AWS Well-Architected 프레임워크

성능 효율성 원칙

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

성능 효율성 원칙: AWS Well-Architected 프레임워크

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon의 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께, Amazon 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon 계열사, 관련 업체 또는 Amazon의 지원 업체 여부에 상관없이 해당 소유자의 자산입니다.

Table of Contents

요약 및 소개 ... 1
소개 ... 1
성능 효율성 ... 3
설계 원칙 ... 3
정의 ... 3
아키텍처 선택 ... 5
PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해 ... 5
구현 지침 ... 6
리소스 ... 7
PERF01-BP02 클라우드 제공업체 또는 적절한 파트너의 지침을 사용하여 아키텍처 패턴 및 모범 사례에 대해 알아보기 ... 8
구현 가이드 ... 6
리소스 ... 7
PERF01-BP03 아키텍처 결정에 비용 고려 ... 10
구현 지침 ... 6
리소스 ... 7
PERF01-BP04 장단점이 고객과 아키텍처 효율성에 미치는 영향 평가 ... 12
구현 지침 ... 6
리소스 ... 7
PERF01-BP05 정책 및 참조 아키텍처 사용 ... 13
구현 지침 ... 6
리소스 ... 7
PERF01-BP06 벤치마킹을 사용하여 아키텍처 결정 ... 15
구현 지침 ... 6
리소스 ... 7
PERF01-BP07 아키텍처 선택에 데이터 기반 접근 방식 사용 ... 17
구현 지침 ... 6
리소스 ... 7
컴퓨팅 및 하드웨어 ... 20
PERF02-BP01 워크로드에 가장 적합한 컴퓨팅 옵션 선택 ... 20
구현 지침 ... 6
구현 단계 ... 6
리소스 ... 7
PERF02-BP02 사용 가능한 컴퓨팅 구성 및 기능 파악 ... 23

구현 지침 6
구현 단계 6
리소스 7
PERF02-BP03 컴퓨팅 관련 지표 수집 27
구현 지침 6
구현 단계 6
리소스 7
PERF02-BP04 컴퓨팅 리소스 구성 및 적정 크기 조정 29
구현 지침 6
리소스 7
PERF02-BP05 컴퓨팅 리소스 동적 확장 31
구현 지침 6
리소스 7
PERF02-BP06 최적화된 하드웨어 기반 컴퓨팅 액셀러레이터 사용 34
구현 지침 6
리소스 7
데이터 관리 37
PERF03-BP01 데이터 액세스 및 스토리지 요구 사항을 가장 잘 지원하는 목적별 데이터 스토어 사용 37
구현 지침 6
리소스 7
PERF03-BP02 데이터 스토어에 사용 가능한 구성 옵션 평가 47
구현 지침 6
리소스 7
PERF03-BP03 데이터 스토어 성과 지표 수집 및 기록 51
구현 지침 6
구현 단계 6
리소스 7
PERF03-BP04 데이터 스토어에서 쿼리 성능을 개선하기 위한 전략 구현 54
구현 지침 6
리소스 7
PERF03-BP05 캐싱을 활용하는 데이터 액세스 패턴 구현 56
구현 지침 6
리소스 7
네트워킹 및 콘텐츠 전송 60
PERF04-BP01 네트워킹이 성능에 미치는 영향 파악 60

구현 지침 6
리소스 7
PERF04-BP02 사용 가능한 네트워킹 기능 평가 63
구현 지침 6
리소스 7
PERF04-BP03 워크로드에 적합한 전용 연결 또는 VPN 선택 68
구현 지침 6
리소스 7
PERF04-BP04 로드 밸런싱을 사용하여 여러 리소스에 트래픽 분산 71
구현 지침 6
리소스 7
PERF04-BP05 성능을 개선할 수 있는 네트워크 프로토콜 선택 75
구현 지침 6
리소스 7
PERF04-BP06 네트워크 요구 사항에 따라 워크로드의 위치 선택 78
구현 지침 6
리소스 7
PERF04-BP07 지표를 기준으로 네트워크 구성 최적화 82
구현 지침 6
리소스 7
프로세스 및 문화 87
PERF05-BP01 워크로드 상태 및 성능을 측정하기 위한 핵심 성과 지표(KPI) 수립 88
구현 지침 6
구현 단계 6
리소스 7
PERF05-BP02 모니터링 솔루션을 사용하여 성능이 가장 중요한 영역 파악 91
구현 지침 6
리소스 7
PERF05-BP03 워크로드 성능 개선을 위한 프로세스 정의 93
구현 지침 6
리소스 7
PERF05-BP04 워크로드 로드 테스트 95
구현 지침 6
리소스 7
PERF05-BP05 자동화를 사용하여 성능 관련 문제 사전 해결 97
구현 지침 6

리소스 7
PERF05-BP06 워크로드 및 서비스 최신 상태 유지 99
구현 지침 6
구현 단계 6
리소스 7
PERF05-BP07 정기적으로 지표 검토 101
구현 지침 6
리소스 7
결론 104
기여자 105
참조 자료 106
문서 수정 107
고지 사항 109
AWS 용어집 110

성능 효율성 원칙 - AWS Well-Architected Framework

발행일: 2024년 11월 6일(문서 수정)

이 백서에서는 AWS Well-Architected Framework의 성능 효율성 원칙을 중점적으로 다룹니다. 또한 AWS 환경 설계, 제공 및 유지 관리에 모범 사례를 적용할 때 참조할 수 있는 지침을 제공합니다.

소개

AWS Well-Architected Framework를 활용하면 AWS에서 워크로드를 구축하면서 내리게 되는 결정의 장단점을 이해할 수 있습니다. 이 프레임워크를 사용하면 클라우드에서 신뢰할 수 있고 안전하며 효율적이고 경제적이면서 지속 가능한 워크로드를 설계하고 운영하기 위한 설계 모범 사례를 알아볼 수 있습니다. 이 프레임워크는 모범 사례를 기준으로 아키텍처를 일관적으로 측정하고 개선할 영역을 파악하는 방법을 제시합니다. 워크로드를 제대로 설계하면 비즈니스 성공 가능성이 커집니다.

이 프레임워크는 다음 6가지 원칙을 기반으로 합니다.

- 운영 우수성
- 보안
- 신뢰성
- 성능 효율성
- 비용 최적화
- 지속 가능성

이 백서는 워크로드에 성능 효율성 원칙을 적용하는 데 중점을 둡니다. 기존 온프레미스 환경에서는 우수한 성능을 유지하기가 어렵습니다. 본 백서의 원칙을 사용하면 AWS에서 시간이 지남에 따라 지속적인 성능을 효율적으로 제공하는 아키텍처를 구축하는 데 도움이 됩니다. 이 문서의 지침과 모범 사례는 AWS에 성능 효율적인 클라우드 솔루션을 구축하기 위한 지침 원칙 역할을 하는 5가지 주요 중점 영역에 걸쳐 있습니다. 이러한 중점 영역은 다음과 같습니다.

- 아키텍처 선택
- 컴퓨팅 및 하드웨어
- 데이터 관리
- 네트워킹 및 콘텐츠 전송
- 프로세스 및 문화

이 문서는 최고 기술 책임자(CTO), 아키텍트, 개발자와 같은 기술 업무 담당자와 운영 팀원을 대상으로 작성되었습니다. 이 백서의 내용을 확인하고 나면 성능이 뛰어난 클라우드 아키텍처를 설계할 때 사용할 AWS 모범 사례와 전략을 파악할 수 있습니다.

성능 효율성

성능 효율성 원칙에는 클라우드 리소스를 성능 요구 사항에 맞게 효율적으로 사용하고, 수요 변화 및 기술 진화에 발맞춰 그러한 효율성을 유지하는 능력이 포함됩니다.

주제

- 설계 원칙
- 정의

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 고려하세요. 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 프로비저닝과 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.
- 기계에 대한 공감 고려: 목표에 가장 일치하는 기술 접근 방식을 사용합니다. 예를 들어 워크로드에 대한 데이터베이스 또는 스토리지를 선택할 때는 데이터 접근 패턴을 고려합니다.

정의

클라우드에서 성능 효율성을 달성하려면 다음 영역에 집중합니다.

- 아키텍처 선택
- 컴퓨팅 및 하드웨어
- 데이터 관리
- 네트워킹 및 콘텐츠 전송
- 프로세스 및 문화

고성능 아키텍처를 구축할 때 데이터 기반 접근 방식을 취합니다. 개괄적 설계부터 리소스 유형 선택과 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 확인할 수 있습니다. 압축 또는 캐싱을 사용하거나 일관성 요구 사항을 완화하는 등의 성능 개선을 위해 아키텍처에서 절충안을 구성합니다.

아키텍처 선택

특정 워크로드에 대한 최적의 솔루션은 다양하며, 종종 여러 접근 방식이 결합된 솔루션을 사용합니다. 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 DynamoDB와 같은 관리형 서비스는 완전관리형 NoSQL 데이터베이스로, 어떤 규모에서도 지연 시간이 한 자릿수 밀리초 단위로 매우 짧습니다.

이 중점 영역에서는 효율적인 고성능 클라우드 리소스 및 아키텍처 패턴을 선택하는 방법에 대한 지침과 모범 사례를 공유합니다.

모범 사례

- PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해
- PERF01-BP02 클라우드 제공업체 또는 적절한 파트너의 지침을 사용하여 아키텍처 패턴 및 모범 사례에 대해 알아보기
- PERF01-BP03 아키텍처 결정에 비용 고려
- PERF01-BP04 장단점이 고객과 아키텍처 효율성에 미치는 영향 평가
- PERF01-BP05 정책 및 참조 아키텍처 사용
- PERF01-BP06 벤치마킹을 사용하여 아키텍처 결정
- PERF01-BP07 아키텍처 선택에 데이터 기반 접근 방식 사용

PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해

워크로드 아키텍처에서 더 나은 아키텍처 결정을 내리고 성능 효율성을 높일 수 있도록 사용 가능한 서비스 및 구성에 대해 지속적으로 학습하고 살펴보세요.

일반적인 안티 패턴:

- 클라우드를 코로케이션된 데이터 센터로 사용합니다.
- 클라우드로 마이그레이션한 후에는 애플리케이션을 현대화하지 않습니다.
- 지속해야 하는 모든 항목에 하나의 스토리지 유형만 사용합니다.
- 현재 표준에 가장 근접한 인스턴스 유형을 사용하지만 필요한 경우 더 큰 인스턴스 유형을 사용합니다.

- 관리형 서비스로 사용할 수 있는 기술을 배포하고 관리합니다.

이 모범 사례 확립의 이점: 새로운 서비스 및 구성을 고려하여 성능을 현저히 개선하고 비용을 절감하며 워크로드를 유지 관리하는 데 필요한 노력을 최적화할 수 있습니다. 또한 클라우드 지원 제품의 가치 실현 시간을 단축하는 데 도움이 될 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

AWS는 성능을 개선하고 클라우드 워크로드 비용을 절감할 수 있는 새로운 서비스와 기능을 지속적으로 출시하고 있습니다. 클라우드에서 성능 효율성을 유지하려면 이러한 새로운 서비스와 기능을 최신 상태로 유지해야 합니다. 또한 워크로드 아키텍처를 현대화하면 생산성을 가속화하고 혁신을 촉진하며 더 많은 성장 기회를 확보하는 데 도움이 됩니다.

구현 단계

- 관련 서비스에 대한 워크로드 소프트웨어 및 아키텍처 인벤토리를 생성합니다. 자세히 알아볼 제품 범주를 결정합니다.
- AWS 제품 및 서비스를 탐색하여 성능을 개선하고 비용 및 운영 복잡성을 줄이는 데 도움이 되는 관련 서비스 및 구성 옵션을 식별하고 알아봅니다.
  - Amazon Web Services Cloud
  - AWS Academy
  - AWS의 새로운 소식
  - AWS 블로그
  - AWS Skill Builder
  - AWS 이벤트 및 웨비나
  - AWS 교육 및 자격증
  - AWS YouTube 채널
  - AWS 워크숍
  - AWS Communities
- Amazon Q를 사용하여 서비스에 대한 관련 정보와 조언을 얻을 수 있습니다.
- 샌드박스(비프로덕션) 환경을 사용하여 추가 비용 없이 새로운 서비스를 알아보고 실험할 수 있습니다.
- 새로운 클라우드 서비스 및 기능에 대해 지속적으로 알아보세요.

리소스

관련 문서:

- Overview of Amazon Web Services
- Amazon EC2 features
- AWS 파트너 학습 계획으로 단계별 학습 시작하기
- AWS 교육 및 자격증
- My learning path to become an AWS solutions architect
- AWS 아키텍처 센터
- AWS Partner Network
- AWS Solutions Library
- AWS 지식 센터
- AWS에서 현대적 애플리케이션 구축

관련 비디오:

- AWS re:Invent 2023 - What’s new with Amazon EC2
- AWS re:Invent 2022 - Reduce your operational and infrastructure costs with Amazon ECS
- AWS re:Invent 2023 - Build with the efficiency, agility & innovation of the cloud with AWS
- AWS re:Invent 2022 - Deploy ML models for inference at high performance and low cost
- This is my Architecture

관련 예제:

- AWS 샘플
- AWS SDK Examples

PERF01-BP02 클라우드 제공업체 또는 적절한 파트너의 지침을 사용하여 아키텍처 패턴 및 모범 사례에 대해 알아보기

클라우드 회사 리소스(예: 설명서, 솔루션스 아키텍트, 전문 서비스 또는 적절한 파트너)를 의사 결정의 지침으로 사용하세요. 이러한 리소스는 최적의 성능을 위해 아키텍처를 검토하고 개선하는 데 도움이 됩니다.

일반적인 안티 패턴:

- AWS를 공통된 클라우드 제공업체로 사용합니다.
- AWS 서비스를 설계 의도와 다른 방식으로 사용합니다.
- 비즈니스 상황을 고려하지 않고 모든 지침을 따릅니다.

이 모범 사례 확립의 이점: 클라우드 제공업체 또는 적절한 파트너의 지침을 사용하면 워크로드에 적합한 아키텍처를 선택하고 의사 결정에 확신을 얻을 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

AWS는 효율적인 클라우드 워크로드를 구축하고 관리하는 데 도움이 되는 다양한 지침, 설명서 및 리소스를 제공합니다. AWS 설명서에는 코드 샘플, 자습서, 자세한 서비스 설명이 포함되어 있습니다. AWS는 설명서 외에도 고객이 클라우드 서비스의 다양한 측면을 탐색하고 AWS에서 효율적인 클라우드 아키텍처를 구현하는 데 도움이 되는 교육 및 인증 프로그램, 솔루션 아키텍트, 전문 서비스를 제공합니다.

이러한 리소스를 활용하면 유용한 지식과 모범 사례에 대한 인사이트를 얻고, 시간을 절약하며, AWS 클라우드에서 더 나은 성과를 달성할 수 있습니다.

구현 단계

- AWS 설명서 및 지침을 검토하고 모범 사례를 따릅니다. 이러한 리소스는 서비스를 효과적으로 선택 및 구성하고 더 나은 성능을 달성하는 데 도움이 될 수 있습니다.
  - AWS 설명서(예: 사용 설명서 및 백서)
  - AWS 블로그
  - AWS 교육 및 자격증
  - AWS YouTube 채널

- AWS 파트너 이벤트(예: AWS Global Summits, AWS re:Invent. 사용자 그룹, 워크숍)에 참가하여 AWS 전문가로부터 AWS 서비스 사용 모범 사례에 대해 알아보세요.
  - AWS 파트너 학습 계획으로 단계별 학습 시작하기
  - AWS 이벤트 및 웨비나
  - AWS 워크숍
  - AWS Communities
- 추가 지침이나 제품 정보가 필요한 경우 AWS에 문의하여 도움을 받으세요. AWS 솔루션스 아키텍트 및 AWS Professional Services는 솔루션 구현에 대한 지침을 제공합니다. AWS 파트너는 비즈니스 민첩성 및 혁신을 달성하는 데 도움이 되는 AWS 전문 지식을 제공합니다.
- 지원를 사용하여 서비스를 효과적으로 사용하는 데 필요한 기술 지원을 받으세요. 지원 플랜은 성능을 최적화하고, 위험을 통제하며, 비용을 제어하면서 AWS를 성공적으로 사용할 수 있도록 적절한 도구 조합 및 전문 지식에 대한 액세스를 제공하도록 설계되었습니다.

리소스

관련 문서:

- AWS 아키텍처 센터
- AWS Partner Network
- AWS Solutions Library
- AWS 지식 센터
- AWS Enterprise Support

관련 비디오:

- This is my Architecture
- AWS re:Invent 2023 - Advanced event-driven patterns with Amazon EventBridge
- AWS re:Invent 2023 - Implementing distributed design patterns on AWS
- AWS re:Invent 2023 - Application architecture as code

관련 예제:

- AWS 샘플
- AWS SDK Examples

- AWS Analytics Reference Architecture

PERF01-BP03 아키텍처 결정에 비용 고려

클라우드 워크로드의 리소스 사용률과 성능 효율성을 개선하기 위해 비용 측면을 고려하여 아키텍처를 결정하세요. 클라우드 워크로드의 비용이 미치는 영향을 인식하면 효율적인 리소스를 활용하고 낭비적인 작업을 줄일 가능성이 커집니다.

일반적인 안티 패턴:

- 인스턴스 패밀리는 하나만 사용합니다.
- 라이선스가 부여된 솔루션과 오픈 소스 솔루션을 비교 평가하지 않습니다.
- 스토리지 수명 주기 정책을 정의하지 않습니다.
- AWS 클라우드의 새로운 서비스 및 기능을 검토하지 않습니다.
- 블록 스토리지만 사용합니다.

이 모범 사례 확립의 이점: 의사 결정에 비용을 고려하면 보다 효율적인 리소스를 사용하고 다른 투자를 모색할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

워크로드를 비용에 맞게 최적화하면 리소스 사용률을 높이고 클라우드 워크로드에서 낭비를 방지할 수 있습니다. 아키텍처 결정 시 비용을 고려하면 일반적으로 워크로드 구성 요소의 크기를 규모에 맞게 조정하고 탄력성을 활성화하여, 클라우드 워크로드 성능 효율성의 향상으로 이어집니다.

구현 단계

- 클라우드 워크로드에 대한 예산 한도와 같은 비용 목표를 설정합니다.
- 워크로드 비용을 유발하는 주요 구성 요소(인스턴스 및 스토리지 등)를 파악합니다. AWS Pricing Calculator 및 AWS Cost Explorer를 사용하여 워크로드의 주요 비용 요인을 식별할 수 있습니다.
- 온디맨드, 예약형 인스턴스, ,절감형 플랜, 스팟 인스턴스에서 가격 책정 모델을 이해합니다.
- Well-Architected 비용 최적화 모범 사례를 사용하여 이러한 주요 구성 요소를 비용에 맞게 최적화합니다.
- 지속적으로 비용을 모니터링하고 분석하여 워크로드에서 비용 최적화 기회를 파악합니다.

- AWS Budgets를 사용하여 수용 불가능한 비용에 대해 알림을 받습니다.
- AWS Compute Optimizer 또는 AWS Trusted Advisor를 사용하여 비용 최적화 권장 사항을 받습니다.
- AWS Cost Anomaly Detection을 사용하여 자동화된 비용 이상 탐지 및 근본 원인 분석을 수행할 수 있습니다.

리소스

관련 문서:

- What is AWS Billing and Cost Management?
- Cost Optimization with AWS
- Choosing an AWS cost management strategy
- A Beginner's Guide to AWS Cost Management
- A Detailed Overview of the Cost Intelligence Dashboard
- AWS 아키텍처 센터
- AWS Solutions Library
- AWS 지식 센터

관련 비디오:

- This is my Architecture
- AWS re:Invent 2023 - What's new with AWS cost optimization
- AWS re:Invent 2023 - Optimize cost and performance and track progress toward mitigation
- AWS re:Invent 2023 - AWS storage cost-optimization best practices
- AWS re:Invent 2023 - Optimize costs in your multi-account environments

관련 예제:

- AWS Compute Optimizer Demo code
- Cost Optimization 워크숍
- 클라우드 재무 관리 기술 구현 플레이북
- Startup optimization: Tuning application performance for maximum efficiency

- Serverless Optimization 워크숍(Performance and Cost)
- Scaling cost effective architectures

PERF01-BP04 장단점이 고객과 아키텍처 효율성에 미치는 영향 평가

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, 이러한 변화의 최종적으로 일관된 특성이 고객에게 미치는 영향을 평가하는 것이 중요합니다.

일반적인 안티 패턴:

- 구현에 대한 장단점이 있더라도 모든 성능 이점을 구현해야 한다고 가정합니다.
- 성능 문제가 심각한 지점에 도달했을 때만 워크로드 변경을 평가합니다.

이 모범 사례 확립의 이점: 잠재적인 성능 관련 개선 사항을 평가할 때는 변경 사항에 대한 장단점이 워크로드 요구 사항에서 수용 가능한지 여부를 결정해야 합니다. 경우에 따라 장단점을 보상하기 위해 추가 제어를 구현해야 할 수도 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

성능과 고객 영향 측면에서 아키텍처의 중요한 영역을 파악합니다. 성능을 개선할 수 있는 방법과 해당 개선 과정의 장단점, 개선 사항이 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.

구현 단계

- 워크로드 요구 사항과 SLA를 이해합니다.
- 평가 요소를 명확하게 정의합니다. 평가 요소는 워크로드의 비용, 신뢰성, 보안 및 성능과 관련될 수 있습니다.
- 요구 사항을 충족할 수 있는 아키텍처 및 서비스를 선택합니다.
- 실험 및 개념 증명(POC)을 수행하여 장단점과 고객 및 아키텍처 효율성에 미치는 영향을 평가합니다. 일반적으로 가용성, 성능, 보안성이 높은 워크로드는 더 많은 클라우드 리소스를 소비하는 반면,

더 나은 고객 경험을 제공합니다. 워크로드의 복잡성, 성능, 비용 전반의 장단점을 이해합니다. 일반적으로 세 번째 요소 대신 첫 두 요소가 우선 고려됩니다.

리소스

관련 문서:

- Amazon Builders' Library
- QuickSight KPIs
- Amazon CloudWatch RUM
- X-Ray 설명서
- Understand resiliency patterns and trade-offs to architect efficiently in the cloud

관련 비디오:

- Optimize applications through Amazon CloudWatch RUM
- AWS re:Invent 2023 - Capacity, availability, cost efficiency: Pick three
- AWS re:Invent 2023 - Advanced integration patterns & trade-offs for loosely coupled systems

관련 예제:

- Measure page load time with Amazon CloudWatch Synthetics
- Amazon CloudWatch RUM Web Client

PERF01-BP05 정책 및 참조 아키텍처 사용

워크로드를 설계하고 구현할 때 보다 효율적으로 서비스 및 구성을 선택할 수 있도록 내부 정책과 기존 참조 아키텍처를 사용하세요.

일반적인 안티 패턴:

- 회사의 관리 오버헤드에 영향을 줄 수 있는 다양한 기술을 허용합니다.

이 모범 사례 확립의 이점: 아키텍처, 기술 및 공급업체 선택에 대한 정책을 수립하면 신속하게 의사 결정을 내릴 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

리소스와 아키텍처를 선택할 때 내부 정책을 마련해 두면 아키텍처 선택 시 따라야 할 표준과 지침을 갖출 수 있습니다. 이러한 지침은 올바른 클라우드 서비스를 선택할 때 의사 결정 프로세스를 간소화하고 성능 효율성을 개선하는 데 도움이 될 수 있습니다. 정책 또는 참조 아키텍처를 사용하여 워크로드를 배포합니다. 서비스를 클라우드 배포에 통합한 다음, 성능 테스트를 사용하여 성능 요구 사항을 계속해서 충족할 수 있는지 확인합니다.

구현 단계

- 클라우드 워크로드의 요구 사항을 명확하게 이해합니다.
- 내부 및 외부 정책을 검토하여 가장 관련성이 높은 정책을 파악합니다.
- AWS에서 제공하는 적절한 참조 아키텍처 또는 업계 모범 사례를 사용합니다.
- 정책, 표준, 참조 아키텍처 및 일반적인 상황에 대한 규범적 지침으로 구성된 일련의 지침을 생성합니다. 이렇게 하면 팀이 더 빠르게 움직일 수 있습니다. 해당되는 경우 업종에 맞게 자산을 조정합니다.
- 샌드박스 환경에서 워크로드에 대한 이러한 정책과 참조 아키텍처를 검증합니다.
- 정책 및 참조 아키텍처가 클라우드 워크로드를 최적화하는 데 도움이 되도록 업계 표준 및 AWS 업데이트를 최신 상태로 유지합니다.

리소스

관련 문서:

- AWS 아키텍처 센터
- AWS Partner Network
- AWS Solutions Library
- AWS 지식 센터
- AWS 아키텍처 블로그

관련 비디오:

- This is my Architecture

- AWS re:Invent 2022 - Accelerate value for your business with SAP & AWS reference architecture

관련 예제:

- AWS 샘플
- AWS SDK Examples

PERF01-BP06 벤치마킹을 사용하여 아키텍처 결정

기존 워크로드의 성능을 벤치마킹하여 클라우드에서 어떻게 작동하는지 파악하고 해당 데이터를 기반으로 아키텍처 결정을 내릴 수 있습니다.

