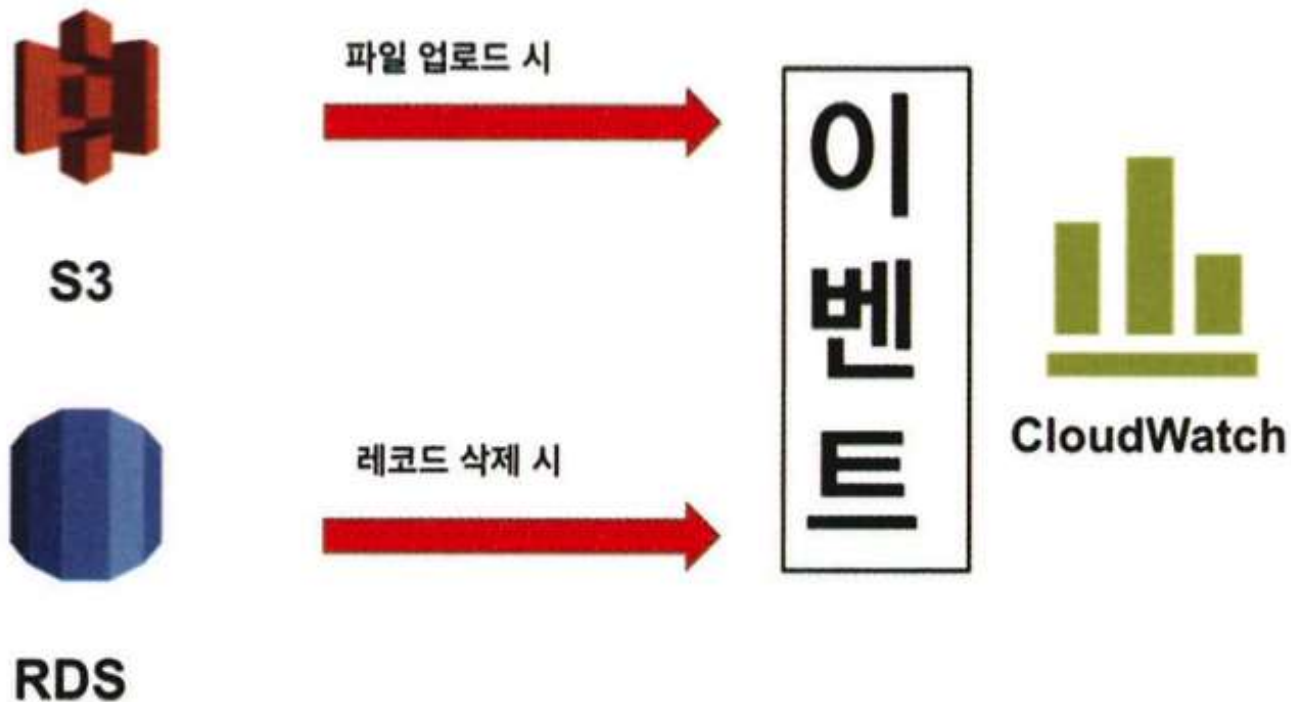


AWS Lambda

Cloud Watch

❖ 개요

- ✓ CloudWatch는 리소스 와 애플리케이션을 실시간으로 모니터링하는 서비스



Cloud Watch

❖ 개요

✓ 로그

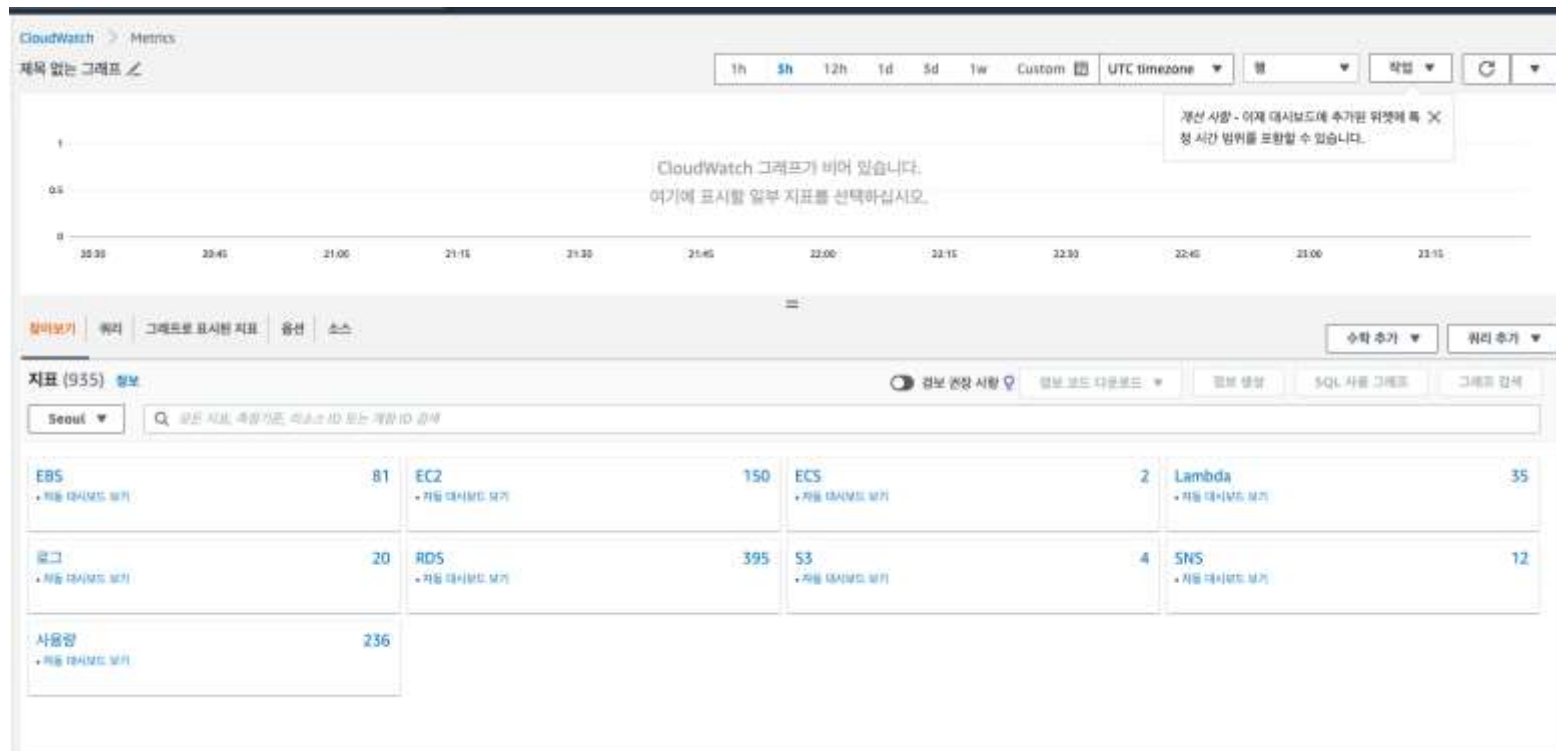
- 애플리케이션의 리소스를 기록
- CloudWatch의 로그는 로그 그룹 단위로 기록
- 여러 서비스에서 발생하는 로그를 기록하며 이를 필요에 따라 필터링하여 원하는 데이터를 검색할 수 있음
- 로그 기록 만료 시점을 설정할 수 있음
- 로그 그룹 이름을 클릭하여 세부 정보를 확인할 수 있음
- 로그 그룹의 현재 저장된 용량을 확인할 수 있고 삭제 및 Amazon S3에 저장할 수 있다
- 로그 그룹에서 로그 데이터를 확인할 수 있음
- 이를 삭제하거나 필터링하여 원하는 기준으로 데이터를 찾을 수 있는데 검색 결과를 복사하여 CSV로 다운로드할 수 있음
- Logs Insights를 사용하면 원하는 로그를 쿼리로 호출하여 확인 가능

Cloud Watch

❖ 개요

✓ 지표

- CloudWatch에서는 지표를 통해 그래프 형식으로 로그를 표현



Cloud Watch

❖ 개요

✓ 이벤트

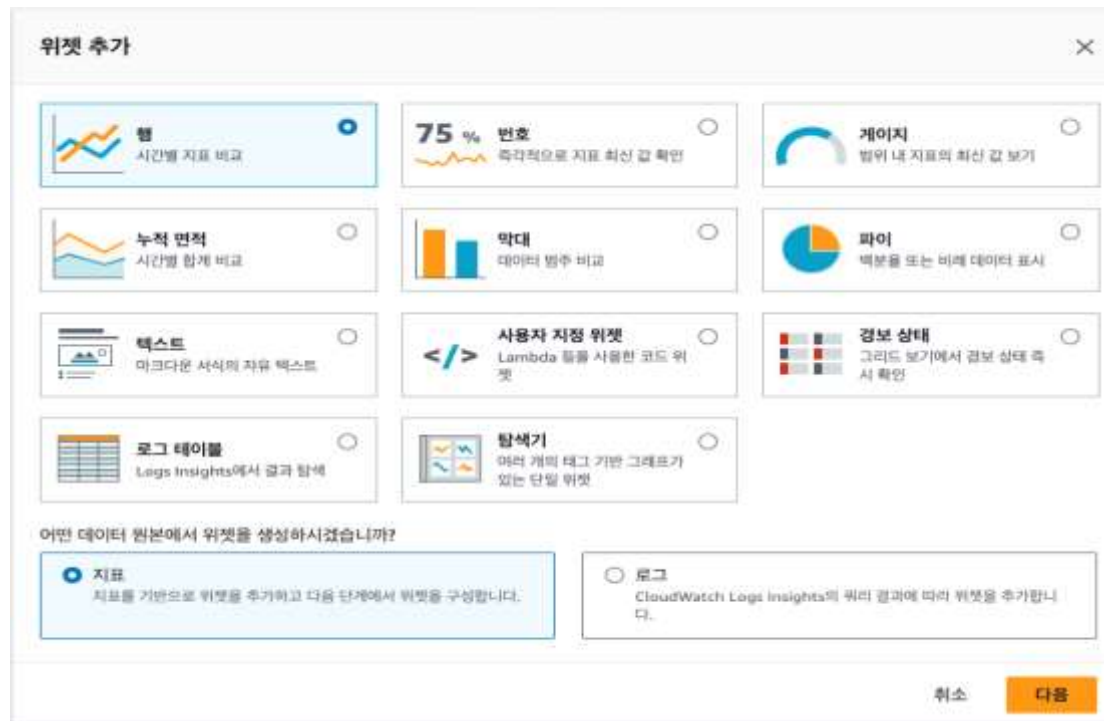
- CloudWatch는 이벤트를 제공
- 규칙을 사용하면 서비스 트리거, Cron job과 같은 형식의 작업이 가능
- 규칙은 이벤트 패턴 또는 일정으로 생성할 수 있음
- 이벤트 패턴은 서비스와 트리거하는 개념으로 특정 서비스를 지정하고 이벤트 유형을 선택한 후 대상 서비스를 지정하여 동작하도록 설정할 수 있음
- 이벤트 패턴은 서비스 단위를 선택할 뿐만 아니라 사용자 지정 이벤트 패턴으로 직접 작성할 수 있음

Cloud Watch

❖ 개요

✓ 대시보드

- CloudWatch에서 대시보드를 사용할 수 있음
- 한 페이지 내에서 원하는 로그를 원하는 형식으로 확인할 수 있음
- 대시보드는 사용자가 직접 생성 및 수정 및 삭제가 가능하며 원하는 레이아웃 형태로 생성 가능



Cloud Watch

❖ 개요

✓ 모니터링 종류

● 기본 모니터링

- ◆ 무료이며 5분 간격으로 최소한의 데이터를 수집하고 사용자에게 이벤트 발생 여부를 알려줌
- ◆ CPU 사용량, 디스크 사용량, 네트워크 I/O 처리량 관련 상황을 체크할 때 사용되는 모니터링 종류
- ◆ AWS 리소스 사용 시 ELB와 RDS를 제외하고 디폴트로 기본 모니터링 종류가 사용됨

● 상세 모니터링

- ◆ 무료가 아니지만 1분 간격으로 매우 자세한 데이터를 수집
- ◆ 대부분의 리소스는 기본 모니터링에서 상세 모니터링으로 변경 가능하나 ELB와 RDS는 상세 모니터링 옵션이 디폴트로 선택되어 있으며 이 둘은 기본 모니터링 기능을 사용할 수 없음

Cloud Watch

❖ 개요

✓ 사용 사례

● 서버 과부하 문제 해결

- ◆ 쇼핑 앱을 하나 만들었고 매일 전 세계에서 얼마나 많은 유저가 앱을 사용하는지 알아야 하는 경우 특정 요일과 시간 별로 나눠서 언제 유저가 많이 몰리는지 패턴을 분석해야 함
- ◆ 만약 공휴일에 비정상적으로 유저가 많이 몰리면 서버 과부하 문제가 발생하며 수많은 트래픽으로 인해 병목 현상이 일어날 수 있는데 매일 개발자가 확인하는 것은 현실적으로 불가능한데 Cloud Watch를 통해 언제 얼마나 많은 사람이 앱에 접속했는지 로그와 매트리스를 통해 한눈에 확인할 수 있으며 이를 기반으로 앱 사용 패턴을 파악하고 병목 현상 문제를 유연하게 대처할 수 있음



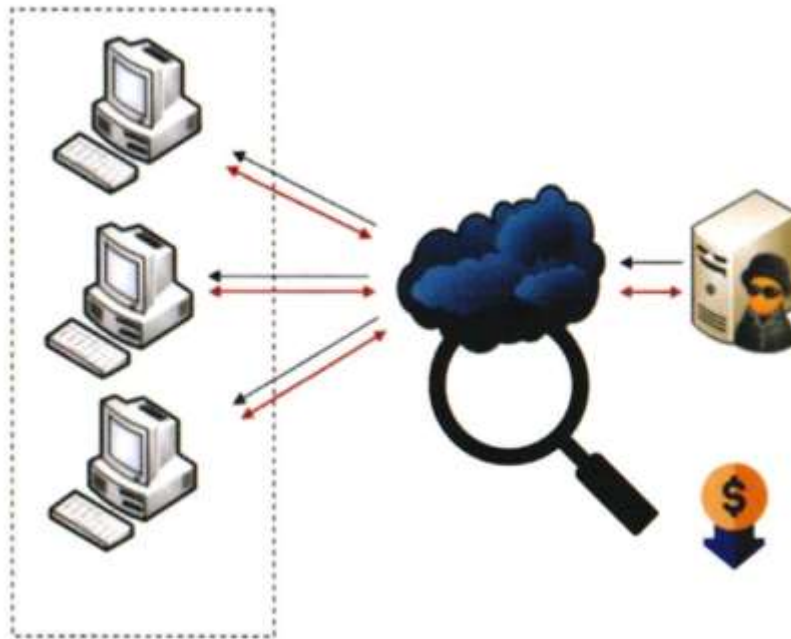
Cloud Watch

❖ 개요

✓ 사용 사례

● 비용 절감 효과

- ◆ Cloud Watch 경보 기능을 활용하면 서버 트래픽이 특정 임계점에 도달했을 때 개발자에게 알람을 알려주고 성능을 자동으로 낮출 수 있음



Cloud Watch

❖ 경보

✓ 세가지 상태

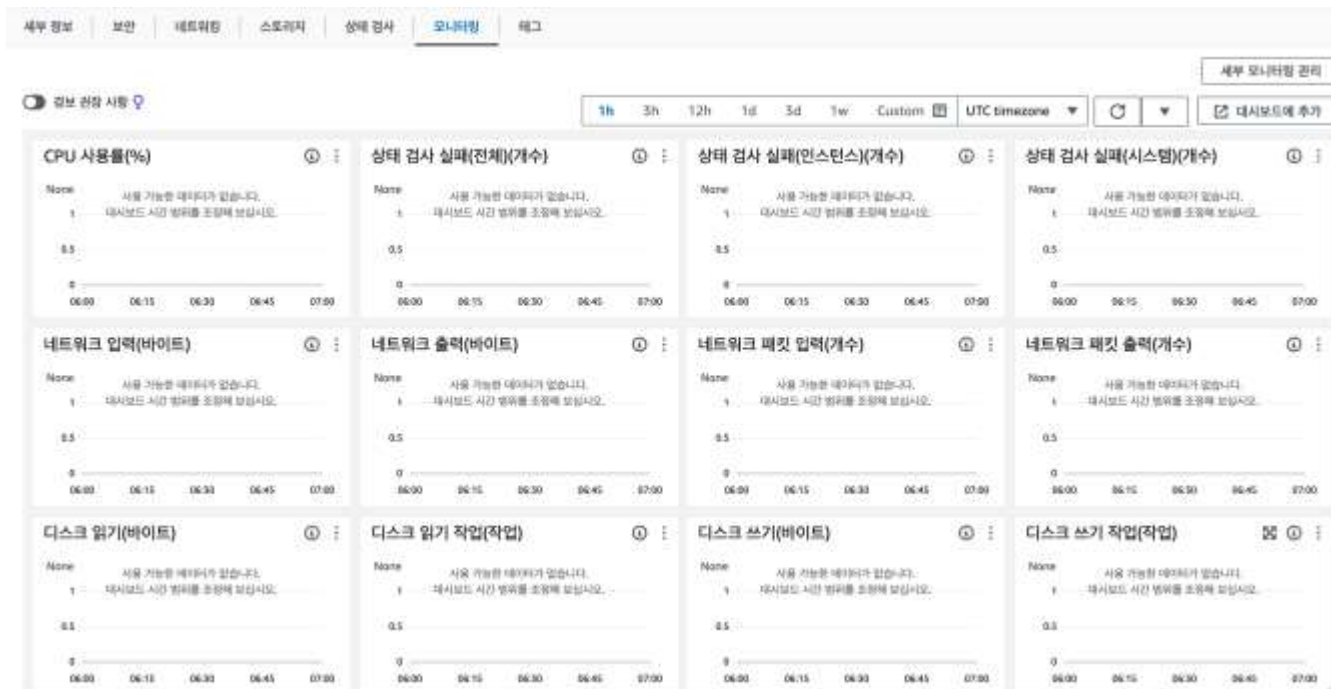
- CloudWatch 경보에는 세 가지 다른 상태가 존재하며 예외는 없음
- OK 상태는 Cloud Watch에서 수집하는 매트리스가 개발자가 지정해 놓은 임계값에서 타당한 값으로 들어온다면 OK 상태로 경보가 울릴 필요없는 정상적인 상태를 의미하며 임계값을 벗어나게 된다면 상태는 ALARM으로 변경되는데 개발자는 경보가 울렸을 경우 취할 수 있는 행동을 정의할 수 있음
- INSUFFICIENT_DATA 상태는 OK 혹은 ALARM으로 구분 지을 수 있을 만큼 충분한 데이터가 쌓이지 않거나 전달받지 못했을 경우에 발생
- IoT 기기에서 매초 데이터를 전달받아야 하는데 알 수 없는 방해 요인으로 데이터 전송 지연이 발생하여 CloudWatch는 데이터가 쌓이지 않고 있다고 판단하면 불충분 데이터라고 알려주는데 데이터가 없다면 매트리스가 존재하지 않기 때문

