1pg)

안녕하십니까. 숭실대학교 팀 빠숭 팀장 김도현입니다.

저희 팀의 넷챌린지 시즌8 공모전 연구주제는 “분산 엣지클라우드 환경에서 이벤트 기반으로 동작하는 Function as a Service 기능개발” 입니다.

2pg)

저희 팀은

개요 – 개발 목적 및 필요성 – 세부 개발 내용 및 범위 – 특징

의 순서로 발표하겠습니다.

3pg) – **개요**

먼저 개요입니다.

4pg)

특정 상황이 발생했을 때, CNN을 이용하여 그 상황의 이미지를 텍스트로 변환시켜주는 서비스 모델이 있다고 가정해 보겠습니다.

이러한 서비스 모델은 서비스가 필요한 상황에서나, 그렇지 않은 상황에서나 항상 백단에서 돌아가며 HW 리소스를 소모하고 있습니다.

5pg)

저희는 이러한 복잡한 기능을 하는 서비스가 백단에서 항상 작동할 필요 없이, 필요한 경우에만 불러 쓸 수 있다면 좋겠다. 라는 생각에서 Function as a Service, 서버리스 모델 개발 이라는 주제를 고안하게 되었습니다.

6pg)

따라서 저희는 이러한 FaaS에 대한 구현을   
컨테이너 관리도구인 쿠버네티스를 이용하여 수도권과 각 지자체를 워커노드로 설정하고 대전을 마스터로 설정한 환경에서 교통사고가 발생했을 때, 사고 발생 상황의 이미지를 텍스트화하여 SNS에 업로드하는 서비스로 구현해 보려 합니다. 해당 지역의 SNS채널에 업로드하므로써 사고 발생 지점 뒤에 있는 차량들과 주변 응급센터에 사고 정보를 전달할 수 있습니다.

7pg)

중앙 클라우드인 대전에서 각 지역들 즉 엣지 클라우드를 지속적으로 모니터링하고, 사고 이벤트 발생 시 그 이벤트를 trigger로 받아 동작하는 Function을 해당 엣지 클라우드로 배포하게 됩니다. 배포한 Function은 이미지를 텍스트화하여 SNS에 업로드하는 함수의 기능을 끝내면, 스스로 자원을 소모하지 않는 상태로 소멸 (scale-to-zero)를 수행합니다.

8pg) – **개발 목적 및 필요성**

다음은 개발 목적 및 필요성입니다.

9pg)

먼저 그에 앞서서, 서버리스에 대한 간단한 설명이 필요할 것 같습니다.

저희가 개발하려는 서버리스 모델은 BaaS, FaaS 중에서도 FaaS에 속하는데요, 이러한 FaaS Serverless 모델은 프로젝트를 여러개의 함수로 쪼개서 (혹은 한개의 함수로 만들어서), 매우 거대하고 분산된 컴퓨팅 자원에 여러분이 준비해둔 함수를 등록하고, 이 함수들이 특정 이벤트가 발생했을 때 실행되는 횟수 (그리고 실행된 시간) 만큼 비용을 내는 방식을 말합니다.

~~우리가 등록한 함수는 특정 이벤트가 발생했을때 실행됩니다.~~

10pg)

이러한 특성으로 이벤트 기반으로 서비스를 생성하고 확장하여 사용자가 인프라를 구축하고 관리할 필요가 없는 효율적인 서비스를 제공합니다. 또한 필요할 때만 함수형 서비스를 불러 쓰고 기능을 다하면 스스로 소멸되는 개념으로 scale-to-zero를 통해 인프라 자원을 매우 효율적으로 운용할 수 있습니다.

11pg) – **세부 개발 내용 및 범위, 예상 결과물**

다음은 세부 개발 내용 및 범위와 예상 결과물 입니다.

12pg)

저희 주제에서 가장 핵심인 scale-to-zero를 KEDA(Kubernetes Event Driven Auto-Scaling)이라는 기술로 구현하려고 합니다.

KEDA의 작동방식은

확장이 필요한 서비스를 REST API를 통하여 정의해 놓고, 외부 이벤트 trigger를감시하며 이벤트가 발생하여 서비스의 확장이 필요할 때 해당 워커노드에 pod(Function)이 존재하면 기존의 쿠버네티스 HPA(Horizontal Pod AutoScaler)를 이용하여 scale-out 하고 그렇지 않으면 KEDA를 통하여 0개에서 1개로 pod를 생성하는 방식입니다.

