# 성능 효율성 원칙

AWS Well-Architected 프레임워크

2020 년 7 월

# 고지 사항

고객은 본 문서에 포함된 정보를 독립적으로 평가할 책임이 있습니다. 본 문서는 (a) 정보 제공만을 위한 것이며, (b)사전 고지 없이 변경될 수 있는 현재의 AWS 제품 제공 서비스 및 사례를 보여 주며, (c) AWS 및 자회사, 공급업체 또는 라이선스 제공자로부터 어떠한 약정 또는 보증도 하지 않습니다. AWS 제품 또는 서비스는 명시적이든 묵시적이든 어떠한 종류의 보증, 진술 또는 조건 없이 "있는 그대로" 제공됩니다. 고객에 대한 AWS 의 책임과 법적 책임은 AWS 계약서에 준하며 본 문서는 AWS 와 고객 간의 계약에 포함되지 않고 계약을 변경하지도 않습니다.

# 목차

# 개요

이 백서에서는 Amazon Web Services(AWS) Well-Architected 프레임워크의 성능 효율성 원칙을 중점적으로 다룹니다. 이 백서는 AWS 환경 설계, 제공 및 유지 관리에 모범 사례를 적용할 때 참조할 수 있는 지침을 제공합니다.

성능 효율성 원칙은 프로덕션 환경 관리를 위한 모범 사례를 다룹니다. 지속적 통합 또는 전달과 같은 비 프로덕션 환경 및 프로세스의 설계 및 관리는 이 백서에서 다루지 않습니다.

# 서문

AWS Well-Architected 프레임워크는 AWS 에서 워크로드를 구축할 때 내리는 의사 결정의 장점과 단점을 이해하는 데 도움이 됩니다. 이 프레임워크를 사용하면 클라우드에서 안정적이고 안전하며 효율적이고 경제적인 워크로드를 설계하고 운영하기 위한 설계 모범 사례를 알아볼 수 있습니다. 이 프레임워크는 모범 사례를 기준으로 아키텍처를 일관적으로 측정하고 개선 영역을 식별하는 방법을 제공합니다. 워크로드를 제대로 설계하면 비즈니스 성공 가능성이 높아집니다.

이 프레임워크에는 다음의 5 가지 원칙이 포함됩니다.

- 운영 우수성
- 보안
- 안정성
- 성능 효율성
- 비용 최적화

이 백서는 워크로드에 성능 효율성 원칙을 적용하는 데 중점을 둡니다. 기존 온프레미스 환경에서는 우수한 성능을 유지하기가 어렵습니다. 본 백서의 원칙을 사용하면 AWS 에서 시간이 지남에 따라 지속적인 성능을 효율적으로 제공하는 아키텍처를 구축하는 데 도움이 됩니다.

이 문서는 CTO(최고 기술 책임자), 아키텍트, 개발자, 운영 팀 팀원 등 기술 업무 담당자를 위해 작성되었습니다. 이 백서의 내용을 확인하고 나면 성능이 뛰어난 클라우드 아키텍처를 설계할 때 사용할 AWS 모범 사례와 전략을 파악할 수 있습니다. 이 백서는 구현 세부 정보나 아키텍처 패턴을 제공하지 않습니다. 하지만 적절한 리소스에 대한 참조가 포함되어 있습니다.

# 성능 효율성

성능 효율성 원칙에서는 컴퓨팅 리소스를 효율적으로 사용하여 요구 사항을 충족하고 수요 변경 및 기술 변화에 따라 이러한 효율성을 유지하는 방법을 중점적으로 알아봅니다.

# 설계 원칙

다음은 클라우드에서 워크로드 효율성을 달성하고 이를 유지하는 데 도움이 되는 설계 원칙입니다.

- **고급 기술의 대중화:** 팀에서 고급 기술을 손쉽게 구현할 수 있도록 복잡한 작업을 클라우드 공급업체에 위임합니다. IT 팀에 새로운 기술의 호스팅 및 실행에 대해 알아볼 것을 요청하는 대신 기술을 서비스 형태로 사용하는 것을 고려해보십시오. 예를 들어 NoSQL 데이터베이스, 미디어 트랜스코딩 및 기계 학습은 모두 전문 지식이 요구되는 기술입니다. 클라우드에서는 이러한 기술이 팀에서 사용할 수 있는 서비스 형식으로 제공되므로 팀은 리소스 프로비저닝 및 관리가 아닌 제품 개발에 집중할 수 있습니다.
- **몇 분 안에 전 세계에 배포:** 전 세계의 여러 AWS 리전에 워크로드를 배포하면 최소한의 비용으로 지연 시간을 줄이고 고객 경험을 개선할 수 있습니다.
- **서버리스 아키텍처 사용:** 서버리스 아키텍처에서는 물리적 서버를 실행하고 유지 관리하지 않고도 기존의 컴퓨팅 활동을 수행할 수 있습니다. 예를 들어 서버리스 스토리지 서비스를 정적 웹 사이트로 사용하고(웹 서버 불필요) 이벤트 서비스를 통해 코드를 호스팅할 수 있습니다. 이렇게 하면 물리적 서버 관리로 인한 운영 부담이 없어집니다. 또한 이러한 관리형 서비스는 클라우드 규모에서 운영되므로 트랜잭션 비용을 절감할 수 있습니다.
- **테스트 횟수 증가:** 자동화할 수 있는 가상 리소스를 활용하며 여러 가지 인스턴스, 스토리지 또는 구성에 대한 비교 테스트를 신속하게 수행할 수 있습니다.
- **기계적 조화 고려:** 목표에 가장 일치하는 기술 접근 방식을 사용합니다. 예를 들어 데이터베이스 또는 스토리지 접근 방식을 선택할 때는 데이터 액세스 패턴을 고려합니다.

# 정의

클라우드에서 성능 효율성을 달성하려면 다음 영역에 집중하십시오.

- 선택
- 검토
- 모니터링
- 절충

고성능 아키텍처 구축 시 데이터 기반 접근 방식을 취합니다. 개괄적 설계부터 리소스 유형 선택 및 구성에 이르는 아키텍처의 모든 측면에 대한 데이터를 수집합니다.

정기적으로 선택 사항을 검토함으로써 계속 진화하는 AWS 클라우드를 최대한 활용할 수 있습니다. 모니터링을 수행하면 예상 성능과의 차이를 인식할 수 있습니다. 압축 또는 캐싱을 사용하거나 일관성 요구 사항을 완화하는 등의 절충을 통해 아키텍처를 설계하면 성능을 개선할 수 있습니다.

# 선택

특정 워크로드에 대한 최적의 솔루션은 다양하며 종종 여러 접근 방식이 결합된 솔루션을 사용하게 됩니다. Well-Architected 워크로드의 경우 다수의 솔루션이 사용되며 다양한 특성을 사용하여 성능을 높일 수 있습니다.

AWS 리소스는 다양한 유형과 구성으로 제공되므로 요구 사항에 가장 근접한 접근 방식을 쉽게 찾을 수 있습니다. 또한 온프레미스 인프라에서는 쉽게 사용할 수 없는 옵션도 제공됩니다. 예를 들어 Amazon DynamoDB 와 같은 관리형 서비스는 완전관리형 NoSQL 데이터베이스로, 어떤 규모에서도 지연 시간이 한 자릿수 밀리초 단위로 매우 짧습니다.

## 성능 아키텍처 선택

워크로드에서 최적의 성능을 얻으려면 여러 가지 방식이 필요합니다. Well-Architected 시스템은 여러 개의 솔루션을 사용하고 다양한 특성을 사용하여 성능을 높입니다.

데이터 기반 접근 방식으로 아키텍처에 적합한 패턴 및 구현을 선택하면 경제적인 솔루션을 구축할 수 있습니다. AWS 솔루션스 아키텍트, AWS 참조 아키텍처 및 APN(AWS 파트너 네트워크) 파트너는 업계 지식을 바탕으로 사용자가 아키텍처를 선택하는 과정을 도울 수 있습니다. 하지만 아키텍처를 최적화하기 위해서는 벤치마킹 또는 로드 테스트를 통해 획득한 데이터가 필요합니다.

아키텍처는 여러 아키텍처 접근 방식(예: 이벤트 기반, ETL 또는 파이프라인)을 결합하게 될 가능성이 높습니다. 아키텍처 구현에는 아키텍처 성능 최적화에 적합한 AWS 서비스를 사용하게 됩니다. 다음 섹션에서는 고려할 네 가지 주요 리소스 유형(컴퓨팅, 스토리지, 데이터베이스, 네트워크)에 대해 살펴봅니다.

**사용 가능한 서비스 및 리소스 파악:** 클라우드에서 사용 가능한 광범위한 서비스 및 리소스를 알아보고 이해합니다. 워크로드와 관련이 있는 서비스 및 구성 옵션을 확인하고, 성능을 최적화하는 방법을 파악합니다.

기존 워크로드를 평가하는 경우에는 워크로드에 사용되는 다양한 서비스 리소스의 인벤토리를 생성해야 합니다. 인벤토리는 관리형 서비스 및 최신 기술로 교체 가능한 구성 요소를 평가하는 데 도움이 됩니다.

**아키텍처 선택 프로세스 정의:** 클라우드에 대한 내부 경험과 지식을 사용하거나 게시된 사용 사례, 관련 설명서 또는 백서 등의 외부 리소스를 사용하여 리소스 및 서비스를 선택하는

프로세스를 정의합니다. 워크로드에 사용할 수 있는 서비스를 사용한 테스트와 벤치마킹을 장려하는 프로세스를 정의해야 합니다.

아키텍처에 대한 중요한 사용자 사례를 작성할 때는 각 중요 사례를 실행해야 하는 속도 등의 성능 요구 사항을 포함해야 합니다. 이러한 중요 사례에서는 스크립트로 작성된 추가 사용자 작업 과정을 구현하여 요구 사항에 따른 이러한 사례의 성능을 파악해야 합니다.

**비용 요구 사항을 고려한 의사 결정:** 워크로드에는 워크로드 운영에 대한 비용 요구 사항이 있는 경우가 많습니다. 내부 비용 제어 기능을 사용하여 예상되는 리소스 요구 사항에 적합한 유형 및 크기의 리소스를 선택하십시오.

관리형 데이터베이스, 인메모리 캐시 및 기타 서비스와 같은 완전관리형 서비스로 대체할 수 있는 워크로드 구성 요소를 결정하십시오. 운영 워크로드를 줄이면 비즈니스 결과에 리소스를 집중시킬 수 있습니다.

비용 요구 사항 모범 사례는 비용 최적화 원칙 백서의 비용 효율적인 리소스 섹션을 참조하십시오.

**정책 또는 참조 아키텍처 사용:** 내부 정책 및 기존 참조 아키텍처를 평가하고 분석을 사용하여 워크로드에 사용할 서비스 및 구성을 선택하면 성능 및 효율성을 극대화할 수 있습니다.

**기존 클라우드 공급자 또는 적절한 파트너의 지침 사용:** 클라우드 회사 리소스(예: 솔루션스 아키텍트, 전문 서비스 또는 적절한 파트너)를 의사 결정의 지침으로 사용하십시오. 이러한 리소스는 최적의 성능을 위해 아키텍처를 검토하고 개선하는 데 도움이 될 수 있습니다.

추가 지침 또는 제품 정보가 필요한 경우 AWS 에 문의하여 지원을 받으십시오. AWS 솔루션스 아키텍트 및 AWS Professional Services 는 솔루션 구현에 대한 지침을 제공합니다. APN 파트너는 비즈니스 민첩성 및 혁신을 달성하는 데 도움이 되는 AWS 전문 지식을 제공합니다.

**기존 워크로드 벤치마크:** 기존 워크로드의 성능을 벤치마크하여 클라우드에서의 성능을 파악합니다. 이러한 벤치마크에서 수집된 데이터를 사용하면 아키텍처를 원활하게 결정할 수 있습니다.

통합 테스트에서 벤치마킹을 사용하여 워크로드 구성 요소의 성능에 관한 데이터를 생성합니다. 벤치마킹은 대개 로드 테스트보다 빠르게 설정할 수 있으며, 특정 구성 요소의 기술을 평가할 때 사용됩니다. 벤치마킹은 새 프로젝트를 시작할 때 로드 테스트를 위한 완전한 솔루션이 부족한 경우 종종 사용됩니다.

사용자 지정 벤치마크 테스트를 직접 작성할 수도 있고 TPC-DS 등의 업계 표준 테스트를 사용하여 데이터 웨어하우징 워크로드를 벤치마킹할 수도 있습니다. 산업 벤치마크는 여러

환경을 비교할 때 유용합니다. 사용자 지정 벤치마크는 아키텍처 내에서 수행될 것으로 예상되는 특정 작업 유형을 타게팅하는 데 유용합니다.

