

為瞬間巨量做好準備

以 Nginx on AWS 為例

黃一樓

20180726

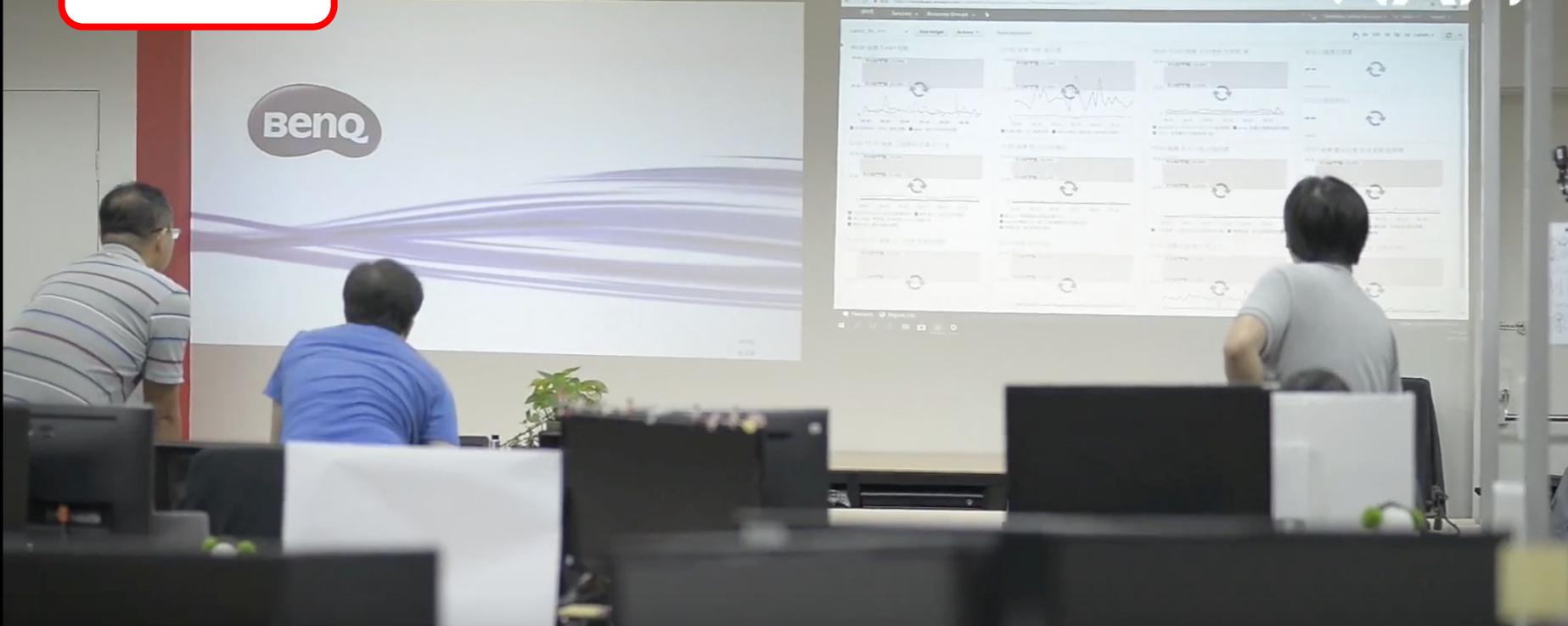
讓全公司可以凝聚在同一個目標上面



圖片出自電影<社群網戰>片段

幕後

91APP



<https://www.youtube.com/watch?v=VvYHzfIKZV4>

幕前

實體品牌迎戰雙11

康星美
COSMED

SO NICE

Timberland

LIVE IN
LOT'S

PHILIPS

FamilyMart



2017 實體品牌迎戰雙11 活動正式開始

準備這件事的**心情**是...

很有**壓力**
也很有**挑戰性**

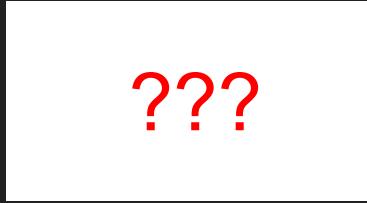
□ 課程F2：Hop & Pop
低空 + 第一次 Solo jump

<https://www.youtube.com/watch?v=8l4Ve9htJg&feature=youtu.be>

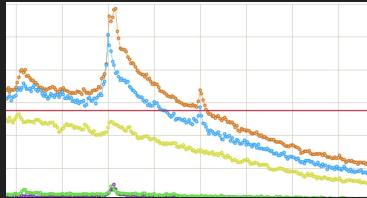
AFF 課程



為雙十一準備



上場



就敢上場了嗎？

準備了，不一定會活下來

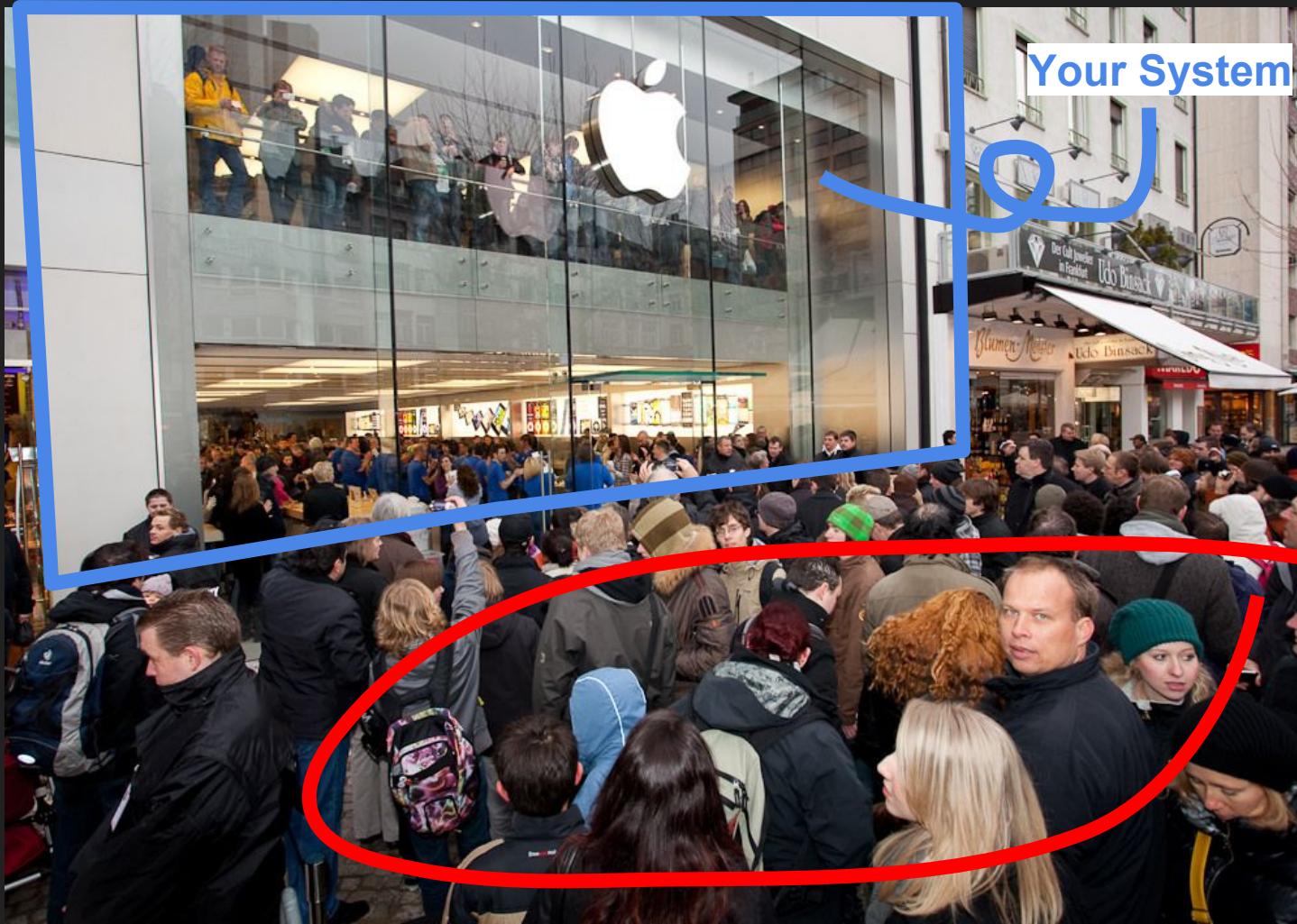
但是不準備就一定會死

而且要用專業，謹慎的態度做準備

老闆還會一直問你：

面對雙十一的大流量
我們的系統沒問題嗎？

寶潔，你怎麼看？



This is 今天的主題

Capacity Test

Benchmark

RPS (Request Per Second)

QPS (Query Per Second)

同時上線人數

單位時間流量

- 針對特定的應用情境
- 定義出一套相同的行為來對系統進行每一次的量測
- 量測結果可以用來比較不同配置、等級、架構之間的能力,
- 這個過程就叫做 Benchmark ← 動詞

針對特定的應用情境
相同的行為
比較不同的配置

Capacity Plan

延伸閱讀：

- [Complete Think - 淺談效能測試](#)
- [3 Types of Load Testing Every Company Should Run](#)
- [Benchmark\(Wikipedia\)](#)

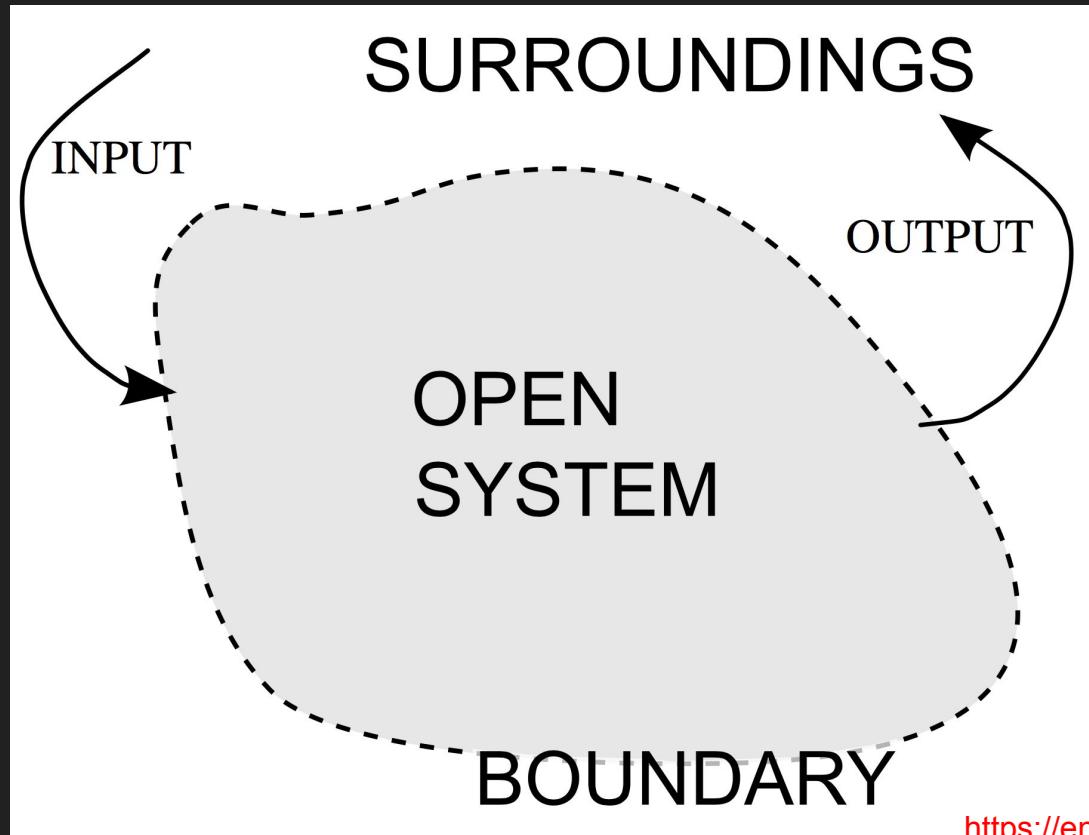
Stress Test
Scalability Test
Stability Test
等等...

今天分成兩個部分

上半場：**心法**

下半場：**案例分享**

先了解你的測試對象 (待測體)



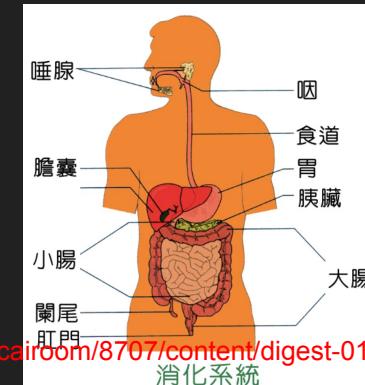
讓我們來做個練習...



