에서 Microsoft 워크로드 비용 최적화 AWS

AWS 권장 가이드

AWS 권장 가이드: 에서 Microsoft 워크로드 비용 최적화 AWS

Table of Contents

에서 Microsoft 워크로드 비용 최적화 AWS

Bill Pfeiffer, Chase Lindeman 및 Kevin Sookhan, Amazon Web Services(AWS)

2024년 10월(문서 기록)

개요

이 가이드에서는 Microsoft 워크로드의 비용을 최적화하는 데 도움이 되는 권장 사항, 모범 사례 및 전략을 제공합니다 AWS. 또한이 가이드에는 비즈니스 목표에 맞는 비용 효율적이고 성능이 뛰어난 워크로드를 구축하고 자동화하는 데 도움이 되는 기본 AWS 지식, 비용 최적화 기법 및 참조 아키텍처가 포함되어 있습니다. 종합적으로이 지침을 Microsoft on AWS Cost Optimization(MACO)이라고 합니다. MACO 지침은 업계 전문가가 개발했으며 실제 시나리오를 기반으로 합니다.

이 가이드에서는 다음 Microsoft 워크로드를 다룹니다.

- Amazon Elastic Compute Cloud의 Windows(Amazon EC2)
- SQL Server
- 컨테이너
- 스토리지
- Active Directory
- .NET

대상

이 가이드는 아키텍트, 엔지니어, 관리자, 이사, CTOs, 기술 의사 결정자 및 AWS 파트너를 대상으로 합니다. 에 대한 이전 경험과 AWS 결제, Microsoft 기술 및 AWS 시스템 관리에 대한 기본적인 이해는 유용하지만 필수는 아닙니다.

이 설명서의 사용법

이 가이드를 사용하여 클라우드로의 MACO 여정을 계획하고 구현할 수 있습니다. Microsoft 워크로드 비용을 최적화하기 위한 옵션과 접근 방식을 포괄적으로 이해하려면이 가이드를 처음부터 끝까지 읽어보는 것이 좋습니다 AWS. 조직의 요구 사항에 따라 다음 워크로드 섹션을 검토할 수 있습니다.

- Amazon EC2의 Windows

- SQL Server
- 컨테이너
- 스토리지
- Active Directory
- .NET

> **Important**
>
> 이 가이드에 제공된 코드 샘플은 데모용입니다. 개발 환경에서 모든 코드를 테스트한 후 프로덕션 환경에서 사용하는 것이 좋습니다. 코드를 구현하기 전에 코드를 소량으로 테스트한 다음를 사용하여 코드로 인한 비용 변경 사항을 검토하는 것이 좋습니다AWS Cost Explorer. 이렇게 하면 나중에 문제가 될 수 있는 엣지 사례 및 기타 문제를 해결하는 데 도움이 될 수 있습니다.

> **Important**
>
> 이 가이드의 요금 예제는 게시 시점의 요금을 기반으로 합니다. 요금은 변경될 수 있습니다. 또한 비용은 , AWS 리전 AWS 서비스 할당량 및 클라우드 환경과 관련된 기타 요인에 따라 달라질 수 있습니다.

목표 비즈니스 성과

이 가이드는 여러분과 조직이 다음과 같은 비즈니스 성과를 달성하는 데 도움이 될 수 있습니다.

- AWS 최적화 및 라이선스 평가(AWS OLA)를 사용하여 리소스 사용률, 타사 라이선스 및 애플리케이션 종속성을 기반으로 현재 온프레미스 및 클라우드 환경을 평가하고 최적화하는 방법을 알아봅니다.
- Microsoft 워크로드용 AWS 현대화 계산기를 사용하여 비용 최적화를 위한 비즈니스 사례를 개발합니다.
- Amazon Elastic Compute Cloud(Amazon EC2), SQL Server, 컨테이너, 스토리지, Active Directory 및 .NET의 Windows용 워크로드를 포함하여 특정 Microsoft 워크로드에 대한 비용을 최적화합니다.

비용 최적화 여정

클라우드 마이그레이션 여정의 범위, 타이밍 및 특정 경로는 비즈니스 목표, 기술 요구 사항 및 기타 요인에 따라 달라집니다. 이 섹션에서는를 사용하여 Cloud Financial Management에 초점을 맞추고 MACO 권장 사항 및 모범 사례를 준수하는 클라우드 AWS 마이그레이션 여정의 예를 제공합니다. 이 예제를 사용하여 Microsoft 워크로드에 대한 클라우드 마이그레이션 여정을 설계하는 방법을 이해할 수 있습니다.

다음 상위 수준 작업은 조직이 MACO 권장 사항 및 모범 사례를 구현하기 위해 취할 수 있는 접근 방식을 보여줍니다.

- 태그 지정 전략을 수립하고 사용자 정의 비용 할당 태그를 활성화합니다. 자세한 내용은 리소스 태그 지정을 위한 AWS 백서 모범 사례를 참조하세요. AWS
- 애플리케이션, 팀 또는 부서를 기반으로 예산을 정의합니다. 자세한 내용은 AWS Billing and Cost Management 사용 설명서의 를 사용하여 비용 관리를 참조하세요 AWS Budgets.
- AWS 최적화 및 라이선스 평가(AWS OLA)를 수행하여 비용 절감을 가속화합니다. 자세한 내용은 AWS 설명서의 AWS 최적화 및 라이선스 평가를 참조하세요.
- 를 사용하여 Windows 및 SQL Server 워크로드에 대한 BYOL(Bring Your Own License)을 제공합니다 Amazon Elastic Compute Cloud Dedicated Hosts. 자세한 내용은 이 설명서의 Windows 및 SQL Server 워크로드에 대한 라이선스 가져오기 단원을 참조하십시오.
- 에서 SQL Server 라이선스를 최적화합니다 AWS. 자세한 내용은 이 설명서의 SQL Server 라이선스 이해 단원을 참조하십시오.
- Windows 워크로드에 적합한 인스턴스 유형을 선택합니다. 자세한 내용은 이 설명서의 Windows 워크로드에 적합한 인스턴스 유형 선택 단원을 참조하십시오.
- SQL 워크로드에 적합한 인스턴스 유형을 선택합니다. 자세한 내용은 이 설명서의 SQL Server 워크로드에 적합한 EC2 인스턴스 선택 단원을 참조하십시오.
- Amazon Elastic Block Store(Amazon EBS)를 gp2에서 gp3로 마이그레이션합니다. 자세한 내용은 이 설명서의 Amazon EBS 볼륨을 gp2에서 gp3로 마이그레이션 단원을 참조하십시오.
- EC2 Instance Scheduler를 켜서 워크로드를 제어합니다 AWS. 자세한 내용은 이 설명서의 중지 및 시작 일정 자동화 단원을 참조하십시오.
- SQL Server Developer Edition을 사용하여 비프로덕션 워크로드에 대한 SQL Server 비용을 제거합니다. 자세한 내용은 이 설명서의 SQL Server 개발자 에디션 평가 단원을 참조하십시오.
- 개발 및 테스트 워크로드에는 Amazon FSx for Windows File Server용 단일 가용 영역을 사용합니다. 자세한 내용은 이 설명서의 단일 가용 영역 사용 단원을 참조하십시오.

- 를 사용하여 Windows 워크로드를 조정합니다 AWS Compute Optimizer. 자세한 내용은 이 설명서의 적절한 크기의 Windows 워크로드 단원을 참조하십시오.
- 절감형 플랜을 사용하여 Amazon EC2의 Windows에서 지출을 최적화합니다. Savings Plans 자세한 내용은 이 설명서의 Amazon EC2에서 Windows에 대한 지출 최적화 단원을 참조하십시오.
- FSx for Windows File Server에서 데이터 중복 제거를 활성화합니다. 자세한 내용은 이 설명서의 Amazon FSx에서 데이터 중복 제거 활성화 단원을 참조하십시오.
- FSx for Windows File Server의 파일 시스템에 데이터 샤딩을 사용합니다. 자세한 내용은 이 설명서의 FSx for Windows File Server의 데이터 샤딩 이해 단원을 참조하십시오.
- SQL Server 백업 전략을 최적화합니다. 자세한 내용은 이 설명서의 SQL Server 백업 전략 최적화 단원을 참조하십시오.
- 정적 .NET 프레임워크 앱이 동적 조정을 지원하도록 합니다. 자세한 내용은이 안내서정적 .NET Framework 앱에 대한 동적 조정 지원의를 참조하세요.
- 서버리스 .NET 마이크로서비스를 사용합니다. 자세한 내용은 이 설명서의 서버리스 .NET 고려 단원을 참조하십시오.
- Windows 앱을 컨테이너로 이동합니다. 자세한 내용은 이 설명서의 .NET 앱 컨테이너화 단원을 참조하십시오.
- AWS Compute Optimizer를 사용하여 Amazon Elastic Container Service(Amazon ECS) AWS Fargate 용에서 실행되는 Windows 컨테이너의 크기를 조정합니다. 자세한 내용은 이 설명서의 Compute Optimizer 활성화 단원을 참조하십시오.
- 최신 .NET으로 리팩터링하고 Linux로 이동합니다. 자세한 내용은 이 설명서의 최신 .NET으로 리팩터링하고 Linux로 전환 단원을 참조하십시오.
- Graviton 인스턴스 및 컨테이너를 활용합니다. 자세한 내용은 이 설명서의 Graviton 인스턴스 및 컨테이너 사용 단원을 참조하십시오.
- SQL Server 데이터베이스를 현대화합니다. 자세한 내용은 이 설명서의 SQL Server 데이터베이스 현대화 단원을 참조하십시오.
- Active Directory 인프라를 설계합니다. 자세한 내용은 이 설명서의 Active Directory 단원을 참조하십시오.

를 사용한 Cloud Financial Management에 초점을 맞춘 고객 여정에 대한 자세한 내용은 AWS 백서 Cloud Financial Management 기능을 AWS참조하세요.

비용 최적화를 위한 주요 권장 사항

개요

비용 최적화는 AWS Well-Architected Framework의 원칙 중 하나이며 클라우드 마이그레이션 계획에서 중요한 역할을 합니다. 이 가이드 전체에서 비용 최적화에 대한 권장 사항을 찾을 수 있지만이 섹션에서는 가장 큰 영향을 미치는 권장 사항을 설명합니다. 이러한 권장 사항을 빠르게 구현할 수 있으며 조직에 상당한 영향을 미칠 수 있습니다. 이러한 권장 사항은 전체 비용 최적화 노력의 토대를 마련하는 데 도움이 될 수 있습니다.

권장 사항

다음 표에는 가장 큰 영향을 미치는 비용 최적화에 대한 권장 사항이 나열되어 있습니다. "구현하기 어려움" 열은 구현하기 가장 쉬운 것(1)부터 구현하기 가장 어려운 것(5)까지의 규모를 기준으로 각 최적화를 평가합니다. "예상 절감액" 열에는 조직에서 각 권장 최적화에 대해 절감할 수 있는 금액의 백분율 기반 추정치가 표시됩니다.

최적화	구현하기 어려움	예상 절감액
적절한 크기의 Windows 워크로드	3	25%
Windows 및 SQL Server 워크로드에 대한 라이선스 가져오기	3	30%
SQL Server 개발자 에디션 평가	2	20%
SQL Server 라이선스 이해	2	최대 50%
중지 및 시작 일정 자동화	3	최대 40%
Windows 워크로드에 적합한 인스턴스 유형 선택	1	10~30%

최적화	구현하기 어려움	예상 절감액
최신 .NET으로 리팩터링하고 Linux로 전환	5	10~20%
Amazon EC2에서 Windows에 대한 지출 최적화	3	최대 20~40%
Amazon EBS 볼륨을 gp2에서 gp3로 마이그레이션	4	최대 20%

Important

위 표의 예상 절감액은 계정 내 전체 AWS 지출이 아닌 각 개별 기술 도메인에 적용됩니다. 예를 들어 잠재적 절감액을 변경할 수 있는 다양한 환경 유형 및 크기로 인스턴스 스케줄러를 구현할 수 있습니다. 이 추정치는 Amazon EC2 인스턴스 비용에 특히 적용되며 다른에 대한 전체 비용 절감을 의미하지 않습니다 AWS 서비스. 이러한 추정치는 보장이 아닌 게이지로 제공됩니다.

MACO 전문가는 비용 최적화에 대해 더 자세히 설명할 수 있습니다. 사용 사례에 대한 심층 분석을 위한 회의를 설정하려면 계정 팀에 문의하거나 optimize-microsoft@amazon.com으로 이메일을 보내세요.

AWS 최적화 및 라이선스 평가

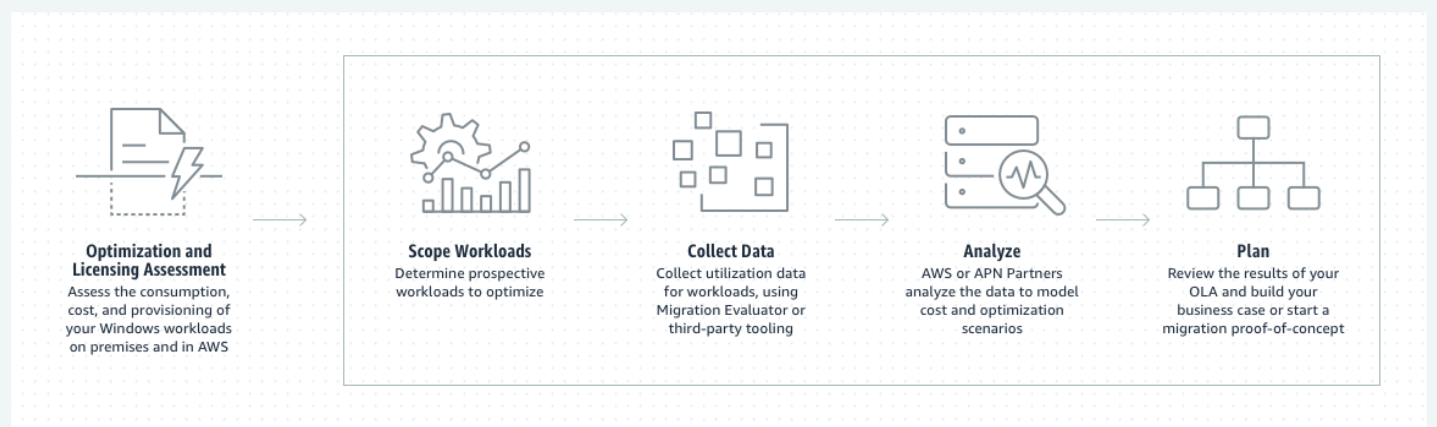
개요

AWS 최적화 및 라이선스 평가(AWS OLA)는 리소스 사용률, 타사 라이선스 및 애플리케이션 종속성을 기반으로 현재 온프레미스 및 기존 클라우드 환경을 평가하고 최적화하는 데 도움이 될 수 있습니다. AWS OLA를 사용하면 기존 Microsoft 워크로드로 마이그레이션 AWS 하거나 기존 Microsoft 워크로드를 평가할 때 비용을 절감할 수 있는 마이그레이션 및 라이선스 전략을 구축할 수 있습니다 AWS. 또한 AWS OLA는 다음을 달성하는 데 도움이 될 수 있습니다.

- 기존 배포, 애플리케이션 성능 및 계약을 이해합니다.
- 리소스의 크기를 조정합니다.
- 에 대한 로드맵을 개발합니다 AWS 클라우드.
- 기존 투자를 사용하고 사용한 만큼만 비용을 지불하여 비용을 줄이거나 제거합니다.

AWS OLA를 비용 최적화 여정의 첫 단계로 만드는 것이 좋습니다. 로 작업 AWS Partner Network 하여 AWS OLA를 완료할 수 있습니다. 이를 통해 평가 데이터를 수집하고 라이선스 및 인스턴스 비용을 최적화하기 위한 권장 사항을 제공할 수 있습니다.

다음 다이어그램은 평가 프로세스에 대한 개요를 제공합니다.



평가 옵션

에서 Microsoft 워크로드에 대한 AWS두 가지 AWS OLA 옵션 중에서 선택할 수 있습니다.

- Lite 버전 -이 사용 사례에서는 모든 워크로드가 VMware에 있습니다. AWS 에 RVTools의 출력을 제공할 수 있습니다. 그런 다음 1~5일의 처리 시간을 제공할 AWS 수 있습니다. 이 접근 방식은 VMware vCenter에서 직접 가져온 point-in-time 정보를 사용하여 크기 조정 권장 사항을 개발하고 온디맨드 요금 옵션을 제공합니다.
- 전체 버전 -이 사용 사례에서는 서로 다른 클라우드 공급자, 물리적 서버 및 가상 서버에서 실행되는 혼합 환경이 있습니다.는 운영 체제 에이전트를 AWS 사용하여 14~30일의 사용 데이터를 수집합니다. 이를 통해 AWS 는 애플리케이션 사용 패턴을 기반으로 정보에 입각한 인스턴스 크기 조정 결정을 내릴 수 있습니다.는 Cloudamize와 같은 여러 타사 도구를 AWS 사용하여 분석을 완료합니다. AWS 는와 협력하여 요금 모델 및 다양한 아키텍처에 고려되는 여러 요금 옵션을 사용하여 최종 총 소유 비용(TCO) 평가를 AWS Partner Network 제공합니다.