Cloud Watch

❖ 모니터링 실습

✓ EC2 인스턴스 생성

- SSH 나 Putty 로 접속할 수 있도록 생성하고 인스턴스 실행
- 모니터링 탭을 눌러서 현재 상태 확인



Cloud Watch

❖ 모니터링 실습

✓ EC2 인스턴스 생성

- SSH로 접속해서 내부 상태 확인: ls

```
ubuntu@ip-172-31-43-98:~$ ls
ubuntu@ip-172-31-43-98:~$
```

Cloud Watch

❖ 모니터링 실습

✓ 파일 전송 후 확인

- 용량이 큰 파일을 다운로드: <https://www.jetbrains.com/datagrip/download/>
- 리눅스 서버로 파일 전송

◆ Mac

scp -i pem파일경로 전송할파일경로 계정@접속할서버:저장될파일경로

◆ Windows

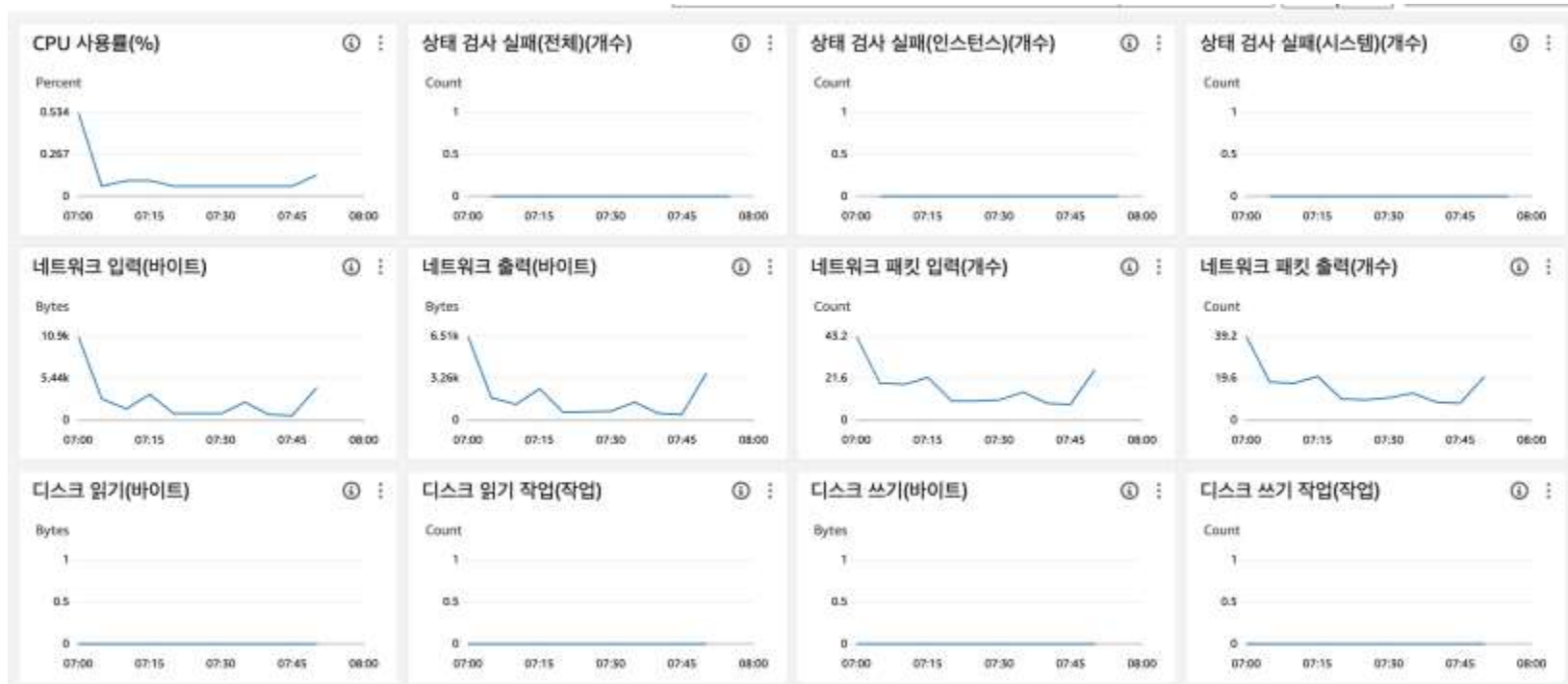
pscp -i ppk파일경로 전송할파일경로 계정@접속할서버:저장될파일경로

Cloud Watch

❖ 모니터링 실습

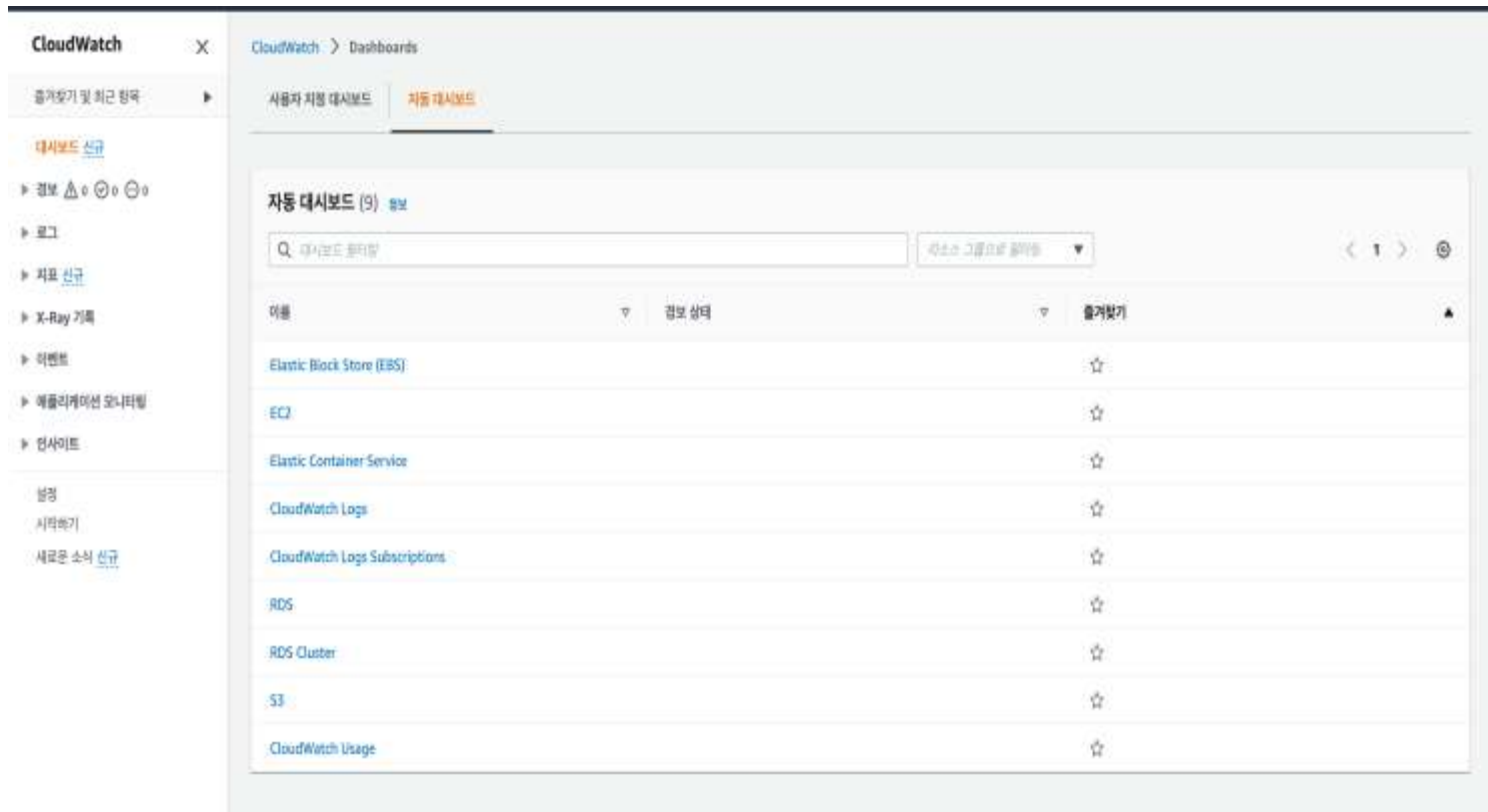
✓ 파일 전송 후 확인

● 모니터링 확인



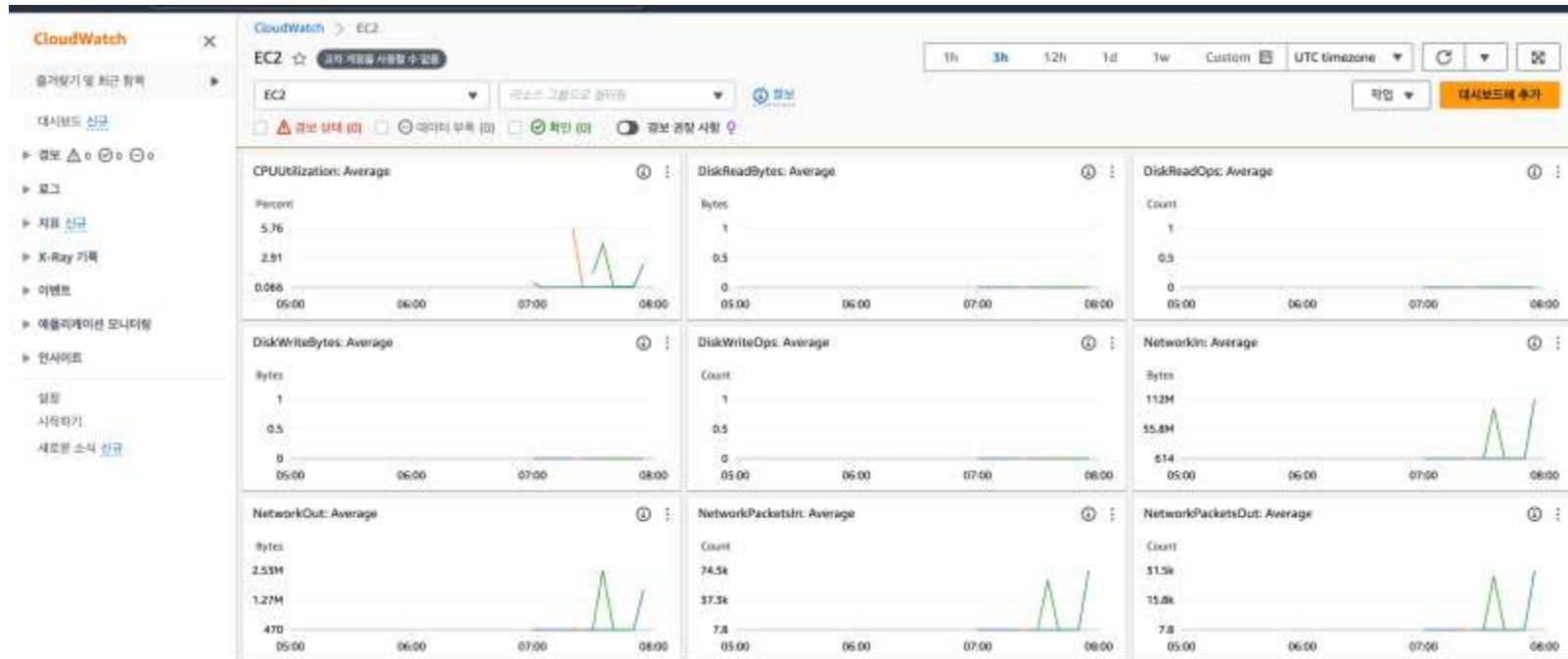
Cloud Watch

- ❖ Cloud Watch 대시보드에서 모니터링
 - ✓ Cloud Watch 에서 EC2 대시보드 확인



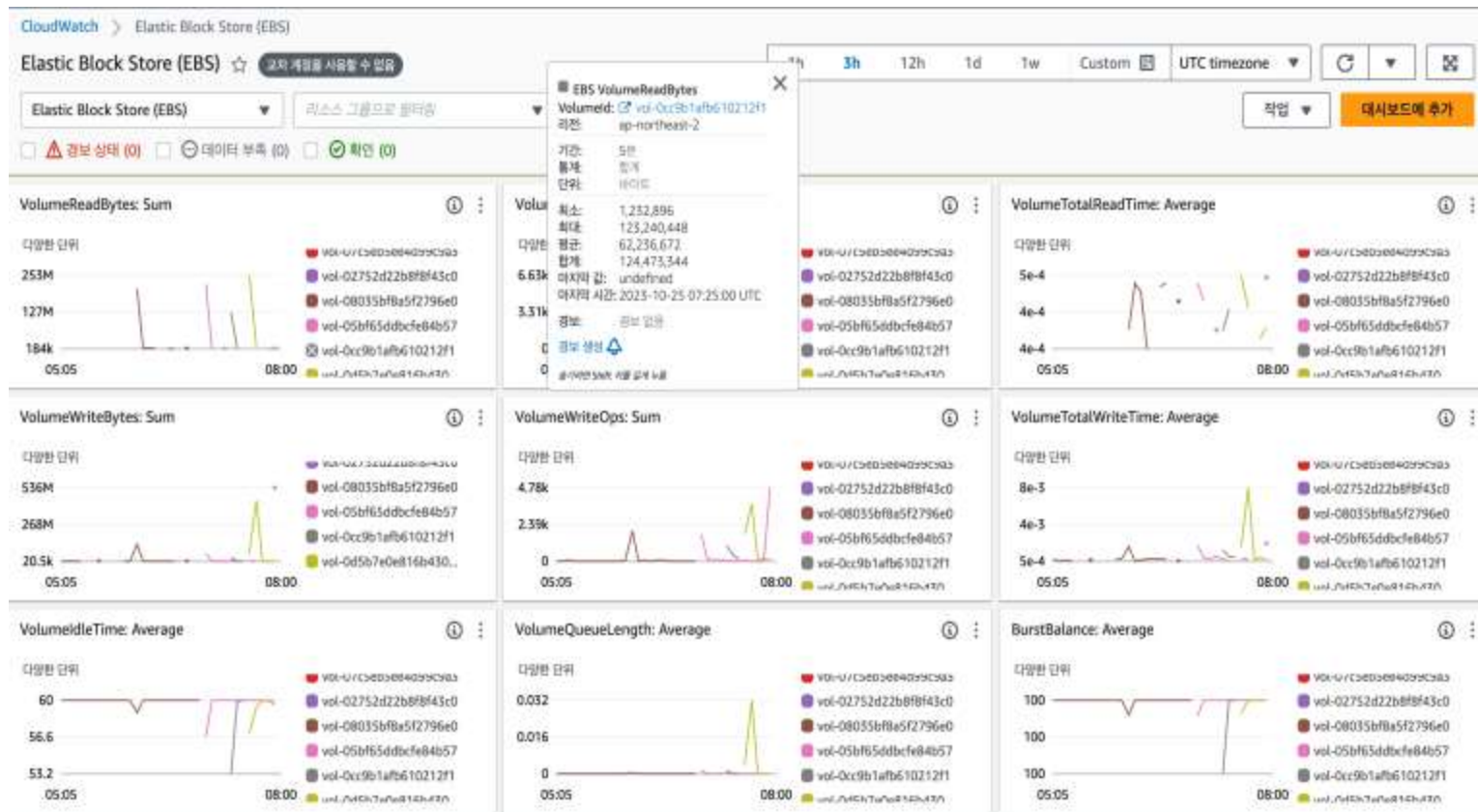
Cloud Watch

- ❖ Cloud Watch 대시보드에서 모니터링
 - ✓ Cloud Watch 에서 EC2 대시보드 확인



Cloud Watch

- ❖ Cloud Watch 대시보드에서 모니터링
 - ✓ Cloud Watch 에서 EBS 대시보드 확인



Cloud Watch

- ❖ Cloud Watch 경보 생성
 - ✓ SNS 관리 콘솔에서 주제 생성

Amazon SNS > 주제 > 주제 생성

주제 생성

세부 정보

유형: 정보
주제를 생성한 후에는 주제 유형을 수정할 수 없음.