道路系統

- 道路系統的 input 是地點, output 也是地點。
- input 地點 A, output 地點 B
- 如果是開車內湖上一高往南, 最近的 output 就是可以下五股。但是不可能 output 內湖。
- 這就是這個系統的機制。
- 那車子是什麼, 你可以先想成是 connection 進去道路系統後建立連線, 然後到達後斷線。

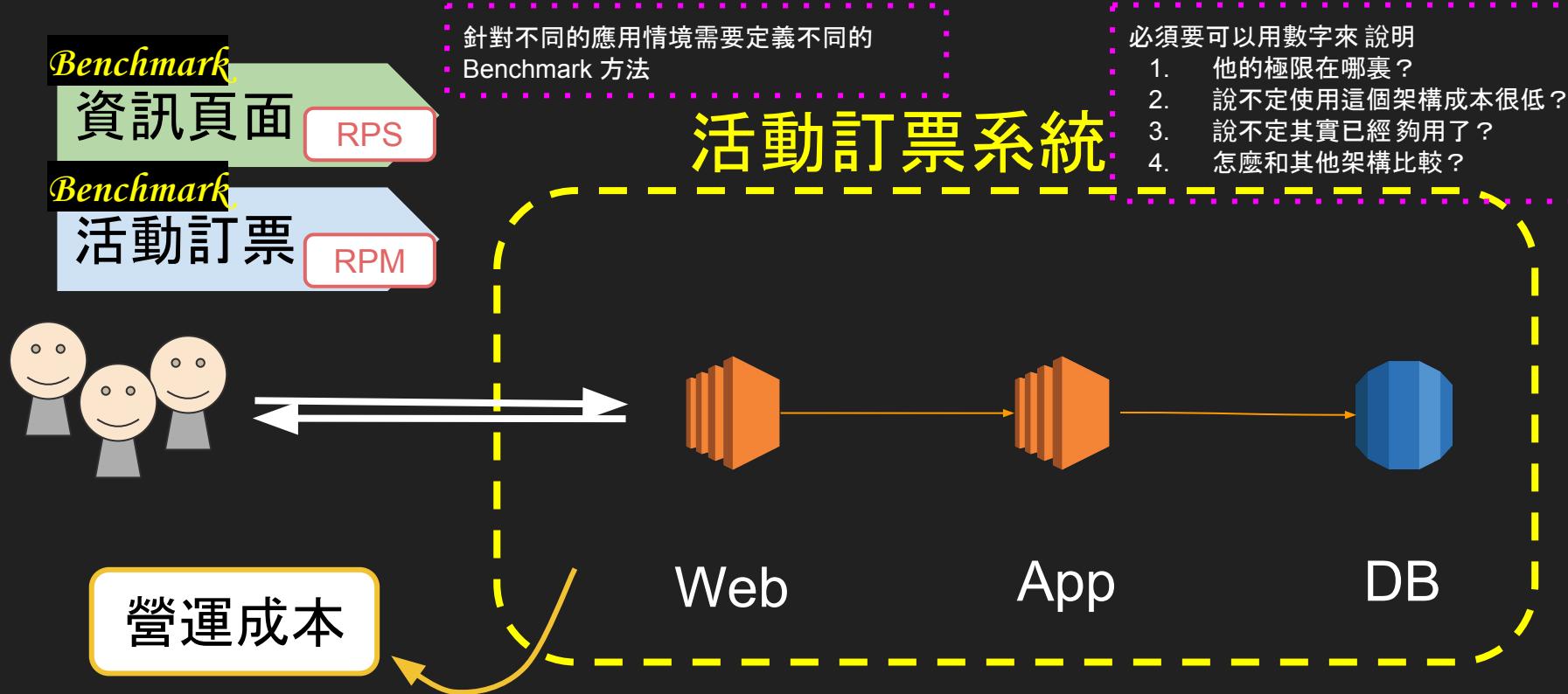
消化系統



https://market.cloud.edu.tw/content/junior/bio/tc_wc/cairroom/8707/content/digest-01.htm

不同的系統需要估量的東西也不一樣

假設我們現在有這樣一個待測對象(以 AWS 為例)



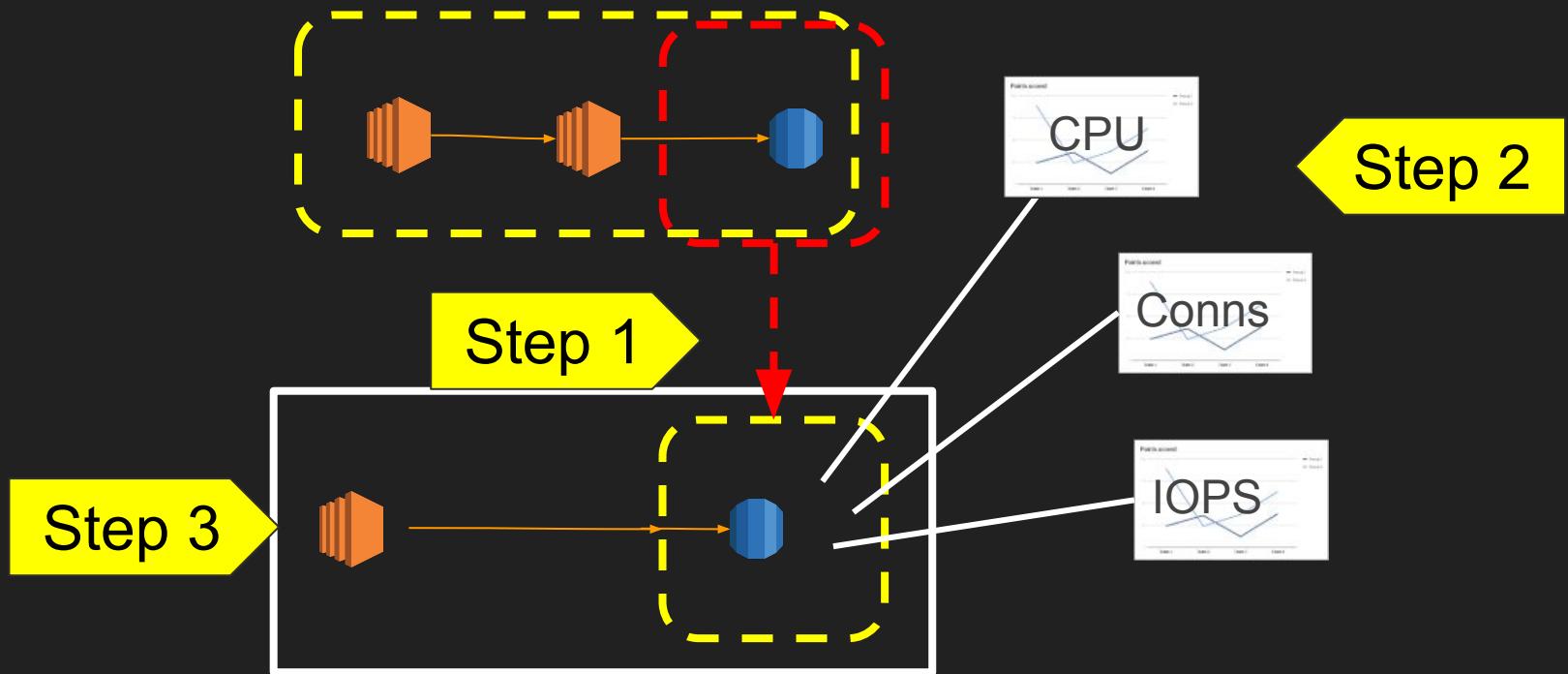
常犯的錯誤

還沒量測出 Capacity 就先著手修正可能存在的效能議題

1. 可能會做白工。現行系統可能夠用或 Capacity 比調整後更好。
2. 無法得知調整的部分到底有沒有影響了什麼。
3. 無法估量調整的成效。(無法 feedback 紿自己, 轉換成經驗)
4. 無法得知到底解了什麼效能議題。

