EKS 아키텍처 가이드

Cloud_서비스/세일즈 툴킷

Exported on 06/19/2023

Table of Contents

Public Cloud에서 컨테이너 형태로 어플리케이션을 개발 및 운영하고자 하는 경우 활용할 수 있는 방법을 정리합니다.

각 Cloud Service Provider가 제공하는 관리형 Kubernetes 서비스를 대상으로 플랫폼 관리 영역과 어플리케이션 개발 영역으로 나눠 서비스의 기능을 비교하고

어플리케이션을 개발 및 관리하는 방법을 가이드합니다.

1 EKS 1. 아키텍처 구성 및 고려사항

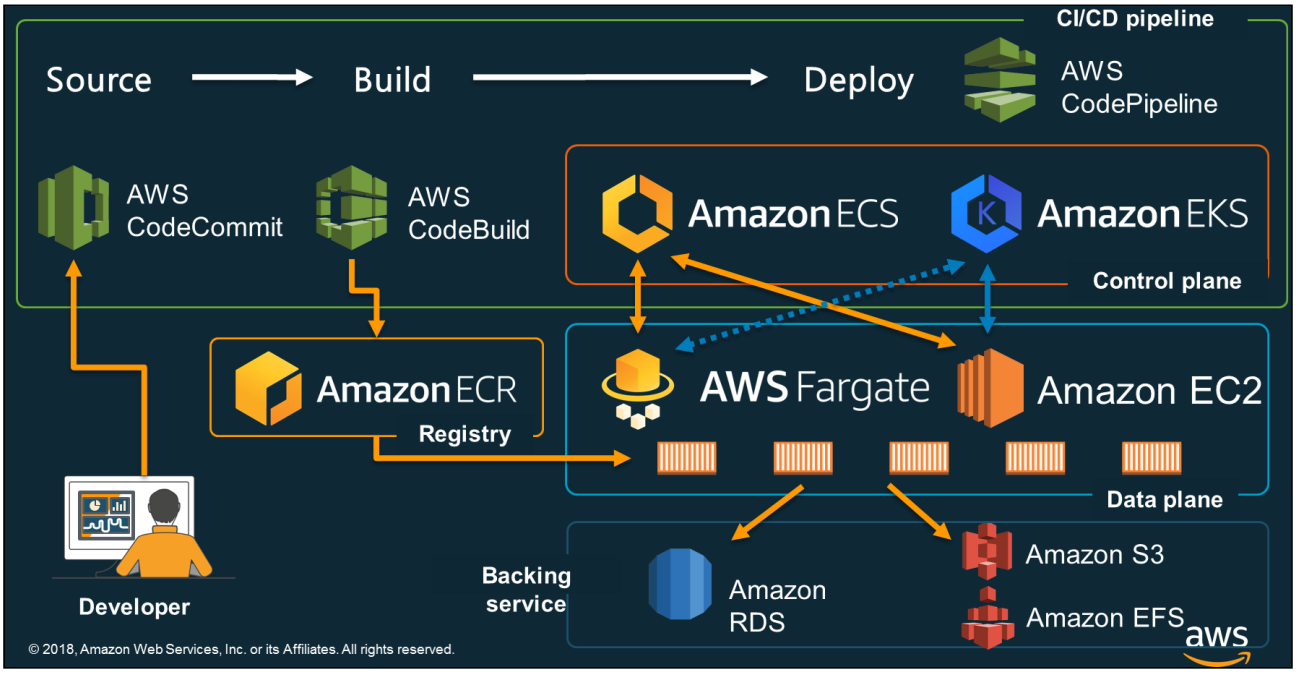
Amazon Elastic Container Service for Kubernetes(EKS)를 활용하여 AWS에서 Kubernetes를 사용할 때 활용할 수 있는 AWS 서비스와 아키텍처 구성에 대해 개괄적으로 설명한다.

1.1 EKS 개요

EKS(Amazon Elastic Container Service for Kubernetes)는 Kubernetes를 사용하여 컨테이너식 애플리케이션을 손쉽게 배포, 관리 및 확장할 수 있게 해주는 서비스이다. EKS는 Kubernetes 커뮤니티의 기존 도구 및 플러그인을 사용할 수 있으며, ECS, ECR, Pipeline 등 Amazon에서 제공하는 각종 서비스와 결합하여 사용도 가능하다.

1.2 아키텍처 구성

AWS Container Service의 전체적인 아키텍쳐이다.



Control Plane은 컨테이너 클러스터 마스터 역할을 수행하는 곳으로 Amazon ECS, EKS가 있다. DataPlane은 실제로 컨테이너들이 배포되는 곳으로 AWS Fargate, EC2가 있다. 이외에 AWS CodePipeline을 이용해 CI/CD 파이프라인 구성이 가능하며, Registry로 Amazon ECR을 사용하는 것이 가능하다. 또한, RDS, S3, EFS 서비스와의 결합도 가능하다.

1.3 EKS vs ECS vs Elastic Beanstalk vs Fargate 비교

컨테이너를 실행할 수 있는 다양한 서비스가 존재하며, 기능 비교를 통해 각자의 환경에 적합한 서비스를 활용할 수 있도록 한다.

EKS(Elastic Container Service for Kubernetes)는 Kubernetes를 사용하는 컨테이너식 어플리케이션 관리형 환경이다. Amazon EKS는 Kubernetes Master를 여러 가용 영역(Availability Zone)에 자동으로 배포함으로 가용성이 높은 아키텍처를 제공하는 클라우드 서비스이다. EKS에서 실행되는 Application은 단일 마스터 혹은 전체 가용 영역의 손실에 대해 복원력을 가지며 마스터에 대한 위험 요소 탐지, 버전 업그레이드를 자동으로 수행한다. 또한 클러스터의 관리, 확장 및 업그레이드와 관련된 작업을 자동화해준다. 뿐만 아니라, AWS의 모든 성능, 확장성, 안정성 및 가용성은 물론 애플리케이션 로드 밸런서, AWS IAM(Identity and Access Management), AWS 프라이빗 링크(PrivateLink), AWS 클라우드 트레일(CloudTrail)을 비롯한 AWS 네트워킹 및 보안 서비스를 사용할 수 있다.