벤치마킹을 사용할 때는 유효한 결과를 얻을 수 있도록 테스트 환경을 사전 준비하는 것이 중요합니다. 동일한 벤치마크를 여러 번 실행하여 시간대별 차이를 파악하십시오.

벤치마크는 대개 로드 테스트보다 더 빠르게 실행되므로 배포 파이프라인 초기에서 성능 편차 관련 피드백을 더 빠르게 제공하려는 경우에 사용할 수 있습니다. 구성 요소나 서비스의 큰 변화를 평가할 때 벤치마킹을 수행하면 변경 작업의 타당성을 빠르게 확인할 수 있습니다. 벤치마킹은 로드 테스트와 함께 사용해야 합니다. 로드 테스트에서는 프로덕션 환경의 워크로드 성능에 대한 정보를 얻을 수 있기 때문입니다.

**워크로드에 대한 로드 테스트:** 다양한 리소스 유형 및 크기의 최신 워크로드 아키텍처를 클라우드에 배포합니다. 배포를 모니터링하여 병목 현상 또는 초과 용량을 식별하는 성능 지표를 캡처합니다. 이 성능 정보를 사용하여 아키텍처 및 리소스 선택을 설계하거나 개선할 수 있습니다.

로드 테스트에는 실제 워크로드가 사용되므로 프로덕션 환경에서 솔루션의 성능을 확인할 수 있습니다. 로드 테스트는 프로덕션 데이터의 통합 또는 제거 버전(민감한 정보 또는 식별 정보 제거)을 사용하여 실행되어야 합니다. 또한 대규모 워크로드를 통해 전체 아키텍처에서 진행되는 재생/사전 프로그래밍 방식의 사용자 작업 과정을 사용합니다. 전송 파이프라인의 일부로 로드 테스트를 자동으로 수행하고 결과를 미리 정의된 KPI 및 임계값과 비교합니다. 이렇게 하면 필요한 성능을 계속해서 달성할 수 있습니다.

Amazon CloudWatch 는 아키텍처의 리소스 전반에서 지표를 수집할 수 있습니다. 또한 사용자 지정 지표를 수집하고 게시하여 비즈니스 또는 파생 지표를 파악할 수도 있습니다. CloudWatch 를 사용하여 임계값이 초과되었음을 나타내는 경보를 설정합니다. 이 경보는 테스트가 기대 성능 범위를 벗어났음을 나타냅니다.

AWS 서비스를 사용하면 프로덕션 규모의 환경을 실행하여 적극적으로 아키텍처를 테스트할 수 있습니다. 테스트 환경이 필요할 때에만 비용을 지불하므로 온프레미스 환경을 사용하는 데 드는 비용의 일부로 필요한 모든 테스트를 수행할 수 있습니다. AWS Cloud 를 활용하여 워크로드를 테스트해 확장이 실패하지 않는지 또는 비선형 방식으로 확장되지 않는지 확인합니다. Amazon EC2 스팟 인스턴스를 사용하면 저렴한 비용으로 로드를 생성하고 병목 현상이 프로덕션 환경에서 발생하기 전에 미리 파악할 수 있습니다.

로드 테스트를 실행하는 데 시간이 오래 걸릴 때는 테스트 환경의 여러 사본을 사용하여 테스트를 병렬 처리합니다. 이 경우 비용은 비슷하지만 테스트 시간은 단축됩니다. 예를 들어 100 시간

동안 EC2 인스턴스 하나를 실행하는 경우와 1 시간 동안 인스턴스 100 개를 실행하는 경우의 비용은 동일합니다. 스팟 인스턴스를 사용하고 프로덕션에 사용하는 리전보다 비용이 저렴한 리전을 선택해 로드 테스트의 비용을 줄일 수도 있습니다.

로드 테스트 클라이언트의 위치는 최종 사용자가 분산되어 있는 지역을 반영해야 합니다.

## 리소스

로드 테스트와 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- Introducing The Amazon Builders' Library (DOP328)

**설명서**

- AWS 아키텍처 센터
- Amazon S3 성능 최적화
- Amazon EBS 볼륨 성능
- AWS CodeDeploy
- AWS CloudFormation
- CloudFront 로드 테스트
- AWS CloudWatch 대시보드

# 컴퓨팅 아키텍처 선택

특정 워크로드에 대한 최적의 컴퓨팅 선택은 애플리케이션 설계, 사용량 패턴 및 구성 설정에 따라 다를 수 있습니다. 아키텍처는 다양한 컴포넌트에 대해 서로 다른 컴퓨팅 옵션을 사용하고 다양한 기능을 활성화하여 성능을 개선할 수 있습니다. 아키텍처에 대해 잘못된 컴퓨팅 옵션을 선택하면 성능 효율성 저하로 이어질 수 있습니다.

**사용 가능한 컴퓨팅 옵션 평가:** 사용 가능한 컴퓨팅 관련 옵션의 성능 특성을 파악합니다. 인스턴스, 컨테이너 및 함수의 작동 방식과 이러한 컴퓨팅 기능이 워크로드에 제공하는 장단점을 확인합니다.

AWS 에서는 세 가지 형식의 컴퓨팅 기능(인스턴스, 컨테이너, 함수)이 제공됩니다.

## 인스턴스

인스턴스는 가상화된 서버이므로 버튼을 누르거나 API 를 호출하는 방법으로 기능을 변경할 수 있습니다. 클라우드에서는 리소스를 한번 결정하면 그대로 고정되는 것이 아니므로 다양한 서버 유형을 시험해 볼 수 있습니다. AWS 에서는 이러한 가상 서버 인스턴스를 다양한 제품군과 크기로 제공하며 SSD(Solid-State Drive)와 GPU(그래픽 처리 장치)를 비롯한 매우 다양한 기능을 제공합니다.

Amazon Elastic Compute Cloud(Amazon EC2) 가상 서버 인스턴스는 다양한 패밀리와 크기로 제공됩니다. 이러한 인스턴스는 SSD(Solid-State Drive) 및 GPU(그래픽 처리 장치)를 비롯하여 다양한 기능을 제공합니다. EC2 인스턴스를 시작할 때 지정하는 인스턴스 유형에 따라 인스턴스에 사용되는 호스트 컴퓨터의 하드웨어가 결정됩니다. 각 인스턴스 유형은 서로 다른 컴퓨팅, 메모리 및 스토리지 기능을 제공합니다. 인스턴스 유형은 이러한 기능에 따라 인스턴스 패밀리로 그룹화됩니다.

데이터를 사용해 워크로드에 가장 적합한 EC2 인스턴스 유형을 선택해야 하고, 네트워킹 및 스토리지 옵션이 정확한지 확인해야 하며, 워크로드 성능을 개선할 수 있는 운영 체제 설정을 고려해야 합니다.

## 컨테이너

컨테이너는 애플리케이션 및 종속성을 리소스가 격리된 프로세스에서 실행할 수 있는 운영 체제 가상화 방식입니다.

AWS 에서 컨테이너를 실행할 때는 두 가지를 선택할 수 있습니다. 첫째, 서버를 관리할지 여부를 선택합니다. AWS Fargate 는 컨테이너용 서버리스 컴퓨팅입니다. 컴퓨팅 환경의 설치, 구성 및 관리를 제어해야 한다면 Amazon EC2 를 사용할 수 있습니다. 둘째, 사용할 컨테이너 오케스트레이터를 선택합니다. Amazon Elastic Container Service(ECS) 또는 Amazon Elastic Kubernetes Service(EKS)를 선택할 수 있습니다.

Amazon Elastic Container Service(Amazon ECS)는 AWS Fargate 를 사용하여 EC2 인스턴스 또는 서버리스 인스턴스의 클러스터에서 컨테이너를 자동으로 실행하고 관리할 수 있는 완전관리형 컨테이너 오케스트레이션 서비스입니다. Amazon ECS 는 Amazon Route 53, Secrets Manager, AWS Identity and Access Management(IAM) 및 Amazon CloudWatch 와 같은 다른 서비스와 기본적으로 통합됩니다.

Amazon Elastic Kubernetes Service(Amazon EKS)는 완전관리형 Kubernetes 서비스입니다. AWS Fargate 를 사용하여 EKS 클러스터를 실행하도록 선택하면 서버 프로비저닝 및 관리 필요성이 제거됩니다. EKS 는 Amazon CloudWatch, Auto Scaling 그룹, AWS Identity and Access Management(IAM) 및 Amazon Virtual Private Cloud(VPC)와 같은 서비스와 긴밀하게 통합됩니다.

컨테이너 사용 시에는 데이터를 사용해 EC2 또는 AWS Fargate 인스턴스 유형을 선택하는 방식과 마찬가지로 데이터를 사용하여 워크로드에 가장 적합한 유형을 선택해야 합니다. 메모리, CPU, 테넌시 구성 등의 컨테이너 구성 옵션도 고려해야 합니다. 컨테이너 서비스 간 네트워크 액세스를 활성화하려면 서비스 통신 방식을 표준화하는 AWS App Mesh 와 같은 서비스 메시를 사용하는 것이 좋습니다. 서비스 메시는 종합적인 가시성을 제공하고 애플리케이션의 고가용성을 보장합니다.

## 함수

함수는 실행하려는 코드로부터 실행 환경을 추상화합니다. 예를 들어, AWS Lambda 를 이용하면 인스턴스를 실행하지 않고도 코드를 실행할 수 있습니다.

AWS Lambda 를 사용하면 관리 작업 없이 모든 유형의 애플리케이션 또는 백엔드 서비스에 대한 코드를 실행할 수 있습니다. 코드를 업로드하기만 하면 AWS Lambda 가 해당 코드를 실행하고 확장하는 데 필요한 모든 것을 관리합니다. 코드가 기타 AWS 서비스에서 자동으로 트리거되도록 설정할 수 있습니다. 코드는 직접 호출할 수도 있고 Amazon API Gateway 에서 사용할 수도 있습니다.

Amazon API Gateway 는 어떤 규모에서든 개발자가 API 를 손쉽게 생성, 게시, 유지 관리, 모니터링 및 보안할 수 있게 해주는 완전관리형 서비스입니다. Lambda 함수의 “프런트 도어”

역할을 하는 API 를 생성할 수 있습니다. API Gateway 는 트래픽 관리, 권한 부여 및 액세스 제어, 모니터링 및 API 버전 관리를 비롯하여 최대 수십만 건의 동시 API 호출을 수락하고 처리하는 데 관련된 모든 작업을 처리합니다.

AWS Lambda 를 통해 최적의 성능을 제공하려면 함수에 사용할 메모리의 양을 선택하십시오. 그러면 해당 메모리에 비례하는 CPU 성능과 기타 리소스가 할당됩니다. 예를 들어 256MB 메모리를 선택하면 128MB 메모리를 요청할 때에 비해 약 2 배의 CPU 파워가 Lambda 함수에 할당됩니다. 각 함수를 실행할 수 있는 시간(최대 300 초)을 제어할 수도 있습니다.

**사용 가능한 컴퓨팅 구성 옵션 파악:** 다양한 옵션을 통해 워크로드를 보완할 수 있는 방식과 시스템에 가장 적합한 구성 옵션을 파악합니다. 이러한 옵션의 예로는 인스턴스 패밀리, 크기, 기능(GPU, I/O), 함수 크기, 컨테이너 인스턴스, 단일/다중 테넌시 등이 있습니다.

인스턴스 패밀리 및 유형을 선택할 때는 워크로드 요구 사항을 충족하는 데 사용할 수 있는 구성 옵션도 고려해야 합니다.

- **GPU(그래픽 처리 장치)** — GPGPU(GPU 의 범용 컴퓨팅) 기능을 사용하는 경우 개발 프로세스에서 CUDA 와 같은 플랫폼을 활용함으로써 GPU 가 제공하는 높은 병렬 처리 수준의 이점이 적용되는 애플리케이션을 빌드할 수 있습니다. 워크로드에 3D 렌더링 또는 비디오 압축 기능이 필요한 경우 GPU 를 사용하면 하드웨어 가속 계산 및 인코딩이 가능하므로 워크로드의 효율성을 더욱 높일 수 있습니다.
- **FPGA(Field Programmable Gate Arrays)** — FPGA 를 사용하면 매우 까다로운 워크로드에 사용자 지정 하드웨어 가속 실행 기능을 사용함으로써 워크로드를 최적화할 수 있습니다. C, Go 등의 지원되는 일반 프로그래밍 언어나 Verilog, VHDL 등의 하드웨어 중심 언어를 활용해 알고리즘을 정의할 수 있습니다.
- **AWS Inferentia(Inf1)** — Inf1 인스턴스는 기계 학습 추론 애플리케이션을 지원하도록 구축되었습니다. Inf1 인스턴스를 사용하는 고객은 이미지 인식, 음성 인식, 자연어 처리, 개인화 및 사기 탐지와 같은 대규모 기계 학습 추론 애플리케이션을 실행할 수 있습니다. TensorFlow, PyTorch 또는 MXNet 과 같은 주요 기계 학습 프레임워크 중 하나에서 모델을 구축하고 P3 또는 P3dn 과 같은 GPU 인스턴스를 사용하여 모델을 훈련할 수 있습니다. 요구 사항을 충족하도록 기계 학습 모델을 훈련한 후에는 Inferentia 칩의 기계 학습 추론 성능을 최적화하는 컴파일러, 런타임 및 프로파일링 도구로 구성된 특수 SDK(소프트웨어 개발 키트)인 AWS Neuron 을 사용하여 Inf1 인스턴스에 모델을 배포할 수 있습니다.