如果別的系統發現這個效能議題, 會找不到解法。

容量測試三部曲



容量測試三部曲

Step 1

切分測試對象

Step 2

列出需要觀察的資源

Step 3

準備測試環境

切分測試對象



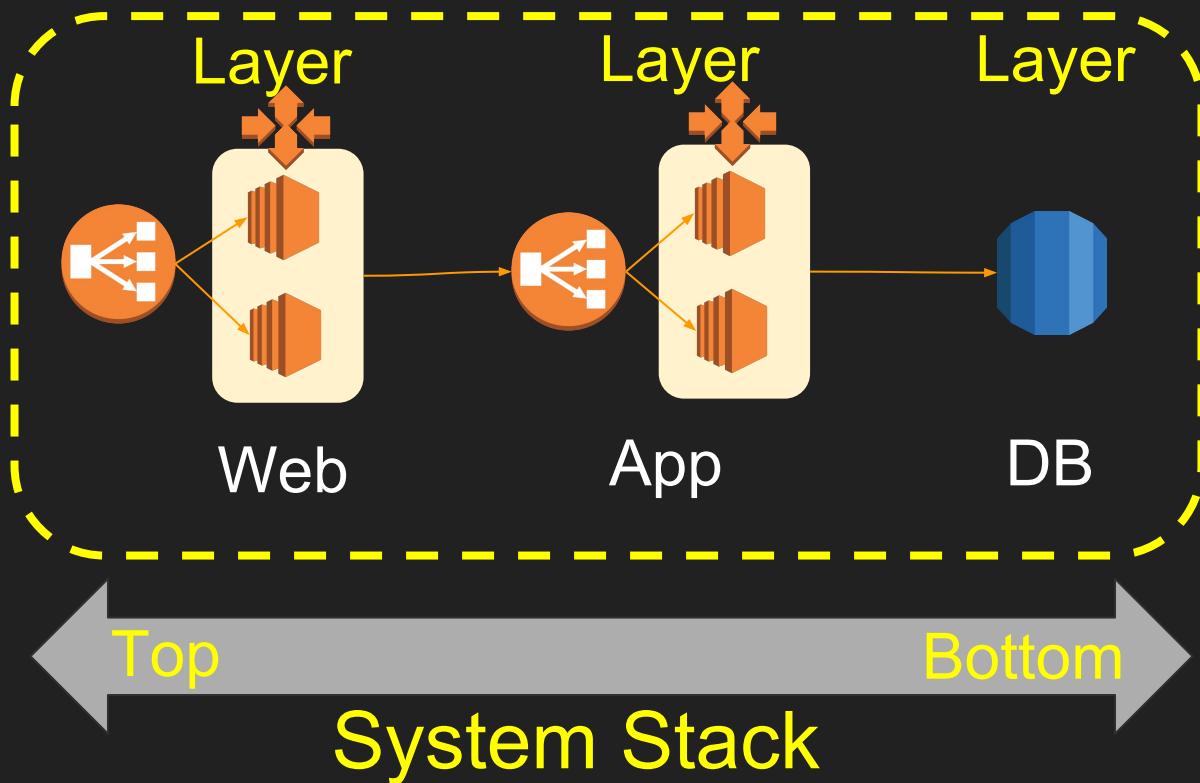
這個活動訂票系統是由三層的架構組成的系統。

- DB 負責資料的存放
- App 負責商業邏輯的控制
- Web 負責使用者介面的呈現

這樣的架構替系統未來的發展提供了一個基本的彈性。
例如 App 商業邏輯的部分可以被其他的服務共用。
儲存方式改變時也不會影響到使用者介面。

這三個角色在 AWS 上可以是一個 Layer, 可以單獨擴展。

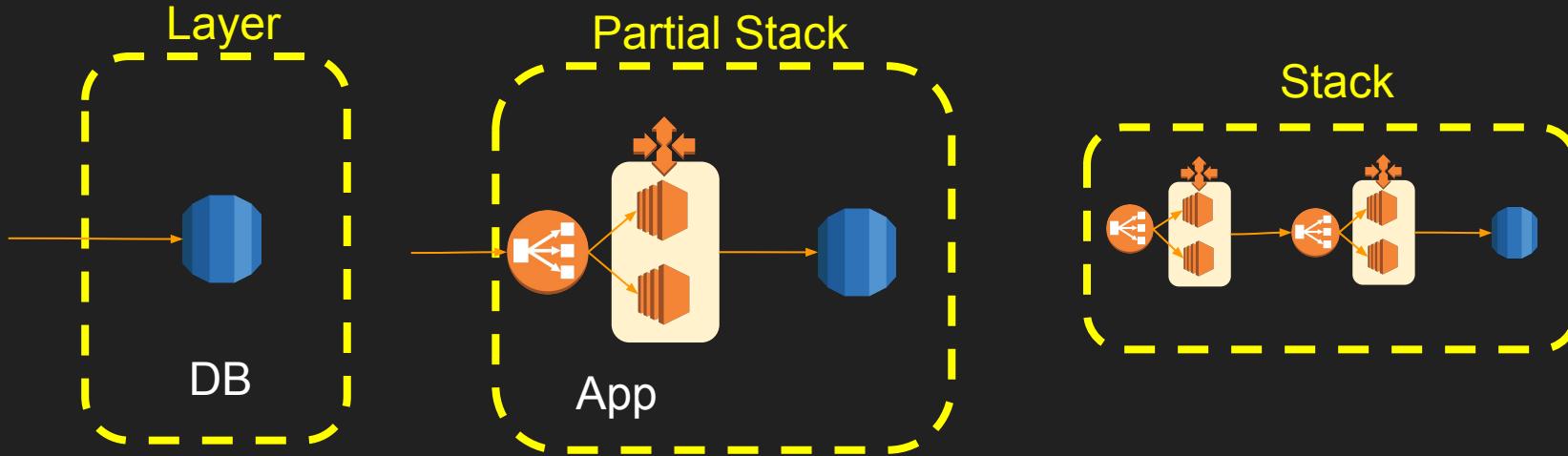
切分測試對象



為什麼要進行切分測試對象呢？

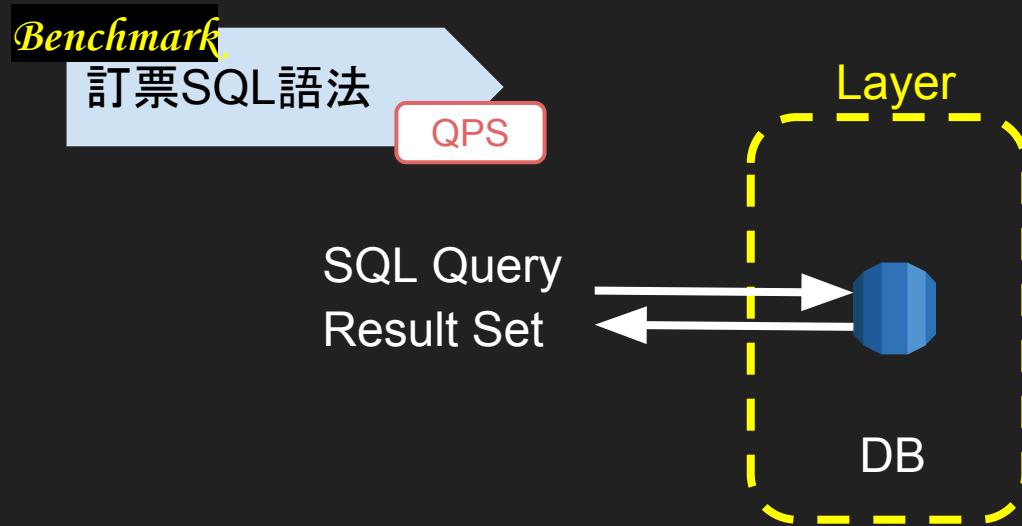
1. 在對系統力還不夠了解的狀況下，直接測試整個系統，很拿抓出效能瓶頸。遇到問題時會迷失方向，會花更多時間抓問題。最後還是要一個一個分開測試找問題。
2. 一開始無法恰當地拿捏每個 Layer 要做到什麼樣程度的配置。
3. 因此就算測出整個系統的 Capacity 是夠用的，在某一層的配置可能出現浪費。

測試順序



最底層的 layer 能力不足，前面的 capacity 在怎麼好也沒用，更不用測整個 system stack 了

每個 layer 也是獨立的測試對象



- 每個 layer 也是獨立的測試對象, 別忘了先定義 Benchmark
- 針對 DB Layer 的定義, 這時候 input 就是 SQL Query
- Benchmark 方法就是 活動訂票會使用到的 SQL 語法
- 去 Query 測試, 看看最大的 QPS 可以到多少。

容量測試三部曲

Step 1

切分測試對象

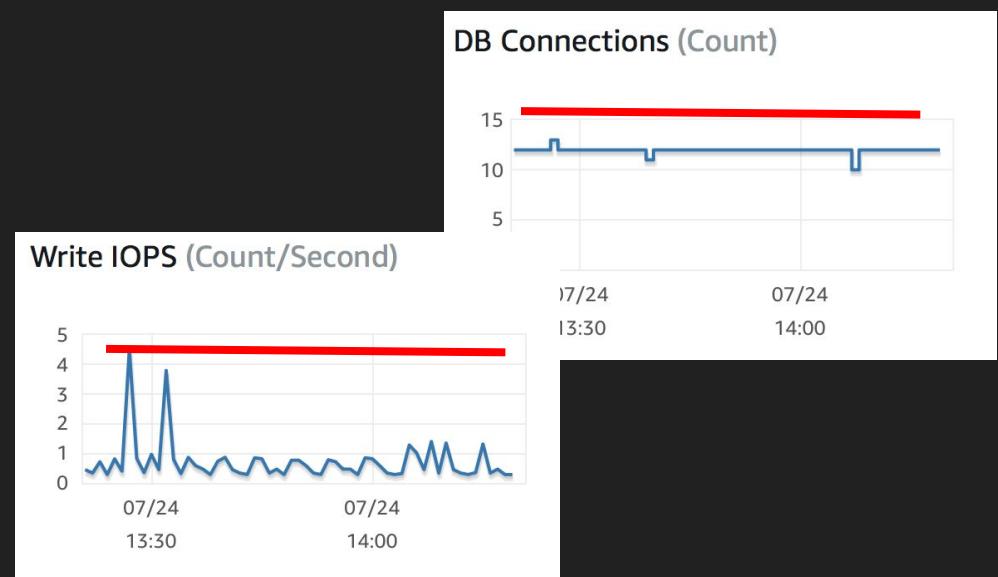
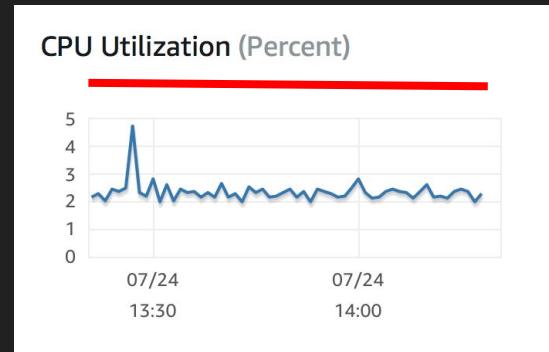
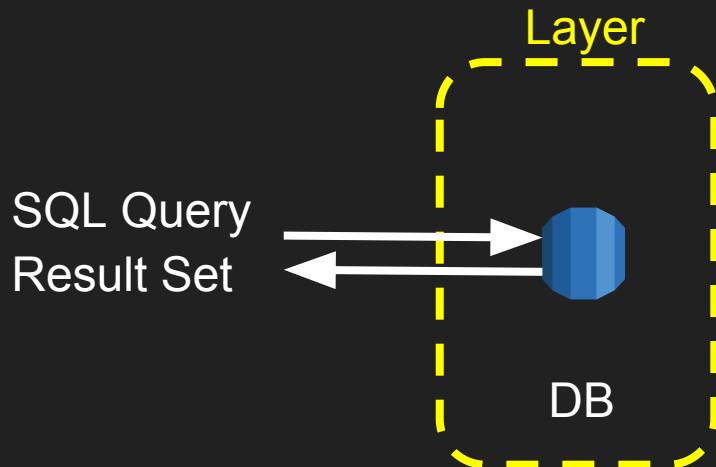
Step 2

列出需要觀察的資源

Step 3

準備測試環境

列出需要觀察的資源

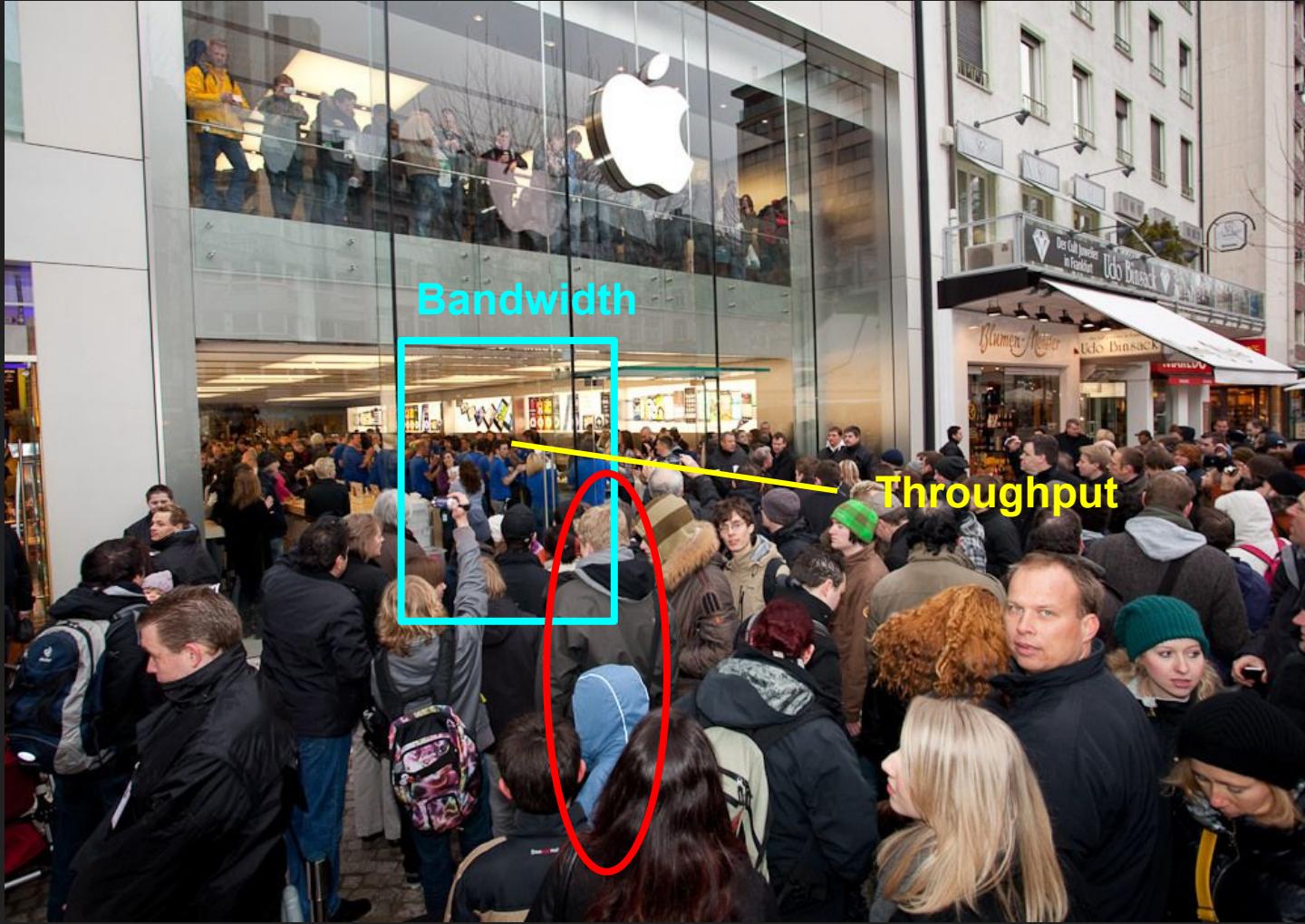


最重要的是。

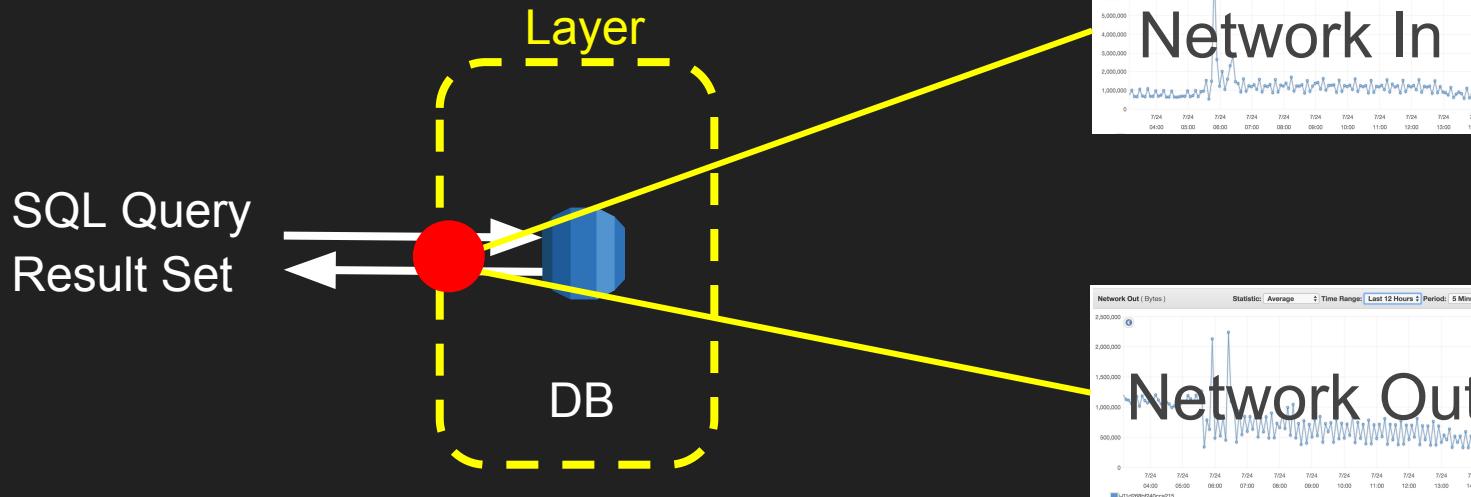
了解每個數據的當前的限制 Limitation。以這個作為是否有效利用資源的依據。