ECS(Elastic Container Service)는 클러스터에서 Docker 컨테이너를 쉽게 실행, 중지, 관리 해주는 컨테이너 관리 서비스이다. AWS에서 컨테이너식 어플리케이션을 쉽게 실행하고 확장, 축소할 수 있으며 시작유형으로 Fargate, EC2를 선택할 수 있다. 간단한 API 호출을 사용하여 컨테이너 기반 어플리케이션을 시작, 중지할 수 있고 중앙 집중식 서비스를 사용하여 클러스터 상태를 확인할 수 있도록 해준다. 또한, 사용자가 자체 클러스터 관리 및 구성 관리 시스템을 운영하거나 인프라 조정을 신경 쓸 필요 없이 일관된 배포 및 구축 환경을 생성하고, MSA 모델에 정교한 어플리케이션 아키텍처를 구축할 수 있다.

Elastic Beanstalk는 AWS 환경에서 가장 쉽게 웹 어플리케이션 / 웹 서비스를 배포하고, 확장하고, 관리할 수 있는 완전 관리형 서비스이다. Java, .NET, PHP, Node.js, Python, Ruby, Go, Docker를 사용하여 Apache, Nginx, Passenger, IIS와 같은 친숙한 서버에서 개발된 웹 애플리케이션 및 서비스를 간편하게 배포하고 조정할 수 있다. 어플리케이션을 업로드만 하면 용량 프로비저닝, 부하분산, 조정, 모니터링, 배포 등을 자동으로 처리해주어 개발자가 어플리케이션을 실행하는 데 필요한 인프라에 대한 고민 없이 AWS 클라우드에서 어플리케이션을 신속하게 배포하고 관리할 수 있게 한다.

구분	Elastic Container Service for Kubernetes	Elastic Container Service	Elastic Beanstalk
지원되는 언어	any	any	Docker, Multi-container Docker, 사전 구성 (GO, .Net, Java, Node.js, Ruby, PHP, Python, Tomcat, GlassFlas, Elastic Beanstalk Packer Builder)
지원되는 버전	any	any	언어별 지원되는 버전이 고정됨
이용모델	IaaS, PaaS	PaaS	PaaS

구분	Elastic Container Service for Kubernetes	Elastic Container Service	Elastic Beanstalk
Runtime 환경	docker container	docker container	VM 형태
	사용자가 만든 docker image를 실행함	사용자가 만든 docker image를 실행함	소스를 기반으로 빌드하여 사전 정의된 실행환경(sandbox)의 디렉터리에 배포하여 실행함
Cluster 자동 확장	cluster EC2 node 확장하는 방식으로 지원 가능	Aamazon ECS 콘솔에서 CloudWatch Metric을 이용하여 정책 설정 가능함	EC2 node 확장하는 방식으로 지원 가능
배포환경	EC2 VM 생성, Fargate 지원 예정	EC2 VM 생성, Fargate 지원	EC2 VM 생성
서비스 리전	일부 지역 (버지니아북부, 오레곤, 아일랜드) 19년 1월 서울 지역 추가	오사카를 제외한 전 지역	전 지역
타서비스 연계 (Cloud Watch, SNS, Lambda등)	CloudWatch 일부 지원 (VM), 그 외는 로드맵에 포함(Master node 통합 로깅, Container 배포 현황, Alert 기능 등)	지원	지원
비용	시간당 0.20 USD + AWS 리소스 사용량에 따른 비용 지불	실제 사용한 AWS 리소스 사용량에 대해서만 비용 지불	실제 사용한 AWS 리소스 사용량에 대해서만 비용 지불
장점	작업자 노드 클러스터를 프로비저닝, Kubernetes 제어 플레인의 프로비저닝, 확장 및 관리를 처리	서비스 스케일링, 태스크 기반 IAM 지원, 애플리케이션 로드 밸런서 지원, 서비스 디스커버리, AWS Batch, Fargate 지원	Amazon EC2 인스턴스, 용량 프로비저닝, 로드 밸런싱, Auto Scaling, 모니터링 등 어플리케이션 실행을 위해 필요한 모든 세부 사항을 자동으로 처리

Fargate는 서버 및 클러스터를 관리할 필요없이 컨테이너를 실행할 수 있도록 해주는 컴퓨팅 엔진이다. AWS Fargate를 이용하면 컨테이너 실행을 위해 가상 머신 클러스터를 프로비저닝, 구성, 확장할 필요가 없다. 사용자는 인스턴스 유형을 선택하는 대신 컨테이너, CPU 및 메모리 요구사항, 네트워킹 정의, IAM 정책 설정 등의 작업을 통해 어플리케이션을 정의해주면 된다. 현재는 ECS 배포를 위한 컴퓨팅 엔진으로 사용되지만 EKS 로드맵에 Fargate가 클러스터 옵션으로 추가될 예정이다.

1.4 Kubernetes와 EKS 비교

오픈소스 Kubernetes 대비 Amazon EKS가 제공하는 차별적인 기능에 대해 소개한다.

- 관리형 Kubernetes 제어 플레인 : 높은 가용성을 보장하기 위해 3개 가용 영역에서 Kubernetes 제어 플레인을 실행하고 비정상 Master를 자동으로 감지해서 교체함
- 각 클러스터의 etcd와 API 서버의 가용성과 확장성을 자동으로 관리함
- AWS IAM의 상세한 정책 관리 : IAM과 Heptio를 이용해 Kubenretes 마스터에 대한 액세스 권한을 세분화 하여 제어할 수 있음
- VPC 지원 : EKS는 AWS의 VPC 내의 워커 노드에 배포되므로 VPC의 기능(Route Table, Nat G/W, Security Group, NACL 등)을 사용하여 높은 수준의 보안 설정이 가능함
- 네트워킹 정책 : Calico와 연동하여 Kubernetes 워크로드에 대한 세분화된 네트워킹 정책을 제공함
- 로깅 및 모니터링 : AWS Cloudtrail을 사용하여 EKS API에 대한 호출 내역을 확인할 수 있으며, AWS Cloudwatch, ElasticSearch를 사용하여 애플리케이션의 상태 확인이 가능함
- Private Container Registry : Amazon Elastic Container Registry와 결합하여 손쉽게 Private Docker 이미지를 저장하고 액세스가 가능함
- 빌드 구성 : AWS Codepipeline를 사용하여 인증을 설정할 필요 없이 Kubernetes Engine에서 안정적으로 컨테이너 배포가 가능함

2 EKS 2. 플랫폼 테스트 결과

2.1 테스트 결과 요약

Amazon Web Services에서 서비스중인 EKS(Elastic Container Service for Kubernetes)의 플랫폼 기능을 사용해보고 테스트한 결과 AWS ECS(Elastic Container Services)와 Google의 GKE와 같은 다른 클라우드 서비스 대비 아직 기능 개발이 부족한 상태이다. 기본적인 모니터링 연계를 위한 기능역시 개발이 진행중이며 AWS 클라우드 서비스의 높은 시장 점유율을 바탕으로 EC2 인스턴스에 직접 Kubernetes를 수동으로 설치하여 사용하는 비중이 높은것으로 보인다.

