю операційних систем, які заклали основу Linux. Unix розроблений в основному для мейнфреймів і знаходиться на підприємствах та університетах. Хоча Linux швидко стає загальновідомим ім'ям для користувачів комп'ютерів, розробників та серверного середовища. Можливо, вам доведеться заплатити за ядро Unix, тоді як в Linux воно безкоштовне.

Але команди, що використовуються в обох операційних системах, як правило, однакові. Між UNIX та Linux не існує великої різниці. Хоча вони можуть здаватися різними, в основному вони по суті однакові. Оскільки Linux є клоном UNIX. Отже, вивчення одного - те саме, що вивчення іншого.

2.8.2 Linux Terminal

Термінал - це інтерфейс, за допомогою якого ви вводите загадкові команди Linux, але термінал - це просто вікно з командним рядком на вашому робочому столі Linux? Ну, справа в тому, що термінал, яким ви зараз користуєтеся, мабуть, не справжній термінал. У цій статті я поясню справжнє значення терміналу та консолі Linux. Розуміння цієї базової концепції важливо, якщо ви хочете освоїти Linux.

Як іменник термінал має багато значень. Словник Merriam-Webster дає нам гарне визначення терміналу в обчислювальній галузі:

комбінація клавіатури та пристрою виводу (наприклад, блок відображення відео), за допомогою якого дані можна вводити або виводити з комп’ютера або системи електронних комунікацій.

Персональні комп’ютери увійшли в повсякденне життя наприкінці 1970-х. До цього ми можемо використовувати лише дорогі мейнфрейми та мінікомп’ютери

Людина та комп’ютер - це дві незалежні сутності. Людині потрібен інтерфейс для введення інформації в комп’ютер і зчитування вихідних даних з нього. У наші дні пристрій введення персональних комп’ютерів включає: клавіатуру, мишу та мікрофон. Вихідний пристрій включає монітор і динамік. Їх зазвичай називають периферійними пристроями. Але у світі Unix та Linux пристрої введення та виводу називаються термінальними. Протилежністю терміналу є хост, який включає такі пристрої, як процесор, оперативна пам'ять, жорсткий диск тощо.

3.4.4 Daemon в Linux

У кожній системі Linux працює безліч процесів. Більшість із цих процесів можуть бути вам знайомими, якщо ви регулярно використовуєте команду типу ps або top для їх відображення. Процеси можуть виглядати просто як елемент у списку. Це насправді складні шматки коду, які приручає менеджер пам'яті. Щоб по-справжньому зрозуміти, як працює ваша система, знання управління процесами (або пам’яттю) є великою допомогою. Тож давайте зробимо стрибок у внутрішній частині Linux, вивчивши наявні у нас інструменти.

Кожна система має певну мету, яку хоче досягти. Такою метою може бути надання веб-сайту анонімним відвідувачам у всьому світі. Для того, щоб це зробити, слід щось прослуховувати запити окремих веб-сайтів, обробляти їх і, нарешті, відправляти назад відповідну сторінку веб-сайту. Ми називаємо це процесом, і він складається з машинного коду. Це індивідуальні вказівки щодо того, що повинна робити система. Ці інструкції включають читання зображення з жорсткого диска, надсилання даних через мережевий інтерфейс або збереження повідомлення про помилку у файлі журналу.

Процеси бувають різних форм. Найпоширеніший тип - це команди, які ви можете ввести в програму оболонки. Оболонка - це «обгортка» для вашої консолі Linux або екрана віртуального терміналу (при використанні SSH). Більшість користувачів використовують типову оболонку Bourne Again Shell (або Bash). Це

дозволяє вам вводити текст, і він діятиме на основі цього вводу. Наприклад, коли ви вводите таку команду, як ls, вона розглядає це як відому команду та виконує її.

Справжня магія відбувається, коли ви запускаєте команди. У цьому випадку оболонка вирішить запустити вбудовану дію або запустити програму з жорсткого диска. Ми називаємо ці програми на диску «двійковими». Сам цей двійковий файл зберігається у певному форматі, який зазвичай є ELF, або виконуваним та зв'язуваним форматом.

Деякі процеси мають на меті тривалий час працювати у системі у фоновому режимі. Це може бути для виконання таких запитів, як сканування вхідного електронного листа або повернення сторінки веб-сайту. Ці процеси називаються демонами. Окрім тривалості, ще одна велика різниця полягає в тому, що демонам не потрібна взаємодія з терміналом. Зазвичай вони не надсилають на нього жодних даних, а використовують натомість файли журналів. Демони часто запускаються безпосередньо після запуску операційної системи. У більшості в кінці назви процесу є знак „d", що натякає, що це процес демона.

Назва демон походить від експерименту, заснованого на демоні Максвелла, який мав завдання сортувати речі у фоновому режимі.

Варто пам'ятати: демон завжди є процесом, але не всі процеси є демоном.

Зазвичай термін "сервіс" використовувався в системах Windows. З введенням systemd цей термін тепер більш застосовний і до Linux. Послуга - це поєднання ресурсів для надання певної функціональності. Наприклад, служба SSH, яка складається із запуску відповідного демона та будь-яких залежностей, таких як мережа.

Існує багато інформації для збору та показу запущених процесів. Загальні інструменти для цієї роботи включають команду ps та top.

Ядро Linux - це складна машина сама по собі. Це утворює міст між апаратним та програмним забезпеченням. Його головна мета - переконатися, що обидві сторони поводяться, поки обробляють якомога більше запитів. Складне завдання з

апаратним забезпеченням перериває постійний заклик до уваги, а програмне забезпечення, як відомо, іноді менш стабільне, ніж очікувалося.

Щоб врахувати все, що працює в системі, ядру потрібно відстежувати кожен рух в системі. Особливо потребує уваги управління пам’яттю. Пам'ять поділяється на кілька зон, а потім надається для запущених процесів. Щоб запобігти неправильному використанню, є охоронці, які стежать за запитами на збільшення пам’яті. Мета полягає в тому, щоб запобігти закінченню пам'яті (OOM). В іншому випадку вбивцю OOM потрібно звільнити з клітки, і він почне вбивати процеси, щоб звільнити пам’ять. Інші охоронці гарантують, що один процес не може бачити дані інших процесів, що буде погано для безпеки. Подібним чином існують засоби захисту, які запобігають неправильному використанню сегментів пам'яті, як-от область даних, яка раптово запускає (зловмисний) машинний код.

Деякі внутрішні дані, що підтримуються ядром, також корисні для системного адміністратора. Для цього використовується псевдо-файлова система / proc. У цій файловій системі для кожного PID створюється каталог. У Linux все є файлом. Отже, кожен каталог складається з купи файлів. Більшість з них можна переглянути за допомогою команди cat.

2.9 Ansible

Ansible спрощує автоматизацію IT, залучаючи масив IT-ресурсів та підтримуючи розгортання багаторівневості з 1-го дня. Ansible консолідує ресурси в декількох системах для управління ними з однієї платформи, а не вимагає управління з однієї системи одночасно. Кодом, життєвим циклом та змінами можна керувати через інвентар, ігрові книги та ролі.

