Week 9 & 10 (Chapter 9 Architectural Design, Chapter 10 Architecture Design Styles)

> [SE6005] Software Engineering
>
> Year: 2024 Spring
> Lecturer: 鄭永斌 (YPC), 梁德容 (DRL), 莊永裕 (YYZ) 業界師資 (EL) 等教授
> Department of Atmospheric Sciences
> Student: 林群賀
> Student ID: 109601003

在這週的作業中，我選擇了第一個討論題目，命題為：「請設計一個能夠負荷短時間、大流量的搶票網站系統架構。」來進行以下討論。

最近新竹工程師開源搶票程式鬧得沸沸揚揚，加上之前常常聽到拓元是 AWS 的客戶，因此這篇的討論決定以這篇文章：「【直擊Modern Web 2015】解決20萬人秒殺搶票系統架構大揭露」作為參考。

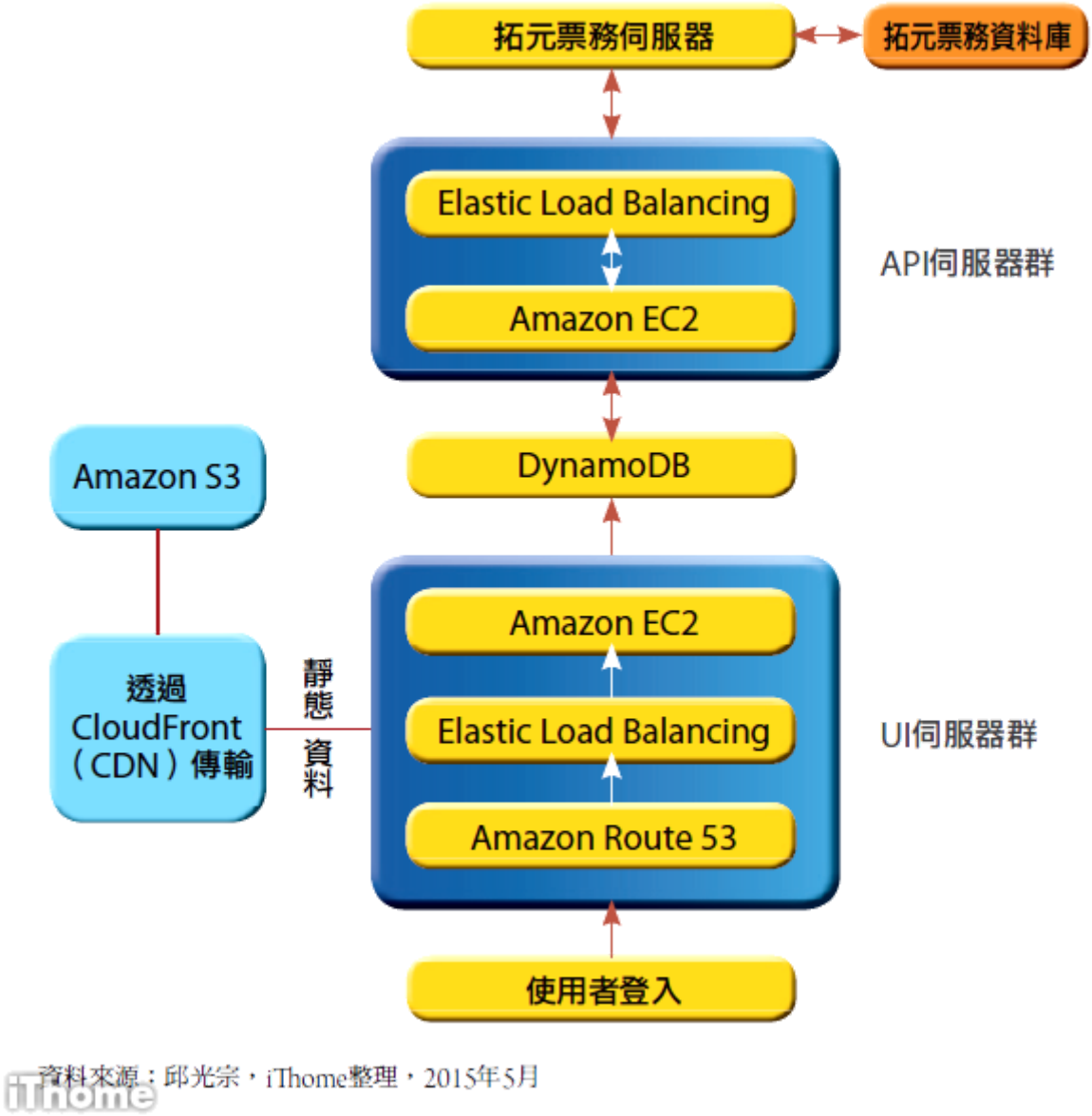
在每次有大型活動或是歌手的時候，總是會有大量的人搶票，這時候就需要一個能夠負荷短時間、大流量的搶票網站系統架構。這篇文章中提到了拓元科技的架構，他們使用了 AWS 的服務，並且使用了 ELB、EC2、RDS、S3、CloudFront、Route 53 等服務，來達到高可用性、高擴展性、高效能的目標。

