

Computer Network Final Project

- 題目

基於NFV技術的ChoiceNet實現

- 學號姓名

611001024 游凱雯

- 研究動機

1. ChoiceNet

現今網路服務度供應服務商大多壟斷，導致消費者對於網路服務沒有很多的選擇，而是被動的被網路服務所選擇，同時網路服務供應商也沒有動力去提供新的網路服務。

在ChoiceNet中，多家底層網路供應商提供多種網路服務，供消費者做選擇，消費者可以根據需求和衡量預算選擇最適合自己的商品。同時，在有同業競爭下，供應商也會有動力去改善、創新網路服務，加快網路技術的進步。

現今網路服務的消費者對於服務內容往往像是黑盒子一般，摸不透是否為自己所需要的服務內容，亦或是是否網路服務供應商有達到當初合約訂定的標準，這也使消費者權益受損。

而在ChoiceNet機制下，提供自省機制來監控網路服務。

2. SDN → NFV → ChoiceNet based on NFV

在SDN章節認識到NFV，學習到NFV可以靈活調節網路功能的特點，同時ChoiceNet也有提供較為多樣化服務此特性，因此決定使用NFV來製作ChoiceNet概念模擬的工具。

- 研究方法

1. 建立ChoiceNet Control Plane

以NFV建立網路環境：使用Kubernetes自動化佈署管理cluster，並讓client對在economy plane所購買的服務對應之cluster有使用權限。

用戶使用網路服務時，提供介面可以作效能測試，作為introspection。

2. 建立ChoiceNet Economy Plane

實作用戶介面，提供用戶關於網路服務的資訊並讓用戶試用網路服務。

3. 模擬ChoiceNet Data Plane

根據用戶選擇的網路服務方案，在設定好用戶權限後，提供該服務供客戶使用。

- 實驗環境

1. Docker

Docker 是一個開源的容器技術，可以讓應用程式及其依賴關係在任何環境中執行。它使用輕量的容器，可以在沒有影響性能的情況下將應用程式與其所需的環境隔離。這樣可以確保應用程式在不同的環境中都能正常運

作。

2. Kubernetes

Kubernetes是一種開源容器管理系統，可以幫助開發人員和系統管理員在雲端或在本地部署、管理和維護容器化應用程式。它提供了自動化部署、擴展、收縮、疏散和維護等功能，可以大大簡化容器環境的管理。

3. GCP GKE

Google Cloud Platform 的 Google Kubernetes Engine 是一個容器管理平台，可以在在 Google Cloud 上快速建立、部署和管理容器化應用程式。它使用 Kubernetes 作為其核心架構，提供自動化部署、擴展、縮減和故障恢復功能。它還提供了一個易於使用的管理介面和豐富的監控和報告功能，可以實時監控應用程式和容器。使用 Google Kubernetes Engine 可以大大簡化容器應用程式的管理和運維工作，因此可以更專注於應用程式的開發和創新。

• 實驗結果

1. 建立ChoiceNet Control Plane

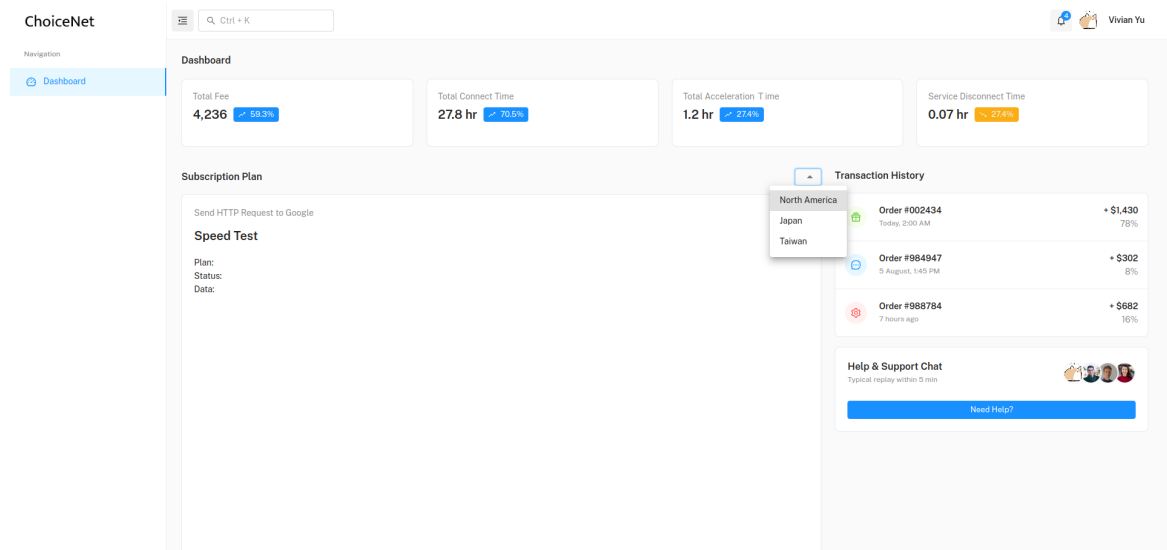
此處使用GKE建立了三個cluster，分別部屬在台灣、日本和美國，在cluster中使用kubernetes控制各個pods上之proxy是否正常運作。

