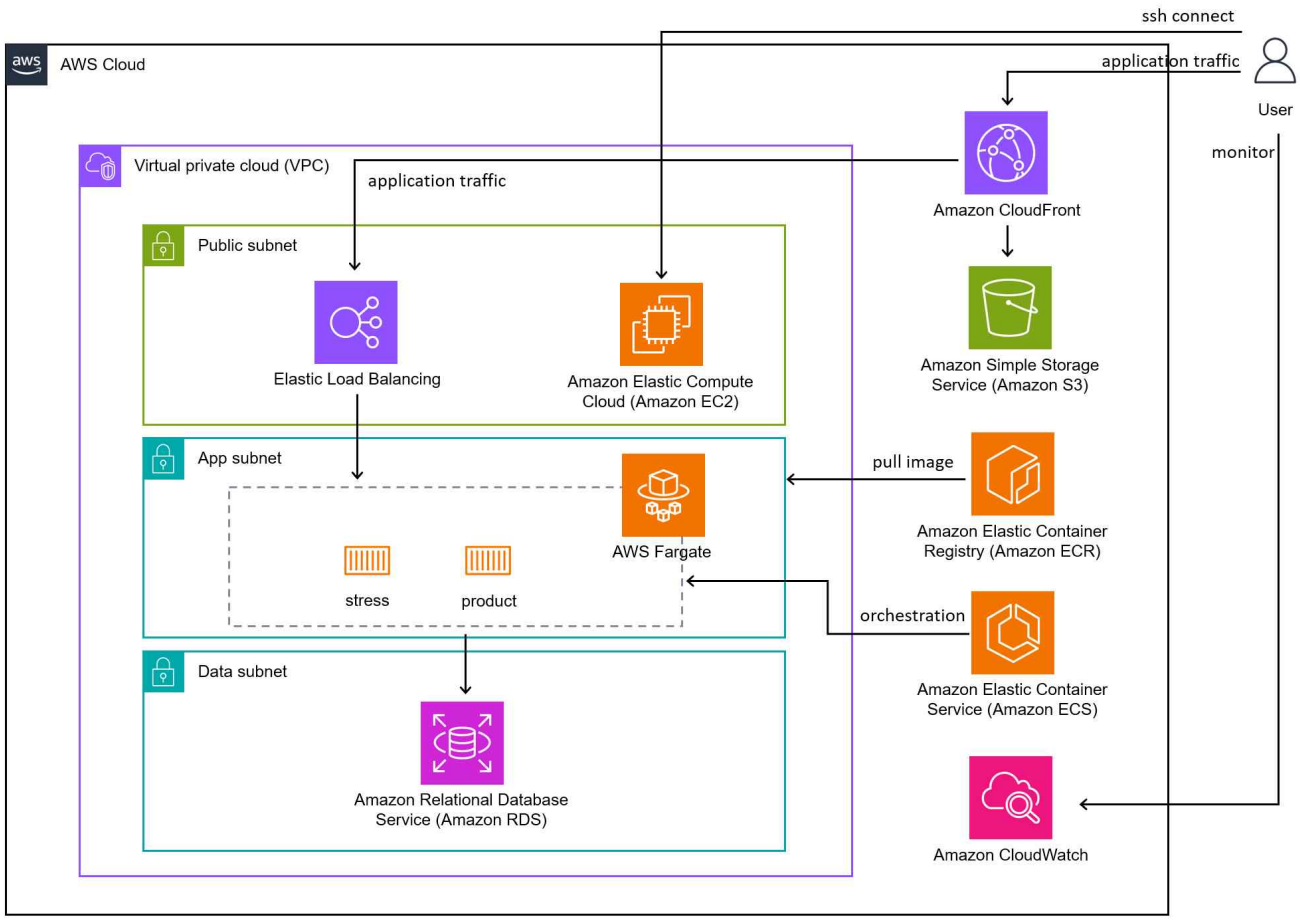
2023년도 전국기능경기대회

직 종 명	클라우드컴퓨팅	과 제 명	Web Service Provisioning	과제번호	제 1과제
경기시간	4시간	비 번 호		심사위원 확 인	(인)