2.2 테스트 결과 상세

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
설치/변경	Host OS 설치/구성 자동화	VM 의 OS 설치 및 구성이 자동화 되는가	20	1	1	- 기능 지원 - EKS와 CloudFormation을 통해 클러스터 구성 및 Worker 노드 자동화 구성 지원 - 클러스터 구성을 위해 사전에 계정, Role, VPC 구성이 사전 완료 되어야 함.
	Host OS 업그레이드 자동화	VM 의 OS Upgrade 가 플랫폼 중단 없이 자동화 되는가		1	0	- 지원 안됨 - 단일 버전만 지원함. - 로드맵에 마이너레벨은 지원예정에 있고, 메이저는 검토 중임, 새로운 노드를 만들어서 붙이는 방법
	Host OS 다운그레이드 자동화	VM 의 OS Downgrade 가 플랫폼 중단 없이 자동화 되는가		1	0	- 지원 안됨 - 단일 버전만 지원함.

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	플랫폼 설치	플랫폼 최초 설치 (플랫폼 권장하는 버전으로) 되는가		3	3	- 기능 지원 - EKS 플랫폼 단일 버전 설치 기능 제공 - 마스터 노드는 기본 API, etcd 3대의 호스트가 설치됨. - Worker node의 규모에 따라 마스터 노드의 서버 스펙이 자동으로 확장됨 (비용은 동일함)
	플랫폼 설치 정의서	정의서(예: Manifest YAML) 를 통해 설치가 자동화 되는가.		2	2	- 기능 지원 - YAML 또는 등록된 플랫폼 템플릿을 통한 설치 기능 지원
	플랫폼 설치 정의서의 민감정보 분리 보관	정의서에서 민감한 정보 (예: 계정, 비번, 인증서, Private Key) 을 분리 보관할 수 있는가.		2	2	- 기능 지원 - IAM 계정 및 역할 권한, Security Group으로 EKS 클러스터의 액세스 권한을 관리
	플랫폼 설치 정의서의 민감정보 암호화	분리된 민감 정보를 안전하게 보관하는 메커니즘이 제공되는가		2	2	- 기능 지원 - IAM 계정 및 역할 권한, Security Group으로 EKS 클러스터의 액세스 권한을 관리
	설치 정의서의 템플릿화	정의서의 변경을 통해 설정 변경이 가능한가.		2	2	- 기능 지원 - 플랫폼 자체 Control Plane에서 제공되는 기능은 없고 YAML 템플릿을 재활용 가능

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	설치/관리 GUI 조회 제공 여부	설치된 플랫폼의 구성 정보를 확인하기 위한 GUI 가 제공되는가		2	1	- 부분 기능 지원 - AWS 콘솔을 통해 기본적인 정보 확인은 가능하나 제공되는 정보는 많지 않음. - EKS 클러스터 생성 기능만 GUI 환경을 제공하며 Worker node 및 Cluster 구성 정보 확인등은 CloudFormation과 CLI를 사용해야 함.
	설치/관리 GUI 변경 제공 여부	GUI 에서 Ansible Inventory 정보 수정/배포 기능이 있는가		2	1	- 부분 기능 지원 - AWS 콘솔을 통해 기본적인 정보 확인은 가능하나 제공되는 정보는 많지 않음. - EKS 클러스터 생성 기능만 GUI 환경을 제공하며 Worker node 및 Cluster 구성 정보 확인등은 CloudFormation과 CLI를 사용해야 함.
	플랫폼 검증 자동화	설치후 시스템 정상 여부 확인 자동화가 가능한가		2	0	- 지원 안함 - 플랫폼에서 자체 제공되는 기능 없음
업그레이드	플랫폼 업그레이드	호환성 유지 버전 업그레이드 기능 확인이 가능한가	15	4	2	- 부분 기능 지원 - 최신 버전의 EKS가 설치되며 기존 cluster는 최신 플랫폼으로 자동 업그레이드 지원
	롤백	업그레이드 도중 장애 발생시 롤백될 수 있는가		4	0	- 지원 안함 - 플랫폼에서 자체 제공되는 기능 없음
	플랫폼 다운그레이드	호환성 유지 버전 다운그레이드 기능 확인이 가능한가		4	0	- 지원 안함 - 플랫폼에서 자체 제공되는 기능 없음

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	자동 플랫폼 업그레이드	자동 버전 업그레이드 기능을 제공하는가		3	1	- 부분 기능 지원 - 최신 버전의 EKS가 설치되며 기존 cluster는 최신 플랫폼으로 자동 업그레이드 지원
내결함성	Master Node 장애	마스터 노드 장애시 플랫폼 가용성 확인이 가능한가	15	3	3	- 기능 지원 - Cluster 호스트 장애 상황 확인 불가
	Workder Node 장애	워커 노드 장애시 플랫폼 가용성 확인이 가능한가		3	3	- 기능 지원 - 호스트 장애 상황 확인이 어려우나 가용존 이중화로 무중단 서비스 구성 가능 - worker node인 자동 복구 지원함, desired count 지정하여 장애 발생시 자동으로 신규 노드 추가함
	Infra Node(운영관리 Node) 장애	인프라 노드 장애시 플랫폼 가용성 확인이 가능한가		2	2	- 기능 지원 - 호스트 장애 상황 확인이 어려우나 가용존 이중화로 무중단 서비스 구성 가능
	Router Node 장애	라우터 노트 장애시 플랫폼 가용성 확인이 가능한가		3	3	- 기능 지원 - 호스트 장애 상황 확인이 어려우나 가용존 이중화로 무중단 서비스 구성 가능
	PV 스토리지 내결함성	스토리지 장비 또는 서비스 장애시 플랫폼 내결함성 범위 확인이 가능한가		2	2	- 기능 지원 - 디스크 장애 확인은 불가
	자동 노드 복구	Worker Host 장애를 자동 인지하고 자동 복구 기능을 제공하는가		2	1	- 기능 지원 - worker node인 자동 복구 지원하며 EC2의 auto scaling 기능의 desired count 및 min/max 지정함