전체 평가

전체 AWS OLA 평가는 1시간 전화 통화로 시작됩니다. 이 호출 중에는 마이그레이션을 지원하는 가장 최적의 AWS 인프라를 결정하고, 데이터 수집 방법을 선택하고, 완료 일정을 설정하는 데 AWS 도움이 됩니다. 조직에서 검색 도구를 구현하는 것은 데이터 수집 방법, 조직의 크기, 조직이 서버 플릿을 관리하는 데 사용하는 도구에 따라 달라집니다. 일반적으로 사용량 데이터를 수집하는 데 2주가 걸립니다.

전체 AWS OLA 프로세스는 30~45일이 소요되며 다음 단계로 구성됩니다.

- 워크로드 범위 지정
- 데이터 수집
- 데이터 분석
- 다음 단계 계획

워크로드 범위 지정

먼저 사용자 및 팀과 AWS 협력하여 평가 범위를 결정합니다. 이는 일반적으로 환경 유형(예: 비프로덕션 및 프로덕션)별로 구분됩니다. 범위에는 워크로드의 위치가 포함됩니다. 마이그레이션 중인 워크로드 AWS, 이미 실행 중인 워크로드 AWS (예: Amazon EC2용 AWS OLA) 또는 다른 클라우드 공급자에서 실행 중인 워크로드가 여기에 해당합니다.

데이터 수집

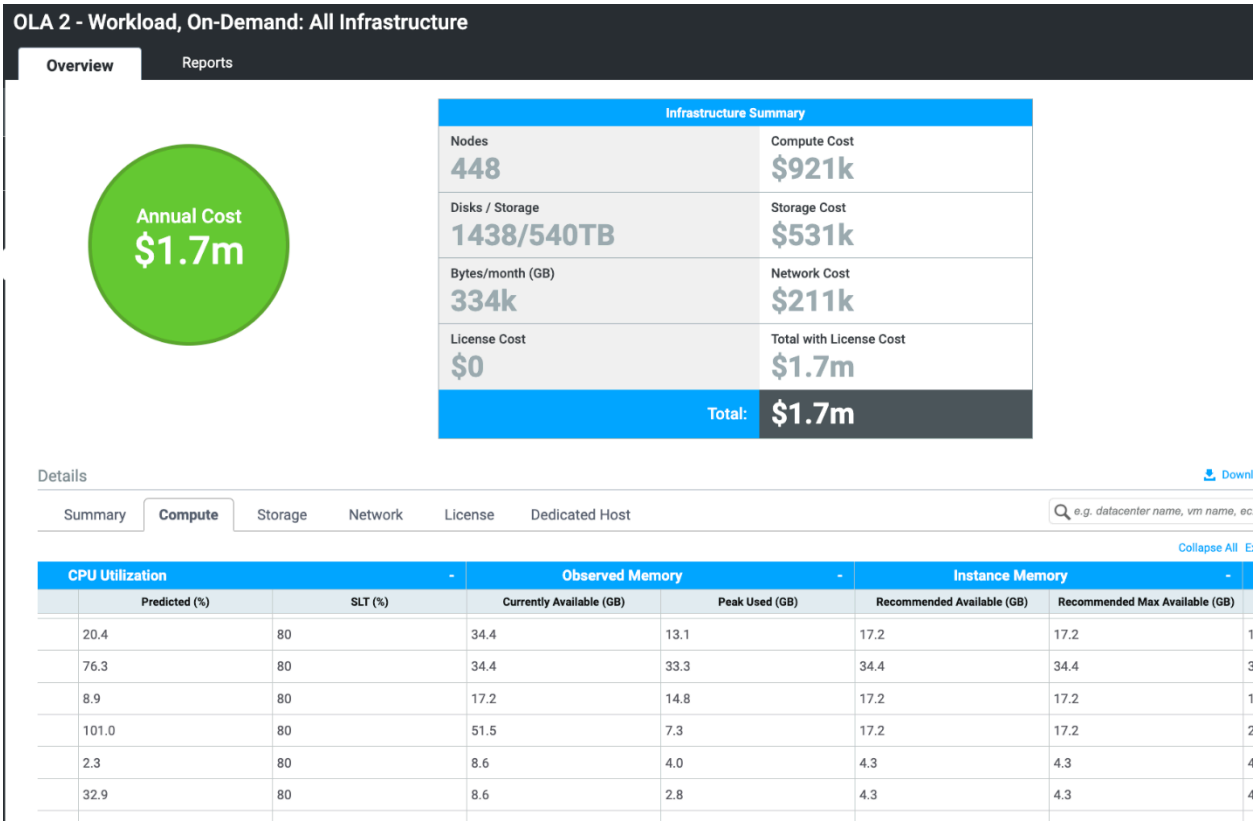
다음으로, 리소스 검색에 도움이 되는 도구를 AWS 배포하고 서버에서 성능 데이터를 수집합니다. 이 도구는 네 가지 배포 옵션으로 제공됩니다.

- 하이퍼바이저를 쿼리할 수 있는 도구(VMware vCenter 또는 Hyper-V 자격 증명만 필요)
- 물리적 또는 가상 머신에 배포할 수 있는 에이전트
- 환경 및 운영 체제에 따라 SSH, Windows 원격 관리(WinRM) 또는 Windows 관리 계측(WMI)을 사용하여 에이전트 없는 검색
- 플랫 파일 데이터 수집 및 분석

도구 배포의 경우 각 옵션을 혼합 및 매칭하고 결과를 통합할 수 있습니다. 선택한 옵션이 IT 리소스를 제한하지 않도록 하는 것이 중요합니다.는 평가 프로세스를 최대한 턴키로 만들기 위해 AWS 노력합니다. AWS OLA 팀과 Microsoft 전문 솔루션 아키텍트는 설정을 지원하기 위한 간단한 전화 통화 외에도 총 소유 비용(TCO) 분석 및 검토를 위한 권장 사항을 준비합니다.

데이터 수집은 일반적으로 CPU 사용률, RAM 사용률, 스토리지 처리량, IOPS 및 네트워크 처리량이 분석되는 데 2~3주가 걸립니다. 이상적으로는이 수집이 한 달의 피크 시간대(예: end-of-month 재무 보고)에 이루어집니다. AWS 는 적절한 크기의 AWS 인스턴스가 무엇이어야 하는지에 대한 좋은 통계 샘플을 제공하면서 성능을 온프레미스에서 사용할 수 있는 것을 초과할 수 있기 때문에 피크 사용량을 캡처하고자 합니다.는 사용률 지표를 다양한 프로세서 세대의 성능 휴리스틱과 AWS 병합하여 주어진 워크로드에 필요한 CPU 및 RAM의 양을 정확하게 목표로 합니다. 이러한 대상은 일반적으로 온프레미스에 할당된 것보다 적습니다. 이렇게 하면 인스턴스 크기의 컴퓨팅 비용이 절감될 뿐만 아니라 라이선스 비용도 최적화됩니다.

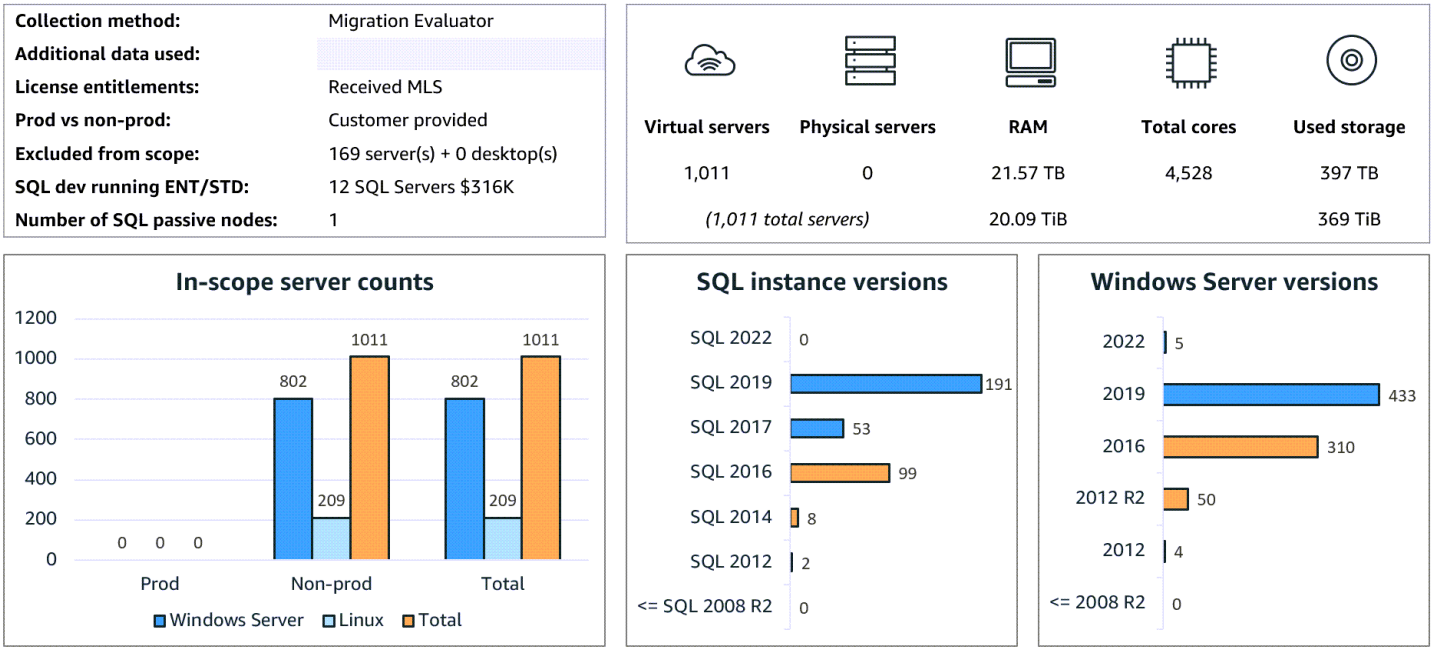
다음 대시보드 보기는 평가를 통해 캡처할 수 있는 인프라 비용의 예를 보여줍니다.



데이터 분석

AWS 는 데이터 수집이 완료된 후 요약 프레젠테이션을 제공합니다.는 데이터를 AWS 검토하고 결과를 요약한 다음 온프레미스 사용 및 클라우드 마이그레이션에 대한 권장 사항을 제시합니다. 통합 기회, 탄력성 향상(워크로드를 끄거나 계절별로 조정할 수 있는 경우), 오른쪽 SKU 기회(예: SQL Server Enterprise 에디션이 사용 중이지만 리소스 요구 사항 및 기능 사용량에 따라 SQL Server Standard 에디션이 적합함)를 검사하여 컴퓨팅 및 라이선스 비용을 줄일 수 있습니다. 코어에서 라이선스를 부여한 SQL Server와 같은 제품의 경우 워크로드를 더 비싼 컴퓨팅 인스턴스에 배치하는 것이 재정적으로 타당한 경우가 많습니다. 즉, RAM과 vCPU의 CPU 프로파일 및 비율이 순 효과를 제공하여 라이선스 포함 및 BYOL(Bring Your Own License) 사용 사례 모두에 대해 라이선스가 부여된 코어 수를 줄이는 경우입니다.

다음은 평가에서 수집한 데이터를 기반으로 한 분석의 예입니다.



일반적인 최적화 시나리오에는 AWS 리소스 최적화 기회와 타사 라이선스 절감을 모두 식별하는 것이 포함됩니다.

AWS 리소스 최적화 기회의 예:

- 사용량이 가장 많을 때는 과도하게 프로비저닝하지 마세요.
- 리소스를 과도하게 지정하거나 과소 사용하지 마세요.
- 인스턴스의 크기를 조정하고 최신 세대의 EC2 인스턴스로 마이그레이션합니다.
- 관리형 데이터베이스로 이동하여 운영 비용을 절감합니다.

타사 라이선스 절감의 예:

- 동일한 워크로드를 실행하는 데 필요한 코어를 줄입니다.
- 불필요한 SQL Server Enterprise Edition 및 추가 기능 팩을 제거합니다.
- 좀비 서버를 제거하고 오래된 하드웨어를 교체합니다.
- BYOL 및 라이선스 포함 옵션을 사용하여 향후 상용 계약을 줄일 수 있습니다.
- 오픈 소스 및 클라우드 네이티브 솔루션으로 현대화합니다.

다음 단계 계획

마지막으로, 수집된 성능 데이터를 AWS 사용하여 특정 워크로드 크기 및 비용을 추정합니다. AWS 는 범위가 지정된 환경에서 집계하여 정량적 분석을 제공할 수도 있습니다. 이를 통해 최적의 옵션이 온프레미스 새로 고침인지 마이그레이션인지 확인할 수 있습니다 AWS. AWS OLA 끝에 제공된 TCO 분석 요약(다음 예제 참조)을 사용하여 클라우드 경제 비즈니스 사례를 구축할 수 있습니다.

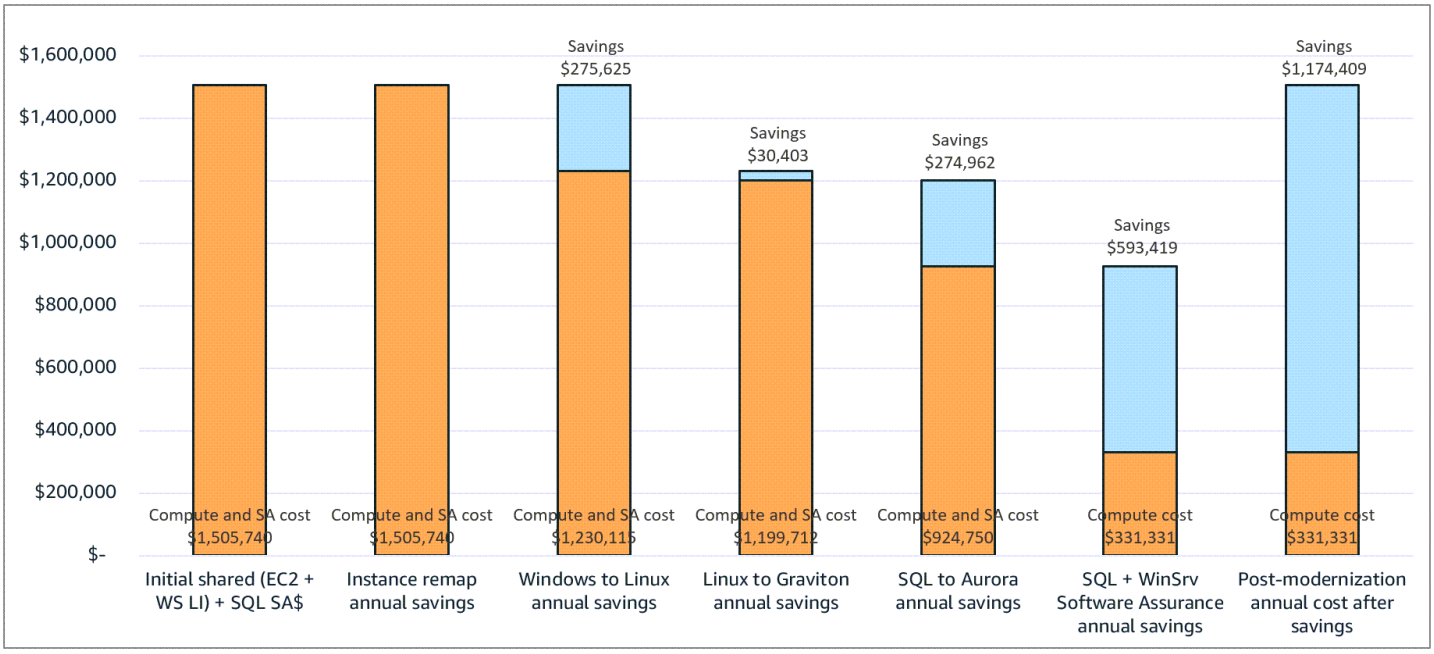
	Option 1: Amazon EC2 shared	**Option 1a: Amazon EC2 shared + power management**	**Option 2: Amazon EC2 mixed**	**Option 2a: Amazon EC2 mixed + power management**
Option details: compute	*100% Reserved Instances (RIs)*	*RIs + on-demand power management*	*100% Ris*	*RIs + on-demand power management*
Option details: Microsoft licenses	*WS LI and SQL BYOL*	*WS LI and SQL BYOL*	*WS BYOL or LI+SQL BYOL*	*WS BYOL or LI+SQL BYOL*
Compute costs[1]				
Year 1 compute cost	$414,546	$482,623	$504,019	$513,941
Year 1 vendor license included cost	$392,858	$244,415	$9,804	$4,783
	$807,404	**$727,038**	**$513,823**	**$518,724**
Total compute savings in year 1, compared to Option 1	—	*10% ($80,366)*	*36% ($293,581)*	*36% ($288,680)*
Storage and networking costs[2]				
Annual estimated storage cost	$336,494	$336,494	$336,494	$336,494
Annual estimated networking cost	$41,455	$41,455	$41,455	$41,455
	$377,949	**$377,949**	**$377,949**	**$377,949**
Microsoft license costs**				
WS/CIS annual Software Assurance (SA) + current SPLA/Subs cost	$0	$0	$0	$0
WS/CIS license + SA + SPLA/Subs true-up cost	$0	$0	$0	$0
SQL annual SA + current SPLA/Subs cost	$0	$0	$0	$0
SQL license SA + current SPLA/Subs true-up cost	$0	$0	$0	$0
	$0	**$0**	**$0**	**$0**
Total estimated costs	**$1,185,353**	**$1,104,987**	**$891,772**	**$896,673**
Annual TCO savings in year 1, compared to Option 1	—	*7% ($80,366)*	*25% ($293,581)*	*24% ($288,680)*

[1] Pricing model used: 3-year, no upfront RI
[2] Software Assurance and true-up costs provided by Microsoft

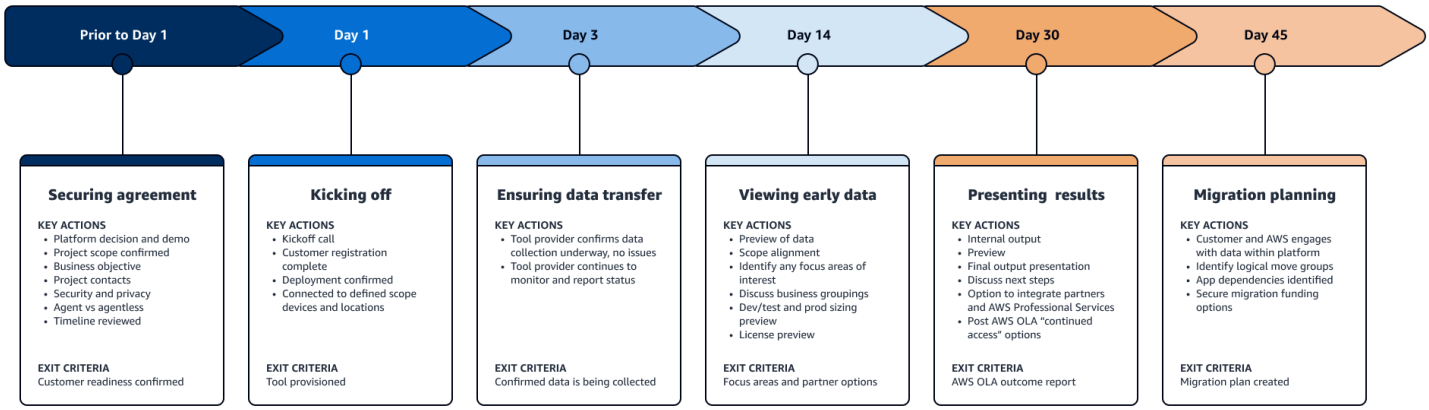
또한 AWS OLA는 다음과 같은 제안을 하여 현대화가 기존 워크로드에 미칠 수 있는 영향에 대한 인사이트를 제공합니다.