Така система управління конфігурацією, як Ansible, складається з декількох компонентів. Керовані системи можуть включати сервери, сховища, мережі та програмне забезпечення. Це цілі системи управління конфігурацією. Метою є підтримка цих систем у відомих, визначених станах. Іншим аспектом системи

управління конфігурацією є опис бажаного стану системи. Третім головним аспектом системи управління конфігурацією є програмне забезпечення для автоматизації, яке відповідає за те, щоб цільові системи та програмне забезпечення підтримувались у бажаному стані.

Використання Ansible значно скорочує час конфігурації та розгортання.

Перш ніж я скажу вам, що таке Ansible, надзвичайно важливо зрозуміти проблеми, з якими стикалися до Ansible.

Давайте трохи повернемося до початку мережевих обчислень, коли надійне та ефективне розгортання та управління серверами було проблемою. Раніше керували серверами вручну, встановлюючи програмне забезпечення, змінюючи конфігурації та адмініструючи послуги на окремих серверах.

Це також ускладнило швидкість роботи розробників, оскільки команда розробників була спритною і часто випускала програмне забезпечення, але IT-операції витрачали більше часу на налаштування систем. Ось чому інструменти надання серверів та управління конфігурацією стали процвітати.

Нам завжди потрібно постійно оновлювати, надсилати зміни, копіювати файли на них тощо. Ці завдання роблять речі дуже складними та трудомісткими.

Але дозвольте сказати вам, що вирішення вищезазначеної проблеми є. Рішення - відповідне.

Але перш ніж я продовжу пояснювати вам все про Ansible, дозвольте мені ознайомити вас з кількома термінологіями Ansible:

Відповідальні умови:

Машина контролера: машина, на якій встановлено Ansible, відповідальна за запуск підготовки на серверах, якими ви керуєте.

Інвентар: Файл ініціалізації, який містить інформацію про сервери, якими ви керуєте.

Playbook: Точка входу для надання Ansible, де автоматизація визначається за допомогою завдань, що використовують формат YAML.

Task: Блок, який визначає одну процедуру, яку потрібно виконати, напр. Встановіть пакет.

Module: Модуль зазвичай абстрагується від системних завдань, таких як робота з пакетами або створення та зміна файлів. Ansible має безліч вбудованих модулів, але ви також можете створювати власні.

Role: заздалегідь визначений спосіб організації ігрових книжок та інших файлів з метою полегшення обміну та повторного використання частин резервування.

Play: Резервування, що виконується від початку до кінця, називається відтворенням. Простими словами, виконання playbooks називається play.

Facts: Глобальні змінні, що містять інформацію про систему, наприклад, мережеві інтерфейси або операційну систему.

Handlers: Використовується для активації змін стану служби, таких як перезапуск або зупинка служби.

Ansible - це корисний інструмент, який дозволяє створювати групи машин, описувати, як ці машини слід налаштовувати або які дії слід вживати з ними. Ansible видає всі команди з центрального місця для виконання цих завдань.

Жодне інше клієнтське програмне забезпечення не встановлюється на машинах вузлів. Він використовує SSH для підключення до вузлів. Ansible потрібно встановити лише на контрольній машині (машині, з якої ви будете виконувати команди), яка може бути навіть вашим ноутбуком. Це просте рішення складної проблеми.

Переваги Ansible, згадані нижче:

Спрощена автоматизація

Ansible - це проста у використанні платформа, проста в установці та налаштуванні, з дуже швидким темпом навчання. Менше ніж за 30 хвилин можна встановити та налаштувати систему та виконати спеціальні команди для серверів для вирішення конкретної проблеми: регулювання переходу на літній час, синхронізація часу, зміна пароля root, оновлення серверів, перезапуск служб тощо.

Ansible легко розгорнути, оскільки він не використовує жодних агентів або додаткової власної інфраструктури безпеки. Він також використовує YAML - просту мову для опису вашої роботи з автоматизації за допомогою ігрових книг.

Книги Play висувають бажані налаштування на хости, визначені в інвентарі, і навіть можуть запускатися спеціально (через командний рядок, не вимагаючи визначень у файлах).

З того моменту, як ви можете пінгувати хостів через Ansible, ви можете почати автоматизувати своє середовище. Почніть із невеликих завдань, дотримуючись найкращих практик, розставляючи пріоритети завданням, які додають вартості бізнесу, вирішують основні проблеми, заробляють час і покращують продуктивність.

Не відставати від темпів ведення бізнесу може спричинити кілька проблем; надання інфраструктурних ресурсів не повинно бути одним із них. При використанні Ansible на будь-якій платформі надання ресурсів стає простим, автоматизованим і повторюваним з 1-го дня. Ви можете автоматизувати трудомісткі IT-завдання та стимулювати спільну культуру для підтримки ініціатив DevOps.

Ansible зазвичай групується разом з іншими інструментами управління конфігурацією, такими як Puppet, Chef, SaltStack тощо. Ну, дозвольте сказати, Ansible не обмежується лише управлінням конфігурацією. Його також можна використовувати різними способами. Деякі з них я згадав нижче:

Надання: Ваші програми повинні десь жити. Якщо ви PXE (Preboot eXecution Environment) завантажуєте та запускаєте голі металеві сервери або віртуальні машини або створюєте віртуальні чи хмарні екземпляри із шаблонів, Ansible & Ansible Tower допомагає впорядкувати цей процес. Наприклад, якщо я хочу протестувати налагоджувальну версію програми, побудованої за допомогою Visual C ++, я повинен відповідати деяким необхідним вимогам, наприклад, наявність бібліотек DLL бібліотеки Visual C ++ (msvcr100d.dll). Мені також знадобиться Visual Studio, встановлений на вашому комп'ютері. Це коли Ansible переконує, що необхідні пакети завантажені та встановлені для надання моєї програми.

Управління конфігурацією: воно встановлює та підтримує послідовність роботи продукту шляхом запису та оновлення детальної інформації, що описує апаратне та програмне забезпечення підприємства. Така інформація зазвичай

включає версії та оновлення, що застосовуються до встановлених програмних пакетів, а також розташування та мережеві адреси апаратних пристроїв. Для напр. Якщо ви хочете встановити нову версію Tomcat на всіх машинах, присутніх у вашому підприємстві, вам неможливо вручну перейти та оновити кожну машину. Ви можете встановити Tomcat одним рухом на всіх своїх машинах, використовуючи ігрові книги та інвентар Ansible, написані найпростішим способом. Все, що вам потрібно зробити, - це перерахувати IP-адреси своїх вузлів в інвентарі та написати програму для встановлення Tomcat. Запустіть посібник із вашої контрольної машини, і він буде встановлений на всіх ваших вузлах.

Розгортання програми: Коли ви визначаєте свою програму за допомогою Ansible та керуєте розгортанням за допомогою Ansible Tower, команди можуть ефективно управляти всім життєвим циклом програми від розробки до виробництва. Наприклад, скажімо, я хочу розгорнути движок сервлетів за замовчуванням. Для розгортання двигуна потрібно виконати ряд кроків.

Перемістіть програму .war із каталогу dropins в каталог додатків

Додайте файл server.xml

Перейдіть на веб-сторінку, щоб побачити свою програму.

Але навіщо турбуватися про виконання цих кроків по одному, коли у нас є такий інструмент, як Ansible. Все, що вам потрібно зробити, це перерахувати ці завдання у вашій книзі ігор Ansible і сидіти, спостерігаючи, як Ansible виконує ці завдання по порядку.

