1. ใช้ git แบ่ง branch การทำงานที่ชัดเจน ยิ่งขึ้นเพื่อ ลดความซับซ้อนและง่ายต่อ manage คนมาทำงานยกตัวอย่าง เช่น มี branch Main จะ deploy ของบน production ทุกครั้งที่ merge ใน branch นี้อาจจะเขียน pipeline หรือทำ ci/cd ไว้เวลา merge code และให้ไป trigger deploy ใหม่

   ส่วน branch feature/xxxxxxx ก็อาจจะแตกเพื่อทำงานเฉพาะแต่ละ ฟีสเจอร์เพื่อให้ไม่กระทบกับโปรดักชั่น และ แยกฟีลเจอร์อย่างชัดเจน พอโค๊ตการทำงานที่เสร็จก็จะถูกกลับมา merge ที่ branch dev เพื่อทำการเทส ก่อนแล้วค่อยเอา merge production ที่เป็น branch main
2. อาจจะทำการ rebase โค๊ต หรือ ถอย branch กลับไป เพื่อให้ไปทำการ deploy ใหม่ หรือไม่ ก็อาจจะสมมุติ แตกไว้ อีก branch นึงเป็น old (ของเก่าก่อน deploy) ทุกครั้งที่ deploy ใหม่ หรืออีกตัวอย่าง อาจจะทำ pipeline อีกตัวนึงแยก replicaset ของ kube แล้วถอยเวอร์ชั่น
3. ถ้าเกิด conflict ของตัวโค๊ด ก็อาจจะ มา accept โค๊ตใน branch ว่าจะเอาอันที่ เป็นเข้ามาใหม่ (income) หรือตัวที่ อยู่จุดเดิม (current) จะเกิดขึ้นบ่อยถ้ากรณี hotfix ไป production แล้ว branch ที่เป็น feature อาจจะแตกจากตัวเก่าของ branch production และไฟล์นั้นเป็นไฟล์ที่แก้ไข
4. สเตตัส 200 ผ่าน แสดงถึง ว่า response ตอบกลับมามีข้อมูล หรือแล้วแต่ api จะตอบกลับ
5. สเตตัส 201 ผ่าน โดยแสดงถึงว่า มีข้อมูลส่ง เป็น method post ไปแล้ว โปรเซสถูกดำเนินการ จาก request เรียบร้อยแล้ว
6. สเตตัส 301 บ่งบอกถึงว่า จะส่งต่อ url ของ ผู้ใช้งาน ไป subdomain ใหม่ หรือ domain ใหม่
7. สเตตัส 400 หมายถึงว่า request ที่ส่งไปมีข้อผิดพลาด ไม่ถูกต้อง ไม่อนุญาติ ดำเนินการต่อไม่ได้
8. สเตตัส 401 หมายถึงไม่มีสิทธิ เข้าถึงในกรณีอาจจะยังไม่ยืนยัน ตัวตน หรือ token หมดอายุ หรือใช้อะไร something สักอย่างในการ authen ไม่มี เลยเข้าใช้งานไม่ได้
9. สเตตัส 403 อันนี้จะคล้ายกับ 401 แต่แตกต่างตรงที่ 403 บอกไปเลยว่า ไม่มีสิทธิเข้าถึง อาจจะ authen แล้วหรือไม่ ก็ได้ เปนการบอกไปเลย
10. สเตตัส 404 ไม่พบหน้าเพจ ไม่เจอ domain ที่หา
11. สเตตัส 500 มีปัญหากับฝั่ง server เกิดข้อผิดพลาด
12. สเตตัส 502 มีปัญหาจากการที่ server เชื่อมต่อกับ server แล้วมีปัญหายกตัวอย่าง user call หา service1 แล้ว service1 server1 คุยกับ service1 server2 แล้ว server2 ดับไปแล้ว ก็จะเป็นสเตตัส 502 ตอบกลับ user
13. สเตตัส 503 หมายถึง service นั้นอาจจะล้ม อาจเกิดจากสาเหตุ database ดับไปแล้ว หรือ คนส่ง request เยอะเกินไปก็เลยดับ
14. สเตตัส 504 หมายถึง server ที่ใช้ นั้นไม่ตอบกลับตามเวลา หรือหมดเวลาไปแล้ว เช่น service1 ใน server1 ทำงานช้าเกินไปก็เลยตอบกลับ trigger ไปหา user ว่า 504
15. ping ไปดูว่าเซิฟเวอร์นั้น เชื่อมต่อได้ไหม
    nslookup domain ดูว่า ip อะไร dns ถูกไหม
    netstat ดู ว่าในเครื่องเชื่อมต่อกับอะไรบ้าง
    traceroute ใช้ตรวจสอบเครือข่ายแสดงจุดที่มีปัญหา