13pg)

따라서 이러한 서버리스 모델로 구현하는 어플리케이션에서는 각 지역별 노드를 코어 클라우드인 대전에서 모니터링하고, 각 지역별 노드의 CCTV영상을 HPC 서버로 전송하여 사고를 감지하고 사고 발생 시 사고 상황 이미지를 텍스트화 하여 SNS 지역채널에 업로드하는 서비스를 제공합니다.

14pg) – **특징**

이러한 결과물의 특징으로는

15pg)

이벤트 기반의 FaaS 서버리스 모델로서 이벤트 발생 시에만 포드를 생성 및 배포하고 그에 따른 scale-to-zero를 수행하며

쿠버네티스의 마스터-워커 구조로 이루어진 클러스터를 통해 각 지역 노드들의 상태를 모니터링 할 수 있습니다.

16pg) – **아이디어 구현 계획**

다음은 아이디어 구현 계획입니다.

17pg) – skip

18pg)

먼저 HW적 측면에서 대전을 마스터 노드로 설정하고, 나머지 지역과 수도권을 각각 워커 노드로 설정하여 인프라를 구축합니다.

19pg)

엣지 클라우드에서의 CCTV 영상은 실시간으로 HPC 이노베이션 허브로 전송되며, HPC에서는 전송 받은 영상의 이미지를 YOLO모델과 PyTorch를 이용하여 사고 발생을 인식합니다.

20pg)

엣지 클라우드가 정상적으로 작동하고 있는지 프로메테우스와 그라파나를 통해 Master에서 모니터링합니다.

21pg)

다음은 FaaS를 Kubernetes Native Serverless Framework라는 Kubeless 프레임워크와 MQTT 프로토콜을 이용해 구현해 본 것인데요, CCTV영상에서 사고가 발생하면 MQTT 프로토콜로 마스터에게 해당 사실을 알리고, 마스터에서 이를 감지하여 워커 노드로 function을 배포하는 시나리오입니다.

22pg)

마스터에서 이런식으로 작성된 function 파일이 존재한다면, kubeless CLI를 이용해 이렇게 function을 쉽게 워커노드로 배포할 수 있습니다.

23pg) – (빠르게)

MQTT 프로토콜의 구성요소에는 Topic을 구독하는 Subscriber가 존재하고 이러한 Topic을 이용하여 클라이언트에게 메시지를 보내는 Publisher가 존재합니다.

24pg)

이러한 MQTT프로토콜을 이용하여 저의 로컬환경에 common 이라는 Topic으로발행하는 Publisher에 대한 코드를 작성하고, 쿠버네티스 클러스터 마스터에 common 토픽을 구독하는 Subscriber에 대한 코드를 작성하였습니다. 이 때, 마스터 노드에서는 메시지를 받았을 때 msg-send-service 라는 function을 배포하도록 작성하였습니다.

25pg)

마스터 노드에 Subscriber 를 먼저 동작시키고, 로컬의 Publisher를 통작시켰을 때, 쿠버네티스 클러스터의 마스터에서 워커노드로 의도한 function이 정상적으로 배포된 것을 확인할 수 있습니다. 또한 이렇게 function call을 했을 때 작성된 메시지가 띄워지는 것을 확인하여 함수가 잘 동작하고 있음을 확인하였습니다.

26pg)

따라서 이러한 방식의 Event기반 scale-to-zero를 구현하며 HPC에서 사고를 감지하면 MQTT 프로토콜로 마스터 노드에게 사고 사실을 알리고, Master의 데이터베이스에 사고 이미지를 저장하며 워커 노드가 사고 이미지를 받아 텍스트화하여 메시지를 생성하는 어플리케이션 포드를 생성 시 마스터의 DB에서 이미지를 가져와 그 이미지를 기반으로 서비스를 생성하게 됩니다.

27pg) – **KOREN 연동 및 활용방안**

28pg)

KOREN 연동 및 활용방안입니다.

KOREN의 지역별 노드와 마스터의 클러스터 구성을 KOREN IaaS VM자원을 이용해 구성합니다.

HPC서버의 고성능 GPU를 사용해 교통사고를 인식하는 딥러닝 알고리즘을 수행합니다.

NIA에서 제공하는 PaaS-TA를 활용해 Redis Database 서비스를 사용하겠습니다.

KOREN이 제공하는 NFV서비스를 이용해 엣지 클라우드 VM들의 네트워킹을 구현하겠습니다.

29pg) – **결과물의 통합 및 시험/ 검증 계획**

30pg)

결과물의 통합 및 시험 / 검증 계획에서는 HPC서버에서 교통사고 감지 시 엣지클라우드에 포드가 정상적으로 배포되는지 확인하고,