Безпека та відповідність: Коли ви визначаєте свою політику безпеки в Ansible, сканування та виправлення загальнополітичної політики безпеки можуть бути інтегровані в інші автоматизовані процеси. І це буде невід’ємною частиною всього, що розгортається. Це означає, що вам потрібно налаштувати деталі безпеки один раз у вашій машині управління, і вони будуть автоматично вбудовані у всі інші вузли. Більше того, усі облікові дані (ідентифікатори та паролі адміністраторів та паролі), які зберігаються в Ansible, не можуть бути отримані у звичайному тексті будь-яким користувачем.

Оркестрація: Конфігурації самі по собі не визначають ваше середовище. Вам потрібно визначити, як взаємодіють кілька конфігурацій, і забезпечити можливість керування різними частинами в цілому. Зі складності та хаосу Ансибл наводить порядок. Ansible забезпечує Оркестрацію у сенсі узгодження бізнес-запиту з програмами, даними та інфраструктурою. Він визначає політики та рівні обслуговування за допомогою автоматизованих робочих процесів, забезпечення та управління змінами. Це створює пристосовану до додатків інфраструктуру, яку можна масштабувати або зменшувати залежно від потреб кожного додатка.

Наприклад, розглянемо ситуацію, коли я хочу розмістити новий веб-сайт замість мого існуючого. Для цього ми видалимо існуючий веб-сайт і розгорнемо наш новий веб-сайт і перезапустимо балансир навантаження або веб-кластер, якщо це необхідно. Тепер, якби ми просто зробили щось подібне, користувачі помітили б простої, оскільки ми не видалили поточний трафік, що надходить на ці машини, через балансування навантаження. Отже, нам потрібен певний тип попереднього завдання, де ми просимо балансировщик навантаження перевести цей веб-сервер в режим обслуговування, щоб ми могли тимчасово відключити трафік від переходу на нього в міру оновлення. Скажімо, я додав сюди блок, який говорить про те, що передзавданням буде відключення веб-вузла в балансаторі навантаження.

Отже, це наше попереднє завдання, де ми вимикаємо трафік, а потім тут, оновлюємо вузол, використовуючи ці різні завдання. Нарешті, нам потрібен певний тип післязавдання, який дозволить знову залучити трафік до цього веб-вузла, виключивши його з режиму обслуговування. Ці завдання можна записати в ігрові книги Ansible.

Завдяки доданню більше 60 нових модулів до бібліотеки Ansible забезпечує найнадійнішу інтеграцію з Ansible будь-якого постачальника сховищ на ринку. Завдяки цій обширній бібліотеці модулів, користувачі Ansible можуть легко розробляти та розгортати playbook для автоматизації завдань зберігання даних без необхідності вивчати нюанси конкретного продукту.

2.10 Docker

Docker - це інструмент з відкритим кодом, який автоматизує розгортання програми всередині програмного контейнера. Найпростіший спосіб зрозуміти ідею, яка стоїть за Docker, - порівняти її зі стандартними контейнерами для транспортування.

Рис.2.9 Порівняння докер контейнеру з фізичним

Тоді як транспортні компанії стикалися з такими проблемами:

Як транспортувати різні (несумісні) типи товарів поруч (наприклад, продукти харчування та хімікати, скло та цегла).

Як обробляти пакунки різних розмірів за допомогою одного і того ж транспортного засобу.

Після введення контейнерів цеглу можна було покласти на скло, а хімікати зберігати поруч з продуктами. Вантаж різних розмірів можна помістити всередину стандартизованого контейнера і завантажити / розвантажити одним і тим же транспортним засобом.

Коли ви розробляєте додаток, вам потрібно надати свій код разом із усіма можливими залежностями, такими як бібліотеки, веб-сервер, бази даних тощо. Ви можете потрапити в ситуацію, коли програма працює на вашому комп'ютері, але навіть не запускається на проміжному сервері, на розробнику або на машині контролю якості.

Цю проблему можна вирішити, ізолювавши програму, щоб зробити її незалежною від системи.

Традиційно для уникнення цієї несподіваної поведінки використовували віртуальні машини. Основна проблема віртуальної машини полягає в тому, що “зайва ОС” поверх операційної системи хоста додає до проекту гігабайти місця. Велику частину часу на вашому сервері буде розміщено кілька віртуальних машин, які займуть ще більше місця. І до речі, на даний момент більшість постачальників хмарних серверів стягуватимуть з вас додатковий простір. Ще одним суттєвим недоліком ВМ є повільне завантаження.

Docker усуває все вищесказане, просто розподіляючи ядро ОС у всіх контейнерах, що працюють як окремі процеси головної ОС.

На діаграмі нижче показано, як це виглядає на прикладі Docker.

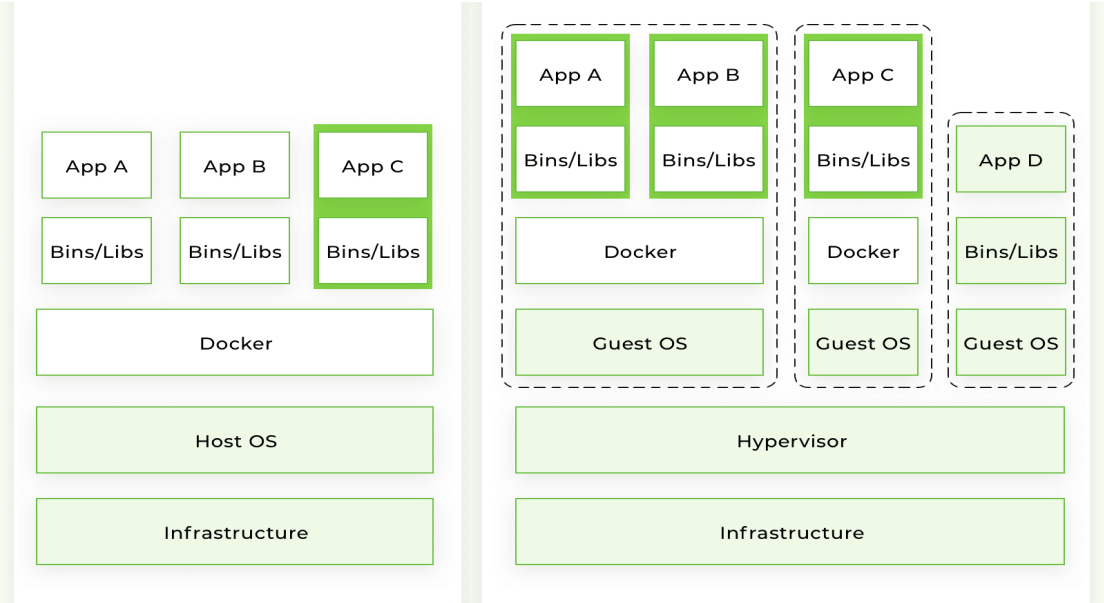

Рис.2.10 Схематичне зображення роботи докеру

Майте на увазі, що Docker - не перша і не єдина платформа для контейнеризації. Однак на даний момент Docker є найбільшим і найпотужнішим гравцем на ринку.