- **버스트 가능 인스턴스 패밀리** — 버스트 가능 인스턴스는 적절한 기준 성능을 제공하는 동시에 워크로드에 필요한 더 높은 성능으로 버스트할 수 있는 기능을 제공하도록 설계되어 있습니다. 전체 CPU 를 자주/일관되게 사용하지는 않지만 가끔씩 버스트해야 하는 워크로드에는 이러한 인스턴스를 사용해야 합니다. 이러한 인스턴스는 웹 서버, 개발자 환경, 소형 데이터베이스 등의 범용 워크로드에 적합합니다. 이러한 인스턴스는 인스턴스가 적절한 성능을 제공해야 할 때 사용할 수 있는 CPU 크레딧을 제공합니다. 크레딧은 인스턴스에서 필요하지 않으면 누적됩니다.
- **고급 컴퓨팅 기능** — Amazon EC2 에서는 C-상태/P-상태 레지스터 관리, 프로세서 터보 부스트 제어 등의 고급 컴퓨팅 기능을 제공합니다. 또한 보조 프로세서에 액세스할 수 있으므로 AES-NI 를 통해 암호화 작업을 오프로드하거나 AVX 확장을 통해 고급 계산을 수행할 수 있습니다.

AWS Nitro System 은 전용 하드웨어와 경량 하이퍼바이저가 결합된 시스템으로, 혁신을 가속화하고 보안을 개선합니다. 사용 가능한 경우 AWS Nitro Systems 를 활용하여 호스트 하드웨어의 컴퓨팅 및 메모리 리소스를 완전히 사용할 수 있습니다. 또한 전용 니트로 카드를 사용하여 고속 네트워킹, 고속 EBS 및 I/O 가속화를 지원할 수 있습니다.

**컴퓨팅 관련 지표 수집:** 컴퓨팅 시스템 성능을 가장 효율적으로 파악하는 방법 중 하나는 다양한 리소스의 실제 사용률을 기록하고 추적하는 것입니다. 이 데이터를 사용하면 리소스 요구 사항을 보다 정확하게 파악할 수 있습니다.

워크로드(예: 마이크로서비스 아키텍처에서 실행되는 워크로드)에서는 지표, 로그 및 이벤트 형식의 데이터가 대량으로 생성될 수 있습니다. 이렇게 생성된 데이터를 기존 모니터링 및 관찰 서비스로 관리할 수 있는지 확인하십시오. Amazon CloudWatch 를 사용하면 AWS 및 온프레미스 서버에서 실행되는 모든 AWS 리소스, 애플리케이션 및 서비스의 단일 플랫폼에서 이러한 데이터를 수집, 액세스 및 상호 연관할 수 있으므로 시스템 전반의 가시성을 손쉽게 확보하고 문제를 신속하게 해결할 수 있습니다.

**적절한 크기 조정을 통해 필요한 구성 결정:** 워크로드의 다양한 성능 특성을 분석하고 이러한 특성과 메모리/네트워크/CPU 사용량 간의 관계를 분석합니다. 워크로드 프로필에 가장 적합한 리소스를 선택할 때 이 데이터를 사용할 수 있습니다. 예를 들어 데이터베이스와 같은 메모리 집약적 워크로드는 r-패밀리 인스턴스로 처리하는 것이 가장 좋습니다. 하지만 버스트 워크로드의 경우 탄력적인 컨테이너 시스템을 사용하는 것이 더 유리할 수 있습니다.

**사용 가능한 리소스 탄력성 사용:** 클라우드는 수요 변화에 맞춰 다양한 메커니즘을 통해 리소스를 동적으로 확장 또는 축소할 수 있는 유연성을 제공합니다. 컴퓨팅 관련 지표를 함께 활용하는 경우 워크로드가 변화에 자동으로 대응하여 목표를 달성하는 데 가장 적합한 리소스 세트를 활용할 수 있습니다.

수요에 가장 적합한 리소스를 공급하면 워크로드의 비용을 최대한 낮출 수 있습니다. 하지만 그와 동시에 프로비저닝 시간과 개별 리소스 장애를 고려해 리소스를 충분히 공급할 수 있는 계획도 마련해야 합니다. 수요는 고정되어 있을 수도 있고 변경될 수도 있으므로, 관리상의 부담을 방지하고 관리 작업에서 비용이 너무 많이 발생하지 않도록 하려면 적절한 지표와 자동화 기능이 필요합니다.

AWS 에서는 다양한 접근 방식을 사용하여 수요와 공급을 일치시킬 수 있습니다. 비용 최적화 원칙 백서에서는 다음과 같은 접근 방식을 사용하여 비용을 절감하는 방법을 설명합니다.

- 수요 기반 접근 방식
- 버퍼 기반 접근 방식
- 시간 기반 접근 방식

워크로드 배포에서 확장 및 축소 이벤트를 모두 처리할 수 있는지 확인해야 합니다. 축소 이벤트에 대한 테스트 시나리오를 생성하여 워크로드가 예상대로 작동하는지 확인하십시오.

**지표를 기준으로 컴퓨팅 요구 재평가:** 시스템 수준 지표를 사용하여 시간별 워크로드 동작 및 요구 사항을 파악합니다. 사용 가능한 리소스를 이러한 요구 사항과 비교해 워크로드 요구를 평가한 다음, 워크로드 프로필에 가장 적합하도록 컴퓨팅 환경을 변경합니다. 예를 들어 시스템을 계속 사용하다 보면 초기 예상보다 메모리가 더 많이 사용될 수 있으므로 다른 인스턴스 패밀리나 크기로 전환하면 성능과 효율성이 모두 개선될 수 있습니다.

## 리소스

컴퓨팅과 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- Amazon EC2 foundations (CMP211-R2)
- Powering next-gen Amazon EC2: Deep dive into the Nitro system
- Deliver high performance ML inference with AWS Inferentia (CMP324-R1)

- Optimize performance and cost for your AWS compute (CMP323-R1)
- Better, faster, cheaper compute: Cost-optimizing Amazon EC2 (CMP202-R1)

**설명서**

- 인스턴스:
  - 인스턴스 유형
  - EC2 인스턴스에 대한 프로세서 상태 제어
- EKS 컨테이너: EKS 작업자 노드
- ECS 컨테이너: Amazon ECS 컨테이너 인스턴스
- 함수: Lambda 함수 구성

# 스토리지 아키텍처 선택

특정 시스템에 대한 최적의 스토리지 솔루션은 액세스 방법의 종류(블록, 파일, 객체), 액세스 패턴(랜덤 또는 순차), 필요한 처리량, 액세스 빈도(온라인, 오프라인, 보관), 업데이트 빈도(WORM, 동적) 및 가용성과 내구성 제약 사항에 따라 다릅니다. Well-Architected 시스템은 여러 개의 스토리지 솔루션을 사용하고 다양한 특성을 사용하여 성능을 향상시킵니다.

AWS 에서 스토리지는 가상화되며 다양한 유형으로 제공됩니다. 따라서 요구 사항에 적합한 스토리지 방식을 보다 쉽게 찾고 온프레미스 인프라에서는 쉽게 얻을 수 없는 다양한 스토리지 옵션을 사용할 수 있습니다. 예를 들어, Amazon S3 는 99.999999999%의 내구성을 제공하도록 설계되었습니다. 또한 마그네틱 HDD(하드 디스크 드라이브)에서 SSD 로 변경할 수 있으며, 몇 초 안에 한 인스턴스에서 다른 인스턴스로 가상 드라이브를 쉽게 이동할 수 있습니다.

처리량, 초당 IOPS(입/출력 작업 수) 및 지연 시간을 파악하여 성능을 측정할 수 있습니다. 이러한 측정값 간의 관계를 파악하면 가장 적절한 스토리지 솔루션을 선택할 수 있습니다.

| 스토리지 | 서비스 | 지연 시간 | 처리량 | 공유 가능 |
|---|---|---|---|---|
| **블록** | Amazon EBS, EC2 인스턴스 스토어 | 가장 낮음, 일관적임 | 단일 | EC2 인스턴스에 탑재되며 스냅샷을 통해 복사됨 |
| **파일 시스템** | Amazon EFS, Amazon FSx | 낮음, 일관적임 | 다중 | 여러 클라이언트 |
| **객체** | Amazon S3 | 짧은 지연 시간 | 웹 규모 | 여러 클라이언트 |
| **아카이브** | Amazon S3 Glacier | 몇 분에서 몇 시간 사이 | 높음 | 아니요 |

지연 시간 측면에서 볼 때 인스턴스 하나만 데이터에 액세스하는 경우에는 Amazon EBS 등의 블록 스토리지를 사용해야 합니다. Amazon EFS 등의 분산 파일 시스템에서는 각 파일 작업 시의 지연 시간 오버헤드가 짧으므로 여러 인스턴스가 데이터에 액세스해야 하는 경우에는 이러한 시스템을 사용해야 합니다.

Amazon S3 에는 지연 시간을 줄이고 처리량을 늘릴 수 있는 기능이 있습니다. CRR(리전 간 복제)를 사용하면 여러 지리적 리전에서 지연 시간이 더 짧은 데이터 액세스 기능을 제공할 수 있습니다.

처리량 측면에서 볼 때 여러 스레드와 EC2 인스턴스에서 동시에 작업을 수행하는 등의 방식을 통해 고도로 병렬화된 워크로드를 지원하는 Amazon EFS 를 사용하면 초당 작업 수와 집계 처리량 수준을 높일 수 있습니다. Amazon EFS 에서는 벤치마크 또는 로드 테스트를 사용해 적절한 성능 모드를 선택합니다.

**스토리지 특성 및 요구 사항 파악:** 객체 스토리지, 블록 스토리지, 파일 스토리지 또는 인스턴스 스토리지 등 워크로드에 가장 적합한 서비스를 선택하는 데 필요한 다양한 특성(예: 공유 가능 여부, 파일 크기, 캐시 크기, 액세스 패턴, 지연 시간, 처리량, 데이터 지속성)을 파악합니다.

워크로드의 예상 증가율을 확인하고 해당 속도에 맞는 스토리지 솔루션을 선택합니다. Amazon S3 및 Amazon Elastic File System 과 같은 객체 및 파일 스토리지 솔루션은 무제한 스토리지를 지원합니다. Amazon EBS 의 스토리지 크기는 미리 결정되어 있습니다. 탄력적 볼륨을 사용하면 가동 중지나 성능 영향 없이 용량을 동적으로 늘리고 성능을 튜닝하며 최신 또는 기존의 현재 세대 볼륨의 유형을 변경할 수 있지만 이를 위해서는 OS 파일 시스템을 변경해야 합니다.

**사용 가능한 구성 옵션 평가:** 다양한 특성 및 구성 옵션과 스토리지와의 관련성을 평가합니다. 프로비저닝된 IOPS, SSD, 마그네틱 스토리지, 객체 스토리지, 아카이브 스토리지 또는 휘발성 스토리지를 사용하는 위치와 방법을 파악하여 워크로드의 성능과 스토리지 공간을 최적화합니다.

Amazon EBS 는 워크로드에 맞게 스토리지 성능과 비용을 최적화할 수 있는 광범위한 옵션을 제공합니다. 이러한 옵션은 두 가지 주요 범주로 구분됩니다. 그중 하나는 데이터베이스 및 부트 볼륨과 같은 트랜잭션 워크로드용 SSD 지원 스토리지이고(주로 IOPS 에 따라 성능이 달라짐), 다른 하나는 MapReduce 및 로그 처리와 같은 처리량이 높은 워크로드용 HDD 지원 스토리지입니다(주로 초당 MB 에 따라 성능이 달라짐).

SSD 지원 볼륨에는 지연 시간에 따라 달라지는 트랜잭션 워크로드용 최고급 프로비저닝된 IOPS SSD 와, 다양한 트랜잭션 데이터용으로 사용 가능하도록 가격과 성능이 적절하게 절충된 범용 SSD 가 포함됩니다.