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
백업 / 복구	플랫폼 백업/복구	플랫폼 삭제 및 동일 버전 재설치 후 데이터 복구 가능 여부 확인이 가능한가	5	5	1	- 지원 안됨 - 플랫폼의 기능을 백업 받기 보다는 template등을 통해 재배포 하는 방식 권고 - OS snapshot 기능을 이용하여 백업/복구 구현 가능
스케일링	Master Node 스케일링	운영자원 효율화를 위해 마스터 노드를 증설하거나 감소할 수 있는가	5	1	1	- 기능지원 - 별도의 Control Plane 증설 관련 기능 확인 불가하며 EKS 플랫폼에서 자동으로 관리함. - 기본적으로 6대 서버가 생성됨(master 3대, etcd 3대), master node의 상세 스펙에 대해서 확인 불가함 - master node에 대해서는 부하에 따라 자동으로 상위 스펙의 Master 노드로 변경됨 - master node 스펙 상승에 따른 추가 비용은 발생하지 않음.
	Workder Node 스케일링	운영자원 효율화를 위해 워커 노드를 증설하거나 감소할 수 있는가		2	2	- 기능 지원 - cluster autoscaling group 기능의 EC2 node 확장하는 방식으로 지원 가능
	Infra Node(운영관리 Node) 스케일링	운영자원 효율화를 위해 인프라 노드를 증설하거나 감소할 수 있는가		1	1	- 기능 지원 - cluster EC2 node 확장하는 방식으로 지원 가능
	Router Node 스케일링	운영자원 효율화를 위해 라우터 노드를 증설하거나 감소할 수 있는가		1	1	- 지원 안됨 - 별도의 Control Plane 증설 관련 기능 확인 불가

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
플랫폼 관리	운영자 관리 GUI 제공	운영자가 PaaS 플랫폼을 관리 할 수 있는 GUI 관리 화면을 제공하는가	20	4	2	- 부분 기능 지원 - 콘솔에서 Cluster 관리 화면은 제공하나 제공정보 및 기능이 거의 없음.
	운영자 계정 관리	LDAP 계정과 연계한 운영자 계정 관리가 가능한가		4	4	- 기능지원 - LDAP/AD 계정 연계를 위해 AWS Directory Servie, AWS SSO, IAM Role을 연계하여 구성 가능
	유관시스템 계정 연계(모니터링/로그/미터링)	LDAP 계정과 연계한 모니터링/미터링 계정 관리가 가능한가		5	5	- 기능 지원 - IAM & admin 서비스에서 사용자 및 권한 통합 관리 가능하며 LDAP 연동 가능
	Maintenance Window 지정	업무 부하가 작은 시간대를 정하여 Maintenance 가능한가		3	0	- 부분 기능 지원 - EKS 플랫폼에서 자체 제공되는 기능 없으며 EC2 인스턴스에 Maintenance Windows 설정 수준 가능
	PV 스토리지 자원 관리 기능	PV 자원의 할당 / 회수 자동화 기능이 가능한가		4	3	- 기능지원 - EBS(하나의 POD), EFS(여러 POD)를 사용할 수 있음
모니터링	Host 자원 모니터링	각 VM의 자원 모니터링을 통합하여 제공하는가	10	2	1	- 부분 기능 지원 - EKS의 CloudWatch 연계는 현재 개발중임. - CloudWatch의 EC2 연계 방식으로 로그 정보 수집 되도록 구성 가능
	컨테이너 배치 모니터링	각 컨테이너의 자원 모니터링을 통합하여 제공하는가		3	1	- 부분 기능 지원 - EKS의 CloudWatch 연계는 현재 개발중임. (로드맵 3개월) - CloudWatch의 EC2 연계 방식으로 로그 정보 수집 되도록 구성 가능

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	데이터 보관기간 설정	모니터링 데이터 보관 기능을 조정할 수 있는가		2	1	- 지원 안됨 - CloudWatch를 통한 로그 통합 기능 지원 안됨(로드맵 3개월)
	알람 기능 제공	플랫폼 운영자 관점에서 알람기능을 제공하는가		3	0	- 지원 안됨 - Cloudwatch와 아직 연동 기능이 개발중이므로 알람 기능과 연동되지 않음
로그통합	Host 로그통합	각 VM의 로그를 통합하여 제공하는가	10	2	1	- 지원 안됨 - CloudWatch를 통한 로그 통합 기능 지원 안됨(로드맵 3개월)
	컨테이너 로그통합	컨테이너 Lifecycle 및 Event 로그를 통합하여 제공하는가		2	1	- 부분 지원 - EFK((Fluentd daemonset -> ElasticSearch -> Kibana) 수동 설치 기능 지원 - AWS 관리형 ElasticSearch 서비스인 AmazonES(Kibana기본 탑재) 연계 구성 가능
	데이터 보관기간 설정	로그 데이터 보관 기능을 조정할 수 있는가		2	1	- 지원 안됨 - CloudWatch를 통한 로그 통합 기능 지원 안됨(로드맵 3개월)
	Audit 로그 제공	보안 관점에서 분석 가능한 audit logs를 제공하는가		1	1	- 기능 지원 - AWS CloudTrail과 연계하여 클러스터 및 사용자 활동 및 audit 로그를 제공함
미터링	사용자 과금 정책 적용	사용자 정의 과금정책 적용 가능한가		1	0	- 지원 안함
	인스턴스 단위 미터링	컨테이너수 단위 미터링 가능한가		1	0	- 지원 안함

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	자원사용량 단위 미터링	CPU, MEMORY, Storage 사용량 단위 미터링 가능한가		1	0	- 지원 안함

- 테스트 결과 요약(see page 8)
- 테스트 결과 상세(see page 8)

3 EKS 3. 어플리케이션 테스트 결과

3.1 테스트 결과 요약

Kubernetes 기본 기능을 활용하는 부분이 많으며, 기존 AWS 서비스와의 결합해서 사용할 수 있는 부분이 아직은 부족하다.