1. 요구사항

MSA(Micro Service Pattern)의 REST API를 포함하는 웹 서비스 환경을 구축하고 운영하고자 합니다. MSA 패턴의 REST API를 운영하는 데 있어 여러 장점이 있는 컨테이너 기반의 환경과 AWS 관리형 서비스인 ECS를 컨테이너 오케스트레이션 툴을 사용해야 합니다. 그 외에도 여러 가지 AWS 서비스를 사용하여 웹 서비스를 운영할 수 있는 클라우드 플랫폼을 구성해야 합니다. 주어진 아키텍처를 바탕으로 고가용성, 성능, 보안, 운영효율성 등 여러 가지 요소를 고려하여 웹 애플리케이션이 구동할 수 있는 클라우드 플랫폼을 구축해야 합니다.

다이어그램



Software Stack

AWS	개발언어/프레임워크
- VPC	- Golang / Gin
- EC2	- Docker
- ECS	
- ELB	
- ECR	
- CloudFront	
- S3	
- Cloudwatch	
- RDS	
- Code Series	

2. 선수 유의사항

1) 기계 및 공구 등의 사용 시 안전에 유의하시고, 필요 시 안전장비 및 복장 등을 착용하여 사고를 예방하여 주시기 바랍니다.
2) 작업 중 화상, 감전, 찰과상 등 안전사고 예방에 유의하시고, 공구나 작업도구 사용 시 안전보호구 착용 등 안전수칙을 준수하시기 바랍니다.
3) 작업 중 공구의 사용에 주의하고, 안전수칙을 준수하여 사고를 예방하여 주시기 바랍니다.
4) 경기 시작 전 가벼운 스트레칭 등으로 긴장을 풀어주시고, 작업도구의 사용 시 안전에 주의하십시오.
5) 선수의 계정에는 비용의 제한이 존재하며, 이보다 더 높게 요금이 부과될 시 계정 사용이 불가능할 수 있습니다.
6) 문제에 제시된 괄호 박스 <>는 변수를 뜻함으로 선수가 적절히 변경하여 사용해야 합니다.
7) EC2 인스턴스의 TCP 80/443 outbound는 anyopen하여 사용할 수 있도록 합니다.
8) 과제의 Bastion 서버에서 대부분의 채점이 이루어짐으로 인스턴스를 생성하지 않았거나 종료된 상태면 채점이 불가능하니 각별히 주의하도록 합니다.
9) 과제 종료 시 진행 중인 테스트를 모두 종료하여 서버에 부하가 발생하지 않도록 합니다.
10) 별도 언급이 없는 경우, ap-northeast-2 리전에 리소스를 생성하도록 합니다.
11) 1페이지의 다이어그램은 구성을 추상적으로 표현한 그림으로, 세부적인 구성은 아래의 요구사항을 만족시킬 수 있도록 합니다. (ex. 서브넷이 2개 이상 존재할 수 있습니다.)
12) 모든 리소스의 이름, 태그, 변수과 변수는 대소문자를 구분합니다.
13) 문제에서 주어지지 않는 값들은 AWS Well-Architected Framework 6 pillars를 기준으로 적절한 값을 설정해야 합니다.
14) 불필요한 리소스를 생성한 경우, 감점의 요인이 될 수 있습니다. (e.g. VPC 추가 생성)

3. 네트워크 구성

클라우드 인프라에 대해 네트워크 레벨의 격리 및 분리를 할 수 있도록 아래 요구사항을 참고하여 VPC를 구성합니다. 컨테이너를 호스팅하는 서브넷(App Subnets)에서는 AWS 내부 네트워크만을 거쳐 안전하게 컨테이너 이미지를 내려받을 수 있도록 구성합니다. ECR 이미지를 내부 네트워크로 다운로드 받기위한 모든 구성을 해야합니다. 서브넷 이름 뒤의 알파벳은 Availability Zone을 의미합니다.