Короткий перелік переваг включає:

Швидший процес розвитку

Зручна інкапсуляція додатків

Така сама поведінка на локальних машинних / розробницьких / проміжних / виробничих серверах

Простий і зрозумілий моніторинг

Легко масштабується

Швидший процес розвитку

Немає необхідності встановлювати в систему сторонні програми, такі як PostgreSQL, Redis, Elasticsearch - ви можете запускати їх у контейнерах. Docker також дає можливість одночасно запускати різні версії одного і того ж додатка. Наприклад, скажімо, вам потрібно виконати ручну міграцію даних зі старої версії Postgres на нову версію. У архітектурі мікросервісів може бути така ситуація, коли ви хочете створити нову мікросервіс з новою версією сторонніх програм.

Зберігати дві різні версії одного додатка в одній хост-ОС може бути досить складно. У цьому випадку контейнери Docker можуть бути ідеальним рішенням - ви отримуєте ізольоване середовище для своїх програм та сторонніх розробників.

Крім того, залежно від вашого стеку, ви можете керувати декількома версіями мови програмування в контейнерах Docker: Python 3.9 та Python 3.7, наприклад.

Зручна інкапсуляція додатків

Ви можете подати свою заявку цілим. Більшість мов програмування, фреймворки та всі операційні системи мають своїх менеджерів упаковки. І навіть якщо вашу програму можна упакувати за допомогою власного диспетчера пакетів, створити порт для іншої системи може бути важко.

Docker надає уніфікований формат зображення для розподілу ваших програм між різними хост-системами та хмарними службами. Ви можете доставити свою заявку цілим із усіма необхідними залежностями (включеними до зображення), готовими до запуску.

Допомагаючи командам розробників з доставкою та розгортанням упакованих програм на Python, контейнери можуть збільшити зручність і швидкість процесу розробки.

Docker зменшує майже до нуля ймовірність помилок, спричинених різними версіями операційних систем, системними залежностями тощо.

При правильному підході до створення образів Docker ваша програма використовуватиме одне і те ж базове зображення з тією ж версією ОС та необхідними залежностями.

2.11 Redis

"Redis", що означає Remote Dictionary Server. За словами офіційного представника Redis, Redis - це сховище структур даних з відкритим кодом (з ліцензією BSD), яке використовується як база даних, кеш-пам’ятки та посередник повідомлень. З іншого боку, AWS зазначає, що Redis - це швидкий сховище даних із відкритим кодом, що зберігається в пам'яті, для використання в якості бази даних, кешу, посередника повідомлень та черги.

Коли програма покладається на зовнішні джерела даних, затримка та пропускна здатність цих джерел може створити вузьке місце в роботі, особливо при збільшенні трафіку або масштабах програми. Одним із способів поліпшити продуктивність у цих випадках є зберігання та обробка даних у пам'яті, фізично ближче до програми. Redis розроблений для цього завдання: він зберігає всі дані в пам'яті - забезпечує найшвидшу можливу продуктивність при зчитуванні або записі даних - і пропонує вбудовані можливості реплікації, які дозволяють розміщувати дані фізично ближче до користувача для найнижчої затримки.

База даних в пам'яті - це база даних, яка зберігає весь набір даних в оперативній пам'яті. Що це означає? Це означає, що кожного разу, коли ви запитуєте базу даних або оновлюєте дані в базі даних, ви отримуєте доступ лише до основної пам'яті. Отже, в цих операціях не задіяний Диск.

Інші характеристики Redis, на які варто звернути увагу, включають підтримку декількох структур даних, вбудований скрипт Lua, кілька рівнів стійкості на диску та високу доступність.

Redis відрізняється від «традиційних» сховищ даних NoSQL як допоміжний компонент, розроблений спеціально для підвищення продуктивності додатків. Ось декілька диференційованих можливостей Redis:

Сеанси кешування Redis

Знову ж таки, на відміну від баз даних NoSQL, таких як MongoDB та PostreSQL, Redis зберігає дані в основній пам'яті сервера, а не на жорстких дисках і твердотільних накопичувачах. Це призводить до значно швидшого часу відгуку під час виконання операцій читання та запису. Це також допомагає забезпечити високу доступність (разом із Redis Sentinel) та масштабованість служб та робочих навантажень додатків.

Черги Redis

Redis може поставити в чергу завдання, для обробки яких веб-клієнтам потрібно більше часу, ніж зазвичай. Багатопроцесорна чергування завдань є звичним явищем у багатьох сучасних веб-додатках, і Redis спрощує впровадження автоматизованих процесів, написаних на Python, які працюють у фоновому режимі циклів запитів / відповідей.

Типи даних Redis

Хоча технічно сховище ключ / значення, Redis є фактичним сервером структури даних, який підтримує кілька типів даних та структур, включаючи наступне:

Унікальні та несортовані рядкові елементи

Бінарні дані

HyperLogLogs

Бітові масиви

Хеш

Списки

Обробка клієнтів Redis

Redis має можливості інтеграції власних клієнтів, які допомагають розробникам маніпулювати та взаємодіяти зі своїми даними. В даний час у клієнтській бібліотеці Redis доступно понад 100 різних клієнтів з відкритим кодом,

і розробники можуть легко додати нові інтеграції для підтримки додаткових функцій та мов програмування.

Redis Sentinel

Redis Sentinel (посилання знаходиться за межами IBM) - це окрема розподілена система, яка допомагає розробникам калібрувати свої екземпляри, щоб бути високодоступними для клієнтів. Sentinel використовує низку процесів моніторингу, сповіщень та автоматичних відмов, щоб інформувати користувачів про те, що щось не так з екземплярами master і slave, при необхідності автоматично переконфігуруючи нові з'єднання для програм.

Кластер Редіс

Кластер Redis (посилання знаходиться за межами IBM) - це розподілена реалізація Redis, яка автоматично розділяє набори даних між кількома вузлами. Це підтримує більш високу продуктивність та масштабованість розгортання баз даних, забезпечуючи при цьому безперервні операції в тому випадку, якщо підмножини вузлів не можуть взаємодіяти з рештою кластера.

Redis persistence

Redis використовує постійне дискове, призначене для перебоїв у процесі роботи та вузьких місць у мережі. Redis може зберігати набори даних, роблячи регулярні знімки даних та додаючи їх із змінами, коли вони стають доступними. Потім Redis можна налаштувати для створення цих резервних копій бази даних на вимогу або з автоматичними інтервалами, щоб забезпечити довговічність та цілісність бази даних.