네트워크, 보안 부분은 AWS VPC, Subnet 등 네트워크 관련 서비스를 사용하여 효과적으로 관리가 가능하며, 이 외에 어플리케이션 관리에 필요한 서비스들(Cloudwatch, Autoscaling 등)은 로드맵에 따라 진행이 되어 EKS와의 결합도가 올라간다면 AWS에서 EKS를 보다 효과적으로 사용할 수 있으며, 사용 편의성 또한 올라갈 것이라고 기대된다.

3.2 테스트 결과 상세

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
컨테이너 빌드/배포	소스코드 빌드/배포	소스 코드를 이용한 빌드/배포 기능	15	5	0	- 제공되지 않음
	DockerHub, 사용자 개발 컨테이너 이미지 배포	컨테이너 이미지를 이용한 빌드/배포 기능		5	5	- Docker 기반으로 이미지 배포, 관리가 가능함
	Recreate 배포기능 확인	Recreate 배포 방식 지원		1	1	- latest가 retry를 계속 하다가 정해진 횟수가 초과되면 fail되며 이전버전이 다시 올라옴
	Rolling 배포기능 확인	Rolling 배포 방식 지원		1	1	- Deployment config 베이스로 배포정책이 설정되고 수행됨
	Rollback 배포기능 확인	Rollback 기능 지원		1	1	-Kubernetes 기능을 이용해 가능함 kubectl rollout undo deployment/hello

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	BlueGreen 배포기능 확인	BlueGreen 배포 방식 지원		1	0	- 기본제공 하지 않으며 spinnaker를 활용하여 제공 가능
	Canary 배포기능 확인	Canary 배포 방식 지원		1	0	- 기본제공 하지 않으며 spinnaker를 활용하여 제공 가능
컨테이너 레지스트리 관리	Container Registry Service 확인	Container Registry 제공 여부, 설치 편의성	5	2	2	- Amazon ECR을 제공함 - 콘솔에서 ECR 설정 후 사용이 가능함
	Container Regsitry 내 이미지 관리 편의성	이미지 버전 관리 편의성		1	0	- Imager Repository 이름을 콘솔에서 설정 후에 사용이 가능함
	이미지 관리를 위한 추가 기능 확인	별도의 관리 Portal 제공여부, 웹취약점, Docker signed 확인 기능		2	1	- AWS 콘솔 내 별도 관리 Page가 존재함
컨테이너 관리	컨테이너 자원할당 지정	CPU, Memory, storage 지정	25	3	3	- Kubernetes의 기능으로 cpu, memory의 Request, Limit 값을 yaml에 설정하여 배포할 수 있음
	컨테이너 이미지 관리(버전 관리)	컨테이너 이미지, 버전 관리 기능		3	3	- Amazon ECR을 이용하여 Private Registry를 사용할 수 있음 - namespace, tag등을 docker hub와 동일하게 사용할 수 있음

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	컨테이너 확장 시 자동식별 및 부하분산	컨테이너 수동/ 자동으로 확장 시 부하분산 기능		5	5	- 수동확장은 CLI를 통해 가능함 (kubectl scale --replicas=10 deployment/nginx-to-scaleout) - 자동확장은 CLI로 설정한 CPU 사용량을 기준으로 가능함 (kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10) -Kubernetes 기능으로 CPU 사용량을 기준으로 자동 부하분산이 가능함 - node의 최소, 최대 개수를 정할 수 있음
	컨테이너 확장 시 부하조정	컨테이너 확장 시 부하 비중 비율 조정 가능 여부		4	4	- service와 label selector를 이용하여 가능하게 함
	Container to Container 통신	컨테이너 간 통신 설정		3	0	- 서비스 단위로 통신 가능
	route 관리	사용자 정의 route 등록/변경/삭제		2	1	- Route53을 이용해 Route추가가 가능함
	사용자 인증서 추가	사용자가 Route에 별도의 인증서 등록, 관리 기능		3	2	- Helm을 이용해 ngnix ingress를 설치 후 Kubernetes secrete 생성 -> 어플리케이션 배포 후 YAML에 tls 정의하여 사용가능
	어플리케이션 health check	http/port/process 기반의 Health Check 기능		2	2	- Kubernetes의 LivenessProbe, ReadnessProbe를 설정하여 Pod의 Health Check가 가능함

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
어플리케이션 개발	로컬에서 Eclipse Plug-in을 이용한 개발	이클립스 기반의 개발자 도구가 제공 여부	15	2	0	- 제공되지 않음
	로컬에서 원격 debugging	로컬 환경에서 원격 디버깅 지원 여부		3	0	- 제공되지 않음
	컨테이너 실시간 로그 분석	컨테이너(Pod) 내 실시간 로그 확인		3	3	- CLI를 통해 확인이 가능함
	컨테이너 SSH 접속	컨테이너(Pod)에 SSH 접속 가능 여부 확인		3	3	- 기능확인함 (CLI를 통해 확인이 가능함)
	파이프라인 생성/수정	파이프라인 생성/수정 확인 (Jenkins, Spinnaker, Concourse 등 활용 가능 여부 확인)		4	2	- AWS CodePipeline을 이용하여 Git에 연동해 Pipeline을 구성하여 사용할 수 있음 - Jenkins 활용 여부에 대해 확인이 필요함
어플리케이션 성능/장애 분석	컨테이너 과거 로그 분석	프로젝트/Pod/서비스 단위의 로그 분석 기능	15	3	2	- AWS Cloudwatch에 컨테이너 로그 수집은 수작업 구성 필요함 - Cloudwatch 수집된 로그 데이터는 AWS Elastic Search, Fluentd, Kibana를 설정을 통해 로그를 관리할 수 있음 - Yaml 파일을 이용해 Fluentd, Kibana 배포가 가능함
	Java 분석 기능 (Jconsole, Dump 수집)	Jconsole을 이용한 모니터링 가능 여부 Java의 Heap, Thread Dump 생성 및 추출 가능 여부		2	2	- Java에서 제공해주는 기본 분석 기능 사용이 가능함

