Шановний голово та члени екзаменаційної комісії, вашій увазі пропонується кваліфікаційна робота за темою: «Модель використання CI/CD процесів під час розгортання інфраструктури на основі IaC»

Виконала: курсант Ксендзук Олена Олегівна

Керівник: викладач кафедри 21 майор Балан Дмитро Дмитрович

Актуальність

У сучасній боротьбі з противником, котрий має значну чисельну перевагу в майже усьому нам вкрай необхідно мати перевагу у веденні мережоцентричної війни, таким чином створивши асиметричну перевагу.

Нагадаю, що мережево-центрична війна орієнтована на досягнення інформаційної переваги за допомогою об'єднання військових об’єктів (наприклад, інформація розвідданих з багатьох можливих джерел) у одну інформаційну систему та мережу.

Побудова системи з мережоцентричними принципами без використання CI/CD та IaC буде вкрай складною. Проблеми, з якими доведеться зіткнутися наступні:

1) Повільна розробка продукту.

2) Значне зниження частоти релізу оновлень.

3) Великі ризики несумісності компонентів.

4) Складність масштабування.

5) Складність, або навіть неможливість повноцінного та комплексного тестування продукту.

6) Велике навантаження на персонал.

Хоча теоретично побудувати таку систему можливо, проте на практиці відсутність IaC та CI/CD зведе нанівець ключові переваги мережоцентричного підходу - гнучкість, адаптивність та швидке реагування. Система стане повільною, ненадійною та вразливою до помилок. Втім це лише один із найважливіших прикладів застосувань

Слайд 5

CI/CD це безперевна інтеграція та безперевна доставка.

Частинка CI стосується практики регулярного збирання та тестування коду програми після кожного злиття у центральний репозиторій коду. Це допомагає виявляти конфлікти та помилки на ранніх стадіях циклу розробки – це суттєва перевага CI.

Щодо наступних його переваг це: можливість частіших релізів, автоматичне тестування, гнучкість та економія часу за рахунок автоматизації.

CD розширює концепцію CI, автоматизуючи процес релізу програмного забезпечення для кінцевих користувачів. Основні кроки CD (*Delivery*):

1. Виконання процесу CI (збирання, тестування коду).

2. Створення образу або пакунку для розгортання.

3. Розгортання збірки у тестове або проміжне середовище.

4. Запуск додаткових тестів (навантажувальне тестування, тести користувацького інтерфейсу тощо).

Важливо зауважити, що є 2 види CD: Delivery та Deployment. Різниця в тому, що Deployment (Безперервне розгортання) повністю автоматичний процес, який після усіх тестів сам закидає готовий код у виробниче середовище (prod). А Delivery (Безперевна доставка) не автоматичний, але автоматизований процес, при якому потрібне ручне схвалення DevOps.

Слайд 6

На слайді зображений типовий виробничий процес.

З використанням CI/CD ми зменшуємо його час. Ось приклад, що відбувається, коли розробник закомітив зміни.

Конвеєр розгортання (pipeline) після будь-якої зміни перевіряє, що код успішно інтегрується у середовище, близьке до виробничого.

За такого сценарію час розгортання вимірюється хвилинами або, у гіршому разі, годинами, а карта потоку цінності має виглядати приблизно так, як показано на рисунку.

Слайд 7

Інфраструктура як код - це парадигма, в якій вся інфраструктура ІТ-систем (обчислювальні ресурси, сховища даних, мережі тощо) визначається як версійований машинозчитуваний код замість ручного налаштування ресурсів.

Сьогодні більшість проєктів живе в «хмарах». Тобто ресурси — це віртуальні машини, контейнери, *load balancers*.

Утім, побутує міф про небезпеку використання хмарних сервісів для інформаційних систем, пов’язаних із сектором оборони, проте справжній досвід виразно показав, що такі побоювання являються марними. Як приклад можна навести досвід використання системи ситуаційної обізнаності *Delta.*

Слайд 8

Проаналізувавши різні IaC інструменти я обрала саме Ansible та Terraform, зокрема тому що це найпопулярніші і прості рішення. Розгляньмо детальніше про їх особливості

Слайд 9

В контексті Інфраструктури як код, *Terraform* це засіб ініцілізації ресурсів. Засоби ініціалізації ресурсів створюють самі ресурси, тобто віртуальні машини, кластер *Kubernetes*, бази даних, мережі, системи моніторингу, сертифікати SSL і майже будь-який аспект інфраструктури.

На слайді показана візуалізація як Terraform працює. DevOps планує зміни, застосовує їх використовуючи різні провайдери, зокрема як у прикладі, які власне розгортають цю ж інфраструктуру.

Слайд 10

Особливості: багатопровайдерна підтримка, декларативний синтаксис, ідемпотентність, зручне управління ресурсами (таких як віртуальні машини, мережі, бази даних), планування змін, автоматичне визначення залежності ресурсів, модульність та повторне використання, легка інтеграція з CI/CD

Слайд 11

Ansible - це потужний інструмент для автоматизації конфігурації та управління системами.

Базове середовище *Ansible* складається з трьох основних компонентів:

1) Вузол управління (*Control node*) де встановлено *Ansible*.

2) Керований вузол (*Managed node*) - віддалена система або хост, яким керує *Ansible*.

3) Інвентар (*Inventory*) - Список керованих вузлів, які логічно організовані.

Слайд 12

Ось деякі ключові особливості Ansible: агентлес архітектура, YAML, ідемпотентність, модульна архітектура, playbooks та ролі, повторне використання та модульність, багатоплатформеність.

Слайд 13 (оцей слайд можна викинути, щоб точно вкластися у 7 хвилин)

Важливим аспектом створення програмного продукту є тестування. DevOps забезпечує різні стратегії розгортання для QA (тестувальника).

*Blue/Green*: стратегія зображена на слайді, вона передбачає створення двох ідентичних виробничих середовищ (*blue* та *green*). Одне середовище активне, а інше використовується для розгортання нової версії. Для релізу нової версії ми розгортаємо його в неактивне середовище, де можна виконати тестування, не втручаючись у роботу користувачів.

Шаблон синьо-зеленого розгортання дуже простий в реалізації і може значно підвищити безпечність релізу ПЗ. Є інші:

На рисунку нижче представлені групи середовищ стратегії canary, створені в компанії Facebook, які використовувалися для підтримки цього шаблона.

Канарковий При використанні цієї стратегії нова версія спочатку розгортається на невелику частину продуктивних ресурсів. Якщо нова версія працює коректно, вона поступово розгортається на решту ресурсів.

Слайд 14

На даному слайді зображена модель.