Amazon S3 Transfer Acceleration 을 사용하면 클라이언트와 S3 버킷 간의 장거리 파일 전송을 빠르게 수행할 수 있습니다. Transfer Acceleration 은 전 세계에 분산된 Amazon CloudFront 의 엣지 로케이션을 활용하여 최적화된 네트워크 경로를 통해 데이터를 라우팅합니다. GET 요청을 많이 수행하는 S3 버킷의 워크로드에는 CloudFront 가 포함된 Amazon S3 를 사용합니다. 큰 파일을 업로드할 때는 여러 부분이 동시에 업로드되는 멀티 파트 업로드를 사용하면 네트워크 처리량을 극대화할 수 있습니다.

Amazon Elastic File System(Amazon EFS)은 AWS 클라우드 서비스 및 온프레미스 리소스에 사용할 수 있는 간편하고 확장 가능하며 탄력적인 완전관리형 NFS 파일 시스템을 제공합니다. Amazon EFS 는 범용 성능 모드와 최대 I/O 성능 모드의 두 가지 성능 모드를 제공하여 다양한 클라우드 스토리지 워크로드를 지원합니다. 또한 버스트 처리량 및 프로비저닝된 처리량이라는 두 가지 처리량 모드 중에서 파일 시스템에 적합한 처리량 모드를 선택할 수 있습니다. 워크로드에 사용할 설정을 결정하려면 Amazon EFS 사용 설명서를 참조하십시오.

Amazon FSx 는 엔터프라이즈 워크로드를 위한 Amazon FSx for Windows File Server 와 고성능 워크로드를 위한 Amazon FSx for Lustre 의 두 가지 파일 시스템을 제공합니다. FSx 는 SSD 를 지원하며 빠르고 예측 가능하고 확장 가능하며 일관된 성능을 제공하도록 설계되었습니다. Amazon FSx 파일 시스템은 지속적인 고속 읽기 및 쓰기 속도와 일관되게 짧은 지연 시간의 데이터 액세스를 제공합니다. 워크로드의 요구 사항에 따라 필요한 처리량 수준을 선택할 수 있습니다.

**액세스 패턴과 지표를 기준으로 결정:** 워크로드의 액세스 패턴을 기준으로 스토리지 시스템을 선택하고, 워크로드에서 데이터에 액세스하는 방법을 결정하여 이러한 스토리지 시스템을 구성합니다. 블록 스토리지 대신 객체 스토리지를 선택하여 스토리지 효율성을 높이십시오. 선택한 스토리지 옵션을 데이터 액세스 패턴과 일치하도록 구성합니다.

데이터에 액세스하는 방식은 스토리지 솔루션의 성능에 영향을 줍니다. 따라서 성능을 극대화하려면 액세스 패턴에 가장 적합한 스토리지 솔루션을 선택하거나, 스토리지 솔루션에 따라 액세스 패턴을 변경하는 것이 좋습니다.

RAID 0 어레이를 생성하면 단일 볼륨에서 프로비저닝할 때보다 파일 시스템의 성능 수준을 더 높일 수 있습니다. 내결함성보다 I/O 성능이 더 중요할 때는 RAID 0 를 사용하는 것이 좋습니다. 예를 들어 데이터 복제가 이미 별도로 설정되어 있으며 사용 빈도가 매우 높은 데이터베이스의 경우에는 RAID 0 를 사용해야 합니다.

워크로드에 사용된 모든 스토리지 옵션에서 워크로드에 적절한 스토리지 지표를 선택합니다. 버스트 크레딧을 사용하는 파일 시스템을 사용하는 경우에는 이러한 크레딧 제한에 근접할 때 알림을 제공하는 경보를 생성합니다. 전체 워크로드 스토리지 상태를 보여주는 스토리지 대시보드를 생성해야 합니다.

Amazon EBS 또는 Amazon FSx 와 같이 크기가 고정된 스토리지 시스템의 경우, 전체 스토리지 크기 대비 사용된 스토리지의 양을 모니터링해야 하며, 임계값에 도달할 때 스토리지 크기를 늘릴 수 있는 자동화 기능을 생성해야 합니다.

## 리소스

스토리지와 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- Deep dive on Amazon EBS (STG303-R1)
- Optimize your storage performance with Amazon S3 (STG343)

**설명서**

- Amazon EBS:
  - Amazon EC2 스토리지
  - Amazon EBS 볼륨 유형
  - I/O 특성
- Amazon S3: 요청 속도 및 성능 고려 사항
- Amazon Glacier: Amazon Glacier 설명서
- Amazon EFS: Amazon EFS 성능
- Amazon FSx:
  - Amazon FSx for Lustre 성능
  - Amazon FSx for Windows File Server 성능

# 데이터베이스 아키텍처 선택

시스템에 대한 최적의 데이터베이스 솔루션은 가용성, 일관성, 파티션 허용 오차, 지연 시간, 내구성, 확장성, 쿼리 기능에 대한 요구 사항에 따라 다릅니다. 많은 시스템이 다양한 하위 시스템에 대해 서로 다른 데이터베이스 솔루션을 사용하고 서로 다른 기능을 활성화하여 성능을 개선합니다. 시스템에 대해 잘못된 데이터베이스 솔루션 및 기능을 선택하면 성능 효율성이 저하될 수 있습니다.

**데이터 특성 파악:** 워크로드에 포함된 데이터의 다양한 특성을 파악합니다. 예를 들어 워크로드에 트랜잭션이 필요한지 여부, 워크로드가 데이터와 상호 작용하는 방식, 성능 요구 사항 등을 확인할 수 있습니다. 이 데이터를 사용하여 가장 우수한 성능을 제공하는 워크로드용 데이터베이스 방식(예: 관계형 데이터베이스, NoSQL 키-값, 문서, 와이드 열, 그래프, 시계열 또는 인메모리 스토리지)을 선택합니다.

관계형, 키-값, 문서, 인메모리, 그래프, 시계열 및 원장 데이터베이스를 비롯하여 특별히 제작된 다양한 데이터베이스 엔진 중에서 선택할 수 있습니다. 특정 문제 또는 문제 그룹을 해결할 수 있는 최고의 데이터베이스를 선택할 수 있으므로 제한적이고 획일적인 범용 데이터베이스에서 벗어나 고객의 요구 사항을 충족하는 애플리케이션을 구축하는 데 집중할 수 있습니다.

관계형 데이터베이스는 미리 정의된 스키마와 스키마 간의 관계를 사용하여 데이터를 저장합니다. 이러한 데이터베이스는 ACID(원자성, 일관성, 격리, 내구성) 트랜잭션을 지원하고 참조 무결성과 강력한 데이터 일관성을 유지하도록 설계되었습니다. 많은 기존 애플리케이션, ERP(엔터프라이즈 리소스 계획), CRM(고객 관계 관리) 및 전자 상거래의 데이터가 관계형 데이터베이스를 사용하여 저장됩니다. 이러한 데이터베이스 엔진의 다수를 Amazon EC2 에서 실행하거나 AWS 관리형 데이터베이스 서비스인 Amazon Aurora, Amazon RDS 및 Amazon Redshift 중에서 하나를 선택할 수 있습니다.

키-값 데이터베이스는 대개 대량의 데이터를 저장 및 검색하는 일반적인 액세스 패턴에 최적화되어 있습니다. 이 데이터베이스는 동시 요청의 양이 매우 많은 경우에도 빠른 응답 시간을 제공합니다.

트래픽이 많은 웹 앱, 전자 상거래 시스템 및 게임 애플리케이션은 키-값 데이터베이스의 일반적인 사용 사례입니다. AWS 에서는 인터넷 규모의 애플리케이션을 위한 보안, 백업 및 복원, 인메모리 캐싱이 내장된 탁월한 내구력의 완전관리형 다중 리전 다중 마스터 데이터베이스인 Amazon DynamoDB 를 활용할 수 있습니다.

인메모리 데이터베이스는 데이터에 실시간으로 액세스해야 하는 애플리케이션에 사용됩니다. 이 데이터베이스는 데이터를 메모리에 직접 저장함으로써 밀리초 단위의 지연 시간도 허용되지 않는 애플리케이션에 마이크로초 단위의 지연 시간을 제공합니다. 애플리케이션 캐싱, 세션 관리, 게임 순위표 및 지리 공간 애플리케이션에 인메모리 데이터베이스를 사용할 수 있습니다. Amazon ElastiCache 는 Redis 또는 Memcached 와 호환되는 완전관리형 인메모리 데이터 스토어입니다.

문서 데이터베이스는 반구조화된 데이터를 JSON 유사 문서로 저장하도록 설계되었습니다. 이러한 데이터베이스는 개발자가 콘텐츠 관리, 카탈로그 및 사용자 프로필과 같은 애플리케이션을 신속하게 구축하고 업데이트하는 데 도움이 됩니다. Amazon DocumentDB 는 MongoDB 워크로드를 지원하는 완전관리형 문서 데이터베이스 서비스로, 탁월한 속도, 확장성 및 고가용성을 제공합니다.

와이드 열 스토어는 NoSQL 데이터베이스의 한 유형입니다. 테이블, 행 및 열을 사용하지만 관계형 데이터베이스와 달리 열의 이름과 형식은 동일한 테이블에서 행마다 다를 수 있습니다. 일반적으로 와이드 열 스토어는 대규모 산업 앱에서 장비 유지 관리, 플릿 관리 및 라우팅 최적화를 위해 사용됩니다. Amazon Managed Apache Cassandra 서비스는 와이드 열 확장성 및 고가용성을 갖춘 관리형 Apache Cassandra 호환 데이터베이스 서비스입니다.

그래프 데이터베이스는 고도로 연결된 그래프 데이터 세트 간에 밀리초 단위의 지연 시간으로 수백만 개의 관계를 탐색하고 쿼리해야 하는 애플리케이션을 위한 솔루션입니다. 많은 기업에서 사기 탐지, 소셜 네트워킹 및 추천 엔진에 그래프 데이터베이스를 사용합니다. 빠르고 안정적인 완전관리형 그래프 데이터베이스 서비스인 Amazon Neptune 을 사용하면 상호 연결성이 높은 데이터 세트를 활용하는 애플리케이션을 쉽게 구축하고 실행할 수 있습니다.

시계열 데이터베이스는 시간이 지남에 따라 변화하는 데이터에서 효율적으로 분석 정보를 수집, 통합 및 도출합니다. IoT 애플리케이션, DevOps 및 산업용 텔레메트리에 시계열 데이터베이스를 활용할 수 있습니다. Amazon Timestream 은 IoT 및 운영 애플리케이션을 위한 고속의 확장 가능한 완전관리형 시계열 데이터베이스 서비스입니다. 이 서비스를 사용하면 하루에 수조 건의 이벤트를 손쉽게 저장하고 분석할 수 있습니다.

원장 데이터베이스는 모든 애플리케이션에 대해 확장 가능하고 변경 불가능하며 암호화 방식으로 확인 가능한 트랜잭션 레코드를 유지하는 신뢰할 수 있는 중앙 집중식 기관으로 사용됩니다. 레코드 시스템, 공급망, 등록 및 심지어 은행 거래 시스템에 원장 데이터베이스가 사용되는 것을 볼 수 있습니다. Amazon Quantum Ledger Database(QLDB)는 신뢰할 수 있는

중앙 기관이 소유한 투명하고 변경 불가능하며 암호화 방식으로 확인 가능한 트랜잭션 로그를 제공하는 완전관리형 원장 데이터베이스입니다. Amazon QLDB 는 모든 애플리케이션 데이터 변경 사항을 추적하고 시간 경과에 따른 완전하고 확인 가능한 변경 기록을 유지합니다.

**사용 가능한 옵션 평가:** 워크로드 스토리지 메커니즘 선택 프로세스의 일환으로 사용 가능한 서비스 및 스토리지 옵션을 평가합니다. 데이터 저장용으로 지정된 서비스나 시스템을 사용하는 방법과 시기를 파악합니다. 또한 프로비저닝된 IOPS, 메모리/컴퓨팅 리소스 및 캐싱 등 데이터베이스 성능이나 효율성을 최적화하는 데 사용할 수 있는 구성 옵션을 확인합니다.

데이터베이스 솔루션에는 일반적으로 워크로드 유형에 맞게 최적화할 수 있는 구성 옵션이 있습니다. 벤치마킹 또는 로드 테스트를 사용하여 워크로드에 중요한 데이터베이스 지표를 식별합니다. 스토리지 최적화, 데이터베이스 수준 설정, 메모리, 캐시 등의 선택한 데이터베이스용 구성 옵션을 고려합니다.

워크로드에 대한 데이터베이스 캐싱 옵션을 평가합니다. 가장 일반적인 3 개의 데이터베이스 캐시 유형은 다음과 같습니다.