以下為原文：

在面對20萬人上線搶12萬張票的同時，拓元至少得讓20萬人上得了售票網站，不僅要短時間內銷售完畢，也必須讓消費者保留選擇的可能性，所有的售票伺服器資料隨時同步備份，邱光宗說：「讓每個人同時上網搶票是最公平的，也可以杜絕黃牛票。」所以，拓元選擇AWS的EC2，目標是能開多少臺虛擬伺服器，就先開多少臺，以張惠妹演唱會售票來說，就開了1,300臺左右。

不過，要因應秒殺搶票的關鍵問題是，如何處理大量使用者大量連線的記錄，拓元使用亞馬遜DynamoDB取代了原有的MySQL資料庫，先將登入使用者的資料存入資料庫後，再往後將資料拋轉以API接手處理。不過，因為開啟伺服器還是需要時間，便參考模擬以及過往經驗，事先算好要開啟的虛擬機器數量，不用優化、直接啟用即可。

透過API拋轉處理的資料再經過亞馬遜的負載平衡設備（Elastic Load Balancing，ELB）進行分流後，再行匯入拓元自己的票務資料庫。邱光宗表示，因為要保留消費者選區選位的即時性，前後端資料庫間，目前還有30～60秒資料無法同步的情況，未來還尋找其他服務來解決。



由此我們可知道其實一個搶票系統為了應應一次的大流量，其實是用了非常多的技術，並非像大家所說搶票系統真的做很爛其實背後還是付出了相當大的心血。

剛好最近的實習都有在碰 AWS 的服務，因此可以簡短描述拓元在這個系統有使用到的服務：

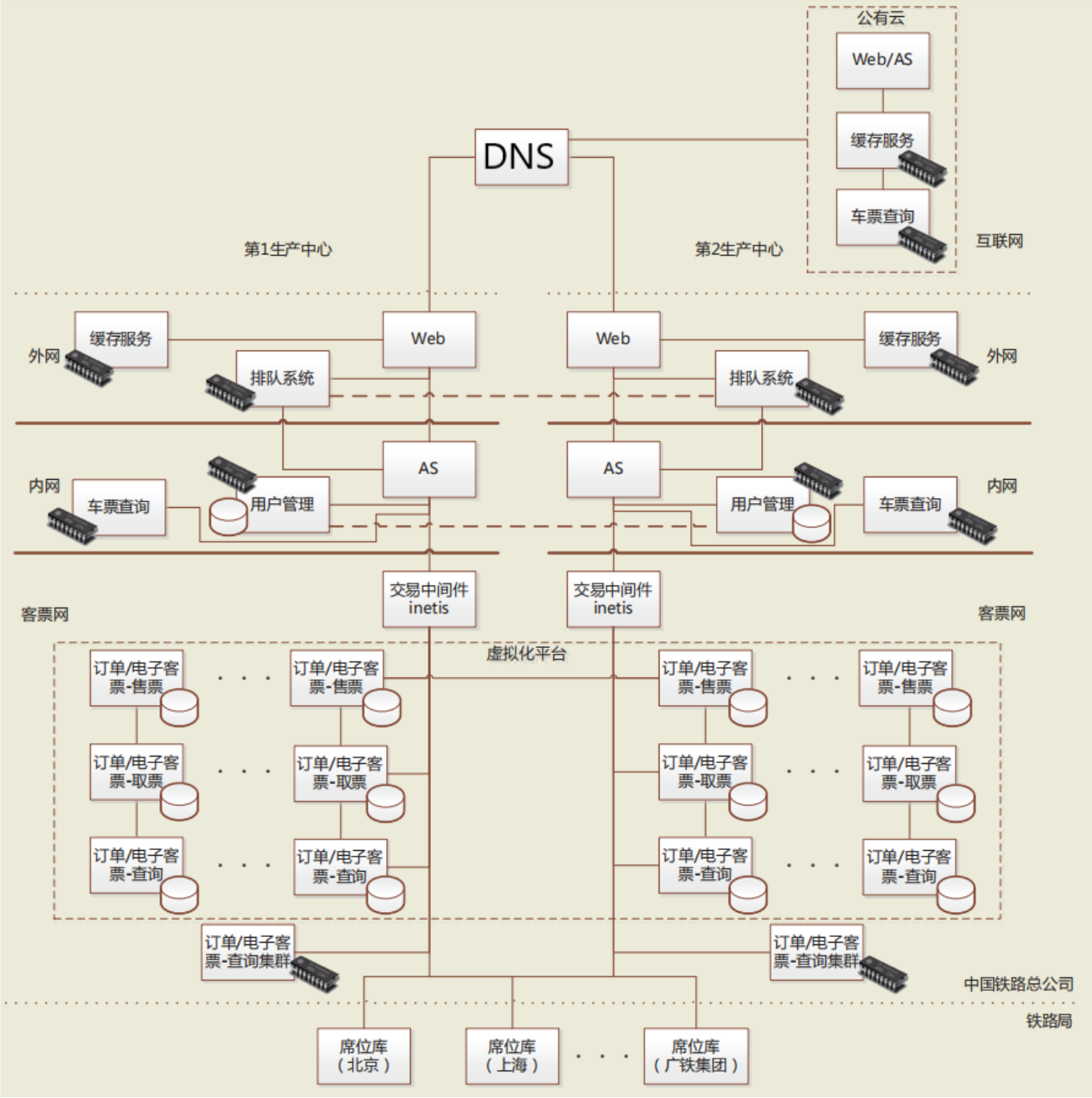
1. EC2：用來開虛擬機器，以應付大量的使用者連線，同時具備水平以及垂直擴展的能力，在這個 scenario，拓元看起來是使用了大量的水平拓展。
2. Elastic Load Balancing (ELB)：用來分流大量的使用者連線，以達到高可用性以及高擴展性。在大量流量進入 EC2 主機之前，ELB 會先將流量分流到不同的 EC2 主機上，以達到平均負載的效果。
3. DynamoDB：用來處理大量使用者大量連線的記錄，以及存放登入使用者的資料。在這個 scenario，拓元使用 DynamoDB 取代了原有的 MySQL 資料庫，以應付大量的使用者連線。
4. Amazon Route 53：用來管理 DNS 設定，以及提供高可用性以及高擴展性的 DNS 服務。
5. CloudFront：用來加速靜態資源的傳輸，以達到更快的存取速度。可以想像是個 cache 伺服器，將靜態資源存放在離使用者最近的地方，以達到更快的存取速度。
6. Amazon s3：用來存放靜態資源，以及提供高可用性以及高擴展性的物件儲存服務。