일반적인 안티 패턴:

- 워크로드의 특성을 나타내지 않는 일반적인 벤치마크를 사용합니다.
- 고객의 피드백과 관점을 유일한 벤치마크로 삼습니다.

이 모범 사례 확립의 이점: 현재 구현을 벤치마킹하면 성능 개선을 측정할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

통합 테스트에서 벤치마킹을 사용하여 워크로드 구성 요소의 성능을 평가합니다. 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 구축하거나 TPC-DS 등의 업계 표준 테스트를 사용하여 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 확인하세요.

벤치마크는 대개 로드 테스트보다 더 빠르게 실행되므로 배포 파이프라인 초기에 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 테스트에서 프로덕션 환경의 워크로드 성능에 대한 정보를 얻을 수 있기 때문입니다.

구현 단계

- 계획 및 정의:
  - 벤치마크의 목표, 기준, 테스트 시나리오, 지표(예: CPU 사용률, 지연 시간, 처리량) 및 KPI를 정의합니다.
  - 사용자 경험 및 응답 시간, 접근성과 같은 요소의 측면에서 사용자 요구 사항에 중점을 둡니다.
  - 워크로드에 적합한 벤치마킹 도구를 식별합니다. AWS 서비스(예: Amazon CloudWatch) 또는 워크로드와 호환되는 서드파티 도구를 사용할 수 있습니다.
- 구성 및 계측:
  - 환경을 설정하고 리소스를 구성합니다.
  - 모니터링 및 로깅을 구현하여 테스트 결과를 캡처합니다.
- 벤치마크 및 모니터링:
  - 벤치마크 테스트를 수행하고 테스트 중에 지표를 모니터링합니다.
- 분석 및 문서화:
  - 벤치마킹 프로세스 및 조사 결과를 문서화합니다.
  - 결과를 분석하여 병목 현상, 추세 및 개선 영역을 식별합니다.
  - 테스트 결과를 사용하여 아키텍처 결정을 내리고 워크로드를 조정합니다. 이 과정에는 서비스가 변경되거나 새로운 기능이 도입될 수 있습니다.
- 최적화 및 반복:
  - 벤치마크를 기반으로 리소스 구성 및 할당을 조정합니다.
  - 조정 후 워크로드를 다시 테스트하여 개선 여부를 검증합니다.
  - 발견한 사실을 문서화하고 프로세스를 반복하여 개선이 필요한 다른 영역을 식별합니다.

리소스

관련 문서:

- AWS 아키텍처 센터
- AWS Partner Network
- AWS Solutions Library
- AWS 지식 센터
- Amazon CloudWatch RUM
- Amazon CloudWatch Synthetics

- Genomics workflows, Part 5: automated benchmarking
- Benchmark and optimize endpoint deployment in Amazon SageMaker AI JumpStart

관련 비디오:

- AWS re:Invent 2023 - Benchmarking AWS Lambda cold starts
- Benchmarking stateful services in the cloud
- This is my Architecture
- Optimize applications through Amazon CloudWatch RUM
- Demo of Amazon CloudWatch Synthetics

관련 예제:

- AWS 샘플
- AWS SDK Examples
- Distributed Load Tests
- Measure page load time with Amazon CloudWatch Synthetics
- Amazon CloudWatch RUM Web Client

PERF01-BP07 아키텍처 선택에 데이터 기반 접근 방식 사용

아키텍처 선택에 대한 명확한 데이터 기반 접근 방식을 정의하여 특정 비즈니스 요구 사항을 충족하는 데 적합한 클라우드 서비스 및 구성이 사용되는지 확인합니다.

일반적인 안티 패턴:

- 시간이 지나면 현재 아키텍처가 정적 아키텍처가 되고 업데이트하지 않아도 된다고 가정합니다.
- 아키텍처는 추측과 가정을 기반으로 선택됩니다.
- 시간이 지나면 타당한 이유 없이 아키텍처 변경을 도입합니다.

이 모범 사례 확립의 이점: 아키텍처 선택에 대한 접근 방식을 잘 정의하면 데이터를 사용하여 워크로드 설계에 영향을 미치고 시간이 지남에 따라 정보에 입각한 의사 결정을 내릴 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

클라우드에 대한 내부 경험과 지식 또는 공개된 사용 사례나 백서 등의 외부 리소스를 사용하여 아키텍처에서 리소스와 서비스를 선택합니다. 워크로드에 사용할 수 있는 서비스를 실험하고 벤치마킹할 수 있는 잘 정의된 프로세스를 갖추고 있어야 합니다.

중요한 워크로드의 백로그는 비즈니스 및 사용자와 관련된 기능을 제공하는 사용자 스토리뿐만 아니라 워크로드의 아키텍처 런웨이를 형성하는 기술 스토리로 구성되어야 합니다. 이 런웨이는 새로운 기술 발전과 새로운 서비스에 대한 정보를 수집하고 데이터와 적절한 근거를 바탕으로 이를 채택합니다. 이를 통해 아키텍처가 미래에 대비할 수 있고 정체되지 않을 수 있습니다.

구현 단계

- 주요 이해관계자와 협력하여 성능, 가용성 및 비용 고려 사항을 포함한 워크로드 요구 사항을 정의합니다. 워크로드의 사용자 수 및 사용 패턴과 같은 요소를 고려하세요.
- 기능 백로그와 함께 우선순위가 지정된 아키텍처 런웨이 또는 기술 백로그를 생성합니다.
- 다양한 클라우드 서비스를 평가합니다(자세한 내용은 PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해 참조).
- 성능 요구 사항을 충족하는 마이크로서비스 또는 서버리스와 같은 다양한 아키텍처 패턴을 탐색합니다(자세한 내용은 PERF01-BP02 클라우드 제공업체 또는 적절한 파트너의 지침을 사용하여 아키텍처 패턴 및 모범 사례에 대해 알아보기 참조).
- 다른 팀, 아키텍처 다이어그램 및 리소스(예: AWS Solution Architects, AWS 아키텍처 센터, AWS Partner Network)를 참조하여 워크로드에 적합한 아키텍처를 선택하는 데 도움을 받으세요.
- 워크로드의 성능을 평가하는 데 도움이 될 수 있는 처리량 및 응답 시간과 같은 성과 지표를 정의합니다.
- 정의된 지표를 실험하고 사용하여 선택한 아키텍처의 성능을 검증합니다.
- 아키텍처의 성능을 최적으로 유지하기 위해 지속적으로 모니터링하고 필요에 따라 조정합니다.
- 선택한 아키텍처와 결정 사항을 문서화하여 향후 업데이트 및 학습을 위한 참고 자료로 활용합니다.
- 학습한 내용, 새로운 기술, 현재 접근 방식에서 필요한 변경이나 문제를 나타내는 지표를 기반으로 아키텍처 선택 접근 방식을 지속적으로 검토하고 업데이트합니다.

리소스

관련 문서:

- AWS Solutions Library
- AWS 지식 센터
- Architectural Patterns to Build End-to-End Data Driven Applications on AWS

관련 비디오:

- This is my Architecture
- AWS re:Invent 2021 - Data-driven enterprise: Going from vision to value
- AWS re:Invent 2022 - Delivering sustainable, high-performing architectures
- AWS re:Invent 2023 - Optimize cost and performance and track progress toward mitigation
- AWS re:Invent 2022 - AWS optimization: Actionable steps for immediate results

관련 예제:

- AWS 샘플
- AWS SDK Examples

컴퓨팅 및 하드웨어

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.

이 중점 영역에서는 클라우드의 성능 효율성을 위해 컴퓨팅 옵션을 식별하고 최적화하는 방법에 대한 지침과 모범 사례를 공유합니다.

모범 사례

- PERF02-BP01 워크로드에 가장 적합한 컴퓨팅 옵션 선택
- PERF02-BP02 사용 가능한 컴퓨팅 구성 및 기능 파악
- PERF02-BP03 컴퓨팅 관련 지표 수집
- PERF02-BP04 컴퓨팅 리소스 구성 및 적정 크기 조정
- PERF02-BP05 컴퓨팅 리소스 동적 확장
- PERF02-BP06 최적화된 하드웨어 기반 컴퓨팅 액셀러레이터 사용

PERF02-BP01 워크로드에 가장 적합한 컴퓨팅 옵션 선택

워크로드에 가장 적합한 컴퓨팅 옵션을 선택하면 성능을 향상하고 불필요한 인프라 비용을 줄이며 워크로드를 유지하는 데 필요한 운영 노력을 줄일 수 있습니다.

일반적인 안티 패턴:

- 온프레미스에서 사용한 것과 동일한 컴퓨팅 옵션을 사용합니다.
- 클라우드 컴퓨팅 옵션, 기능 및 솔루션과 이러한 솔루션이 컴퓨팅 성능을 어떻게 개선할 수 있는지 잘 모릅니다.
- 대체 컴퓨팅 옵션이 워크로드 특성에 보다 적절한데도 규모 조정 또는 성능 요구 사항을 충족하려고 기존 컴퓨팅 옵션을 과도하게 프로비저닝합니다.

이 모범 사례 확립의 이점: 컴퓨팅 요구 사항을 파악하고 사용 가능한 옵션과 비교하여 평가하면 워크로드를 보다 리소스 효율적으로 만들 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

성능 효율성을 위해 클라우드 워크로드를 최적화하려면 사용 사례 및 성능 요구 사항에 가장 적합한 컴퓨팅 옵션을 선택하는 것이 중요합니다. AWS에서는 클라우드의 다양한 워크로드에 맞는 다양한 컴퓨팅 옵션을 제공합니다. 예를 들어, 가상 서버를 시작하고 관리하려면 Amazon EC2를 사용하고, 서버를 프로비저닝하거나 관리할 필요 없이 코드를 실행하려면 AWS Lambda를 사용하며, 컨테이너를 실행하고 관리하려면 Amazon ECS 또는 Amazon EKS를 사용하고, 대량의 데이터를 병렬로 처리하려면 AWS Batch을 사용할 수 있습니다. 규모와 컴퓨팅 요구 사항에 따라 상황에 맞는 최적의 컴퓨팅 솔루션을 선택하고 구성해야 합니다. 또한 각 솔루션에는 고유한 장단점이 있으므로 단일 워크로드에 여러 유형의 컴퓨팅 솔루션을 사용하는 것도 고려할 수 있습니다.

다음 단계에서는 워크로드 특성 및 성능 요구 사항에 맞는 올바른 컴퓨팅 옵션을 선택하는 방법을 안내합니다.

구현 단계

- 워크로드 컴퓨팅 요구 사항을 이해합니다. 고려해야 할 주요 요구 사항에는 처리 요구 사항, 트래픽 패턴, 데이터 액세스 패턴, 규모 조정 요구 사항, 지연 시간 요구 사항 등이 포함됩니다.
- 워크로드에 맞는 다양한 AWS 컴퓨팅 서비스에 대해 알아보세요. 자세한 내용은 PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해 단원을 참조하십시오. 다음은 몇 가지 주요 AWS 컴퓨팅 옵션, 특성 및 일반적인 사용 사례입니다.

AWS 서비스	주요 특징	일반 사용 사례
Amazon Elastic Compute Cloud(Amazon EC2)	하드웨어, 라이선스 요구 사항, 다양한 인스턴스 제품군, 프로세서 유형 및 컴퓨팅 액셀러레이터에 대한 전용 옵션 제공	리프트 앤 시프트 마이그레이션, 단일 애플리케이션, 하이브리드 환경, 엔터프라이즈 애플리케이션
Amazon Elastic Container Service(Amazon ECS), Amazon Elastic Kubernetes Service(Amazon EKS)	간편한 배포, 일관된 환경, 확장 가능	마이크로서비스, 하이브리드 환경
AWS Lambda	서버리스 컴퓨팅 서비스(이벤트에 응답하여 코드를 실행하	마이크로서비스, 이벤트 기반 애플리케이션

AWS 서비스	주요 특징	일반 사용 사례
	고 기본 컴퓨팅 리소스를 자동으로 관리하는 서비스).	
AWS Batch	효율적이고 동적으로 프로비저닝 및 규모 조정 Amazon Elastic Container Service(Amazon ECS), Amazon Elastic Kubernetes Service(Amazon EKS) 및 AWS Fargate 컴퓨팅 리소스(작업 요구 사항에 따라 온디맨드 또는 스팟 인스턴스를 사용할 수 있는 옵션 포함)	HPC, ML 모델 학습
Amazon Lightsail	소규모 워크로드를 실행할 수 있도록 사전 구성된 Linux 및 Windows 애플리케이션	간단한 웹 애플리케이션, 맞춤형 웹 사이트

- 각 컴퓨팅 옵션과 관련된 비용(시간당 과금 또는 데이터 전송 등) 및 관리 오버헤드(패치 및 규모 조정 등)를 평가합니다.
- 비프로덕션 환경에서 실험 및 벤치마킹을 수행하여 워크로드 요구 사항을 가장 잘 해결할 수 있는 컴퓨팅 옵션을 파악합니다.
- 새로운 컴퓨팅 솔루션을 실험하고 파악한 후에는 마이그레이션을 계획하고 성과 지표를 검증합니다.
- Amazon CloudWatch와 같은 AWS 모니터링 도구 및 AWS Compute Optimizer와 같은 최적화 서비스를 사용해 실제 사용 패턴에 따라 지속적으로 컴퓨팅 리소스를 최적화합니다.

리소스

관련 문서:

- AWS에서의 클라우드 컴퓨팅
- Amazon EC2 인스턴스 유형
- Amazon EKS Containers: Amazon EKS Worker Nodes
- Amazon ECS 컨테이너: Amazon ECS 컨테이너

- 함수: Lambda 함수 구성
- Prescriptive Guidance for Containers
- Prescriptive Guidance for Serverless

관련 비디오:

- AWS re:Invent 2023 - AWS Graviton: The best price performance for your AWS workloads
- AWS re:Invent 2023 - New Amazon Elastic Compute Cloud generative AI capabilities in AMS
- AWS re:Invent 2023 - What's new with Amazon Elastic Compute Cloud
- AWS re:Invent 2023 - Smart savings: Amazon Elastic Compute Cloud cost-optimization strategies
- AWS re:Invent 2021 - Powering next-gen Amazon Elastic Compute Cloud: Deep dive on the Nitro System
- AWS re:Invent 2019 - Optimize performance and cost for your AWS compute
- AWS re:Invent 2019 - Amazon Elastic Compute Cloud foundations
- AWS re:Invent 2022 - Deploy ML models for inference at high performance and low cost
- AWS re:Invent 2019 - Optimize performance and cost for your AWS compute
- Amazon EC2 foundations
- 추론을 위한 ML 모델을 고성능 및 저렴한 비용으로 배포

관련 예제:

- Migrating the Web application to containers
- Run a Serverless Hello World
- Amazon EKS 워크샵
- Amazon EC2 워크숍
- Efficient and Resilient Workloads with Amazon Elastic Compute Cloud Auto Scaling
- Migrating to AWS Graviton with Container Services

PERF02-BP02 사용 가능한 컴퓨팅 구성 및 기능 파악

컴퓨팅 서비스에 사용 가능한 구성 옵션과 기능을 이해하면 적절한 양의 리소스를 프로비저닝하고 성능 효율성을 개선하는 데 도움이 됩니다.

일반적인 안티 패턴:

- 워크로드 특성에 대해 컴퓨팅 옵션 또는 사용 가능한 인스턴스 제품군을 평가하지 않습니다.
- 최대 수요 요구 사항을 충족하기 위해 컴퓨팅 리소스를 과도하게 프로비저닝합니다.

이 모범 사례 확립의 이점: AWS 컴퓨팅 기능 및 구성을 숙지하여 워크로드 특성과 필요 사항에 맞게 최적화된 컴퓨팅 솔루션을 사용할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

각 컴퓨팅 솔루션에는 다양한 워크로드 특성과 요구 사항을 지원하기 위해 사용할 수 있는 고유한 구성과 기능이 있습니다. 이러한 옵션을 통해 워크로드를 보완할 수 있는 방식을 알아보고 애플리케이션에 가장 적합한 구성 옵션을 결정합니다. 이러한 옵션의 예로는 인스턴스 패밀리, 크기, 기능(GPU, I/O), 버스팅, 시간 초과, 함수 크기, 컨테이너 인스턴스 및 동시성이 있습니다. 워크로드에서 4주 이상 동일한 컴퓨팅 옵션을 사용하고 있으며 앞으로도 특성이 변하지 않을 것으로 예상되는 경우 AWS Compute Optimizer를 사용하여 CPU 및 메모리 관점에서 현재 컴퓨팅 옵션이 워크로드에 적합한지 알아볼 수 있습니다.

구현 단계

- 워크로드 요구 사항(예: CPU 필요량, 메모리, 지연 시간)을 이해합니다.
- AWS 설명서 및 모범 사례를 검토하여 컴퓨팅 성능을 개선하는 데 도움이 되는 권장 구성 옵션에 대해 알아봅니다. 고려해야 할 몇 가지 주요 구성 옵션은 다음과 같습니다.

구성 옵션	예시
인스턴스 유형	• 컴퓨팅 최적화 인스턴스는 높은 vCPU 대 메모리 비율이 필요한 워크로드에 이상적입니다. • 메모리 최적화 인스턴스는 상당한 메모리를 제공하여 메모리를 많이 사용하는 워크로드를 지원합니다. • 스토리지 최적화 인스턴스는 로컬 스토리지에 대해 높은 순차 읽기 및 쓰기 액세스

구성 옵션	예시
	(IOPS)가 필요한 워크로드에 적합하도록 설계되었습니다.
요금 모델	• 온디맨드 인스턴스를 사용하면 장기 약정 없이 시간 또는 초 단위로 컴퓨팅 용량을 이용할 수 있습니다. 이러한 인스턴스는 성능 기준 요구를 넘어서 버스팅하는 데 도움이 됩니다. • 절감형 플랜은 1년 또는 3년 동안 특정 양의 컴퓨팅 성능을 사용하기로 약정하는 대신 온디맨드 인스턴스에 비해 상당한 비용을 절감할 수 있습니다. • 스팟 인스턴스를 사용하면 내결함성을 갖춘 상태 비저장 워크로드를 위해 미사용 인스턴스 용량을 할인된 가격으로 활용할 수 있습니다.
Auto Scaling	Auto Scaling 구성을 사용하여 컴퓨팅 리소스를 트래픽 패턴에 일치시킵니다.
크기 조정	• Compute Optimizer를 사용하여 기계 학습 기반 추천을 통해 컴퓨팅 특성에 가장 적합한 컴퓨팅 구성을 추천받습니다. • AWS Lambda Power Tuning을 사용하여 Lambda 함수에 가장 적합한 구성을 선택합니다.
하드웨어 기반 컴퓨팅 액셀러레이터	• 가속 컴퓨팅 인스턴스는 그래픽 처리 또는 데이터 패턴 일치와 같은 기능을 CPU 기반 대안보다 더 효율적으로 수행합니다. • 기계 학습 워크로드의 경우 AWS Trainium, AWS Inferentia, Amazon EC2 DL1과 같이 워크로드에 특정한 목적별 하드웨어를 활용합니다.

리소스

관련 문서:

- AWS에서의 클라우드 컴퓨팅
- Amazon EC2 인스턴스 유형
- Amazon EC2 Linux 인스턴스에 대한 프로세서 상태 제어
- Amazon EKS Containers: Amazon EKS Worker Nodes
- Amazon ECS 컨테이너: Amazon ECS 컨테이너
- 함수: Lambda 함수 구성

관련 비디오:

- AWS re:Invent 2023 – AWS Graviton: The best price performance for your AWS workloads
- AWS re:Invent 2023 – New Amazon EC2 generative AI capabilities in AWS Management Console
- AWS re:Invent 2023 – What's new with Amazon EC2
- AWS re:Invent 2023 – Smart savings: Amazon EC2 cost-optimization strategies
- AWS re:Invent 2021 – Powering next-gen Amazon EC2: Deep dive on the Nitro System
- AWS re:Invent 2019 – Amazon EC2 foundations
- AWS re:Invent 2022 – Optimizing Amazon EKS for performance and cost on AWS

관련 예제:

- Compute Optimizer demo code
- Amazon EC2 spot instances 워크숍
- Efficient and Resilient Workloads with Amazon EC2 AWS Auto Scaling
- Graviton 개발자 워크숍
- AWS for Microsoft workloads immersion day
- AWS for Linux workloads immersion day
- AWS Compute Optimizer Demo code
- Amazon EKS 워크숍

PERF02-BP03 컴퓨팅 관련 지표 수집

컴퓨팅 관련 지표를 기록하고 추적하여 컴퓨팅 리소스의 성능을 더 잘 파악하고 성능과 사용률을 높입니다.

일반적인 안티 패턴:

- 지표에 대해 수동 로그 파일 검색만 사용합니다.
- 해당 모니터링 소프트웨어에서 기록한 기본 지표만 사용합니다.
- 문제가 발생한 경우에만 지표를 검토합니다.

이 모범 사례 확립의 이점: 성능 관련 지표를 수집하면 애플리케이션 성능을 비즈니스 요구 사항에 맞게 조정하여 워크로드 필요 사항을 충족하는 데 도움이 됩니다. 또한 워크로드의 리소스 성능과 사용률을 지속적으로 개선하는 데 도움이 될 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

클라우드 워크로드를 통해 지표, 로그 및 이벤트와 같은 대량의 데이터가 생성될 수 있습니다. AWS 클라우드에서 지표 수집은 보안, 비용 효율성, 성능 및 지속 가능성을 개선하기 위한 중요한 단계입니다. AWS는 Amazon CloudWatch와 같은 모니터링 서비스를 통해 다양한 성능 관련 지표를 제공하여 중요한 인사이트를 제공합니다. CPU 사용률, 메모리 사용률, 디스크 I/O, 네트워크 인바운드 및 아웃바운드와 같은 지표는 사용률 수준 또는 성능 병목 현상에 대한 인사이트를 제공할 수 있습니다. 데이터 기반 접근 방식의 일환으로 이 지표를 사용하면 워크로드 리소스를 능동적으로 튜닝하고 최적화할 수 있습니다. 비용 및 운영 목표를 지원하기 위해 보존 정책이 구현된 단일 플랫폼에서 컴퓨팅 리소스와 관련된 모든 지표를 수집하는 것이 이상적입니다.

구현 단계

- 워크로드와 관련된 성능 관련 지표를 파악합니다. 리소스 사용률과 클라우드 워크로드의 운영 방식(응답 시간 및 처리량 등)에 대한 지표를 수집해야 합니다.
  - Amazon EC2 기본 지표
  - Amazon ECS 기본 지표
  - Amazon EKS 기본 지표
  - Lambda 기본 지표
  - Amazon EC2 메모리 및 디스크 지표

- 워크로드에 적합한 로깅 및 모니터링 솔루션을 선택하고 설정합니다.
  - AWS 네이티브 관찰성
  - AWS Distro for OpenTelemetry
  - Amazon Managed Service for Prometheus
- 워크로드 요구 사항에 따라 지표에 필요한 필터 및 집계를 정의합니다.
  - Quantify custom application metrics with Amazon CloudWatch Logs and metric filters
  - Collect custom metrics with Amazon CloudWatch strategic tagging
- 보안 및 운영 목표에 맞게 지표에 대한 데이터 보존 정책을 구성합니다.
  - CloudWatch 지표에 대한 기본 데이터 보존
  - CloudWatch Logs에 대한 기본 데이터 보존
- 필요한 경우 지표에 대한 경보 및 알림을 생성하여 성능 관련 문제에 미리 대응할 수 있습니다.
  - Create alarms for custom metrics using Amazon CloudWatch anomaly detection
  - Create metrics and alarms for specific web pages with Amazon CloudWatch RUM
- 자동화를 사용하여 지표 및 로그 집계 에이전트를 배포합니다.
  - AWS Systems Manager 자동화
  - OpenTelemetry Collector

리소스

관련 문서:

- 모니터링 및 관찰성
- Best practices: implementing observability with AWS
- Amazon CloudWatch 설명서
- CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버에서 지표 및 로그 수집
- AWS Lambda에 대한 Amazon CloudWatch logs 액세스
- 컨테이너 인스턴스와 함께 CloudWatch Logs 사용
- 사용자 지정 지표 게시
- AWS Answers: Centralized Logging
- CloudWatch 지표를 게시하는 AWS 서비스
- Monitoring Amazon EKS on AWS Fargate

관련 비디오:

- AWS re:Invent 2023 – [LAUNCH] Application monitoring for modern workloads
- AWS re:Invent 2023 – Implementing application observability
- AWS re:Invent 2023 – Building an effective observability strategy
- AWS re:Invent 2023 – Seamless observability with AWS Distro for OpenTelemetry
- Application Performance Management on AWS

관련 예제:

- AWS for Linux Workloads Immersion Day- Amazon CloudWatch
- Monitoring Amazon ECS clusters and containers
- Monitoring with Amazon CloudWatch dashboards
- Amazon EKS 워크숍

PERF02-BP04 컴퓨팅 리소스 구성 및 적정 크기 조정

워크로드의 성능 요구 사항에 맞게 컴퓨팅 리소스를 구성하고 규모에 맞게 크기를 조정하여 리소스의 사용률이 부족하거나 과도하게 활용되는 것을 방지하세요.