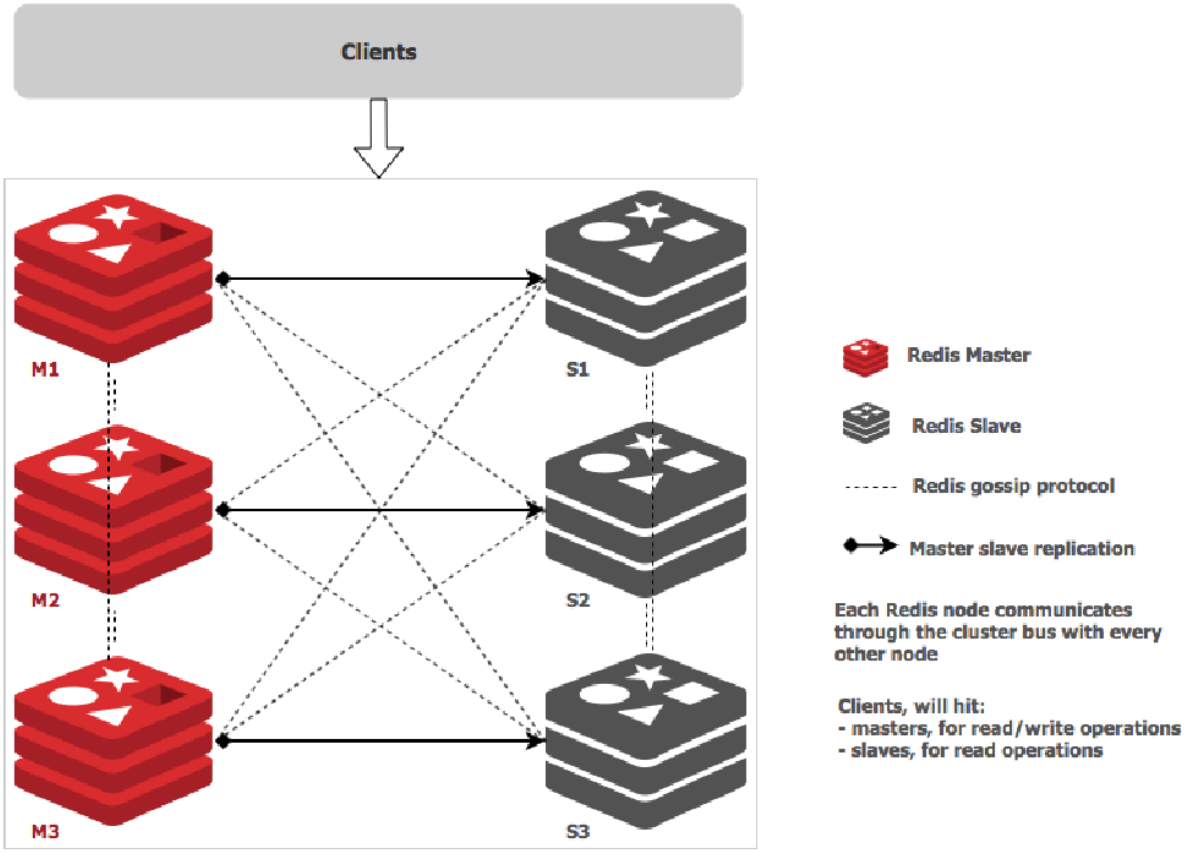

Рис. 2.11 Схема кластерів Редісу

Ось кілька типових випадків використання, які вигідні підприємствам при роботі з Redis:

Аналітика в режимі реального часу: оскільки Redis може обробляти дані із затримкою до мілісекунд, вона ідеально підходить для аналітики в режимі реального часу, рекламних кампаній в Інтернеті та процесів машинного навчання, керованих ШІ.

Додатки, що базуються на розташуванні: Redis спрощує розробку додатків та послуг, що базуються на розташуванні, забезпечуючи геопросторову індексацію, набори та операції. Використовуючи відсортовані набори, Redis може розвантажити трудомісткий пошук та сортування даних про місцезнаходження, а також використовуючи інтелектуальну реалізацію геогешування.

Кешування для баз даних: Redis здатний обробляти великі обсяги даних у режимі реального часу, використовуючи свої можливості зберігання даних у пам'яті, щоб допомогти підтримувати швидко реагуючі конструкції баз даних.

Кешування за допомогою Redis дозволяє зменшити кількість звернень до бази даних, що допомагає зменшити кількість трафіку та необхідних екземплярів. Використовуючи Redis для кешування, команди розробників можуть суттєво покращити пропускну здатність своїх програм, досягнувши затримки, що перевищує мілісекунди. І оскільки шар кешування Redis може масштабуватися швидко та економічно, організації можуть розробляти ці програми, що швидко реагують, зменшуючи загальні витрати.

2.12 Mysql

MySQL - це система управління реляційними базами даних із відкритим кодом (СУБД), яку можна легко впровадити та керувати як локально, так і через хмару через хостинг-провайдера. Він підтримує безліч одночасних записів та масштабування за допомогою реплікації (хоча це може ускладнитися). З цієї причини та через відносно низькі витрати на обслуговування / масштабованість, він в основному використовується як виробнича база даних.

Створений у 1994 році, MySQL є однією з найбільш створених СУБД і з тих пір значно розвинувся. Дев'яносто відсотків веб-сайтів використовують MySQL. Сюди входять такі гігантські сервіси, як Youtube, Twitter, Wikipedia та Facebook.

Можливості з відкритим кодом

MySQL був побудований як рішення з відкритим кодом, тому його можна використовувати на товарному обладнанні та масштабувати з передбачуваними витратами. Широка сумісність MySQL також означає, що її можна в будь-який час переключити на більш адаптоване рішення (багато конкуруючих баз даних транзакцій сумісні з MySQL саме для цієї мети).

Висока продуктивність

MySQL - це еквівалент бази даних гоночного автомобіля, який позбавлений задніх сидінь, щоб зробити його швидшим. Розробники MySQL прийняли рішення надати пріоритет швидкості та продуктивності над можливостями, що робить

MySQL більш швидкою, хоча і більш обмеженою базою даних, ніж інші провайдери в категорії транзакційних баз даних.

Доступність та масштабованість

Можливість реплікації та розповсюдження MySQL для високої доступності та масштабованості надзвичайно потужна. Однак слід зазначити, що більш високі рівні доступності та масштабованості повинні бути збалансовані вищою складністю та витратами.

Веб-додатки

MySQL був в першу чергу розроблений для веб-додатків. Це особливо чудово для структурованих та добре спланованих веб-додатків. Oracle створив чудовий ресурс, в якому описується, чому MySQL є придатним рішенням для веб-програм.

Нові компанії без складної команди даних

MySQL часто є першим вибором для невеликих компаній або стартапів через свою надійність, повсюдність та продуктивність. MySQL також є привабливим варіантом, оскільки легко створити копію виробничої бази даних MySQL для використання в якості аналітичної бази даних. MySQL має надійну спільноту розробників, є однією з найкраще задокументованих систем БД на ринку і продається великою кількістю постачальників.

MySQL підтримує транзакції з інтеграцією механізмів BDB та InnoDB (механізм за замовчуванням). Це дозволяє безпечніше обробляти паралельні операції запису, що започаткувало тенденцію додавання функцій, необхідних середовищам Enterprise.

MySQL працює на Linux, Solaris, Windows, AIX та HPUX і пропонує як 32-розрядні, так і 64-розрядні версії.

MySQL краще працює з добре структурованими транзакційними даними та даними, структурованими у третій звичайній формі. MySQL також буде краще працювати з меншими, повними наборами даних. Він не любить розріджені дані або широкі таблиці. Чим рідше стіл, тим важче писати; чим ширша таблиця, тим важче її читати.

MySQL може підтримувати великі або навіть дуже великі таблиці, але з набагато більшою складністю та фінансовими витратами. Оскільки MySQL значною мірою залежить від індексації, знання, які типи запитів ви будете виконувати заздалегідь, може сильно змінити ефективність.

Транзакція означає виконання декількох операцій як один блок. Найважливішою особливістю є те, що або всі операції виконуються правильно в транзакції, або жодна з них не може бути виконана. Отже, транзакції дозволяють програмістам переривати / відкликати вже виконані команди - це лише один приклад. Загалом ця функція спрощує всі процеси. MySQL також підтримує транзакції, хоча не для табличного формату MyISAM. Однак ви можете застосувати інші формати в MySQL і забезпечити цю корисну функцію.