舉例來說，當所有收集的資源數據都沒有達到 Limitation，卻又遇到測試瓶頸時（測不上去）。就可以知道一定是漏掉了某個關鍵數據。

這時候怎麼調整都是沒有意義的，因為你在與看不見的敵人戰鬥，需要先讓他現形。

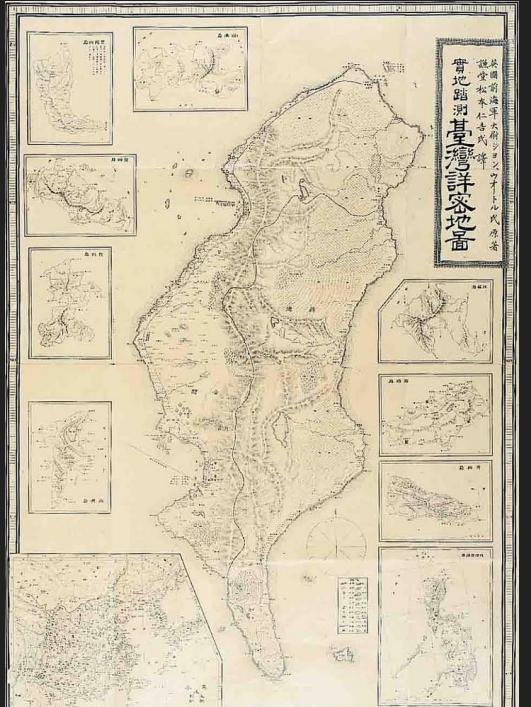


容易被忽略的部分 Network Bandwidth



『地圖不等於疆域』

Alfred Korzybski



圖片來源：

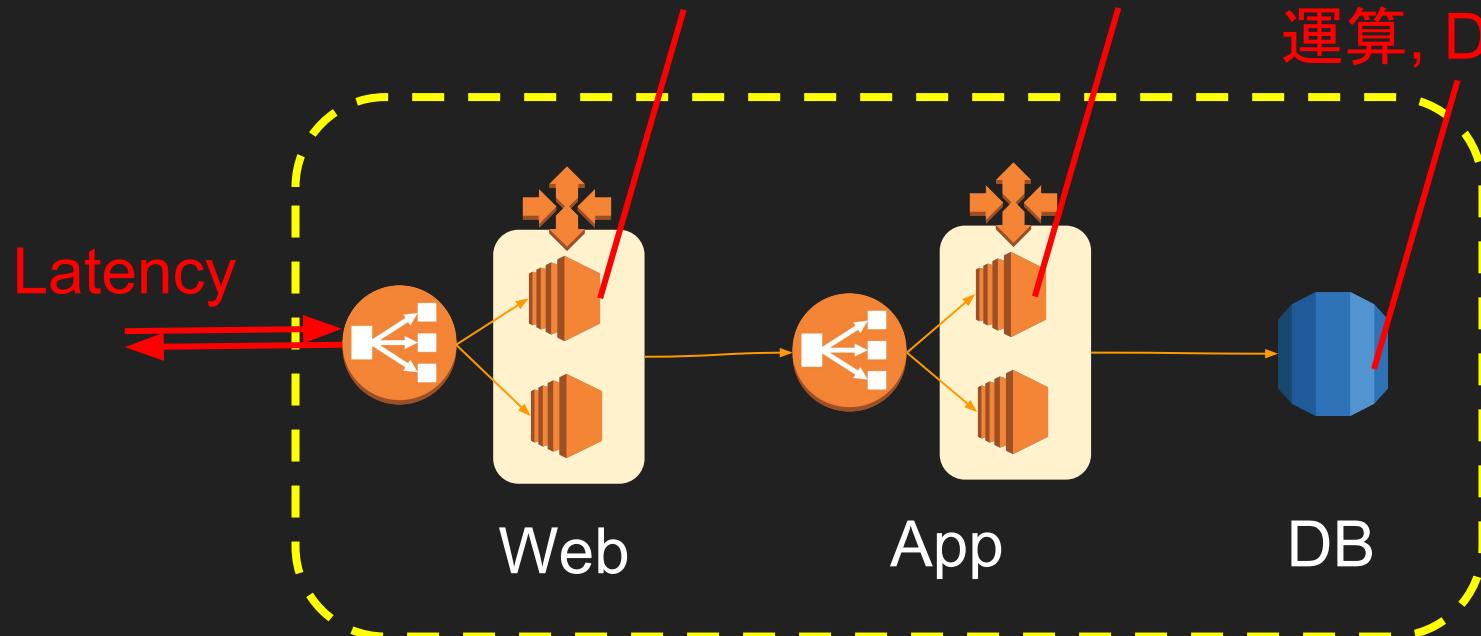
https://en.wikipedia.org/wiki/Alfred_Korzybski

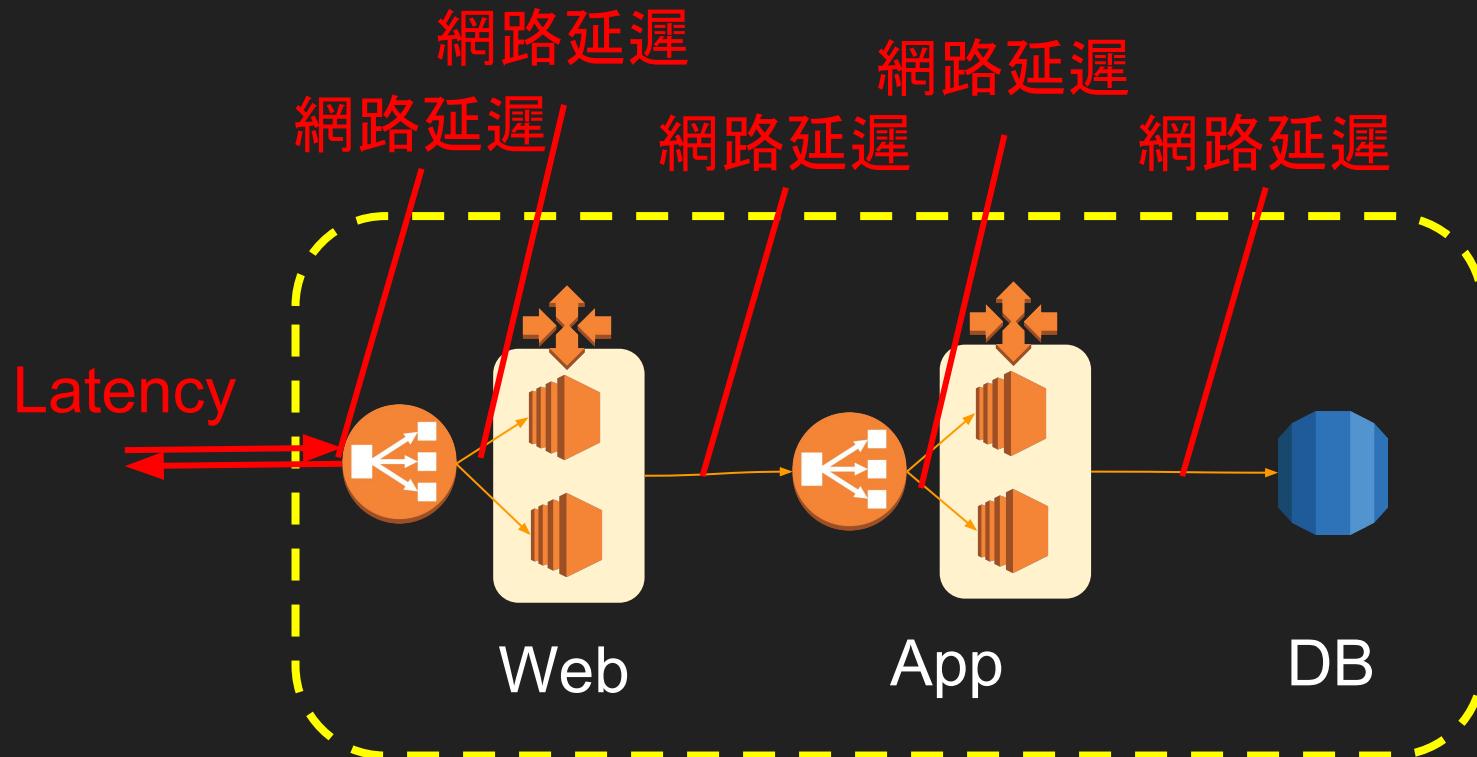
http://140.128.219.245/home/local/www/theme_1.html

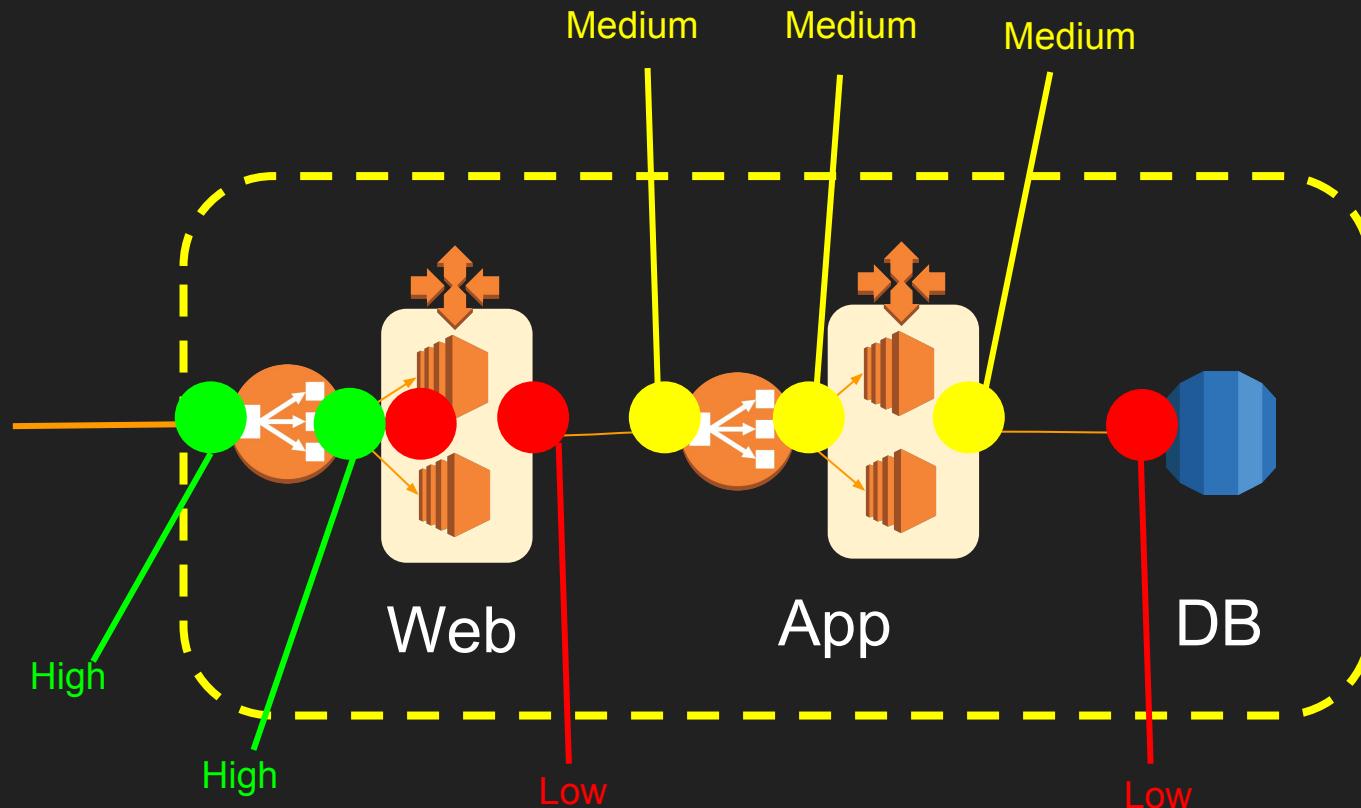
<https://sites.google.com/a/cts.edu.my/jiao-yu-zhi-yuan-fen-xiang/hemengao-ji-li-chi/3-1880nian-de-tai-wan-de-1>