- **데이터베이스 통합 캐시:** 일부 데이터베이스(예: Amazon Aurora)는 데이터베이스 엔진 내에서 관리되고 연속 쓰기 기능을 기본적으로 포함된 통합 캐시를 제공합니다.
- **로컬 캐시:** 로컬 캐시는 자주 사용되는 데이터를 애플리케이션 내에 저장합니다. 이렇게 하면 데이터 검색 속도가 빨라지고 데이터 검색과 관련된 네트워크 트래픽이 제거되어 다른 캐싱 아키텍처보다 데이터 검색 속도가 빨라집니다.
- **원격 캐시:** 원격 캐시는 전용 서버에 저장되며 일반적으로 Redis 및 Memcached 와 같은 키/값 NoSQL 스토어를 기반으로 구축됩니다. 각 캐시 노드에 대해 초당 최대 100 만 개의 요청을 처리할 수 있습니다.

Amazon DynamodDB 워크로드의 경우 DynamoDB Accelerator(DAX)를 통해 완전관리형 인메모리 캐시를 사용할 수 있습니다. DAX 는 완전관리형 인메모리 캐시를 통해 대규모 테이블에 대한 빠른 읽기 성능을 제공하는 인메모리 캐시입니다. DAX 를 사용하면 DynamoDB 테이블의 읽기 성능을 최대 10 배까지 개선할 수 있습니다. 초당 수백만 건의 요청에서 읽기에 필요한 시간을 밀리초에서 마이크로초로 단축할 수 있습니다.

**데이터베이스 성능 지표 수집 및 기록:** 데이터베이스 성능과 관련된 성능 측정값을 기록하는 도구, 라이브러리 및 시스템을 사용합니다. 예를 들어 초당 트랜잭션 수, 속도가 느린 쿼리 또는

데이터베이스 액세스 시에 발생하는 시스템 지연 시간을 측정할 수 있습니다. 이 데이터를 사용하면 데이터베이스 시스템의 성능을 파악할 수 있습니다.

워크로드에서 수집 가능한 만큼의 데이터베이스 활동 지표를 계측합니다. 이러한 지표를 워크로드에서 직접 게시하거나 애플리케이션 성능 관리 서비스에서 수집해야 할 수 있습니다. AWS X-Ray 를 사용하면 마이크로서비스 아키텍처를 사용하여 구축된 것과 같은 프로덕션의 분산 애플리케이션을 분석하고 디버깅할 수 있습니다. X-Ray 추적에는 단일 구성 요소에 대한 모든 데이터 포인트를 캡슐화하는 세그먼트가 포함될 수 있습니다. 예를 들어 애플리케이션은 요청에 대한 응답으로 데이터베이스를 호출할 때 데이터베이스 호출 및 그 결과를 나타내는 하위 세그먼트와 함께 해당 요청에 대한 세그먼트를 생성합니다. 하위 세그먼트에는 쿼리, 사용된 테이블, 타임스탬프 및 오류 상태와 같은 데이터가 포함될 수 있습니다. 계측이 완료된 후에는 데이터베이스 지표에 대해 임계값 위반 시점을 나타내는 경보를 활성화해야 합니다.

**액세스 패턴을 기준으로 데이터 스토리지 선택:** 워크로드의 액세스 패턴을 기준으로 하여 사용할 서비스와 기술을 결정합니다. 예를 들어 트랜잭션에 필요한 워크로드에는 관계형 데이터베이스를 활용하거나, 처리량은 더 높지만 최종 처리량은 일정한 키-값 저장소(해당하는 경우)를 활용합니다.

**액세스 패턴 및 지표를 기준으로 데이터 스토리지 최적화:** 데이터의 저장 또는 쿼리 방법을 최적화하는 성능 특성 및 액세스 패턴을 사용하여 최적의 성능을 달성합니다. 인덱싱, 키 분산, 데이터 웨어하우스 설계, 캐싱 전략 등의 최적화가 시스템 성능이나 전반적인 효율성에 미치는 영향을 측정합니다.

## 리소스

데이터베이스와 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- AWS purpose-built databases (DAT209-L)
- Amazon Aurora storage demystified: How it all works (DAT309-R)
- Amazon DynamoDB deep dive: Advanced design patterns (DAT403-R1)

**설명서**

- AWS Database Caching
- AWS 의 클라우드 데이터베이스

- Amazon Aurora 모범 사례
- Amazon Redshift 성능
- Amazon Athena 10 가지 최상의 성능 팁
- Amazon Redshift Spectrum 모범 사례
- Amazon DynamoDB 모범 사례
- Amazon DynamoDB Accelerator

# 네트워크 아키텍처 선택

워크로드에 대한 최적의 네트워크 솔루션은 지연 시간, 처리량 요구 사항, 지터 및 대역폭에 따라 다릅니다. 위치 옵션은 사용자 또는 온프레미스 리소스와 같은 물리적 제약에 따라 결정됩니다. 엣지 로케이션 또는 리소스 배치를 통해 이러한 제약을 상쇄할 수 있습니다.

AWS 에서 네트워킹은 가상화되고 다양한 유형 및 구성으로 제공됩니다. 따라서 요구 사항에 일치하는 네트워킹 방법을 보다 쉽게 찾을 수 있습니다. AWS 는 네트워크 트래픽을 최적화하는 제품 기능(예: 향상된 네트워킹, Amazon EC2 네트워킹 최적화 인스턴스, Amazon S3 Transfer Acceleration, 동적 Amazon CloudFront)을 제공합니다. 또한 AWS 는 네트워크 거리 또는 지터를 줄이기 위한 네트워킹 기능(예: Amazon Route53 지연 시간 기반 라우팅, Amazon VPC 엔드포인트, AWS Direct Connect 및 AWS Global Accelerator)도 제공합니다.

**네트워킹이 성능에 미치는 영향 파악:** 네트워크 관련 기능이 워크로드 성능에 미치는 영향을 분석하고 파악합니다. 예를 들어 네트워크 지연 시간은 사용자 경험에 영향을 미치며 네트워크 용량을 충분히 제공하지 않으면 워크로드 성능에 병목 현상이 발생할 수 있습니다.

네트워크는 모든 애플리케이션 구성 요소 사이에 있기 때문에 애플리케이션 성능 및 동작에 긍정적, 부정적인 영향을 미칠 수 있습니다. 또한 클러스터 성능을 높이는 데 있어서 심층적인 네트워크 지식이 요구되는 HPC(고성능 컴퓨팅)와 같이 네트워크 성능에 크게 의존하는 애플리케이션도 있습니다. 따라서 대역폭, 지연 시간, 지터 및 처리량에 대한 워크로드 요구 사항을 결정해야 합니다.

**사용 가능한 네트워킹 기능 평가:** 클라우드에서 성능을 높일 수 있는 네트워킹 기능을 평가합니다. 테스트, 지표 및 분석을 통해 이러한 기능의 영향을 측정할 수 있습니다. 예를 들어 지연 시간, 네트워크 거리 또는 지터를 줄이는 데 사용할 수 있는 네트워크 수준 기능을 활용합니다.

일반적으로 네트워크 성능을 최적화하는 기능을 제공하는 서비스가 많습니다. 그러므로 네트워크 트래픽을 최적화하는 제품 기능(예: EC2 인스턴스 네트워크 기능, 강화된 네트워킹 인스턴스 유형, Amazon EBS 최적화 인스턴스, Amazon S3 Transfer Acceleration, 동적 CloudFront)을 고려해야 합니다.

AWS Global Accelerator 는 AWS 글로벌 네트워크를 사용하여 글로벌 애플리케이션 가용성 및 성능을 개선하는 서비스입니다. 이 서비스는 정체가 없는 방대한 AWS 글로벌 네트워크를 활용하여 네트워크 경로를 최적화합니다. DNS 구성을 업데이트하거나 클라이언트용 애플리케이션을 변경하지 않고도 가용 영역 또는 AWS 리전 간에 엔드포인트를 쉽게 이동할 수 있는 고정 IP 주소를 제공합니다.

외부 사용자는 Amazon S3 콘텐츠 가속 기능을 통해 CloudFront 의 네트워킹 최적화 이점을 활용하여 Amazon S3 에 데이터를 업로드할 수 있습니다. 이렇게 하면 AWS 클라우드 전용 연결을 사용할 수 없는 원격 위치에서 대량의 데이터를 전송할 수 있습니다.

최신 버전의 EC2 인스턴스는 강화된 네트워킹 기능을 활용할 수 있습니다. M5n 및 M5dn 과 같은 N 시리즈 EC2 인스턴스는 4 세대 사용자 지정 니트로 카드 및 Elastic Network Adapter(ENA) 디바이스를 활용하여 단일 인스턴스에 최대 100Gbps 의 네트워크 처리량을 제공합니다. 이러한 인스턴스는 기본 M5 인스턴스에 비해 4 배 더 높은 네트워크 대역폭과 패킷 프로세스를 제공하며 네트워크 집약적 애플리케이션에 적합합니다. 또한 특정 인스턴스 크기의 M5n 및 M5dn 인스턴스에서 Elastic Fabric Adapter(EFA)를 활성화하여 네트워크 지연 시간을 짧고 일관성 있게 유지할 수도 있습니다.

단일 배치 그룹 내의 인스턴스용으로 네트워크 용량 20Gbps 를 제공하는 Amazon Elastic Network Adapter(ENA)를 활용하면 성능을 더욱 최적화할 수 있습니다. Elastic Fabric Adapter(EFA)는 AWS 에서 높은 수준의 대규모 노드 간 통신이 필요한 워크로드를 실행할 때 사용할 수 있는 Amazon EC2 인스턴스용 네트워크 인터페이스입니다. EFA 를 사용하면 MPI(메시지 전달 인터페이스)를 사용하는 HPC(고성능 컴퓨팅) 애플리케이션과 NCCL(NVIDIA Collective Communications Library)을 사용하는 ML(기계 학습) 애플리케이션을 수천 개 CPU 또는 GPU 로 확장할 수 있습니다.

최적화된 구성 스택을 사용하는 Amazon EBS 최적화 인스턴스는 Amazon EBS I/O 전용 용량을 추가로 제공합니다. 이 최적화를 수행하면 Amazon EBS I/O 와 인스턴스의 기타 트래픽 간에 경합이 최소화되므로 EBS 볼륨의 성능을 최대한 높일 수 있습니다.

Amazon Route 53 용 LBR(지연 시간 기반 라우팅)을 사용하면 전 세계 고객을 대상으로 워크로드 성능을 개선할 수 있습니다. LBR 은 워크로드가 실행되는 다양한 AWS 리전의 실제 성능 측정값을 기준으로 하여 가장 빠른 환경을 제공하는 AWS 엔드포인트(예: EC2 인스턴스, 탄력적 IP 주소 또는 ELB 로드 밸런서)로 고객을 라우팅하는 방식으로 작동합니다.

Amazon VPC 엔드포인트는 인터넷 게이트웨이 또는 NAT(네트워크 주소 변환) 인스턴스가 없어도 Amazon S3 등의 AWS 서비스에 안정적으로 연결할 수 있는 기능을 제공합니다.

**하이브리드 워크로드에 적절한 규모의 전용 연결 또는 VPN 선택:** 온프레미스 통신에 대한 요구 사항이 있는 경우 대역폭이 워크로드 성능을 제공하기에 충분한지 확인합니다. 대역폭 요구 사항에 따라 단일의 전용 연결 또는 단일 VPN 으로는 충분하지 않을 수 있으며, 여러 연결 간에 트래픽 로드 밸런싱을 활성화해야 합니다.

하이브리드 워크로드에 대한 대역폭 및 지연 시간 요구 사항을 추정해야 합니다. AWS Direct Connect 또는 VPN 엔드포인트의 크기 요구 사항은 이러한 수치를 바탕으로 결정됩니다.

AWS Direct Connect 는 AWS 환경으로의 전용 연결(50Mbps~10Gbps)을 제공합니다. 그러므로 지연 시간을 관리/제어하고 대역폭을 프로비저닝할 수 있습니다. 따라서 워크로드에서 우수한 성능을 유지하면서 다른 환경에 쉽게 연결할 수 있습니다. AWS Direct Connect 파트너 중 하나를 활용하면 여러 환경에서 엔드 투 엔트 연결 기능을 사용할 수 있으므로 성능이 일관되게 유지되는 확장 네트워크가 제공됩니다.

AWS Site-to-Site VPN 은 VPC 를 위한 관리형 VPN 서비스입니다. VPN 연결을 생성하면 두 개의 다른 VPN 엔드포인트로 연결되는 터널이 제공됩니다. AWS Transit Gateway 를 사용하면 여러 VPC 간의 연결을 간소화하고 단일 VPN 연결을 통해 AWS Transit Gateway 에 연결된 모든 VPC 에 연결할 수 있습니다. 또한 AWS Transit Gateway 를 사용하면 여러 VPN 터널에서 ECMP(Equal Cost Multi-Path) 라우팅 지원을 활성화하여 1.25Gbps IPsec VPN 처리량 제한 이상으로 확장할 수 있습니다.