ВИСНОВОК ДО РОЗДІЛУ 2

Другий розділ вийшов найбільш інформативним, у ньому ми дізнались про різні технології і продукти.

Мікрокомп’ютер Raspberry PI кількість моделей якого підійде для будь-яких потреб, один з найпопулярніших веб-серверів nginx який ми будемо використовувати для доступу до сховища та ansible софт CI/CD за допомогою якого зробимо деплой нашого сховища автоматичним і відтворюваним

Мова PHP хоч і застаріла, але все ще оновлюється і має велику кількість прихильників, кількість готових бібліотек, доступних фреймворків які можуть пришвидшити розробку тим самим зєкономивши гроші замовника, але якщо планується довгострокова підтримка продукту - тоді варто подумати перед тим як обирати PHP.

3 ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ. ВСТАНОВЛЕННЯ НЕОБХІДНИХ КОМПОНЕНТІВ.

3.1 Owncloud

Оскільки в наш час так багато служб, як iCloud і Dropbox, зламуються, не дивно, що все більше людей хочуть витягувати свої дані з хмари. Замість того, щоб втратити ці чудові функції синхронізації, ви можете створити власну службу хмарного сховища, якою ви керуєте за допомогою служби під назвою ownCloud. З його допомогою ви отримаєте синхронізуючі файли, нотатки, календарі тощо.

OwnCloud - це безкоштовне програмне забезпечення з відкритим кодом, яке діє як дуже простий спосіб створити власну синхронізуючу хмарну систему зберігання, схожу на Dropbox, на власному сервері чи веб-сайті. Він досить надійний, що замінив мені Dropbox у всіх, крім кількох випадків вибору. Це також швидке та просте налаштування, яке не вимагає передових технічних знань. OwnCloud настільки ж потужний, як Dropbox, але він також дозволяє людям створювати та ділитися своїми програмами, які працюють на ownCloud, включаючи текстові редактори, списки завдань тощо. Це означає, що ви можете отримати від цього трохи більше, ніж просто синхронізувати файли, якщо хочете.

По суті, ownCloud пропонує надзвичайно просту синхронізацію файлів з робочого столу в хмару. Як і Dropbox, ви можете отримувати доступ до своїх файлів з будь-якого місця, синхронізувати дані та ділитися файлами з іншими.

Крім цього, ви також отримуєте музичний плеєр, вбудований безпосередньо у ownCloud, просте місце для зберігання контактів, диспетчер завдань, календар синхронізації, послугу закладок та надійну фотогалерею. Ви зможете синхронізувати ownCloud практично з будь-яким настільним або мобільним календарем та програмою контактів. Це означає, що якщо ви хочете відмовитися від подібних iCloud, ownCloud робить це легко. Нещодавнє оновлення також додало простий спосіб встановлення, так що кожен може відразу почати використовувати ownCloud.

Для початку роботи з ownCloud вам не потрібно багато. Просто збирайся:

Веб-хост, який підтримує PHP5 та MySQL (або SQLite): Це може звучати трохи жаргонно, але все це означає, що вам потрібно зареєструватися в такій службі, як Dreamhost (якщо ви ще цього не зробили). Якщо у вас вже є доменне ім'я, наприклад http://www.yourname.com, через веб-хостинг (і потрібно), можливо, ви можете встановити ownCloud за пару хвилин. Це звучить складно, але насправді вам не потрібно мати справу з такими речами, як PHP та MySQL для простої інсталяції ownCloud. Тому, що Ansible зробить усе за вас автоматично. Просто переконайтеся, що ваш сервіс хостингу їх підтримує.

Оскільки ви, мабуть, захочете скористатись власним Cloud з будь-якого місця, для цього вам знадобиться URL-адреса. Якщо у вас ще немає доменного імені, ви можете придбати його.

Приємна річ у ownCloud полягає в тому, що він сумісний практично з будь-яким сервером, який ви можете собі уявити. Ряд постачальників послуг пропонують встановлення в один клік і такі господарі, як Dreamhost, навіть надають власні інструкції з встановлення. Вам також знадобиться поглянути на Умови використання веб-хостингу, щоб переконатися, що вони прямо не забороняють налаштовувати власне хмарне сховище на своїх серверах.

Особливості:

Відкривши платформу, можна об'єднатись з OwnCloud інших та обмінюватися файлами.

Легко завантажувати документи.

Є мобільний додаток, але у платній версії

Масштабоване та легке рішення для зберігання у хмарі.

Швидкий та інтуітивно зрозумілий

Немає онлайн-редактора документів.

3.2 Nextcloud

Nextcloud - це програма для розміщення файлів із відкритим кодом, яка дозволяє користувачам створювати власну мережу хмарних сховищ, використовуючи власні сервери або персональні комп'ютери. Написаний на PHP та Javascript, Nextcloud був випущений на початку 2020 року і швидко став основним елементом для користувачів, які хочуть створити власне рішення для хмарного сховища з відкритим кодом, над яким вони мають повний контроль. Це на відміну від інших служб, таких як Google Drive та Dropbox, які розміщуються на сторонніх серверах поза вашим прямим контролем. Для користувачів, які хочуть мати більше контролю над тим, як і де зберігаються їх дані, Nextcloud є очевидним вибором.

Nextcloud сумісний з Windows, macOS, а також різними дистрибутивами Linux, що робить його універсальним рішенням для користувачів із різноманітними пристроями, що використовуються у їхній хмарній мережі зберігання даних. Як модульний компонент програмного забезпечення, Nextcloud можна налаштувати за допомогою плагінів, які надають додаткові функції та функціональність. Через App Store користувачі можуть придбати схвалені плагіни, які включають календарі, списки контактів, фотогалереї, потокове передавання медіа та багато іншого.

Крім того, Nextcloud дозволяє користувачам генерувати загальнодоступні URL-адреси для обміну файлами, минаючи необхідність використання FTP або інших методів передачі файлів. Файли Nextcloud передаються через WebDAV і шифруються під час транзиту, що робить його надійною та ефективною платформою обміну файлами. Крім того, програмне забезпечення розроблено для інтеграції із системами управління базами даних, такими як MySQL та Oracle Database, додаючи додатковий рівень управління даними до вашої хмарної мережі зберігання даних. Завдяки такому різноманітному набору функцій стає зрозуміло, чому стільки користувачів почали використовувати Nextcloud для своїх потреб у сховищі.

Особливості:

Nextcloud пропонує кращі можливості зв’язку за допомогою простого у використанні веб-інтерфейсу.

Максимальний захист завдяки наскрізному шифруванню

Потужна повнотекстова пошукова система

Гнучкий інтерфейс, який базується на плагінах, дозволяє шукати файли та підтримку.

Шифрування як локального, так і віддаленого архівування

Nextcloud забезпечує прозорий перехід на аудіо / відео дзвінки та чат.

3.3 Seafile

Seafile - це хмарне рішення для зберігання та синхронізації даних, що розміщується на платформі, з відкритим вихідним кодом. Іншими словами, це, як не дивно, також схоже на Dropbox або Google Drive, за винятком того, що ви повністю контролюєте свій екземпляр платформи. Таким чином, Seafile працює в прямій конкуренції з Nextcloud та Owncloud.