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	컨테이너 자원 모니터링	Pod별 CPU, 메모리, 디스크 사용량에 대한 모니터링		5	3	- Cluster 자원 모니터링은 CloudWatch를 통해 확인이 가능함 - Grafana, Prometheus를 이용하여 설치 후 자원 모니터링이 가능함
	어플리케이션 호출 분석	MSA 서비스간 call trace 분석 가능 여부		2	0	- 지원하지 않음 - 사용자 zipkin을 추가적으로 설치하고 구성해야 함
	사용자를 위한 알람 기능 제공	플랫폼 사용자 관점의 알람 기능 제공 여부		3	1	-기본제공 하지 않으며 Prometheus를 활용하여 제공 가능
프로젝트 관리	개발자 사용을 위한 GUI 제공	개발자용 웹콘솔 확인	15	1	1	- 웹콘솔이 제공됨
	CLI 설치 편의성	CLI 환경 구성 시 설치 편의 및 자동화 여부		1	0	- aws cli, aws-iam-authenticator, kubectl, eksctl 설치가 필요함 - Cloud9을 사용할 경우 aws cli와 docker는 설치하지 않아도 됨
	사용자 관리	사용자 관리 기능 및 편의성 LDAP, SSO 등과의 연동을 통한 외부 사용자 정보를 이용해 일괄 사용자 등록이 가능한지에 대한 확인		2	2	- AWS 사용자 직접 추가가 가능함 - 외부 자격 연동을 하여 조직 사용자들에 대한 권한 관리가 가능함. 한 예로 조직에서 SAML 2.0을 사용하는 경우 LDP와 연동하여 사용이 가능함, SAML을 사용하여 사용자에게 AWS Management 콘솔에 대한 연동 Single-Sing On(SSO) 또는 AWS API를 호출하기 위한 연동 액세스를 제공

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
	권한 관리	리소스별/ 사용자별 권한 관리		3	3	- AWS내 모든 동작이 사용자 Rule, Policy 기반으로 동작하며, Kubernetes 시작 전에 적절한 권한 셋업이 필수적임 - EKS와 관련된 Role은 AmazonEKSClusterPolicy, AmazonEKSServicePolicy가 제공됨 - 필요에 따라 여러 Role을 조합하여 새로운 Role을 만드는 것도 가능함
	프로젝트 생성/ 삭제	프로젝트 생성 및 프로젝트 관리자 할당		2	0	- 지원 안함 - EKS 서비스 접근 수준의 계정관리이며 프로젝트 생성/삭제등의 관리자 권한 지원 안함
	프로젝트 내에서 하위그룹 관리	개발, 테스트, 운영 등 개발단계별 관리		1	0	- 제공되지 않음
	Quota 관리	CPU, MEMORY, Container수, Route 수 제한		2	2	- Kubernetes 기능으로 Quota 설정이 가능함 Compute Resource - CPU, Memory, Pod 개수와 Object 개수 - configmap, PVC, secrets, services, LB 수를 설정할 수 있음
	Instance App Template 생성	사용자 정의 Application Template 생성이 가능한가 이미지와 템플릿 사이의 workflow 차이 확인		3	2	- Helm을 이용하여 가능함

구분	항목	설명	배점	가중치	Amazon Elastic Container Service for Kubernetes	
					평가결과	평가의견
보안/네트워킹	프로젝트 단위 보안 설정	프로젝트간 통신 설정이 가능한지 확인 (Multi tenency Enabled)	10	2	1	- Namespace를 이용하여 프로젝트 개념으로 사용이 가능함
	컨테이너 그룹 단위 보안 설정	컨테이너 그룹을 정의하고 그룹간 통신 설정이 가능한지 확인		2	0	- 지원안됨
	지정 컨테이너 단위 보안 설정	컨테이너별 통신 설정이 가능한지 확인		2	1	- Istio를 사용하여 Service 단위로 보안 설정을 제어할 수 있음
	특정 Role로 통신 제한(inbound)	특정 APP 이 외부의 정해진 IPs, PORT 에 대해서만 inbound 설정 가능한지 확인		2	0	- 지원안됨 - 방화벽을 통해서 차단하거나, istio를 사용하여 Cookie 정보를 이용하여 접근 권한을 제어할 수 있음
	특정 Role로 통신 제한(outbound)	특정 APP 이 외부 연계를 위해 특정 IPs, PORT 로 outbound 가능하도록 설정 가능한지 확인		2	2	- Network Policy를 지원함

- 테스트 결과 요약(see page 16)
- 테스트 결과 상세(see page 16)

4 EKS 4. 도입/구축 시 고려사항

4.1 서비스 리전

EKS는 현재 시점('18년 10월)에 3개 리전에서 서비스가 되고 있으며, 아직 한국에서는 서비스가 되고 있지 않다.

- 미국 동부(버지니아 북구)
- 미국 서부(오레곤)
- EU(아일랜드)

4.2 비용 산정

4.2.1 Amazon ECR(Elastic Container Registry)

Amazon Elastic Container Registry는 Repository에 저장한 데이터 용량과 인터넷으로 전송한 데이터 양에 대한 네트웍 사용량 비용으로 구성됨. Amazon ECR 프리 티어에는 매월 500MB의 스토리지 사용료까지는 무료이며 500MB 이상 사용량만 과금되며 스토리지 사용료는 GB당 0.10$/USD 입니다.

- Amazon ECR 가격표[1]

4.2.2 EKS(Elastic Container Service for Kubernetes)

GKE/AKS와 달리 EKS는 클러스터 노드에 대한 비용이 청구되며 시간당 0.20$/USD가 과금 됩니다. 또한 Worker 노드의 사용료는 GKE/AKS와 마찬가지로 EC2 인스턴스 사용료 및 EBS 스토리지 사용료로 과금 됩니다.