**로드 밸런싱 및 암호화 오프로드 활용:** 클라우드의 탄력성을 워크로드에 활용할 수 있도록 여러 리소스 또는 서비스에 트래픽을 분산합니다. 로드 밸런싱을 사용하여 암호화 종료를 오프로드하면 성능을 개선하고 트래픽을 효율적으로 관리/라우팅할 수 있습니다.

서비스 콘텐츠용으로 여러 인스턴스를 사용하려는 확장 아키텍처를 구현할 때는 Amazon VPC 내의 로드 밸런서를 활용할 수 있습니다. AWS 는 ELB 서비스에서 애플리케이션용으로 여러 모델을 제공합니다. HTTP 및 HTTPS 트래픽을 로드 밸런싱하는 데 가장 적합한 Application Load Balancer 는 마이크로서비스 및 컨테이너를 비롯한 최신 애플리케이션 아키텍처를 제공할 때 사용할 수 있는 고급 요청 라우팅 기능을 제공합니다.

Network Load Balancer 는 성능이 매우 우수해야 하는 TCP 트래픽 로드 밸런싱을 수행하려는 경우에 사용하면 가장 효율적입니다. 또한 지연 시간을 매우 짧게 유지하면서 초당 수백만 개의 요청을 처리할 수 있으며, 예상치 못한 휘발성 트래픽 패턴도 처리할 수 있도록 최적화되어 있습니다.

Elastic Load Balancing 이 제공하는 통합 인증서 관리 및 SSL/TLS 복호화를 활용하면 로드 밸런서의 SSL 설정을 중앙에서 유연하게 관리하고 CPU 집약적인 작업을 워크로드에서 오프로드할 수 있습니다.

**네트워크 트래픽을 최적화하는 네트워크 프로토콜 선택:** 워크로드 성능에 미치는 영향을 기준으로 시스템과 네트워크 간의 통신에 사용할 프로토콜을 결정합니다.

원하는 처리량을 달성하려면 지연 시간과 대역폭 간의 관계를 고려해야 합니다. 파일 전송에서 TCP 를 사용하는 경우 지연 시간이 길수록 전체 처리량이 줄어듭니다. 이 문제는 TCP 튜닝 및 최적화된 전송 프로토콜을 사용하여 해결되며 경우에 따라 UDP 를 사용하기도 합니다.

**네트워크 요구 사항에 따라 위치 선택:** 제공되는 클라우드 위치 옵션을 사용하여 네트워크 지연 시간을 줄이고 처리량을 개선합니다. AWS 리전, 가용 영역, 배치 그룹, 엣지 로케이션(예: Outposts, Local Zones 및 Wavelength)을 활용하여 네트워크 지연 시간을 줄이거나 처리량을 늘립니다.

AWS 클라우드 인프라는 리전 및 가용 영역을 중심으로 구축됩니다. 리전은 가용 영역이 여러 개 포함된 전 세계의 실제 위치입니다.

가용 영역은 하나 이상의 개별 데이터 센터로 구성됩니다. 각 데이터 센터는 분리된 시설에 구축되며 이중화된 전력, 네트워킹 및 연결 기능을 갖추고 있습니다. 이러한 가용 영역을 사용하면 단일 데이터 센터에서 기대할 수 있는 것보다 더 높은 가용성, 내결함성 및 확장성을 지닌 프로덕션 애플리케이션과 데이터베이스를 운영할 수 있습니다.

다음과 같은 주요 요소를 토대로 하여 배포용으로 적절한 리전을 하나 이상 선택합니다.

- **사용자의 위치**: 워크로드 사용자 근처의 리전을 선택하면 사용자가 워크로드를 사용할 때 지연 시간이 단축됩니다.
- **데이터의 위치**: 데이터를 많이 사용하는 애플리케이션에서는 데이터 전송 시 지연 시간 병목 현상이 가장 많이 발생합니다. 따라서 애플리케이션 코드는 최대한 데이터와 가까운 위치에서 실행되어야 합니다.
- **기타 제약 조건**: 보안 및 규정 준수 등의 제약을 고려해야 합니다.

Amazon EC2 에서는 네트워킹용 배치 그룹을 제공합니다. 배치 그룹은 단일 가용 영역 내에 있는 인스턴스의 논리적 그룹입니다. 지원되는 인스턴스 유형이 포함된 배치 그룹과 ENA(Elastic Network Adapter)를 사용하면 지연 시간이 짧은 25Gbps 네트워크에 워크로드를 연결할 수 있습니다. 네트워크 지연 시간이 짧거나 처리량이 높은 경우 또는 두 조건을 모두 충족하는 경우 성능이 개선되는 워크로드에는 배치 그룹을 사용하는 것이 좋습니다. 배치 그룹을 사용하면 네트워크 통신의 지터를 줄일 수 있습니다.

지연 시간이 중요한 서비스는 전 세계 엣지 로케이션 네트워크를 사용하여 엣지에서 전송됩니다. 이러한 엣지 로케이션에서는 보통 CDN(콘텐츠 전송 네트워크) 및 DNS(Domain Name System)와 같은 서비스를 제공합니다. 엣지에서 이러한 서비스를 사용함으로써 워크로드 지연 시간을 짧게 유지하면서 콘텐츠 또는 DNS 확인 요청에 응답할 수 있습니다. 이러한 서비스는 지리적 콘텐츠 타게팅(최종 사용자의 위치를 기준으로 각기 다른 콘텐츠 제공) 등의 지리적 서비스나 최종 사용자를 가장 가까운 리전으로 라우팅하는 지연 시간 기반 라우팅(지연 시간이 최소화됨)도 제공합니다.

이미지, 스크립트, 비디오 등의 정적 콘텐츠와 API, 웹 애플리케이션 등의 동적 콘텐츠를 모두 빠르게 전송하는 데 사용할 수 있는 글로벌 CDN 인 Amazon CloudFront 는 사용자에게 고성능 네트워크 연결을 제공하며 콘텐츠를 캐시하는 전 세계 엣지 로케이션 네트워크를 활용합니다. 또한 CloudFront 에서는 콘텐츠 업로딩 및 동적 애플리케이션과 같은 기타 여러 기능도 더 빠르게 사용할 수 있으므로, 인터넷을 통해 트래픽을 처리하는 모든 애플리케이션의 성능이 향상됩니다. Amazon CloudFront 의 기능 중 하나인 Lambda@Edge 를 사용하면 워크로드의 사용자와 가까운 위치에서 코드를 실행하여 성능을 개선하고 지연 시간을 줄일 수 있습니다.

Amazon Route 53 는 가용성과 확장성이 뛰어난 클라우드 DNS 웹 서비스입니다. 이 서비스는 www.example.com 과 같은 이름을 컴퓨터가 서로 연결하는 데 사용하는 IP 주소(예:192.168.2.1)로 변환하는 방식을 통해 최종 사용자를 인터넷 애플리케이션으로 라우팅하는 매우 안정적이면서도 경제적인 방식을 개발자와 기업에 제공합니다. 또한 Route 53 는 IPv6 와 완벽히 호환됩니다.

AWS Outposts 는 지연 시간 요구 사항으로 인해 온프레미스에 유지해야 하지만 AWS 의 나머지 워크로드와 함께 원활하게 실행되어야 하는 워크로드를 위해 설계되었습니다. AWS Outposts 는 AWS 에서 설계한 하드웨어로 구축된 완전관리형의 구성 가능한 컴퓨팅 및 스토리지 랙이며 온프레미스에서 컴퓨팅 및 스토리지를 실행하고 클라우드의 광범위한 AWS 서비스에 원활하게 연결할 수 있습니다.

AWS Local Zones 는 동영상 렌더링 및 그래픽 집약적인 가상 데스크톱 애플리케이션처럼 10 밀리초 미만의 지연 시간이 요구되는 워크로드를 실행하도록 설계된 새로운 유형의 AWS 인프라입니다. Local Zones 에서는 최종 사용자와 가까운 위치에 컴퓨팅 및 스토리지 리소스를 배치함으로써 제공되는 모든 이점을 실현할 수 있습니다.

AWS Wavelength 는 AWS 인프라, 서비스, API 및 도구를 5G 네트워크로 확장하여 지연 시간이 아주 짧은 애플리케이션을 5G 디바이스에 제공하도록 설계되었습니다. Wavelength 는 통신 공급자의 5G 네트워크 안에 스토리지와 컴퓨팅을 내장하여 IoT 디바이스, 게임 스트리밍, 자율 주행 차량 및 라이브 미디어 제작 등 10 밀리초 미만의 지연 시간이 요구되는 5G 워크로드를 지원합니다.

지연 시간을 줄이고 콘텐츠 캐싱을 활성화하려면 엣지 서비스를 사용합니다. 이러한 방식의 이점을 최대한 활용하려면 DNS 및 HTTP/HTTPS 용으로 캐시 제어를 올바르게 구성했는지 확인하십시오.

**지표를 기준으로 네트워크 구성 최적화:** 수집 및 분석된 데이터가 제공하는 정보를 사용하여 네트워크 구성 최적화를 결정합니다. 이러한 변경의 영향을 측정한 다음 영향 측정값을 활용해 향후 결정을 내립니다.

워크로드에 사용되는 모든 VPC 네트워크에 대해 VPC 흐름 로그를 활성화합니다. VPC 흐름 로그는 VPC 의 네트워크 인터페이스에서 전송되고 수신되는 IP 트래픽에 대한 정보를 포착하는 데 사용할 수 있는 기능입니다. VPC 흐름 로그는 특정 트래픽이 인스턴스에 도달하지 않는 이유를 해결하여 과도하게 제한적인 보안 그룹 규칙을 진단하는 것과 같은 다수의 작업을 수행하는 데 도움이 됩니다. 흐름 로그를 보안 도구로 사용하여 인스턴스에 도달하는 트래픽을 모니터링하고, 네트워크 트래픽을 프로파일링하고, 비정상적인 트래픽 동작을 찾을 수 있습니다.

워크로드가 변경되면 네트워킹 지표를 사용하여 네트워킹 구성을 변경합니다. 클라우드 기반 네트워크는 빠르게 재구축될 수 있으므로 성능 효율성을 유지하려면 네트워크 아키텍처를 지속적으로 변경해야 합니다.

## 리소스

네트워킹과 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- Connectivity to AWS and hybrid AWS network architectures (NET317-R1)
- Optimizing Network Performance for Amazon EC2 Instances (CMP308-R1)

**설명서**

- Amazon Route 53 에서 지연 시간 기반 라우팅으로 이전
- AWS 의 네트워킹 제품
- EC2
  - Amazon EBS 최적화 인스턴스
  - Linux 기반 EC2 향상된 네트워킹
  - Windows 의 EC2 향상된 네트워킹
  - EC2 배치 그룹
  - Linux 인스턴스에서 Elastic Network Adapter(ENA)를 사용하여 향상된 네트워킹 활성화
- VPC
  - Transit Gateway
  - VPC 엔드포인트
  - VPC 흐름 로그
- Elastic Load Balancer
  - Application Load Balancer
  - Network Load Balancer

# 검토

워크로드 설계 시 선택할 수 있는 옵션은 한정되어 있습니다. 그러나 시간이 지나면 워크로드의 성능을 개선할 수 있는 새로운 기술과 접근 방식을 사용할 수 있게 됩니다. 클라우드에서는 인프라가 코드로 되어 있으므로 새 기능과 서비스를 훨씬 쉽게 사용해 볼 수 있습니다.

아키텍처에 데이터 기반 방식을 도입하려면 다음 사항을 고려하는 성능 검토 프로세스를 구현해야 합니다.