Програмне забезпечення Seafile складається з трьох частин: серверного програмного забезпечення, робочого столу та програм для синхронізації мобільних пристроїв та програмного забезпечення Drive. Це останнє лише для робочих столів і створює віртуальний диск для доступу та завантаження файлів, що зберігаються на вашому сервері Seafile.

Цей 100% безкоштовний і відкритий код спільноти Seafile. Це випускається під загальною публічною ліцензією GNU Affero v3, за підтримки спільноти, яку надає форум Seafile.

Існує також власна професійна версія лише для Linux, яка включає підтримку електронної пошти та деякі додаткові функції, орієнтовані на корпоративні середовища.

Якщо ви розміщуєте Seafile на сторонньому сервері, тоді, звичайно, будуть пов’язані витрати на оренду сервера.

Особливості:

Міжплатформна синхронізація з повною підтримкою для Windows, macOS, Linux, Android та iOS

Віртуальне відображення дисків

Шифрування на стороні клієнта (наскрізне)

Вбудована підтримка документів Wiki

Версія файлів та знімки

Блокування файлів, щоб запобігти одночасному редагуванню файлів, не створюючи конфліктів

Інтернет-редагування та співавторство

Журнал аудиту для моніторингу вашої системи

Мобільне завантаження фото

Спільний доступ до файлів та контроль дозволів

Двофакторна автентифікація

Вбудоване сканування вірусів

Підтримка WebDAV

3.4 Підсумки аналізу ринку опенсорс ПЗ хмарних сховищ.

Під час пошуку ПЗ було проаналізовано багато опенсорс рішень, найпопулярніші представлені вище, це Owncloud, його форк Nextcloud та Seafile.

Після аналізу стало зрозуміло що між Owncloud і Nextcloud різниці майже немає крім того, що Nextcloud більш продуманий під підприємства, у ньому є чат, календар, і ща більше 100 різних додатків, але працює повільніше за Owncloud, для використання у продакшені я б обрав Seafile тому що він працює швидше, стабільніше , але у нього є суттєвий недолік для домашнього використання – він зберігає файли у своєму форматі навідміну від попередніх (вони зберігають файли у такому вигляді в якому їх поклали).

Отже зваживши усі за і проти я обрав Owncloud.

3.5 Підготовка до встановлення усіх необхідних компонентів

Для початку, нам необхідно оновити нашу операційну систему та усі системні пакети. Для цього, ми під'єднуємося до нашої Raspberry Pi за допомогою SSH клієнту.

Виконуємо наступні команди:

Оновлення списку пакетів

```
sudo apt-get update
```

Встановлюємо пакет який далі нам надасть можливість додавання PPA репозиторіїв

```
sudo apt-get install software-properties-common
```

Додаємо репозиторій Ansible

```
sudo add-apt-repository --yes --update ppa:ansible/ansible
```

Встановлюємо Ansible

```
sudo apt-get install ansible
```

3.6 Автоматизація за допомогою Ansible

У цьому розділі ми створимо playbook за допомогою якої і автоматизуємо розгортання нашої хмарної системи. Для початко створимо папку під назвою private_cloud.

3.6.1 Створення папок проекту

Концепція ролі Ansible проста: це група змінних, завдань, файлів та обробників, що зберігаються у стандартизованій структурі файлів. Важкою частиною є згадування структури каталогів, бо ansible дуже чутливий до структури директорій і файлів.

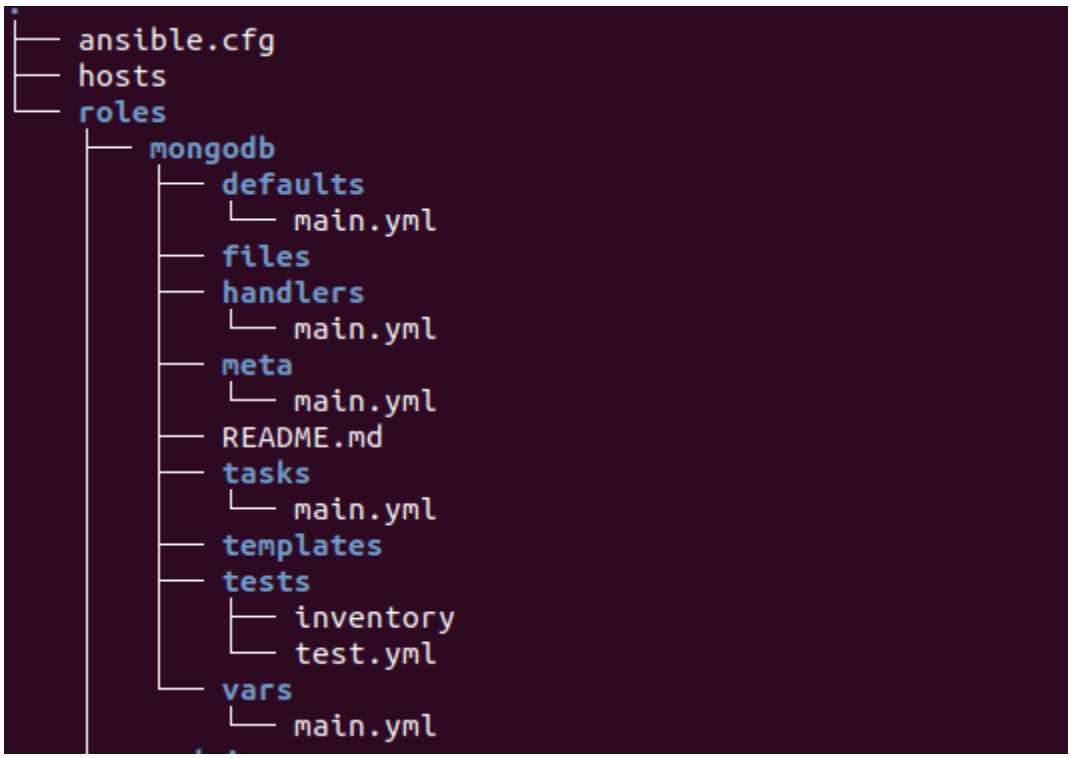

Рис. 3.1 Структура ansible

Тепер заходимо у папку private_cloud і створюємо дерево папок і файлів:

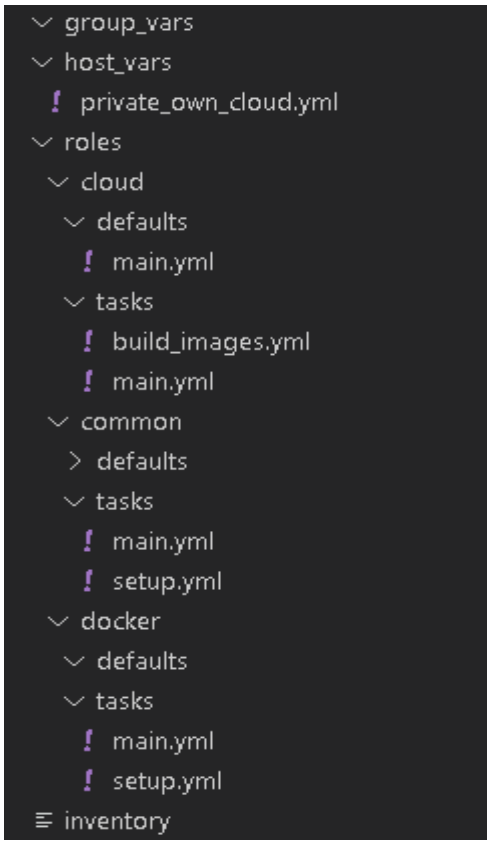

Рис 3.2 Заповнення директорії

3.6.2 Створення inventory

Файл хостів Ansible або файл інвентаризації повідомляє Ansible про хости, до яких він може підключитися.