- Linux 운영 체제로 이동합니다.
- ARM 프로세서(AWS Graviton)에 대한 애플리케이션 지원을 추가합니다.
- SQL Server 워크로드를 Amazon Aurora로 이동합니다.
- Windows 및 SQL Server 워크로드를 오픈 소스 기술로 이동하여 소프트웨어 보증을 제거합니다.

다음 다이어그램은 Windows에서 Linux로 또는 SQL Server에서 Aurora로 이전하는 것과 같은 현대화 기술을 통해 달성할 수 있는 비용 절감 효과를 보여줍니다.



전체 AWS OLA 프로세스는 시작부터 완료까지 약 45일이 걸립니다. 다음 다이어그램은 타임라인의 예를 보여줍니다.



순수 VMware 환경이 있고 RVTools의 출력을 제공할 수 있는 경우이 타임라인을 영업일 기준 1주일로 줄일 수 있습니다. 또한는 CPU 평균, CPU 피크, RAM 평균 및 RAM 피크와 같은 자산 및 사용률 데이터가 포함된 플랫 파일을 분석할 AWS 수 있습니다.

평가 영향

평균 고객은 일반적으로 적절한 크기 조정 작업에서 20~30%의 비용 절감을 경험합니다. 올바른 크기 조정은 소스 워크로드를 사용량 데이터를 기반으로 가장 적합한 크기의 AWS 인스턴스와 일치시킵니다. 이러한 적절한 크기 조정은 AWS 환경의 월별 비용을 줄일 뿐만 아니라 조직의 다른 곳에서도 비용 절감을 가져오는 경우가 많습니다. 예를 들어 Windows 또는 SQL Server 라이선스 획득률이 20~30%

이면 Microsoft의 다음 트루업을 줄이거나 추가 line-of-business 애플리케이션에 대한 라이선스를 확보할 수 있습니다. SQL Server 워크로드의 통합 및 올바른 크기 조정은 일반적으로 가장 극적인 재정적 이득이 실현되는 경우입니다.

AWS 를 사용하면 시스템을 현대화 버킷으로 분류할 수 있습니다. 일부 시스템은 레거시 시스템이며 재정적으로 접촉할 수 없는 반면, 가장 큰 비용 절감이 실현되는 컨테이너 또는 서버리스 애플리케이션으로 현대화되는 시스템도 있습니다. AWS 팀과의 대화는 클라우드가 지원하는 것에 대한 일반적인 주제에서 특정 워크로드를 현대화하는 방법과 이유에 대한 보다 구체적인 논의로 이동합니다. AWS 또한 잠재적인 혁신 기회를 탐색하는 데 도움이 됩니다.

다음 단계

온프레미스 환경 또는에서 실행 중인 Microsoft 워크로드에 대한 비용 최적화 여정을 시작하는 경우 AWS 계정 팀에 AWS문의하여 AWS OLA를 요청하세요. AWS 팀 구성원은 질문에 답변하고 AWS OLA가 궁극적으로 사용자와 조직에 적합한 선택인지 결정하는 데 도움을 줄 수 있습니다. 또는 AWS OLA를 온라인으로 요청할 수 있습니다.

추가 리소스

- AWS 최적화 및 라이선스 평가(AWS 문서)
- AWS re:Invent 2022 - (ENT205)에서 AWS 비용을 절감하고 Microsoft 워크로드를 최적화하는 방법(YouTube)

Amazon EC2의 Windows

Amazon Elastic Compute Cloud(Amazon EC2)는 Windows 워크로드를 실행하는 데 적합한 매우 유연하고 확장 가능한 클라우드 컴퓨팅 플랫폼입니다. Amazon EC2를 사용하여의 안전하고 안정적이며 가용성과 적응력이 뛰어난 인프라에서 Windows Server 워크로드를 배포, 관리 및 확장할 수 있습니다 AWS 클라우드. Amazon EC2에서 Windows 워크로드를 실행할 때 얻을 수 있는 다음과 같은 주요 이점을 고려하세요.

- 확장성 - Amazon EC2를 사용하면 변화하는 요구 사항에 맞게 Windows 워크로드를 쉽게 확장할 수 있습니다. 새로운 EC2 인스턴스를 빠르게 생성하여 수요 증가를 처리하고 더 이상 필요하지 않을 때 인스턴스를 쉽게 종료할 수 있습니다. 실제로 사용하는 리소스에 대해서만 비용을 지불합니다.
- 유연성 - Amazon EC2의 Windows는 범용 인스턴스에서 메모리 또는 컴퓨팅 최적화 인스턴스에 이르기까지 다양한 워크로드 요구 사항에 맞게 설계된 다양한 인스턴스 유형을 지원합니다. 이러한 유연성을 통해 특정 Windows 기반 애플리케이션에 가장 적합한 인스턴스 유형을 선택하여 성능을 극대화하고 비용을 최소화할 수 있습니다.
- 보안 - 네트워크 방화벽, 데이터 암호화, 보안 액세스 제어를 포함하여 Windows 워크로드에 대한 여러 계층의 보안을 AWS 제공합니다. 즉, 보안 설정 및 구성을 완전히 제어하면서 애플리케이션과 데이터를 보호할 수 있습니다.
- 비용 효율성 pay-as-you-go 요금 모델을 사용하면 사용하는 리소스에 대해서만 비용을 지불할 수 있으므로 하드웨어 및 소프트웨어에 대한 선결제 투자가 필요하지 않습니다. 또한이 모델을 사용하면 비용을 최적화하고, 자본 지출을 줄이고, 운영 효율성을 높일 수 있습니다. 모든 규모의 비즈니스에 이상적인 요금 모델입니다.

가이드의이 섹션에서는 다음 주제를 다룹니다.

- 중지 및 시작 일정 자동화
- 적절한 크기의 Windows 워크로드
- Windows 워크로드에 적합한 인스턴스 유형 선택
- Windows 및 SQL Server 워크로드에 대한 라이선스 가져오기
- Amazon EC2에서 Windows에 대한 지출 최적화
- AWS 도구를 사용하여 비용 모니터링

중지 및 시작 일정 자동화

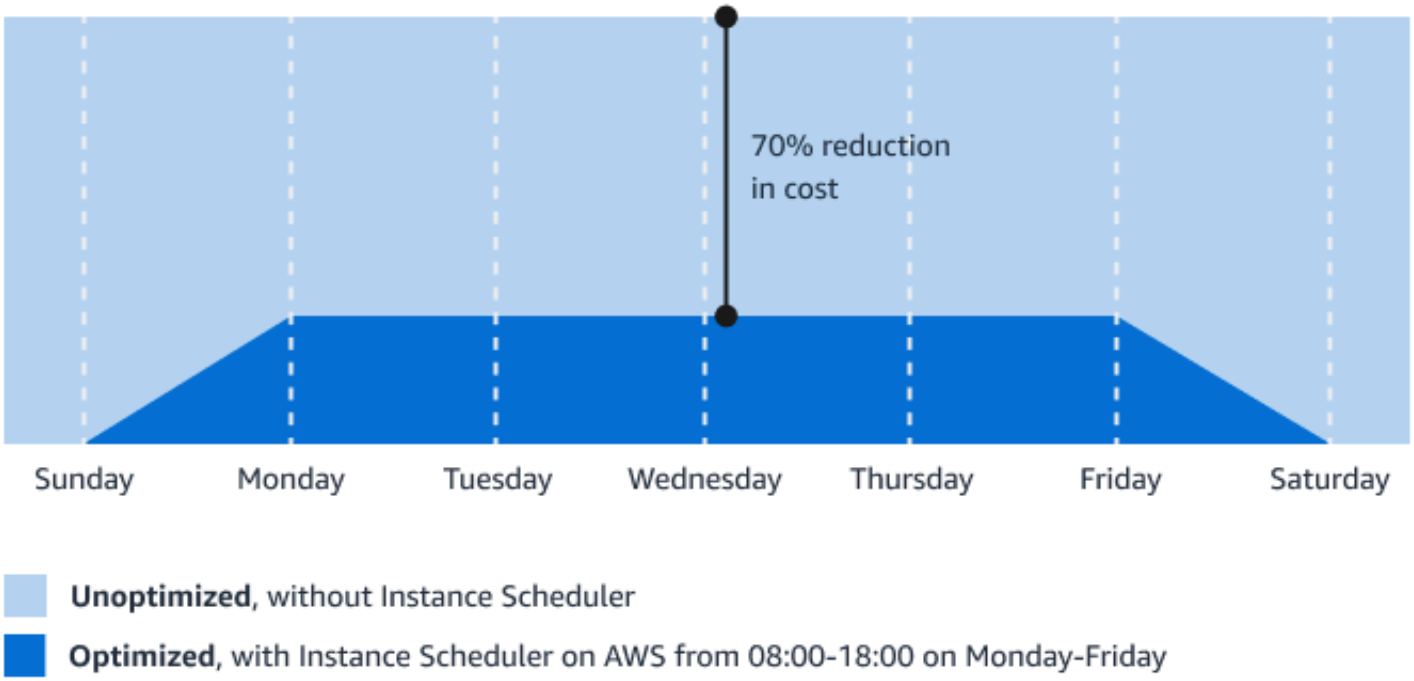
개요

의 인스턴스 스케줄러 AWS를 사용하면 Amazon EC2 및 Amazon Relational Database Service(RDS) 인스턴스의 시작 및 중지를 자동화하여 운영 비용을 절감할 수 있습니다. 모든 인스턴스를 최대 사용률로 계속 실행하면 사용되지 않는 리소스에 대한 비용을 지불하게 될 수 있습니다. 의 인스턴스 스케줄러 AWS 를 사용하면 업무 외 시간, 주말 또는 사용량이 적은 기타 기간과 같이 필요하지 않은 시간에 인스턴스를 끌 수 있습니다. 이로 인해 시간이 지남에 따라 상당한 비용 절감이 발생할 수 있습니다.

또한의 인스턴스 스케줄러는 교차 계정 인스턴스 예약, 자동 태그 지정, 명령줄 인터페이스 또는 AWS Systems Manager 유지 관리 기간을 사용하여 일정 또는 기간을 구성하는 기능 AWS 도 제공합니다. 이러한 기능을 통해 인스턴스를 보다 효과적이고 정확하게 관리하고 다양한 프로젝트 또는 팀에 비용을 할당할 수 있습니다.

사례 연구

프로덕션 환경에서의 인스턴스 스케줄러 AWS 를 사용하여 매일 업무 시간 외에 인스턴스를 자동으로 중지하는 회사의 예를 생각해 보세요. 이 회사는 모든 인스턴스를 최대 사용률로 실행 상태로 두면 정규 업무 시간 동안에만 필요한 인스턴스에 대해 최대 70%의 비용 절감을 달성할 수 있습니다. 다음 차트는 주간 사용률을 168시간에서 50시간으로 줄이는 방법을 보여줍니다.

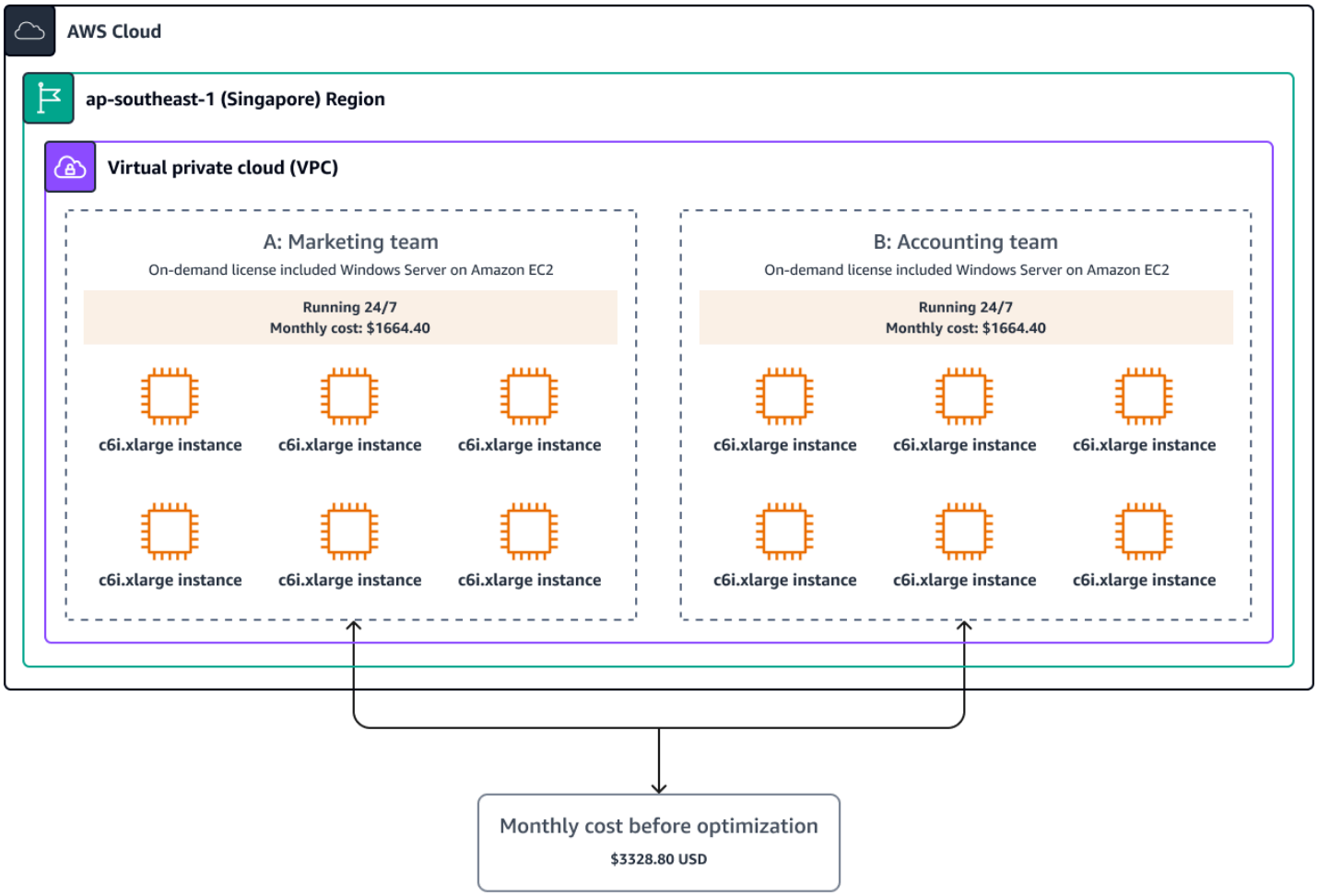


다른 예를 생각해 보세요. 전기 유틸리티 회사인 Jamaica Public Service Company Limited(JPS)는 데이터베이스를 Amazon RDS로 마이그레이션했습니다. 이제 JPS는 Amazon EC2를 사용하여 API 서비스를 호스팅하고 다른 애플리케이션을 실행합니다. JPS의 경우의 인스턴스 스케줄러가 비프로덕션 환경을 관리하는 주요 도구가 AWS 되었습니다. JPS는의 인스턴스 스케줄러 AWS 를 사용하여 개발 비용을 절감하고 팀 요구 사항 및 작업 일정에 따라 EC2 인스턴스를 관리했습니다. 이를 통해 JPS는 비용을 40% 절감할 수 있었습니다. 자세한 내용은 AWS 사례 연구 Jamaica Public Service가 클라우드로 효율적으로 마이그레이션하고 AWS 인스턴스 스케줄러를 사용하여 비용을 40% 절감하는 방법을 참조하세요.