일반적인 안티 패턴:

- 워크로드 성능 요구 사항을 무시하여 컴퓨팅 리소스의 프로비저닝이 과도하거나 부족해지는 경우가 있습니다.
- 모든 워크로드에 사용할 수 있는 최소 또는 최대 인스턴스만 선택합니다.
- 관리하기 쉽도록 하나의 인스턴스 제품군만 사용합니다.
- 규모에 맞는 크기로 조정하기 위해 AWS Cost Explorer 또는 Compute Optimizer의 권장 사항을 무시합니다.
- 새로운 인스턴스 유형의 적합성을 위해 워크로드를 재평가하지 않습니다.
- 조직에 대해 소수의 인스턴스 구성만 인증합니다.

이 모범 사례 확립의 이점: 컴퓨팅 리소스의 크기를 규모에 맞게 적절히 조정하면 리소스의 프로비저닝이 과도하거나 부족해지는 것을 방지하여 클라우드 운영을 최적화할 수 있습니다. 일반적으로 컴퓨팅 리소스의 크기를 적절하게 조정하면 성능과 고객 경험이 향상되는 동시에 비용도 절감됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

규모에 맞게 크기를 조정하면 조직은 비즈니스 요구 사항을 해결하면서 효율적이고 비용 효과적인 방식으로 클라우드 인프라를 운영할 수 있습니다. 클라우드 리소스를 과도하게 프로비저닝하면 추가 비용이 발생할 수 있는 반면, 과소 프로비저닝은 성능 저하와 부정적인 고객 경험으로 이어질 수 있습니다. AWS는 AWS Compute Optimizer 및 AWS Trusted Advisor와 같은 도구를 제공합니다. 이러한 도구는 과거 데이터를 사용하여 컴퓨팅 리소스를 적정 크기로 조정하기 위한 권장 사항을 제공합니다.

구현 단계

- 요구 사항을 가장 잘 충족하는 인스턴스 유형을 선택하세요.
  - 워크로드에 적합한 EC2 인스턴스 유형을 선택하려면 어떻게 해야 하나요?
  - Amazon EC2 플릿의 속성 기반 인스턴스 유형 선택
  - 속성 기반 인스턴스 유형 선택을 사용하여 Auto Scaling 그룹 생성
  - Optimizing your Kubernetes compute costs with Karpenter consolidation
- 워크로드의 다양한 성능 특성 그리고 이러한 특성과 메모리, 네트워크, CPU 사용량 간의 관계를 분석합니다. 이 데이터를 사용하면 워크로드 프로필 및 성과 목표에 가장 적합한 리소스를 선택할 수 있습니다.
- Amazon CloudWatch와 같은 AWS 모니터링 도구를 사용하여 리소스 사용량을 모니터링합니다.
- 컴퓨팅 리소스에 적합한 구성을 선택합니다.
  - 임시 워크로드의 경우 인스턴스 Amazon CloudWatch 지표(예: `CPUUtilization`)를 평가하여 인스턴스의 사용률이 낮거나 높은지 식별합니다.
  - 안정적인 워크로드를 위해 정기적으로 AWS Compute Optimizer 및 AWS Trusted Advisor 등의 AWS 적정 크기 조정 도구를 확인하여 컴퓨팅 리소스를 최적화하고 크기를 조정할 수 있는 기회를 식별합니다.
- 실제 환경에서 구현하기 전에 비프로덕션 환경에서 구성 변경 사항을 테스트합니다.
- 새로운 컴퓨팅 오퍼링을 지속적으로 재평가하고 워크로드 요구 사항과 비교합니다.

리소스

관련 문서:

- AWS에서의 클라우드 컴퓨팅

- Amazon EC2 인스턴스 유형
- Amazon ECS 컨테이너: Amazon ECS 컨테이너
- Amazon EKS Containers: Amazon EKS Worker Nodes
- 함수: Lambda 함수 구성
- Amazon EC2 Linux 인스턴스에 대한 프로세서 상태 제어

관련 비디오:

- Amazon EC2 foundations
- AWS re:Invent 2023 – AWS Graviton: The best price performance for your AWS workloads
- AWS re:Invent 2023 – New Amazon EC2 generative AI capabilities in AWS Management Console
- AWS re:Invent 2023 – What's new with Amazon EC2
- AWS re:Invent 2023 – Smart savings: Amazon EC2 cost-optimization strategies
- AWS re:Invent 2021 – Powering next-gen Amazon EC2: Deep dive on the Nitro System
- AWS re:Invent 2019 – Amazon EC2 foundations

관련 예제:

- AWS Compute Optimizer Demo code
- Amazon EKS 워크숍
- Right-sizing recommendations

PERF02-BP05 컴퓨팅 리소스 동적 확장

클라우드의 탄력성을 사용하여 필요에 따라 컴퓨팅 리소스를 동적으로 스케일 업 또는 스케일 다운하고 워크로드에 대한 프로비저닝 용량이 과도하거나 부족하지 않도록 방지할 수 있습니다.

일반적인 안티 패턴:

- 용량을 수동으로 늘려 경보에 대응합니다.
- 온프레미스에서와 동일한 크기 조정 가이드라인(일반적으로 정적 인프라)을 사용합니다.
- 조정 이벤트 후에 다시 스케일 다운하는 대신 증가된 용량을 그대로 둡니다.

이 모범 사례 확립의 이점: 컴퓨팅 리소스의 탄력성을 구성 및 테스트하면 비용을 절감하고, 성능 벤치마크를 유지하며, 트래픽 변화에 따른 신뢰성을 개선할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

AWS는 다양한 조정 메커니즘을 통해 리소스를 동적으로 스케일 업 또는 스케일 다운하여 수요 변화에 대응할 수 있는 유연성을 제공합니다. 컴퓨팅 관련 지표와 결합된 동적 규모 조정을 통해 워크로드는 변경 사항에 자동으로 대응하고 최적의 컴퓨팅 리소스 세트를 사용하여 목표를 달성할 수 있습니다.

다양한 접근 방식을 사용하여 리소스 공급과 수요를 일치시킬 수 있습니다.

- 타겟 추적 접근 방식: 규모 조정 지표를 모니터링하고 필요에 따라 용량을 자동으로 늘리거나 줄입니다.
- 예측 규모 조정: 일별 및 주별 추세를 고려하여 스케일 인합니다.
- 일정 기반 접근 방식: 예측 가능한 로드 변화에 따라 자체 규모 조정 일정을 설정합니다.
- 서비스 규모 조정: 설계상 자동으로 규모가 조정되는 서비스(예: 서버리스)를 선택합니다.

워크로드 배포에서 스케일 업 및 스케일 다운 이벤트를 모두 처리할 수 있는지 확인해야 합니다.

구현 단계

- 컴퓨팅 인스턴스, 컨테이너 및 함수는 Auto Scaling과 함께 또는 서비스의 기능으로 탄력성을 위한 메커니즘을 제공합니다. 다음은 자동 규모 조정 메커니즘 예제입니다.

Auto Scaling 메커니즘	사용 장소
Amazon EC2 Auto Scaling	애플리케이션의 로드를 처리할 수 있는 올바른 수의 Amazon EC2 인스턴스를 유지하도록 보장합니다.
Application Auto Scaling	AWS Lambda 함수 또는 Amazon Elastic Container Service(Amazon ECS) 서비스와 같이 Amazon EC2 이외의 개별 AWS 서비스에서 리소스 규모를 자동으로 조정합니다.
Kubernetes Cluster Autoscaler/Karpenter	Kubernetes 클러스터를 자동으로 조정합니다.

- 규모 조정은 대개 Amazon EC2 인스턴스나 AWS Lambda 함수 등의 컴퓨팅 서비스와 관련하여 설명하는 경우가 많습니다. 이때 AWS Glue와 같은 비컴퓨팅 서비스의 구성도 수요에 맞게 고려해야 합니다.
- 규모 조정에 대한 지표가 배포 중인 워크로드의 특성과 일치하는지 확인합니다. 동영상 트랜스코딩 애플리케이션을 배포하는 경우 100%의 CPU 활용률이 예상되므로, 기본 지표로 사용해서는 안 됩니다. 대신 트랜스코딩 작업 대기열의 깊이를 사용합니다. 필요한 경우 규모 조정 정책에 사용자 지정 지표를 사용할 수 있습니다. 올바른 지표를 선택하려면 Amazon EC2에 대한 다음 지침을 고려하세요.
  - 지표는 유효한 사용률 지표여야 하며 인스턴스가 얼마나 많이 사용되는지를 설명해야 합니다.
  - 지표 값은 Auto Scaling 그룹의 인스턴스 수에 비례하여 증가하거나 감소합니다.
- Auto Scaling 그룹의 경우 수동 규모 조정 대신 동적 규모 조정을 사용해야 합니다. 또한 동적 규모 조정에서 목표 추적 조정 정책을 사용하는 것이 좋습니다.
- 워크로드 배포가 규모 조정 이벤트(스케일 업 및 스케일 다운)를 모두 처리할 수 있는지 확인합니다. 예를 들어 활동 내역을 사용하여 Auto Scaling 그룹의 조정 활동을 확인할 수 있습니다.
- 워크로드의 예측 가능한 패턴을 평가하고 예측 및 계획된 수요 변화에 따라 사전 예방적으로 확장합니다. 예측 규모 조정에서는 용량을 과도하게 프로비저닝할 필요가 없습니다. 자세한 내용은 Predictive Scaling with Amazon EC2 Auto Scaling을 참조하세요.

리소스

관련 문서:

- AWS에서의 클라우드 컴퓨팅
- Amazon EC2 인스턴스 유형
- Amazon ECS 컨테이너: Amazon ECS 컨테이너
- Amazon EKS Containers: Amazon EKS Worker Nodes
- 함수: Lambda 함수 구성
- Amazon EC2 Linux 인스턴스에 대한 프로세서 상태 제어
- Deep Dive on Amazon ECS Cluster Auto Scaling
- Introducing Karpenter – An Open-Source High-Performance Kubernetes Cluster Autoscaler

관련 비디오:

- AWS re:Invent 2023 – AWS Graviton: The best price performance for your AWS workloads

- AWS re:Invent 2023 – New Amazon EC2 generative AI capabilities in AWS Management Console
- AWS re:Invent 2023 – What's new with Amazon EC2
- AWS re:Invent 2023 – Smart savings: Amazon EC2 cost-optimization strategies
- AWS re:Invent 2021 – Powering next-gen Amazon EC2: Deep dive on the Nitro System
- AWS re:Invent 2019 – Amazon EC2 foundations

관련 예제:

- Amazon EC2 Auto Scaling Group Examples
- Amazon EKS 워크샵
- Scale your Amazon EKS workloads by running on IPv6

PERF02-BP06 최적화된 하드웨어 기반 컴퓨팅 액셀러레이터 사용

하드웨어 액셀러레이터를 사용하면 CPU 기반 대안보다 특정 기능을 더 효율적으로 수행할 수 있습니다.

일반적인 안티 패턴:

- 워크로드에서 범용 인스턴스를 더 높은 성능과 더 낮은 비용을 제공할 수 있는 목적별 인스턴스와 비교하여 벤치마킹하지 않았습니다.
- CPU 기반 대안을 사용하는 것이 더 효율적일 수 있는 작업에 하드웨어 기반 컴퓨팅 액셀러레이터를 사용합니다.
- GPU 사용을 모니터링하지 않습니다.

이 모범 사례 확립의 이점: 그래픽 처리 디바이스(GPU) 및 Field Programmable Gate Array(FPGA)와 같은 하드웨어 기반 액셀러레이터를 사용하면 특정 프로세싱 기능을 보다 효율적으로 수행할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

가속 컴퓨팅 인스턴스는 GPU 및 FPGA와 같은 하드웨어 기반 컴퓨팅 가속기에 대한 액세스를 제공합니다. 이러한 하드웨어 액셀러레이터는 CPU 기반 대안보다 더 효율적인 그래픽 처리 또는 데이터 패턴

일치와 같은 특정 기능을 수행합니다. 렌더링, 트랜스코딩, 기계 학습 등 많은 가속 워크로드는 리소스 사용 면에서 매우 가변적입니다. 이 하드웨어는 필요한 시간 동안만 실행하고 필요하지 않은 경우 자동화를 통해 해제하여 전반적인 성능 효율성을 향상합니다.

구현 단계

- 요구 사항을 해결할 수 있는 가속 컴퓨팅 인스턴스를 식별합니다.
- 기계 학습 워크로드의 경우 AWS Trainium, AWS Inferentia 및 Amazon EC2 DL1과 같이 워크로드에 따라 특정한 목적별 하드웨어의 장점을 활용합니다. AWS Inf2 인스턴스와 같은 Inferentia 인스턴스는 동급 Amazon EC2 인스턴스에 비해 최대 50% 더 우수한 와트당 성능을 제공합니다.
- 가속 컴퓨팅 인스턴스의 사용량 지표를 수집합니다. 예를 들어, Amazon CloudWatch의 NVIDIA GPU 지표 수집에서와 같이 CloudWatch 에이전트를 사용하여 GPU에 대한 `utilization_gpu` 및 `utilization_memory`와 같은 지표를 수집할 수 있습니다.
- 코드, 네트워크 운영 및 하드웨어 액셀러레이터 설정을 최적화하여 기본 하드웨어가 반드시 제대로 활용되도록 해야 합니다.
  - GPU 설정 최적화
  - GPU Monitoring and Optimization in the Deep Learning AMI
  - Optimizing I/O for GPU performance tuning of deep learning training in Amazon SageMaker AI
- 최신 고성능 라이브러리 및 GPU 드라이버를 사용합니다.
- 자동화를 사용하여 사용하지 않는 GPU 인스턴스 사용을 해제합니다.

리소스

관련 문서:

- Amazon Elastic Container Service에서 GPU 작업
- GPU 인스턴스
- Instances with AWS Trainium
- Instances with AWS Inferentia
- Let’s Architect! Architecting with custom chips and accelerators

- 가속 컴퓨팅
- Amazon EC2 VT1 Instances
- 워크로드에 적합한 EC2 인스턴스 유형을 선택하려면 어떻게 해야 하나요?

- Choose the best AI accelerator and model compilation for computer vision inference with Amazon SageMaker AI

관련 비디오:

- AWS re:Invent 2021 - How to select Amazon Elastic Compute Cloud GPU instances for deep learning
- AWS re:Invent 2022 - [NEW LAUNCH!] Introducing AWS Inferentia2-based Amazon EC2 Inf2 instances
- AWS re:Invent 2022 - Accelerate deep learning and innovate faster with AWS Trainium
- AWS re:Invent 2022 - Deep learning on AWS with NVIDIA: From training to deployment

관련 예제:

- Amazon SageMaker AI and NVIDIA GPU Cloud (NGC)
- Use SageMaker AI with Trainium and Inferentia for optimized deep learning training and inferencing workloads
- Optimizing NLP models with Amazon Elastic Compute Cloud Inf1 instances in Amazon SageMaker AI

데이터 관리

특정 시스템에 대한 최적의 스토리지 솔루션은 데이터 유형의 종류(블록, 파일, 객체), 액세스 패턴(랜덤 또는 순차), 필요한 처리량, 액세스 빈도(온라인, 오프라인, 아카이브), 업데이트 빈도(WORM, 동적) 및 가용성과 내구성 제약 사항에 따라 다릅니다. Well-Architected 워크로드는 용도에 맞게 구축된 데이터 저장소를 사용하므로 다양한 기능을 통해 성능을 개선할 수 있습니다.

이 중점 영역에서는 데이터 스토리지, 이동 및 액세스 패턴, 데이터 스토어의 성능 효율성을 최적화하기 위한 지침과 모범 사례를 공유합니다.

모범 사례

- PERF03-BP01 데이터 액세스 및 스토리지 요구 사항을 가장 잘 지원하는 목적별 데이터 스토어 사용
- PERF03-BP02 데이터 스토어에 사용 가능한 구성 옵션 평가
- PERF03-BP03 데이터 스토어 성과 지표 수집 및 기록
- PERF03-BP04 데이터 스토어에서 쿼리 성능을 개선하기 위한 전략 구현
- PERF03-BP05 캐싱을 활용하는 데이터 액세스 패턴 구현

PERF03-BP01 데이터 액세스 및 스토리지 요구 사항을 가장 잘 지원하는 목적별 데이터 스토어 사용

데이터 특성(공유 가능 여부, 크기, 캐시 크기, 액세스 패턴, 지연 시간, 처리량, 데이터 지속성 등)을 이해하여 워크로드에 적합한 목적별 데이터 스토어(스토리지 또는 데이터베이스)를 선택해야 합니다.

일반적인 안티 패턴:

- 하나의 특정 데이터베이스 솔루션에 대한 내부 경험과 지식만 갖춘 탓에 하나의 데이터 스토어만 고수합니다.
- 모든 워크로드의 데이터 스토리지 및 액세스 요구 사항이 비슷하다고 가정합니다.
- 데이터 자산의 인벤토리 등록을 위한 데이터 카탈로그를 구현하지 않았습니다.

이 모범 사례 확립의 이점: 데이터 특성과 요구 사항을 파악하면 워크로드 요구 사항을 충족하는 가장 효율적이고 성능이 뛰어난 스토리지 기술을 결정할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

데이터 스토리지를 선택하고 구현할 때는 쿼리, 규모 조정 및 스토리지 특성이 워크로드 데이터 요구 사항을 지원하는지 확인해야 합니다. AWS는 블록 스토리지, 객체 스토리지, 스트리밍 스토리지, 파일 시스템, 관계형, 키 값, 문서, 인메모리, 그래프, 시계열, 원장 데이터베이스를 포함한 다양한 데이터 스토리지 및 데이터베이스 기술을 제공합니다. 각 데이터 관리 솔루션에는 사용 사례 및 데이터 모델을 지원하는 옵션과 구성이 있습니다. 데이터 특성과 요구 사항을 이해하면 모놀리식 스토리지 기술과 제한적인 획일적 접근 방식에서 벗어나 데이터를 적절하게 관리하는 데 집중할 수 있습니다.

구현 단계

- 워크로드에 존재하는 다양한 데이터 유형의 인벤토리를 수행합니다.
- 다음과 같은 데이터 특성 및 요구 사항을 이해하고 문서화합니다.
  - 데이터 형식(비정형, 반정형, 관계형)
  - 데이터 볼륨 및 증가
  - 데이터 내구성: 영구, 임시, 일시적
  - 원자성, 일관성, 격리, 내구성(ACID) 요구 사항
  - 데이터 액세스 패턴(읽기 중심 또는 쓰기 중심)
  - 지연 시간
  - 처리량
  - 초당 입출력 연산 수(IOPS)
  - 데이터 보존 기간
- 데이터 특성에 맞는 AWS의 워크로드에 사용할 수 있는 다양한 데이터 스토어(스토리지 및 데이터베이스 서비스)에 대해 알아봅니다(PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해 참조). 다음은 AWS 스토리지 기술의 몇 가지 예와 주요 특성입니다.

유형	AWS 서비스	주요 특징
객체 스토리지	Amazon S3	무제한 확장성, 고가용성, 다양한 접근성 옵션. Amazon S3 안팎에서 객체를 전송하고 객체에 액세스할 때 전송 가속화 또는 액세스 포인트와 같은 서비스를 사용하여 위치, 보안

유형	AWS 서비스	주요 특징
		요구 사항 및 액세스 패턴을 지원할 수 있습니다.
아카이빙 스토리지	Amazon S3 Glacier	데이터 보관을 위해 제작되었습니다.
스트리밍 스토리지	Amazon Kinesis Amazon Managed Streaming for Apache Kafka(Amazon MSK)	스트리밍 데이터의 효율적인 수집 및 저장.
공유 파일 시스템	Amazon Elastic File System(Amazon EFS)	여러 유형의 컴퓨팅 솔루션에서 액세스할 수 있는 탑재 가능한 파일 시스템입니다.
공유 파일 시스템	Amazon FSx	최신 AWS 컴퓨팅 솔루션을 기반으로 구축되어 일반적으로 사용되는 네 가지 파일 시스템인 NetApp ONTAP, OpenZFS, Windows 파일 서버, Lustre를 지원합니다. Amazon FSx 지연 시간, 처리량, IOPS는 파일 시스템에 따라 달라지며 워크로드의 요구 사항에 적합한 파일 시스템을 선택할 때 고려해야 합니다.

유형	AWS 서비스	주요 특징
블록 스토리지	Amazon Elastic Block Store(Amazon EBS)	Amazon Elastic Compute Cloud(Amazon EC2)를 위해 설계된 확장 가능한 고성능 블록 스토리지 서비스. Amazon EBS에는 IOPS 집약적 트랜잭션 워크로드를 위한 SSD 지원 스토리지와 처리량 집약적 워크로드를 위한 HDD 지원 스토리지가 포함됩니다.
관계형 데이터베이스	Amazon Aurora, Amazon RDS, Amazon Redshift.	원자성, 일관성, 격리, 내구성(ACID) 트랜잭션을 지원하고 참조 무결성과 강력한 데이터 일관성을 유지하도록 설계되었습니다. 많은 기존 애플리케이션, 엔터프라이즈 리소스 계획(ERP), 고객 관계 관리(CRM) 및 전자 상거래의 데이터가 관계형 데이터베이스를 사용하여 저장됩니다.
키 값 데이터베이스	Amazon DynamoDB	대개 대량의 데이터를 저장 및 검색하는 일반적인 접근 패턴에 최적화되어 있습니다. 트래픽이 많은 웹 앱, 전자 상거래 시스템, 게임 애플리케이션은 키 값 데이터베이스의 일반적인 사용 사례입니다.

유형	AWS 서비스	주요 특징
문서 데이터베이스	Amazon DocumentDB	반정형 데이터를 JSON과 유사한 문서로 저장하도록 설계되었습니다. 이러한 데이터베이스는 개발자가 콘텐츠 관리, 카탈로그 및 사용자 프로필과 같은 애플리케이션을 신속하게 구축하고 업데이트하는 데 도움이 됩니다.
인 메모리 데이터베이스	Amazon ElastiCache, Amazon MemoryDB for Redis	데이터에 대한 실시간 액세스, 최저 지연 시간 및 최대 처리량을 요하는 애플리케이션에 사용됩니다. 애플리케이션 캐싱, 세션 관리, 게임 순위표, 저지연 ML 특성 저장소, 마이크로서비스 메시징 시스템, 고처리량 스트리밍 메커니즘에 인 메모리 데이터베이스를 사용할 수 있습니다.
그래프 데이터베이스	Amazon Neptune	밀접한 관계가 있는 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.

유형	AWS 서비스	주요 특징
시계열 데이터베이스	Amazon Timestream	시간이 지남에 따라 변화하는 데이터에서 효율적으로 분석 정보를 수집, 통합 및 도출합니다. IoT 애플리케이션, DevOps 및 산업용 원격 측정에 시계열 데이터베이스를 활용할 수 있습니다.
와이드 컬럼	Amazon Keyspaces(Apache Cassandra용)	테이블, 행 및 열을 사용하지만 관계형 데이터베이스와 달리 동일한 테이블 내의 열의 이름과 형식이 행마다 다를 수 있습니다. 일반적으로 와이드 컬럼 스토어는 대규모 산업 앱에서 장비 유지 관리, 플릿 관리 및 라우팅 최적화를 위해 사용됩니다.
원장	Amazon Quantum Ledger Database(Amazon QLDB)	모든 애플리케이션에 대해 확장 가능하고 변경 불가능하며 암호화 방식으로 확인 가능한 트랜잭션 레코드를 유지하는 신뢰할 수 있는 중앙 집중식 권한을 제공합니다. 레코드 시스템, 공급망, 등록 및 심지어 은행 거래 시스템에 원장 데이터베이스가 사용되는 것을 볼 수 있습니다.

- 데이터 플랫폼을 구축하는 경우 AWS의 최신 데이터 아키텍처를 활용하여 데이터 레이크, 데이터 웨어하우스, 목적별 데이터 스토어를 통합합니다.
- 워크로드에 맞는 데이터 스토어를 선택할 때 고려해야 할 주요 질문은 다음과 같습니다.