Щоб Ansible автоматизував сервер Linux, мережевий пристрій або хмарний сервер, він повинен існувати в інвентарі (також відомий як файл хостів Ansible) і зберігатись у форматі YAML або INI.

Файл також може бути статичним або динамічно створюватися сценарієм. Усі варіанти будуть розглянуті в цьому підручнику нижче.

У файлі inventory додаємо рядок:

```
private_own_cloud
```

Здатність Ansible інтелектуально об’єднувати змінні з хосту / групи варіантами, а також отримати доступ до цих змінних на хості - це чудова особливість. А також якщо потрібно виконати завдання на одному хості та отримати доступ до змінних з іншого хосту. Одним із варіантів використання може бути динамічне збирання IP-адреси основи.

Файл host_vars/private_own_cloud.yml

```
ansible_remote: 192.168.48.46
```

3.6.3 Робота із залежностями

Файл common/tasks/main.yml

```
- name: Install aptitude
  apt:
    name: aptitude
    state: latest
    update_cache: yes
    force_apt_get: yes
```

```
# Встановлюємо бібліотеки необхідні для роботи docker та ansible
- name: Install dependencies
  apt:
    name: "{{ item }}"
    state: latest
    update_cache: yes
  with_items:
    - apt-transport-https
    - ca-certificates
    - curl
    - software-properties-common
    - python3-pip
    - virtualenv
    - python3-setuptools
```

3.6.4 Docker

```
# Додаємо ключі від репозиторію
- name: Add Docker GPG apt Key
  apt_key:
    url: https://download.docker.com/linux/ubuntu/gpg
    state: present

# Додаємо репозиторій
- name: Add Docker Repository
  apt_repository:
    repo: deb https://download.docker.com/linux/ubuntu bionic stable
    state: present
```

```
# Встановлюємо докер
- name: Update apt and install docker-ce
  apt:
    update_cache: yes
    name: docker-ce
    state: latest

# Встановлюємо модуль docker for python для роботи з ansible
- name: Install Docker Module for Python
  pip:
    name: docker
```

3.6.5 Ініціалізація змінних

```
# Main variables
private_cloud_container_name: owncloud-server
mariadb_container_name: owncloud-mariadb-server
redis_container_name: owncloud-redis-server
domain_name: "{{ private_cloud_container_name }}"

# Dry run credentials
dry_run_username: cloud
dry_run_password: cloud

# Mysql variables
mysql_user: "{{ dry_run_username }}"
mysql_password: "{{ dry_run_password }}"
```

```
mysql_root_password: "{{ dry_run_password }}"
mysql_db_name: cloud

# Web authentication
admin_username: "{{ dry_run_username }}"
admin_password: "{{ dry_run_password }}"

# Змінні для контейнерів
containers:
  - name: "{{ private_cloud_container_name }}"
    image: owncloud/server
    restart_policy: always
    recreate: yes
    volumes: "{{ private_cloud_container_name }}-data"
    env:
      DOMAIN={{ domain_name }}
      DB_TYPE=mysql
      DB_NAME={{ mysql_db_name }}
      DB_USERNAME={{ mysql_user }}
      DB_PASSWORD={{ mysql_password }}
      ADMIN_USERNAME={{ admin_username }}
      ADMIN_PASSWORD={{ admin_password }}
      MYSQL_UTF8MB4=true
      REDIS_ENABLED=true
      DB_HOST={{ mariadb_container_name }}
      REDIS_HOST={{ owncloud-redis-server }}

  - name: "{{ mariadb_container_name }}"
    image: mariadb:10.5
    restart_policy: always
```

```
    recreate: yes
    volumes: "{{ mariadb_container_name }}-data"
    env:
      MYSQL_DATABASE={{ mysql_db_name }}
      MYSQL_USER={{ mysql_user }}
      MYSQL_PASSWORD={{ mysql_password }}
      MYSQL_ROOT_PASSWORD={{ mysql_root_password }}

  - name: "{{ owncloud-redis-server }}"
    image: redis:6
    restart_policy: always
    recreate: yes
    volumes: "{{ owncloud-redis-server }}-data"
```

3.6.6 Генерація контейнерів для роботи хмари

```
# Пулимо останні версії докер імейджів
- name: Pull images
  docker_image:
    name: "{{ item.image }}"
    source: pull
  with_items: "{{ containers }}"

# Створюємо контейнери на основі змінних із roles/cloud/defaults/main.yml
- name: Create containers
  docker_container:
    name: "{{ item.name }}"
```

```
    image: "{{ item.image }}"
    restart_policy: "{{ item.restart_policy }}"
    recreate: "{{ item.recreate }}"
    volumes: "{{ item.volumes }}"
    env: "{{ item.env }}"
    state: present
  with_items: "{{ containers }}"
```

3.6.7 Створення playbook

Playbook використовує формат YAML для визначення однієї або декількох play. Playbook - це набір упорядкованих завдань, які упорядковані таким чином, щоб автоматизувати процес, наприклад, налаштування веб-сервера або деплою.

```
- hosts: cloud
  become: yes
  roles:
    - common
    - docker
    - cloud
```

ВИСНОВОК ДО РОЗДІЛУ 3

У цьому розділі ми обрали найбільш оптимальній софт для нашого домашнього хмарного сховища Owncloud через переваги над іншими продуктами.

Після проведення аналізу було розроблено автоматизоване розгортання персональної хмари, а також Redis для зберігання сессій та Mysql на Raspberry Pi за допомогою ansible .

ВИСНОВКИ

У даній дипломній роботі було представлено вплив хмарних технологій, а саме хмарних сховищ, на життя кожного з нас.

Хмарні технології дають нам можливість отримувати свої дані з будь-якої точки планети де є інтернет, ділитися ними або отримувати їх від інших людей без необхідності використання флешок і подібних пристосувань які легко втратити, забути або вони можуть вийти з ладу.

У роботі були розглянуті найпопулярніші сервіси хмарних сховищ які знаходяться у відкритому доступі у кожного з них є свої переваги і недоліки для різних варіантів їх застосування.

Також було розглянуто програмне забезпечення необхідне для створення персонального екземпляру хмарного сховища такі як мова програмування PHP, бази даних Mysql, Redis і т.д. і була розроблена система автоматизованого розгортання або поновлення софта для такої системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. База проектів для Rasbperry Pi - https://pimylifeup.com/
2. Основний пошук ресурсів - https://pimylifeup.com/
3. Допоміжний пошуковий ресурс - https://duckduckgo.com/
4. Nginx - https://nginx.org/en/docs/
5. Raspberry Pi офіційний ресурс- https://nginx.org/en/docs/
6. Завантаження NOOBS - https://www.raspberrypi.org/downloads/
7. Хмарні сховища – https://uk.wikipedia.org/wiki/%D0%A5%D0%BC%D0%B0%D1%80%D0%BD%D1%96_%D1%81%D1%85%D0%BE%D0%B2%D0%B8%D1%89%D0%B0
8. Raspberry Pi OS https://www.raspberrypi.org/documentation/raspbian/