☐ FIFO(선입선출)

- 엄격하게 보존된 메시지 순서 지정
- 정확히 1회 메시지 전송
- 높은 처리량, 초당 최대 300회 게시
- 구독 프로토콜: SQS

☒ 표준

- 최선의 메시지 순서 지정
- 최소 1회 메시지 전송
- 가장 높은 처리량(초당 게시 횟수)
- 구독 프로토콜: SQS, Lambda, HTTP, SMS, 이메일, 모바일 애플리케이션 엔드포인트

이름

itstudytopic

최대 256자이며 영숫자, 하이픈(-) 및 밑줄(_)을 포함할 수 있습니다.

표시 이름 - 선택 사항 정보

이 주제를 SMS 구독과 함께 사용하려면 표시 이름을 입력하십시오. 처음 10자만 SMS 메시지에 표시됩니다.

alam

최대 100자.

Cloud Watch

- ❖ Cloud Watch 경보 생성
 - ✓ SNS 관리 콘솔에서 주제 안에서 구독 생성

구독 생성

세부 정보

주제 ARN

프로토콜

구독할 엔드포인트 유형

엔드포인트

Amazon SNS의 알림을 수신할 수 있는 이메일 주소입니다.

① 구독을 생성한 후에는 확인해야 합니다. 정보

▶ 구독 필터 정책 - 선택 사항 정보

이 정책은 구독자가 받는 메시지를 필터링합니다.

▶ 제드라이브 정책(배달 못한 편지 대기열) - 선택 사항 정보

전송할 수 없는 메시지를 배달 못한 편지 대기열로 전송합니다.

취소 구독 생성

Cloud Watch

- ❖ Cloud Watch 경보 생성
 - ✓ 이메일이 전송되는데 이 때 confirm

itstudytopic

편집삭제메시지 게시

세부 정보

이름
itstudytopic

ARN
arn:aws:sns:ap-northeast-2:755510247212:itstudytopic

유형
표준

프시 이름
alam

주제 소유자
755510247212

구독

액세스 정책

데이터 보호 정책

전송 정책(HTTP/S)

전송 상태 로깅

암호화

태그

통합

구독 (1)

편집삭제확인 요청구독 확인구독 생성

< 1 > ⌂

ID	핸드포인트	상태	프로토콜
64425c53-7d44-456c-a32a-c7040313e27e	itstudy@kakao.com	확인됨	EMAIL

Cloud Watch

❖ Cloud Watch 경보 생성

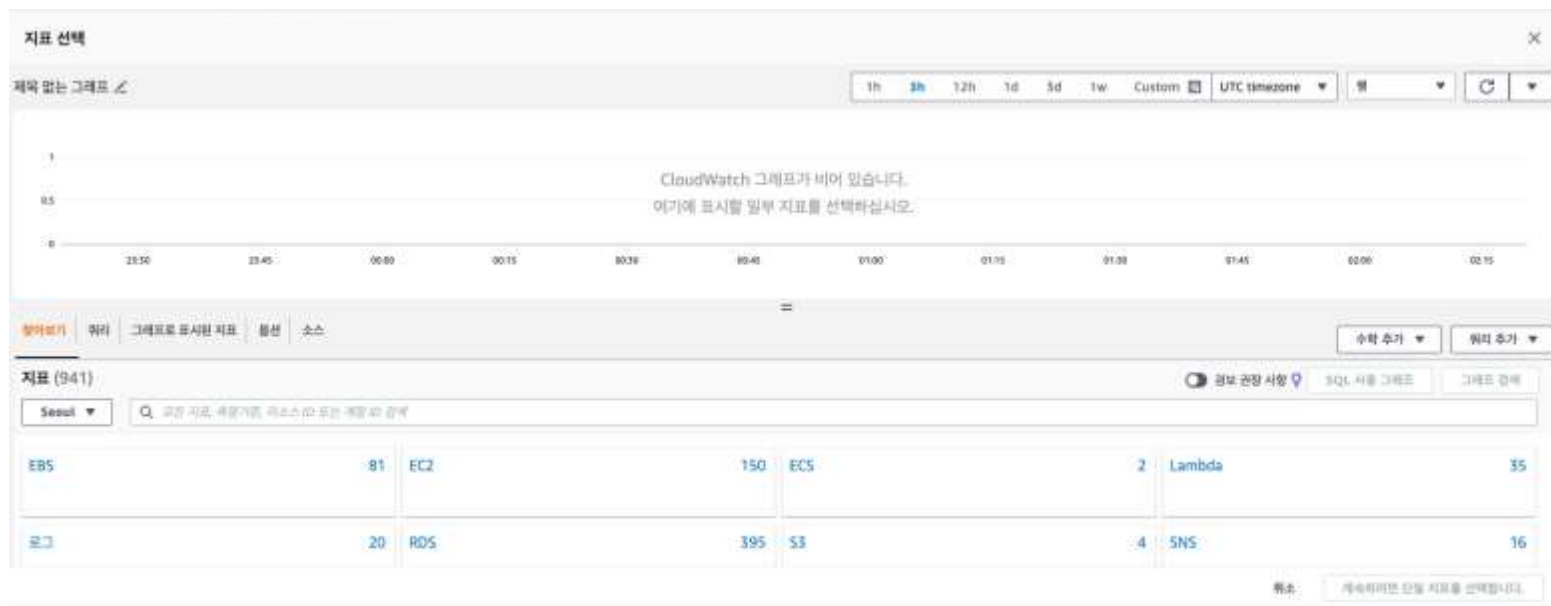
✓ 경보 생성

● 경보 생성 – 지표를 선택



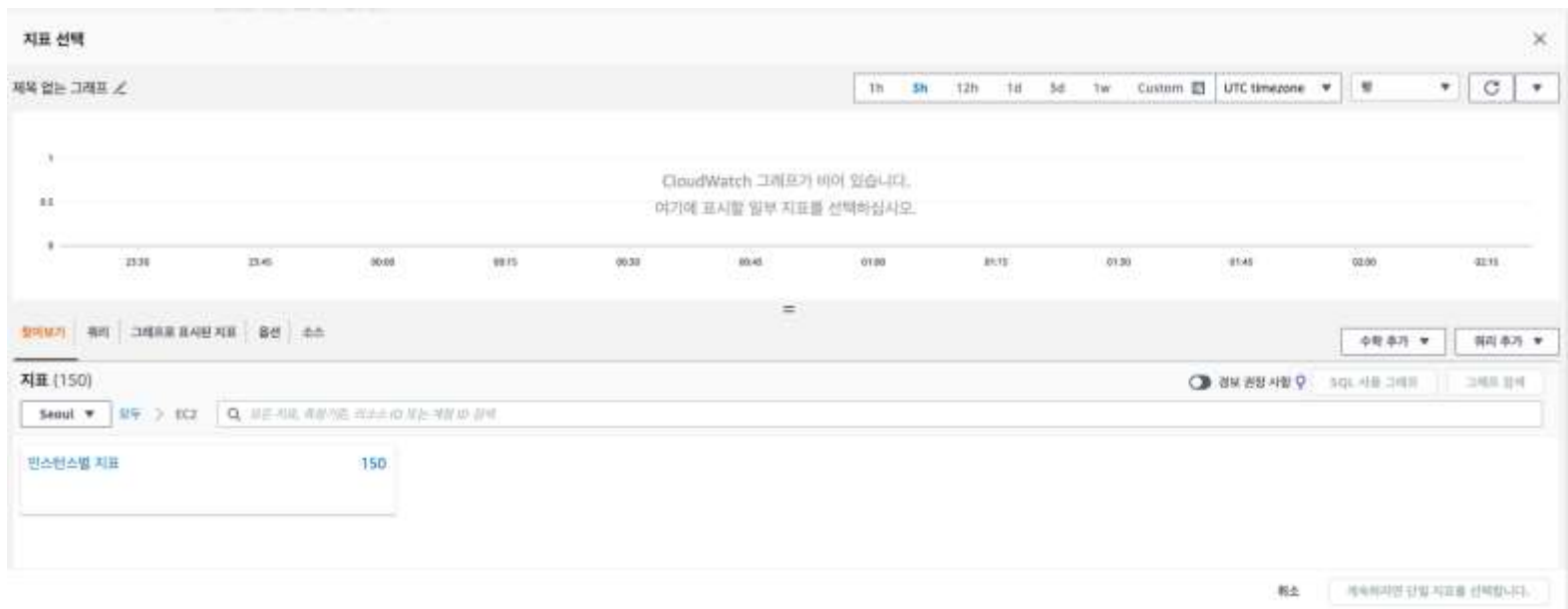
Cloud Watch

- ❖ Cloud Watch 경보 생성
 - ✓ 경보 생성
 - 경보 생성 – 지표를 선택



Cloud Watch

- ❖ Cloud Watch 경보 생성
 - ✓ 경보 생성
 - 경보 생성 – 지표를 선택



Cloud Watch

❖ Cloud Watch 경보 생성

✓ 경보 생성

● 경보 생성 – 지표를 선택

찾아보기	쿼리	그래프로 표시된 지표(1)	옵션	소스
☐	i-0442a136a3d5cf30d	StatusCheckFailed ⓘ		경보 없음
☐	i-0442a136a3d5cf30d	MetadataNoToken ⓘ		경보 없음
☐	i-099ad793dee38d156	MetadataNoToken ⓘ		경보 없음
☒	i-099ad793dee38d156	CPUUtilization ⓘ		1개의 경보
☐	i-099ad793dee38d156	NetworkOut ⓘ		경보 없음
☐	i-099ad793dee38d156	DiskReadOps ⓘ		경보 없음

Cloud Watch

- ❖ Cloud Watch 경보 생성
 - ✓ 경보 생성
 - 경보 생성 – 조건 설정

조건

임계값 유형

☒ 정적
값을 임계값으로 사용

☐ 이상 탐지
대역을 임계값으로 사용

CPUUtilization이(가) 다음과 같은 경우에 항상...
경보 조건을 정의합니다.

☒ 보다 큼
> 임계값

☐ 보다 크거나 같음
≥ 임계값

☐ 보다 작거나 같음
≤ 임계값

☐ 보다 작음
< 임계값

...보다
임계값을 정의합니다.

숫자여야 함

Cloud Watch

❖ Cloud Watch 경고 생성

✓ 경고 생성

- 작업 구성 – 이전에 만든 주제를 선택

알림

경보 상태 트리거
이 작업을 트리거하는 경보 상태를 정의합니다.

☒ **경보 상태**
지표 또는 표현식이 정의된 임계값을 벗어났습니다.

☐ **정상**
지표 또는 표현식이 정의된 임계값 범위에 있습니다.

☐ **데이터 부족**
경보가 방금 시작되었거나 사용 가능한 데이터가 부족합니다.

다음 SNS 주제에 알림을 보냅니다.
알림을 수신할 SNS(Simple Notification Service) 주제를 정의합니다.

☒ **기존 SNS 주제 선택**
☐ 새 주제 생성
☐ 주제 ARN을 사용하여 다른 계정에 알림

다음으로 알림 전송...

이 계정의 이메일 목록만 사용할 수 있습니다.

이메일(엔드포인트)
itstudy@kakao.com - SNS 콘솔에서 보기

알림 추가

Cloud Watch

- ❖ Cloud Watch 경고 생성
 - ✓ 경고 생성
 - 이름 설정

이름 및 설명 추가

이름 및 설명

경보 이름

itstudyalarm

경보 설명 - 선택 사항 서식 가이드라인 보기

편집 미리 보기

이것은 H1입니다
이중 별표는 강조 문자를 제공합니다.
이것은 [예제](https://example.com/) 인라인 링크입니다.

최대 1024자(0/1024)

❗ 마크다운 서식은 콘솔에서 경보를 볼 때만 적용됩니다. 설명은 경고 알림에서 일반 텍스트 서식으로 유지됩니다.

취소

이전

다음

Cloud Watch

❖ Cloud Watch 경보 생성

✓ 컴퓨터에 부하 발생

- ssh로 EC2에 접속
- 패키지 정보 업데이트: `sudo apt-get update`
- stress 패키지 설치: `sudo apt install stress`
- CPU 코어 수 확인: `grep -c processor /proc/cpuinfo`
- CPU 스트레스: `stress -c 2`
- 메모리 스트레스: `stress --vm 3 --vm-bytes 1024m`
- Disk 스트레스: `stress --hdd 3 -hdd-bytes 1024m`
- 다른 컴퓨터에서 EC2에 파일 전송
 - ◆ `scp -i <pem/ppk 파일명> <업로드할 파일> ec2-user@<퍼블릭 IPv4 DNS이름>:<인스턴스에 업로드될 파일명>`
 - ◆ Windows 는 scp 대신에 pscp
- 잠시 후 메일 확인

ServerLess

❖ Serverless

✓ 개요

- 서버가 없는건 아니고 그저 특정 작업을 수행하기 위해서 서버 프로그래밍을 별도로 할 필요가 없다 라는 의미
- BaaS(Backend as a Service) 혹은 FaaS(Function as a Service)에 의존하여 작업을 처리
- BaaS를 제공하는 서비스에는 Firebase, Kinvey 등이 있고 FaaS를 제공하는 서비스로는 AWS Lambda, Azure Functions, Google Cloud Functions 등이 있음

ServerLess

❖ Serverless

✓ 기존 기술

● 자체적 시스템 설계

- ◆ 시스템에서 필요한 모든 인프라를 직접 관리하는 것
- ◆ 공간, 하드웨어, 네트워크, 운영체제, 모두 직접 관리
- ◆ 가장 큰 문제점은 시스템이 커지면 전산실을 유지 할 관리자가 필요 할 것이고 이 인력에 대한 비용이 나가면 시스템의 확장이 어려움

● IaaS (Infrastructure as a Service)

- ◆ AWS, Azure 등의 서비스를 이용하는 방식으로 서버 자원, 네트워크, 전력 등의 인프라를 모두 직접 구축 할 필요 없음
- ◆ 인프라를 가상화하여 관리하기 쉽게 해주는 서비스를 통하여 관리자 패널에서 인프라를 구성하고 사용하는 방식으로 사용자는 가상 머신을 만들고 네트워크, 와 하드웨어를 설정하고 운영체제를 설치해서 애플리케이션을 구동하는 방식

● PaaS (Platform as a Service)

- ◆ IaaS 에서 한번 더 추상화된 모델로 네트워크 및 런타임까지 제공
- ◆ 사용자는 애플리케이션만 배포하면 바로 구동시킬 수 있는 방식으로 AWS Elastic Beanstalk, Azure App Services 등이 있으며 이를 사용하면 Auto Scaling 및 Load Balancing 도 손쉽게 적용 할 수 있음

ServerLess

❖ Serverless

✓ BaaS 와 FaaS

● BaaS(Backend as a Service)

- ◆ 모바일 혹은 웹 애플리케이션을 만들게 될 때 데이터를 저장하고 다른 기기에서도 접근하고 공유하기 위해서는 서버 개발은 필수적
- ◆ 서버 개발을 하다 보면 고려할 사항이 꽤 많은데 유저가 늘어나게 된다면 서버의 확장도 고려해야 하고 보안성 또한 고려해야 하는데 이는 어려운 작업
- ◆ Firebase 같은 BaaS 에서는 앱 개발에 있어서 필요한 다양한 서버의 기능들 (데이터베이스, 소셜 서비스 연동, 파일시스템 등)을 API로 제공해 줌으로써 개발자들이 서버 개발을 하지 않고서도 필요한 기능을 쉽고 빠르게 구현 할 수 있게 해주고 서버의 이용자가 순식간에 늘어나게 되면 자동 확장
- ◆ 현존하는 BaaS 중 가장 대중화 되어있는 것은 Firebase
- ◆ 장점
 - 개발 시간의 단축
 - 서버 확장 작업의 불필요함
- ◆ 단점
 - 클라이언트 위주의 코드
 - 가격
 - 복잡한 쿼리가 불가능함

ServerLess

❖ Serverless

✓ BaaS 와 FaaS

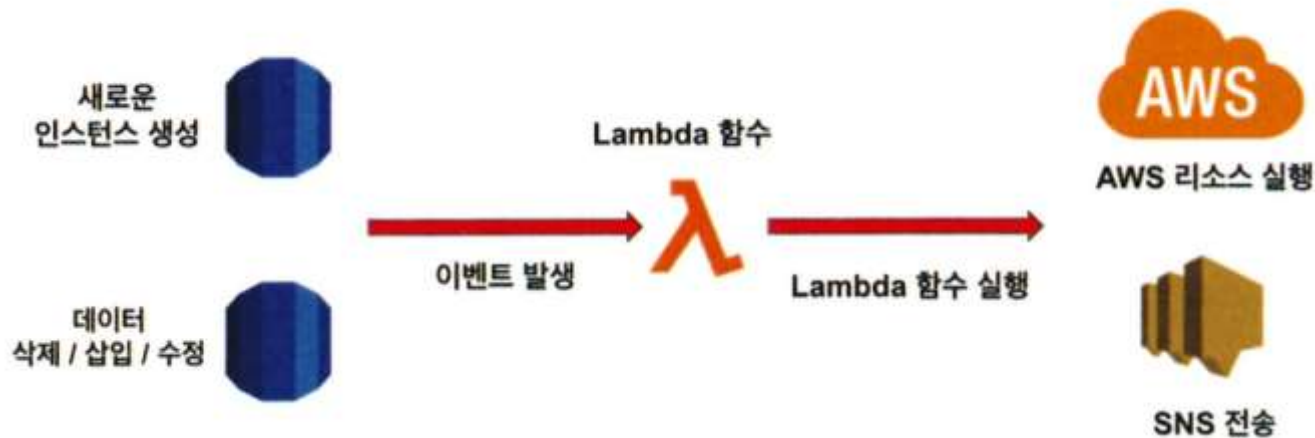
● FaaS(Function as a Service)