有了以上 AWS 服務的加持，我們最主要是不需要在地端的環境架設主機環境，以傳統上的方法，我們需要有大量的主機，甚至會需要很多容器化的技術去大量啟用運送資源去應應大量的運算需求，甚至我們還需要

顧慮資訊安全的問題，因為有大量 users 的時候，安全憑證也會是個問題需要去考量，甚至說可能會有攻擊者想要趁虛而入，因此我們需要有一個完善的安全機制去應應這些問題。

同場加映 -- 中國鐵路 12306 網站

另外在研究能夠負荷短時間、大流量的搶票網站系統架構時我想到了之間中國網友流傳的話：「國外知名大廠都無法做出一個能夠應付全中國返鄉流量的 12306」，我發現這個服務確實是不簡單，因為中國有多個省份，並且每次節日假期像是春運，幾乎都是全中國的人民都在使用這個網站，因此這個網站的負載量是非常大的，因此我們可以看到這個網站的架構是非常複雜的，並且有很多的技術在裡面，像是負載均衡、高可用性、高擴展性等等，這些都是我們在設計一個能夠負荷短時間、大流量的搶票網站系統架構時需要考慮的因素。



我想就以上架構圖做討論，我們從 DNS 進入的時候，我們需要有一塊可以查詢內容、並且存有緩存的公有雲，再者會需要能夠管理排隊的系統，光到這裡就非常敬佩，因為我們到底要怎去規劃這個排隊，主機到底

需要多少的效能？再來假如北京到南京，這個距離相差甚遠的情況，因為地理距離問題造成的時間差，要怎麼透過算法將之順利地加進隊列裡面？

再來就需要有用戶管理查詢了，同時要應付大量的 DB Query，Server 肯定需要能夠應付快速地讀跟寫，此外還需要再多台機器間相互溝通，這時候就需要有一個高效能的網路，這樣才能夠讓整個系統運作得更加順暢。

再來就是要把訂票的環節繼續進行下去，甚至還要分配給席位到不同的地區過去，不過這個架構圖裡面還有大量的技術還是我目前無法掌握的，或許這會是我未來努力的方向！

另外，我也可以觀察到當我們今天要做出一個能應付高流量的網站或服務時，如何運用有效地資源，並且要能夠讓機器能夠自動化地運作，而且最重要的就是讓服務不中斷，才是我們最需要關切的目標。

https://www.cnblogs.com/crazymakercircle/p/15058702.html