Intermediant

1. Api gateway มีคนเรียกเยอะ นั้น อาจจะแก้ด้วยการทำ load balance แนวนอนกระจาย node ของ api gateway ไปในหลายๆ node

   ถ้าเกิดจาก ตัว service A เองก็ อาจจะทำการ จัดการ code ใหม่ optimal มัน ให้ดีกว่าเดิมดูว่ามันไปโหลด ตัวไหนเยอะ big O ดูว่าจะแก้ยังไง

   ถ้าเกิดจาก database ก็ optimal มันอาจจะทำ index ทำ querie ใหม่ ให้เร็ว ขึ้น

   ถ้าเกิดจาก network ก็ ทำแคช ข้อมูลไว้ ใน server

2. ใช้ alpine ของ image เพื่อลดไซต์และตอน build ลง library จะ มีแคช ให้เคลีย แคชทิ้ง เคลีย apt
3. เมื่อ cluster ถูกสร้างขึ้นระหว่างภายในแล้วจะมี ip address และ ก็ dns อยู่ภายในจะมี component ตัวนึงของ kubenertes ที่เรียกว่า kube-proxy คุยกันระหว่างใน cluster อีกที เพื่อใช้ service คุยกันภายใน cluster ครับ.
4. L2 คือ data link layer ใช้ที่อยู่ ใน Media access control เพื่อกระจายส่งข้อมูลของโหลดบาลานซ์
   L4 คือ transpot layer ทำหน้าที่จัดการภาระงาน โดยใช้ ip address และพอร์ตเพื่อแยกโหลด ของการ traffic network เพื่อคุยกันข้างใน load

balance

L7 คือ application layer ทำหน้าที่ ใช้แยกการเชื่อมต่อระหว่าง http header เป็น หน้าที่หนึ่งที่ใช้ใน load balance

When use it ?

ใช้เมื่อ L2 ใช้สำหรับภายใน local network และ load balance L4 ใช้สำหรับกระจาย network traffic ข้ามเซิร์ฟเวอร์หลายเครื่อง และ load balance L7 ใช้สำหรับกระจายทราฟฟิกแอปพลิเคชันข้ามเซิร์ฟเวอร์หลายเครื่อง

Professional

1. การ manage logs ผมอาจจะใช้ elastic search ในการทำ store ข้อมูลไว้ในนั้น

   metric ใช้ Prometheus ในการ store ข้อมูล CPU RAM I/O MEMORY STORAGE

   alerting สำหรับ infrastructure and application ผมจะใช้ Prometheus Alerting ทำ notify แจ้งเตือนอาจจะ slack discord หรือ tools chat อาจจะเป็น (Line Notify) Control alert ด้วย Promgen condition ในการส่งต่างๆ

   infrastructure as code ผมจะเลือกใช้ terraform ในการทำ provision automatic scale up , down

   ทำไมถึงเลือกใช้เพราะอะไร จริงๆ ทุกตัวเปรียบเสมือน data ที่ทำมาเพื่อให้ grafana เชื่อมถ้าถามถึงเหตุผลลึกๆอาจจะยังตอบไม่ได้เนื่องจาก ส่วนตัวคิดว่า tools open source และ ถ้า เริ่มจาก budget น้อยก็คงจะเลือก form ประมาณนี้

2. Application

   - ทำ autorized access ของฟังก์ชั่น หรือข้อมูล ต่างๆ อาจจะ authen token สำหรับ feature access
   - เขียนโค๊ต ให้ปลอดภัย ด้วยการ ทำ validation ซ้ำ หรือ ทำ hash การ escape ข้อมูล
   - ทำ penetration test ต่างๆ ลองทดสอบกับแอพเราว่า มีช่องโหว่ตรงไหนบ้าง

Infrastructure

- ตรวจสอบ infrastructure ที่ออกแบบว่ามีปัญหาช่องโหว่อะไรบ้างรึปาว
- ใช้ firewall ในการ control traffic
- ทำ network segment แบ่ง เครือข่ายเป็นย่อยๆ กระจายไป

data

- ใช้ encrypetion เพื่อ ป้องกันข้อมูลสำคัญระเอียดอ่อน เพื่อให้ปลอดภัยเบื้องต้น
- ควรทำ backup และ disaster recovery ป้องกันข้อมูลหายหรือเกิดข้อมูลเสียหายเพื่อเวลาเกิดปัญหาอาจจะสวิทต์ มาใช้ ตัวที่ backup หรือจะ rebase ข้อมูลใหม่
- ทำ tracking การเข้าถึงข้อมูล เพื่อให้มั่นใจ ว่าคนที่เข้าถึงข้อมูลนั้น ถูกต้องและ เหมาะสมสำหรับ ดึงข้อมูล

3. Software upgrade

- อัพเดท แบบ rolling เพื่อปรับใช้ซอฟต์แวร์ปรับใหม่ ทุกครั้งที่ถูก deploy ใหม่

- วางแผน rollback ทำ trackversion กรณีมีปัญหา

Database migration

- migrate database ตามไฟล์ แล้ว migrate ไฟล์ไปเลยจาก ไฟล์ migrate เพื่อสร้าง database ตามสเปค