- ◆ FaaS는 개발자에게 서버를 관리할 필요 없이 이벤트에 대응하여 애플리케이션을 실행하도록 지원하는 추상화 기능을 제공
- ◆ FaaS 인프라는 주로 이벤트 기반 실행 모델을 통해 서비스 제공업체가 온디맨드로 미터링하는 것이 일반적
- ◆ 서비스로서의 플랫폼(Platform-as-a-service, PaaS) 처럼 서버 프로세스를 백그라운드에서 계속 실행할 필요는 없음
- ◆ 현대적인 PaaS 솔루션은 개발자가 애플리케이션을 배포하는 데 사용할 수 있는 일반적인 워크플로우의 일부로 서버리스 기능을 제공하여 PaaS와 FaaS 간 구분이 모호해짐
- ◆ 실제로 전체 애플리케이션은 기능, 마이크로서비스, 지속 실행 서비스 같은 여러 솔루션이 혼합된 형태로 구성
- ◆ FaaS의 이점
 - 개발자 생산성 향상 및 개발 시간 단축
 - 서버 관리의 부담이 없음
 - 손쉬운 확장 및 플랫폼에서 관리하는 수평적 스케일링
 - 필요한 경우에만 리소스를 사용하거나 리소스 비용 지불
 - 거의 모든 프로그래밍 언어로 기능 작성 가능

ServerLess

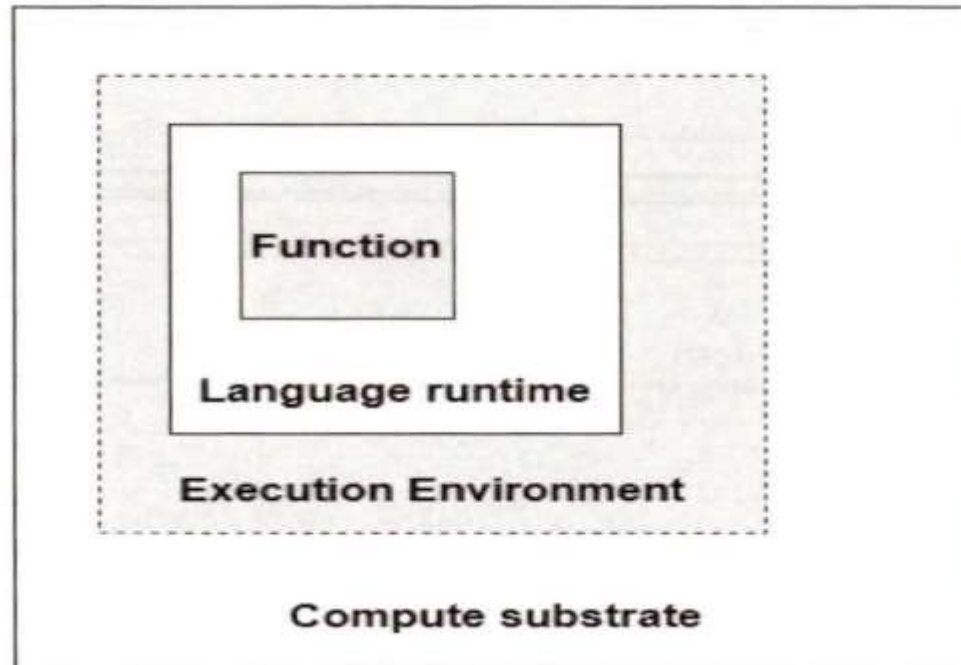
❖ AWS Lambda

- ✓ AWS Lambda란 Amazon Web Services에서 2014년부터 제공하는 FaaS 서비스
- ✓ AWS Lambda는 이벤트를 감지하여 아마존 리눅스 환경의 Micro VM을 띄우고 함수를 실행하고 결과를 처리
- ✓ 함수가 실행될 때 필요한 환경이 있는데 이것을 런타임이라고 하고 런타임은 어떤 언어로 작성하는지에 따라 다르며 그 환경에 따라 성능 차이가 있음



ServerLess

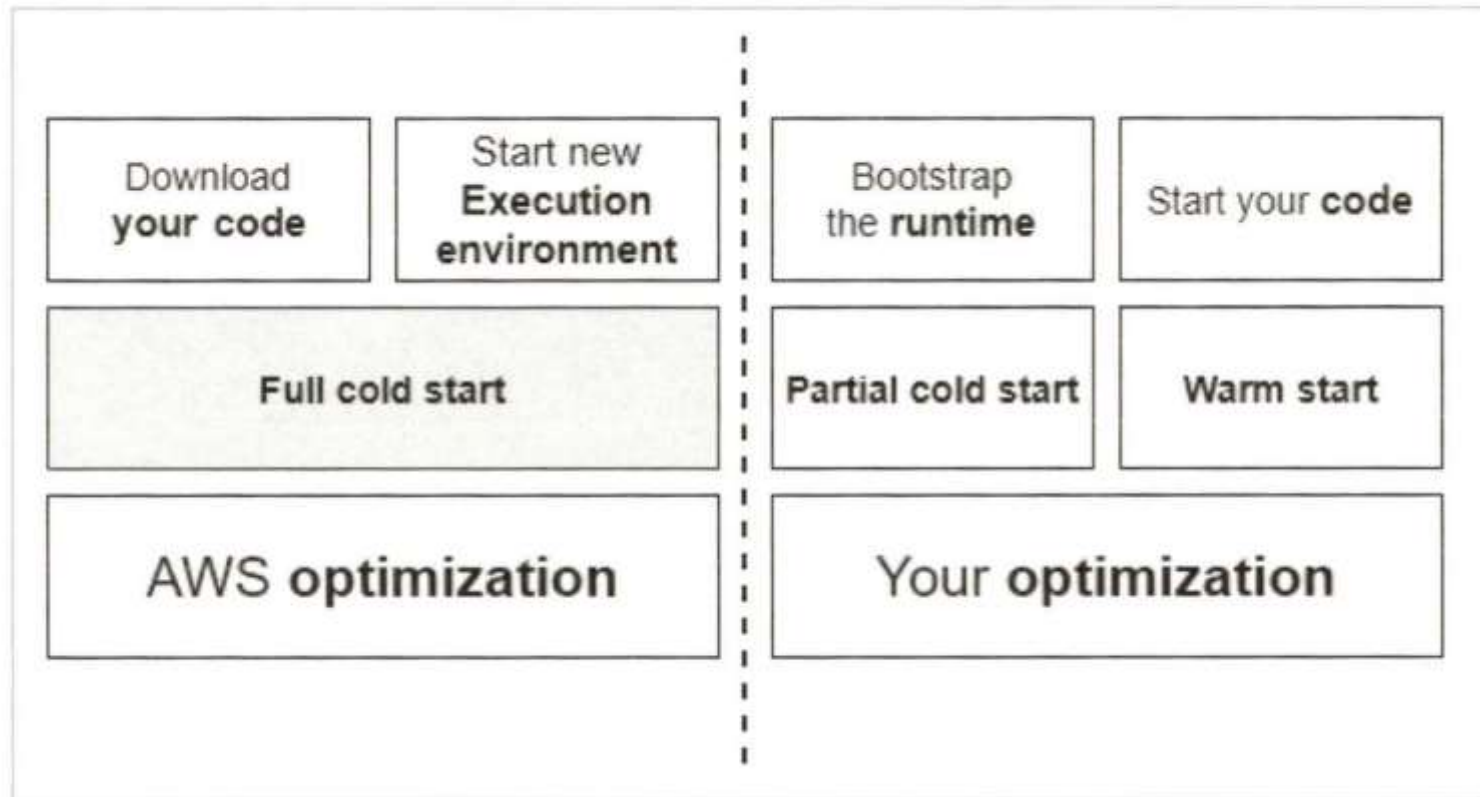
- ❖ AWS Lambda
 - ✓ 내부 구조



Compute substrate	함수가 실행될 Micro VM
Execution Environment	환경 변수 등의 실행환경
Language runtime	언어별 런타임
Function	함수

ServerLess

- ❖ Cold Start 와 Warm Start
 - ✓ Life Cycle



ServerLess

❖ Cold Start 와 Warm Start

✓ Cold Start 발생함으로써 딜레이 존재

- AWS Lambda가 실행되면 작성한 코드를 다운로드하고 실행 환경을 구성하는데 이때를 Full cold start 라고 하며 런타임을 준비하는 과정은 Partial cold start 라고 하며 마지막으로 함수가 실행 될 때를 Warm start 라고 함
- 함수가 실행될 때 순서가 정해져 있고 이에 따라 지연 시간이 발생하는데 Micro VM이 올라갔다가 내려간 뒤 다시 실행하면 Full cold start 부터 시작하고 만약 Micro VM이 유지되는 시간 이전에 재요청하면 Partial cold start가 바로 진행되어 지연 시간을 줄일 수 있음
- Cold Start를 줄이기 위해서는 지속적인 호출을 통해 구동되어 있는 Micro VM을 유지하는 것인데 이를 보통 5분 마다 호출하라고 권고하지만 실제로 각자 구현한 람다의 환경이 다르기에 정확한 수치가 아님

<https://theburningmonk.com/2017/06/aws-lambda-compare-coldstart-time-with-different-languages-memory-and-code-sizes/>

- cold start time은 언어/런타임, 리소스 양(mb) 및 함수 실행 시 갖고 오는 패키지/종속성에 따라 다름

ServerLess

❖ Runtime

- ✓ AWS Lambda에서는 Runtime을 사용하여 각각 다른 언어들로 작성한 코드도 동일한 기본 실행 환경에서 실행할 수 있음
- ✓ Runtime은 AWS Lambda 서비스 와 작성한 함수 코드 사이에 위치해 이벤트와 컨텍스트 정보 등 응답을 중계해주는 역할을 수행
- ✓ 기본적으로 제공하는 런타임 환경이 있고 직접 필요한 런타임 환경을 만들어 사용할 수 있음
- ✓ 제공하는 런타임 환경에는 여러 언어를 지원
- ✓ 함수를 생성할 때 런타임을 선택할 수 있고 필요시 런타임 환경을 변경할 수 있음
- ✓ 런타임 환경은 Amazon Linux
- ✓ 제공하는 런타임 환경에서는 각 언어의 버전 별 또는 운영체제의 수명이 60일 이내로 예정될 때 마이그레이션 안내 메일을 발송해 줌
- ✓ 런타임 지원이 중단되면 AWS Lambda는 호출을 비활성화 하는데 사용 중단된 런타임은 보안 업데이트 기술 지원을 받을 수 없기에 마이그레이션이 필수적
- ✓ 현재는 Node, Python, Java, Ruby, Go, .NET 런타임을 제공

ServerLess

❖ Event

✓ 호출 방법

동기적으로 호출하는 서비스	비동기적으로 호출하는 서비스
Elastic Load Balancing (Application Load Balancer) Amazon Cognito Amazon Lex Amazon Alexa Amazon API Gateway Amazon CloudFront (Lambda@Edge) Amazon Kinesis Data Firehose AWS Step Functions	Amazon Simple Storage Service Amazon Simple notification Service Amazon Simple Email Service AWS CloudFormation Amazon CloudWatch Logs Amazon CloudWatch Events AWS CodeCommit AWS Config AWS IoT AWS IoT Events AWS CodePipeline

ServerLess

❖ Event

✓ 호출 방법 차이

- 동기식은 클라이언트가 람다 함수에 이벤트를 보내고 클라이언트는 함수의 응답을 받을 수 있음
- 동기식으로 호출할 경우 파라미터 값을 invoke로 실행
- 비동기식은 클라이언트의 요청 이벤트와 람다 함수 사이에 대기열을 이용하는 방식으로 클라이언트는 성공 응답만 받음
- 비동기식으로 호출할 경우에는 호출 유형 파라미터 값을 Event로 설정해야 함
- 비동기식으로 진행할 때 오류가 발생하면 계속해서 재시도하는데 동시성 설정이 작게 되어 있을 경우(오류 = 492) 이거나 서버 오류(500번대) 일 때는 이벤트를 대기열로 다시 보내고 이는 최대 6시간 동안 재시도하며 재시도 간격은 최대 5분까지 증가
- 이벤트는 대기열에서 삭제될 수 있는데 동시성을 어떻게 조절하느냐 에 따라 이벤트를 보장할 수 있는데 대기열은 백업이 되기도 하며 이때는 람다 함수로 전송하기 전에 이벤트가 만료될 수 있으며 이 때문에 이벤트가 삭제될 수 있음

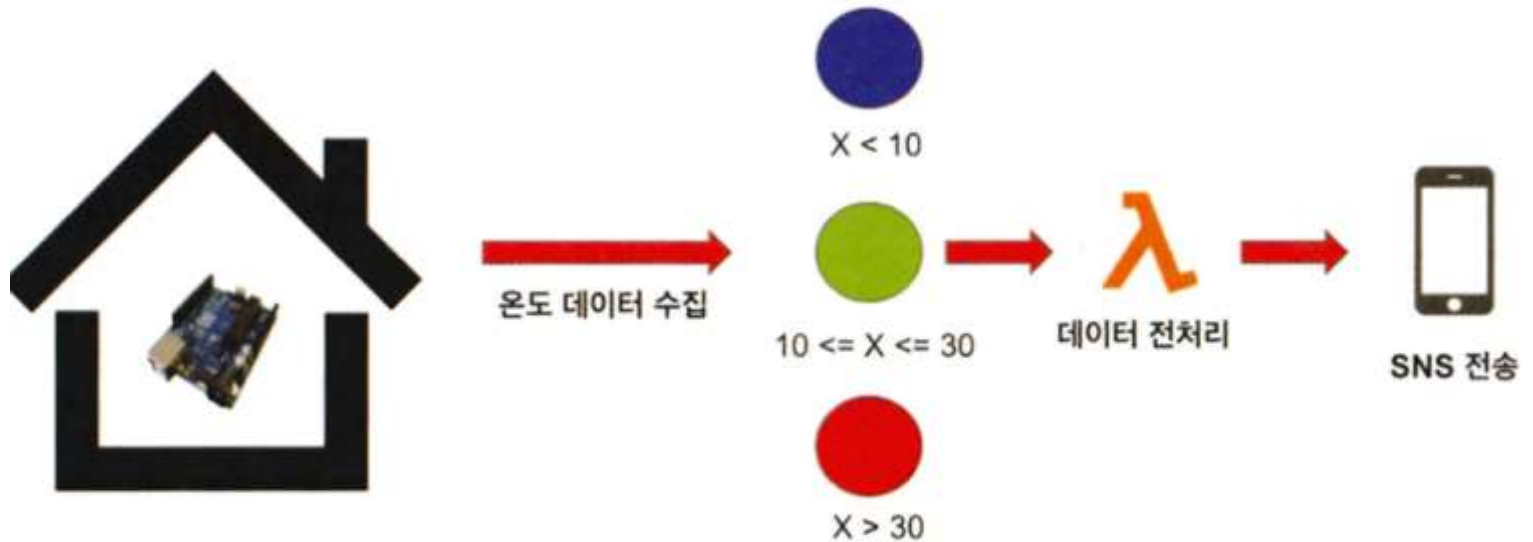
ServerLess

- ❖ AWS Lambda 사용 사례
 - ✓ 데이터 전처리



ServerLess

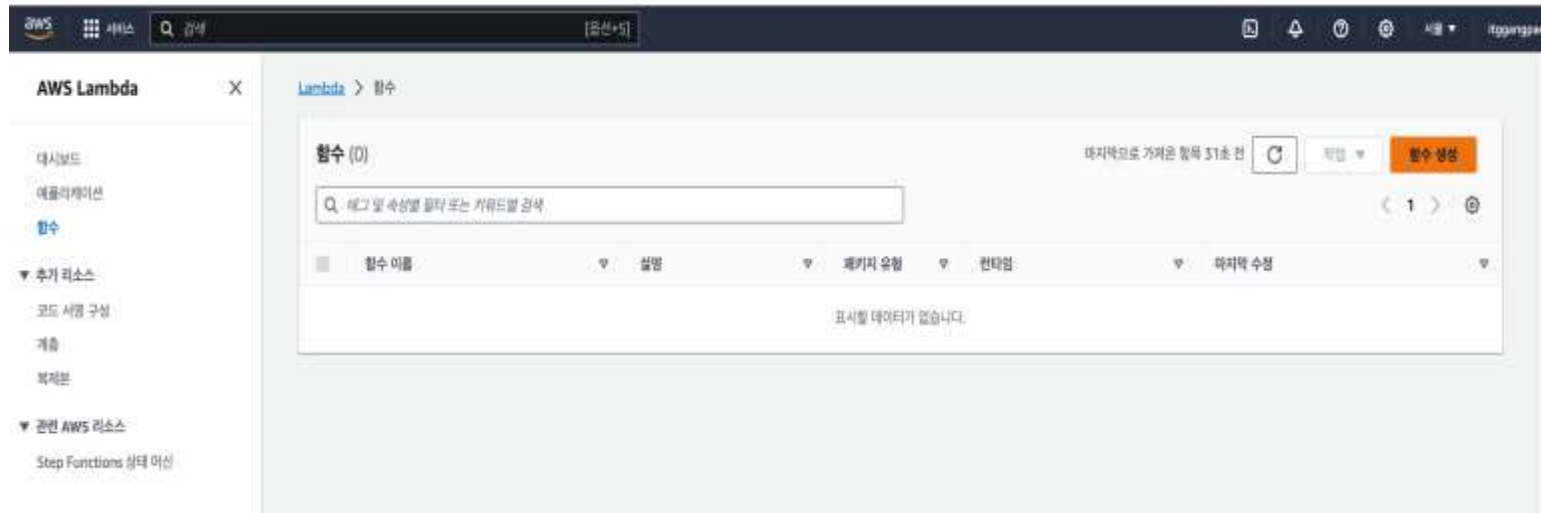
- ❖ AWS Lambda 사용 사례
 - ✓ 데이터 전처리



ServerLess

❖ 람다 함수 만들기

- ✓ 람다 콘솔에서 오른쪽 상단의 함수 생성 클릭



ServerLess

- ❖ 람다 함수 만들기
 - ✓ 생성 방법 선택



- 새로 작성: 간단한 로직이 구현된 코드를 기반으로 작성
- 블루프린트 사용: AWS에서 자주 사용되는 기능을 고려하여 개발자를 위한 블루프린트를 만들어서 제공
- 컨테이너 이미지: AWS에서 운영하는 가상 컨테이너 이미지 저장소 경로를 이용해서 배포하고자 하는 경우 사용