질문	고려할 사항
데이터는 어떻게 구성되어 있나요?	• 비정형 데이터인 경우 Amazon S3와 같은 객체 저장소 또는 Amazon DocumentDB와 같은 NoSQL 데이터베이스를 고려하세요. • 키 값 데이터의 경우 DynamoDB, Amazon ElastiCache(Redis OSS) 또는 Amazon MemoryDB를 고려하세요.
어느 정도 수준의 참조 무결성이 필요한가요?	• 외래 키 제약 조건의 경우 Amazon RDS 및 Aurora와 같은 관계형 데이터베이스가 이 수준의 무결성을 제공할 수 있습니다. • 일반적으로 NoSQL 데이터 모델 내에서는 문서나 테이블을 조인하는 대신 단일 요청으로 검색하도록 데이터를 단일 문서 또는 문서 모음으로 비정규화합니다.
원자성, 일관성, 격리, 내구성(ACID) 규정을 준수해야 하나요?	• 관계형 데이터베이스와 연결된 ACID 속성이 필요한 경우 Amazon RDS 및 Aurora와 같은 관계형 데이터베이스를 고려하세요. • NoSQL 데이터베이스에 강력한 일관성이 필요한 경우 DynamoDB를 사용하여 강력하게 일관된 읽기를 사용할 수 있습니다.
시간의 경과에 따라 스토리지 요구 사항이 어떻게 변하나요? 이러한 변화가 확장성에 어떤 영향을 미치나요?	• DynamoDB 및 Amazon Quantum Ledger Database(Amazon QLDB)와 같은 서버리스 데이터베이스는 동적으로 규모가 조정됩니다. • 관계형 데이터베이스는 프로비저닝된 스토리지에 대한 상한선이 있으며, 한도에 도달하면 샤딩과 같은 메커니즘을 통해 수평으로 분할해야 하는 경우가 많습니다.

질문	고려할 사항
쓰기 쿼리 대비 읽기 쿼리의 비율이 어떻게 되나요? 캐싱으로 성능이 향상될 가능성이 있나요?	• 읽기 중심의 워크로드는 데이터베이스가 DynamoDB인 경우 ElastiCache 또는 DAX와 같은 캐싱 계층을 활용할 수 있습니다. • Amazon RDS와 같은 관계형 데이터베이스를 사용하여 읽기 복제본으로 읽기를 오프로드할 수도 있습니다.
저장 및 수정(OLTP - 온라인 트랜잭션 처리) 또는 검색 및 보고(OLAP - 온라인 분석 처리)의 우선순위가 더 높은가요?	• 처리량이 많은 읽기 그대로의 트랜잭션 처리의 경우 DynamoDB와 같은 NoSQL 데이터베이스를 고려합니다. • 처리량이 많고 일관성이 있는 복잡한 읽기 패턴(예: 조인)의 경우 Amazon RDS를 사용합니다. • 분석 쿼리의 경우 Amazon Redshift와 같은 열 형식 데이터베이스를 고려하거나 Amazon S3로 데이터를 내보내고 Athena 또는 Amazon QuickSight를 사용하여 분석을 수행하는 방법을 고려하세요.
데이터에 필요한 내구성은 어느 정도인가요?	• Aurora는 리전 내 3개의 가용 영역에서 데이터를 자동으로 복제하므로, 데이터 손실 가능성이 적으면서도 데이터의 내구성이 매우 높아집니다. • DynamoDB는 여러 가용 영역에 걸쳐 자동으로 복제되므로, 높은 가용성과 데이터 내구성을 제공합니다. • Amazon S3는 99.999999999%의 내구성을 지원합니다. Amazon RDS 및 DynamoDB와 같은 많은 데이터베이스 서비스는 장기 보존 및 아카이브를 위해 Amazon S3로 데이터 내보내기를 지원합니다.

질문	고려할 사항
상용 데이터베이스 엔진을 쓰고 싶지 않거나 라이선싱 비용을 들이고 싶지 않은가요?	• Amazon RDS 또는 Aurora에서 PostgreSQL 및 MySQL과 같은 오픈 소스 엔진을 고려합니다. • AWS Database Migration Service 및 AWS Schema Conversion Tool을 활용하여 상용 데이터베이스 엔진에서 오픈 소스로 마이그레이션을 수행합니다.
운영상 데이터베이스에 대해 어떤 점을 기대하나요? 주된 관심 사항이 관리형 서비스로 전환하는 것인가요?	• Amazon EC2 대신 Amazon RDS를 활용하고 자체 호스팅한 NoSQL 데이터베이스 대신 DocumentDB 또는 Amazon DocumentDB를 활용하여 운영 오버헤드를 줄일 수 있습니다.
현재 데이터베이스에 어떻게 액세스하고 있나요? 애플리케이션 액세스만 가능한가요? 아니면 비즈니스 인텔리전스(BI) 사용자와 기타 연결된 상용 애플리케이션이 있나요?	• 외부 도구에 대한 종속성이 있는 경우 해당 도구가 지원하는 데이터베이스와의 호환성을 유지해야 할 수 있습니다. Amazon RDS는 Microsoft SQL Server, Oracle, MySQL, PostgreSQL 등 지원하는 다양한 엔진 버전과 완벽하게 호환됩니다.

- 비프로덕션 환경에서 실험 및 벤치마킹을 수행하여 워크로드 요구 사항을 가장 잘 해결할 수 있는 데이터 스토어를 파악합니다.

리소스

관련 문서:

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

- AWS 기반 클라우드 스토리지
- Amazon EBS I/O Characteristics
- AWS 클라우드 데이터베이스
- AWS Database Caching
- DynamoDB Accelerator
- Amazon Aurora 모범 사례
- Amazon Redshift 성능
- Amazon Athena top 10 performance tips
- Amazon Redshift Spectrum best practices
- Amazon DynamoDB 모범 사례
- Choose between Amazon EC2 and Amazon RDS
- Best Practices for Implementing Amazon ElastiCache

관련 비디오:

- AWS re:Invent 2023: Improve Amazon Elastic Block Store efficiency and be more cost-efficient
- AWS re:Invent 2023: Optimizing storage price and performance with Amazon Simple Storage Service
- AWS re:Invent 2023: Building and optimizing a data lake on Amazon Simple Storage Service
- AWS re:Invent 2022: Building modern data architectures on AWS
- AWS re:Invent 2022: Building data mesh architectures on AWS
- AWS re:Invent 2023: Deep dive into Amazon Aurora and its innovations
- AWS re:Invent 2023: Advanced data modeling with Amazon DynamoDB
- AWS re:Invent 2022: Modernize apps with purpose-built databases
- Amazon DynamoDB deep dive: Advanced design patterns

관련 예제:

- AWS Purpose Built Databases 워크숍
- Databases for Developers
- AWS Modern Data Architecture Immersion Day

- Build a Data Mesh on AWS
- Amazon S3 예제
- Optimize Data Pattern using Amazon Redshift Data Sharing
- Database Migrations
- MS SQL Server - AWS Database Migration Service (AWS DMS) Replication Demo
- Database Modernization Hands On 워크숍
- Amazon Neptune Samples

PERF03-BP02 데이터 스토어에 사용 가능한 구성 옵션 평가

데이터 스토어에 사용할 수 있는 다양한 기능과 구성 옵션을 이해하고 평가하여 워크로드에 맞는 스토리지 공간과 성능을 최적화하세요.

일반적인 안티 패턴:

- 모든 워크로드에 Amazon EBS와 같은 하나의 스토리지 유형만 사용합니다.
- 모든 스토리지 계층을 기준으로 한 실제 테스트 없이 워크로드 전체에 프로비저닝된 IOPS를 사용합니다.
- 선택한 데이터 관리 솔루션의 구성 옵션을 알지 못합니다.
- 다른 사용 가능한 구성 옵션을 고려하지 않고 인스턴스 크기만 늘립니다.
- 데이터 스토어의 규모 조정 특성을 테스트하고 있지 않습니다.

이 모범 사례 확립의 이점: 데이터 스토어 구성을 탐색하고 실험하여 인프라 비용을 절감하고, 성능을 개선하며, 워크로드를 유지하는 데 들여야 하는 수고를 줄일 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

워크로드에는 데이터 스토리지 및 액세스 요구 사항에 따라 하나 이상의 데이터 스토어가 사용될 수 있습니다. 성능 효율성과 비용을 최적화하려면 데이터 액세스 패턴을 평가하여 적절한 데이터 스토어 구성을 결정해야 합니다. 데이터 스토어 옵션을 탐색하는 동안 스토리지 옵션, 메모리, 컴퓨팅, 읽기 복제본, 일관성 요구 사항, 연결 풀링 및 캐싱 옵션과 같은 다양한 측면을 고려하세요. 이러한 다양한 구성 옵션을 실험하여 성능 효율성 지표를 개선해 보세요.

구현 단계

- 데이터 스토어의 현재 구성(인스턴스 유형, 스토리지 크기, 데이터베이스 엔진 버전 등)을 파악합니다.
- AWS 설명서 및 모범 사례를 검토하여 데이터 스토어의 성능을 개선하는 데 도움이 되는 권장 구성 옵션에 대해 알아보세요. 다음과 같은 주요 데이터 스토어 옵션을 고려해야 합니다.

구성 옵션	예시
읽기 오프로드(예: 읽기 전용 복제본 및 캐싱)	• DynamoDB 테이블의 경우 캐싱을 위해 DAX를 사용하여 읽기를 오프로드할 수 있습니다. • Amazon ElastiCache(Redis OSS) 클러스터를 생성하고 요청된 항목이 없는 경우 데이터베이스로 되돌아가서 캐시에서 먼저 읽도록 애플리케이션을 구성할 수 있습니다. • Amazon RDS 및 Aurora와 같은 관계형 데이터베이스와 Neptune 및 Amazon DocumentDB와 같은 프로비저닝된 NoSQL 데이터베이스는 모두 읽기 전용 복제본을 추가하여 워크로드의 읽기 부분을 오프로드할 수 있도록 지원합니다. • DynamoDB와 같은 서버리스 데이터베이스는 자동으로 규모가 조정됩니다. 워크로드를 처리하기에 충분한 읽기 용량 단위(RCU)가 프로비저닝되었는지 확인합니다.
쓰기 규모 조정(예: 파티션 키 샤딩 또는 대기열 도입)	• 관계형 데이터베이스의 경우 인스턴스 크기를 늘려 확장된 워크로드를 수용하거나 프로비저닝된 IOPS를 늘려 기본 스토리지에 대한 처리량을 증대할 수 있습니다. • 데이터베이스에 직접 쓰는 대신 데이터베이스 앞에 대기열을 적용할 수도 있습니다. 이 패턴을 사용하면 데이터베이스에서 수집 정보를 분리하고 흐름 속도를 제어하여 데이터베이스가 과부하되지 않도록 할 수 있습니다.

구성 옵션	예시
	• 단기간 트랜잭션을 많이 만들지 않고 쓰기 요청을 일괄 처리하면 쓰기 볼륨이 많은 관계형 데이터베이스의 처리량을 향상시킬 수 있습니다. • DynamoDB와 같은 서버리스 데이터베이스는 자동으로 쓰기 처리량을 확장하거나 용량 모드에 따라 프로비저닝된 쓰기 용량 단위(WCU)를 조정하여 확장할 수 있습니다. • 특정 파티션 키에 대한 처리량 제한에 도달하면 핫 파티션과 관련된 문제가 발생할 수 있습니다. 이 문제는 보다 고르게 분산된 파티션 키를 선택하거나 파티션 키를 쓰기 샤딩하여 해결할 수 있습니다.
데이터 세트의 수명 주기 관리 정책	• Amazon S3 수명 주기를 사용하여 전체 수명 주기 동안 객체를 관리할 수 있습니다. 액세스 패턴을 알 수 없거나 패턴이 변화하거나 예측할 수 없는 경우 Amazon S3 Intelligent-Tiering을 사용할 수 있으며, 이를 통해 액세스 패턴을 모니터링하고, 액세스하지 않은 객체를 더 저렴한 액세스 계층으로 자동으로 이동할 수 있습니다. Amazon S3 Storage Lens 지표를 활용하여 수명 주기 관리에서 최적화 기회와 격차를 식별할 수 있습니다. • Amazon EFS 수명 주기 관리는 파일 시스템에 대한 파일 스토리지를 자동으로 관리합니다.

구성 옵션	예시
연결 관리 및 풀링	• Amazon RDS 프록시는 Amazon RDS 및 Aurora와 함께 사용하여 데이터베이스에 대한 연결을 관리할 수 있습니다. • DynamoDB와 같은 서버리스 데이터베이스에는 연계된 연결이 없으니, 프로비저닝된 용량 및 자동 규모 조정 정책을 고려하여 로드 급증을 처리합니다.

- 비프로덕션 환경에서 실험 및 벤치마킹을 수행하여 워크로드 요구 사항을 해결할 수 있는 구성 옵션을 파악합니다.
- 실험을 마친 후에는 마이그레이션을 계획하고 성과 지표를 검증합니다.
- AWS 모니터링(예: Amazon CloudWatch) 및 최적화(예: Amazon S3 Storage Lens) 도구를 활용하고 실제 사용 패턴에 기반하여 데이터 스토어를 지속적으로 최적화합니다.

리소스

관련 문서:

- AWS 기반 클라우드 스토리지
- Amazon EBS 볼륨 유형
- Amazon EC2 스토리지
- Amazon EFS: Amazon EFS Performance
- Amazon FSx for Lustre Performance
- Amazon FSx for Windows File Server Performance
- Amazon S3 Glacier: S3 Glacier 설명서
- Amazon S3: 요청 속도 및 성능 고려 사항
- Amazon EBS I/O Characteristics
- AWS 클라우드 데이터베이스
- AWS Database Caching
- DynamoDB Accelerator
- Amazon Aurora 모범 사례
- Amazon Redshift 성능

- Amazon Athena top 10 performance tips
- Amazon Redshift Spectrum best practices
- Amazon DynamoDB 모범 사례

관련 비디오:

- AWS re:Invent 2023: Improve Amazon Elastic Block Store efficiency and be more cost-efficient
- AWS re:Invent 2023: Optimize storage price and performance with Amazon Simple Storage Service
- AWS re:Invent 2023: Building and optimizing a data lake on Amazon Simple Storage Service
- AWS re:Invent 2023: What's new with AWS file storage
- AWS re:Invent 2023: Dive deep into Amazon DynamoDB

관련 예제:

- AWS Purpose Built Databases 워크숍
- Databases for Developers
- AWS Modern Data Architecture Immersion Day
- Amazon EBS Autoscale
- Amazon S3 예제
- Amazon DynamoDB Examples
- AWS Database migration samples
- Database Modernization 워크숍
- Working with parameters on your Amazon RDS for Postgress DB

PERF03-BP03 데이터 스토어 성과 지표 수집 및 기록

데이터 스토어에 대한 관련 성과 지표를 추적하고 기록하여 데이터 관리 솔루션의 성능을 파악할 수 있습니다. 이러한 지표는 데이터 스토어를 최적화하고, 워크로드 요구 사항이 충족되는지 확인하며, 워크로드 성능에 대한 명확한 개요를 제공하는 데 도움이 될 수 있습니다.

일반적인 안티 패턴:

- 지표에 대해 수동 로그 파일 검색만 사용합니다.

- 팀에서 사용하는 내부 도구에만 지표를 게시하고 워크로드를 종합적으로 바라보고 있지 않습니다.
- 선택한 모니터링 소프트웨어에서 기록한 기본 지표만 사용합니다.
- 문제가 발생한 경우에만 지표를 검토합니다.
- 시스템 수준 지표만 모니터링하며 데이터 액세스나 사용량 지표는 캡처하지 않습니다.

이 모범 사례 확립의 이점: 성능 기준선을 설정하면 워크로드의 정상적인 동작과 요구 사항을 이해하는 데 도움이 됩니다. 비정상적인 패턴을 더 빨리 식별하고 디버깅할 수 있어 데이터 스토어의 성능과 신뢰성이 향상됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

데이터 스토어의 성능을 모니터링하려면 일정 기간에 걸쳐 여러 성과 지표를 기록해야 합니다. 이를 통해 이상 징후를 탐지할 수 있을뿐더러 비즈니스 지표를 기준으로 성능을 측정하여 워크로드 요구 사항을 충족하는지 확인할 수 있습니다.

지표에는 데이터 스토어를 지원하는 기본 시스템과 데이터베이스 지표가 모두 포함되어야 합니다. 기본 시스템 지표에는 CPU 사용률, 메모리, 사용 가능한 디스크 스토리지, 디스크 I/O, 캐시 적중률, 네트워크 인바운드 및 아웃바운드 지표가 포함될 수 있습니다. 데이터베이스 지표는 초당 트랜잭션, 상위 쿼리, 평균 쿼리 속도, 응답 시간, 인덱스 사용량, 테이블 잠금, 쿼리 시간 초과 및 열린 연결 수로 구성 가능합니다. 이 데이터는 워크로드의 성능과 데이터 관리 솔루션의 사용 방법을 이해하는 데 중요합니다. 이러한 지표를 데이터 중심 전략에 포함시켜 워크로드의 리소스를 조정하고 최적화할 수 있습니다.

데이터베이스 성능과 관련된 성능 측정값을 기록하는 도구, 라이브러리 및 시스템을 사용합니다.

구현 단계

- 추적할 데이터 스토어의 주요 성과 지표를 식별하세요.
  - Amazon S3 지표 및 차원
  - Amazon RDS 인스턴스에서 지표 모니터링
  - Amazon RDS의 성능 개선 도우미로 DB 로드 모니터링
  - Enhanced Monitoring 개요
  - DynamoDB 지표 및 차원
  - Monitoring DynamoDB Accelerator

  - Monitoring Amazon MemoryDB with Amazon CloudWatch
  - 어떤 메트릭을 모니터링해야 합니까?
  - Amazon Redshift 클러스터 성능 모니터링
  - Timestream metrics and dimensions
  - Amazon Aurora에 대한 Amazon CloudWatch 지표
  - Logging and monitoring in Amazon Keyspaces (for Apache Cassandra)
  - Monitoring Amazon Neptune Resources
- 승인된 로깅 및 모니터링 솔루션을 사용하여 이러한 지표를 수집합니다. Amazon CloudWatch는 아키텍처의 리소스 전반에서 지표를 수집할 수 있습니다. 또한 사용자 지정 지표를 수집하고 게시하여 비즈니스 또는 파생 지표를 파악할 수도 있습니다. CloudWatch 또는 서드파티 솔루션을 사용하여 임곗값 위반 시점을 나타내는 경보를 설정합니다.
- 데이터 스토어 모니터링에 성능 이상을 탐지하는 기계 학습 솔루션의 이점을 활용할 수 있는지 확인합니다.
  - Amazon RDS에 대한 Amazon DevOps Guru에서는 성능 문제에 대한 가시성을 제공하고 권장 수정 조치를 제안합니다.
- 보안 및 운영 목표에 맞게 모니터링 및 로깅 솔루션에서 데이터 보존을 구성합니다.
  - CloudWatch 지표에 대한 기본 데이터 보존
  - CloudWatch Logs에 대한 기본 데이터 보존

리소스

관련 문서:

- AWS Database Caching
- Amazon Athena top 10 performance tips
- Amazon Aurora 모범 사례
- DynamoDB Accelerator
- Amazon DynamoDB 모범 사례
- Amazon Redshift Spectrum best practices
- Amazon Redshift 성능
- AWS 클라우드 데이터베이스
- Amazon RDS 성능 개선 도우미

관련 비디오:

- AWS re:Invent 2022 - Performance monitoring with Amazon RDS and Aurora, featuring Autodesk
- Database Performance Monitoring and Tuning with Amazon DevOps Guru for Amazon RDS
- AWS re:Invent 2023 - What's new with AWS file storage
- AWS re:Invent 2023 - Dive deep into Amazon DynamoDB
- AWS re:Invent 2023 - Building and optimizing a data lake on Amazon S3
- AWS re:Invent 2023 - What's new with AWS file storage
- AWS re:Invent 2023 - Dive deep into Amazon DynamoDB
- Best Practices for Monitoring Redis Workloads on Amazon ElastiCache

관련 예제:

- AWS Dataset Ingestion Metrics Collection Framework
- Amazon RDS Monitoring 워크숍
- AWS Purpose Built Databases 워크숍

PERF03-BP04 데이터 스토어에서 쿼리 성능을 개선하기 위한 전략 구현

데이터를 최적화하고 데이터 쿼리를 개선하는 전략을 구현하여 워크로드에 대한 확장성과 효율적인 성능을 향상할 수 있습니다.

일반적인 안티 패턴:

- 데이터 스토어에서는 데이터를 파티션하지 않습니다.
- 데이터 스토어에 하나의 파일 형식으로만 데이터를 저장합니다.
- 데이터 스토어에서 인덱스를 사용하지 않습니다.

이 모범 사례 확립의 이점: 데이터 및 쿼리 성능을 최적화하면 효율성이 향상되고 비용이 절감되며 사용자 경험이 향상됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

데이터 최적화 및 쿼리 튜닝은 전체 클라우드 워크로드의 성능과 응답성에 영향을 미치기 때문에 데이터 스토어의 성능 효율성에 있어 매우 중요한 측면입니다. 쿼리가 최적화되지 않으면 리소스 사용량과 병목 현상이 증가하여 데이터 스토어의 전반적인 효율성이 저하될 수 있습니다.

데이터 최적화는 효율적인 데이터 스토리지와 액세스를 위한 여러 기술을 포함합니다. 또한 데이터 저장소의 쿼리 성능을 개선하는 데도 도움이 됩니다. 주요 전략에는 데이터 파티셔닝, 데이터 압축, 데이터 비정규화가 포함되며, 이는 스토리지와 액세스 모두에 대해 데이터를 최적화하는 데 도움이 됩니다.

구현 단계

- 데이터 스토어에서 수행되는 중요한 데이터 쿼리를 이해하고 분석합니다.
- 데이터 스토어에서 느린 쿼리를 식별하고 쿼리 계획을 사용하여 현재 상태를 파악합니다.
  - Amazon Redshift에서 쿼리 계획 분석
  - Athena에서 EXPLAIN 및 EXPLAIN ANALYZE 사용
- 쿼리 성능을 개선하기 위한 전략을 구현합니다. 몇 가지 주요 전략은 다음과 같습니다.
  - 열 파일 형식(예: Parquet 또는 ORC)을 사용합니다.
  - 스토리지 공간 및 I/O 작업을 줄이기 위해 데이터 스토어의 데이터 압축.
  - 데이터를 더 작은 부분으로 분할하고 데이터 스캔 시간을 줄이기 위한 데이터 파티셔닝.
    - Athena에서 데이터 분할
    - 파티션 및 데이터 배포
  - 쿼리의 공통 열에서 데이터 인덱싱.
  - 빈번하게 사용되는 쿼리에는 구체화된 뷰를 사용하세요.
    - Understanding materialized views
    - Amazon Redshift에서 구체화된 뷰 생성
  - 쿼리에 적합한 조인 작업을 선택합니다. 두 테이블을 조인하는 경우 조인 왼쪽에 큰 테이블을 지정하고 조인 오른쪽에 작은 테이블을 지정합니다.
  - 지연 시간을 개선하고 데이터베이스 I/O 작업 횟수를 줄이기 위한 분산 캐싱 솔루션.
  - 정리, 리인덱싱, 통계 실행과 같은 정기적 유지 관리.
- 비운영 환경에서의 실험 및 테스트 전략.

리소스

관련 문서:

- Amazon Aurora 모범 사례
- Amazon Redshift 성능
- Amazon Athena top 10 performance tips
- AWS Database Caching
- Best Practices for Implementing Amazon ElastiCache
- Athena에서 데이터 분할

관련 비디오:

- AWS re:Invent 2023 - AWS storage cost-optimization best practices
- AWS re:Invent 2022 - Performance monitoring with Amazon RDS and Aurora, featuring Autodesk
- Optimize Amazon Athena Queries with New Query Analysis Tools

관련 예제:

- AWS Purpose Built Databases 워크숍

PERF03-BP05 캐싱을 활용하는 데이터 액세스 패턴 구현

자주 액세스하는 데이터를 빠르게 검색하기 위해 데이터를 캐싱하여 이점을 얻을 수 있는 액세스 패턴을 구현하세요.

일반적인 안티 패턴:

- 자주 변경되는 데이터를 캐시합니다.
- 캐시된 데이터가 마치 영구적으로 저장되고 항상 사용 가능한 것처럼 의존합니다.
- 캐시된 데이터의 일관성은 고려하지 않습니다.
- 캐싱 구현의 효율성을 모니터링하지 않습니다.

이 모범 사례 확립의 이점: 캐시에 데이터를 저장하면 읽기 지연 시간, 읽기 처리량, 사용자 경험 및 전반적인 효율성을 개선하고 비용을 절감할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

캐시는 동일한 데이터에 대한 향후 요청을 더 빠르고 효율적으로 처리할 수 있도록 데이터를 저장하는 것을 목적으로 하는 소프트웨어 또는 하드웨어 구성 요소입니다. 캐시에 저장된 데이터는 이전 계산을 반복하거나 다른 데이터 저장소에서 가져와서 손실된 경우 재구성할 수 있습니다.

데이터 캐싱은 전체 애플리케이션 성능을 개선하고 기본 데이터 소스의 부담을 줄이는 가장 효과적인 전략 중 하나가 될 수 있습니다. 데이터를 원격 직접 호출을 수행하는 애플리케이션 내에서(클라이언트 측 캐싱이라고 함) 또는 빠른 보조 서비스를 사용하여 데이터를 저장하는 등(원격 캐싱이라고 함) 애플리케이션의 여러 수준에서 데이터를 캐싱할 수 있습니다.