VPC 정보

- VPC CIDR : 10.1.0.0/16
- VPC Tag : Name=wsi-vpc
- Internet G/W Tag : Name=wsi-igw

App subnet A 정보

- CIDR : 10.1.0.0/24
- Tag : Name=wsi-app-a
- 외부 통신 : NAT G/W를 구성하여 인터넷 접근이 가능하도록 구성
- Route table Tag : Name=wsi-app-a-rt
- NAT G/W Tag : Name=wsi-natgw-a

App subnet B 정보

- CIDR : 10.1.1.0/24
- Tag : Name=wsi-app-b
- 외부 통신 : NAT G/W를 구성하여 인터넷 접근이 가능하도록 구성
- Route table Tag : Name=wsi-app-b-rt
- NAT G/W Tag : Name=wsi-natgw-b

Public subnet A 정보

- CIDR : 10.1.2.0/24
- Tag : Name=wsi-public-a
- 외부 통신 : Internet G/W 를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-public-rt

Public subnet B 정보

- CIDR : 10.1.3.0/24
- Tag : Name=wsi-public-b
- 외부 통신 : Internet G/W를 구성하여 인터넷을 접근
- Route table Tag : Name=wsi-public-rt

Data subnet A 정보

- CIDR : 10.1.4.0/24
- Tag: Name=wsi-data-a
- 외부 통신 : 인터넷 접근이 필요하지 않음
- Route table Tag : Name=wsi-data-rt

Data subnet B 정보

- CIDR : 10.1.5.0/24
- Tag: Name=wsi-data-b
- 외부 통신 : 인터넷 접근이 필요하지 않음
- Route table Tag : Name=wsi-data-rt

4. Bastion 서버

EC2를 활용해 Bastion 서버를 구성합니다. bastion 서버의 접근을 위해서 TCP 4272 포트를 SSH로 사용합니다. bastion 서버는 외부에서 SSH 프로토콜만을 허용하도록 Security Group을 구성하고, 만약 다른 Security Group Ingress Rule이 추가되면 1분 이내에 자동으로 Revoke 되어야 합니다. (기존 규칙은 Revoke 되지 않아야 합니다.)

- Instance type : t3.small
- Subnet : Public subnet A
- 설치 패키지 : awscli, jq, curl, git
- Tag : Name=wsi-bastion
- Security group name: wsi-bastion-sg (*명시된 Security Group 하나만을 사용합니다.)
- EC2 IAM Role : 모든 리소스에 대해 full access를 가지는 role을 생성하여 붙입니다. role 이름은 wsi-bastion-role로 설정합니다.

5. 관계형 데이터베이스

Product 애플리케이션의 데이터를 저장하기 위해서 관계형 데이터베이스를 구성합니다. 애플리케이션은 MySQL Community 엔진을 지원합니다. 애플리케이션 동작 설명을 기반으로 쿼리에 지연이 생기지 않도록 테이블을 설계하고 생성하세요. 정상적인 동작을 위해서 insert.sql 파일을 사용하여 데이터를 삽입해야 하며, 삽입한 데이터는 임의로 수정하거나 삭제하지 않도록 합니다. 또한 테이블에 임의의 데이터를 삽입하면 성능 저하가 생길 수 있으므로 추가적인 데이터 삽입을 하지 않도록 합니다.

- rds db identifier : wsi-rds-instance
- instance type : db.t3.medium
- Engine : MySQL Community 8.0
- database name : dev
- table name : product
- columns : id(VARCHAR 255), name(VARCHAR 255)

6. 웹 애플리케이션

Stress와 Product 2개의 애플리케이션이 있습니다. 제공된 binary는 x86기반 EC2의 Amazon Linux2에서 빌드하고 동작을 확인하였습니다. go version은 go1.18.9 linux/amd64입니다. 애플리케이션 실행 시 바인딩되는 포트는 TCP/8080입니다. 모든 애플리케이션은 access log를 log/app.log 파일(상대 경로)에 저장합니다.

- Stress

부하 테스트를 위해 cpu 부하를 주입하는 API를 제공합니다. Body로 전달한 숫자만큼 연산을 반복하여 cpu 부하를 발생시킵니다. Request body는 json 포맷을 사용하며 iterator라는 key를 사용하여 반복 횟수를 결정합니다.

Path	Method	Request body	Response body
/v1/stress	POST	{"iterator": 10000}	{"message": "The cpu is loaded"}
/v1/stress	GET	-	{"version": "v1.0"}
/healthcheck	GET	-	{"status": "ok"}

- Product

Query String으로 전달한 문자열로 product 존재 여부를 확인하는 API를 제공합니다. name이라는 key를 전달하면 서버에서 관계형 데이터베이스에 저장된 데이터를 쿼리하여 그 결과를 사용자에게 응답합니다. 애플리케이션 내부에서 쿼리는 "select id, name from dev.product where name = $name" 형태로 이뤄집니다. Product 애플리케이션은 5초 이내에 응답할 수 있어야 합니다.