- Sharing เพื่อกระจายข้อมูลไปยังหลายฐานข้อมูลเพื่อลดผลกระทบของการย้ายฐานข้อมูลเดียว

Incident

- มีแผนรับมือเหตุการณ์ที่มีการจัดทำเป็นเอกสารไว้อย่างดี ซึ่งรวมถึงขั้นตอนสำหรับการระบุและแก้ไขเหตุการณ์อย่างรวดเร็ว

- ตรวจสอบระบบและแอปพลิเคชัน หาปัญหาตั้งแต่เนิ่นๆ และลดเวลาหยุดทำงานให้เหลือน้อยที่สุด

4. How ?

ออกแบบคลัสเตอร์ Kubernetes ต้องดูจุดประสงค์ เพื่อให้แน่ใจว่าคลัสเตอร์มีประสิทธิภาพ ปรับขนาดได้ และยืดหยุ่นได้

1.Node Configuration: กำหนดจำนวนและความต้องการของทรัพยากร CPU, Memory และ Storage สำหรับแต่ละ Node

2.Cluster Networking: กำหนดเงื่อนไขการเชื่อมต่อเครือข่ายสำหรับ Cluster รวมถึงการใช้ Container Network Interface (CNI) plugin และการปรับใช้ Network Policies

3.High Availability: ให้แน่ใจว่า Components ของ Kubernetes Control Plane มีความพร้อมใช้งานสูงโดยการรัน instance หลายๆ ตัว

4.Storage: กำหนดประเภทของ Storage ที่เหมาะสมกับความต้องการของแอปพลิเคชัน เช่น Block Storage, File Storage หรือ Object Storage

5.Security: ปฏิบัติการเพื่อความปลอดภัย เช่น Access Control, Encryption และ Network Segmentation เพื่อป้องกันการโจมตีต่อ Cluster และทรัพยากรของมัน

6.Monitoring: ติดตั้งเครื่องมือในการตรวจสอบสถานะของ Cluster รวมถึง Metrics, Logging และ Tracing เพื่อตรวจจับและวินิจฉัยปัญหา

7.Scalability: ให้แน่ใจว่า Cluster สามารถขยายตัวได้ทั้งแนวนอนและแนวตั้ง เพื่อตอบสนองความต้องการที่เพิ่มขึ้น

What DNS Ingress, CNI ?

DNS คือ Kubernetes ใช้ DNS เพื่อแปลงชื่อโดเมนเป็นที่อยู่ IP CoreDNS เป็นโซลูชัน DNS ยอดนิยมที่ทำหน้าที่เป็นตัวแทนที่แบบดร็อปอินสำหรับ kube-dns
CNI เป็นเครื่องมือจัดการเครือข่ายใน Kubernetes โดยใช้ Flannel เป็นตัวเลือกที่นิยม สามารถกำหนด IP address ให้กับ pod ได้ง่ายดาย
Ingression คือ ใช้เพื่อจัดการเปิดเผยการเชื่อมต่อของโปรแกรมภายนอกที่ต้องการเข้าถึง Kubernetes โดยการใช้เทคโนโลยี HTTP(S) Load Balancer และ Reverse Proxy ที่จัดการการเชื่อมต่อ

DNS, CNI, และ Ingress ใน Kubernetes คือเครื่องมือที่ใช้ช่วยในการเชื่อมต่อและการทำงานของระบบ
เลือกใช้ตัวควบคุมแต่ละอย่างนั้นขึ้นอยู่กับความต้องการของระบบและปัจจัยต่างๆ เช่น ประสิทธิภาพ ความปลอดภัย และการใช้งาน
จึงต้องประเมินและเลือกใช้ตัวควบคุมที่เหมาะสมสำหรับ Kubernetes Cluster ของเราให้เหมาะสมที่สุด

5. i can't do this

6. SLO คือ เป้าหมายหรือค่าเป้าหมายของระดับบริการที่ต้องการให้กับผู้ใช้งาน เป็นตัววัดคุณภาพบริการเพื่อให้ผู้ใช้งานได้รับประสบการณ์ที่ดีในการใช้งานบริการของเรา
   SLA คือ ข้อตกลงระหว่างลูกค้ากับผู้ให้บริการ เพื่อกำหนดระดับคุณภาพของบริการ ซึ่งจะรวมถึงระยะเวลาการให้บริการ ระดับความสำคัญของบริการ และระดับผลลัพธ์ที่ควรได้รับ รักษาความพึงพอใจของลูกค้าในการให้บริการ
   สรุป เราควรใช้ SLO และ SLA ช่วยปรับปรุงคุณภาพบริการและเพิ่มประสิทธิภาพในการให้บริการของผู้ให้บริการ
และช่วยให้ลูกค้าได้รับประสบการณ์การใช้งานที่ดีขึ้น

7. i can't do this