ServerLess

❖ 람다 함수 만들기

- ✓ 블루프린트를 선택하고 hello-world-python을 선택하고 이름 입력

기본 정보 정보

블루프린트 이름
Hello world function
A starter AWS Lambda function. python3.10

함수 이름
함수의 용도를 설명하는 이름을 입력합니다.
aws_lambda_test
공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

Runtime
python3.10

Architecture
x86_64

실행 역할
함수에 대한 권한을 정의하는 역할을 선택합니다. 사용자 지정 역할을 생성하려면 IAM 콘솔로 이동하십시오.
☒ 기본 Lambda 권한을 가진 새 역할 생성
☐ 기존 역할 사용
☐ AWS 정책 템플릿에서 새 역할 생성

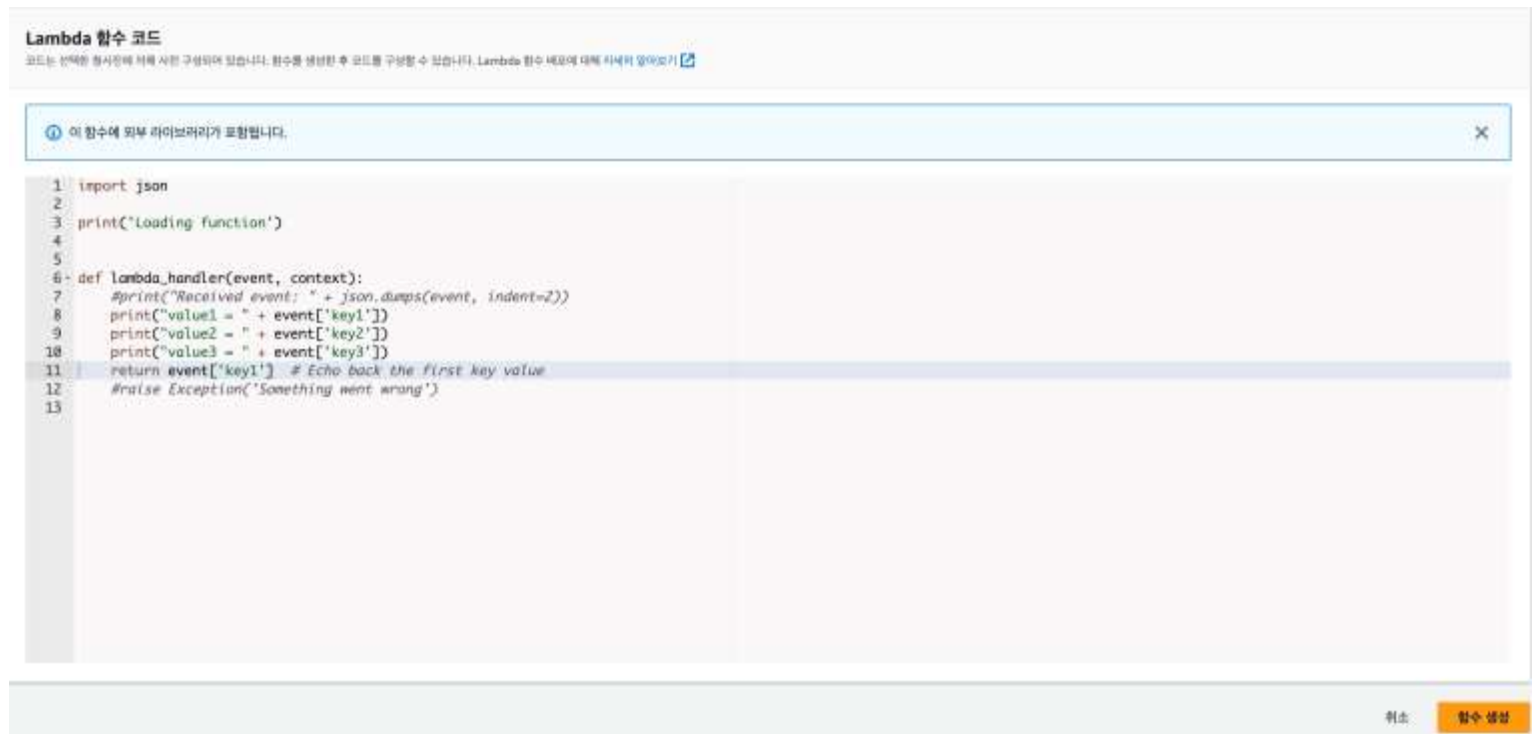
① 역할을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 역할을 삭제하거나 이 역할에서 신뢰 또는 권한 정책을 편집하지 마십시오.

Lambda가 이름이 aws_lambda_test-role-m5izvwp이고 Amazon CloudWatch Logs에 로그를 업로드할 수 있는 권한이 포함된 실행 역할을 생성합니다.

ServerLess

❖ 람다 함수 만들기

- ✓ 함수 코드를 확인하고 함수 생성을 클릭



ServerLess

❖ 람다 함수 만들기

- ✓ 코드 소스 창에서 Test를 클릭하고 옵션을 설정한 후 간접 호출 클릭

테스트 이벤트 구성

테스트 이벤트는 Lambda 함수를 호출하기 위해 AWS 서비스에서 생성하는 요청의 구조를 모방한 JSON 객체입니다. 이 객체를 사용하여 함수의 호출 결과를 확인합니다.

이벤트를 저장하지 않고 함수를 호출하려면 JSON 이벤트를 구성한 다음 테스트를 선택합니다.

이벤트 작업 테스트

새 이벤트 생성

기존 이벤트 편집

이벤트 이름

lambda_test

문자, 숫자, 밑줄, 하이픈 및 대문자를 사용하여 최대 25자로 구성합니다.

이벤트 공유 설정

☒ 프라이빗

☐ 공유 가능

이 이벤트는 Lambda 콘솔 및 이벤트 생성자만 사용할 수 있습니다. 총 10개를 구성할 수 있습니다. 자세히 알아보기

이 이벤트는 공유 가능한 이벤트에 액세스하고 이를 사용할 수 있는 권한이 있는 동일한 계층 내 IAM 사용자가 사용할 수 있습니다. 자세히 알아보기

템플릿 - 선택 사항

hello-world

이벤트 JSON

JSON 형식 지정

```
1 {
2   "key1": "소녀시대",
3   "key2": "레드벨벳",
4   "key3": "엑스파"
5 }
```

취소

간접 호출

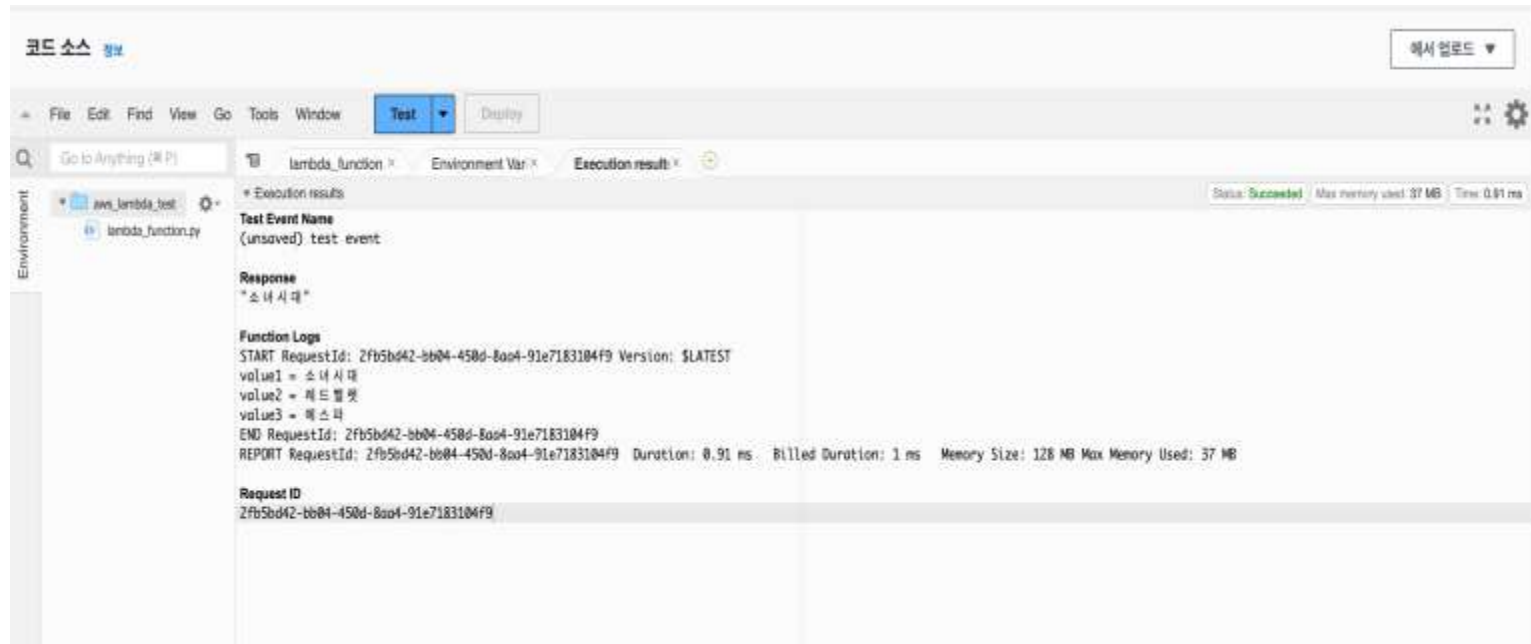
저장

46

ServerLess

❖ 람다 함수 만들기

- ✓ 코드 소스 창에서 Test를 클릭하고 옵션을 설정한 후 간접 호출 클릭

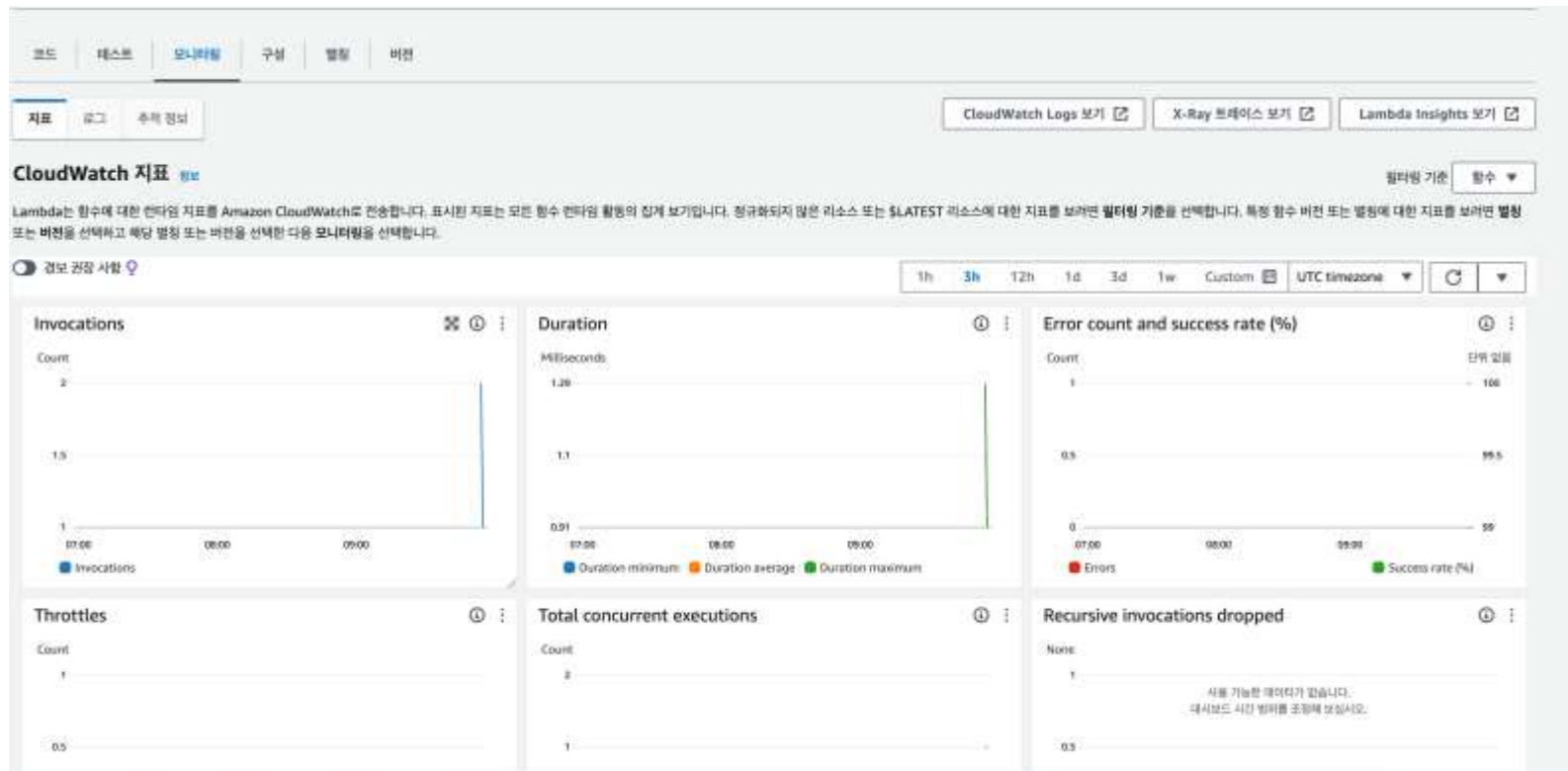


ServerLess

❖ 람다 함수 만들기

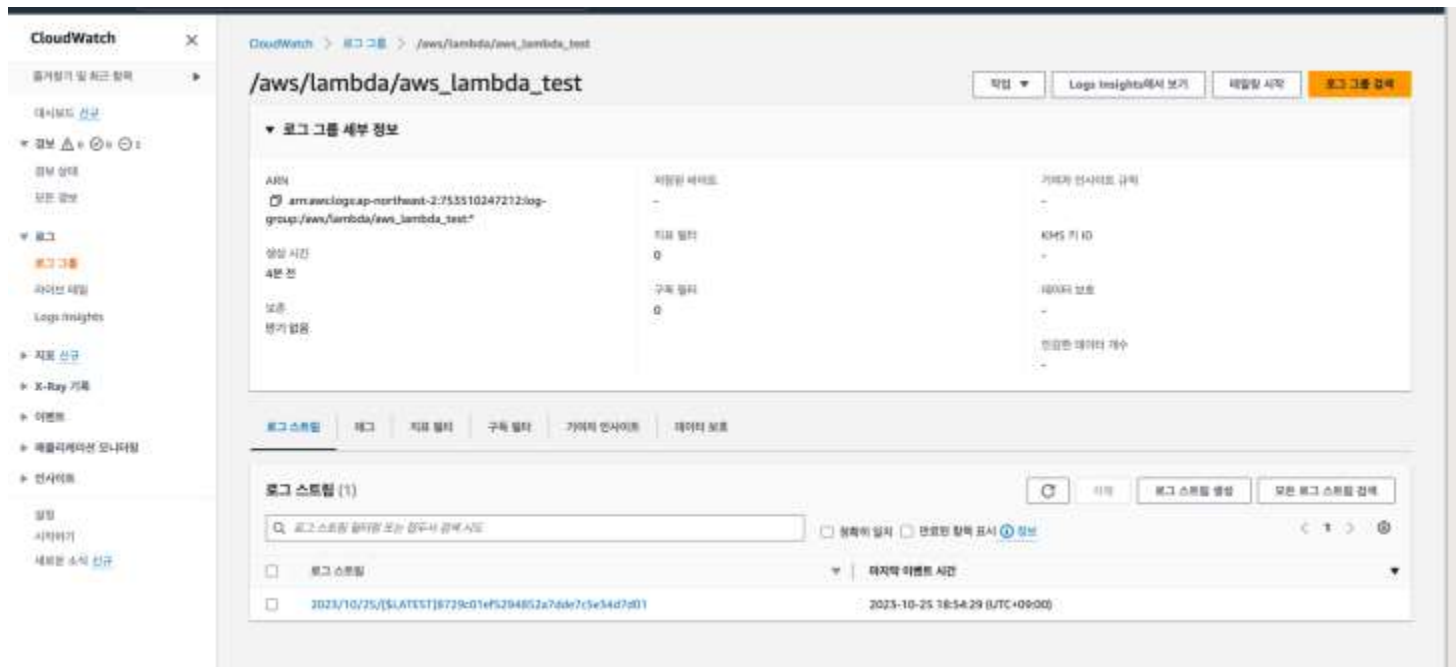
✓ 클라우드 와치에서 확인

- 모니터링 탭에서 CloudWatch에서 로그 보기를 클릭



ServerLess

- ❖ 람다 함수 만들기
 - ✓ 클라우드 와치에서 확인
 - 모니터링 탭에서 CloudWatch에서 로그 보기를 클릭



ServerLess

❖ 람다 함수가 람다 함수 호출하기

✓ 람다 함수 2개 생성

● 첫번째 함수 – lambda_a

```
import json
```

```
def lambda_handler(event, context):
```

```
    print(event)
```

```
    return event
```