비용 최적화 시나리오

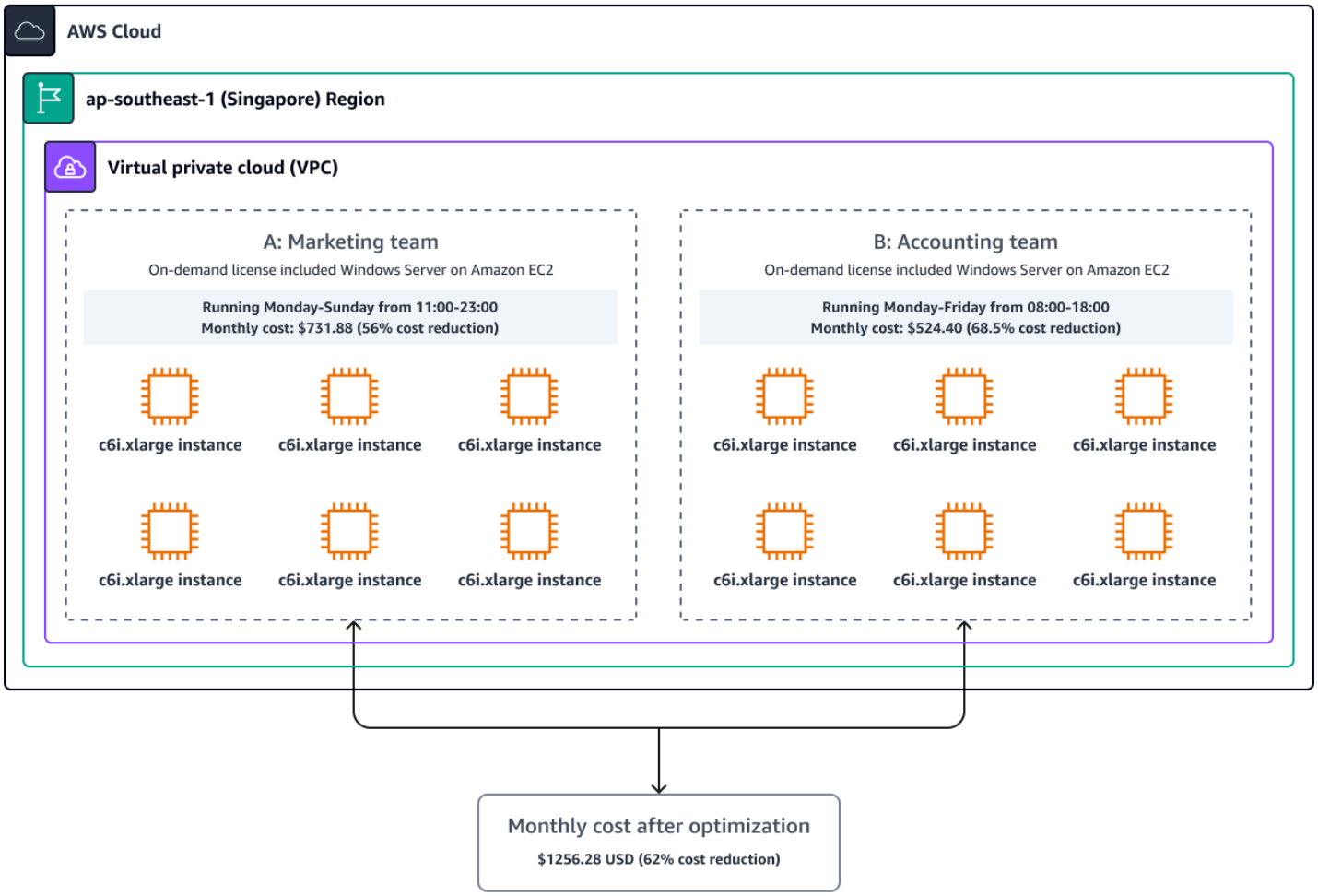
다음 예제 시나리오는에서 인스턴스 스케줄러를 사용할 때의 비용 이점을 보여 줍니다 AWS. 이 시나리오에서 싱가포르의 한 주요 소매업체는 Amazon EC2에 두 개의 Windows 환경을 배포합니다. 워크로드 A라고 하는 첫 번째 환경은 마케팅 팀에서 매장이 열려 있는 동안 실시간 매장 내 트랜잭션을 분석하는 데 사용됩니다. 워크로드 B라고 하는 두 번째 환경은 정규 업무 시간에만 작동하는 회계 팀을 위해 예약되어 있습니다. 두 환경의 현재 운영 일정(24/7)은 현재 사용 패턴을 고려할 때 이상적이지 않으며 회사의 운영 비용을 줄이기 위해 최적화가 필요합니다.

다음 다이어그램은 최적화 전 월별 비용을 보여줍니다.



예를 들어, 3월에는 31일이 있으며이 중 23일은 평일입니다. 마케팅 팀이에서 인스턴스 스케줄러를 사용하고 필요할 때만 인스턴스를 AWS 운영하는 경우(즉, 매월 730시간 대신 매월 321시간 동안) 매월 932.52 USD를 절감할 수 있습니다. 이로 인해 운영 비용이 56% 절감됩니다. 회계 팀은 인스턴스 사용 시간이 매월 730시간에서 230시간으로 감소하여 상당한 이점을 경험할 수도 있습니다. 이로 인해 1,140 USD 또는 68.5%가 감소합니다. 회사는 매월 총 2,072.52 USD(62% 감소) 또는 연간 24,870.24 USD를 절감할 수 있습니다.

다음 다이어그램은 최적화 후 월별 비용을 보여줍니다.



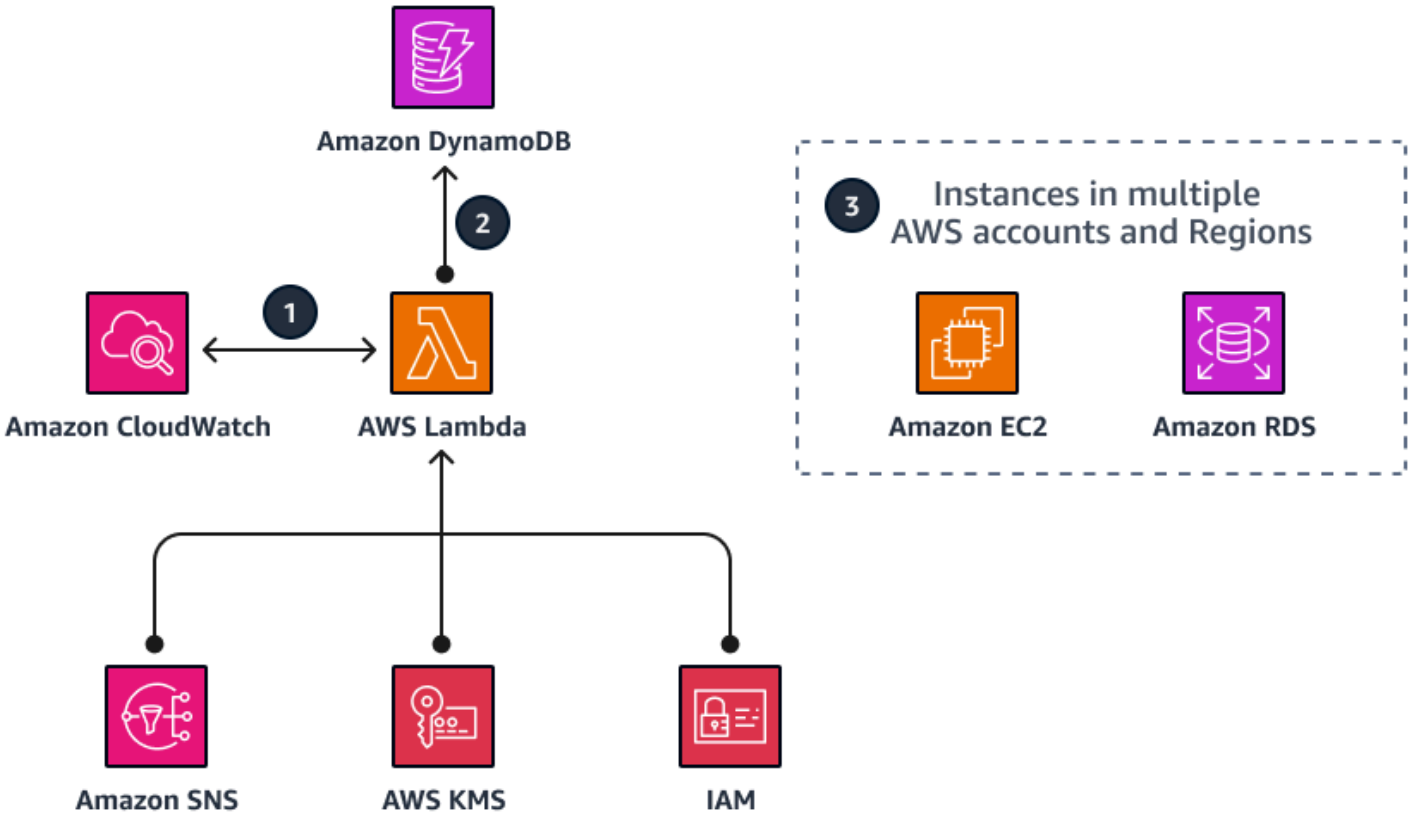
> **Note**
>
> 이 예제의 요금은 2023년 3월에 AWS Pricing Calculator를 사용하여 결정되었습니다.

비용 최적화 권장 사항

이 섹션에서는 이전 비용 최적화 시나리오 섹션에서 다룬 예제 시나리오를 기반으로에 인스턴스 스케줄러 AWS 를 배포하고 구성하는 방법을 설명합니다. AWS다음 단계를 수행하여에서 인스턴스 스케줄러를 사용하여 비용을 최적화하는 것이 좋습니다.

1. 인스턴스 스케줄러 스택 시작
2. 기간 구성
3. 일정 구성
4. 인스턴스 태그 지정

다음 아키텍처 다이어그램은 인스턴스 스케줄러 스택 AWS 클라우드 에서에 생성된 내용을 보여줍니다.



다이어그램은 다음 워크플로 단계를 보여줍니다.

1. AWS CloudFormation 템플릿은 사용자가 정의한 간격으로 Amazon CloudWatch 이벤트를 설정합니다. 이 이벤트는 AWS Lambda 함수를 호출합니다. 구성 중에 AWS 리전 및 계정을 정의합니다. 또한의 인스턴스 스케줄러가 일정을 해당 Amazon EC2 인스턴스, Amazon RDS 인스턴스 및 클러스터와 연결하는 데 AWS 사용하는 사용자 지정 태그를 정의합니다.
2. 일정 구성 값은 Amazon DynamoDB에 저장되며 Lambda 함수는 실행될 때마다 해당 값을 검색합니다. 그런 다음 사용자 지정 태그를 해당 인스턴스에 적용할 수 있습니다.
3. 인스턴스 스케줄러를 처음 구성하는 동안 해당 Amazon EC2 및 Amazon RDS 인스턴스를 식별하기 위한 태그 키를 정의합니다. 일정을 생성할 때 지정한 이름이 태그가 지정된 리소스에 적용할 일정을 식별하는 태그 값으로 사용됩니다.

인스턴스 스케줄러 스택 시작

이 섹션에서는 인스턴스 스케줄러에 대한 CloudFormation 스택을 시작하는 방법을 보여줍니다 AWS.

> **Note**
>
> 에서 인스턴스 스케줄러를 실행하는 동안 AWS 서비스 사용한 비용은 사용자가 부담합니다 AWS. 2023년 1월부터 `us-east-1` 리전의 기본 설정으로이 솔루션을 실행하는 데 드는 비용은 Lambda 요금의 경우 매월 약 9.90 USD이고 Lambda 프리 티어 월별 사용 크레딧이 있는 경우 더 적습니다. 자세한 내용은 AWS 솔루션 라이브러리의 AWS 구현 가이드에 대한 인스턴스 스케줄러의 비용 섹션을 참조하세요.

인스턴스 스케줄러 스택을 시작하려면 다음 단계를 완료합니다.

1. 에AWS Management Console 로그인하고 솔루션 시작(다운로드 가능 템플릿)을 선택하여 `instance-scheduler-on-aws.template` CloudFormation 템플릿을 시작합니다.

   > **Note**
   >
   > 또한 구현의 시작점으로 사용할 템플릿을 다운로드할 수도 있습니다.

2. 이 템플릿은 기본적으로 미국 동부(버지니아 북부) 리전에서 시작됩니다. 다른 리전에서 인스턴스 스케줄러를 시작하려면 콘솔 탐색 모음에서 리전 선택기를 사용합니다.

   > **Note**
   >
   > 이 예제에서는 아시아 태평양(싱가포르) 리전을 사용합니다.

3. 스택 생성 페이지의 사전 조건 - 템플릿 준비 섹션에서 템플릿 준비 완료 옵션이 선택되어 있는지 확인합니다. 템플릿 소스 섹션에서 Amazon S3 URL 옵션이 선택되어 있는지 확인합니다.
4. Amazon S3 URL 텍스트 상자에 올바른 템플릿 URL이 있는지 확인한 후 다음을 선택합니다.
5. 스택 세부 정보 지정 페이지에서 솔루션 스택 이름을 할당합니다. 문자 제한 이름 지정에 대한 자세한 내용은 AWS Identity and Access Management (IAM) 설명서의 IAM 및 STS 제한을 참조하세요. 이 가이드의 예제에 대한 스택 이름을 라고 합니다`MyInstanceScheduler`.

   > **Note**
   >
   > 스택 이름은 28자를 초과할 수 없습니다.

6. 파라미터에서 템플릿의 파라미터를 검토하고 필요에 따라 수정합니다.

7. Next(다음)를 선택합니다. Configure stack options(스택 옵션 구성) 페이지에서 Next(다음)를 선택합니다.
8. 검토 페이지에서 설정을 검토하고 확인합니다. 템플릿이 IAM 리소스를 생성할 것임을 확인하는 상자를 선택합니다.
9. 생성을 선택하여 스택을 배포합니다.

기간 구성

CloudFormation 템플릿을 배포한 후 솔루션은 사용자 지정 기간 규칙 및 일정을 생성하는 데 참조로 사용할 수 있는 샘플 기간 규칙 및 일정이 포함된 DynamoDB 테이블을 생성합니다. 예제 기간 구성은 설명서의 인스턴스 스케줄러의 샘플 일정을 참조하세요 AWS .

이 시나리오의 단계를 완료하려면 각 워크로드에 해당하고 특정 요구 사항을 충족하는 기간을 생성해야 합니다. 예시:

```
Period 1 (Workload A):
    Name: retail-hours
    Days: Monday to Sunday
    Hours: 1100 - 2300
Period 2 (Workload B):
    Name: office-hours
    Days: Monday to Friday
    Hours: 0800 - 1800
```

기간을 구성하려면 다음 단계를 완료하세요.

1. DynamoDB 콘솔에 로그인하고에서 인스턴스 스케줄러에 대한 CloudFormation 템플릿을 시작한 리전과 동일한 리전에 있는지 확인합니다 AWS.
2. 탐색 창에서 테이블을 선택한 다음 ConfigTable이라는 테이블을 선택합니다.
3. 테이블 항목 탐색을 선택합니다.
4. 근무 시간 기간을 생성하려면 근무 시간 항목의 기간을 선택합니다.
5. 항목 편집 페이지에서 시작 시간 값을 0800으로 변경하고 종료 시간을 1800으로 변경합니다. 평일에는 기본값을 그대로 둡니다.

> **Note**
>
> 시작 시간 및 종료 시간 값은 인스턴스를 시작하고 중지해야 하는 시기를 결정하는 반면, 평일 값은이 일정이 적용되는 요일(이 예제의 경우 월요일~금요일)을 결정합니다.

6. Save changes(변경 사항 저장)를 선택합니다.
7. 업무 시간 기간을 복제하여 소매 시간에 대한 새 기간을 생성하는 데 사용하려면 업무 시간 항목의 기간을 선택합니다. 그런 다음 작업 메뉴에서 중복 항목을 선택합니다.
8. 필요에 맞게 속성을 수정합니다. 다음 속성은 예제 시나리오의 요구 사항을 충족하는 데 사용됩니다.

```
type: period
name: retail-hours
begintime: 11:00
description: Retail hours
endtime: 23:00
weekdays: mon-sun
```

9. 항목 생성을 선택합니다.

10DynamoDB ConfigTable에서 항목 목록에 나열된 방금 생성한 두 기간을 식별합니다.