運算, Disk I/O 運算, Disk I/O

運算, Disk I/O







容量測試三部曲

Step 1

切分測試對象

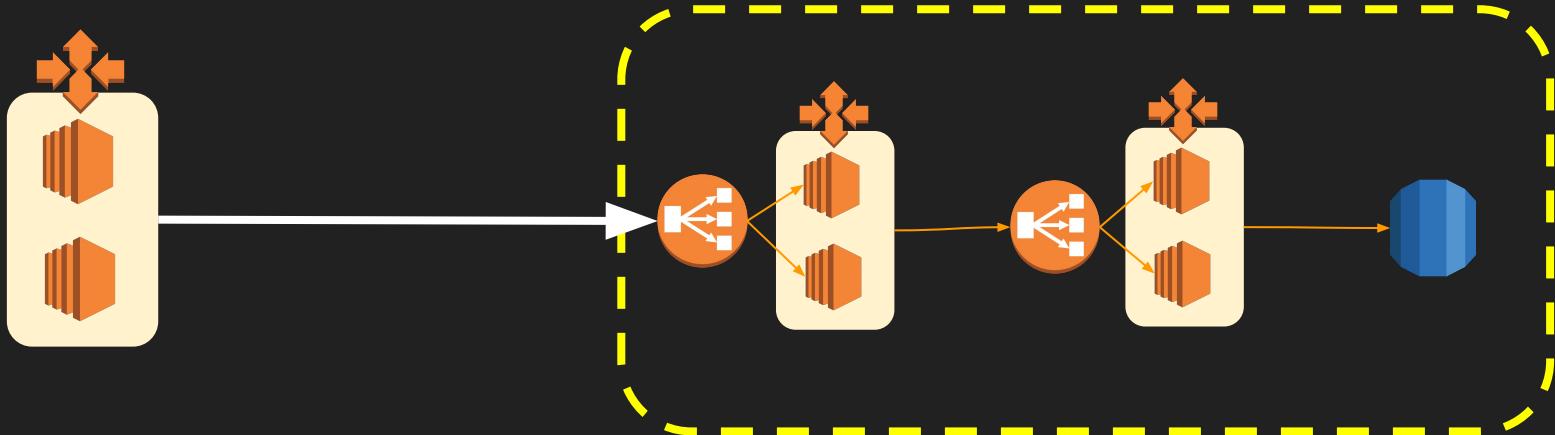
Step 2

列出需要觀察的資源

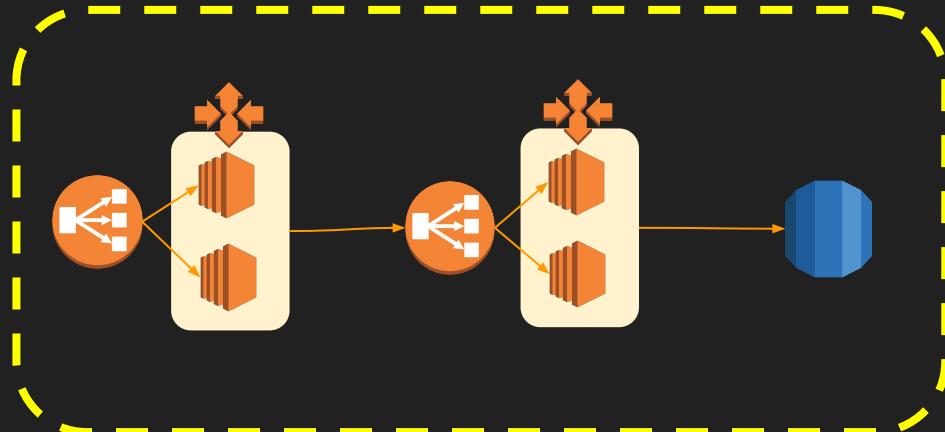
Step 3

準備測試環境

建立測試環境



準備待測體



所有的 Config

機器等級

網路配置

Application 版本

與 Production 一致

如果平常沒有做 Infra as code 和部署流水線會很痛苦

常犯的錯誤

手動建立起環境後就開始測試

先著手在如何快速建立可以執行測試環境

AMI + Launchconfig + Cloudformation

在開發的版本上進行測試

理想是大版號都要進 Capacity Test

1. 通常需要測試大流量的對象，需要的使用的實體配置會比較好，就會比較花錢，無法一直放在那邊給你測試。
2. 就好是可以需要測試時，建立起來，不需要時，整個砍掉。
3. 使用 AMI 快速 snapshot prodution 的機器，配合 Cloudformation 將整個 Service Stack 建起來。Launchconfig 可以用來修改配置。

釐清使用者行為，挑選測試工具



我推薦使用 JMeter, 原因：

1. 內建許多 Protocol 可以用, HTTP , HTTPS, SOAP, REST API, ODBC, JDBC, TCP, 或是 shell script。
2. 可以完整的模擬使用者行為，尤其是針對 Web 或 REST API
3. Client 端 Log, Report 完整，自動生成 HTML
4. 可以併發執行



- ▼ Test Plan
- └ Thread Group
 - IDEAS Connector
 - IDEAS Request**
 - View Results in Table
- WorkBench

IDEAS Request

Name: IDEAS Request

Comments:

Action

Connection ID \${user} Send Receive

Sending Command

Header

TO 1

Payload

Name:

5

Value

FROM 1000091

2.0

COS 1

CMD CONNECT

GROUPNAME AUTH

Detail

Add

Add from Clipboard

Delete

Up

Down

Receiving Command

Timeout: 5 seconds

Command	Groupname
CONNECT_ACK	AUTH

Detail

Add

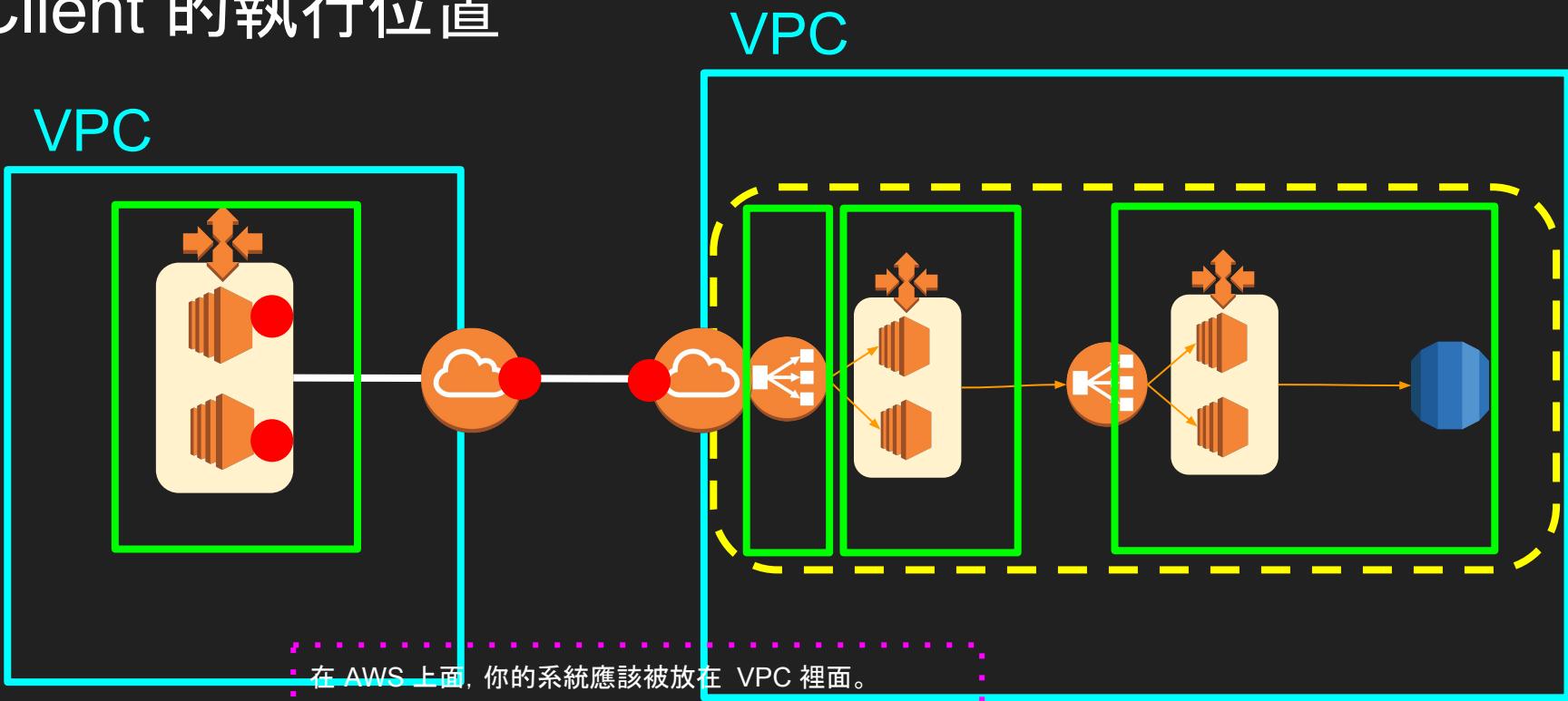
Add from Clipboard

Delete

Up

Down

Client 的執行位置



所以 Client 應該是放在 VPC 外面。

可以放在一個獨立的 VPC 裡。

測試資料的準備

1. 使用 production log 來收集使用者行為
2. 將使用者行為格式話成 csv 檔
3. 準備對應的 script 讓資料可以倒入資料儲存空間

測試結果數據

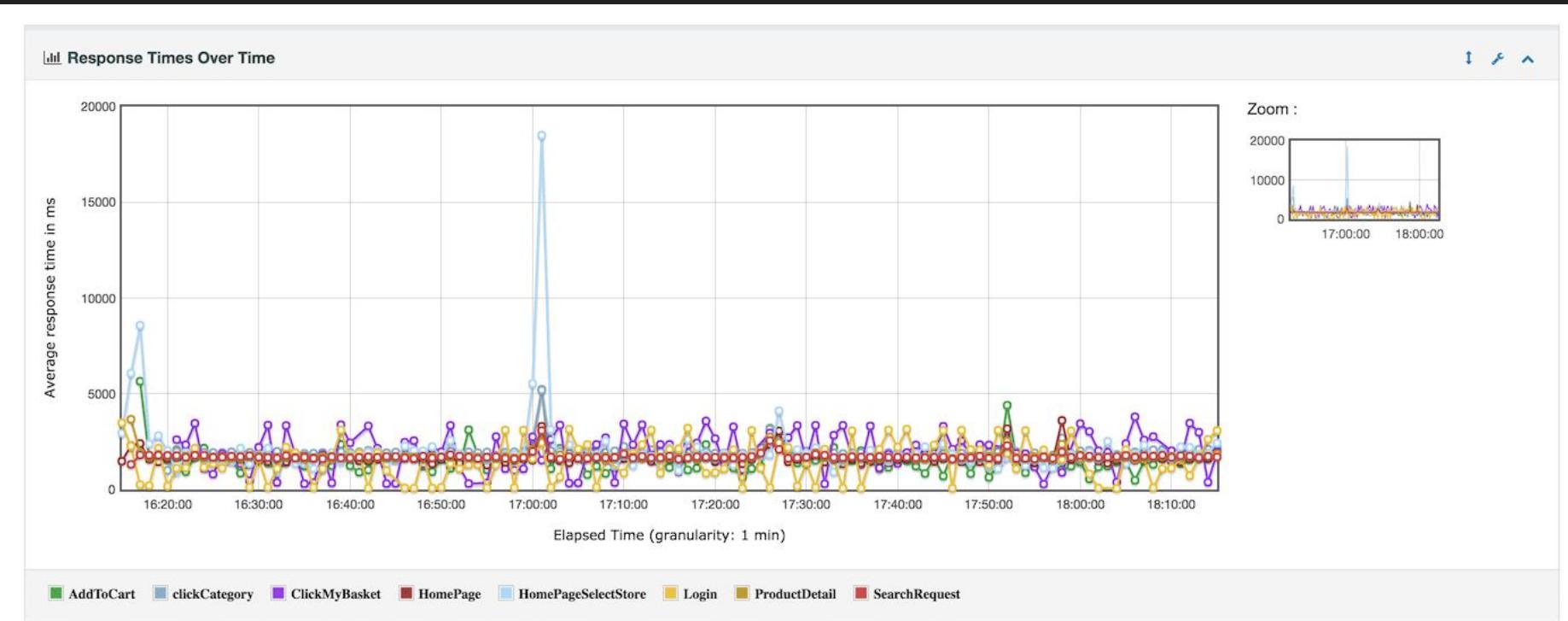
- Response Time (Latency) Overtime
- Successful Rate % (Error Rate)
- Network Throughput (Client side)
- RPS

雖然說 AWS 的 ELB 也有提供 Request Count 和 Avg latency 的數據。

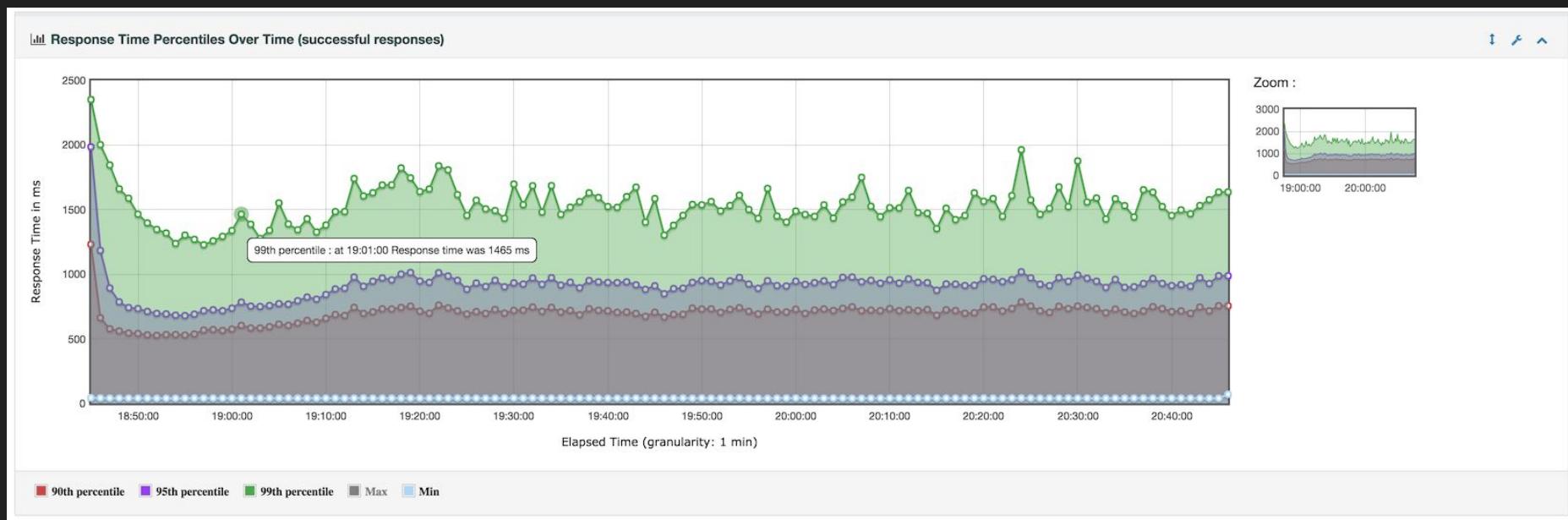
但是我們要的 Capacity 測試結果必需由 Client 端收集，原因如下：