클라이언트 측 캐싱

클라이언트측 캐싱을 사용하면 각 클라이언트(백엔드 데이터 스토어를 쿼리하는 애플리케이션 또는 서비스)가 고유한 쿼리 결과를 지정된 시간 동안 로컬에 저장할 수 있습니다. 이렇게 하면 로컬 클라이언트 캐시를 먼저 확인하여 네트워크를 통해 데이터 스토어에 대한 요청 수를 줄일 수 있습니다. 결과가 없는 경우 애플리케이션은 데이터 스토어를 쿼리하고 해당 결과를 로컬에 저장할 수 있습니다. 이 패턴을 사용하면 각 클라이언트가 가능한 가장 가까운 위치(클라이언트 자체)에 데이터를 저장할 수 있으므로 지연 시간을 최소화할 수 있습니다. 또한 클라이언트는 백엔드 데이터 스토어를 사용할 수 없는 경우에도 일부 쿼리를 계속 제공하여 전체 시스템의 가용성을 높일 수 있습니다.

이 접근 방식의 한 가지 단점은 여러 클라이언트가 관련된 경우 동일한 캐시된 데이터를 로컬에 저장할 수 있다는 것입니다. 이로 인해 클라이언트 간에 중복 스토리지 사용과 데이터 불일치가 모두 발생합니다. 한 클라이언트가 쿼리 결과를 캐시하고 1분 후에 다른 클라이언트가 동일한 쿼리를 실행하여 다른 결과를 얻을 수 있습니다.

원격 캐싱

클라이언트 간 데이터 중복 문제를 해결하기 위해 빠른 외부 서비스 또는 원격 캐시를 사용하여 쿼리된 데이터를 저장할 수 있습니다. 각 클라이언트는 로컬 데이터스토어를 확인하는 대신 백엔드 데이터스토어를 쿼리하기 전에 원격 캐시를 확인합니다. 이 전략을 사용하면 클라이언트와 독립적으로 스토리지 공간이 확장되므로 클라이언트 간 응답 일관성이 향상되고 저장된 데이터의 효율성이 향상되며 캐시된 데이터의 볼륨이 커집니다.

원격 캐시의 단점은 원격 캐시를 확인하기 위해 추가 네트워크 홉이 필요하기 때문에 전체 시스템의 지연 시간이 더 길어질 수 있다는 것입니다. 클라이언트 측 캐싱은 다중 레벨 캐싱을 위한 원격 캐싱과 함께 사용하여 지연 시간을 개선할 수 있습니다.

구현 단계

- 캐싱의 이점을 누릴 수 있는 데이터베이스, API 및 네트워크 서비스를 식별합니다. 읽기 워크로드가 많거나 쓰기 대 읽기 비율이 높거나 규모 조정 비용이 많이 드는 서비스는 캐싱의 대상입니다.
  - 데이터베이스 캐싱
  - 응답성 향상을 위한 API 캐싱 활성화
- 액세스 패턴에 가장 적합한 유형의 캐싱 전략을 식별하세요.
  - 캐싱 전략
  - AWS Caching Solutions
- 데이터 스토어에 대한 Caching Best Practices를 따릅니다.
- 데이터의 최신 상태를 유지하고 백엔드 데이터스토어에 대한 부담을 줄이는 모든 데이터에 대해 Time-To-Live(TTL)와 같은 캐시 무효화 전략을 구성하세요.
- 자동 연결 재시도, 지수 백오프, 클라이언트 측 시간 제한, 연결 풀링과 같은 기능을 클라이언트에서 활성화하여 성능과 신뢰성을 개선할 수 있습니다.
  - Best practices: Redis clients and Amazon ElastiCache (Redis OSS)
- 모니터 캐시 적중률을 80% 이상 목표로 합니다. 값이 낮으면 캐시 크기가 충분하지 않거나 액세스 패턴이 캐싱의 이점을 얻지 못한다는 의미일 수 있습니다.
  - Which metrics should I monitor?
  - Best practices for monitoring Redis workloads on Amazon ElastiCache
  - Monitoring best practices with Amazon ElastiCache (Redis OSS) using Amazon CloudWatch
- 데이터 복제를 구현하여 여러 인스턴스로 읽기를 오프로드하고 데이터 읽기 성능과 가용성을 개선합니다.

리소스

관련 문서:

- Using the Amazon ElastiCache Well-Architected Lens
- Monitoring best practices with Amazon ElastiCache (Redis OSS) using Amazon CloudWatch
- 어떤 메트릭을 모니터링해야 합니까?
- Performance at Scale with Amazon ElastiCache 백서
- 캐싱 관련 당면 과제 및 전략

관련 비디오:

- Amazon ElastiCache Learning Path
- Design for success with Amazon ElastiCache best practices
- AWS re:Invent 2020 - Design for success with Amazon ElastiCache best practices
- AWS re:Invent 2023 - [LAUNCH] Introducing Amazon ElastiCache Serverless
- AWS re:Invent 2022 - 5 great ways to reimagine your data layer with Redis
- AWS re:Invent 2021 - Deep dive on Amazon ElastiCache (Redis OSS)

관련 예제:

- Boosting MySQL database performance with Amazon ElastiCache (Redis OSS)

네트워킹 및 콘텐츠 전송

워크로드에 대한 최적의 네트워킹 솔루션은 지연 시간, 처리량 요구 사항, 지터, 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.

AWS에서 네트워킹은 가상화되고 다양한 유형 및 구성으로 제공됩니다. 따라서 네트워킹 요구 사항에 보다 쉽게 부합할 수 있습니다. AWS에서는 제품 기능(예: 강화된 네트워킹, Amazon EC2 네트워킹 최적화 인스턴스, Amazon S3 Transfer Acceleration, 동적 Amazon CloudFront)을 제공하여 네트워크 트래픽을 최적화합니다. 또한 AWS Global Accelerator에서는 네트워킹 기능(예: Amazon Route 53 지연 속도 기반 라우팅, Amazon VPC 엔드포인트, AWS, AWS Direct Connect)도 제공하여 네트워크 거리 또는 지터를 줄입니다.

이 중점 영역에서는 클라우드에서 효율적인 네트워킹 및 콘텐츠 전송 솔루션을 설계, 구성 및 운영하기 위한 지침과 모범 사례를 공유합니다.

모범 사례

- PERF04-BP01 네트워킹이 성능에 미치는 영향 파악
- PERF04-BP02 사용 가능한 네트워킹 기능 평가
- PERF04-BP03 워크로드에 적합한 전용 연결 또는 VPN 선택
- PERF04-BP04 로드 밸런싱을 사용하여 여러 리소스에 트래픽 분산
- PERF04-BP05 성능을 개선할 수 있는 네트워크 프로토콜 선택
- PERF04-BP06 네트워크 요구 사항에 따라 워크로드의 위치 선택
- PERF04-BP07 지표를 기준으로 네트워크 구성 최적화

PERF04-BP01 네트워킹이 성능에 미치는 영향 파악

네트워크 관련 의사 결정이 워크로드에 미치는 영향을 분석하고 이해하여 효율적인 성능과 향상된 사용자 경험을 제공합니다.

일반적인 안티 패턴:

- 모든 트래픽이 기존 데이터 센터를 통과합니다.
- 클라우드 네이티브 네트워크 보안 도구를 사용하는 대신 중앙 방화벽을 통해 모든 트래픽을 라우팅합니다.
- 실제 사용 요구 사항을 이해하지 않고 AWS Direct Connect 연결을 프로비저닝합니다.

- 네트워킹 솔루션을 정의할 때 워크로드 특성과 암호화 오버헤드를 고려하지 않습니다.
- 클라우드의 네트워킹 솔루션에 온프레미스 개념과 전략을 적용합니다.

이 모범 사례 확립의 이점: 네트워킹이 워크로드 성능에 미치는 영향을 이해하면 잠재적인 병목 현상을 식별하고, 사용자 경험을 개선하며, 신뢰성을 높이고, 워크로드 변화에 따라 운영 유지 관리 작업을 줄이는 데 도움이 됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

네트워크는 애플리케이션 구성 요소, 클라우드 서비스, 엣지 네트워크 및 온프레미스 데이터 간의 연결을 담당하므로, 워크로드 성능에 큰 영향을 미칠 수 있습니다. 사용자 경험은 워크로드 성능 외에 네트워크 지연 시간, 대역폭, 프로토콜, 위치, 네트워크 정체, 지터, 처리량 및 라우팅 규칙에도 영향을 받습니다.

지연 시간, 패킷 크기, 라우팅 규칙, 프로토콜 및 지원 트래픽 패턴을 포함한 워크로드의 네트워킹 요구 사항 목록을 문서화합니다. 사용 가능한 네트워킹 솔루션을 검토하고 워크로드 네트워킹 특성을 충족하는 서비스를 파악합니다. 클라우드 기반 네트워크는 빠르게 재구축될 수 있으므로 성능 효율성을 개선하려면 네트워크 아키텍처를 지속적으로 변경해야 합니다.

구현 단계:

- 네트워크 지연 시간, 대역폭, 프로토콜, 위치, 트래픽 패턴(급증 및 빈도), 처리량, 암호화, 검사 및 라우팅 규칙과 같은 지표를 포함한 네트워킹 성능 요구 사항을 정의하고 문서화하세요.
- VPC, AWS Direct Connect, Elastic Load Balancing(ELB), Amazon Route 53과 같은 주요 AWS 네트워킹 서비스에 대해 알아봅니다.
- 다음과 같은 주요 네트워킹 특성을 파악하세요.

특성	도구 및 지표
기본적인 네트워킹 특성	• VPC 흐름 로그 • AWS Transit Gateway 흐름 로그 • AWS Transit Gateway 지표 • AWS PrivateLink 지표
애플리케이션 네트워킹 특성	• Elastic Fabric Adapter(EFA)

특성	도구 및 지표
	• AWS App Mesh 지표 • Amazon API Gateway 지표
엣지 네트워킹 특성	• Amazon CloudFront 지표 • Amazon Route 53 지표 • AWS Global Accelerator 지표
하이브리드 네트워킹 특성	• AWS Direct Connect 지표 • AWS Site-to-Site VPN 지표 • AWS Client VPN 지표 • AWS 클라우드 WAN 지표
보안 네트워킹 특성	• AWS Shield, AWS WAF, AWS Network Firewall 지표
추적 특성	• AWS X-Ray • VPC Reachability Analyzer • Network Access Analyzer • Amazon Inspector • Amazon CloudWatch RUM

- 네트워크 성능을 벤치마킹하고 테스트합니다.
  - 네트워크 처리량을 벤치마킹합니다(인스턴스가 동일한 VPC에 있는 경우 일부 요인이 Amazon EC2 네트워크 성능에 영향을 미칠 수 있음). 동일한 VPC에 있는 Amazon EC2 Linux 인스턴스 간의 네트워크 대역폭을 측정합니다.
  - 로드 테스트를 수행하여 네트워킹 솔루션과 옵션을 실험합니다.

리소스

관련 문서:

- Application Load Balancer
- Linux 기반 EC2의 향상된 네트워킹
- Windows 기반 EC2의 향상된 네트워킹

- EC2 배치 그룹
- Linux 인스턴스에서 Elastic Network Adapter(ENA)로 향상된 네트워킹 활성화
- Network Load Balancer
- AWS의 네트워킹 제품
- Transit Gateway
- Transitioning to latency-based routing in Amazon Route 53
- VPC 엔드포인트

관련 비디오:

- AWS re:Invent 2023 - AWS networking foundations
- AWS re:Invent 2023 - What can networking do for your application?
- AWS re:Invent 2023 - Advanced VPC designs and new capabilities
- AWS re:Invent 2023 - A developer's guide to cloud networking
- AWS re:Invent 2019 - Connectivity to AWS and hybrid AWS network architectures
- AWS re:Invent 2019 - Optimizing Network Performance for Amazon EC2 Instances
- AWS Summit Online - Improve Global Network Performance for Applications
- AWS re:Invent 2020 - Networking best practices and tips with the Well-Architected Framework
- AWS re:Invent 2020 - AWS networking best practices in large-scale migrations

관련 예제:

- AWS Transit Gateway and Scalable Security Solutions
- AWS Networking 워크숍
- Hands-on Network Firewall 워크숍
- Observing and Diagnosing your Network on AWS
- Finding and addressing Network Misconfigurations on AWS

PERF04-BP02 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.

일반적인 안티 패턴:

- 본사가 물리적으로 위치한 한 리전 내에 머무릅니다.
- 트래픽 필터링에 보안 그룹 대신 방화벽을 사용합니다.
- 보안 그룹, 엔드포인트 정책 및 기타 클라우드 네이티브 기능에 의존하지 않고 트래픽 검사를 위해 TLS를 중단합니다.
- 보안 그룹 대신 서브넷 기반 조각화만 사용합니다.

이 모범 사례 확립의 이점: 모든 서비스 기능 및 옵션을 평가하면 워크로드 성능을 개선하고 인프라 비용을 줄이며 워크로드를 유지 관리하는 데 필요한 작업을 줄이며 전반적인 보안 상태를 개선할 수 있습니다. 전 세계의 AWS 백본을 사용하여 고객에게 최상의 네트워킹 환경을 제공할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

AWS에서는 네트워크 성능을 개선하는 데 도움이 되는 AWS Global Accelerator 및 Amazon CloudFront와 같은 서비스를 제공하지만, 대부분의 AWS 서비스에는 네트워크 트래픽을 최적화하는 제품 기능(예: Amazon S3 Transfer Acceleration 기능)이 있습니다.

사용 가능한 네트워크 관련 구성 옵션과 이러한 옵션이 워크로드에 미치는 영향을 검토합니다. 성능 최적화는 이러한 옵션이 아키텍처와 상호 작용하는 방식과 측정된 성능 및 사용자 경험 모두에 미치는 영향을 이해하는 데 달려 있습니다.

구현 단계

- 워크로드 구성 요소 목록을 만듭니다.
  - 통합 글로벌 네트워크를 구축할 때 조직 네트워크를 구축, 관리 및 모니터링하를 데 AWS 클라우드 WAN 사용을 고려하세요.
  - Amazon CloudWatch Logs 지표로 글로벌 및 코어 네트워크를 모니터링합니다. 사용자의 디지털 경험을 식별하고, 이해하며, 개선하는 데 도움이 되는 인사이트를 제공하는 Amazon CloudWatch RUM을 활용하세요.
  - AWS Network Manager를 사용하여 애플리케이션 성능이 기본 AWS 네트워크의 성능과 어떤 관계가 있는지 파악할 수 있도록 AWS 리전 및 가용 영역 간 그리고 각 가용 영역 내부의 총 네트워크 지연 시간을 확인합니다.
  - 기존 구성 관리 데이터베이스(CMDB) 도구를 사용하거나 AWS Config 등과 같은 도구를 사용하여 워크로드 인벤토리 또는 구성 방식을 생성합니다.

- 기존 워크로드인 경우에는 성과 지표의 벤치마크를 식별하고 문서화하여 병목 현상과 개선해야 할 부분에 집중적으로 살펴봅니다. 성능 관련 네트워킹 지표는 비즈니스 요구 사항 및 워크로드 특성을 기준으로 워크로드에 따라 다릅니다. 무엇보다 대역폭, 지연 시간, 패킷 손실, 지터 및 재전송 등과 같은 지표가 워크로드 검토에 중요할 수 있습니다.
- 새 워크로드인 경우 로드 테스트를 수행하여 성능 병목 현상을 식별합니다.
- 파악한 성능 병목 현상에 대해 솔루션의 구성 옵션을 검토하여 성능 개선 기회를 파악합니다. 다음과 같은 주요 네트워킹 옵션 및 기능을 확인합니다.

개선 기회	Solution
네트워크 경로 또는 라우트	Network Access Analyzer를 사용하여 경로나 라우트를 식별합니다.
네트워크 프로토콜	PERF04-BP05 성능을 개선할 수 있는 네트워크 프로토콜 선택 섹션 참조
네트워크 토폴로지	여러 계정을 연결할 때 VPC 피어링 및 AWS Transit Gateway 사이에서 운영 및 성능과 관련된 장단점을 평가합니다. AWS Transit Gateway에서는 수천 개의 AWS 계정에 걸쳐 있는 모든 VPC를 온프레미스 네트워크로 상호 연결하는 방법을 단순화합니다. AWS Resource Access Manager를 사용하여 여러 계정 사이에서 AWS Transit Gateway를 공유합니다. PERF04-BP03 워크로드에 적합한 전용 연결 또는 VPN 선택 섹션 참조
네트워크 서비스	AWS Global Accelerator는 AWS 글로벌 네트워크 인프라를 사용하여 사용자 트래픽의 성능을 최대 60%까지 개선하는 네트워킹 서비스입니다. Amazon CloudFront는 전 세계적으로 워크로드 콘텐츠 전송 성능 및 지연 시간을 개선할 수 있습니다.

개선 기회	Solution
	Lambda@Edge를 사용하여 CloudFront가 사용자에게 더 가까이 제공하는 콘텐츠를 사용자 지정하고, 지연 시간을 줄이며, 성능을 개선하는 함수를 실행합니다. Amazon Route 53은 지연 시간 기반 라우팅, 지리적 위치 라우팅, 지리 근접 라우팅 및 IP 기반 라우팅 옵션을 제공하여 전 세계 고객을 대상으로 워크로드 성능을 개선할 수 있습니다. 워크로드가 전 세계에 분산되어 있는 경우, 워크로드 트래픽 및 사용자 위치를 검토하여 어떤 라우팅 옵션이 워크로드 성능을 최적화하는지 파악합니다.
스토리지 리소스 기능	Amazon S3 Transfer Acceleration은 외부 사용자가 Amazon S3로 데이터를 업로드할 때 CloudFront의 네트워킹 최적화 이점을 활용할 수 있는 기능입니다. 이렇게 하면 AWS 클라우드 전용 연결을 사용할 수 없는 원격 위치에서 대량의 데이터를 전송할 수 있습니다. Amazon S3 다중 리전 액세스 포인트는 콘텐츠를 여러 리전으로 복제하고 액세스 포인트 하나를 제공하여 워크로드를 간소화합니다. 다중 리전 액세스 포인트를 사용하는 경우 가장 낮은 지연 시간 버킷을 식별하는 서비스로 데이터를 요청하거나 Amazon S3에 데이터를 쓸 수 있습니다.

개선 기회	Solution
컴퓨팅 리소스 기능	Amazon EC2 인스턴스, 컨테이너 및 Lambda 함수에서 사용하는 탄력적 네트워크 인터페이스(ENA)는 흐름 기준으로 제한됩니다. 배치 그룹을 검토하여 EC2 네트워킹 처리량을 최적화합니다. 흐름 기준에서 병목 현상을 방지하려면 여러 흐름을 사용하도록 애플리케이션을 설계합니다. 컴퓨팅 관련 네트워크 지표를 모니터링하고 이러한 지표에 대한 가시성을 얻으려면 CloudWatch 지표 및 ethtool을 사용합니다. `ethtool` 명령은 ENA 드라이버에 포함되어 있으며 사용자 지정 지표로 CloudWatch에 게시할 수 있는 추가 네트워크 관련 지표를 노출할 수 있습니다. Amazon Elastic Network Adapter(ENA)는 클러스터 배치 그룹 내에서 인스턴스에 대해 더 나은 처리량을 제공하여 한층 더 최적화합니다. Elastic Fabric Adapter(EFA)는 AWS에서 높은 수준의 대규모 노드 간 통신이 필요한 워크로드를 실행할 때 사용할 수 있는 Amazon EC2 인스턴스용 네트워크 인터페이스입니다. Amazon EBS 최적화 인스턴스는 최적화된 구성 스택을 사용하고 Amazon EBS I/O를 늘리기 위해 전용 용량을 추가로 제공합니다.

리소스

관련 문서:

- Application Load Balancer
- Linux 기반 EC2의 향상된 네트워킹
- Windows 기반 EC2의 향상된 네트워킹
- EC2 배치 그룹

- Linux 인스턴스에서 Elastic Network Adapter(ENA)로 향상된 네트워킹 활성화
- Network Load Balancer
- AWS의 네트워킹 제품
- Transitioning to Latency-Based Routing in Amazon Route 53
- VPC 엔드포인트
- VPC 흐름 로그

관련 비디오:

- AWS re:Invent 2023 – Ready for what's next? Designing networks for growth and flexibility
- AWS re:Invent 2023 – Advanced VPC designs and new capabilities
- AWS re:Invent 2023 – A developer's guide to cloud networking
- AWS re:Invent 2022 – Dive deep on AWS networking infrastructure
- AWS re:Invent 2019 – Connectivity to AWS and hybrid AWS network architectures
- AWS re:Invent 2018 – Optimizing Network Performance for Amazon EC2 Instances
- AWS Global Accelerator

관련 예제:

- AWS Transit Gateway and Scalable Security Solutions
- AWS Networking 워크숍
- Observing and diagnosing your network
- Finding and addressing network misconfigurations on AWS

PERF04-BP03 워크로드에 적합한 전용 연결 또는 VPN 선택

온프레미스 및 클라우드 리소스를 연결하는 데 하이브리드 연결이 필요한 경우 성능 요구 사항을 충족할 수 있는 충분한 대역폭을 프로비저닝해야 합니다. 하이브리드 워크로드에 대한 대역폭 및 지연 시간 요구 사항을 예측합니다. 이 수치는 크기 조정 요구 사항을 결정합니다.

일반적인 안티 패턴:

- 네트워크 암호화 요구 사항에 대해서만 VPN 솔루션을 평가합니다.
- 백업 또는 이중화된 연결 옵션은 평가하지 않습니다.

- 모든 워크로드 요구 사항(암호화, 프로토콜, 대역폭 및 트래픽 요구 사항)을 식별할 수는 없습니다.

이 모범 사례 확립의 이점: 적절한 연결 솔루션을 선택하고 구성하면 워크로드의 신뢰성이 향상되고 성능이 극대화됩니다. 워크로드 요구 사항을 파악하고, 미리 계획하며, 하이브리드 솔루션을 평가하여 비용이 많이 드는 물리적 네트워크 변경과 운영 오버헤드를 최소화하는 동시에 가치 실현 시간을 단축할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

대역폭 요구 사항을 기반으로 하이브리드 네트워킹 아키텍처를 개발합니다. AWS Direct Connect에서는 AWS와 온프레미스 네트워크를 비공개로 연결할 수 있습니다. 일관된 성능을 달성하면서 고대역폭 그리고 짧은 지연 시간이 필요할 때 적합합니다. VPN 연결은 인터넷을 통해 보안 연결을 설정합니다. 임시 연결만 필요한 경우, 비용이 중요한 경우 또는 AWS Direct Connect를 사용 시 복원력이 뛰어난 물리적 네트워크 연결이 설정되기를 기다리는 동안 비상용으로 사용됩니다.

대역폭 요구 사항이 높으면 여러 AWS Direct Connect 또는 VPN 서비스를 고려할 수 있습니다. 트래픽은 서비스 간에 부하를 분산할 수 있지만 지연 시간과 대역폭 차이 때문에 AWS Direct Connect와 VPN 간의 부하 분산을 권장하지 않습니다.

구현 단계

- 기존 애플리케이션의 대역폭 및 지연 시간 요구 사항을 추정합니다.
  - AWS로 이동하는 기존 앱의 경우 내부 네트워크 모니터링 시스템의 데이터를 활용합니다.
  - 모니터링 데이터가 없는 새로운 또는 기존 워크로드의 경우 제품 소유자에게 문의하여 적절한 성과 지표를 결정하고 우수한 사용자 경험을 제공합니다.
- 연결 옵션으로 전용 연결 또는 VPN을 선택합니다. 모든 워크로드 요구 사항(암호화, 대역폭 및 트래픽 요구 사항)을 기반으로 AWS Direct Connect 또는 AWS VPN(또는 둘 다)을 선택할 수 있습니다. 다음 다이어그램은 적절한 연결 유형을 선택하는 데 도움이 됩니다.
  - AWS Direct Connect에서는 전용 연결 또는 호스팅된 연결을 사용하여 50Mbps에서 최대 100Gbps까지 AWS 환경에 대한 전용 연결을 제공합니다. 이렇게 하면 지연 시간을 관리 및 제어하고 대역폭을 프로비저닝할 수 있으므로 워크로드를 다른 환경에 효율적으로 연결할 수 있습니다. AWS Direct Connect 파트너를 사용하면 여러 환경에 엔드 투 엔드로 연결할 수 있으며 일관된 성능을 갖춘 확장 네트워크를 제공할 수 있습니다. AWS는 기본 100Gbps, Link Aggregation Group(LAG) 또는 BGP Equal-Cost Multipath(ECMP)를 사용하여 확장된 Direct Connect 대역폭을 제공합니다.