일정 구성

의 인스턴스 스케줄러 컨텍스트에서 AWS일정은 하나 이상의 기간 및 관련 시간대의 적용을 참조합니다. 그런 다음 이러한 일정이 인스턴스에 태그로 할당됩니다. 이 단원에서는 두 예제 워크로드의 다양한 시간 패턴을 수용하기 위해 두 개의 일정(아래 참조)을 생성한 다음 일정을 이전 단원에서 생성한 기간과 연결하는 방법을 보여줍니다.

```
Schedule 1:
    Name: singapore-office-hours
    Period: office-hours
    Timezone: Asia/Singapore
Schedule 2:
    Name: singapore-retail-hours
    Period: retail-hours
    Timezone: Asia/Singapore
```

일정을 생성하고 구성하려면 다음 단계를 완료하세요.

1. DynamoDB 콘솔에 로그인하고에서 인스턴스 스케줄러에 대한 CloudFormation 템플릿을 시작한 리전과 동일한 리전에 있는지 확인합니다 AWS.
2. 탐색 창에서 테이블을 선택한 다음 ConfigTable이라는 테이블을 선택합니다.
3. 테이블 항목 탐색을 선택합니다.
4. 영국 근무 시간 일정을 복제하고 이를 사용하여 근무 시간(이 예에서는 싱가포르 근무 시간)에 대한 새 일정을 생성하려면 uk-office-hours 항목에 대한 일정을 선택합니다. 그런 다음 작업 메뉴에서 중복 항목을 선택합니다.
5. 필요에 맞게 속성을 수정합니다. 다음 속성은 예제 시나리오의 요구 사항을 충족하는 데 사용됩니다.

```
type: schedule
name: singapore-office-hours
description: Office hours in Singapore
periods: office-hours
timezone: Asia/Singapore
```

6. 항목 생성을 선택합니다.
7. 4~6단계를 반복하여 다음 속성 값을 사용하여 싱가포르 소매 시간에 대한 일정을 생성합니다.

```
type: schedule
name: singapore-retail-hours
description: Retail hours in Singapore
periods: retail-hours
timezone: Asia/Singapore
```

8. DynamoDB ConfigTable에서 생성한 두 개의 일정과 두 개의 기간을 식별합니다.

인스턴스 태그 지정

일정을 설정한 후에는 태그를 사용하여 사용하려는 특정 인스턴스에 일정을 할당해야 합니다. 내에서 태그 편집기를 사용하여 Amazon EC2 인스턴스AWS Resource Groups에 태그를 생성하고 할당할 수 있습니다.

1. 에 로그인AWS Management Console하고 이전에 CloudFormation 템플릿을 시작한 리전과 동일한 리전에 있는지 확인합니다.
2. Resource Groups 콘솔을 엽니다. 탐색 창에서 태그 지정을 확장한 다음 태그 편집기를 선택합니다.
3. 태그를 지정할 리소스 찾기 섹션의 리전에서 리전을 선택합니다. 리소스 유형에서 Amazon EC2 또는 Amazon RDS를 선택합니다. 이 시나리오는 워크로드 A의 Amazon EC2 인스턴스에 중점을 둡니

다. 마케팅 팀은 싱가포르 리전에서 워크로드 A를 사용하고 있습니다. 이 워크로드의 리소스에는 이미 부서 키와 마케팅 값으로 태그가 지정되어 있습니다. 이 태그를 사용하여 인스턴스를 검색할 수 있습니다.

4. 리소스 검색을 선택합니다.
5. 검색 결과 목록에서 일정에 포함할 인스턴스를 선택한 다음 선택한 리소스의 태그 관리를 선택합니다.
6. 선택한 모든 리소스의 태그 편집 섹션에서 태그 추가를 선택하여 인스턴스 스케줄러 일정 태그를 EC2 인스턴스에 추가합니다. schedulea(이전에는 DynamoDB에서 생성됨)와 일치하는 태그 키와 값을 사용할 수 있습니다.
7. 태그 키에 일정을 추가합니다. 태그 값에 singapore-retail-hours 입력합니다.
8. 변경 사항 검토 및 적용을 선택합니다.
9. 선택한 모든 EC2 인스턴스에 태그를 적용하려면 선택한 모든 인스턴스에 변경 사항 적용을 선택합니다.
10. 적용하려는 추가 일정에 대해 3~9단계를 반복합니다.

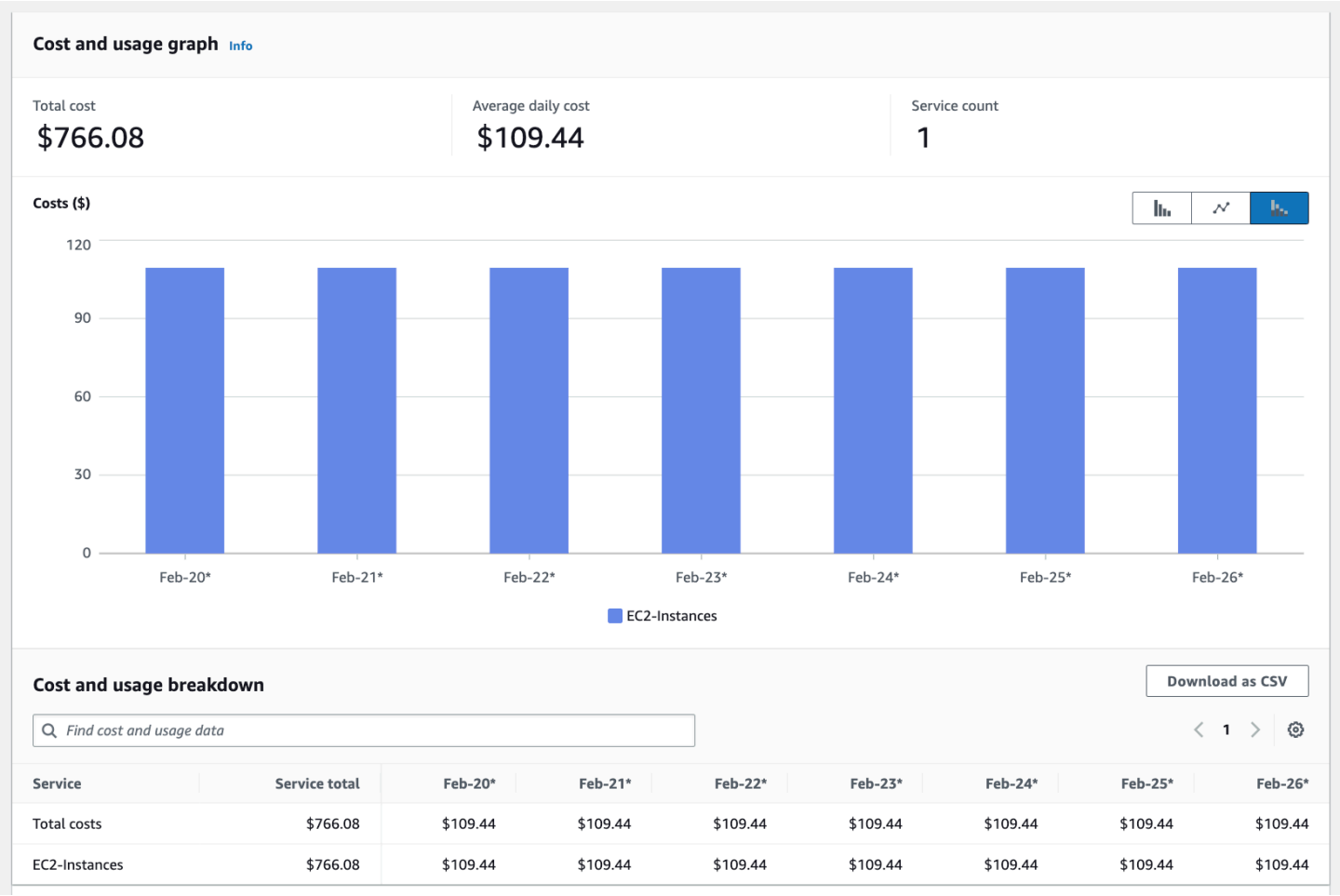
결과 검증

AWS Cost Explorer를 사용하여에서 인스턴스 스케줄러 사용의 비용 이점을 측정하는 것이 좋습니다 AWS. Cost Explorer를 사용하여 다음을 수행할 수 있습니다.

- 인스턴스 스케줄러에서 관리하는 인스턴스를 포함하여 EC2 인스턴스와 관련된 비용을 보고 분석합니다.
- 특정 워크로드에 집중하고 인스턴스 스케줄러를 사용하여 달성한 비용 절감을 세부적으로 볼 수 있도록 태그별로 Cost Explorer 보기를 필터링합니다.
- 인스턴스 스케줄러 사용의 재정적 영향에 대한 인사이트를 얻습니다.
- 추가 비용 최적화 기회를 식별하고 데이터 기반 결정을 내려 AWS 지출을 최적화합니다.

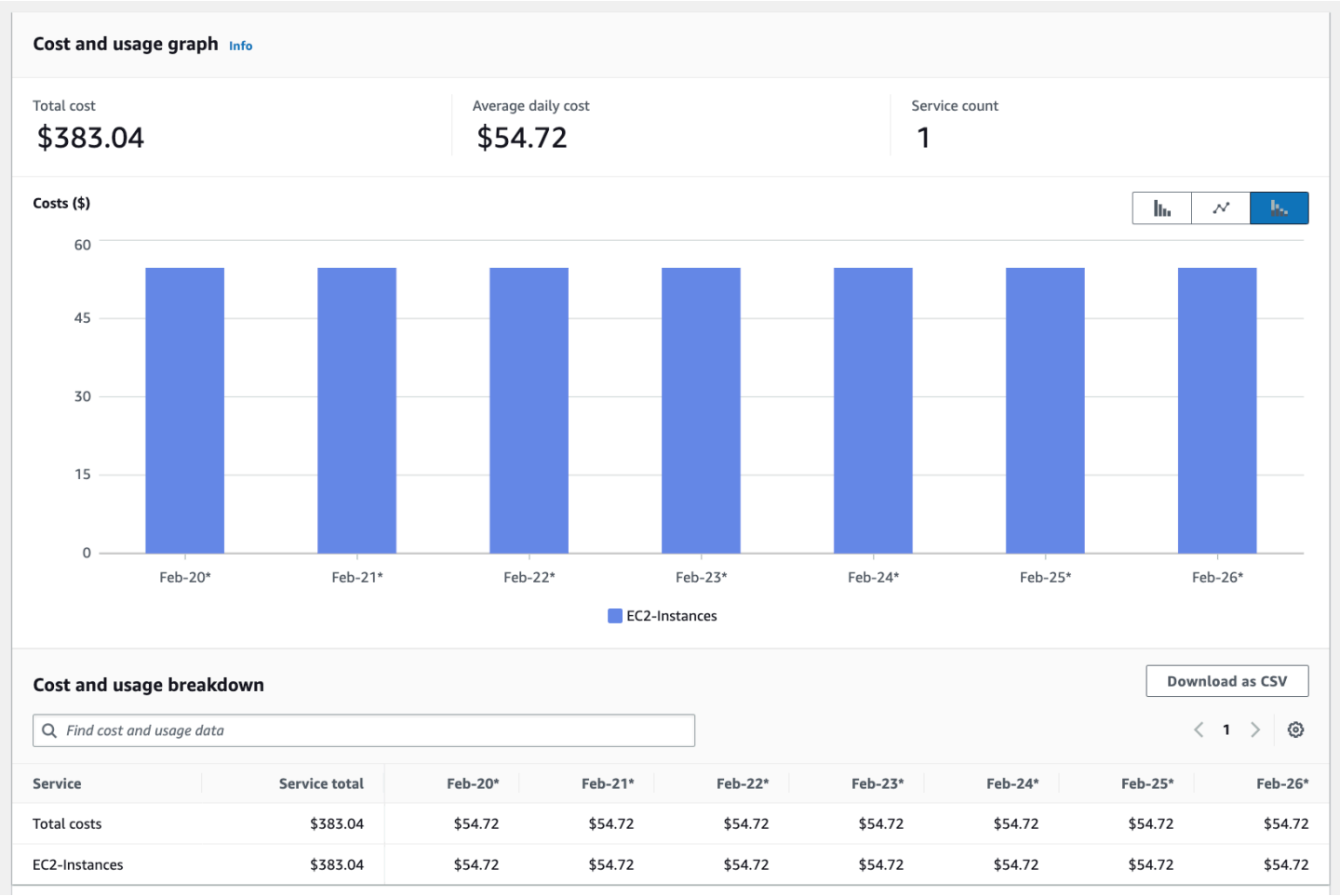
다음 차트는 인스턴스 스케줄러를 사용하여 최적화하기 전 7일(월요일~일요일) 동안 워크로드 A 및 워크로드 B를 운영하는 비용을 보여줍니다.

워크로드 A와 B의 총 비용 합계



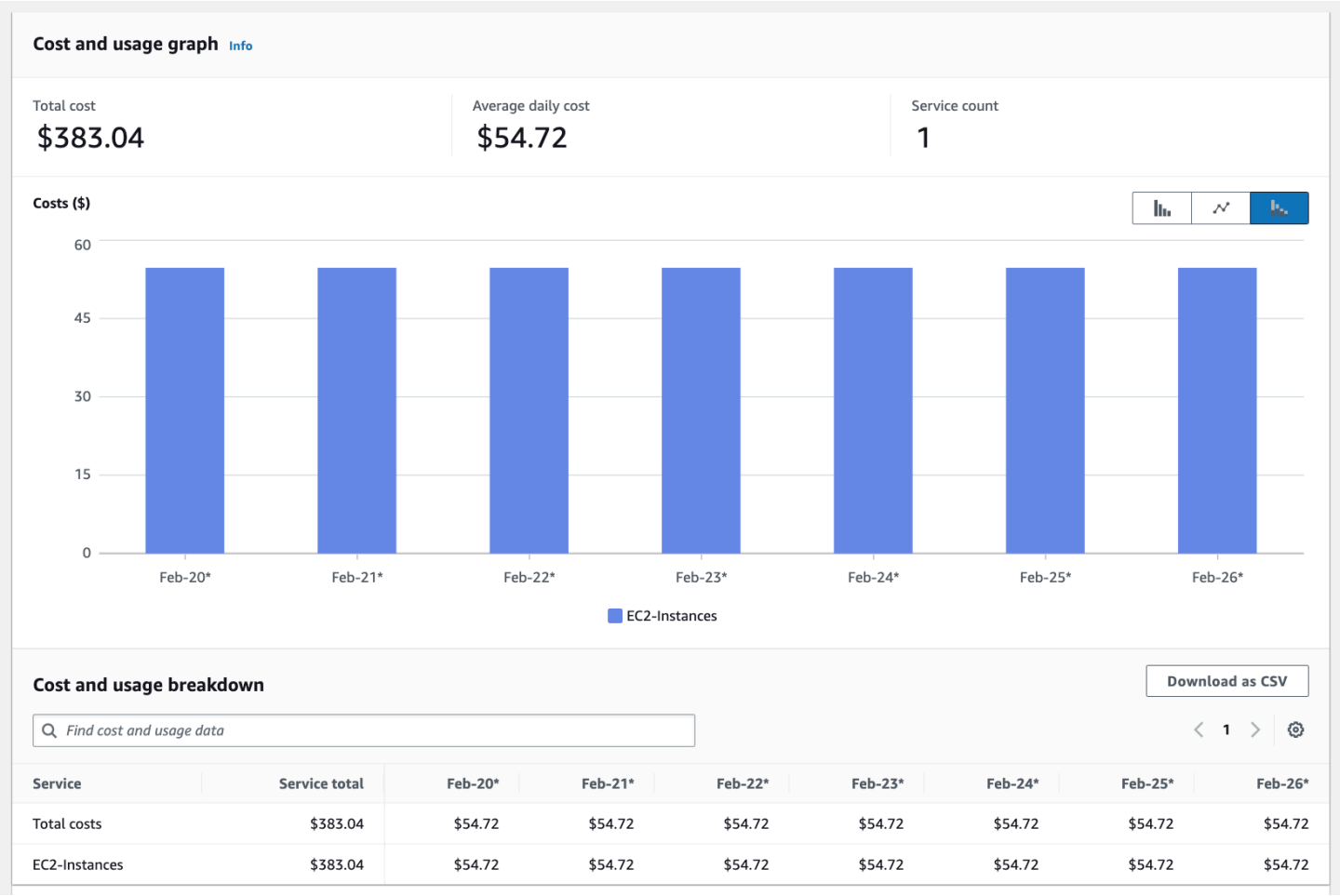
Service	Service total	Feb-20*	Feb-21*	Feb-22*	Feb-23*	Feb-24*	Feb-25*	Feb-26*
Total costs	$766.08	$109.44	$109.44	$109.44	$109.44	$109.44	$109.44	$109.44
EC2-Instances	$766.08	$109.44	$109.44	$109.44	$109.44	$109.44	$109.44	$109.44