1. Client 端得到的結果才是真正使用者看到的
2. AWS ELB 的數據少了 Client 到 ELB 這段的網路延遲數據，
3. 如果 Client 發生 504 Gateway Timeout, AWS ELB 得不到這個數據。

Response Time overtime (AVG Latency) 平均值



Response Time overtime (Percentiles)



- 有少部分的請求會 latency 拉到非常高，可能因為網路連線的關係
- 這些樣本會影響整體的平均值(實際的使用者感受其實可能會比平均直來得快)
- 導致從平均值看不出多數 request 的回應時間
- 所以我們可能需要從 90位, 95位, 99位來看，去長尾的意思

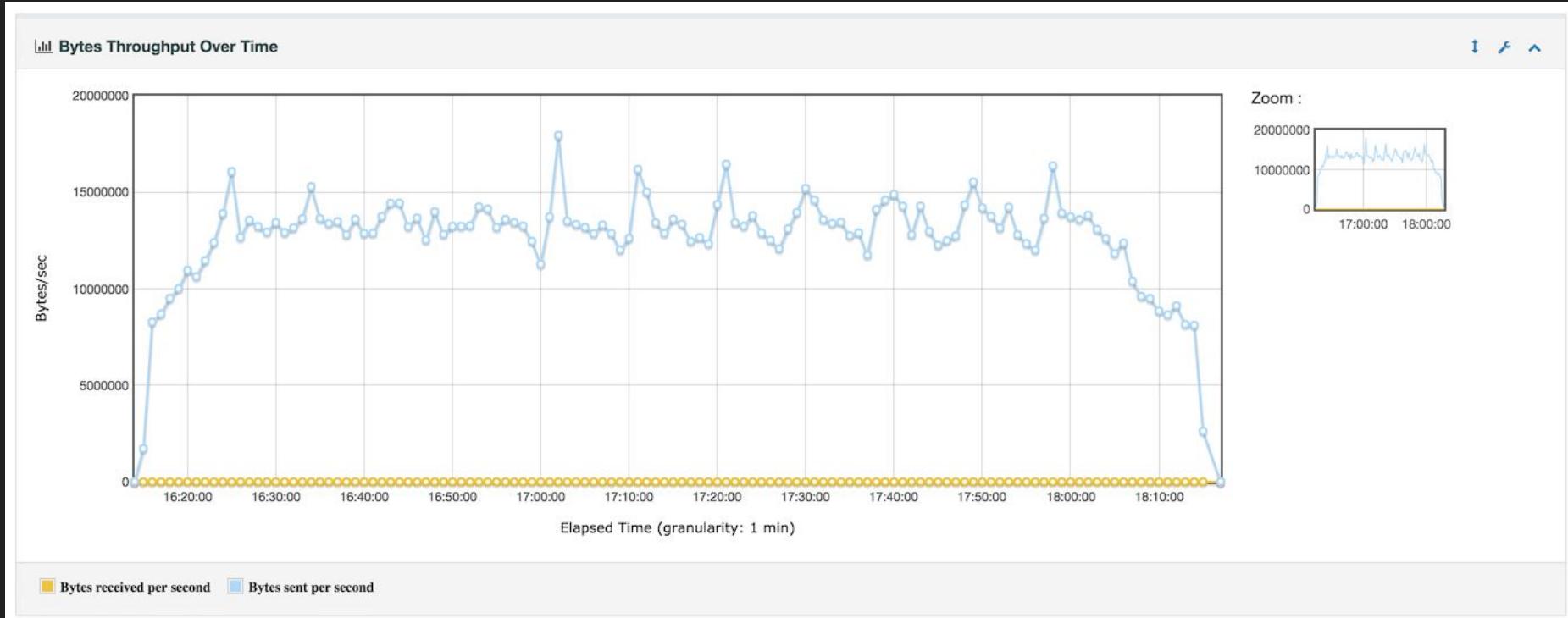
Successful Rate

Requests Summary



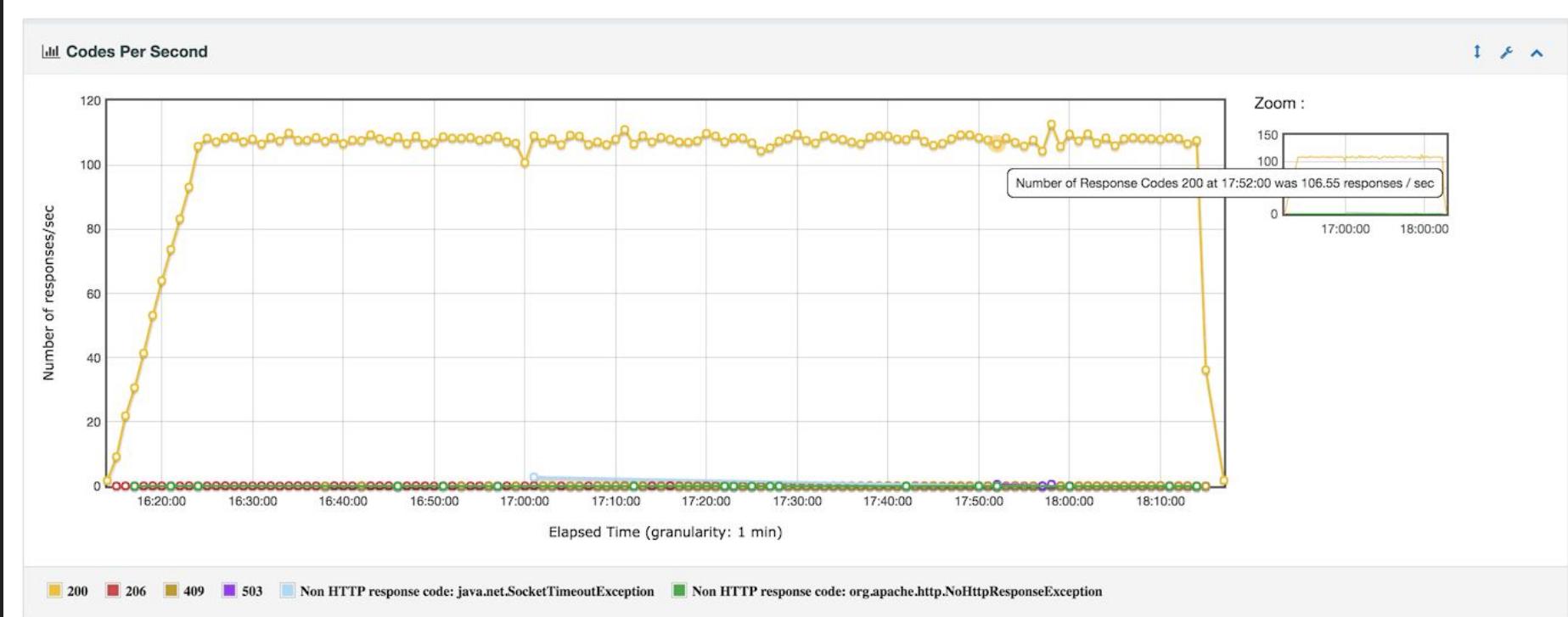
Network Throughput (Client Side)

- Client 端的 Network Throughput, 非常重要,
- 如果增加了 Request 數量 Throughput 沒有增加,
代表可能是 Client 的頻寬不夠來不及把增加的 request 傳送出去