ServerLess

❖ 람다 함수가 람다 함수 호출하기

✓ 람다 함수 2개 생성

● 두번째 함수 – lambda_b

```
import json
```

```
import boto3
```

```
def lambda_handler(event, context):
```

```
    payload = {}
```

```
    payload['hello'] = 'hi'
```

```
    lan = boto3.client(service_name='lambda',region_name='ap-northeast-2')
```

```
    lan.invoke(FunctionName="lambda_a", InvocationType='Event',  
Payload=json.dumps(payload))
```

```
    print(payload)
```

```
    return payload
```

ServerLess

- ❖ 람다 함수가 람다 함수 호출하기
 - ✓ lambda_b의 Test를 눌러서 실행 – 권한이 없다는 오류 발생



The screenshot displays the AWS Lambda console's 'Execution results' tab for a function named 'lambda_function'. The status is 'Failed' with a message: 'Max memory used: 75 MB Time: 1439.05 ms'. The 'Test Event Name' is '(unsaved) test event'. The 'Response' section shows a JSON object with an 'errorMessage' indicating an 'AccessDeniedException' when calling the 'Invoke' operation. The 'Function Logs' section shows a detailed traceback of the error, starting from the 'lambda_handler' file and moving through the 'botocore/client.py' file. The logs also include performance metrics: 'RequestId: a02e929c-3ded-4120-81c1-fae429c46501', 'Duration: 1439.05 ms', 'Billed Duration: 1440 ms', 'Memory Size: 128 MB', 'Max Memory Used: 75 MB', and 'Init Duration: 251.03 ms'.

```
Execution results
Status: Failed Max memory used: 75 MB Time: 1439.05 ms

Test Event Name
(unsaved) test event

Response
{
  "errorMessage": "An error occurred (AccessDeniedException) when calling the Invoke operation: User: arn:aws:sts::753510247212:assumed-role/lambda_b-role-yzxp690/lambda_b is not authorize",
  "errorType": "ClientError",
  "requestId": "a02e929c-3ded-4120-81c1-fae429c46501",
  "stackTrace": [
    "File \"/var/task/lambda_function.py", line 8, in lambda_handler\n  lan.invoke(FunctionName=\"lambda_a\", InvocationType='Event', Payload=json.dumps(payload))\n",
    "File \"/var/lang/lib/python3.11/site-packages/botocore/client.py", line 534, in _api_call\n  return self._make_api_call(operation_name, kwargs)\n",
    "File \"/var/lang/lib/python3.11/site-packages/botocore/client.py", line 976, in _make_api_call\n  raise error_class(parsed_response, operation_name)\n"
  ]
}

Function Logs
START RequestId: a02e929c-3ded-4120-81c1-fae429c46501 Version: $LATEST
[ERROR] ClientError: An error occurred (AccessDeniedException) when calling the Invoke operation: User: arn:aws:sts::753510247212:assumed-role/lambda_b-role-yzxp690/lambda_b is not authorize
Traceback (most recent call last):
  File "/var/task/lambda_function.py", line 8, in lambda_handler
    lan.invoke(FunctionName="lambda_a", InvocationType="Event", Payload=json.dumps(payload))
  File "/var/lang/lib/python3.11/site-packages/botocore/client.py", line 534, in _api_call
    return self._make_api_call(operation_name, kwargs)
  File "/var/lang/lib/python3.11/site-packages/botocore/client.py", line 976, in _make_api_call
    raise error_class(parsed_response, operation_name)END RequestId: a02e929c-3ded-4120-81c1-fae429c46501
REPORT RequestId: a02e929c-3ded-4120-81c1-fae429c46501 Duration: 1439.05 ms Billed Duration: 1440 ms Memory Size: 128 MB Max Memory Used: 75 MB Init Duration: 251.03 ms

Request ID
a02e929c-3ded-4120-81c1-fae429c46501
```

ServerLess

❖ 람다 함수가 람다 함수 호출하기

- ✓ lambda_b에 람다 권한 부여: [구성] - [권한]에서 실행 역할에서 이름을 클릭

실행 역할

↻ 편집 역할 문서 보기

역할 이름
[lambda_b-role-yzxp690](#)

리소스 요약

함수에 액세스 권한이 있는 리소스 및 작업을 보려면 서비스를 선택합니다.

 Amazon CloudWatch Logs
3 actions, 2 resources

작업별

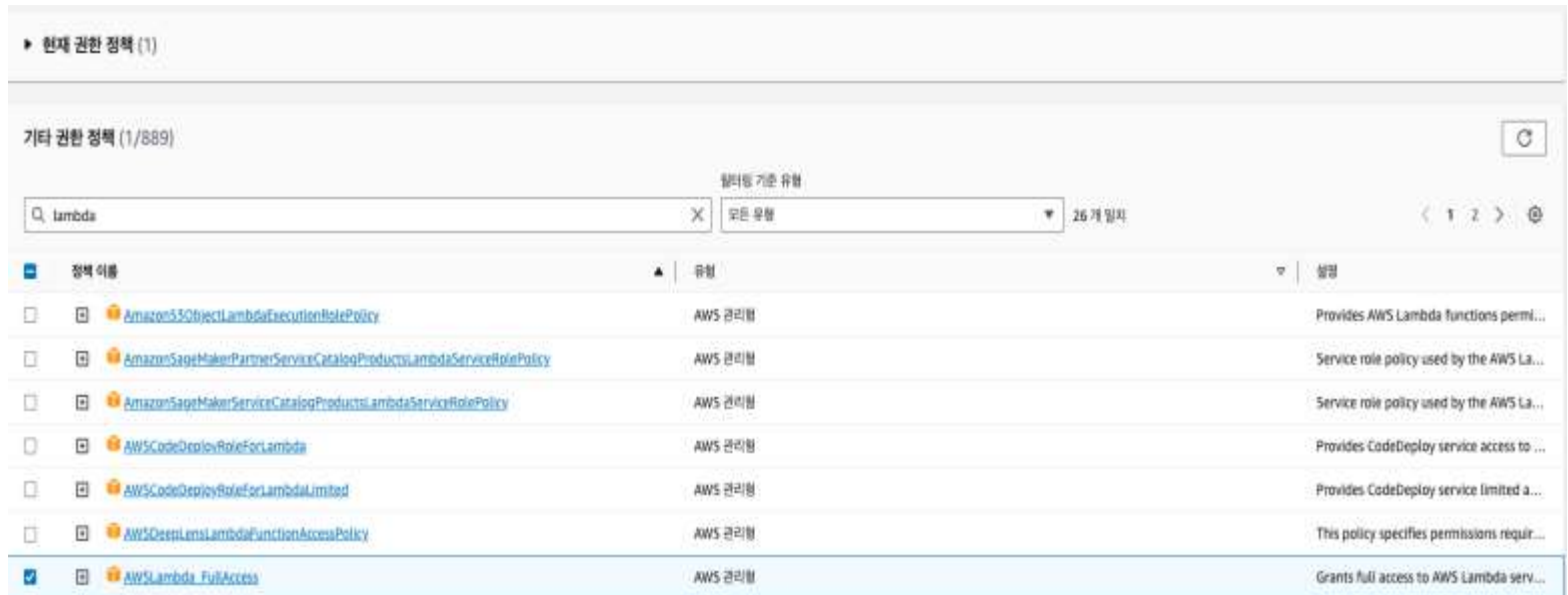
리소스별

리소스	작업
arn:aws:logs:ap-northeast-2:753510247212:*	Allow: logs:CreateLogGroup
arn:aws:logs:ap-northeast-2:753510247212:log-group:/aws/lambda/lambda_b:*	Allow: logs:CreateLogStream Allow: logs:PutLogEvents

ServerLess

❖ 람다 함수가 람다 함수 호출하기

- ✓ lambda_b에 람다 권한 부여: [권한 추가]에서 정책 연결 클릭 후 람다 권한 추가



ServerLess

- ❖ 람다 함수가 람다 함수 호출하기
 - ✓ lambda_b에 다시 테스트 수행

The screenshot displays the AWS Lambda console's 'Execution results' tab for a function named 'lambda_function'. The status is 'Succeeded', with a maximum memory usage of 75 MB and an execution time of 1398.28 ms. The 'Test Event Name' is '(unsaved) test event'. The 'Response' is a JSON object: `{ "hello": "hi" }`. The 'Function Logs' section shows the following details:
START RequestId: 45541919-99d3-4637-b9aa-78e6a6986568 Version: \$LATEST
{'hello': 'hi'}
END RequestId: 45541919-99d3-4637-b9aa-78e6a6986568
REPORT RequestId: 45541919-99d3-4637-b9aa-78e6a6986568 Duration: 1398.28 ms Billed Duration: 1399 ms Memory Size: 128 MB Max Memory Used: 75 MB Init Duration: 285.62 ms
The 'Request ID' is 45541919-99d3-4637-b9aa-78e6a6986568.

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 파이썬에서 실행되는 함수 생성

Lambda > 함수 > 함수 생성

함수 생성

AWS Serverless Application Repository 애플리케이션을 애플리케이션 템플릿으로 이용하실 수 있습니다.

☒ 새로 작성
간단한 Hello World 예제는 시작하십시오.

☐ 블루프린트 사용
템플릿 코드 및 구조를 Lambda 애플리케이션을 위한 구현 사례를 포함한 일련적인 사용 사례를 제공합니다.

☐ 템플릿에서 이미지
함수에 대해 필요한 컨테이너 이미지를 선택합니다.

기본 정보

함수 이름
함수의 이름을 설정하는 이름을 입력합니다.

공백 없이 문자, 숫자, 밑줄 또는 마침표만 사용하십시오.

런타임 정보
함수를 실행하는 데 사용할 언어를 선택합니다. 코딩 언어는 Node.js, Python 및 Ruby만 지원됩니다.

아키텍처 정보
함수 코드에 대해 함수는 단일 또는 여러 아키텍처를 선택합니다.
☒ x86_64
☐ arm64

권한 정보
기본적으로 Lambda는 Amazon CloudWatch Logs에 로그를 기록하는 권한을 가진 기본 역할을 생성합니다. 이 기본 역할은 사설망 관리자 권한을 추가할 때 사용될 수 있습니다.

▶ 기본 실행 역할 변경

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기

- ✓ 코드 작성 후 deploy

```
import json
import boto3
from datetime import datetime
```

```
client = boto3.client('s3')
```

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기

- ✓ 코드 작성 후 deploy

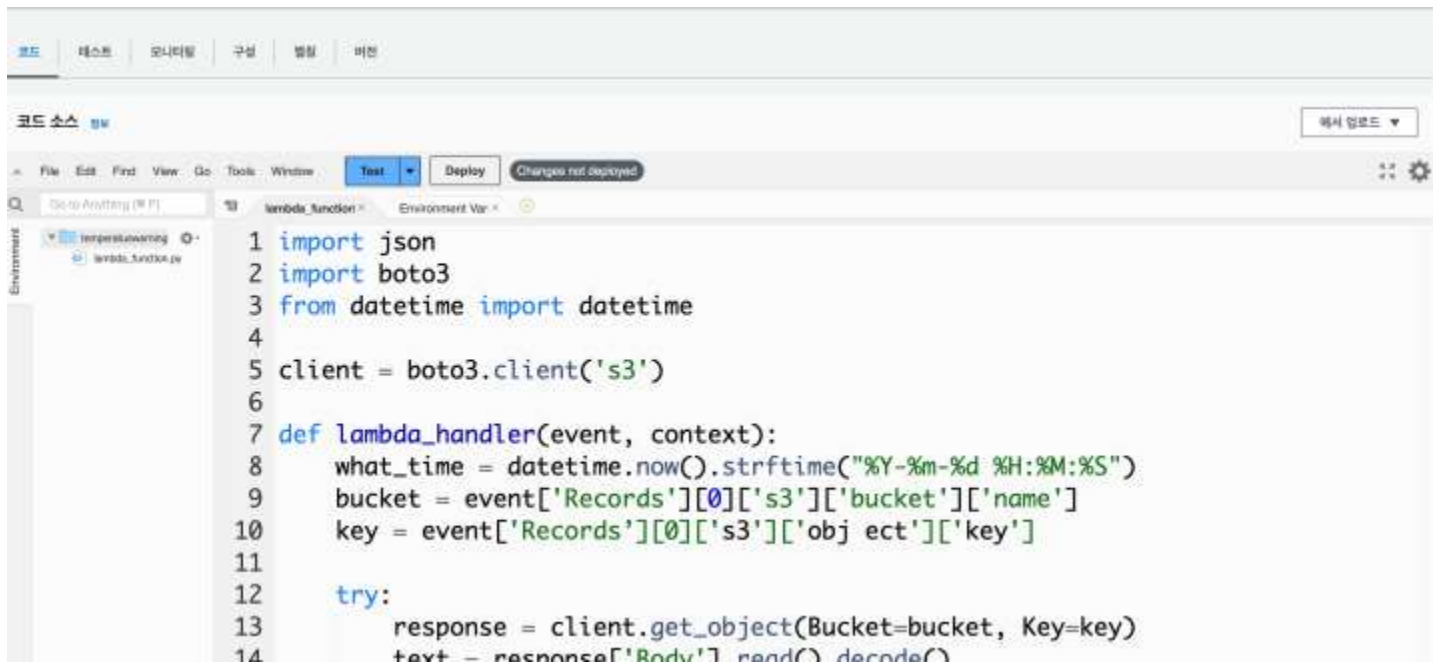
```
def lambda_handler(event, context):
    what_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']

    try:
        response = client.get_object(Bucket=bucket, Key=key)
        text = response['Body'].read().decode()
        data = json.loads(text)

        if data['temperature'] > 40:
            print(f" {data['temperature']} at {what_time}")
            print("주의하세요 매우 덥습니다.")
        else:
            print("나쁘지 않은 날씨")
    except Exception as e:
        print(e)
        raise e
```

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 코드 작성 후 deploy



The screenshot shows the AWS Lambda console interface. The top navigation bar includes tabs for '코드' (Code), '테스트' (Test), '모니터링' (Monitoring), '구성' (Configuration), '발행' (Publish), and '버전' (Versions). Below the navigation bar, the '코드 소스' (Code Source) tab is selected, showing the code for the 'lambda_function'. The code is written in Python and is as follows:

```
1 import json
2 import boto3
3 from datetime import datetime
4
5 client = boto3.client('s3')
6
7 def lambda_handler(event, context):
8     what_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
9     bucket = event['Records'][0]['s3']['bucket']['name']
10    key = event['Records'][0]['s3']['object']['key']
11
12    try:
13        response = client.get_object(Bucket=bucket, Key=key)
14        text = response['Body'].read().decode()
```

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ S3 버킷 생성 후 [권한] - [버킷 권한 설정]에서 버킷 정책 추가