워크로드 A 비용



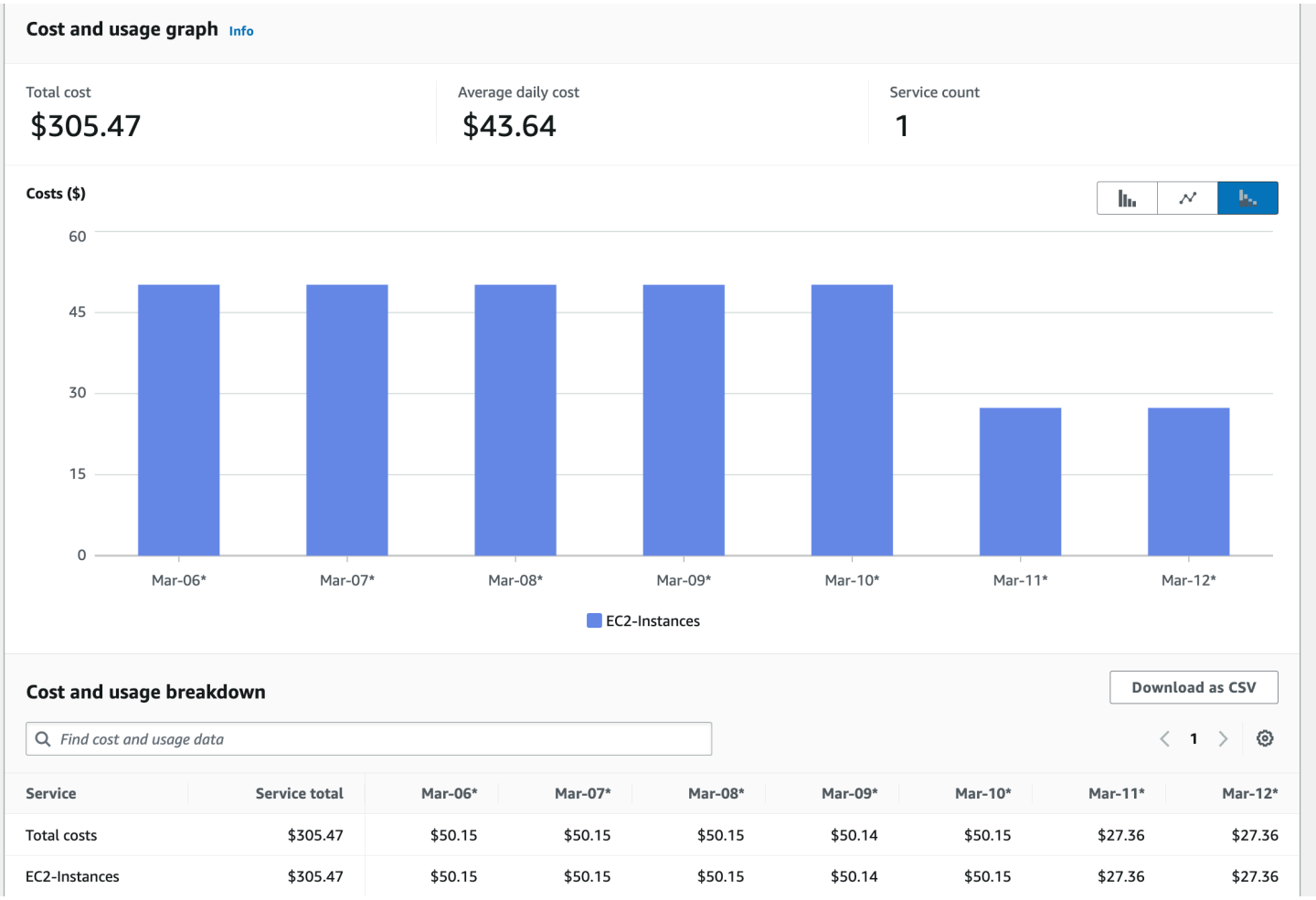
Service	Service total	Feb-20*	Feb-21*	Feb-22*	Feb-23*	Feb-24*	Feb-25*	Feb-26*
Total costs	$383.04	$54.72	$54.72	$54.72	$54.72	$54.72	$54.72	$54.72
EC2-Instances	$383.04	$54.72	$54.72	$54.72	$54.72	$54.72	$54.72	$54.72

워크로드 B 비용



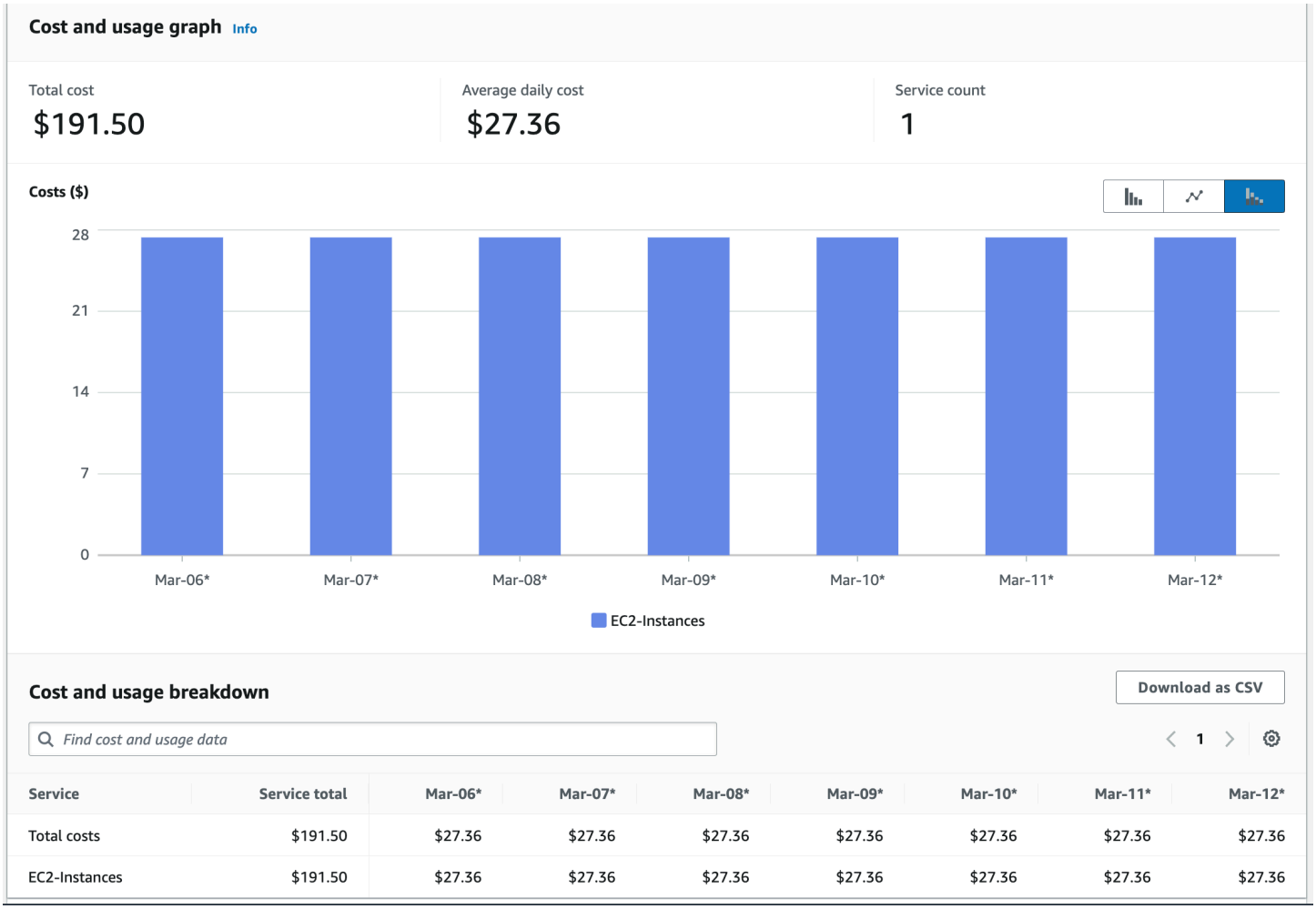
이 시나리오에서 Cost Explorer는 인스턴스 스케줄러를 구현하여 발생하는 비용 절감을 보여줍니다 AWS. 다음 차트는 최적화 후 7일(월요일~일요일) 동안 워크로드 A 및 워크로드 B의 운영 비용을 보여줍니다.

워크로드 A와 B의 총 비용 합계

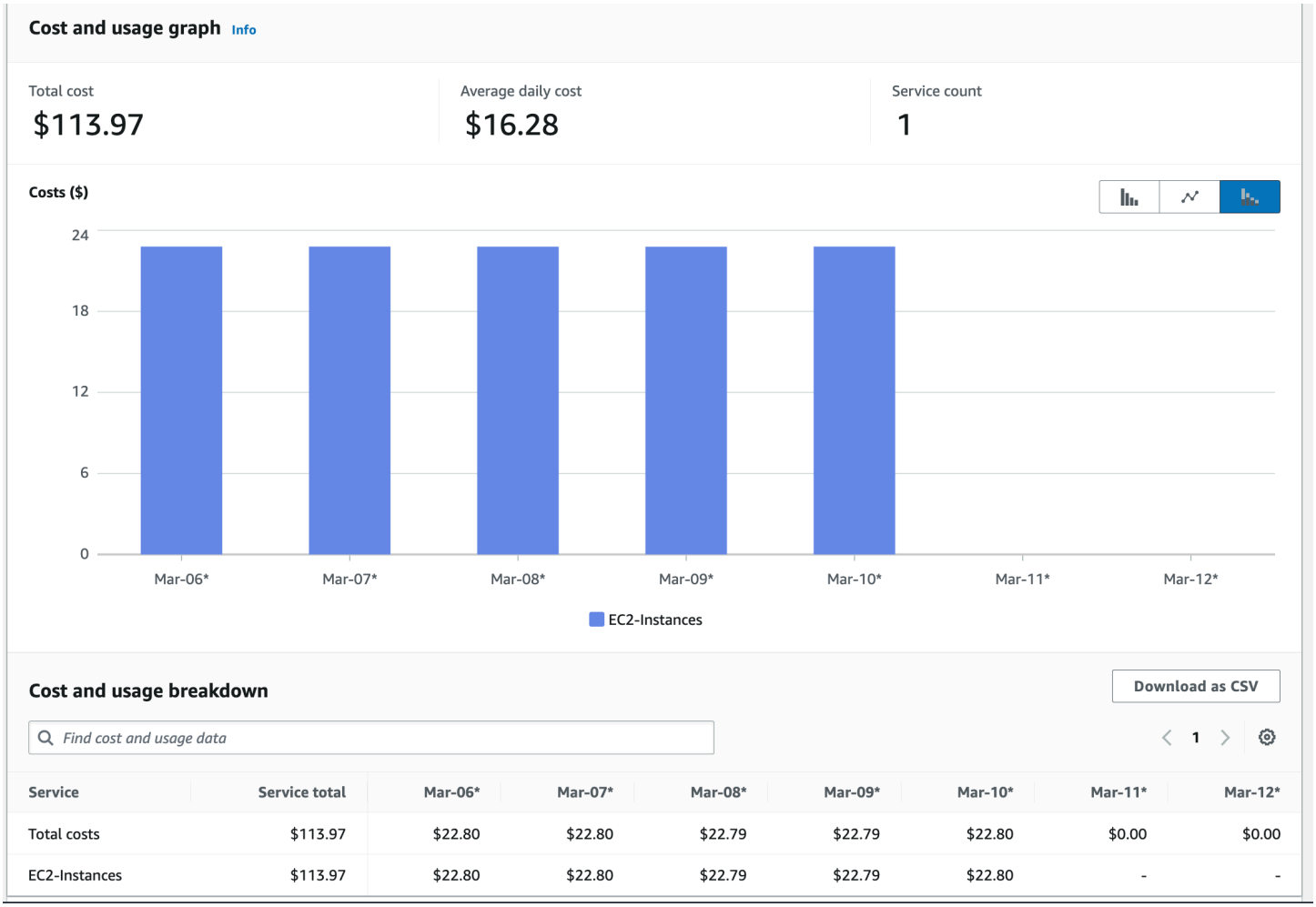


Service	Service total	Mar-06*	Mar-07*	Mar-08*	Mar-09*	Mar-10*	Mar-11*	Mar-12*
Total costs	$305.47	$50.15	$50.15	$50.15	$50.14	$50.15	$27.36	$27.36
EC2-Instances	$305.47	$50.15	$50.15	$50.15	$50.14	$50.15	$27.36	$27.36

워크로드 A 비용


Cost and usage graph Info
Total cost
$191.50
Average daily cost
$27.36
Service count
1
Costs ($)
28
21
14
7
0
Mar-06*
Mar-07*
Mar-08*
Mar-09*
Mar-10*
Mar-11*
Mar-12*
EC2-Instances
Cost and usage breakdown
Download as CSV
Find cost and usage data
1
Service
Service total
Mar-06*
Mar-07*
Mar-08*
Mar-09*
Mar-10*
Mar-11*
Mar-12*
Total costs
$191.50
$27.36
$27.36
$27.36
$27.36
$27.36
$27.36
$27.36
EC2-Instances
$191.50
$27.36
$27.36
$27.36
$27.36
$27.36
$27.36
$27.36

워크로드 B 비용



추가 리소스

- AWS 인스턴스 시작 및 중지 자동화(AWS 설명서의 인스턴스 스케줄러)
- 기본 사항으로 돌아가기: 인스턴스 스케줄러를 사용하여 Amazon EC2 및 Amazon RDS 리소스 비용 제어(YouTube)
- AWS 리소스 태그 지정(AWS 리소스 태그 지정 사용 설명서)
- 를 사용한 비용 분석 AWS Cost Explorer(AWS 결제 및 비용 관리 문서)

적절한 크기의 Windows 워크로드

개요

올바른 크기 조정은 가장 강력한 비용 절감 도구 중 하나입니다.는 AWS 최적화 및 라이선스 평가(AWS OLA)를 사용하여 잠재적 워크로드를 검토하는 것부터를 사용하여 기존 워크로드를 검토하는 것까지 올바른 크기 조정 정보를 수집하는 다양한 방법을 AWS 제공합니다AWS Cost Explorer.

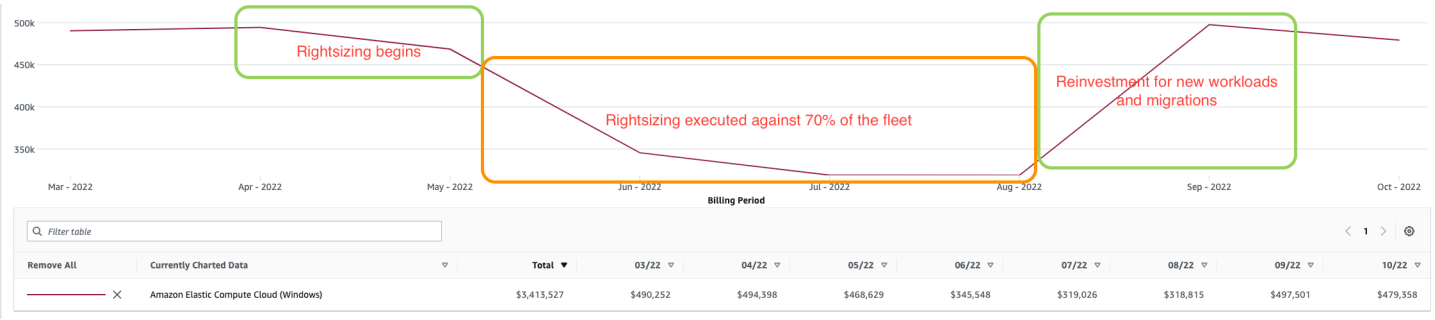
이 섹션에서는를 AWS Compute Optimizer 사용하여 Amazon EC2 올바른 크기 조정 기회를 식별하는 방법을 보여줍니다. Compute Optimizer는 다음 유형의 AWS 리소스에 대한 과다 프로비저닝 및 과소 프로비저닝을 방지하는 데 도움이 됩니다.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 유형
- Amazon Elastic Block Store(Amazon EBS) 볼륨
- 의 Amazon Elastic Container Service(Amazon ECS) 서비스 AWS Fargate
- AWS Lambda Amazon CloudWatch에서 제공하는 사용률 데이터를 기반으로 하는 함수

비용 최적화 시나리오

올바른 크기 조정의 효과를 측정하는 것은 어려울 수 있습니다. 올바른 크기 조정 작업은 특정 앱, 팀 또는 전체 조직을 대상으로 할 수 있기 때문입니다. 예를 들어 플릿의 90%가 Windows 워크로드로 구성된 수천 개의 인스턴스를 AWS로 마이그레이션하는 조직을 가정해 보겠습니다. 조직은 Compute Optimizer를 사용하여 플릿을 분석하고 계정 및 전체에서 상당한 오버프로비저닝을 발견할 수 있습니다 AWS 리전. 그런 다음 AWS Systems Manager Automation을 사용하여 여러 유지 관리 기간을 통해 플릿의 크기를 조정할 수 있습니다. 따라서 조직은 플릿의 70%에 적합한 크기의 인스턴스 유형을 조정하고 35%의 비용 절감을 달성합니다.

다음 대시보드는이 예제 조직에서 Compute Optimizer의 올바른 크기 조정 권장 사항을 전략적으로 구현했기 때문에 몇 개월 동안 달성한 절감액을 보여줍니다. 이들의 목표는 계약이 거의 종료되는 콜로케이션 데이터 센터에서 지연된 마이그레이션을 재개하기 위해 기존 워크로드를 최대한 효율적으로 운영하는 것이었습니다.