RPS

RPS 因為基礎單位是秒，所以 Latency 超過 1 秒就不會算進去。



Capacity Plan

資訊頁面

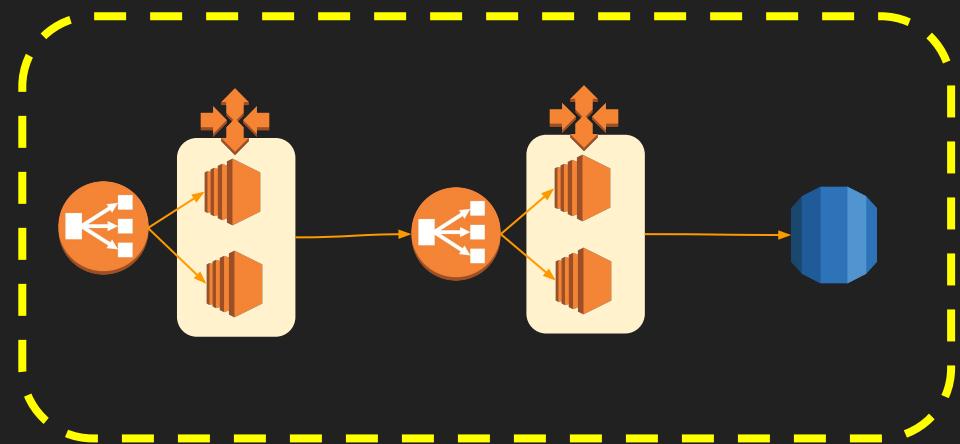
100 RPS

1000 RPS

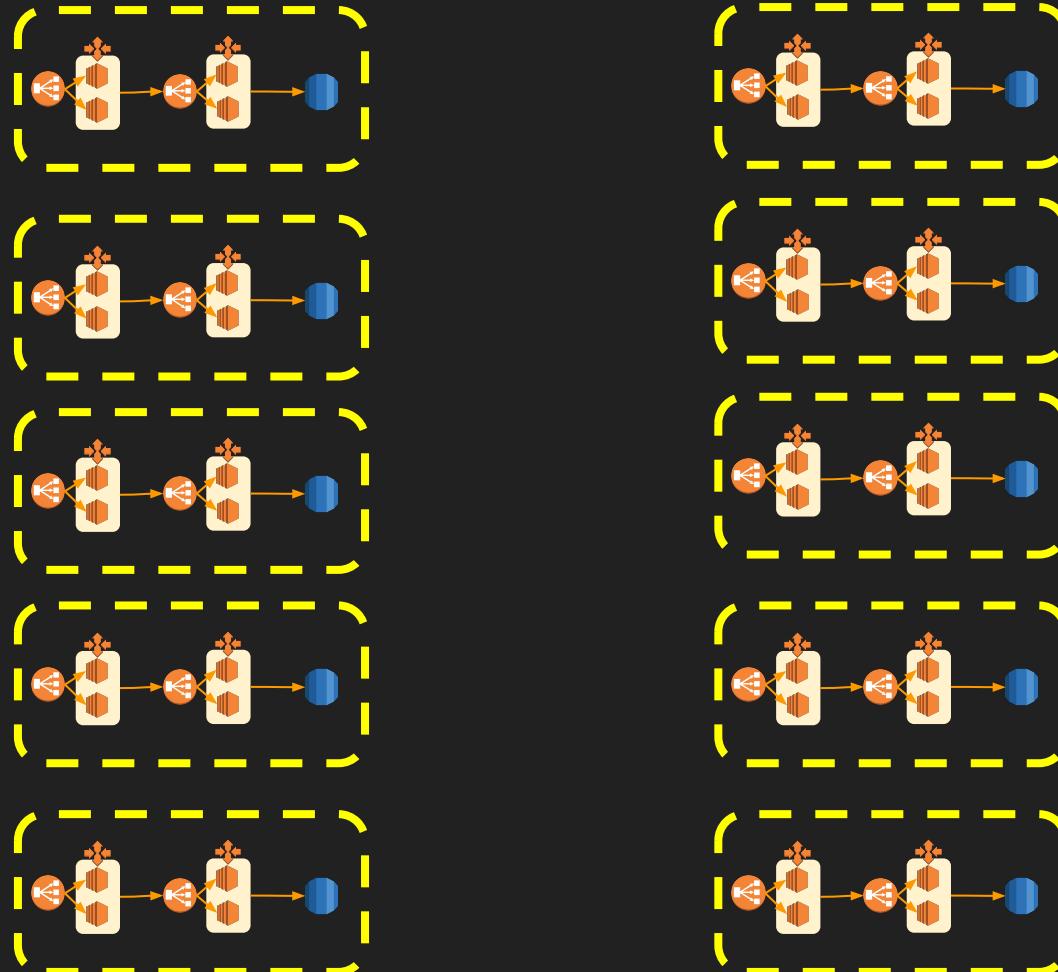
活動訂票

100 RPM

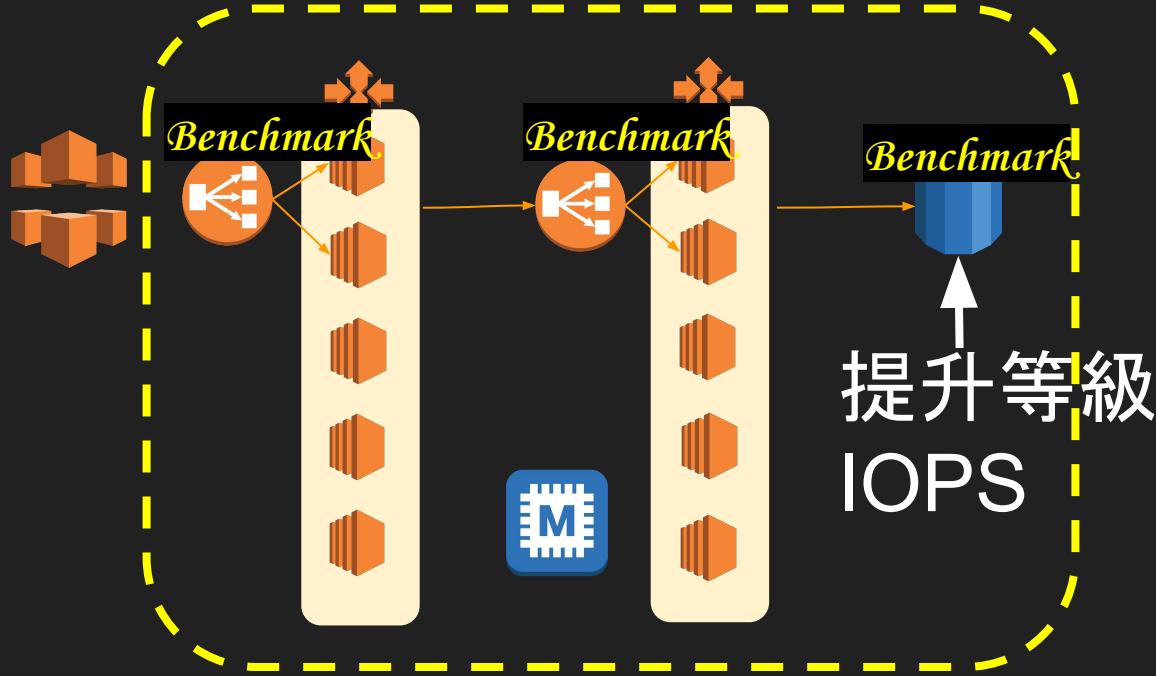
1000 RPM



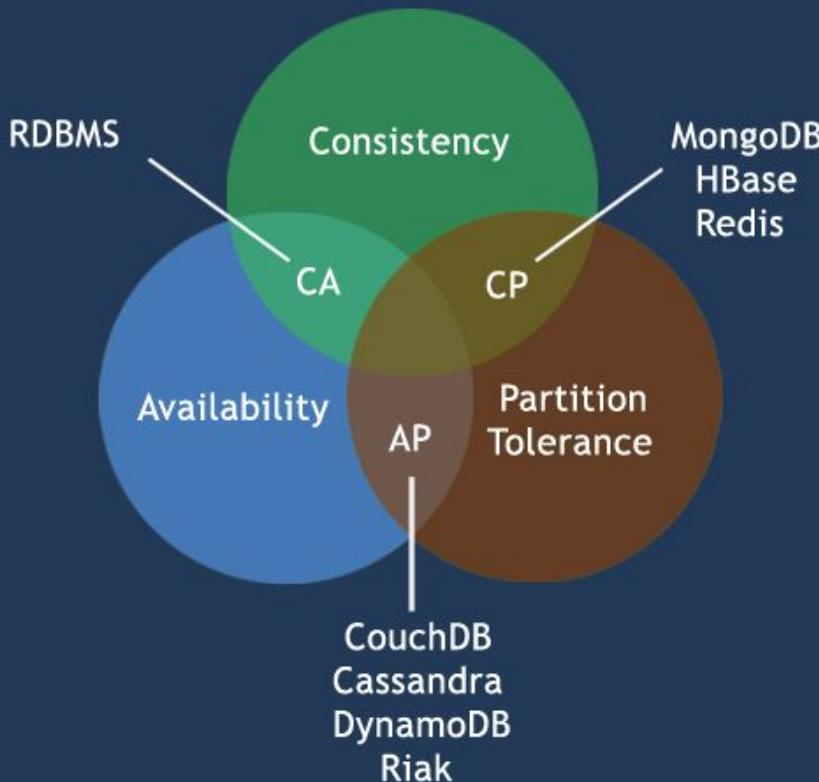
XN
分伺服器
資料獨立



內部擴展



CAP Theorem





林
3小時 · 人

...

看來想花錢也得先抽號碼牌～ Orz

504 Gateway Time-out

1. 這是什麼情況造成的？

2. 當下該網站的監控系統會看到什麼？



和其他 8 人

4則留言

讚

留言

這種事，是一定要做一下的

```
➔ / nslookup donate.teamkp.tw
Server:          10.11.1.5
Address:         10.11.1.5#53

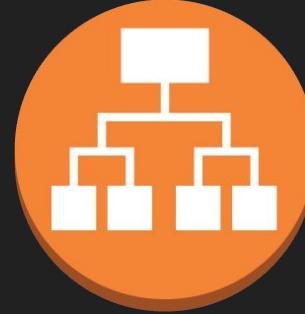
Non-authoritative answer:
donate.teamkp.tw      canonical name = kpdonate-alb-1039184351.ap-northeast-1.elb.amazonaws.com.
Name:   kpdonate-alb-1039184351.ap-northeast-1.elb.amazonaws.com
Address: 54.64.8.121
Name:   kpdonate-alb-1039184351.ap-northeast-1.elb.amazonaws.com
Address: 54.92.115.244
Name:   kpdonate-alb-1039184351.ap-northeast-1.elb.amazonaws.com
Address: 54.249.83.28
Name:   kpdonate-alb-1039184351.ap-northeast-1.elb.amazonaws.com
Address: 13.112.195.202
```

AWS 的 ELB 目前有三種



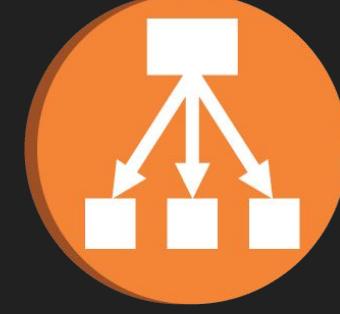
**Classic Load
Balancer**

HTTP, HTTPS
TCP



**Application
Load Balancer**

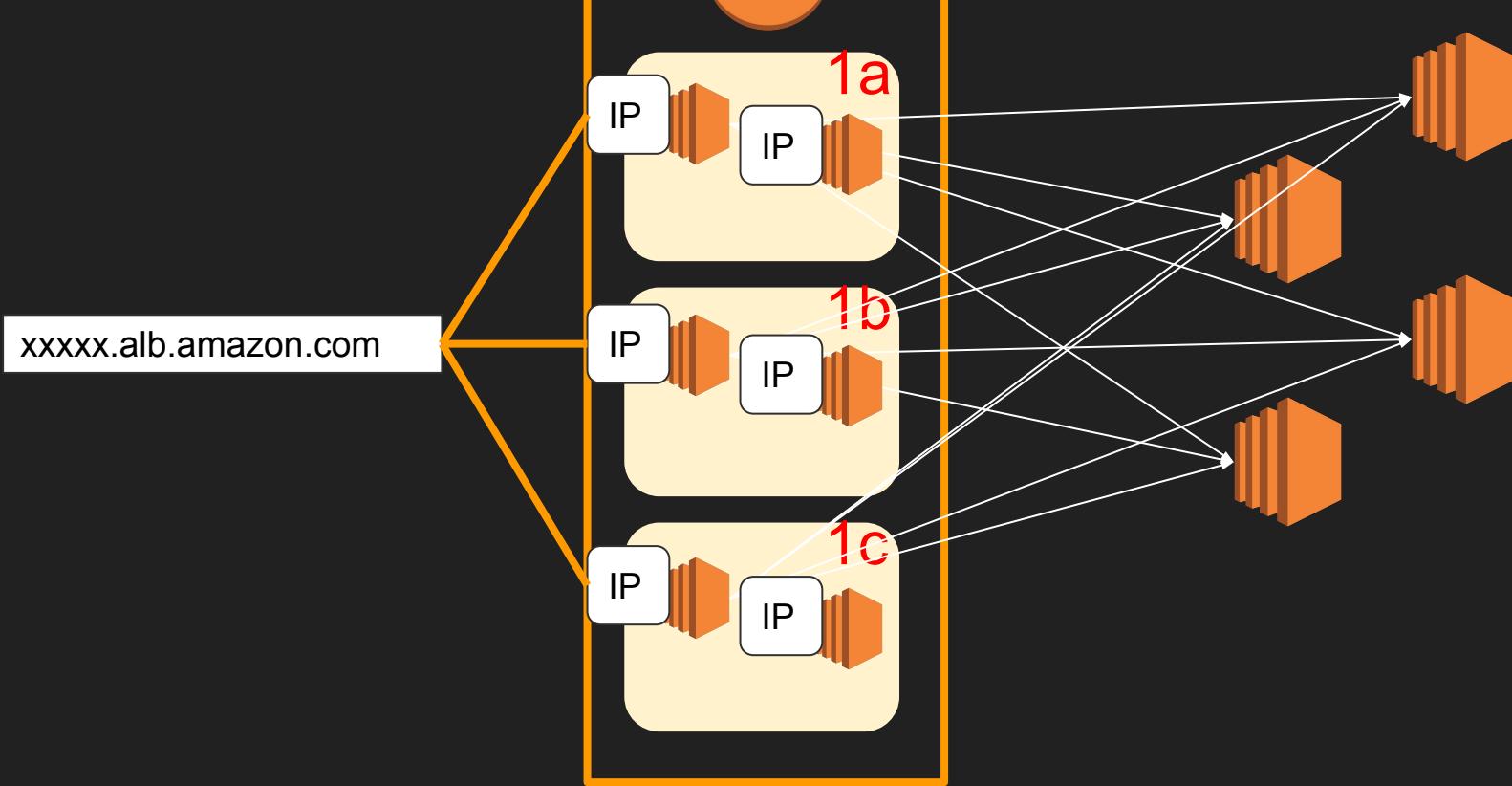
HTTP, HTTPS



**Network Load
Balancer**

TCP

CLB & ALB



Create Network Interface						
	Name	Network interface ID	Subnet ID	VPC ID	Zone	Actions
	eni-64233568	subnet-61a9f238	vpc-57660932	ap-northeast...	default, Private-De...	ELB TMP-PT-Back
	eni-8ae834b7	subnet-32521c...	vpc-57660932	ap-northeast...	default, Private-De...	ELB TMP-PT-Back
	eni-b9daccb5	subnet-61a9f238	vpc-57660932	ap-northeast...	default, Private-De...	ELB TMP-PT-Back
	eni-e316cade	subnet-32521c...	vpc-57660932	ap-northeast...	default, Private-De...	ELB TMP-PT-Back
	eni-1419c529	subnet-a85b15df	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB TMP-PT-Front
	eni-252e3829	subnet-68a0fb31	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB TMP-PT-Front

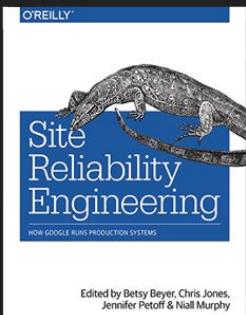
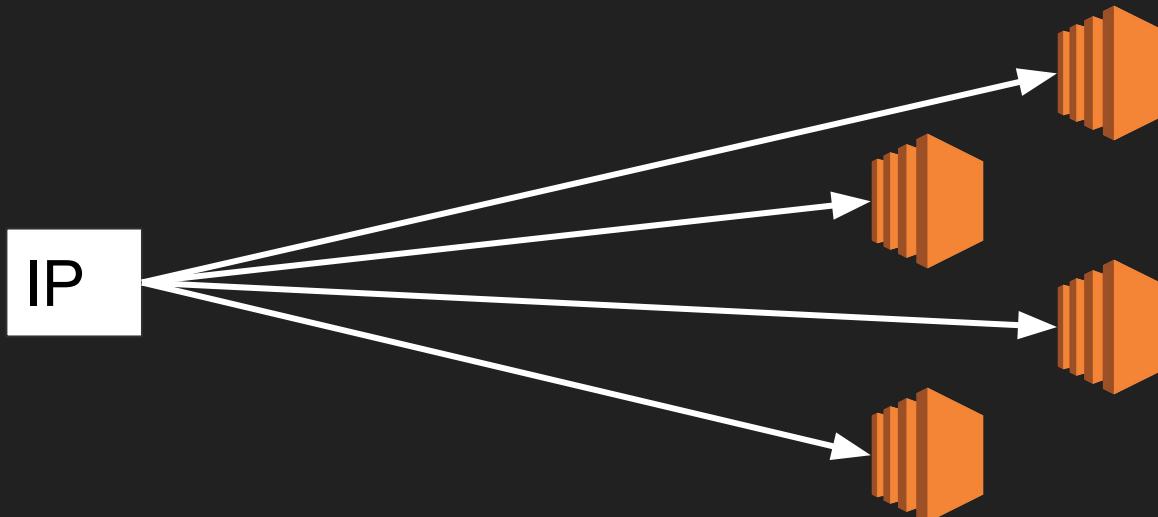
	Name	Network interface II	Subnet ID	VPC ID	Zone	Security groups	Description
	eni-0dc31f30	subnet-32521c...	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-16c4182b	subnet-a85b15df	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-1adf0327	subnet-a85b15df	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-2706102b	subnet-61a9f238	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-2715032b	subnet-61a9f238	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-3303153f	subnet-68a0fb31	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-38ac7005	subnet-32521c...	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-46d60a7b	subnet-32521c...	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-5fd50962	subnet-a85b15df	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-6f0e1863	subnet-61a9f238	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-7233257e	subnet-61a9f238	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-7c0d1b70	subnet-61a9f238	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-a1c9159c	subnet-32521c...	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-b41600b8	subnet-68a0fb31	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-b93721b5	subnet-68a0fb31	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-d3b06cee	subnet-a85b15df	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-d4df03e9	subnet-32521c...	vpc-57660932	ap-northeast...	Private-Default-EC2	ELB LetSSL-1-0-PT-201709	
	eni-df0d1bd3	subnet-68a0fb31	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	
	eni-ef0610e3	subnet-68a0fb31	vpc-57660932	ap-northeast...	Protected-RP-PT-E...	ELB LetSSL-1-0-PT-201709	

```

east-1.elb.amazonaws.com. 26 IN A 52.199.215.80
east-1.elb.amazonaws.com. 26 IN A 52.193.205.230
east-1.elb.amazonaws.com. 26 IN A 54.65.249.80
east-1.elb.amazonaws.com. 26 IN A 52.198.76.161
east-1.elb.amazonaws.com. 26 IN A 13.113.151.19
east-1.elb.amazonaws.com. 26 IN A 13.114.116.52