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicListGet",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:List*",
        "s3:Get*",
        "s3:Put*",
        "s3:Delete*"
      ],
      "Resource": [
        "arn:aws:s3:::adamsoft",
        "arn:aws:s3:::adamsoft/*"
      ]
    }
  ]
}
```

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 버킷에서 속성 탭을 열어서 이벤트 알림 생성을 클릭

이벤트 알림 (0)

버킷에서 특정 이벤트가 발생하면 알림을 보냅니다. [자세히 알아보기](#)

이름	이벤트 유형	필터	대상 유형	대상
이벤트 알림 없음				

특정 이벤트가 발생할 때 알림을 받으려면 [이벤트 알림 생성]을 선택합니다.

[이벤트 알림 생성](#)

Amazon EventBridge

추가 기능을 사용하려면 Amazon EventBridge를 통해 S3 이벤트 알림을 사용하여 대규모로 이벤트 중심 애플리케이션을 구축합니다. [자세히 알아보기](#) [EventBridge 모형을 참조하세요](#)

이 버킷의 모든 이벤트에 대해 Amazon EventBridge로 알림 전송

비용성

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 이벤트 설정

이벤트 알림 생성 정보

알림을 활성화하려면 먼저 Amazon S3에서 게시할 이벤트와 Amazon S3에서 알림을 전송할 대상을 식별하는 알림 구성을 추가해야 합니다.

일반 구성

이벤트 이름

aws_temperature_event

이벤트 이름은 255자 이내여야 합니다.

접두사 - 선택 사항

지정된 문자로 시작하는 키가 있는 객체로 알림을 제한합니다.

images/

접미사 - 선택 사항

지정된 문자로 끝나는 키가 있는 객체로 알림을 제한합니다.

.json

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 이벤트 설정

이벤트 유형
말씀을 수신할 이벤트를 하나 이상 지정합니다. 각 그룹에 대해 모든 이벤트에 적용되는 이벤트 유형을 선택하거나 하나 이상의 개별 이벤트를 선택할 수 있습니다.

객체 생성

- ☒ 모든 객체 생성 이벤트
s3:ObjectCreated:*
 - ☐ 전송
s3:ObjectCreated:Put
 - ☐ 게시
s3:ObjectCreated:Post
 - ☐ 복사
s3:ObjectCreated:Copy
 - ☐ 멀티파트 업로드 완료
s3:ObjectCreated:CompleteMultipartUpload

객체 제거

- ☐ 모든 객체 제거 이벤트
s3:ObjectRemoved:*
 - ☐ 영구 삭제됨
s3:ObjectRemoved:Delete
 - ☐ 삭제 마커 생성됨
s3:ObjectRemoved:DeleteMarkerCreated

객체 복원

- ☐ 모든 객체 복원 이벤트
s3:ObjectRestore:*
 - ☐ 복원 시작됨
s3:ObjectRestore:Post
 - ☐ 복원 완료됨
s3:ObjectRestore:Completed

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 이벤트 설정

대상

Amazon S3가 대상에 메시지를 게시하려면, 먼저 관련 API를 호출하여 SNS 주제, SQS 대기열 또는 Lambda 함수에 메시지를 게시하는 데 필요한 권한을 Amazon S3 보안 주제에 부여해야 합니다. [자세히 알아보기](#)

대상
이벤트를 게시할 대상을 선택합니다. [자세히 알아보기](#)

☒ **Lambda 함수**
S3 이벤트를 기반으로 Lambda 함수 스크립트를 실행합니다.

☐ **SNS 주제**
병렬 처리를 위해 시스템으로 메시지를 편아웃하거나 사람에게 직접 메시지를 편아웃합니다.

☐ **SQS 대기열**
서버에서 읽을 수 있도록 SQS 대기열로 알림을 전송합니다.

Lambda 함수 지정

☒ **Lambda 함수에서 선택**

☐ **Lambda 함수 ARN 입력**

Lambda 함수
temperatuewarning ▼

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 이벤트 설정

The screenshot shows the AWS IAM console interface for configuring an event. At the top, there's a section titled "이벤트 알림 (1)" with buttons for "편집", "삭제", and "이벤트 알림 생성". Below this is a table with columns: "이름", "이벤트 유형", "필터", "대상 유형", and "대상". The table contains one entry: "aws_temperature_event", "모든 객체 생성 이벤트", ".json", "Lambda 함수", and "temperaturewarning". Below the table, there's a section titled "Amazon EventBridge" with a "편집" button. The text below this section says: "추가 기능을 사용하려면 Amazon EventBridge를 통해 S3 이벤트 알림을 사용하여 대규모로 이벤트 중심 애플리케이션을 구축합니다. 자세한 알아보기 > EventBridge 요금을 참조하세요 >". At the bottom, there's a note: "이 버킷의 모든 이벤트에 대해 Amazon EventBridge로 알림 전송 비활성".

이름	이벤트 유형	필터	대상 유형	대상
aws_temperature_event	모든 객체 생성 이벤트	.json	Lambda 함수	temperaturewarning

Amazon EventBridge

추가 기능을 사용하려면 Amazon EventBridge를 통해 S3 이벤트 알림을 사용하여 대규모로 이벤트 중심 애플리케이션을 구축합니다. [자세히 알아보기](#) > [EventBridge 요금을 참조하세요](#) >

이 버킷의 모든 이벤트에 대해 Amazon EventBridge로 알림 전송 비활성

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기

- ✓ 테스트

- json 파일을 만들어서 업로드

```
{  
    "temperature":45  
}
```

ServerLess

- ❖ S3 버킷에 확장자가 json 데이터가 업로드되면 PutObject 이벤트가 발생하게 되고 이 이벤트가 발생했을 때 데이터 중에서 temperature 값을 읽어서 그 시간 과 메시지를 출력하도록 하기
 - ✓ 테스트
 - json 파일을 만들어서 업로드
 - ◆ Cloud Watch에서 로그 확인

로그 이벤트

아래의 필터 맥대를 사용하여 로그 이벤트의 용어, 구문 또는 값을 검색하고 매칭할 수 있습니다. [필터 패턴에 대해 자세히 알아보기](#)

이벤트 필터링

Clear 1m 30m 1h 12h Custom Local timezone 디스플레이

타임스탬프	메시지
현재 이전 이벤트가 없습니다. 재시도	
2023-10-25T19:38:04.754+09:00	START RequestId: 4143c5c1-b4cc-4128-ac45-e43300d7958a Version: \$LATEST
2023-10-25T19:38:04.934+09:00	45 at 2023-10-25 18:38:04
2023-10-25T19:38:04.934+09:00	주의하세요! 패우입니다.
2023-10-25T19:38:04.940+09:00	END RequestId: 4143c5c1-b4cc-4128-ac45-e43300d7958a
2023-10-25T19:38:04.940+09:00	REPORT RequestId: 4143c5c1-b4cc-4128-ac45-e43300d7958a Duration: 186.12 ms Billed Duration: 187 ms Memory Size: 128 MB Max Memory Used: 77 MB
현재 최신 이벤트가 없습니다. 자동 재시도를 일시 중지했습니다 재개	

ServerLess

❖ Lambda에서 Amazon SNS 실행

✓ 람다 함수 생성

● 메시지를 전송하는 함수 – sns_a

```
import json
import boto3
```

```
def lambda_handler(event, context):
    client = boto3.client(service_name='sns')
    response = client.publish(
        TargetArn = '', #'arn:aws:sns:ap-northeast-2:1234567890:ab_sns',
        Message = 'hello',
        Subject = 'sns_a lambda',
        MessageStructure='text'
    )
    print(response)
    return response
```

ServerLess

- ❖ Lambda에서 Amazon SNS 실행
 - ✓ SNS 주제 생성
 - 주제 설정

주제 생성

세부 정보

유형 [정보](#)
주제를 생성한 후에는 주제 유형을 수정할 수 없음

☐ FIFO(선입선출)

- 엄격하게 보존된 메시지 순서 지정
- 정확히 1회 메시지 전송
- 높은 처리량, 초당 최대 300회 게시
- 구독 프로토콜: SQS

☒ 표준

- 최선의 메시지 순서 지정
- 최소 1회 메시지 전송
- 가장 높은 처리량(초당 게시 횟수)
- 구독 프로토콜: SQS, Lambda, HTTP, SMS, 이메일, 모바일 애플리케이션 엔드포인트

이름

ab_sns

최대 256자이며 영숫자, 하이픈(-) 및 밑줄(_)을 포함할 수 있습니다.

표시 이름 - 선택 사항 [정보](#)
이 주제를 SMS 구독과 함께 사용하려면 표시 이름을 입력하십시오. 처음 10자만 SMS 메시지에 표시됩니다.

EmailSend

최대 100자.

ServerLess

❖ Lambda에서 Amazon SNS 실행

✓ SNS 주제 생성

● 구독 생성

주제 생성

세부 정보

유형 **정보**

주제를 생성한 후에는 주제 유형을 수정할 수 없음

☐ FIFO(선입선출)

- 엄격하게 보존된 메시지 순서 지정
- 정확히 1회 메시지 전송
- 높은 처리량, 초당 최대 300회 게시
- 구독 프로토콜: SQS

☒ 표준

- 최선의 메시지 순서 지정
- 최소 1회 메시지 전송
- 가장 높은 처리량(초당 게시 횟수)
- 구독 프로토콜: SQS, Lambda, HTTP, SMS, 이메일, 모바일 메
몰리케이션 엔드포인트

이름

ab_sns

최대 256자이며 영숫자, 하이픈(-) 및 밑줄(_)를 포함할 수 있습니다.

표시 이름 - 선택 사항 **정보**

이 주제를 SMS 구독과 함께 사용하려면 표시 이름을 입력하십시오. 처음 10자만 SMS 메시지에 표시됩니다.

EmailSend

최대 100자.

ServerLess

❖ Lambda에서 Amazon SNS 실행

✓ SNS 주제 생성

- 수신을 받는 메일에서 확인
- 구독 생성 – ARN 복사

구독: d6eb3c87-7ac4-4d1d-8f54-aceb8b88b78a 편집 삭제

세부 정보

ARN	상대
arn:aws:sns:ap-northeast-2:753510247212:ab_sns:d6eb3c87-7ac4-4d1d-8f54-aceb8b88b78a	확인 대기 중
엔드포인트	프로토콜
ltstudy@kakao.com	EMAIL
주제	
ab_sns	
구독 보안 주제	
arn:aws:iam::753510247212:root	

ServerLess

❖ Lambda에서 Amazon SNS 실행

✓ SNS 주제 생성

● sns_a 함수 수정

```
import json
```

```
import boto3
```

```
def lambda_handler(event, context):
```

```
    client = boto3.client(service_name='sns')
```

```
    response = client.publish(
```

```
        TargetArn = 'arn:aws:sns:ap-northeast-2:753510247212:ab_sns',
```

```
        #'arn:aws:sns:ap-northeast-2:1234567890:ab_sns',
```

```
        Message = 'hello',
```

```
        Subject = 'sns_a lambda',
```

```
        MessageStructure='text'
```

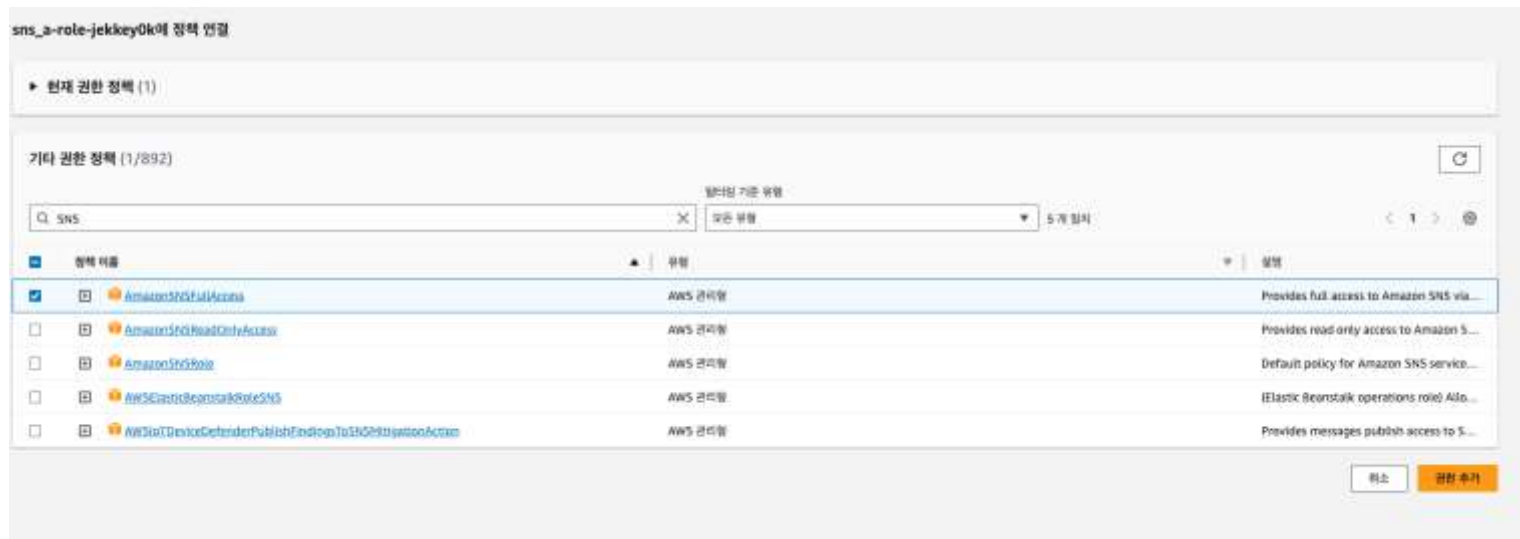
```
    )
```