  - AWS Site-to-Site VPN에서는 인터넷 프로토콜 보안(IPsec) 프로토콜을 지원하는 관리형 VPN 서비스를 제공합니다. VPN 연결이 생성되면 각 VPN 연결에는 고가용성을 위해 두 개의 터널이 포함됩니다.
- AWS 설명서에 따라 적절한 연결 옵션을 선택합니다.
  - AWS Direct Connect를 사용하기로 결정한 경우 연결에 적합한 대역폭을 선택합니다.
  - 여러 위치에서 AWS Site-to-Site VPN을 사용하여 AWS 리전에 연결하는 경우에는 가속화된 Site-to-Site VPN 연결을 사용해 네트워크 성능을 개선하세요.
  - 네트워크 설계가 AWS Direct Connect에서 IPSec VPN 연결로 구성된 경우 보안을 강화하고 세분화를 달성하기 위해 프라이빗 IP VPN 사용을 고려하세요. AWS Site-to-Site 프라이빗 IP VPN은 전송 가상 인터페이스(VIF) 위에 배포됩니다.
  - AWS Direct Connect SiteLink를 사용하면 AWS Direct Connect 위치 사이에서 AWS 리전을 우회하는 가장 빠른 경로로 데이터를 전송해 전 세계 데이터 센터 간에 지연 시간이 짧고 중복 연결을 생성합니다.
- 프로덕션에 배포하기 전에 연결 설정을 확인하세요. 보안 및 성능 테스트를 수행하여 대역폭, 신뢰성, 지연 시간 및 규정 준수 요구 사항을 충족하는지 확인하세요.
- 연결 성능 및 사용량을 정기적으로 모니터링하고 필요한 경우 최적화하세요.

결정론적 성능 흐름도

리소스

관련 문서:

- AWS의 네트워킹 제품
- AWS Transit Gateway
- VPC Endpoints
- 확장 가능하고 안전한 다중 VPC AWS 네트워크 인프라 구축
- Client VPN

관련 비디오:

- AWS re:Invent 2023 – Building hybrid network connectivity with AWS

- AWS re:Invent 2023 – Secure remote connectivity to AWS
- AWS re:Invent 2022 – Optimizing performance with Amazon CloudFront
- AWS re:Invent 2019 – Connectivity to AWS and hybrid AWS network architectures
- AWS re:Invent 2020 – AWS Transit Gateway Connect

관련 예제:

- AWS Transit Gateway and Scalable Security Solutions
- AWS Networking 워크숍

PERF04-BP04 로드 밸런싱을 사용하여 여러 리소스에 트래픽 분산

클라우드의 탄력성을 워크로드에 활용할 수 있도록 여러 리소스 또는 서비스에 트래픽을 분산합니다. 로드 밸런싱을 사용하여 암호화 종료를 오프로드하면 성능과 신뢰성을 개선하고 트래픽을 효율적으로 관리 및 라우팅할 수 있습니다.

일반적인 안티 패턴:

- 로드 밸런서 유형을 선택할 때 워크로드 요구 사항을 고려하지 않습니다.
- 성능 최적화 시 로드 밸런서 기능을 활용하지 않습니다.
- 워크로드는 로드 밸런서 없이 인터넷에 직접 노출됩니다.
- 기존 로드 밸런서를 통해 모든 인터넷 트래픽을 라우팅합니다.
- 일반 TCP 로드 밸런싱을 사용하고 각 컴퓨팅 노드에서 SSL 암호화를 처리하도록 합니다.

이 모범 사례 확립의 이점: 로드 밸런서는 단일 가용 영역 또는 여러 가용 영역에서 애플리케이션 트래픽의 다양한 부하를 처리하고 고가용성, 자동 규모 조정, 워크로드 활용도 향상을 지원합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

로드 밸런서는 워크로드의 진입점 역할을 하며 여기에서 트래픽을 컴퓨팅 인스턴스나 컨테이너와 같은 백엔드 대상으로 분산하여 사용률을 개선합니다.

아키텍처를 최적화하는 첫 번째 단계는 적절한 로드 밸런서 유형을 선택하는 것입니다. 먼저 프로토콜(예: TCP, HTTP, TLS 또는 WebSockets), 대상 유형(예: 인스턴스, 컨테이너 또는 서버리스), 애플

리케이션 요구 사항(장기간 실행되는 연결, 사용자 인증 또는 고정성 등) 및 배치(예: 리전, 로컬 영역, Outpost 또는 영역 격리)와 같은 워크로드 특성을 나열해 봅니다.

AWS는 애플리케이션이 로드 밸런싱을 사용할 수 있도록 여러 모델을 제공합니다. Application Load Balancer는 HTTP 및 HTTPS 트래픽을 로드 밸런싱하는 데 가장 적합하며, 마이크로서비스 및 컨테이너를 비롯한 최신 애플리케이션 아키텍처를 제공할 때 사용할 수 있는 고급 요청 라우팅 기능을 제공합니다.

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

적절한 로드 밸런서를 선택한 후 해당 기능을 활용하면 백엔드가 트래픽을 처리하는 데 필요한 노력을 줄일 수 있습니다.

예를 들어 Application Load Balancer(ALB) 및 Network Load Balancer(NLB)를 모두 사용하면 SSL/TLS 암호화 오프로딩을 수행할 수 있습니다. 이를 통해 대상에서 완료되는 CPU 집약적 TLS 핸드셰이크를 방지하고 인증서 관리를 개선할 수 있습니다.

로드 밸런서에서 SSL/TLS 오프로딩을 구성하면 백엔드에 암호화되지 않은 트래픽을 전달하고 백엔드 리소스를 확보하며 클라이언트에 대한 응답 시간을 개선하는 동시에 클라이언트에서 들어오고 나가는 트래픽의 암호화를 담당하게 됩니다.

Application Load Balancer도 대상에서 지원할 필요 없이 HTTP2 트래픽을 처리할 수 있습니다. HTTP2가 TCP 연결을 보다 효율적으로 사용하므로 이렇게 간단한 결정이 애플리케이션 응답 시간을 개선할 수 있습니다.

아키텍처를 정의할 때 워크로드 지연 시간 요구 사항도 고려해야 합니다. 예를 들어 지연 시간에 민감한 애플리케이션이 있는 경우 지연 시간이 매우 짧은 Network Load Balancer를 사용하기로 결정할 수 있습니다. 또는 AWS 로컬 영역 또는 AWS Outposts에서 Application Load Balancer를 활용하여 워크로드를 고객에게 더 가까이 가져오기로 결정할 수 있습니다.

지연 시간에 민감한 워크로드에 대한 또 다른 대응책은 크로스 영역 로드 밸런싱입니다. 크로스 영역 로드 밸런싱을 활용하면 각 로드 밸런서 노드를 사용하도록 설정된 모든 가용 영역의 등록된 대상에 트래픽을 분산합니다.

로드 밸런서와 통합된 Auto Scaling을 사용합니다. 성능 효율적인 시스템의 주요 측면 중 하나는 백엔드 리소스의 크기를 적절하게 조정하는 것과 관련이 있습니다. 이를 위해서는 백엔드 대상 리소스에 대한 로드 밸런서 통합을 활용할 수 있습니다. Auto Scaling 그룹과 로드 밸런서 통합을 사용하면 수신 트래픽에 대한 응답으로 필요에 따라 로드 밸런서에서 대상이 추가되거나 제거됩니다. 로드 밸런서는 컨테이너식 워크로드를 위해 Amazon ECS 및 Amazon EKS와 통합할 수도 있습니다.

- Amazon ECS - 서비스 로드 밸런싱
- Amazon EKS에서의 애플리케이션 로드 밸런싱
- Amazon EKS의 네트워크 로드 밸런싱

구현 단계

- 트래픽 볼륨, 가용성 및 애플리케이션 확장성을 포함한 로드 밸런싱 요구 사항을 정의합니다.
- 애플리케이션에 적합한 로드 밸런서 유형을 선택하세요.
  - HTTP/HTTPS 워크로드에 대해 Application Load Balancer를 사용합니다.
  - TCP 또는 UDP에서 실행되는 비HTTP 워크로드에 대해 Network Load Balancer를 사용합니다.
  - 두 제품의 기능을 모두 활용하려면 두 제품의 조합(NLB의 대상인 ALB)을 사용합니다. 예를 들어 ALB의 HTTP 헤더 기반 라우팅과 함께 NLB의 고정 IP를 사용하려는 경우 또는 AWS PrivateLink에 HTTP 워크로드를 노출하려는 경우 이를 수행할 수 있습니다.
  - 로드 밸런서의 전체 비교를 위해서 ELB 제품 비교를 참조하세요.
- 가능하면 SSL/TLS 오프로딩을 사용하세요.
  - AWS Certificate Manager와 통합된 Application Load Balancer 및 Network Load Balancer 모두에서 HTTPS/TLS 리스너를 구성합니다.
  - 일부 워크로드는 규정 준수상의 이유로 엔드 투 엔드 암호화가 필요할 수 있습니다. 이 경우 대상에서 암호화를 사용하도록 설정해야 합니다.
  - 보안 모범 사례는 SEC09-BP02 전송 중 암호화 적용을 참조하세요.
- 적절한 라우팅 알고리즘(ALB만)을 선택합니다.
  - 라우팅 알고리즘에 따라 백엔드 대상에서의 활용도 및 성능에 미치는 영향에 차이를 만들 수 있습니다. 예를 들어 ALB는 라우팅 알고리즘에 대한 두 가지 옵션을 제공합니다.
  - 미해결 요청 최소화: 애플리케이션에 대한 요청의 복잡성이 다양하거나 대상의 처리 능력이 다양한 경우 백엔드 대상에 로드를 더 효율적으로 분산하기 위해 사용합니다.
  - 라운드 로빈: 요청과 대상이 유사하거나 대상 간에 요청을 균등하게 분산해야 하는 경우에 사용합니다.

- 크로스 영역 또는 영역 격리를 고려합니다.
  - 지연 시간 개선 및 영역 장애 도메인을 위해 크로스 영역 끔(영역 격리)을 사용합니다. 이 기능은 NLB에서 기본적으로 꺼져 있으며 ALB의 경우 대상 그룹별로 해제할 수 있습니다.
  - 가용성과 유연성 향상을 위해 크로스 영역을 사용합니다. 기본적으로 크로스 영역은 ALB에서 활성화되어 있으며 NLB의 경우 대상 그룹별로 활성화할 수 있습니다.
- HTTP 워크로드(ALB만)에 대해 HTTP 연결 유지를 활성화합니다. 이 기능을 사용하면 로드 밸런서는 연결 유지 제한 시간이 만료될 때까지 백엔드 연결을 재사용하여 HTTP 요청 및 응답 시간을 개선하고 백엔드 대상의 리소스 사용률을 줄일 수 있습니다. Apache 및 Nginx에 대해 이 작업을 수행하는 방법에 대한 자세한 내용은 Apache 또는 NGINX를 ELB의 백엔드 서버로 사용하기 위한 최적의 설정은 무엇인가요?를 참조하세요.
- 로드 밸런서에 대한 모니터링을 켜세요.
  - Application Load Balancer 및 Network Load Balancer의 액세스 로그를 활성화합니다.
  - ALB에 대해 고려해야 할 주요 필드는 `request_processing_time`, `request_processing_time`, `response_processing_time`입니다.
  - NLB에 대해 고려해야 할 주요 필드는 `connection_time` 및 `tls_handshake_time`입니다.
  - 필요할 때 로그 쿼리를 준비합니다. Amazon Athena를 사용하여 ALB 로그와 NLB 로그를 모두 쿼리할 수 있습니다.
  - ALB의 `TargetResponseTime`과 같은 성능 관련 지표에 대한 경보를 생성합니다.

리소스

관련 문서:

- ELB 제품 비교
- AWS 글로벌 인프라
- Improving Performance and Reducing Cost Using Availability Zone Affinity
- Step by step for Log Analysis with Amazon Athena
- Application Load Balancer 로그 쿼리
- Monitor your Application Load Balancers
- Monitor your Network Load Balancer
- Use Elastic Load Balancing to distribute traffic across the instances in your Auto Scaling group

관련 비디오:

- AWS re:Invent 2023: What can networking do for your application?
- AWS re:Inforce 20: How to use Elastic Load Balancing to enhance your security posture at scale
- AWS re:Invent 2018: Elastic Load Balancing: Deep Dive and Best Practices
- AWS re:Invent 2021 - How to choose the right load balancer for your AWS workloads
- AWS re:Invent 2019: Get the most from Elastic Load Balancing for different workloads

관련 예제:

- Gateway Load Balancer
- CDK and AWS CloudFormation samples for Log Analysis with Amazon Athena

PERF04-BP05 성능을 개선할 수 있는 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 전송이 전송 제어 프로토콜(TCP)을 사용하는 경우 지연 시간이 길수록 전체 처리량이 줄어들 가능성이 높습니다. 이 문제는 TCP 튜닝 및 최적화된 전송 프로토콜을 사용하여 해결되지만 한 가지 해결 방법은 사용자 데이터그램 프로토콜(UDP)을 사용하는 것입니다.

일반적인 안티 패턴:

- 성능 요구 사항과 관계없이 모든 워크로드에 TCP를 사용합니다.

이 모범 사례 확립의 이점: 사용자와 워크로드 구성 요소 간의 통신에 적절한 프로토콜이 사용되는지 확인하면 애플리케이션의 전반적인 사용자 경험을 개선하는 데 도움이 됩니다. 예를 들어, 연결 없는 UDP는 빠른 속도를 허용하지만 재전송 기능 또는 높은 신뢰성을 제공하지 않습니다. TCP는 모든 기능을 갖춘 프로토콜이지만 패킷 처리에 더 많은 오버헤드가 필요합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

애플리케이션에 맞는 다양한 프로토콜을 선택할 수 있고 이 분야에 대한 전문 지식이 있다면 다른 프로토콜을 사용하여 애플리케이션과 최종 사용자 경험을 최적화하세요. 이 접근 방식은 상당히 어려우므로 먼저 다른 방법으로 애플리케이션을 최적화한 경우에만 시도해야 합니다.

워크로드의 성능을 향상하기 위한 주요 고려 사항은 지연 시간 및 처리량 요구 사항을 이해한 다음 성능을 최적화하는 네트워크 프로토콜을 선택하는 것입니다.

TCP 사용을 고려해야 할 시기

TCP는 신뢰할 수 있는 데이터 전달을 제공하며 신뢰성과 보장된 데이터 전달이 중요한 워크로드 구성 요소 간의 통신에 사용될 수 있습니다. 많은 웹 기반 애플리케이션은 HTTP 및 HTTPS와 같은 TCP 기반 프로토콜을 사용하여 구성 요소 간 통신하기 위한 TCP 소켓을 열어줍니다. 이메일 및 파일 데이터 전송은 TCP를 사용하는 일반적인 애플리케이션입니다. TCP는 애플리케이션 구성 요소 간의 간단하고 안정적인 전송 메커니즘이기 때문입니다. TCP와 함께 TLS를 사용하면 통신에 약간의 오버헤드가 추가되어 대기 시간이 증가하고 처리량이 감소할 수 있지만 보안상의 이점이 있습니다. 오버헤드는 주로 완료하는 데 여러 번의 왕복이 필요할 수 있는 핸드셰이크 프로세스의 추가 오버헤드에서 발생합니다. 핸드셰이크가 완료되면 데이터 암호화 및 복호화의 오버헤드는 비교적 작습니다.

UDP 사용을 고려해야 할 시기

UDP는 연결이 필요 없는 프로토콜이므로 로그, 모니터링 및 VOIP 데이터와 같이 빠르고 효율적인 전송이 필요한 애플리케이션에 적합합니다. 또한 워크로드의 최적 성능을 보장하기 위해 많은 클라이언트의 작은 쿼리에 응답하는 워크로드 구성 요소가 있는 경우 UDP를 사용하는 것이 좋습니다. 데이터그램 전송 계층 보안(DTLS)은 전송 계층 보안 전송 계층 보안(TLS)과 동일한 UDP입니다. UDP와 함께 DTLS를 사용하면 핸드셰이크 프로세스가 간소화되므로 오버헤드가가 데이터를 암호화하고 복호화할 때 발생합니다. DTLS는 또한 보안 파라미터를 나타내고 변조를 감지하기 위한 추가 필드를 포함하기 때문에 UDP 패킷에 소량의 오버헤드를 추가합니다.

SRD 사용을 고려해야 할 시기

SRD(Scalable Reliable Datagram)는 여러 경로에 걸쳐 트래픽을 로드 밸런싱할 수 있어 고처리량 워크로드에 최적화된 네트워크 전송 프로토콜이며 패킷 감소 또는 연결 장애에서 빠르게 복구합니다. 따라서 SRD는 컴퓨팅 노드 간에 처리량이 많고 지연 시간이 짧은 통신이 필요한 고성능 컴퓨팅(HPC) 워크로드에 가장 효과적입니다. 여기에는 노드 간에 대량의 데이터 전송을 수반하는 시뮬레이션, 모델링 및 데이터 분석과 같은 병렬 처리 작업이 포함될 수 있습니다.

구현 단계

- AWS Global Accelerator 및 AWS Transfer Family 서비스를 사용하여 온라인 파일 전송 애플리케이션의 처리량을 향상합니다. AWS Global Accelerator 서비스를 사용하면 클라이언트 디바이스와 AWS 워크로드 간에 지연 시간을 단축하는 데 도움이 됩니다. AWS Transfer Family를 사용하면 Secure Shell File Transfer Protocol(SFTP) 및 File Transfer Protocol over SSL(FTPS)과 같은 TCP 기반 프로토콜을 사용하여 파일 전송을 AWS 스토리지 서비스로 안전하게 확장하고 관리할 수 있습니다.

- 네트워크 지연 시간을 사용하여 TCP가 워크로드 구성 요소 간의 통신에 적합한지 확인합니다. 클라이언트 애플리케이션과 서버 간의 네트워크 지연 시간이 길면 TCP 3방향 핸드셰이크에 시간이 걸릴 수 있습니다. 그렇게 되면 애플리케이션의 응답성에 영향을 줄 수 있습니다. 첫 번째 바이트까지의 시간(TTFB) 및 왕복 시간(RTT)과 같은 지표를 사용하여 네트워크 지연 시간을 측정할 수 있습니다. 워크로드가 사용자에게 동적 콘텐츠를 제공하는 경우, 연결 설정 시간을 제거하도록 동적 콘텐츠에 대한 각 원본에 영구 연결을 설정하도록 Amazon CloudFront를 사용하는 것이 좋습니다. 그렇지 않으면 각 클라이언트 요청이 느려집니다.
- TCP 또는 UDP와 함께 TLS를 사용하면 암호화 및 복호화의 영향으로 인해 지연 시간이 증가하고 워크로드에 대한 처리량이 감소합니다. 이러한 워크로드의 경우, 백엔드 인스턴스 대신, 로드 밸런서가 SSL/TLS 암호화 및 복호화 프로세스를 처리하도록 허용하여 워크로드 성능을 개선하기 위해 Elastic Load Balancing에서 SSL/TLS 오프로딩을 고려하세요. 이를 통해 백엔드 인스턴스의 CPU 사용률을 줄여 성능을 향상시키고 용량을 늘릴 수 있습니다.
- Network Load Balancer(NLB)를 사용하여 인증 및 권한 부여, 로깅, DNS, IoT, 스트리밍 미디어 등 UDP 프로토콜에 의존하는 서비스를 배포하세요. NLB는 여러 대상에 걸쳐 수신되는 UDP 트래픽을 배포하여 워크로드를 수평 확장하고 용량을 늘리고 단일 대상의 오버헤드를 줄일 수 있습니다.
- 고성능 컴퓨팅(HPC) 워크로드의 경우 Elastic Network Adapter(ENA) Express 기능 사용을 고려하세요. 이 기능은 SRD 프로토콜을 사용하여 EC2 인스턴스 간의 네트워크 트래픽에 대해 더 높은 단일 흐름 대역폭(25Gbps) 및 더 짧은 테일 지연 시간(99.9 백분위수)을 제공함으로서 네트워크 성능을 개선합니다.
- Application Load Balancer(ALB)를 사용하여 워크로드 구성 요소 간에 또는 gRPC 지원 클라이언트와 서비스 간에 gRPC(원격 프로시저 호출) 트래픽을 라우팅하고 로드를 분산합니다. gRPC는 전송에 TCP 기반 HTTP/2 프로토콜을 사용하며 더 가벼운 네트워크 공간, 압축, 효율적인 이진 직렬화, 다양한 언어 지원, 양방향 스트리밍과 같은 성능상의 이점을 제공합니다.

리소스

관련 문서:

- How to route UDP traffic into Kubernetes
- Application Load Balancer
- Linux 기반 EC2의 향상된 네트워킹
- Windows 기반 EC2의 향상된 네트워킹
- EC2 배치 그룹
- Linux 인스턴스에서 Elastic Network Adapter(ENA)로 향상된 네트워킹 활성화
- Network Load Balancer

- AWS의 네트워킹 제품
- Transitioning to Latency-Based Routing in Amazon Route 53
- VPC 엔드포인트

관련 비디오:

- AWS re:Invent 2022 – Scaling network performance on next-gen Amazon Elastic Compute Cloud instances
- AWS re:Invent 2022 – Application networking foundations

관련 예제:

- AWS Transit Gateway and Scalable Security Solutions
- AWS Networking 워크숍

PERF04-BP06 네트워크 요구 사항에 따라 워크로드의 위치 선택

리소스 배치 옵션을 평가하여 네트워크 지연 시간을 줄이고 처리량을 향상시켜 페이지 로드 및 데이터 전송 시간을 줄임으로써 최적의 사용자 경험을 제공합니다.

일반적인 안티 패턴:

- 모든 워크로드 리소스를 하나의 지리적 위치로 통합합니다.
- 워크로드 최종 사용자가 아니라 본인과 가장 가까운 리전을 선택했습니다.

이 모범 사례 확립의 이점: 사용자 경험은 사용자와 애플리케이션 간의 지연 시간에 크게 영향을 받습니다. 적절한 AWS 리전 및 AWS 프라이빗 글로벌 네트워크를 사용하면 대기 시간을 줄이고 원격 사용자에게 더 나은 경험을 제공할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

Amazon EC2 인스턴스와 같은 리소스는 AWS 리전의 가용 영역, AWS 로컬 영역, AWS Outposts 또는 AWS Wavelength 영역에 배치됩니다. 이 위치를 선택하면 주어진 사용자 위치의 네트워크 지연 시간 및 처리량에 영향을 미칩니다. Amazon CloudFront 및 AWS Global Accelerator와 같은 엣지 서비스를

사용하여 엣지 로케이션의 콘텐츠를 캐싱하거나 AWS 글로벌 네트워크를 통해 워크로드에 대한 최적의 경로를 사용자에게 제공함으로써 네트워크 성능을 향상시킬 수 있습니다.

Amazon EC2에서는 네트워킹용 배치 그룹을 제공합니다. 배치 그룹은 지연 시간을 줄이기 위한 인스턴스의 논리적 그룹입니다. 지원되는 인스턴스 유형이 포함된 배치 그룹과 Elastic Network Adapter(ENA)를 사용하면 지연 시간이 짧고 지터가 감소된 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.

지연 시간에 민감한 서비스는 Amazon CloudFront와 같은 AWS 글로벌 네트워크를 사용하여 엣지 로케이션에서 제공됩니다. 이러한 엣지 로케이션에서는 보통 콘텐츠 전송 네트워크(CDN) 및 도메인 이름 시스템(DNS)과 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.

지연 시간을 줄이고 콘텐츠 캐싱을 활성화하려면 엣지 서비스를 사용합니다. 이러한 방식의 이점을 최대한 활용하려면 DNS 및 HTTP/HTTPS용으로 캐시 제어를 올바르게 구성하세요.

구현 단계

- 네트워크 인터페이스를 오가는 IP 트래픽에 대한 정보를 캡처합니다.
  - VPC 흐름 로그를 사용하여 IP 트래픽 로깅
  - How the client IP address is preserved in AWS Global Accelerator
- 워크로드의 네트워크 액세스 패턴을 분석하여 사용자가 애플리케이션을 사용하는 방법을 식별합니다.
  - Amazon CloudWatch 및 AWS CloudTrail과 같은 모니터링 도구를 사용하여 네트워크 활동에 대한 데이터를 수집합니다.
  - 데이터를 분석하여 네트워크 액세스 패턴을 식별합니다.
- 다음과 같은 주요 요소를 토대로 하여 워크로드 배포용 리전을 선택합니다.
  - 데이터 위치: 데이터를 많이 사용하는 애플리케이션의 경우(예: 빅 데이터 및 기계 학습) 애플리케이션 코드는 최대한 데이터와 가까운 위치에서 실행되어야 합니다.
  - 사용자 위치: 사용자가 직접 사용하는 애플리케이션의 경우 워크로드의 사용자와 가까운 하나 이상의 리전을 선택합니다.
  - 기타 제약 조건: What to Consider when Selecting a Region for your Workloads에 나와 있는 비용 및 규정 준수 등 제약 요건을 고려합니다.