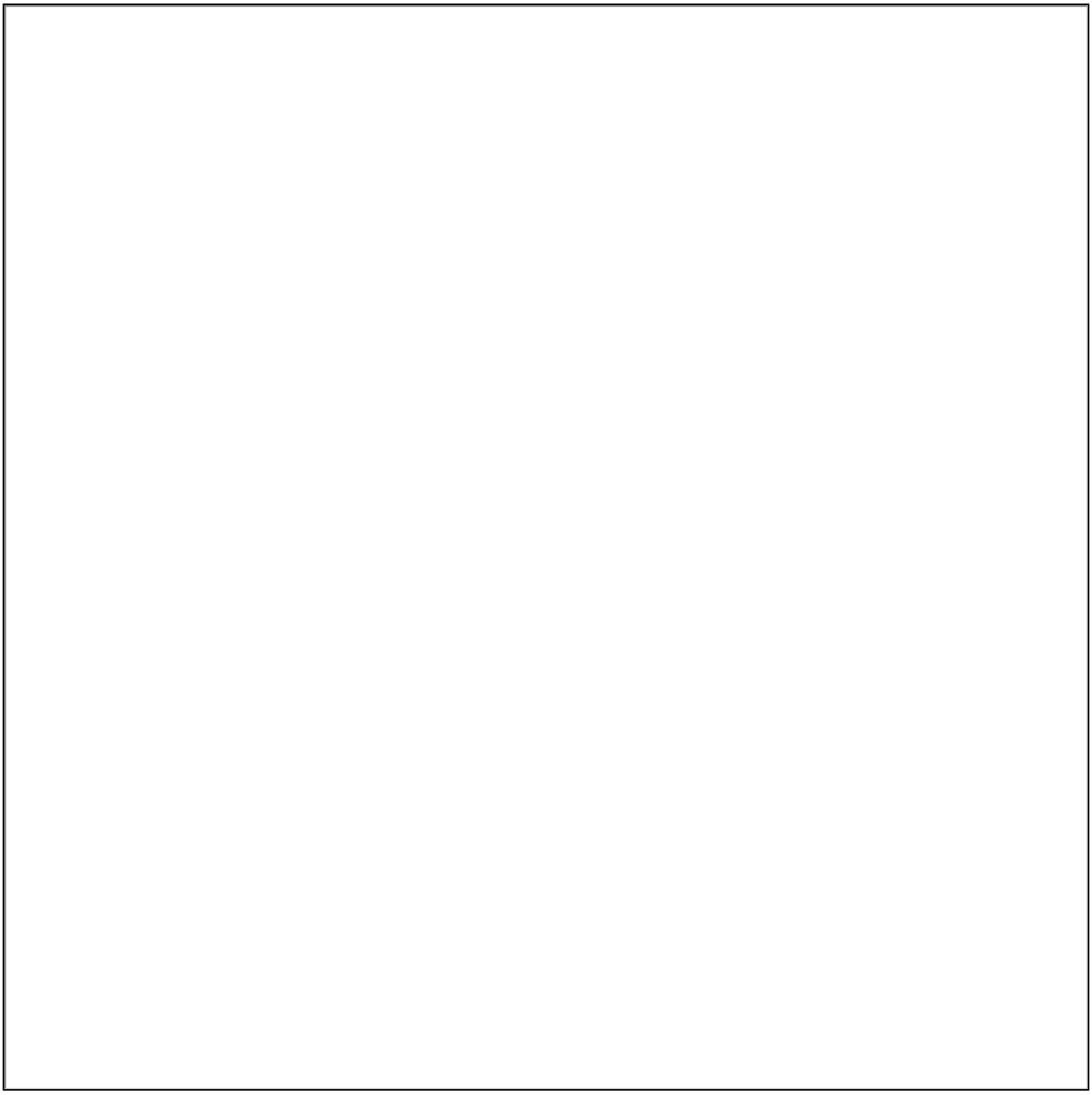
- EKS 가격표[2]
- EC2 가격표[3]

4.3 사용자 Role 설정에 대한 관리 방안

1 https://aws.amazon.com/ko/ecr/pricing/
2 https://aws.amazon.com/ko/eks/pricing/
3 https://aws.amazon.com/ko/ec2/pricing/

4.3.1 EKS Master node의 Role

EKS Master node의 Role은 EKS Cluster 생성단계에서 생성할 수 있으며, 기본적으로 AmazonEKSServicePolicy, AmazonEKSClusterPolicy 두 개의 Policy가 할당된다.



두 개의 Policy의 리소스별 권한은 아래에서 확인할 수 있다.

AmazonEKSClusterPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:UpdateAutoScalingGroup",
                "ec2:AttachVolume",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateRoute",
                "ec2:CreateSecurityGroup",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:DeleteRoute",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteVolume",
                "ec2:DescribeInstances",
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVolumes",
                "ec2:DescribeVolumesModifications",
                "ec2:DescribeVpcs",
                "ec2:DetachVolume",
                "ec2:ModifyInstanceAttribute",
                "ec2:ModifyVolume",
                "ec2:RevokeSecurityGroupIngress",
                "elasticloadbalancing:AddTags",
                "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
                "elasticloadbalancing:AttachLoadBalancerToSubnets",
                "elasticloadbalancing:ConfigureHealthCheck",
                "elasticloadbalancing:CreateListener",
                "elasticloadbalancing:CreateLoadBalancer",
                "elasticloadbalancing:CreateLoadBalancerListeners",
                "elasticloadbalancing:CreateLoadBalancerPolicy",
                "elasticloadbalancing:CreateTargetGroup",
                "elasticloadbalancing:DeleteListener",
                "elasticloadbalancing:DeleteLoadBalancer",
                "elasticloadbalancing:DeleteLoadBalancerListeners",
                "elasticloadbalancing:DeleteTargetGroup",
                "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
                "elasticloadbalancing:DeregisterTargets",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeLoadBalancerAttributes",
                "elasticloadbalancing:DescribeLoadBalancerPolicies",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeTargetGroupAttributes",
                "elasticloadbalancing:DescribeTargetGroups",
```