此處較為誇張的模擬，部屬在美國的cluster當中的proxy-server，因為美國較為遙遠，因此設定的proxy建立connect的時間，使用者試用時能明顯感受時間較長，日本居中，位於台灣的使用者使用部屬於台灣的proxy則速度較快，模擬選擇到不同地區的proxy-server會有的情況，並在頁面上列出幾個方案讓客戶試用並根據自己所在的地區做選擇。

用戶使用網路服務時，在介面上提供網路服務測試之數值，作為introspection。

2. 建立ChoiceNet Economy Plane

做了一個網站來作為用戶使用介面，提供用戶關於網路服務的資訊並讓用戶試用網路服務。



▼ 程式碼路徑：source_code/proxy

2. HTTP Server

```
cd source_code/http_server
python app.py
```

3. Frontend

```
cd source_code/frontend
npm install
npm start
```

• 研究心得

這次作業前面花了很多時間在學習如何使用Docker和Kubernetes等等實用的技術，以前只有聽別人提到過這些技術產生的原因和理念，卻沒有實際操作過，正好有動機將這些技術初淺的學起來，了解這些技術後覺得非常實用，花費的前置學習時間很值得。

上課學習到的ChoiceNet給我很大的實做想像空間，雖然時間關係下只是做了一個借用其概念但也許偏離ChoiceNet本意，並且有些沒有實用價值的小模擬，但是對我來說已經獲益良多！

• 參考文獻

1. The course on Udemy:

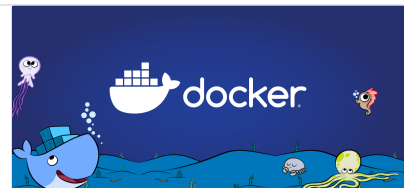
<https://www.udemy.com/course/docker-kubernetes-the-practical-guide/>

2. Docker:

Docker: Accelerated, Containerized Application Development

Docker is a platform designed to help developers build, share, and run modern applications. We handle the tedious setup, so you can focus on the code.

 <https://www.docker.com/>



3. Google Kubernetes Engine documentation:

Google Kubernetes Engine documentation | Google Kubernetes Engine (GKE) | Google Cloud

Best practice Best practices for running cost-optimized Kubernetes applications on GKE Take advantage of the elasticity provided by Google Cloud when running cost-optimized applications on GKE.

Modernization path for .NET applications on Google Cloud Learn a gradual and structured process for

 <https://cloud.google.com/kubernetes-engine/docs>



4. MUI:

MUI: The React component library you always wanted

Skip to content MUI offers a comprehensive suite of UI tools to help you ship new features faster. Start with Material UI, our fully-loaded component library, or bring your own design system to our production-ready components. Get started npm install @mui/material

 <https://mui.com/>




The React UI library
you always wanted

5. Node.js for linux:

Node.js

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

 <https://nodejs.org/en/>

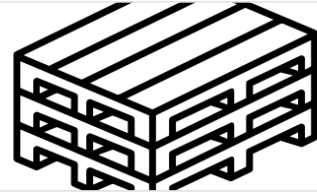


6. Flask:

Flask

Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular

 <https://palletsprojects.com/p/flask/>



7. Boost.Asio(for timer):

第一章 Boost.Asio入门

首先，让我们先来了解一下什么是Boost.Asio？怎么编译它？了解的过程中我们会给出一些例子。然后在发现Boost.Asio不仅仅是一个网络库的同时你也会接触到Boost.Asio中最核心的类-- io_service 。简单来说，Boost.Asio是一个跨平台的、主要用于网络和其他一些底层输入/输出编程的C++库。计算机网络的设计方式有很多种，但是Boost.Asio的方式远远优于其它的设计方式。它在2005年就被包含进Boost，然后被大

<https://mmoay.gitbooks.io/boost-asio-cpp-network-programming-chinese/content/Chapter1.html>