- AWS 로컬 영역을 사용하여 비디오 렌더링과 같은 워크로드를 실행합니다. 로컬 영역에서는 최종 사용자와 가까운 위치에 컴퓨팅 및 스토리지 리소스를 배치함으로써 이점을 얻을 수 있습니다.
- 온프레미스에 남아 있어야 하는 워크로드 및 해당 워크로드를 AWS의 나머지 워크로드와 함께 원활하게 실행하려는 워크로드에 대해서는 AWS Outposts를 사용합니다.
- 고해상도 라이브 비디오 스트리밍, 고음질 오디오 및 증강 현실 또는 가상 현실(AR/VR)과 같은 5G 디바이스용 애플리케이션은 지연 시간이 매우 짧아야 합니다. 이러한 애플리케이션에서는 AWS Wavelength를 고려합니다. AWS Wavelength에서는 5G 네트워크 내에 AWS 컴퓨팅 및 스토리지 서비스를 포함하여 지연 시간이 짧은 애플리케이션을 개발, 배포 및 확장하기 위한 모바일 엣지 컴퓨팅 인프라를 제공합니다.
- 자주 사용하는 자산에 로컬 캐싱 또는 AWS Caching Solutions를 사용하여 성능을 개선하고, 데이터 이동을 줄이며, 환경에 미치는 영향을 줄입니다.

Service	사용해야 하는 경우
Amazon CloudFront	이미지, 스크립트, 동영상 등의 정적 콘텐츠와 API 응답 또는 웹 애플리케이션 등의 동적 콘텐츠를 캐시하는 데 사용합니다.
Amazon ElastiCache	웹 애플리케이션의 콘텐츠를 캐시하는 데 사용합니다.
DynamoDB Accelerator	DynamoDB 테이블에 인 메모리 가속화를 추가하는 데 사용합니다.

- 워크로드 사용자에게 더 가까운 위치에서 코드를 실행할 수 있는 서비스를 사용합니다.

Service	사용해야 하는 경우
Lambda@Edge	객체가 캐시에 없는 경우 시작되는 컴퓨팅 집약적 작업에 사용합니다.
Amazon CloudFront Functions	HTTP(s) 요청 또는 응답 조작 등과 같이 단기 기능으로 실행할 수 있는 간단한 사용 사례에 사용합니다.
AWS IoT Greengrass	커넥티드 디바이스를 위한 로컬 컴퓨팅, 메시징 및 데이터 캐시를 실행하는 데 사용합니다.

- 일부 애플리케이션은 첫 번째 바이트까지의 지연 시간과 지터를 줄이고 처리량을 늘려 고정 진입 지점 또는 그 이상의 성능이 필요합니다. 이러한 애플리케이션은 엣지 로케이션에서 정적 애니캐스트 IP 주소 및 TCP 종료를 제공하는 네트워킹 서비스를 활용할 수 있습니다. AWS Global Accelerator는 애플리케이션의 성능을 최대 60%까지 향상시키고 다중 리전 아키텍처에 빠른 장애 조치를 제공합니다. AWS Global Accelerator에서는 하나 이상의 AWS 리전에 호스팅되는 애플리케이션의 고정 진입 지점으로 사용되는 정적 애니캐스트 IP 주소를 제공합니다. 이 IP 주소를 사용하면 가능한 한 사용자와 가까운 AWS 글로벌 네트워크으로 트래픽이 유입될 수 있습니다. AWS Global Accelerator는 클라이언트와 가장 가까운 AWS 엣지 로케이션 간에 TCP 연결을 설정하여 초기 연결 설정 시간을 줄입니다. AWS Global Accelerator의 사용을 검토하여 TCP/UDP 워크로드의 성능을 향상시키고 다중 리전 아키텍처에 빠른 장애 조치를 제공합니다.

리소스

관련 모범 사례:

- COST07-BP02 비용을 기준으로 리전 구현
- COST08-BP03 데이터 전송 비용을 줄이기 위한 서비스 구현
- REL10-BP01 워크로드를 여러 위치에 배포
- REL10-BP02 다중 위치 배포에 적합한 위치 선택
- SUS01-BP01 비즈니스 요구 사항과 지속 가능성 목표를 기준으로 리전 선택
- SUS02-BP04 네트워킹 요구 사항에 따라 워크로드의 지리적 배치 최적화
- SUS04-BP07 네트워크 간 데이터 이동 최소화

관련 문서:

- AWS 글로벌 인프라
- AWS Local Zones and AWS Outposts, choosing the right technology for your edge workload
- 배치 그룹
- AWS 로컬 영역
- AWS Outposts
- AWS Wavelength
- Amazon CloudFront
- AWS Global Accelerator
- AWS Direct Connect

- AWS Site-to-Site VPN
- Amazon Route 53

관련 비디오:

- AWS Local Zones Explainer Video
- AWS Outposts: Overview and How it Works
- AWS re:Invent 2023 - A migration strategy for edge and on-premises workloads
- AWS re:Invent 2021 - AWS Outposts: Bringing the AWS experience on premises
- AWS re:Invent 2020: AWS Wavelength: Run apps with ultra-low latency at 5G edge
- AWS re:Invent 2022 - AWS Local Zones: Building applications for a distributed edge
- AWS re:Invent 2021 - Building low-latency websites with Amazon CloudFront
- AWS re:Invent 2022 - Improve performance and availability with AWS Global Accelerator
- AWS re:Invent 2022 - Build your global wide area network using AWS
- AWS re:Invent 2020: Global traffic management with Amazon Route 53

관련 예제:

- AWS Global Accelerator Custom Routing 워크숍
- Handling Rewrites and Redirects using Edge Functions

PERF04-BP07 지표를 기준으로 네트워크 구성 최적화

수집 및 분석된 데이터가 제공하는 정보를 사용하여 네트워크 구성 최적화를 결정합니다.

일반적인 안티 패턴:

- 모든 성능 관련 문제가 애플리케이션 관련 문제라고 가정합니다.
- 워크로드를 배포한 위치와 가까운 위치에서만 네트워크 성능을 테스트합니다.
- 모든 네트워크 서비스에 기본 구성을 사용합니다.
- 충분한 용량을 제공하려고 네트워크 리소스를 과도하게 프로비저닝합니다.

이 모범 사례 확립의 이점: AWS 네트워크와 관련된 필수 지표를 수집하고 네트워크 모니터링 도구 구현을 통해 네트워크 성능을 이해하고 네트워크 구성을 최적화할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

VPC, 서브넷 또는 네트워크 인터페이스를 오가는 트래픽을 모니터링하는 것은 AWS 네트워크 리소스를 활용하는 방법과 네트워크 구성을 최적화하는 방법을 이해하는 데 중요합니다. 다음 AWS 네트워킹 도구를 사용하면 트래픽 사용, 네트워크 액세스 및 로그에 대한 정보를 추가로 검사할 수 있습니다.

구현 단계

- 수집할 대기 시간 또는 패킷 손실과 같은 핵심 성과 지표를 식별합니다. AWS는 이러한 지표를 수집하는 데 도움이 되는 몇 가지 도구를 제공합니다. 다음 도구를 사용하면 트래픽 사용, 네트워크 액세스 및 로그에 대한 정보를 추가로 검사할 수 있습니다.

AWS 도구	사용 장소
Amazon VPC IP 주소 관리자.	IPAM을 사용하여 AWS 및 온프레미스 워크로드를 위해 IP 주소를 계획, 추적 및 모니터링할 수 있습니다. 이는 IP 주소 사용 및 할당을 최적화하는 모범 사례입니다.
VPC 흐름 로그	VPC 흐름 로그를 사용하여 VPC의 네트워크 인터페이스에서 전송되고 수신되는 IP 트래픽에 대한 정보를 캡처합니다. VPC 흐름 로그를 사용하면 지나치게 제한적이거나 허용되는 보안 그룹 규칙을 진단하고 네트워크 인터페이스와의 트래픽 방향을 결정할 수 있습니다.
AWS Transit Gateway 흐름 로그	AWS Transit Gateway 흐름 로그를 사용하여 전송 게이트웨이를 오가는 IP 트래픽에 대한 정보를 캡처하세요.
DNS 쿼리 로깅	Route 53에서 수신하는 퍼블릭 또는 프라이빗 DNS 쿼리에 대한 정보를 기록합니다. DNS 로그를 사용하면 요청된 도메인 또는 하위 도메인 또는 DNS 쿼리에 응답한 Route 53 엣지 로케이션을 이해하여 DNS 구성을 최적화할 수 있습니다.

AWS 도구	사용 장소
Reachability Analyzer	Reachability Analyzer를 사용하여 네트워크 도달 가능성을 분석하고 디버깅합니다. Reachability Analyzer는 VPC의 소스 리소스와 대상 리소스 간의 연결을 테스트할 수 있는 구성 분석 도구입니다. 이 도구를 사용하면 네트워크 구성이 의도한 연결과 일치하는지 확인하는 데 도움이 됩니다.
Network Access Analyzer	Network Access Analyzer를 사용하면 리소스에 대한 네트워크 액세스를 파악할 수 있습니다. Network Access Analyzer를 사용하면 네트워크 액세스 요구 사항을 지정하고 지정된 요구 사항을 충족하지 않는 네트워크 경로를 식별할 수 있습니다. 해당 네트워크 구성을 최적화하면 네트워크 상태를 이해하고 확인하며 AWS의 네트워크가 규정 준수 요구 사항을 충족하는지 입증할 수 있습니다.
Amazon CloudWatch	Amazon CloudWatch를 사용하고 네트워크 옵션에 적합한 지표를 켭니다. 워크로드에 적합한 네트워크 지표를 선택해야 합니다. 예를 들어 VPC 네트워크 주소 사용, VPC NAT 게이트웨이, AWS Transit Gateway, VPN 터널, AWS Network Firewall, Elastic Load Balancing 및 AWS Direct Connect에 대한 지표를 활성화할 수 있습니다. 지표를 지속적으로 모니터링하는 것은 네트워크 상태와 사용량을 관찰하고 파악하는 좋은 방법이며 관찰을 기반으로 네트워크 구성을 최적화하는 데 도움이 됩니다.

AWS 도구	사용 장소
AWS Network Manager	AWS Network Manager를 사용하면 운영 및 계획 목적으로 AWS 글로벌 네트워크의 실시간 및 과거 성능을 모니터링할 수 있습니다. Network Manager에서는 AWS 리전 및 가용 영역 간 그리고 각 가용 영역 내에서 집계된 네트워크 지연 시간을 제공하므로 애플리케이션 성능이 기본 AWS 네트워크의 성능과 어떻게 연관되는지 더 잘 이해할 수 있습니다.
Amazon CloudWatch RUM	Amazon CloudWatch RUM을 사용하여 사용자 경험을 식별, 이해 및 개선하는 데 도움이 되는 인사이트를 제공하는 지표를 수집하세요.

- VPC 및 AWS Transit Gateway 흐름 로그를 사용하여 상위 발화자와 애플리케이션 트래픽 패턴을 식별합니다.
- VPC, 서브넷, 라우팅을 포함한 현재 네트워크 아키텍처를 평가하고 최적화합니다. 예를 들어, 다양한 VPC 피어링 또는 AWS Transit Gateway가 아키텍처의 네트워킹을 개선하는 데 어떻게 도움이 되는지 평가할 수 있습니다.
- 네트워크의 라우팅 경로를 평가하여 대상 간 최단 경로가 항상 사용되는지 확인하세요. Network Access Analyzer는 이를 수행하는 데 도움이 됩니다.

리소스

관련 문서:

- Public DNS query logging
- IPAM이란?
- What is Reachability Analyzer?
- What is Network Access Analyzer?
- VPC의 CloudWatch 지표
- Apache Parquet 형식의 VPC 흐름 로그를 사용하여 네트워크 분석 성능을 최적화하고 비용을 절감합니다.
- Monitoring your global and core networks with Amazon CloudWatch metrics

- Continuously monitor network traffic and resources

관련 비디오:

- AWS re:Invent 2023 – A developer's guide to cloud networking
- AWS re:Invent 2023 – Ready for what’s next? Designing networks for growth and flexibility
- AWS re:Invent 2023 – Advanced VPC designs and new capabilities
- AWS re:Invent 2022 – Dive deep on AWS networking infrastructure
- AWS re:Invent 2020 – Networking best practices and tips with the AWS Well-Architected Framework
- AWS re:Invent 2020 – Monitoring and troubleshooting network traffic

관련 예제:

- AWS Networking 워크숍
- AWS Network Monitoring
- Observing and diagnosing your network on AWS
- Finding and addressing network misconfigurations on AWS

프로세스 및 문화

워크로드를 설계할 때는 효율적인 고성능 클라우드 워크로드를 더 잘 실행하는 데 도움이 되도록 채택할 수 있는 원칙과 관행이 있습니다. 이 중점 영역에서는 클라우드 워크로드의 성능 효율성을 촉진하는 문화를 채택하는 데 도움이 되는 모범 사례를 제공합니다.

이러한 문화를 구축하기 위해 다음과 같은 핵심 원칙을 고려하세요.

- 코드형 인프라: AWS CloudFormation 템플릿 등의 접근 방식을 사용하여 코드형 인프라를 정의합니다. 템플릿을 사용하면 애플리케이션 코드와 구성과 함께 인프라를 소스 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 지속적 통합 및 지속적 전달(CI/CD) 파이프라인을 사용하여 인프라를 배포합니다. 이렇게 하면 반복 가능하며 일관성 있는 저렴한 방식으로 배포를 반복해서 진행할 수 있습니다.
- 잘 정의된 지표: 핵심 성과 지표(KPI)를 캡처하기 위해 지표를 설정하고 모니터링합니다. 기술과 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 99번째 백분위수를 선택할 수 있습니다.
- 자동 성능 테스트: 배포 프로세스의 일환으로 더 빠르게 실행되는 테스트가 정상적으로 완료되고 나면 성능 테스트를 자동으로 시작합니다. 이 자동 테스트에서는 새 환경을 설정하고, 테스트 데이터 등의 초기 조건을 설정한 다음 일련의 벤치마크와 로드 테스트를 실행해야 합니다. 시간별 성능 변화를 추적할 수 있도록 이러한 테스트의 결과를 빌드에 다시 연결해야 합니다. 오래 실행되는 테스트의 경우에는 테스트를 빌드의 다른 부분과 비동기식으로 파이프라인에 포함할 수 있습니다. Amazon EC2 스팟 인스턴스를 사용하여 야간에 성능을 테스트할 수도 있습니다.
- 로드 생성: 가상 또는 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 서비스형 소프트웨어(SaaS) 솔루션을 사용하여 로드를 생성할 수 있습니다. 비용 효율적인 방식으로 로드를 생성할 수 있도록 AWS Marketplace 솔루션 및 스팟 인스턴스를 사용하는 방법을 고려하세요.
- 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 성과 지표를 겹쳐서 표시합니다. 콜 그래프나 코드는 문제를 빠르게 확인하는 데 도움을 줍니다.
- 정기적인 검토 프로세스: 아키텍처의 성능 저하는 성능 검토 프로세스가 없거나 효과적이지 않은 경우 주로 발생합니다. 아키텍처의 성능이 좋지 않은 경우 성능 검토 프로세스를 구현하면 반복적으로 개선을 주도할 수 있습니다.
- 지속적 최적화: 클라우드 워크로드의 성능 효율성을 지속적으로 최적화할 수 있는 문화를 채택합니다.

모범 사례

- PERF05-BP01 워크로드 상태 및 성능을 측정하기 위한 핵심 성과 지표(KPI) 수립
- PERF05-BP02 모니터링 솔루션을 사용하여 성능이 가장 중요한 영역 파악
- PERF05-BP03 워크로드 성능 개선을 위한 프로세스 정의
- PERF05-BP04 워크로드 로드 테스트
- PERF05-BP05 자동화를 사용하여 성능 관련 문제 사전 해결
- PERF05-BP06 워크로드 및 서비스 최신 상태 유지
- PERF05-BP07 정기적으로 지표 검토

PERF05-BP01 워크로드 상태 및 성능을 측정하기 위한 핵심 성과 지표(KPI) 수립

워크로드 성능을 양적 및 질적으로 측정하는 KPI를 식별합니다. KPI는 비즈니스 목표와 관련된 워크로드의 상태와 성과를 측정하는 데 도움이 됩니다.

일반적인 안티 패턴:

- 시스템 수준 지표만 모니터링하여 워크로드에 대한 인사이트를 얻고, 해당 지표에 대한 비즈니스 영향을 이해하지 못합니다.
- KPI가 이미 표준 지표 데이터로 게시 및 공유되고 있다고 가정합니다.
- 정량적이고 측정 가능한 KPI를 정의하지 않습니다.
- KPI를 비즈니스 목표나 전략에 맞추지 않습니다.

이 모범 사례 확립의 이점: 워크로드 상태 및 성능을 나타내는 특정 KPI를 식별하면 팀의 우선순위를 조정하고 성공적인 비즈니스 결과를 정의하는 데 도움이 됩니다. 이러한 지표를 모든 부서와 공유하면 임곗값, 기대치 및 비즈니스에 미치는 영향을 파악하고, 이에 따른 조정이 가능해집니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

KPI를 통해 비즈니스 및 엔지니어링 팀은 목표 측정과 전략 그리고 이러한 요인을 조합하여 비즈니스 성과를 도출하는 방법에 대해 협의할 수 있습니다. 예를 들어, 웹 사이트 워크로드에는 전체 성능을 나타내는 지표로 페이지 로드 시간을 사용할 수 있습니다. 이 지표는 사용자 경험을 측정하는 여러 데이터 포인트 중 하나입니다. 페이지 로드 시간 임곗값을 파악하는 것 말고도 이상적인 성능이 충족되지 않을 경우 예상되는 결과나 비즈니스 위험도 문서화해야 합니다. 페이지 로드 시간이 길면 최종 사용자에게 직접적인 영향을 주고, 사용자 경험 수준이 떨어져 고객이 이탈하는 결과가 발생할 수 있습니다. KPI 임곗값을 정의할 때는 업계 벤치마크와 최종 사용자 기대치를 모두 고려해야 합니다. 가령 현재 업계 벤치마크에 따르면 웹 페이지를 2초 안에 로드하면 되지만 최종 사용자는 웹 페이지가 1초 안에 로드될 것으로 기대한다면, 이러한 데이터 포인트를 모두 고려해서 KPI를 설정해야 합니다.

팀은 참조용으로 실시간 세분화된 데이터와 기록 데이터를 사용하여 워크로드 KPI를 평가하고, KPI 데이터에 대한 지표 산술을 수행하여 운영 및 사용률 인사이트를 도출하는 대시보드를 만들어야 합니다. KPI는 문서화되어야 하며 비즈니스 목표와 전략을 지원하는 임곗값을 포함해야 하고 모니터링 중인 지표에 매핑되어야 합니다. 비즈니스 목표, 전략 또는 최종 사용자 요구 사항이 변경되면 KPI를 다시 검토해야 합니다.

구현 단계

- 이해관계자 식별: 개발 및 운영 팀을 비롯한 주요 비즈니스 이해관계자를 식별하고 문서화합니다.
- 목표 정의: 이러한 이해관계자들과 협력하여 워크로드의 목표를 정의하고 문서화합니다. 처리량, 응답 시간, 비용 등 워크로드 성능의 중요한 측면과 사용자 만족도와 같은 비즈니스 목표를 고려합니다.
- 업계 모범 사례 검토: 업계 모범 사례를 검토하여 워크로드 목표에 부합하는 관련 KPI를 파악합니다.
- 지표 식별: 워크로드 목표에 부합하며 성능 및 비즈니스 목표를 측정하는 데 도움이 될 수 있는 지표를 식별합니다. 이러한 지표를 기반으로 KPI를 설정합니다. 지표의 예로는 평균 응답 시간이나 동시 사용자 수와 같은 측정값이 있습니다.
- KPI 정의 및 문서화: 업계 모범 사례와 워크로드 목표를 사용하여 워크로드 KPI의 목표를 설정합니다. 이 정보를 사용하여 심각도 또는 경보 수준에 대한 KPI 임곗값을 설정합니다. KPI가 충족되지 않을 경우의 위험과 영향을 식별하고 문서화합니다.

- 모니터링 구현: Amazon CloudWatch 또는 AWS Config와 같은 모니터링 도구를 사용하여 지표를 수집하고 KPI를 측정합니다.
- KPI 시각적 전달: KPI를 시각화하고 이해관계자에게 전달하기 위한 Amazon QuickSight와 같은 대시보드 도구를 사용합니다.
- 분석 및 최적화: 정기적으로 KPI를 검토하고 분석하여 개선이 필요한 워크로드 영역을 파악합니다. 이해관계자와 협력하여 이러한 개선 사항을 구현합니다.
- 보관 및 개선: 지표와 KPI를 정기적으로 검토하여 효과를 평가합니다. 이는 특히 비즈니스 목표나 워크로드 성능이 변경되는 경우에는 더욱 중요합니다.

리소스

관련 문서:

- CloudWatch 설명서
- 모니터링, 로깅 및 성능 AWS Partner
- AWS observability tools
- The Importance of Key Performance Indicators (KPIs) for Large-Scale Cloud Migrations
- How to track your cost optimization KPIs with the KPI Dashboard
- X-Ray 설명서
- Amazon CloudWatch 대시보드 사용
- QuickSight KPIs

관련 비디오:

- AWS re:Invent 2023 - Optimize cost and performance and track progress toward mitigation
- AWS re:Invent 2023 - Manage resource lifecycle events at scale with AWS Health
- AWS re:Invent 2023 - Performance & efficiency at Pinterest: Optimizing the latest instances
- AWS re:Invent 2022 - AWS optimization: Actionable steps for immediate results
- AWS re:Invent 2023 - Building an effective observability strategy
- AWS Summit SF 2022 - Full-stack observability and application monitoring with AWS
- AWS re:Invent 2023 - Scaling on AWS for the first 10 million users
- AWS re:Invent 2022 - How Amazon uses better metrics for improved website performance
- Creating an Effective Metrics Strategy for Your Business | AWS Events

관련 예제:

- Creating a dashboard with QuickSight

PERF05-BP02 모니터링 솔루션을 사용하여 성능이 가장 중요한 영역 파악

워크로드 성능을 개선하여 효율성을 높이고 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 이동하는 방법으로 성능을 개선할 수 있습니다.

일반적인 안티 패턴:

- CPU 사용률 또는 메모리 압력과 같은 표준 컴퓨팅 지표로 성능 문제를 파악하기에 충분하다고 가정합니다.
- 선택한 모니터링 소프트웨어에서 기록한 기본 지표만 사용합니다.
- 문제가 발생한 경우에만 지표를 검토합니다.

이 모범 사례 확립의 이점: 성능의 중요 영역을 이해함으로써 워크로드 소유자가 KPI를 모니터링하고 큰 영향을 미치는 개선에 우선순위를 지정할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

트래픽 패턴, 지연 시간 및 중요한 성능 영역을 파악할 수 있는 엔드 투 엔드 추적을 설정합니다. 데이터 액세스 패턴을 모니터링하여 쿼리 속도가 느리거나 데이터 조각이 잘못되거나 잘못 분할된 데이터를 찾습니다. 로드 테스트 또는 모니터링을 사용하여 워크로드의 제한된 영역을 파악합니다.

아키텍처, 트래픽 패턴 및 데이터 액세스 패턴을 이해하여 성능 효율성이 향상되고 지연 시간 및 처리 시간을 파악합니다. 워크로드가 증가하면서 고객 환경에 영향을 미칠 수 있는 잠재적 병목 현상을 파악합니다. 이러한 영역을 조사한 후에는 이러한 성능 문제를 해결하기 위해 어떤 솔루션을 배포할 수 있는지 살펴보세요.

구현 단계

- 엔드 투 엔드 모니터링을 설정하여 모든 워크로드 구성 요소 및 지표를 캡처합니다. 다음은 AWS에 대한 모니터링 솔루션의 예입니다.

Service	사용 장소
Amazon CloudWatch 실제 사용자 모니터링 (RUM)	실제 사용자 클라이언트 측 및 프론트엔드 세션에서 애플리케이션 성과 지표를 포착합니다.
AWS X-Ray	애플리케이션 계층을 통해 트래픽을 추적하고 구성 요소와 종속성 간의 지연 시간을 식별합니다. X-Ray 서비스 맵을 사용하여 워크로드 구성 요소 간 관계 및 지연 시간을 확인합니다.
Amazon Relational Database Service 성능 개선 도우미	데이터베이스 성과 지표를 보고 성능 향상을 식별합니다.
Amazon RDS 향상된 모니터링	데이터베이스 OS 성과 지표를 확인합니다.
Amazon DevOps Guru	비정상적인 운영 패턴을 감지하여 고객에게 영향을 미치기 전에 운영 문제를 식별할 수 있습니다.