- **코드형 인프라:** AWS CloudFormation 템플릿 등의 방식을 사용하여 코드형 인프라를 정의합니다. 템플릿을 사용하면 애플리케이션 코드 및 구성과 함께 인프라를 원본 제어에 포함할 수 있습니다. 이렇게 하면 소프트웨어를 개발하는 데 사용하는 것과 동일한 사례를 인프라에 적용할 수 있으므로 검토를 빠르게 반복할 수 있습니다.
- **배포 파이프라인:** 소스 코드 리포지토리, 빌드 시스템, 배포, 테스트 자동화 등의 CI/CD(지속적 통합/지속적 전달) 파이프라인을 사용하여 인프라를 배포합니다. 이렇게 하면 반복 가능하며 일관성 있는 저렴한 방식으로 배포를 반복해서 진행할 수 있습니다.
- **잘 정의된 지표:** 핵심 성과 지표(KPI) 측정을 위한 지표와 모니터링을 설정합니다. 기술 및 비즈니스 지표를 모두 사용하는 것이 좋습니다. 웹 사이트 또는 모바일 앱의 경우 주요 지표는 첫 번째 바이트가 수신되거나 첫 번째 렌더링이 완료될 때까지의 시간을 측정합니다. 그 외에 일반적으로 적용되는 지표에는 스레드 수, 가비지 수집 속도, 대기 상태 등이 있습니다. 요청당 누적 비용 집계액 등의 비즈니스 지표에서는 비용 절감 방법을 파악할 수 있습니다. 지표를 해석할 방법을 신중하게 고려합니다. 예를 들어 평균이 아닌 최대값이나 99 번째 백분위수를 선택할 수 있습니다.
- **자동 성능 테스트:** 배포 프로세스의 일환으로 더 빠르게 실행되는 테스트가 정상적으로 완료되고 나면 성능 테스트를 자동으로 트리거합니다. 이 자동 테스트에서는 새 환경을 설정하고, 테스트 데이터 등의 초기 조건을 설정한 다음 일련의 벤치마크 및 로드 테스트를 실행해야 합니다. 시간별 성능 변화를 추적할 수 있도록 이러한 테스트의 결과를 빌드에 다시 연결해야 합니다. 오래 실행되는 테스트의 경우에는 이 자동 테스트를 빌드의 나머지 부분과 동기화되지 않도록 파이프라인의 일부분으로 포함할 수 있습니다. Amazon EC2 스팟 인스턴스를 사용하여 야간에 성능 테스트를 실행할 수도 있습니다.

- **로드 생성:** 통합/사전 녹화 방식의 사용자 여정을 복제하는 일련의 테스트 스크립트를 생성해야 합니다. 이러한 스크립트는 항상 동일한 결과를 반환하고 결합되지 않아야 합니다. 유효한 결과를 얻기 위해 "사전 준비" 스크립트를 포함해야 할 수도 있습니다. 따라서 스크립트를 최대한 많이 테스트하여 프로덕션 환경의 사용 동작을 복제하는 것이 좋습니다. 소프트웨어 또는 SaaS(Software-as-a-Service) 솔루션을 사용하여 로드를 생성할 수 있습니다. 경제적으로 로드를 생성할 수 있는 AWS Marketplace 솔루션 및 스팟 인스턴스를 사용하는 것을 고려하십시오.
- **성능 확인:** 팀이 주요 지표(특히 각 빌드 버전 관련 지표)를 확인할 수 있도록 제공해야 합니다. 이렇게 하면 시간 경과에 따른 긍정적이거나 부정적인 주요 추세를 확인할 수 있습니다. 또한 오류나 예외 수 관련 지표도 표시하여 작동 중인 시스템을 테스트하고 있는지를 확인해야 합니다.
- **시각화:** 성능 문제, 핫스팟, 대기 상태, 낮은 사용률 등이 확인되는 위치를 명확하게 표시하는 시각화 기술을 사용합니다. 아키텍처 다이어그램 위에 성능 지표를 겹쳐서 표시합니다. 그래프나 코드를 호출하면 문제를 빠르게 확인할 수 있습니다.

이 성능 검토 프로세스는 기존 배포 파이프라인의 단순 확장으로 구현한 다음 테스트 요구 사항이 복잡해짐에 따라 장기적으로 개선할 수 있습니다. 그러면 이후 아키텍처에서는 일반화된 방식을 적용하고 동일한 프로세스와 아티팩트를 재사용할 수 있습니다.

아키텍처의 성능 저하는 성능 검토 프로세스가 없거나 효과적이지 않은 경우 주로 발생합니다. 아키텍처의 성능이 불량한 경우 이 성능 검토 프로세스를 시행하면 Deming 의 PDCA(계획-작업-검사-조치) 주기를 적용해 반복 과정을 개선할 수 있습니다.

## 새 릴리스를 활용하도록 워크로드를 점진적으로 변경

고객의 요구 사항에 따른 AWS 의 지속적인 혁신을 활용하십시오. AWS 는 정기적으로 새로운 리전, 엣지 로케이션, 서비스 및 기능을 출시합니다. 이러한 릴리스를 적용하면 아키텍처의 성능 효율성을 개선할 수 있습니다.

**새로운 리소스 및 서비스에 대한 최신 정보 숙지:** 새로운 서비스, 설계 패턴 및 제품 오퍼링이 제공되면 이를 사용하여 성능을 개선할 방법을 평가하십시오. 임시 평가, 내부 논의 또는 외부 분석을 통해 워크로드의 효율성이나 성능을 개선할 수 있는 방법을 결정하십시오.

AWS 의 업데이트, 새 기능과 서비스를 평가하기 위한 프로세스를 정의합니다. 새 기술을 사용하는 개념 증명 작성, 내부 그룹과의 상담 등을 예로 들 수 있습니다. 새 아이디어나 서비스를 사용해 볼 때는 성능 테스트를 실행하여 해당 아이디어/서비스가 워크로드의 효율성이나 성능에 주는 영향을 측정합니다. AWS 에서 제공되는 유연한 기능을 활용하면 비용이나 위험을 최소화하면서 새 아이디어나 기술을 자주 테스트할 수 있습니다.

**워크로드 성능 개선을 위한 프로세스 정의:** 새 서비스, 설계 패턴, 리소스 유형 및 구성을 사용할 수 있게 되면 평가를 하기 위한 프로세스를 정의해야 합니다. 예를 들어 새로운 인스턴스 오퍼링에서 기존 성능 테스트를 실행하여 워크로드 개선 가능성을 결정합니다.

워크로드의 성능에는 몇 가지 주요 제약 사항이 있습니다. 워크로드의 성능을 향상시킬 수 있는 혁신이 어떤 것인지 파악할 수 있도록 이러한 내용을 문서화합니다. 새 서비스나 기술을 사용할 수 있게 되면 해당 서비스/기술을 습득할 때 이 정보를 사용하여 제약 조건이나 병목 현상을 완화하는 방법을 파악합니다.

**장기적인 워크로드 성능 개선:** 조직에서 평가 프로세스를 통해 수집된 정보를 사용하여 제공되는 새 서비스나 리소스를 적극적으로 도입합니다.

새 서비스나 기술을 평가할 때 수집한 정보를 사용하여 변경을 진행합니다. 업무 내용이나 워크로드가 변경되면 성능도 변경되어야 합니다. 워크로드 지표에서 수집한 데이터를 사용해 효율성이나 성능을 가장 많이 개선할 수 있는 영역을 평가하고, 수요를 충족할 수 있도록 새 서비스와 기술을 사전에 도입합니다.

## 리소스

벤치마킹과 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

### 동영상

- Amazon Web Services YouTube 채널
- AWS Online Tech Talks YouTube 채널
- AWS Events YouTube 채널

# 모니터링

아키텍처를 구현한 후에는 고객이 영향을 받기 전에 모든 문제를 해결할 수 있도록 성능을 모니터링해야 합니다. 임계값을 초과할 경우 경보가 생성되도록 모니터링 지표를 사용해야 합니다.

AWS 에서 수행하는 모니터링에서는 다음의 5 가지 고유 단계를 진행합니다. 이러한 단계와 관련된 자세한 설명은 안정성 원칙 백서에 나와 있습니다.

1. **생성** – 모니터링 범위, 지표 및 임계값
2. **집계** – 여러 원본에서 모든 모니터링 정보를 확인할 수 있는 보기 생성
3. **실시간 처리 및 경보 생성** – 인지 및 대응
4. **저장** – 데이터 관리 및 보존 정책
5. **분석** – 대시보드, 보고 및 분석 정보 파악

CloudWatch 는 AWS 클라우드 리소스 및 AWS 에서 실행되는 워크로드용 모니터링 서비스입니다. CloudWatch 를 사용하면 지표를 수집/추적하고 로그 파일을 수집/모니터링하고 경보를 설정할 수 있습니다. CloudWatch 는 EC2 인스턴스/RDS DB 인스턴스 등의 AWS 리소스는 물론 워크로드와 서비스에서 생성된 사용자 지정 지표와 애플리케이션에서 생성한 모든 로그 파일을 모니터링할 수 있습니다. CloudWatch 를 사용하면 리소스 사용률, 애플리케이션 성능 및 운영 상태에 대한 시스템 차원의 가시성을 확보할 수 있습니다. 이러한 분석 정보를 활용해 문제에 빠르게 대응하고 워크로드가 원활하게 실행되는 상태를 유지할 수 있습니다.

CloudWatch 대시보드를 사용하면 AWS 리소스 및 사용자 지정 지표의 재사용 가능 그래프를 생성할 수 있으므로 운영 상태를 모니터링하고 문제를 한눈에 파악할 수 있습니다.

효율적인 모니터링 솔루션에서는 오탐이 발생하지 않아야 합니다. 자동 트리거는 수동 작업으로 인한 오류를 방지하고 문제를 해결하는 데 걸리는 시간을 줄일 수 있습니다. 프로덕션 환경에서 시뮬레이션이 진행되는 게임 데이를 계획하여 경보 솔루션을 테스트하고 해당 솔루션이 문제를 정확하게 인지하는지를 확인합니다.

모니터링 솔루션에는 AM(활성 모니터링)과 PM(수동 모니터링)의 두 가지 유형이 있습니다. AM 과 PM 은 상호 보완 관계이므로 워크로드 성능과 관련된 모든 정보를 파악할 수 있습니다.

**활성 모니터링**에서는 제품의 주요 경로를 따라 진행되는 스크립트로 작성된 사용자 여정의 사용자 활동을 시뮬레이션합니다. 워크로드의 성능과 가용성을 테스트하려면 AM 을 지속적으로

수행해야 합니다. 지속적으로 수행되는 간편하며 예측이 가능한 AM 은 PM 을 보완합니다. 모든 환경(특히 프로덕션 전 환경)에서 AM 을 실행하여 문제나 성능 관련 문제점이 최종 사용자에게 영향을 주기 전에 파악할 수 있습니다.

**수동 모니터링**은 대개 웹 기반 워크로드에 사용됩니다. PM 은 브라우저에서 성능 지표를 수집합니다. 웹을 기반으로 하지 않는 워크로드 역시 유사한 방식을 사용할 수 있습니다. 모든 사용자나 일부 사용자, 지역, 브라우저 및 디바이스 유형에서 지표를 수집할 수 있습니다. PM 을 사용하여 다음 문제를 확인하십시오.

- **사용자 환경 성능**: PM 에서는 사용자에게 발생하는 문제 관련 지표를 제공합니다. 이 지표를 통해 프로덕션 환경의 작동 방식을 지속적으로 파악할 수 있을 뿐 아니라 시간에 따른 변경의 영향도 확인할 수 있습니다.
- **지리적 성능 가변성**: 워크로드가 전 세계적으로 사용되며 사용자들이 전 세계에서 워크로드에 액세스하는 경우 PM 을 사용하면 특정 지역의 사용자에게 영향을 미치는 성능 문제를 포착할 수 있습니다.
- **API 사용의 영향**: 최신 워크로드는 내부 API 와 타사 API 를 사용합니다. PM 을 수행하면 API 사용 방식을 확인할 수 있으므로 내부 API 및 타사 API 공급자에서 발생하는 성능 병목 현상을 파악할 수 있습니다.

CloudWatch 는 모니터링 기능 및 알림 경보 전송 기능을 제공합니다. 자동화를 이용하면 Amazon Kinesis, Amazon Simple Queue Service(Amazon SQS) 및 AWS Lambda 를 통해 작업을 트리거하여 성능 문제를 해결할 수 있습니다.

## 리소스를 모니터링하여 예상 성능을 제공하는지 확인

시스템 성능은 시간이 지남에 따라 저하될 수 있습니다. 시스템 성능을 모니터링하여 성능 저하 상태를 식별하고 운영 체제 또는 애플리케이션 부하와 같은 내부 또는 외부 요인을 해소합니다.

**성능 관련 지표 기록:** 모니터링 및 관찰 서비스를 사용하여 성능 관련 지표를 기록합니다. 예를 들어 데이터베이스 트랜잭션, 속도가 느린 쿼리, I/O 지연 시간, HTTP 요청 처리량, 서비스 지연 시간 또는 기타 주요 데이터를 기록할 수 있습니다.

워크로드에 중요한 성능 지표를 확인하여 기록합니다. 워크로드의 전반적인 성능이나 효율성에 영향을 미치는 구성 요소를 파악하려면 이 데이터가 필요합니다.

고객 경험을 바탕으로 중요한 지표를 식별하십시오. 각 지표에 대해 목표, 측정 방식 및 우선 순위를 정합니다. 이러한 지표를 사용하여 성능 관련 문제를 사전에 해결할 수 있도록 경보와 알림을 작성합니다.

**이벤트 또는 인시던트 발생 시의 지표 분석**: 이벤트나 인시던트에 대응하는 과정에서 모니터링 대시보드나 보고서를 사용해 이벤트/인시던트의 영향을 파악하고 진단합니다. 이러한 대시보드나 보고서에서는 예상 성능을 제공하지 못하는 워크로드 부분을 파악할 수 있습니다.