```
                "elasticloadbalancing:DescribeTargetHealth",
                "elasticloadbalancing:DetachLoadBalancerFromSubnets",
                "elasticloadbalancing:ModifyListener",
                "elasticloadbalancing:ModifyLoadBalancerAttributes",
                "elasticloadbalancing:ModifyTargetGroup",
                "elasticloadbalancing:ModifyTargetGroupAttributes",
                "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer",
                "elasticloadbalancing:SetLoadBalancerPoliciesOfListener",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "elasticloadbalancing.amazonaws.com"
                }
            }
        }
    ]
}
```

AmazonEKSServicePolicy

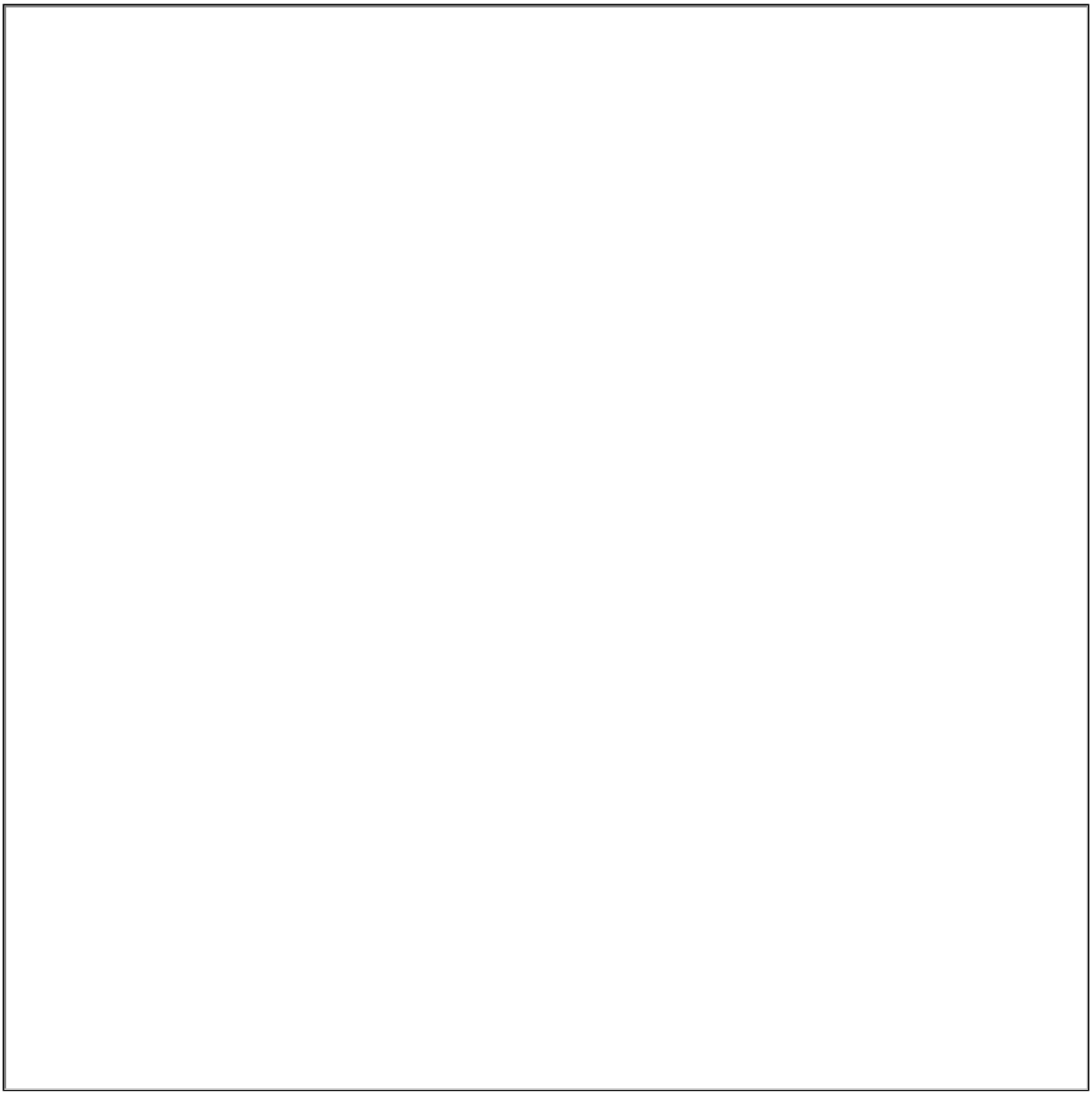
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeInstances",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:ModifyNetworkInterfaceAttribute",
                "iam:ListAttachedRolePolicies"
            ],
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:DeleteTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:vpc/*",
                "arn:aws:ec2:*:*:subnet/*"
            ]
        }
    ]
}
```

4.3.2 EKS Worker node의 Role

Worker Node의 Role은 eksctl을 통해 EKS Cluster 생성 시 자동 생성되며, Worker Node의 Role은 내부에서 생성되는 Pod에 상속되며 추가로 필요한 Policy는 추가로 IAM Policy를 통해 Attach 할 수 있다.

EKS Worker node에게 할당되는 Role은 AmazonEKSWorkerNodePolicy이다.



해당 Role의 Resource별 권한은 아래와 같다.

AmazonEKSWorkerNodePolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
```

```
                "ec2:DescribeInstances",
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVolumes",
                "ec2:DescribeVolumesModifications",
                "ec2:DescribeVpcs",
                "eks:DescribeCluster"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Pod 레벨의 Role은 지원되지 않으며 필요시 별도로 Kube2iam 플러그인을 통해 사용할 수 있다.

kube2iam 플러그인을 이용하면 어노테이션을 기반으로 Kubernetes 클러스터 내에서 실행되는 컨테이너에 IAM 자격 증명을 제공해준다. (https://github.com/jtblin/kube2iam)

4.4 EKS와 AWS CloudWatch와의 연계

AWS CloudWatch 서비스는 아마존 내 각종 리소스에 대한 모니터링 및 알림, 로깅 기능 사용을 제공하며, 해당 지표를 이용해 SNS, Lambda 등의 서비스와도 손쉽게 연동이 가능하다.

하지만, 아직 CloudWatch에서 EKS에 대한 지원은 되지 않는다. (로드맵 3개월 내)
추후 EKF와 Cloudwatch가 연계되면 Master node에 대한 통합 로깅, 컨테이너 모니터링, SNS를 연계한 알람 서비스 등을 사용할 수 있을 것이다.

현재는 Cloudwatch를 통해 확인할 수 있는 EKS 정보는 EC2에 생성된느 Worker Node에 대한 모니터링, 로깅만 가능하다.
하여, 각 Node별 Container 배포 현황 등의 컨테이너 레벨의 모니터링이 필요할 때에는 Kubernetes native dashboard, 또는 Prometheus, Grafana를 구성하여 확인해야한다.
SNS 연동이 필요한 때에는 Pod, Node 레벨의 로그를 FluentD를 통해 S3, Cloudwatch로 받은 후 해당 이벤트를 트리거하여 사용할 수 있다.