Path	Method	Query String	Response body
/v1/product	GET	?name=p-xxxxxxxx	{"message": "The product is well in database"}
/healthcheck	GET	-	{"status": "ok"}

Product 애플리케이션은 실행 시 dbinfo 환경 변수를 읽어 관계형 데이터베이스에 연결합니다. dbinfo 환경변수의 값은 json으로 직렬화되어야 하며, 아래 테이블의 정보를 담고 있어야 합니다. 환경변수 값은 Task Definition에 직접적으로 명시해선 안 되며 Secrets Manager에 저장된 값을 읽어와야 합니다. 환경변수 값을 저장하기 위한 Secrets Manager의 Secret 이름은 dbsecret으로 설정합니다.

Key	Data Type	Value
username	string	db 접근 권한을 가지는 유저의 이름
password	string	db 접근 권한을 가지는 유저의 암호
engine	string	db 엔진 (mysql or postgresql)
host	string	db 읽기/쓰기가 가능한 주소
port	string	db 읽기/쓰기가 가능한 주소의 포트번호
dbname	string	논리적인 데이터베이스 이름 (dev)

7. 컨테이너 오케스트레이션

ECS를 통해 컨테이너를 배포하고 관리합니다. 관리 부담을 줄이기 위해서 Fargate 컴퓨팅 타입을 사용합니다. Stress 애플리케이션과, Product 애플리케이션을 각각 서비스로 배포합니다. Stress 애플리케이션은 CPU load에 따라 자동으로 스케일링 될 수 있도록 구성합니다. 모든 애플리케이션은 하나의 Task가 비정상적으로 종료되어도 오류 없이 서비스할 수 있도록 구성해야 합니다. 또한 트레픽을 없을때는 고가용성을 고려한 최소한의 Task만 유지 되도록 해야 합니다. 스케일링 테스트시 5분 이내에 scale-out이 완료 되어야 합니다.

- ECS Cluster name: wsi-ecs-cluster
- Computing Type: Fargate
- Service name: stress, product
- CPU/Memory size: 0.5 vCPU / 1GB

8. 로드밸런서

ALB를 구성하여 외부에서 Stress 애플리케이션과 Product 애플리케이션으로 접근할 수 있도록 구성합니다. Stress 애플리케이션과 Product 애플리케이션은 ALB를 통해서만 접근할 수 있도록 구성합니다. 공개 도메인이 없어 ACM을 통해 Listener에 인증서를 적용할 수 없는 환경입니다. 약간의 성능 저하를 감수하고 보안성을 향상시키기 위해 외부에서 CloudFront를 통해 ALB에 접근할 수 있도록 구성합니다. ALB는 CloudFront 외에는 HTTP 접근을 허용하지 않도록 구성합니다. 접근 제어는 IP 레벨의 구성이 되어 잘못된 접근시 timeout과 같은 에러가 발생해야 합니다. Path 기반으로 규칙을 작성하여 꼭 필요한 API 접근만 허용하고 불필요한 API 접근시 404 에러와 "Contents Not Found" 메시지를 반환하도록 구성합니다. Path 라우팅 시 *을 이용한 라우팅은 금지합니다.

- Network facing : Internet-facing
- Listen : HTTP 80
- ALB Name : wsi-alb
- Tag : Name=wsi-alb

8. S3

S3를 통하여 정적 콘텐츠를 저장하고 제공합니다. 정적 콘텐츠는 ap-northeast-2 리전과 us-east-1 리전에 거쳐 저장되어야 합니다. 하나의 리전의 버킷에 콘텐츠를 업데이트하거나 새로 저장하면 다른 리전에 자동으로 복제되도록 합니다. Delete marker는 복제되지 않도록 구성합니다. 콘텐츠를 다른리전에 복제시 1분 이내에 완료되어야 합니다. 모든 콘텐츠는 업로드 시 자동으로 KMS의 임의의 CMK로 암호화되어야 합니다. 모든 콘텐츠 파일은 /static/ 접두사를 가진 오브젝트 키(e.g. /static/index.html)로 업로드 합니다.