비용 최적화 권장 사항

Compute Optimizer를 사용하여 비용을 최적화하려면 다음 단계를 수행하는 것이 좋습니다.

- Compute Optimizer 활성화
- Windows 노드에 대한 메모리 지표 수집 활성화
- Compute Optimizer 권장 사항 사용
- 올바른 크기 조정을 위한 태그 인스턴스
- 비용 할당 태그가 AWS 결제 도구에서 작동하도록 활성화
- AWS Systems Manager Automation을 사용하여 올바른 크기 조정 권장 사항 구현
- 대체 크기 조정 방법 고려
- Cost Explorer에서 비용 전후 검토

Compute Optimizer 활성화

조직 또는의 단일 계정 수준에서 Compute Optimizer를 활성화할 수 있습니다 AWS Organizations. 조직 전체 구성은 모든 멤버 계정의 전체 플릿에서 신규 및 기존 인스턴스에 대한 지속적인 보고서를 제공합니다. 이를 통해 올바른 크기 조정은 point-in-time 활동으로 사용할 수 있습니다.

조직 수준

대부분의 조직에서 Compute Optimizer를 사용하는 가장 효율적인 방법은 조직 수준에서 사용하는 것입니다. 이를 통해 조직에 대한 다중 계정 및 다중 리전 가시성을 제공하고 검토를 위해 데이터를 하나의 소스로 중앙 집중화할 수 있습니다. 조직 수준에서 이를 활성화하려면 다음을 수행합니다.

1. 필요한 권한이 있는 역할로 Organizations 관리 계정에 로그인하고이 조직 내의 모든 계정에 대해 옵트인하도록 선택합니다. 조직의 모든 기능을 활성화해야 합니다.
2. 관리 계정을 활성화한 후 계정에 로그인하고, 다른 모든 멤버 계정을 보고, 권장 사항을 찾아볼 수 있습니다.

> **Note**
>
> Compute Optimizer에 대해 위임된 관리자 계정을 구성하는 것이 가장 좋습니다. 이렇게 하면 최소 권한 원칙을 적용할 수 있습니다. 이렇게 하면 조직 전체 서비스에 대한 액세스를 제공하면서 조직의 관리 계정에 대한 액세스를 최소화할 수 있습니다.

단일 계정 수준

비용이 많이 드는 계정을 대상으로 하지만에 액세스할 수 없는 경우에도 해당 계정 및 리전에 대해 Compute Optimizer를 활성화 AWS Organizations할 수 있습니다. 옵트인 프로세스에 대한 자세한 내용은 Compute Optimizer 설명서의 시작하기 AWS Compute Optimizer를 참조하세요.

Windows 노드에 대한 메모리 지표 수집 활성화

메모리 지표는 Compute Optimizer에 조직에서 올바른 크기 조정 권장 사항을 제공하는 데 필요한 필수 지표를 제공합니다. 이는 권장 사항을 제공하기 전에 CPU, 메모리, 네트워크 및 스토리지를 분석하기 때문입니다.

Windows EC2 인스턴스의 메모리 지표를 Compute Optimizer로 전달하려면 CloudWatch 에이전트를 활성화하고 60초마다 메모리 지표를 수집하도록 구성해야 합니다. CloudWatch에서 메모리 지표를 사용하는 데 드는 추가 비용은 없습니다.

CloudWatch 에이전트 활성화 및 메모리 지표 구성

ComputeOptimize.yml 파일을 다운로드합니다. 이 파일을 사용하여 계정의 모든 인스턴스에 대해 메모리 수집을 활성화할 수 있습니다. 템플릿 파일은 다음 구성 요소를 생성합니다.

- AWS Systems Manager 파라미터 스토어 - 메모리 지표를 수집하는 데 필요한 CloudWatch 에이전트의 구성을 저장합니다.
- AWS Identity and Access Management AWS 에 대한 관리형 정책 AWS Systems Manager이 연결된 (IAM) 역할 - Systems Manager Automation 문서용입니다.
- AWS Systems Manager 문서 - CloudWatch 에이전트를 설치하고 구성합니다(기존 CloudWatch 구성 대체).
- AWS Systems Manager 상태 관리자 연결 - 이를 통해 Systems Manager 문서를 계정의 모든 인스턴스에서 실행할 수 있습니다.

> **Important**
>
> 이 템플릿을 실행하면 인스턴스의 기존 CloudWatch 구성을 덮어씁니다.

그런 후, 다음 작업을 수행합니다.

1. 에 로그인 AWS Management Console 하고 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택을 선택합니다.
3. 스택 생성을 선택한 다음 기존 리소스 사용(리소스 가져오기))를 선택합니다.
4. Next(다음)를 선택합니다.
5. 템플릿 리소스에서 템플릿 파일 업로드를 선택합니다.
6. 파일을 선택한 다음 `ComputeOptimize.yml` 파일을 업로드합니다.
7. Next(다음)를 선택합니다.
8. 스택 세부 정보 지정 페이지의 스택 이름에 스택 이름을 입력한 후 다음을 선택합니다.
9. 리소스 식별 페이지에서 가져오려는 리소스의 식별자 값을 입력합니다.
10. 리소스 가져오기를 선택합니다.
11. 스택이 배포된 후 출력 탭을 선택하여 연결에 대한 키, 값 및 설명을 찾습니다.

연결 진행 상황 모니터링

1. CloudFormation 스택 배포가 완료되면 Systems Manager 콘솔을 엽니다.
2. 탐색 창의 노드 관리 섹션에서 상태 관리자를 선택합니다.
3. 연결 페이지에서 연결의 연결 ID를 선택합니다.
4. Execution history(실행 내역) 탭을 선택합니다.
5. 실행 ID 열에서 연결의 실행 ID를 선택합니다. 상태는 성공이어야 합니다.

CloudWatch에서 지표 보기

지표가 CloudWatch를 채울 때까지 5분 이상 기다리는 것이 좋습니다.

1. CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표 섹션을 확장한 다음 모든 지표를 선택합니다.
3. 지표가 CWAgent 네임스페이스 아래에 나타나는지 확인합니다.

> **Note**
>
> 설정을 새 인스턴스에 적용하려면 연결을 다시 실행합니다.

Compute Optimizer 권장 사항 사용

단일 계정 및 단일 리전 내에서 올바른 크기 조정을 수행하는 데 중점을 둔 예를 생각해 보세요. 이 예제에서는 Compute Optimizer가 모든 계정의 조직 수준에서 활성화됩니다. 올바른 크기 조정은 대부분의 경우 몇 주 동안 예약된 유지 관리 기간 동안 애플리케이션 소유자가 정밀도로 수행하는 중단적인 프로세스입니다.

조직의 관리 계정 내에서 Compute Optimizer로 이동하는 경우(다음 단계에 표시됨) 조사하려는 계정을 선택할 수 있습니다. 이 예제에서는 `us-east-1` 리전의 단일 계정에서 실행되는 인스턴스가 6개 있습니다. 6개의 인스턴스가 모두 과다 프로비저닝됩니다. 목표는 Compute Optimizer의 권장 사항에 따라 인스턴스의 크기를 조정하는 것입니다.

과다 프로비저닝된 인스턴스 식별 및 권장 사항 세부 정보 내보내기

1. 에 로그인 AWS Management Console 하고 Compute Optimizer 콘솔을 엽니다.
2. 탐색 창에서 대시보드를 선택합니다.
3. 대시보드 페이지의 검색 상자에 Region=US East(버지니아 북부)를 입력합니다. 그런 다음 Findings=Over-provisioned를 입력합니다. 이러한 필터를 사용하면 `us-east-1` 리전에서 과다 프로비저닝된 모든 인스턴스를 볼 수 있습니다.
4. 과다 프로비저닝된 EC2 인스턴스에 대한 자세한 권장 사항을 검토하려면 EC2 인스턴스 카드까지 아래로 스크롤한 다음 권장 사항 보기를 선택합니다.
5. 내보내기를 선택하고 나중에 사용할 수 있도록 파일을 저장합니다.
6. S3 버킷에 내보내기 파일의 대상으로 사용할 Amazon S3 버킷의 이름을 입력합니다.

> **Note**
>
> 향후 검토를 위해 권장 사항을 저장하려면 Compute Optimizer가 각 리전에서에 쓸 수 있는 S3 버킷이 있어야 합니다. 자세한 내용은 Compute Optimizer 설명서의에 대한 Amazon S3 버킷 정책을 AWS Compute Optimizer 참조하세요.

7. 필터 내보내기 섹션에서 조직의 모든 멤버 계정에 대한 권장 사항 포함 확인란을 선택합니다.
8. 리소스 유형에서 EC2 인스턴스를 선택합니다.

9. 포함할 열 섹션에서 모두 선택 확인란을 선택합니다.
10. 내보내기를 선택합니다.

권장 사항에 따라 인스턴스 선택

인스턴스 권장 사항은 Compute Optimizer에서 수집하고 분석한 성능 지표를 기반으로 합니다. 최상의 인스턴스를 선택하려면 인스턴스에서 실행되는 워크로드를 인식해야 합니다. 이 예제에서는 최신 세대의 Amazon EC2 R6i, R5 및 T3 인스턴스 중에서 선택할 수 있다고 가정합니다. T3 인스턴스는 버스트 가능하며 네트워크 대역폭 기능이 낮습니다. R5 및 R6 인스턴스는 시간당 비용이 동일하며 거의 동일합니다. 그러나 R6 인스턴스는 네트워크 대역폭 용량이 더 크고, 최신 세대의 Intel 프로세서를 갖추고, R5와 동일한 컴퓨팅 공간을 제공합니다. 이 예제에서 R6는 크기 조정을 위해 선택할 수 있는 가장 좋은 옵션입니다.

1. Compute Optimizer 콘솔의 탐색 모음에서 EC2 인스턴스에 대한 권장 사항을 선택합니다. 이 페이지에서는 현재 인스턴스 유형을 교체하는 권장 옵션과 비교합니다.
2. 크기를 조정하려는 인스턴스의 ID를 가져오려면 관리 계정에서 Amazon S3 콘솔을 엽니다 AWS Organizations.
3. 탐색 창에서 버킷을 선택한 다음 내보낸 결과를 저장하는 데 사용할 버킷을 선택합니다.
4. 객체 탭의 객체 목록에서 내보내기 파일을 선택한 다음 다운로드를 선택합니다.
5. 파일에서 인스턴스 정보를 추출하려면 Microsoft Excel의 데이터 탭에 있는 열에 텍스트 버튼을 사용하면 됩니다.

> **Note**
>
> 인스턴스 IDs는 Amazon 리소스 이름(ARNs)으로 표시됩니다. 구분 기호를 "/"로 설정하고 인스턴스 ID를 추출해야 합니다. 또는 스크립트를 작성하거나 통합 개발 환경(IDE)을 사용하여 ARN을 잘라낼 수 있습니다.

6. Excel에서 결과 열을 필터링하여 OVER_PROVISIONED 인스턴스만 표시합니다. 올바른 크기 조정을 위해 대상으로 하는 인스턴스입니다.
7. 나중에 쉽게 액세스할 수 있도록 텍스트 편집기에 인스턴스 IDs를 저장합니다.

올바른 크기 조정을 위한 태그 인스턴스

워크로드에 태그를 지정하는 것은에서 리소스를 구성하기 위한 강력한 도구입니다 AWS. 태그를 사용하면 비용에 대한 세분화된 가시성을 확보하고 차지백을 용이하게 할 수 있습니다. AWS 리소스에 태

그를 추가하는 전략 및 방법에 대한 자세한 내용은 리소스 태그 지정을 위한 AWS 백서 모범 사례를 참조하세요. AWS 이 예제에서는 AWS Tag Editor를 사용하여 유지 관리 기간 동안 크기 조정을 위해 대상으로 지정하려는 과다 프로비저닝된 인스턴스에서 태깅을 조정할 수 있습니다. 또한이 태그를 사용하여 변경 전후의 비용을 볼 수 있습니다.

1. 에 로그인 AWS Management Console 하고 크기 조정 대상 인스턴스가 포함된 계정의 AWS Resource Groups 콘솔을 엽니다.
2. 탐색 모음의 태그 지정 섹션에서 태그 편집기를 선택합니다.
3. 리전에서 대상 리전을 선택합니다.
4. 리소스 유형에서 AWS::EC2::Instance를 선택합니다.
5. 리소스 검색을 선택합니다.
6. 리소스 검색 결과 페이지에서 적절한 크기를 지정할 모든 인스턴스를 선택한 다음 선택한 리소스의 태그 관리를 선택합니다.
7. 태그 추가를 선택합니다.
8. 태그 키에 크기 조정을 입력합니다. 태그 값에 활성화를 입력합니다. 그런 다음 태그 변경 사항 검토 및 적용을 선택합니다.

> **Note**
>
> 나중에 Cost Explorer에서 필터링하는 데 도움이 되도록 팀 또는 사업부와 같은 추가 메타데이터를 포함할 수 있습니다.

리소스에 사용자 정의 태그를 생성하고 적용한 후 활성화를 위해 비용 할당 태그 페이지에 태그가 표시되는 데 최대 24시간이 걸릴 수 있습니다. 활성화할 태그를 선택한 후 태그가 활성화되는 데 24시간이 더 걸릴 수 있습니다.

고급 사용자의 경우 대상 계정 및 리전 AWS CloudShell 내에서를 사용하여 여러 인스턴스에 태그를 지정할 수 있습니다. 예시:

```
bash
#!/bin/bash
# Set variables
TAG_KEY="rightsizing"
TAG_VALUE="type-m5"
# Get a list of instance IDs
INSTANCE_IDS=$(aws ec2 describe-instances –query
 "Reservations[].Instances[].InstanceId" –output text)
```

```
# Loop through each instance ID and add the tag
for INSTANCE_ID in $INSTANCE_IDS; do
 aws ec2 create-tags –resources $INSTANCE_ID –tags Key=$TAG_KEY,Value=$TAG_VALUE
done
```

비용 할당 태그가 AWS 결제 도구에서 작동하도록 활성화

사용자 정의 비용 할당 태그를 활성화하는 것이 좋습니다. 이렇게 하면 AWS 청구 도구(예: Cost Explorer 및)에서 크기 조정 태그를 인식하고 필터링할 수 있습니다 AWS Cost and Usage Report. 이를 활성화하지 않으면 태그 필터링 옵션과 데이터를 사용할 수 없습니다. 비용 할당 태그 사용에 대한 자세한 내용은 설명서의 사용자 정의 비용 할당 태그 활성화를 AWS 결제 및 비용 관리 참조하세요.

1. 에 로그인 AWS Management Console 하고 AWS Billing 콘솔을 엽니다.
2. 탐색 창의 결제 섹션에서 비용 할당 태그를 선택합니다.
3. 사용자 정의 비용 할당 태그 탭에서 크기 조정을 입력합니다.
4. 크기 조정 태그 키를 선택한 다음 활성화를 선택합니다.

24시간이 지나면 태그가 Cost Explorer에 나타나야 합니다.

Systems Manager Automation을 사용하여 올바른 크기 조정 권장 사항 구현

크기 조정은 인스턴스를 중지하고 시작해야 하는 시나리오입니다. 이 시나리오에서는 유지 관리 기간에 이러한 중단을 처리해야 할 수 있으며 자체 크기 조정을 처리하기 위해 다른 팀이 필요할 수 있습니다. 인스턴스 유형을 변경하기 전에 Amazon EC2 설명서의 호환 가능한 인스턴스 유형에 대한 고려 사항을 검토하세요.

이 섹션의 예제 단계에서는 AWS-ResizeInstance라는 Systems Manager Automation 문서를 사용하여 계정 및 리전별로 올바른 크기 조정 권장 사항을 구현합니다. 이 접근 방식은 대부분의 조직에서 다양한 목적으로 다양한 인스턴스 유형을 요구하므로 대부분의 조직에서 일반적입니다. 동일한 `AWS-ResizeInstance` 자동화 문서를 사용하여 단일 및 다중 계정 배포를 대상으로 지정할 수도 있습니다.

1. 에 로그인 AWS Management Console 하고 Systems Manager 콘솔을 엽니다.
2. 탐색 창의 공유 리소스 섹션에서 문서를 선택합니다.
3. 검색 창에 AWS-ResizeInstance를 입력한 다음 검색 결과에서 AWS-ResizeInstance를 선택합니다.
4. 자동화 실행(Execute automation)을 선택합니다.
5. 자동화 실행서 페이지에서 단순 실행을 선택합니다.

6. 입력 파라미터 섹션에서 InstanceId 및 InstanceType 입력합니다. 나머지 기본값을 유지합니다.
7. 실행 을 선택한 다음 자동화가 인스턴스 유형을 변경하는 단계를 거칠 때까지 기다립니다.