아키텍처에 대한 중요한 사용자 사례를 작성할 때 사례별로 얼마나 긴급히 처리되어야 하는가 등의 성능 요구 사항을 포함합니다. 이러한 중요 사례의 경우 스크립트로 작성된 사용자 여정을 구현하여 이러한 사례의 성능이 요구 사항에 부합하는지 확인합니다.

**워크로드 성능을 측정하는 KPI(핵심 성과 지표) 설정:** 워크로드 성능이 의도한 성능과 같은지 여부를 나타내는 KPI 를 식별합니다. 예를 들어 API 기반 워크로드는 전반적인 성능의 지표로 전체 응답 지연 시간을 사용할 수 있으며, 전자상거래 사이트는 구매 건 수를 KPI 로 사용하도록 선택할 수 있습니다.

고객이 요구하는 성능 경험 및 고객이 워크로드의 성능을 판단하는 근거를 문서화합니다. 이러한 요구 사항을 토대로 하여 시스템의 전반적인 성능을 표시하는 KPI(핵심 성과 지표)를 설정합니다.

**모니터링을 사용하여 경보 기반 알림 생성:** 정의한 성능 관련 KPI(핵심 성과 지표)를 사용하고, 이러한 측정치가 예상 범위를 벗어날 때 경보를 자동으로 생성하는 모니터링 시스템을 사용합니다.

Amazon CloudWatch 는 아키텍처의 리소스 전반에서 지표를 수집할 수 있습니다. 또한 사용자 지정 지표를 수집하고 게시하여 비즈니스 또는 파생 지표를 파악할 수도 있습니다. CloudWatch 또는 타사 모니터링 서비스를 사용하여 임계값이 초과되었음을 나타내는 경보를 설정합니다. 이 경보는 지표가 필요한 경계를 벗어났음을 나타냅니다.

**일정한 간격으로 지표 검토:** 주기적인 유지 관리의 일환으로 또는 이벤트나 인시던트 대응 과정에서 수집된 지표를 검토합니다. 이러한 검토를 진행하면 문제를 해결하는 데 반드시 필요했던 지표와 문제를 확인/해결/방지하는 데 도움이 되었던 지표(추적한 경우)를 추가로 파악할 수 있습니다.

인시던트나 이벤트 대응의 일환으로 문제를 해결하는 데 도움이 되었던 지표와, 현재는 추적 중이지 않지만 도움이 되었을 수 있는 지표를 평가합니다. 이 평가 결과를 토대로 하여 수집한 지표의 품질을 개선하면 이후 인시던트를 예방하거나 더 빨리 해결할 수 있습니다.

**사전 모니터링 및 경보 생성:** KPI(핵심 성과 지표)를 모니터링 및 경보 시스템과 함께 사용하여 성능 관련 문제를 선제적으로 해결합니다. 경보를 사용하여 가능한 경우 문제를 해결하는 자동화 작업을 트리거합니다. 자동 대응이 불가능한 경우 대응을 수행할 수 있는 담당자에게 경보를 에스컬레이션합니다. 예를 들어 필요한 KPI(핵심 성과 지표) 값을 예측하고 해당 값이 특정 임계값을 초과하는 경우 경보를 생성할 수 있는 시스템이나, KPI 가 필요한 값의 범위를 벗어나는 경우 배포를 자동으로 중지하거나 롤백할 수 있는 도구로 경보를 에스컬레이션할 수 있습니다.

워크로드가 실행됨에 따라 성능을 시각적으로 확인할 수 있는 프로세스를 구현합니다. 워크로드가 최적의 상태로 작동하고 있는지를 확인할 수 있도록 성능 기대치 관련 기준을 설정하고 모니터링 대시보드를 구축합니다.

## 리소스

성능 효율성 개선을 위한 모니터링한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- Cut through the chaos: Gain operational visibility and insight (MGT301-R1)

**설명서**

- X-Ray 설명서
- CloudWatch 설명서

# 절충

솔루션을 설계할 때는 최적의 방식이 적용되도록 각 방식의 장단점을 고려해야 합니다. 상황에 따라 더 우수한 성능을 제공하기 위해 일관성/내구성/공간 및 시간/지연 시간 중 우선적으로 고려할 요소를 선택할 수 있습니다.

AWS 를 사용하면 몇 분 내에 전 세계의 여러 위치에 리소스를 배포하여 최종 사용자에게 더욱 가깝게 다가갈 수 있습니다. 또한 데이터베이스 시스템과 같은 정보 스토어에 읽기 전용 복제본을 동적으로 추가하여 기본 데이터베이스의 부하를 줄일 수 있습니다.

AWS 는 인메모리 스토어 또는 캐시를 제공하는 Amazon ElastiCache 와 같은 캐싱 솔루션과 정적 콘텐츠의 사본을 최종 사용자에게 더 가깝게 캐싱하는 Amazon CloudFront 를 제공합니다. Amazon DynamoDB Accelerator(DAX)는 Amazon DynamoDB 앞에 연속 읽기/연속 쓰기 분산 캐싱 계층을 제공하여 동일한 API 를 지원하면서도 캐시 내에 있는 엔터티에 대해 1 밀리초 미만의 지연 시간을 제공합니다.

## 절충을 통한 성능 개선

솔루션을 설계할 때 절충을 능동적으로 고려하면 최적의 접근 방식을 선택할 수 있습니다. 종종 일관성, 내구성, 공간을 희생하여 시간과 지연을 개선하면 성능이 개선될 수 있습니다. 절충 과정에서 아키텍처의 복잡성이 증가할 수 있으므로 측정 가능한 이점의 달성 여부를 확인하는 로드 테스트를 수행해야 합니다.

**성능이 가장 중요한 영역 파악:** 워크로드 성능을 개선하여 효율성을높이고 고객 환경을 개선할 수 있는 영역을 파악합니다. 예를 들어 많은 양의 고객 상호 작용이 수행되는 웹 사이트에서는 엣지 서비스를 사용하여 콘텐츠 전송 위치를 고객과 더 가까운 곳으로 이동하면 성능을 개선할 수 있습니다.

**설계 패턴 및 서비스 파악:** 워크로드 성능 개선에 도움이 되는 다양한 설계 패턴과 서비스를 조사하고 파악합니다. 분석을 수행하는 동안 성능 개선을 위해 절충할 수 있는 요소를 파악합니다. 예를 들어 캐시 서비스를 사용하면 데이터베이스 시스템의 로드를 줄일 수 있습니다. 하지만 안전한 캐싱을 구현하기 위한 엔지니어링을 수행해야 하거나, 특정 영역에서 최종 일관성 개념을 도입해야 할 수 있습니다.

사용 가능한 성능 구성 옵션 및 이러한 옵션이 워크로드에 영향을 미치는 방식을 확인합니다. 이러한 옵션이 아키텍처와 상호 작용하는 방식, 그리고 측정된 성능과 사용자의 체감 성능에 주는 영향을 파악해야 워크로드 성능을 최적화할 수 있습니다.

Amazon Builders' Library 는 Amazon 의 기술 구축 및 운영 방식에 관한 상세한 설명을 제공합니다. 무료로 제공되는 이러한 문서는 Amazon 의 선임 엔지니어가 작성하며 아키텍처, 소프트웨어 전송 및 운영 전반에 걸친 주제를 다룹니다. 예를 들어 Amazon 이 소프트웨어 전송을 자동화하여 연간 1 억 5 천만 개 이상의 배포를 달성하는 방법 또는 Amazon 엔지니어가 셔플 샤딩과 같은 원칙을 구현하여 고가용성과 내결함성을 갖춘 복원력이 뛰어난 시스템을 구축하는 방법을 확인할 수 있습니다.

**절충이 고객과 효율성에 미치는 영향 식별:** 성능 관련 개선 사항을 평가할 때는 고객 및 워크로드 효율성에 영향을 미치는 옵션을 결정합니다. 예를 들어 키–값 데이터 스토어를 사용하여 시스템 성능이 개선되는 경우, 이 옵션의 지속적 특성이 결과적으로 고객에 미치는 영향을 평가하는 것이 중요합니다.

지표와 모니터링을 통해 시스템에서 성능 수준이 낮은 영역을 파악합니다. 성능을 개선할 수 있는 방법과 해당 개선 과정에서 절충해야 하는 요소와 성능 개선 작업이 시스템과 사용자 환경에 미치는 영향을 확인합니다. 예를 들어 데이터 캐싱 구현 시에는 성능을 크게 개선할 수 있지만, 캐시된 데이터를 업데이트하거나 무효화할 방법 및 시기와 관련된 명확한 전략을 마련해야 잘못된 시스템 동작을 방지할 수 있습니다.

**성능 개선의 영향 측정:** 성능 개선을 위해 변경을 수행하는 동안 수집된 지표와 데이터를 평가합니다. 이 정보를 사용하여 성능 개선이 워크로드, 워크로드의 구성 요소 및 고객에게 미치는 영향을 확인합니다. 이 측정을 수행하면 절충을 통한 성능 개선을 파악할 수 있으며 부정적인 부작용 발생 여부를 확인할 수 있습니다.

Well-Architected 시스템은 다양한 성능 관련 전략을 조합하여 활용합니다. 따라서 지정된 핫스팟이나 병목 현상을 가장 많이 개선할 수 있는 전략을 결정해야 합니다. 예를 들어 여러 관계형 데이터베이스 시스템에서 데이터를 샤딩하면 전반적인 처리량이 높아지는 동시에 트랜잭션이 계속 지원됩니다. 각 샤드 내에서 캐싱을 수행하면 로드를 줄일 수 있습니다.

**다양한 성능 관련 전략 사용:** 해당하는 경우 여러 전략을 활용하여 성능을 개선합니다. 예를 들어 데이터 캐싱 등의 전략을 사용해 과도한 네트워크 또는 데이터베이스 호출을 방지하고, 데이터베이스 엔진용 읽기 전용 복제본을 사용해 읽기 속도를 높이고, 가능한 경우 데이터

샤딩/압축을 수행하여 데이터 볼륨을 줄이고, 제공되는 결과를 버퍼링/스트리밍하여 차단을 방지하는 등의 전략을 사용할 수 있습니다.

워크로드를 변경할 때는 지표를 수집 및 평가하여 변경의 영향을 확인합니다. 시스템 및 최종 사용자에 대한 영향을 모두 측정하여 절충 작업이 워크로드에 미치는 영향을 파악합니다. 로드 테스트 등의 체계적인 방식을 사용해 개별 요소 절충 시, 성능을 개선할 수 있는지 여부를 파악합니다.

### 리소스

캐싱과 관련한 AWS 모범 사례에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

**동영상**

- Introducing The Amazon Builders' Library (DOP328)

**설명서**

- Amazon Builders' Library
- Amazon ElastiCache 구현 모범 사례

## 결론

성능 효율성을 높이고 유지하려면 데이터 기반 방식을 사용해야 합니다. 이 과정에서는 성능을 더욱 높이기 위해 환경을 최적화할 수 있는 액세스 패턴과 절충 요소를 적극적으로 고려해야 합니다. 벤치마크 및 로드 테스트를 기반으로 하는 검토 프로세스를 사용하면 적절한 리소스 유형과 구성을 선택할 수 있습니다. 인프라를 코드로 처리하면 데이터를 사용해 아키텍처와 관련하여 사실에 입각한 결정을 내리면서 아키텍처를 빠르고 안전하게 개선할 수 있습니다. 활성 모니터링과 수동 모니터링을 적절히 조합하여 활용하면 시간이 지나도 아키텍처 성능이 저하되지 않습니다.

AWS 는 비즈니스 가치를 제공하는 동시에 성능 효율성을 높여 주는 아키텍처를 구축할 수 있도록 지원합니다. 이 백서에 설명된 도구와 기술을 사용하여 성공을 보장하십시오.

# 기고자

다음은 본 문서 작성에 도움을 준 개인 및 조직입니다.

- Eric Pullen, Well-Architected 성능 효율성 부문 책임자, Amazon Web Services
- Philip Fitzsimons, Well-Architected 선임 관리자(Amazon Web Services)
- Julien Lépine, 전문 SA 관리자(Amazon Web Services)
- Ronnen Slasky, 솔루션즈 아키텍트(Amazon Web Services)

# 추가 자료

자세한 내용은 다음 출처를 참조하십시오.

- AWS Well-Architected 프레임워크

# 문서 개정

| 날짜 | 설명 |
|---|---|
| **2020 년 4 월** | v2 에 대한 주 업데이트 |
| **2018 년 7 월** | 문법 문제에 대한 부 업데이트 |
| **2017 년 11 월** | AWS 의 변경 사항을 반영하기 위해 백서를 새로 고침 |
| **2016 년 11 월** | 최초 게시 |