```

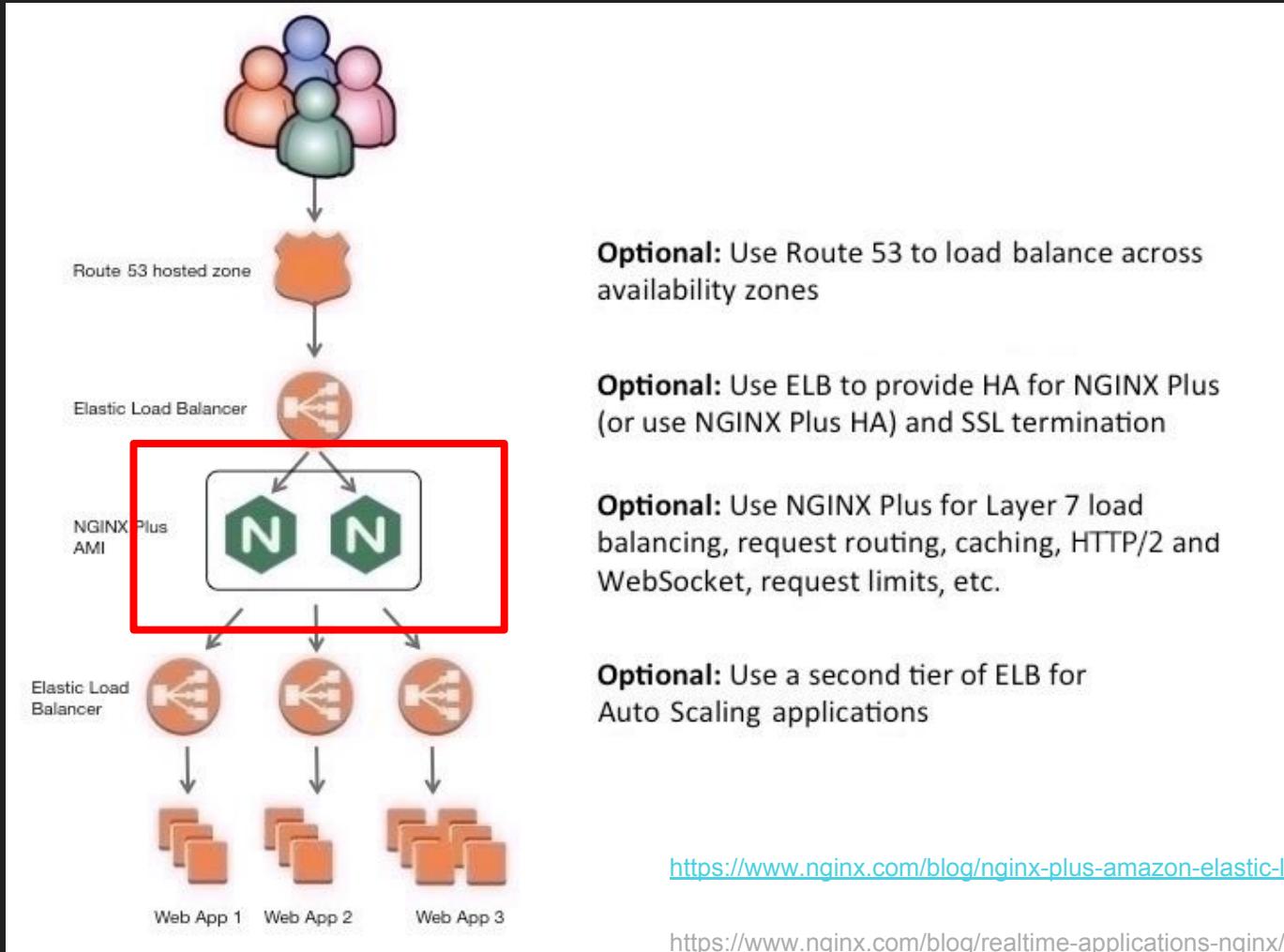
NLB



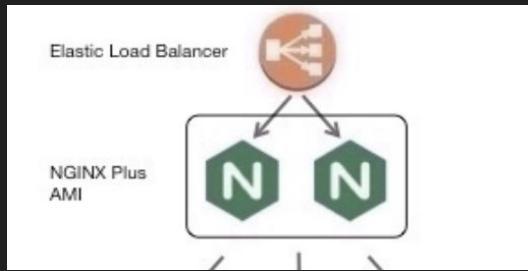
延伸閱讀：

Chapter 19 - Load Balancing at the Frontend

案例分享



切分測試對象



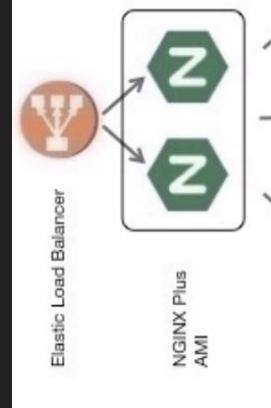
收集系統數據

別忘了還有
ELB 的數據

Benchmark

頁面加密

RPS

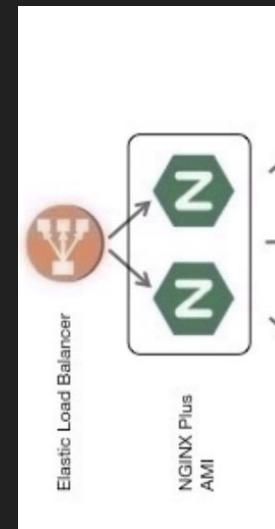


測試環境

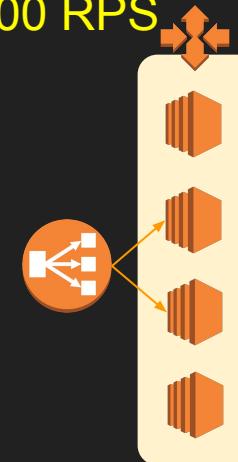
Cloudformation



Cloudformation



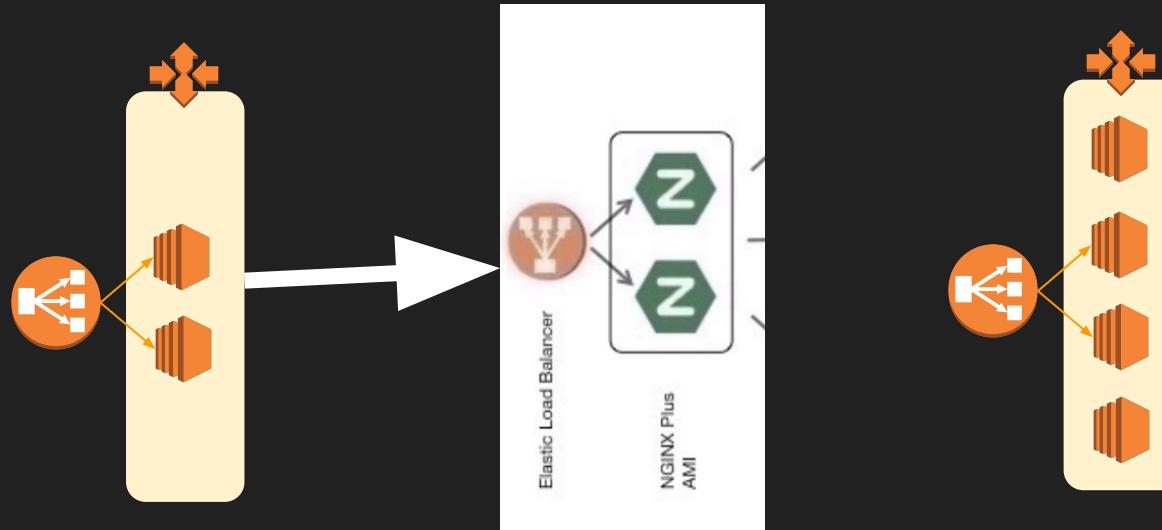
2000 RPS



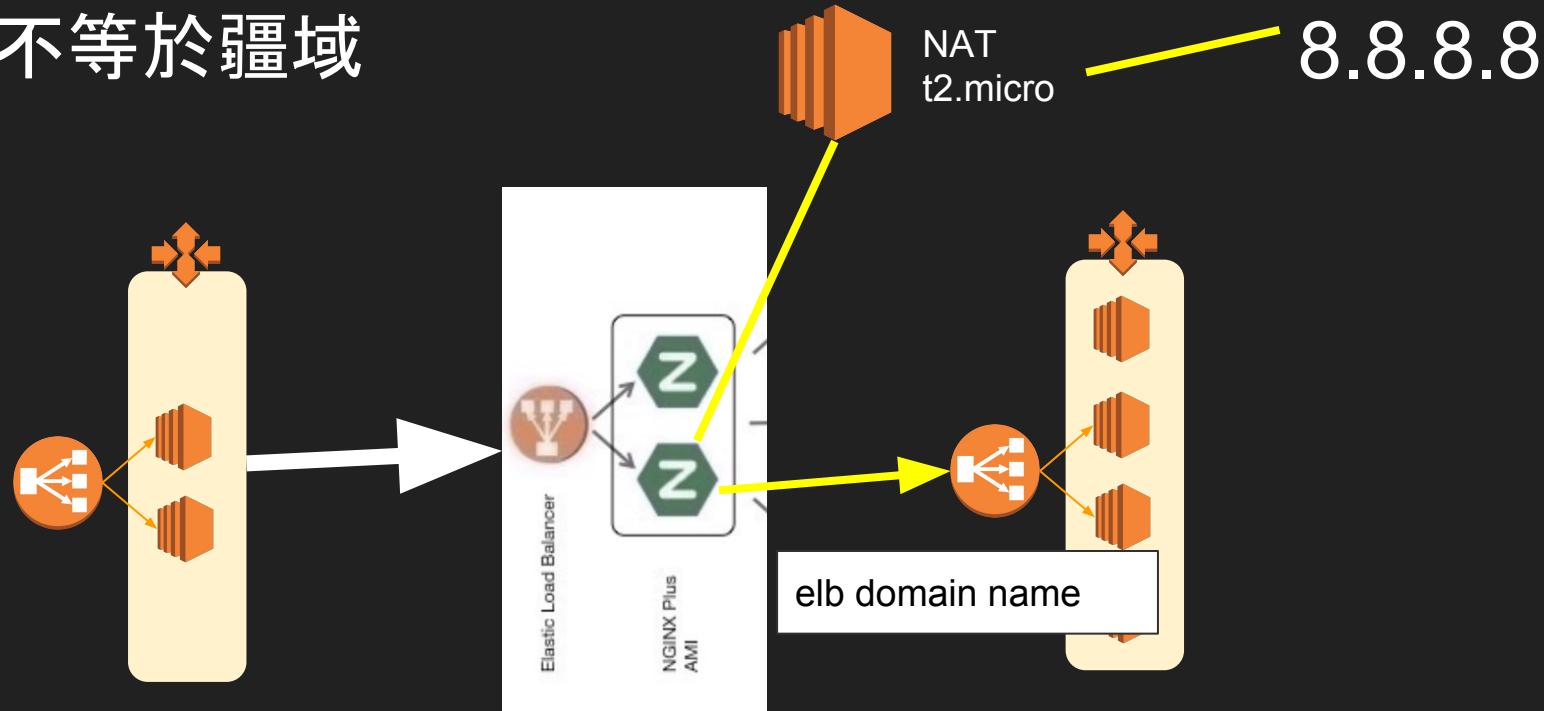
100K

Fake App

開始測試



地圖不等於疆域



總結

定義好系統和 Benchmark 方法是第一要務

想要輕鬆，先搞定用 Infra as code 建立測試環境

『地圖不等於疆域』

別忘了 Network Bandwidth

Thank You



□ 課程F2：Hop & Pop
低空 + 第一次 Solo jump