```
    print(response)
```

```
    return response
```


ServerLess

- ❖ Lambda에서 Amazon SNS 실행
 - ✓ SNS 주제 생성
 - sns_a 함수에 SNS 권한 부여



ServerLess

- ❖ Lambda에서 Amazon SNS 실행
 - ✓ 함수를 호출한 후 이메일 과 로그 확인

로그 이벤트
아래의 필터 막대를 사용하여 로그 이벤트의 용어, 구문 또는 값을 검색하고 매칭할 수 있습니다. [필터 제한에 대해 자세히 알아보기](#)

이벤트 필터링

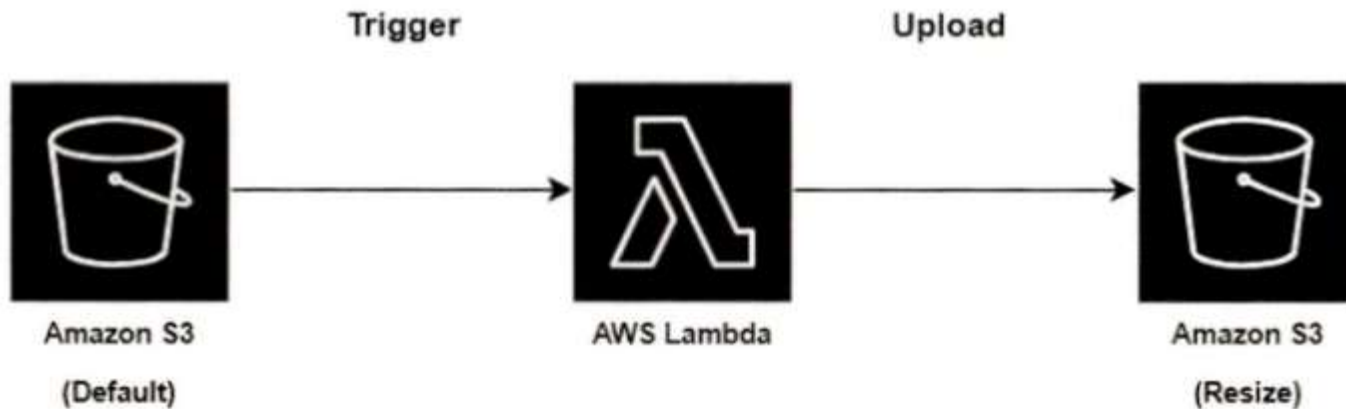
Clear 1m 30m 1h 12h Custom Local timezone 디스플레이

타임스탬프	메시지
	현재 이전 이벤트가 없습니다. 재시도
2023-10-26T16:36:26.777+09:00	INIT_START Runtime Version: python:3.11.v16 Runtime Version ARN: arn:aws:lambda:ap-northeast-2::runtime:6cf63f1a78b5c5e19617d6b4b111370fdbd0415ea91bdfdc5aacef9fee76b64a
2023-10-26T16:36:27.032+09:00	START RequestId: 831ba06c-42fc-4fb9-bb3c-e560473c7f0e Version: \$LATEST
2023-10-26T16:36:28.384+09:00	{'MessageId': 'c7171361-b05d-517e-8cb9-4ed5a9610163', 'ResponseMetadata': {'RequestId': 'a60fe770-7dad-515a-8d27-bac46491c147', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-...}}
2023-10-26T16:36:28.403+09:00	END RequestId: 831ba06c-42fc-4fb9-bb3c-e560473c7f0e
2023-10-26T16:36:28.403+09:00	REPORT RequestId: 831ba06c-42fc-4fb9-bb3c-e560473c7f0e Duration: 1369.48 ms Billed Duration: 1370 ms Memory Size: 128 MB Max Memory Used: 74 MB Init Duration: 253.75 ms
	현재 최신 이벤트가 없습니다. 자동 재시도를 일시 중지했습니다. 재개

ServerLess

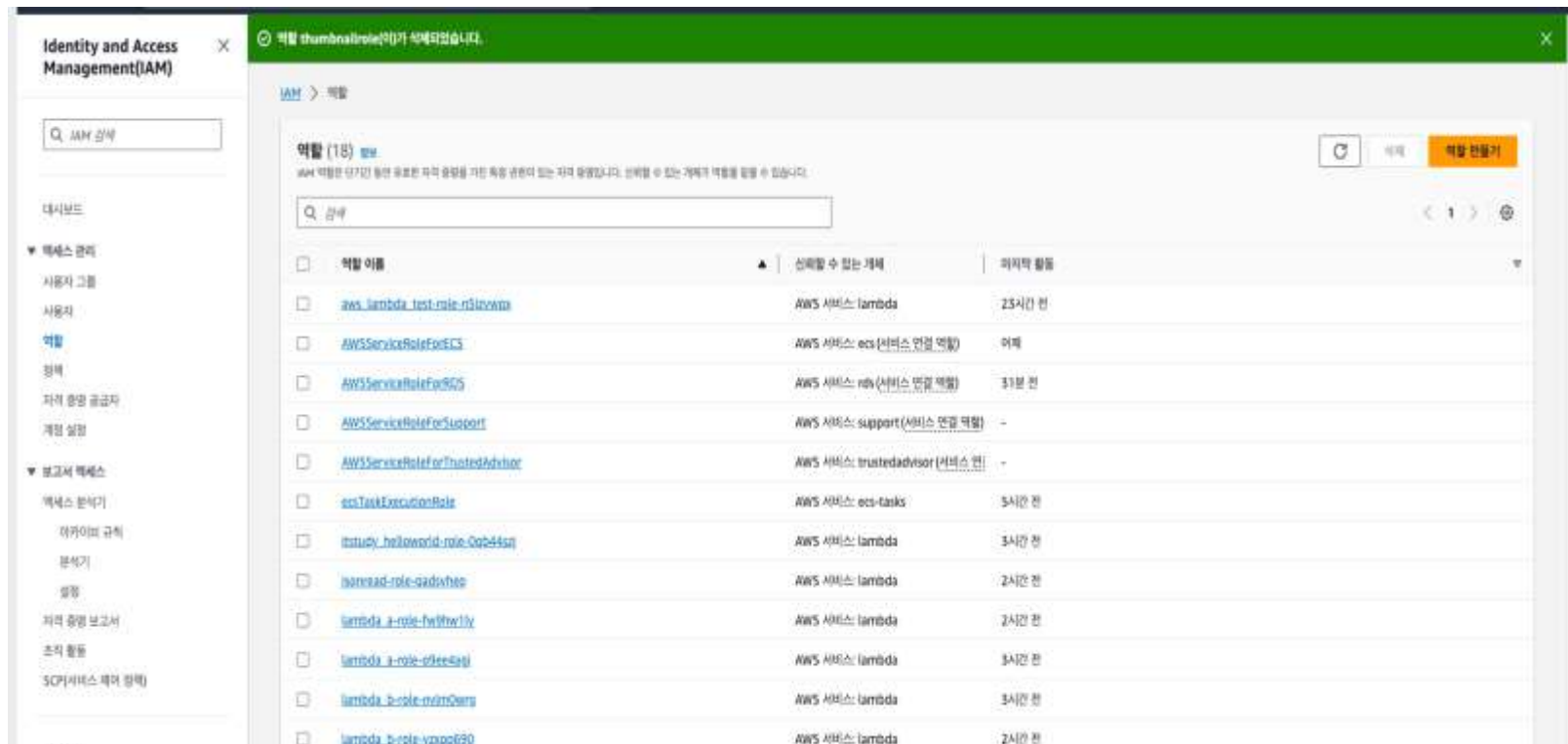
❖ 썸네일 이미지 만들기

- ✓ 버킷에 이미지를 업로드하거나 삭제하면 Amazon S3가 AWS Lambda 에 Trigger 하여 다른 S3 버킷에 Thumbnail 이미지를 생성해서 업로드



ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 역할 생성 – S3 와 Lambda 실행 권한 소유
 - IAM에서 [엑세스 관리] – [역할] – [역할 만들기]



ServerLess

❖ 썸네일 이미지 만들기

✓ 역할 생성 – S3 와 Lambda 실행 권한 소유

- IAM에서 [엑세스 관리] – [역할] – [역할 만들기]

신뢰할 수 있는 엔터티 선택 정보

신뢰할 수 있는 엔터티 유형

☒ AWS 서비스 EC2, Lambda 등의 AWS 서비스가 이 계정에서 작업을 수행하도록 허용합니다.	☐ AWS 계정 사용자 또는 세드 코디에 속한 다른 AWS 계정의 엔터티가 이 계정에서 작업을 수행하도록 허용합니다.	☐ 앱 자격 증명 지정된 외부 앱 자격 증명 공급자와 연동된 사용자의 이 역할을 맡아 이 계정에서 작업을 수행하도록 허용합니다.
☐ SAML 2.0 연동 기타 디렉터리에서 SAML 2.0과 연동된 사용자가 이 계정에 이 작업을 수행할 수 있도록 허용합니다.	☐ 사용자 지정 신뢰 정책 다른 사용자가 이 계정에서 작업을 수행할 수 있도록 사용자 지정 신뢰 정책을 생성합니다.	

사용 사례
EC2, Lambda 등의 AWS 서비스가 이 계정에서 작업을 수행하도록 허용합니다.

서비스 또는 사용 사례

Lambda ▼

지정된 서비스에 대한 사용 사례를 선택합니다.

사용 사례

☒ **Lambda**
Allows Lambda functions to call AWS services on your behalf.

취소 **다음**

ServerLess

❖ 썸네일 이미지 만들기

✓ 역할 생성 – S3 와 Lambda 실행 권한 소유

- IAM에서 [엑세스 관리] – [역할] – [역할 만들기]

역할 이름

이 역할을 식별하는 의미 있는 이름을 입력합니다.

thumbnailrole

최대 64자입니다. 영숫자 및 '+', '@', '_' 문자를 사용하세요.

설명

이 역할에 대하여 간단한 설명을 추가합니다.

Allows Lambda functions to call AWS services on your behalf.

최대 1000자입니다. 영숫자 및 '+', '@', '_' 문자를 사용하세요.

ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 람다 함수 생성

ServerLess

❖ 썸네일 이미지 만들기

✓ 역할 추가

- IAM에서 [엑세스 관리] - [역할] - 생성된 람다 함수에 S3 와 Lambda 실행 권한 추가

2단계: 권한 추가 편집

권한 정책 요약

정책 이름	유형	다음으로 연결됨
AmazonS3FullAccess	AWS 관리형	권한 정책
AWSLambda_FullAccess	AWS 관리형	권한 정책

3단계: 태그 추가

태그 추가 - 선택 사항 [도움](#)

태그는 리소스를 식별, 정리 또는 검색하는 데 도움이 되도록 AWS 리소스에 추가할 수 있는 키-값 레이블입니다.

리소스와 연결된 태그가 없습니다.

새 태그 추가

최대 50개의 태그를 더 추가할 수 있습니다.

취소 이전 다음 생성

ServerLess

❖ 썸네일 이미지 만들기

- ✓ 일반 버킷 생성 - 속성 탭에서 객체가 생성되거나 소멸될 때 람다 함수가 수행되도록 설정

일반 구성

이벤트 이름

itstudythumbnail

이벤트 이름은 255자 이내여야 합니다.

접두사 - 선택 사항

지정된 문자로 시작하는 키가 있는 객체로 알림을 제한합니다.

images/

접미사 - 선택 사항

지정된 문자로 끝나는 키가 있는 객체로 알림을 제한합니다.

.jpg

ServerLess

❖ 썸네일 이미지 만들기

- ✓ 일반 버킷 생성 – 속성 탭에서 객체가 생성되거나 소멸될 때 람다 함수가 수행되도록 설정

이벤트 유형

알림을 수신할 이벤트를 하나 이상 지정합니다. 각 그룹에 대해 모든 이벤트에 적용되는 이벤트 유형을 선택하거나 하나 이상의 개별 이벤트를 선택할 수 있습니다.

객체 생성

☒ 모든 객체 생성 이벤트
s3:ObjectCreated:*

☐ 전송
s3:ObjectCreated:Put

☐ 게시
s3:ObjectCreated:Post

☐ 복사
s3:ObjectCreated:Copy

☐ 멀티파트 업로드 완료
s3:ObjectCreated:CompleteMultipartUpload

객체 제거

☒ 모든 객체 제거 이벤트
s3:ObjectRemoved:*

☐ 영구 삭제됨
s3:ObjectRemoved:Delete

☐ 삭제 마커 생성됨
s3:ObjectRemoved:DeleteMarkerCreated

객체 복원

☐ 모든 객체 복원 이벤트
s3:ObjectRestore:*

☐ 복원 시작됨
s3:ObjectRestore:Post

☐ 복원 완료됨
s3:ObjectRestore:Completed

ServerLess

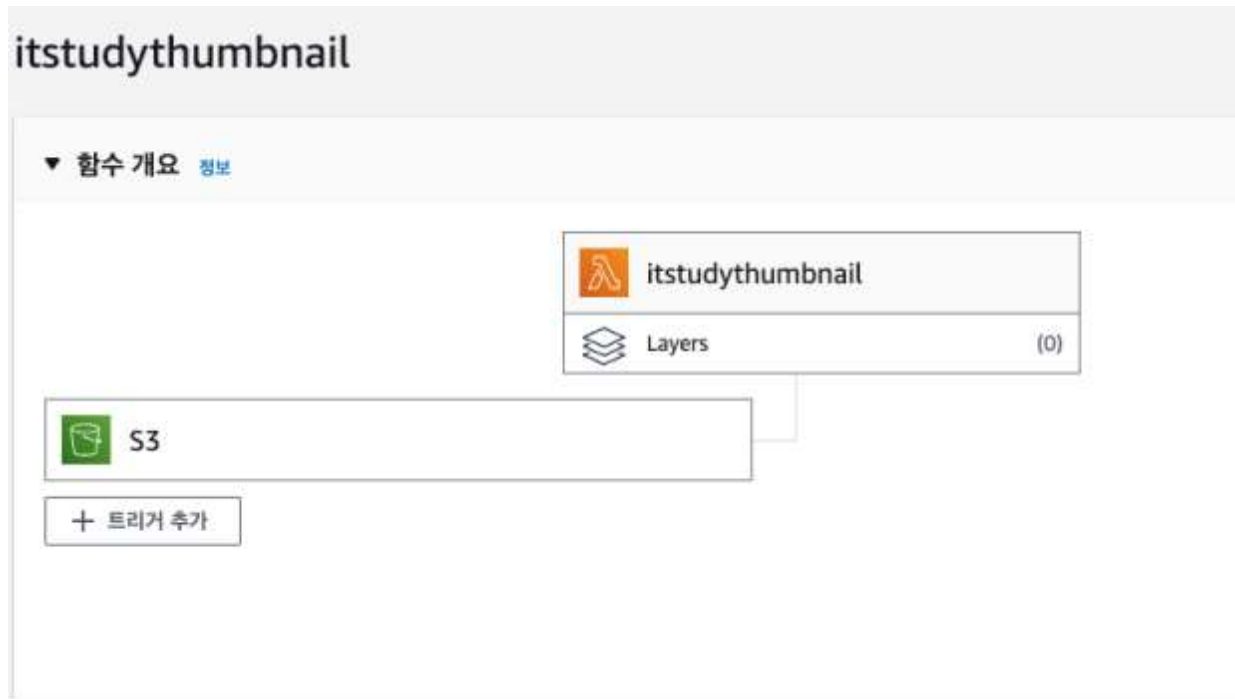
❖ 썸네일 이미지 만들기

- ✓ 일반 버킷 생성 – 속성 탭에서 객체가 생성되거나 소멸될 때 람다 함수가 수행되도록 설정



ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 람다 함수 확인



ServerLess

❖ 썸네일 이미지 만들기

- ✓ 이미지를 업로드해서 람다 함수를 실행하고 CloudWatch에서 로그 확인

로그 이벤트
아래의 필터 박스를 사용하여 로그 이벤트의 필터, 구분 또는 값을 검색하고 매칭할 수 있습니다. [필터 패턴에 대해 자세히 알아보기](#)

🔍 이벤트 필터링

Clear 1m 30m 1h 12h Custom Local timezone 디스플레이

▶ 타임스탬프	메시지
	현재 이전 이벤트가 없습니다. 재시도
▶ 2023-10-26T19:15:04.580+09:00	INIT_START Runtime Version: python:3.8.v31 Runtime Version ARN: arn:aws:lambda:ap-northeast-2::runtime:85a1944859083528568a8398e50ec22be94bdaefef94fa061158bf4530956
▶ 2023-10-26T19:15:04.700+09:00	START RequestId: 7bdb667-6ab6-4335-bc5b-20487e5bbd6d Version: \$LATEST
▶ 2023-10-26T19:15:04.702+09:00	END RequestId: 7bdb667-6ab6-4335-bc5b-20487e5bbd6d
▶ 2023-10-26T19:15:04.702+09:00	REPORT RequestId: 7bdb667-6ab6-4335-bc5b-20487e5bbd6d Duration: 1.48 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 38 MB Init Duration: 119.31 ms
	현재 최신 이벤트가 없습니다. 자동 재시도를 일시 중지했습니다. 재개

ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 썸네일 이미지 버킷 생성 – 이전버킷이름-resized라는 이름으로 생성

ServerLess

❖ 썸네일 이미지 만들기

✓ 람다 함수 코드 작성

```
import boto3
import os
import sys
import uuid
from urllib.parse import unquote_plus
from PIL import Image
import PIL.Image

s3_client = boto3.client('s3')

def resize_image(image_path, resized_path):
    with Image.open(image_path) as image:
        image.thumbnail(tuple(x / 2 for x in image.size))
        image.save(resized_path)
```

ServerLess

❖ 썸네일 이미지 만들기

✓ 람다 함수 코드 작성

```
def lambda_handler(event, context):  
    for record in event['Records']:  
        bucket = record['s3']['bucket']['name']  
        key = unquote_plus(record['s3']['object']['key'])  
        tmpkey = key.replace('/', '')  
        download_path = '/tmp/{}'.format(uuid.uuid4(), tmpkey)  
        upload_path = '/tmp/resized-{}'.format(tmpkey)  
        s3_client.download_file(bucket, key, download_path)  
        resize_image(download_path, upload_path)  
        s3_client.upload_file(upload_path, '{}-resized'.format(bucket), key)
```


ServerLess

❖ 썸네일 이미지 만들기

✓ 이미지를 업로드해서 람다 함수를 실행하고 CloudWatch에서 로그 확인

● 모듈이 없다는 에러 발생

The screenshot shows the AWS CloudWatch Logs console for a Lambda function. The log stream is named 'tailstream'. The log entries show the function's execution details, including the runtime version (python:3.8.v31), the request ID, and the error message: '[ERROR] Runtime.ImportModuleError: Unable to import module 'lambda_function': No module named 'PIL' Traceback (most recent call last):'. The log also shows the function's duration, billed duration, memory size, and max memory used.

로그 이벤트
아래의 필터 막대를 사용하여 로그 이벤트의 용어, 구문 또는 값을 검색하고 매칭할 수 있습니다. [필터 패턴에 대해 자세히 알아보기](#)

이벤트 필터링

Clear 1m 30m 1h 12h Custom Local timezone 디스플레이

타임스탬프	메시지
	현재 이전 이벤트가 없습니다. 재시도
2023-10-26T19:18:06.503+09:00	INIT_START Runtime Version: python:3.8.v31 Runtime Version ARN: arn:aws:lambda:ap-northeast-2::runtime:85a1944859083528568a8398e5a0c22be94bdaefef94fa0fa061158bf4530956
2023-10-26T19:18:06.762+09:00	START RequestId: f3d4d82b-d599-44de-8f8c-405fe71c7314 Version: \$LATEST
2023-10-26T19:18:06.763+09:00	[ERROR] Runtime.ImportModuleError: Unable to import module 'lambda_function': No module named 'PIL' Traceback (most recent call last):
2023-10-26T19:18:06.764+09:00	END RequestId: f3d4d82b-d599-44de-8f8c-405fe71c7314
2023-10-26T19:18:06.764+09:00	REPORT RequestId: f3d4d82b-d599-44de-8f8c-405fe71c7314 Duration: 1.75 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 52 MB Init Duration: 258.46 ms
	현재 최신 이벤트가 없습니다. 자동 재시도를 일시 중지했습니다. 재개

ServerLess

❖ 썸네일 이미지 만들기

✓ 도커에서 모듈을 생성해서 추가

- 도커에서 파이썬 컨테이너를 실행

```
docker run --name lambda-img -it lambci/lambda:build-python3.8 bash
```

- 배시 셸에서 디렉토리 생성

```
mkdir -p opt/python
```

- 필요한 패키지 설치

```
pip install pillow -t opt/python
```

```
pip install boto3 -t opt/python
```

- 필요한 패키지 확인

```
cd opt/python
```

```
ls
```

```
exit
```

- 패키지를 로컬로 복사

```
docker container cp lambda-img:/var/task/opt/python  
/Users/adam/Documents/dofkerfiledown
```

ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 도커에서 모듈을 생성해서 추가
 - 복사된 디렉토리를 zip 파일로 압축

ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 도커에서 모듈을 생성해서 추가
 - 람다 함수 아래의 Layers를 클릭



ServerLess

- ❖ 썸네일 이미지 만들기
 - ✓ 도커에서 모듈을 생성해서 추가
 - Layer 추가

계층 추가

함수 런타임 설정

런타임 Python 3.8	아키텍처 x86_64
-------------------	----------------

계층 선택

계층 소스 정보
호환 런타임 및 열릴 세트 아키텍처가 있는 계층에서 선택하거나 계층 버전의 Amazon 리소스 이름(ARN)을 지정합니다. 새로운 계층을 생성할 수도 있습니다.

☐ **AWS 계층**
AWS에서 제공하는 계층 목록에서 계층을 선택합니다.

☒ **사용자 지정 계층**
AWS 계층 또는 조직에서 생성한 계층 목록에서 계층을 선택합니다.

☐ **ARN 지정**
ARN을 제공하여 계층을 지정합니다.

사용자 지정 계층
함수의 런타임과 호환되는 AWS 계층 또는 조직 생성 계층입니다.

pythonlayer ▼

버전

1 ▼

[취소](#) [추가](#)