- 버킷이름(ap-northeast-2): ap-wsi-static-<임의의 4자리 영문>
- 버킷이름(us-east-1): us-wsi-static-<임의의 4자리 영문>

9. Cloud Front

CloudFront를 통하여 정적 콘텐츠 및 애플리케이션에 접근이 가능하도록 합니다. S3에 업로드 되는 정적 콘텐츠를 캐싱할 수 있도록 구성합니다. us-east-1 리전의 버킷에 콘텐츠가 없을 경우 ap-northeast-2 리전의 버킷에 요청을 전달하도록 구성합니다. ALB로의 요청에 대해서는 캐싱하지 않고 Query String도 모두 Origin으로 전달해야 합니다. 사용자가 CloudFront에 HTTP 접근 시에도 HTTPS로 리디렉션 되어 HTTPS로만 접근할 수 있도록 구성합니다.

- Origin : S3와 ALB 2개의 origin을 가지도록 구성
- Edge : 한국뿐만 아니라 전 세계의 유저가 빠른 속도로 접근할 수 있도록 구성
- Tag : Name=wsi-cdn
- 기타 : 채점 시 오동작 예방으로 IPv6는 비활성화하고, 하나의 CloudFront만 생성

10. 로그

Cloudwatch Logs를 통해 중앙 집중식 로깅 솔루션을 구성합니다. Stress 애플리케이션과 Product 애플리케이션에서 발생하는 access log를 각 log group에 저장합니다. 다만, /healthcheck 경로에 대한 접근 로그는 cloudwatch Logs 그룹에 저장되지 않아야 합니다.
로그는 최소 1분 이내에 Cloudwatch Logs 로 수집되어야 합니다.

- Stress 애플리케이션 로그
  - 로그 그룹 이름 : /wsi/webapp/stress
  - 로그 스트림 이름 : <ecs task id> 예) 37026c72b86a4847b71e2c86ebaa4fc9
- Product 애플리케이션 로그
  - 로그 그룹 이름 : /wsi/webapp/product
  - 로그 스트림 접두사 : <ecs task id> 예) 75b8b7d4edc24c1d927c8432d83d6398

12. CI/CD 파이프라인

Stress 애플리케이션을 위한 CI/CD 파이프라인을 구성합니다. Stress 이름의 CodeCommit Repository를 생성하여 애플리케이션 바이너리와 구성파일을 관리합니다. Bastion 호스트의 /home/ec2-user/stress/ 경로에 바이너리와 구성파일을 위치시키고, main 브랜치에 commit을 생성하여 Repository로 푸쉬합니다. 애플리케이션 바이너리는 이름을 변경하지 않고, Repository의 최상위 경로에 위치시킵니다. 컨테이너 빌드를 위한 dockerfile 이름은 Dockerfile로 명명하고, Repository의 최상위 경로에 위치시킵니다. push.sh 스크립트 파일을 만들고, 스크립트 실행 시 자동으로 변경 점이 CodeCommmit Repository에 푸쉬 되도록 구성합니다. 스크립트 파일은 /home/ec2-user 에 위치하도록 합니다.

ex. 아래처럼 명령어 입력 시, 자동으로 CodeCommit Repository에 푸쉬 되어야 합니다.
$ /home/ec2-user/push.sh "커밋 메시지"

컨테이너 이미지 빌드 시에는 이미지를 스캔하여 CVE를 포함한 알려진 취약점이 발견되면 배포가 중지되도록 구성합니다. 배포에는 Blue/Green 무중단 전략을 사용하며, main 브랜치에 커밋이 발생하면 자동으로 운영 중인 시스템에 배포되어야 합니다. 배부된 바이너리를 사용하여 테스트를 수행할 수 있습니다. 배포는 5분 이내에 완료 되어야 합니다. 하지만, 채점 전에는 stress-v1.0 폴더 하위의 바이너리가 배포되어 있어야 합니다.

- stress-v1.0 폴더 하위의 바이너리: 기존 바이너리
- stress-v1.1 폴더 하위의 바이너리: 새로운 버전의 바이너리