- 지표를 생성하고 트래픽 패턴, 병목 현상 및 중요한 성능 영역을 파악하기 위한 테스트를 수행합니다. 다음은 테스트를 수행하는 방법에 대한 몇 가지 예입니다.
  - 시간이 지남에 따라 일관된 지표를 생성하기 위해 Linux 크론 작업 또는 rate 표현식을 사용하여 프로그래밍 방식으로 브라우저 기반 사용자 활동을 모방하도록 CloudWatch Synthetic Canary를 설정합니다.
  - AWS 분산 로드 테스트 솔루션을 사용하여 피크 트래픽을 생성하거나 예상되는 증가율로 워크로드를 테스트합니다.
- 지표 및 원격 측정을 평가하여 중요한 성능 영역을 파악합니다. 팀과 함께 이러한 영역을 검토하여 병목 현상을 방지할 수 있는 모니터링 및 솔루션을 논의합니다.
- 성능 개선을 실험하고 데이터로 이러한 변경 사항을 측정합니다. 예를 들어 워크로드에 대한 새로운 개선 사항과 성능 영향을 테스트하기 위해 CloudWatch Evidently를 사용할 수 있습니다.

리소스

관련 문서:

- What's new in AWS Observability at re:Invent 2023

- Amazon Builders' Library
- X-Ray 설명서
- Amazon CloudWatch RUM
- Amazon DevOps Guru

관련 비디오:

- AWS re:Invent 2023 - [LAUNCH] Application monitoring for modern workloads
- AWS re:Invent 2023 - Implementing application observability
- AWS re:Invent 2023 - Building an effective observability strategy
- AWS Summit SF 2022 - Full-stack observability and application monitoring with AWS
- AWS re:Invent 2022 - AWS optimization: Actionable steps for immediate results
- AWS re:Invent 2022 - The Amazon Builders' Library: 25 years of Amazon operational excellence
- AWS re:Invent 2022 - How Amazon uses better metrics for improved website performance
- Visual Monitoring of Applications with Amazon CloudWatch Synthetics

관련 예제:

- Measure page load time with Amazon CloudWatch Synthetics
- Amazon CloudWatch RUM Web Client
- X-Ray SDK for Python
- Distributed Load Testing on AWS

PERF05-BP03 워크로드 성능 개선을 위한 프로세스 정의

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

일반적인 안티 패턴:

- 시간이 지나면 현재 아키텍처가 정적 아키텍처가 되고 업데이트되지 않는다고 가정합니다.
- 시간이 지나면 타당한 지표 없이 아키텍처 변경을 도입합니다.

이 모범 사례 확립의 이점: 아키텍처 변경을 위한 프로세스를 정의하면 수집된 데이터를 사용하여 워크로드 설계에 지속적으로 영향을 줄 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

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 서비스나 기술을 습득할 때 이 정보를 사용하여 제약 조건이나 병목 현상을 완화하는 방법을 파악합니다.

워크로드에 대한 주요 성능 제약을 파악합니다. 워크로드의 성능 제약을 문서화하면 어떤 종류의 혁신이 워크로드의 성능을 개선할 수 있는지 알 수 있습니다.

구현 단계

- KPI 식별: PERF05-BP01 워크로드 상태 및 성능을 측정하기 위한 핵심 성과 지표(KPI) 수립에 설명된 대로 워크로드 성능 KPI를 식별하여 워크로드의 기준을 설정합니다.
- 모니터링 구현: AWS 관찰성 도구를 사용하여 성과 지표를 수집하고 KPI를 측정합니다.
- 분석 수행: 심층 분석을 수행하여 PERF05-BP02 모니터링 솔루션을 사용하여 성능이 가장 중요한 영역 파악에 설명된 대로 워크로드에서 성능이 저조한 영역(예: 구성 및 애플리케이션 코드)을 식별합니다. 분석 및 성능 도구를 사용하여 성능 개선 전략을 식별합니다.
- 개선 사항 검증: 샌드박스 또는 사전 프로덕션 환경을 사용하여 개선 전략의 유효성을 검증합니다.
- 변경 사항 구현: 프로덕션 환경에서 변경 사항을 구현하고 워크로드의 성능을 지속적으로 모니터링합니다. 개선 사항을 문서화하고 변경 사항을 이해관계자에게 전달합니다.
- 보관 및 개선: 성과 개선 프로세스를 정기적으로 검토하여 개선이 필요한 영역을 식별합니다.

리소스

관련 문서:

- AWS 블로그
- AWS의 새로운 소식
- AWS Skill Builder

관련 비디오:

- AWS re:Invent 2022 - Delivering sustainable, high-performing architectures
- AWS re:Invent 2023 - Optimize cost and performance and track progress toward mitigation
- AWS re:Invent 2022 - AWS optimization: Actionable steps for immediate results
- AWS re:Invent 2022 - Optimize your AWS workloads with best-practice guidance

관련 예제:

- AWS Github

PERF05-BP04 워크로드 로드 테스트

워크로드 로드 테스트를 통해 프로덕션 로드를 처리할 수 있는지 확인하고 성능 병목 현상을 파악할 수 있습니다.

일반적인 안티 패턴:

- 전체 워크로드가 아니라 워크로드의 개별 부분을 로드 테스트를 수행합니다.
- 프로덕션 환경과 동일하지 않은 인프라에서 로드 테스트를 수행합니다.
- 향후 문제가 발생할 수 있는 위치를 예측할 때 예상 로드에 대해서만 로드 테스트를 수행합니다.
- Amazon EC2 테스트 정책을 참조하거나 시뮬레이션된 이벤트 제출 양식을 제출하지 않고도 로드 테스트를 수행할 수 있습니다. 이는 서비스 거부 이벤트처럼 보이기 때문에 테스트 실행이 실패하게 됩니다.

이 모범 사례 확립의 이점: 로드 테스트에서 성능을 측정하면 로드 증가의 영향을 받게 될 위치를 알 수 있습니다. 이렇게 하면 워크로드에 영향을 미치기 전에 필요한 변경 사항을 예상할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

클라우드에서의 로드 테스트는 예상되는 사용자 부하가 있는 실제 조건에서 클라우드 워크로드의 성능을 측정하는 프로세스입니다. 이 프로세스에는 프로덕션과 유사한 클라우드 환경을 프로비저닝하고, 로드 테스트 도구를 사용하여 로드를 생성하며, 지표를 분석하여 실제 워크로드 처리 능력을 평가하는 작업이 포함됩니다. 로드 테스트는 프로덕션 데이터의 통합 또는 제거 버전(민감한 정보 또는 식별 정보 제거)을 사용하여 실행되어야 합니다. 전송 파이프라인의 일부로 로드 테스트를 자동으로 수행

하고 결과를 미리 정의된 KPI 및 임곗값과 비교합니다. 이 프로세스를 통해 필요한 성능을 계속해서 달성할 수 있습니다.

구현 단계

- 테스트 목표 정의: 평가하려는 워크로드의 성능 측면(예: 처리량 및 응답 시간)을 식별합니다.
- 테스트 도구 선택: 워크로드에 적합한 로드 테스트 도구를 선택하고 구성합니다.
- 환경 설정: 프로덕션 환경에 따라 테스트 환경을 설정합니다. AWS 서비스를 사용하면 프로덕션 규모의 환경을 실행하여 아키텍처를 테스트할 수 있습니다.
- 모니터링 구현: Amazon CloudWatch와 같은 모니터링 도구를 사용하여 아키텍처의 리소스 전반에서 지표를 수집합니다. 사용자 지정 지표를 수집하고 게시할 수도 있습니다.
- 시나리오 정의: 로드 테스트 시나리오 및 파라미터(테스트 기간 및 사용자 수 등)를 정의합니다.
- 로드 테스트 수행: 대규모로 테스트 시나리오를 수행합니다. AWS 클라우드를 활용하여 워크로드를 테스트하여 확장에 실패하거나 비선형적인 방식으로 확장되는 부분을 발견하세요. 예를 들어, 스팟 인스턴스를 사용하여 저렴한 비용으로 로드를 생성하고 프로덕션 환경에서 발생하기 전에 병목 현상을 발견할 수 있습니다.
- 테스트 결과 분석: 결과를 분석하여 성능 병목 현상과 개선이 필요한 영역을 파악합니다.
- 조사 결과 문서화 및 공유: 조사 결과 및 권장 사항을 문서화하고 보고합니다. 이해관계자와 이 정보를 공유하면 이해관계자가 성능 최적화 전략과 관련하여 정보에 입각한 결정을 내리는 데 도움이 됩니다.
- 지속적인 반복: 로드 테스트는 정기적으로, 특히 시스템 변경 또는 업데이트 후에 수행해야 합니다.

리소스

관련 문서:

- Amazon CloudWatch RUM
- Amazon CloudWatch Synthetics
- Distributed Load Testing on AWS

관련 비디오:

- AWS Summit ANZ 2023: Accelerate with confidence through AWS Distributed Load Testing
- AWS re:Invent 2022 - Scaling on AWS for your first 10 million users

- Solving with AWS Solutions: Distributed Load Testing

- AWS re:Invent 2021 - Optimize applications through end user insights with Amazon CloudWatch RUM
- Demo of Amazon CloudWatch Synthetics

관련 예제:

- Distributed Load Testing on AWS

PERF05-BP05 자동화를 사용하여 성능 관련 문제 사전 해결

핵심 성과 지표(KPI)를 모니터링 및 경보 시스템과 함께 사용하여 성능 관련 문제를 선제적으로 해결합니다.

일반적인 안티 패턴:

- 워크로드에 대한 운영 변경을 수행할 수 있는 기능을 운영 직원에게만 허용합니다.
- 사전 조치 없이 모든 경보를 운영 팀으로 필터링합니다.

이 모범 사례 확립의 이점: 경보 작업을 사전에 해결하면 지원 직원이 자동으로 실행할 수 없는 항목에 집중할 수 있습니다. 이를 통해 운영 담당자는 모든 알람을 처리하는 데 부담을 느끼지 않고 중요한 경보에만 집중할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

경보를 사용하여 가능한 경우 문제를 해결하는 자동화 작업을 트리거합니다. 자동 대응이 불가능한 경우 대응을 수행할 수 있는 담당자에게 경보를 에스컬레이션합니다. 예를 들어 필요한 핵심 성과 지표(KPI) 값을 예측하고 해당 값이 특정 임곗값을 초과하는 경우 경보를 생성할 수 있는 시스템이나, KPI가 필요한 값의 범위를 벗어나는 경우 배포를 자동으로 중지하거나 롤백할 수 있는 도구가 있습니다.

워크로드가 실행 중일 때 성능을 확인할 수 있는 프로세스를 구현합니다. 워크로드가 최적의 상태로 작동하고 있는지를 확인할 수 있도록 성능 기대치 관련 기준을 설정하고 모니터링 대시보드를 구축합니다.

구현 단계

- 수정 워크플로 식별: 자동으로 해결할 수 있는 성능 문제를 식별하고 이해합니다. Amazon CloudWatch 또는 AWS X-Ray와 같은 AWS 모니터링 솔루션을 사용하여 문제의 근본 원인을 더 잘 파악할 수 있습니다.
- 자동화 프로세스 정의: 문제를 자동으로 해결하는 데 사용할 수 있는 단계별 수정 프로세스를 만듭니다.
- 시작 이벤트 구성: 수정 프로세스를 자동으로 시작하도록 이벤트를 구성합니다. 예를 들어 CPU 사용률이 특정 임곗값에 도달하면 인스턴스를 자동으로 다시 시작하도록 트리거를 정의할 수 있습니다.
- 수정 자동화: AWS 서비스 및 기술을 사용하여 수정 프로세스를 자동화합니다. 예를 들어, AWS Systems Manager Automation에서는 수정 프로세스를 자동화할 수 있는 안전하고 확장 가능한 방법을 제공합니다. 문제를 성공적으로 해결하지 못한 경우 자체 복구 논리를 사용하여 변경 내용을 되돌립니다.
- 워크플로 테스트: 사전 프로덕션 환경에서 자동화된 수정 프로세스를 테스트합니다.
- 워크플로우 구현: 프로덕션 환경에서 자동화된 문제 해결을 구현합니다.
- 플레이북 개발: 시작 이벤트, 수정 논리 및 실행된 조치를 포함하여 수정 계획의 단계를 설명하는 플레이북을 개발하고 문서화합니다. 이해관계자가 자동화된 수정 이벤트에 효과적으로 대응할 수 있도록 교육해야 합니다.
- 검토 및 개선: 자동화된 수정 워크플로의 효과를 정기적으로 평가합니다. 필요한 경우 시작 이벤트 및 수정 논리를 조정합니다.

리소스

관련 문서:

- CloudWatch 설명서
- 모니터링, 로깅 및 성능 AWS Partner Network 파트너
- X-Ray 설명서
- Using Alarms and Alarm Actions in CloudWatch
- Build a Cloud Automation Practice for Operational Excellence: Best Practices from AWS Managed Services
- Automate your Amazon Redshift performance tuning with automatic table optimization

관련 비디오:

- AWS re:Invent 2023 - Strategies for automated scaling, remediation, and smart self-healing
- AWS re:Invent 2023 - [LAUNCH] Application monitoring for modern workloads
- AWS re:Invent 2023 - Implementing application observability
- AWS re:Invent 2021 - Intelligently automating cloud operations
- AWS re:Invent 2022 - Setting up controls at scale in your AWS environment
- AWS re:Invent 2022 - Automating patch management and compliance using AWS
- AWS re:Invent 2022 - How Amazon uses better metrics for improved website performance
- AWS re:Invent 2023 - Take a load off: Diagnose & resolve performance issues with Amazon RDS
- AWS re:Invent 2021 -{New Launch} Automatically detect and resolve issues with Amazon DevOps Guru
- AWS re:Invent 2023 - Centralize your operations

관련 예제:

- CloudWatch Logs Customize Alarms

PERF05-BP06 워크로드 및 서비스 최신 상태 유지

새로운 클라우드 서비스 및 기능에 대한 최신 정보를 파악하여 효율적인 기능을 채택하고 문제를 제거하며 워크로드의 전반적인 성능 효율성을 개선할 수 있습니다.

일반적인 안티 패턴:

- 시간이 지나면 현재 아키텍처가 정적 아키텍처가 되고 업데이트되지 않는다고 가정합니다.
- 업데이트된 소프트웨어 및 패키지가 워크로드와 호환되는지 평가하는 시스템 또는 정기적인 주기가 없습니다.

이 모범 사례 확립의 이점: 새로운 서비스 및 제품에 대한 최신 정보를 파악하는 프로세스를 구축하면 새로운 기능을 채택하고 문제를 해결하며 워크로드 성능을 개선할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

새로운 서비스, 설계 패턴 및 제품 기능이 제공되면 성능을 개선할 방법을 평가합니다. 평가, 내부 논의 또는 외부 분석을 통해 워크로드의 효율성을 높이고 성능을 개선할 수 있는 방법을 결정합니다. 워크로

드 관련 업데이트, 새 기능 및 서비스를 평가하는 프로세스를 정의합니다. 새 기술을 사용하는 개념 증명 작성, 내부 그룹과의 상담 등을 그 예로 들 수 있습니다. 새 아이디어나 서비스를 시도할 때는 성능 테스트를 실행하여 해당 아이디어나 서비스가 워크로드의 성능에 주는 영향을 측정합니다.

구현 단계

- 인벤토리 작업 수행: 워크로드 소프트웨어 및 아키텍처를 조사하여 업데이트하는 데 필요한 구성 요소를 식별합니다.
- 업데이트 소스 식별: 워크로드 구성 요소 관련 뉴스 및 업데이트 소스를 파악합니다. 예를 들어, AWS의 새로운 소식 블로그 구독하여 워크로드 구성 요소와 일치하는 제품을 구입할 수 있습니다. RSS 피드를 구독하거나 이메일 구독을 관리할 수 있습니다.
- 업데이트 일정 정의: 워크로드에 대한 새로운 서비스 및 기능을 평가할 일정을 정의할 수 있습니다.
  - AWS Systems Manager Inventory를 사용하여 Amazon EC2 인스턴스에서 운영 체제(OS), 애플리케이션, 인스턴스 메타데이터를 수집하고 소프트웨어를 실행 중인 인스턴스, 소프트웨어 정책에 필요한 구성, 업데이트해야 할 인스턴스를 신속하게 파악합니다.
- 새 업데이트 평가: 워크로드 구성 요소의 업데이트 방법을 파악합니다. 클라우드의 민첩성을 활용하여 새로운 기능이 워크로드를 어떻게 개선하여 성능 효율성을 높일 수 있는지 빠르게 테스트합니다.
- 자동화 사용: 업데이트 프로세스에 자동화를 사용하여 새 기능 배포에 필요한 작업 수준을 줄이고 수동 프로세스로 인한 오류를 제한합니다.
  - CI/CD를 사용하면 클라우드 애플리케이션과 관련된 AMI, 컨테이너 이미지 및 기타 아티팩트를 자동으로 업데이트할 수 있습니다.
  - AWS Systems Manager Patch Manager와 같은 도구를 사용하여 시스템 업데이트 프로세스를 자동화하고 AWS Systems Manager Maintenance Windows를 사용하여 활동을 예약할 수 있습니다.
- 프로세스 문서화: 업데이트 및 새로운 서비스를 평가하기 위한 프로세스를 문서화합니다. 업데이트 및 새로운 서비스를 연구, 테스트, 실험 및 검증하는 데 필요한 시간과 공간을 담당자에게 제공합니다. 문서화된 비즈니스 요구 사항 및 KPI를 다시 참조하여 비즈니스에 긍정적인 영향을 줄 업데이트에 대한 우선순위를 지정할 수 있습니다.

리소스

관련 문서:

- AWS 블로그
- AWS의 새로운 소식
- Implementing up-to-date images with automated EC2 Image Builder pipelines

관련 비디오:

- AWS re:Inforce 2022 - Automating patch management and compliance using AWS
- All Things Patch: AWS Systems Manager | AWS Events

관련 예제:

- Inventory and Patch Management
- One Observability 워크숍

PERF05-BP07 정기적으로 지표 검토

정기적인 유지 관리의 일환으로 또는 이벤트나 인시던트 대응 과정에서 수집된 지표를 검토합니다. 이러한 검토를 수행하면 문제를 해결하는 데 반드시 필요했던 지표와 문제를 식별, 해결 또는 방지하는 데 도움이 되었던 지표(추적한 경우)를 추가로 파악할 수 있습니다.

일반적인 안티 패턴:

- 지표가 장기간 경보 상태로 유지되는 것을 허용합니다.
- 자동화 시스템으로 수행할 수 없는 경보를 생성합니다.

이 모범 사례 확립의 이점: 수집 중인 지표를 지속적으로 검토하여 문제가 올바르게 식별, 해결 또는 방지되는지 확인합니다. 지표를 장기간 경보 상태로 유지할 경우에도 지표가 부실해질 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

지표 수집 및 모니터링을 지속적으로 개선합니다. 인시던트나 이벤트 대응의 일환으로 문제를 해결하는 데 도움이 되었던 지표와 현재는 추적 중이지 않지만 도움이 되었을 수 있는 지표를 평가합니다. 이 방법을 사용하여 수집한 지표의 품질을 개선하면 사후 인시던트를 예방하거나 더 빨리 해결할 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.

구현 단계

- 지표 정의: 응답 시간 및 리소스 사용률과 같은 지표 등 워크로드 목표에 맞춰 모니터링할 중요한 성과 지표를 정의합니다.
- 기준선 설정: 각 지표에 대한 기준과 원하는 값을 설정합니다. 기준은 편차 또는 이상을 식별하기 위한 기준을 제공해야 합니다.
- 주기 설정: 주기(예: 주별 또는 월별)를 설정하여 중요 지표를 검토합니다.
- 성능 문제 식별: 각 검토 과정에서 추세와 기준값과의 편차를 평가합니다. 성능 병목 현상 또는 이상 징후가 있는지 찾아봅니다. 식별된 문제의 경우 심층적인 근본 원인 분석을 수행하여 문제의 주요 원인을 파악합니다.
- 수정 조치 식별: 분석을 사용하여 수정 조치를 식별합니다. 여기에는 파라미터 조정, 버그 수정, 리소스 규모 조정이 포함될 수 있습니다.
- 조사 결과 문서화: 식별된 문제, 근본 원인 및 시정 조치를 포함하여 조사 결과를 문서화합니다.
- 반복 및 개선: 지표 검토 프로세스를 지속적으로 평가하고 개선합니다. 이전 검토에서 배운 내용을 활용하여 시간이 지남에 따라 프로세스를 개선합니다.

리소스

관련 문서:

- CloudWatch 설명서
- CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버에서 지표 및 로그 수집
- CloudWatch Metrics Insights를 사용하는 지표 쿼리
- 모니터링, 로깅 및 성능 AWS Partner Network 파트너
- X-Ray 설명서

관련 비디오:

- AWS re:Invent 2022 - Setting up controls at scale in your AWS environment
- AWS re:Invent 2022 - How Amazon uses better metrics for improved website performance
- AWS re:Invent 2023 - Building an effective observability strategy
- AWS Summit SF 2022 - Full-stack observability and application monitoring with AWS
- AWS re:Invent 2023 - Take a load off: Diagnose & resolve performance issues with Amazon RDS

관련 예제:

- Creating a dashboard with QuickSight
- CloudWatch Dashboards

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. 코드형 인프라를 처리하면 데이터를 사용해 아키텍처에 대한 사실에 입각한 결정을 내리면서 아키텍처를 빠르고 안전하게 개선할 수 있습니다. 활성 모니터링과 수동 모니터링을 적절히 조합하여 활용하면 시간이 지나도 아키텍처 성능이 저하되지 않습니다.

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

기여자

다음 개인과 조직이 이 문서에 기여했습니다.

- Sam Mokhtari, Amazon Web Services의 Senior Efficiency Lead Solutions Architect
- Josh Hart, Amazon Web Services의 Solutions Architect
- Richard Trabing, Amazon Web Services의 Solutions Architect
- Brett Looney, Amazon Web Services의 Principal Solutions Architect
- Nina Vogl, Amazon Web Services의 Principal Solutions Architect
- Eric Pullen, Amazon Web Services의 Solutions Architect
- Julien Lépine, Amazon Web Services의 Specialist SA Manager
- Ronnen Slasky, Amazon Web Services의 Solutions Architect

참조 자료

자세한 내용은 다음 출처를 참조하세요.

- AWS Well-Architected Framework
- AWS 아키텍처 센터

문서 수정

이 백서에 대한 업데이트 알림을 받으려면 RSS 피드를 구독하면 됩니다.

변경 사항	설명	날짜
사소한 모범 사례 업데이트	PERF03-BP04가 새 서비스 권장 사항으로 업데이트되었습니다.	2024년 11월 6일
모범 사례 지침 업데이트됨	전체 원칙에서 여러 개의 소규모 업데이트.	2024년 6월 27일
주요 업데이트 및 구조 조정	원칙은 8개에서 5개 모범 사례 영역으로 재편되었습니다. 콘텐츠는 5개 영역으로 통합되고 업데이트되었습니다. 새로운 모범 사례 영역은 아키텍처 선택, 컴퓨팅 및 하드웨어, 데이터 관리, 네트워킹 및 콘텐츠 전송, 프로세스 및 문화입니다.	2023년 10월 3일
마이너 업데이트	포용적이지 않은 표현을 삭제했습니다.	2023년 4월 13일
새 프레임워크 관련 업데이트	권장 가이드 및 새로운 모범 사례를 추가하여 모범 사례를 업데이트했습니다.	2023년 4월 10일
백서 업데이트	새로운 구현 가이드와 함께 모범 사례를 업데이트했습니다.	2022년 12월 15일
백서 업데이트	모범 사례를 확대하고 개선 계획을 추가했습니다.	2022년 10월 20일

마이너 업데이트	포용적이지 않은 표현을 삭제했습니다.	2022년 4월 22일
마이너 업데이트	링크를 업데이트했습니다.	2021년 3월 10일
마이너 업데이트	AWS Lambda 제한 시간이 900초로 변경되었고 Amazon Keyspaces(Apache Cassandra용) 이름이 수정되었습니다.	2020년 10월 5일
마이너 업데이트	연결되지 않는 링크를 수정했습니다.	2020년 7월 15일
새 프레임워크 관련 업데이트	중요한 콘텐츠 검토 및 업데이트	2020년 7월 8일
백서 업데이트	문법 문제에 대한 마이너 업데이트	2018년 7월 1일
백서 업데이트	AWS의 변경 사항을 반영하기 위해 백서를 새로 고침	2017년 11월 1일
최초 게시	성능 효율성 원칙 - AWS Well-Architected Framework를 게시했습니다.	2016년 11월 1일

고지 사항

고객은 본 문서의 정보를 독립적으로 평가할 책임이 있습니다. 본 문서는 (a) 정보 제공의 목적으로만 제공되고, (b) 사전 통지 없이 변경될 수 있는 현재 AWS 제품 및 관행을 나타내고, (c) AWS 및 그 계열사, 공급업체 또는 라이선스 제공자로부터 어떠한 약속이나 보증도 하지 않습니다. AWS 제품 또는 서비스는 명시적이든 묵시적이든 어떠한 종류의 보증, 진술 또는 조건 없이 '있는 그대로' 제공됩니다. 고객에 대한 AWS의 책임 및 채무는 AWS 계약에 준거합니다. 본 문서는 AWS와 고객 간의 어떠한 계약도 구성하지 않으며 이를 변경하지도 않습니다.

© 2023 Amazon Web Services, Inc. 또는 계열사. All rights reserved.

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조서의 AWS 용어집을 참조하십시오.