대체 크기 조정 방법 고려

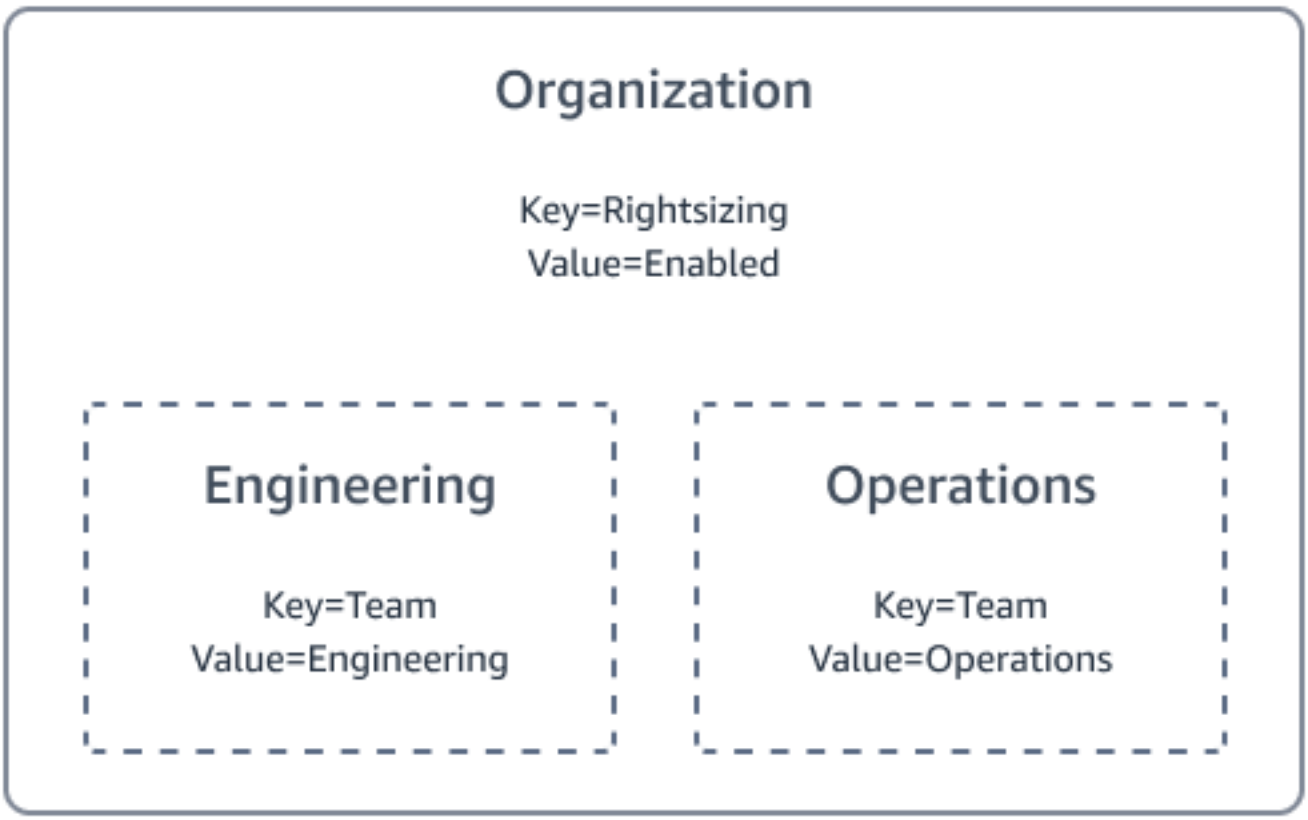
시작 템플릿을 사용하여 인스턴스를 배포하는 경우 시작 템플릿을 적절한 크기의 인스턴스 유형으로 업데이트한 다음 인스턴스 새로 고침을 수행하여 인스턴스를 적절한 크기의 버전으로 바꿀 수 있습니다.

여러 계정 및 리전에서 올바른 크기 조정 프로세스를 사용하려는 경우 사용자 지정 Systems Manager Automation 문서를 생성해야 합니다. 이 문서를 사용하면 여러 인스턴스에서 동일한 대상 인스턴스 유형으로 이동하는 파라미터 및 대상 인스턴스로 피드할 수 있습니다(예: 소스 인스턴스 유형에 관계없이 t3a.medium으로 전환되는 모든 인스턴스).

Cost Explorer에서 비용 전후 검토

리소스의 크기를 올바르게 조정한 후에는 Cost Explorer를 사용하여 적정 크기 조정 태그를 사용하여 비용 전후를 표시할 수 있습니다. 리소스 태그를 사용하여 비용을 추적할 수 있다는 점을 기억하세요. 여러 계층의 태그를 사용하면 비용을 세부적으로 파악할 수 있습니다. 이 가이드에서 다루는 예제에서 Rightsizing 태그는 모든 대상 인스턴스에 일반 태그를 적용하는 데 사용됩니다. 그런 다음 팀 태그를 사용하여 리소스를 추가로 구성합니다. 다음 단계는 애플리케이션 태그를 도입하여 특정 애플리케이션 운영에 따른 비용 영향을 추가로 보여주는 것입니다.

다음 다이어그램은 조직의 태그 구조를 보여줍니다.



운영 팀이 소유한 프로덕션 웹 서버의 크기를 적절하게 조정하는 비즈니스의 예를 생각해 보세요. Cost Explorer에서 크기 조정 태그는 활성화됨으로 설정되고 팀 태그는 작업으로 설정됩니다. 이 예제에서 적절한 크기 조정 작업은 운영 비용을 시간당 0.89센트에서 0.28센트로 줄입니다. 매월 744시간을 가정할 때 올바른 크기 조정 전 연간 비용은 7,945.92 USD입니다. 크기 조정을 마쳤을 때 연간 비용은 2,499.84 USD로 떨어집니다. 따라서 연간 워크로드 비용이 68.5% 감소합니다. 대규모 조직 전체에서 이 문제가 미치는 영향을 상상해 보세요. 이 작업은 샘플 환경에서 수행되며 인스턴스는 대부분 유휴 상태입니다. 프로덕션 환경에서는 10~35%의 절감 효과를 볼 수 있습니다.

이제 엔지니어링 팀이 소유한 프로덕션 접속 호스트의 올바른 크기 조정이 미치는 영향을 고려하세요. Cost Explorer에서 크기 조정 태그는 활성화됨으로 설정되고 팀 태그는 엔지니어링으로 설정됩니다. 이 예제에서는 적절한 크기 조정을 통해 시간당 0.75센트에서 0.44센트로 비용이 절감됩니다. 매월 744시간을 가정하면 올바른 크기 조정 전 연간 비용은 6,696.00 USD입니다. 크기 조정 후 연간 비용은 3,928.32 USD로 떨어집니다.

여러 태그를 사용하는 경우 데이터를 세분화된 비용 세부 정보로 필터링할 수 있습니다. 이 예제에서는 팀 태그가 노이즈를 줄여 팀 수준에서 영향을 볼 수 있습니다. 크기 조정 태그가 활성화되어 있으므로 값이 활성화되었거나 값이 없는 해당 태그가 있는 인스턴스를 필터링할 수도 있습니다. 이는 특히 Cost Explorer 수준의 관리 계정(지급인)에서 볼 때 적절한 크기 조정 노력을 전체적으로 파악할 수 있습니다. 이 보기를 사용하면 모든 계정과 인스턴스를 볼 수 있습니다.

크기 조정 태그가 활성화로 설정된 단일 계정 수준의 예를 생각해 보세요. 운영 비용은 시간당 1.64 USD에서 시간당 0.72센트로 떨어집니다. 매월 744시간을 가정할 때 올바른 크기 조정 전 연간 비용은 14,641.92 USD입니다. 크기 조정을 마쳤을 때 연간 비용은 6,428.16 USD로 떨어집니다. 따라서이 계정의 컴퓨팅 비용이 56% 감소합니다.

적절한 크기 조정 여정을 시작하기 전에 다음 사항을 고려하세요.

- AWS 는 비용 절감을 위한 다양한 옵션을 제공합니다. 여기에는가 이동하기 전에 온프레미스 인스턴스를 AWS 검토하는 AWS OLA가 포함됩니다 AWS. 또한 AWS OLA는 적절한 크기 조정 권장 사항 및 라이선스 지침을 제공합니다.
- Savings Plans을 구매하기 전에 적절한 크기 조정을 모두 완료합니다. 이렇게 하면 Savings Plans 약정에서 과다 구매를 방지하는 데 도움이 될 수 있습니다.

추천

다음 단계를 수행하는 것이 좋습니다.

1. 기존 환경을 검토하고 Amazon EBS gp2 볼륨을 gp3 볼륨으로 변환하는 것을 고려합니다.
2. Savings Plans 검토합니다.

추가 리소스

- AWS Compute Optimizer (AWS 설명서)
- AWS 리소스 태그 지정 모범 사례(AWS 백서)
- 에서 AWS Compute Optimizer 또는 AWS Trusted Advisor 에서 데이터를 수집하는 방법 AWS Organizations(YouTube)
- 성능 최적화 및 라이선스 비용 절감: AWS Compute Optimizer Amazon EC2 SQL Server 인스턴스 활용(AWS 블로그의 Microsoft 워크로드)

Windows 워크로드에 적합한 인스턴스 유형 선택

개요

온프레미스 환경과 비교하여 클라우드에서 작동하는 워크로드의 중요한 차이점은 오버프로비저닝입니다. 온프레미스 사용을 위해 물리적 하드웨어를 구매할 때 미리 정해진 기간, 일반적으로 3~5년 동안

지속될 것으로 예상되는 자본 지출을 합니다. 하드웨어 수명 동안 예상되는 증가를 수용하기 위해 하드웨어는 현재 워크로드에 필요한 것보다 많은 리소스를 사용하여 획득됩니다. 따라서 물리적 하드웨어는 실제 워크로드의 요구 사항을 훨씬 초과하여 과다 프로비저닝되는 경우가 많습니다.

가상 머신(VM) 기술은 잉여 하드웨어 리소스를 활용하는 효과적인 수단으로 부상했습니다. 관리자가 vCPUs 및 RAM으로 VMs을 과도하게 프로비저닝하여 하이퍼바이저가 각 VM에 미사용 리소스를 할당하여 사용량이 많은 서버와 유휴 서버 간의 물리적 리소스 사용량을 관리할 수 있습니다. VMs을 관리할 때 각 VM에 할당된 vCPU 및 RAM 리소스는 실제 사용량의 지표가 아닌 리소스 조절자 역할을 더 많이 했습니다. VM 리소스 과다 할당은 사용 가능한 컴퓨팅 리소스의 3배를 쉽게 초과할 수 있습니다.

Amazon Elastic Compute Cloud(Amazon EC2)는 기본 하드웨어에서 VMs을 과도하게 프로비저닝하지 않습니다. 불필요하기 때문입니다. 클라우드 컴퓨팅은 자본 비용이 아닌 운영 비용이며 사용한 만큼만 비용을 지불합니다. 워크로드에 향후 더 많은 리소스가 필요한 경우 선제적으로 프로비저닝하는 대신 실제로 필요할 때 프로비저닝합니다.

올바른 Amazon EC2 인스턴스 유형을 선택할 수 있는 수백 가지 옵션이 있습니다. Windows 워크로드를 클라우드로 마이그레이션하려는 경우는 현재 워크로드를 더 잘 이해하고에서 성능의 예를 제공하는 데 도움이 되는 AWS OLA를 AWS 제공합니다 AWS. AWS OLA 분석은 적절한 EC2 인스턴스 유형 및 크기를 실제 온프레미스 사용량에 맞추는 것을 목표로 합니다.

Amazon EC2에서 이미 실행 중인 워크로드가 있고 비용 최적화 전략을 찾고 있는 경우, 가이드의이 섹션에서는 Amazon EC2 인스턴스와 일반적인 Windows 워크로드에 대한 적용 가능성 간의 차이를 식별하는 데 도움이 됩니다.

비용 최적화 권장 사항

EC2 인스턴스 유형의 비용을 최적화하려면 다음을 수행하는 것이 좋습니다.

- 워크로드에 적합한 인스턴스 패밀리 선택
- 프로세서 아키텍처 간의 가격 차이 이해
- EC2 세대 전반의 가격 대비 성능 차이 이해
- 최신 인스턴스로 마이그레이션
- 버스트 가능 인스턴스 사용

워크로드에 적합한 인스턴스 패밀리 선택

워크로드에 적합한 인스턴스 패밀리를 선택하는 것이 중요합니다.

Amazon EC2 인스턴스는 다음과 같은 다양한 그룹으로 나뉩니다.

- 범용
- 컴퓨팅 최적화
- 메모리 최적화
- 액셀러레이티드 컴퓨팅
- 스토리지 최적화
- HPC 최적화

대부분의 Windows 워크로드는 다음 범주에 속합니다.

- 범용
- 컴퓨팅 최적화
- 메모리 최적화

이를 더욱 단순화하려면 각 범주에서 기준 EC2 인스턴스를 고려하세요.

- 컴퓨팅 최적화 - C6i
- 범용 - M6i
- 메모리 최적화 - R6i

이전 세대의 EC2 인스턴스는 프로세서 유형에서 약간의 차이를 보였습니다. 예를 들어 C5 컴퓨팅 최적화 인스턴스는 M5 범용 인스턴스 또는 R5 메모리 최적화 인스턴스보다 프로세서 속도가 빠릅니다. 최신 세대의 EC2 인스턴스(C6i, M6i, R6i, C6a, M6a 및 R6a)는 모두 인스턴스 패밀리 간에 동일한 프로세서를 사용합니다. 프로세서는 최신 세대의 인스턴스 간에 일관되므로 이제 인스턴스 패밀리 간의 가격 차이는 RAM 양에 따라 달라집니다. 인스턴스의 RAM이 많을수록 비용이 더 많이 듭니다.

다음 예제는 `us-east-1` 리전에서 실행되는 Intel 기반 4 vCPU 인스턴스의 시간당 요금을 보여줍니다.

Instance	vCPU	RAM	시간당 가격
c6i.xlarge	4	8	0.17 USD
m6i.xlarge	4	16	0.19 USD

Instance	vCPU	RAM	시간당 가격
r6i.xlarge	4	32	0.25 USD

> **Note**
>
> 요금은 us-east-1 리전의 온디맨드 시간당 요금을 기준으로 합니다.

버스트 가능한 인스턴스

요금을 방지하기 위해 사용하지 않는 컴퓨팅 리소스를 끄는 것이 클라우드 컴퓨팅의 모범 사례이지만 필요할 때마다 모든 워크로드를 끄거나 켤 수 있는 것은 아닙니다. 일부 워크로드는 장기간 유휴 상태로 유지되지만 하루 24시간 액세스할 수 있어야 합니다.

버스트 가능 인스턴스(T3)는 컴퓨팅 비용을 낮게 유지하면서 급증하거나 사용률이 낮은 워크로드를 하루 종일 온라인으로 유지하는 방법을 제공합니다. 버스트 가능한 EC2 인스턴스에는 인스턴스가 짧은 기간 동안 사용할 수 있는 최대 vCPU 리소스가 있습니다. 이러한 인스턴스는 버스트 가능한 CPU 크레딧을 기반으로 시스템을 사용합니다. 이러한 크레딧은 하루 종일 유휴 기간에 누적됩니다. 버스트 가능한 인스턴스는 다양한 vCPU-to-RAM 비율을 제공하므로 경우에 따라 컴퓨팅 최적화 인스턴스와 다른 경우에 다른 범용 인스턴스를 대체할 수 있습니다.

다음 예제는 us-east-1 리전에서 실행되는 T3 인스턴스(즉, 버스트 가능 인스턴스)의 시간당 요금을 보여줍니다.

Instance	vCPU	RAM(GB)	시간당 가격
t3.nano	2	0.5	$0.0052
t3.micro	2	1	0.0104 USD
t3.small	2	2	0.0208 USD
t3.medium	2	4	$0.0416
t3.large	2	8	0.0832 USD
t3.xlarge	4	16	$0.1664

Instance	vCPU	RAM(GB)	시간당 가격
t3.2xlarge	8	32	0.3328 USD

> **Note**
>
> 요금은 us-east-1 리전의 온디맨드 시간당 요금을 기준으로 합니다.

프로세서 아키텍처 간의 가격 차이 이해

Intel 프로세서는 시작 이후 EC2 인스턴스의 표준이 되었습니다. C5, M5, R5와 같은 이전 세대의 EC2 인스턴스는 Intel을 프로세서 아키텍처로 표시하지 않습니다(기본값). C6i, M6i, R6i와 같은 최신 EC2 인스턴스에는 Intel 프로세서 사용을 나타내는 "i"가 포함되어 있습니다.

프로세서 아키텍처 주석의 변경 사항은 추가 프로세서 옵션이 도입되었기 때문입니다. Intel과 가장 유사한 프로세서는 AMD("a"로 표시됨)입니다. AMD EPYC 프로세서는 동일한 x86 아키텍처를 사용하며 Intel 프로세서와 비슷하지만 저렴한 가격으로 성능을 제공합니다. 다음 요금 예제에서 볼 수 있듯이 AMD EC2 인스턴스는 Intel 인스턴스에 비해 컴퓨팅 비용을 약 10% 할인합니다.

인텔 인스턴스	시간당 가격	AMD 인스턴스	가격	% 차이
c6i.xlarge	0.17 USD	c6a.xlarge	0.153 USD	10%
m6i.xlarge	0.192 USD	m6a.xlarge	0.1728 USD	10%
r6i.xlarge	0.252 USD	r6a.xlarge	0.2268 USD	10%

> **Note**
>
> 요금은 us-east-1 리전의 온디맨드 시간당 요금을 기준으로 합니다.

세 번째 주요 프로세서 아키텍처 옵션은 EC2 인스턴스의 AWS Graviton 프로세서("g"로 표시됨)입니다. 에서 설계한 AWS Graviton 프로세서는 Amazon EC2에서 최상의 가격 대비 성능을 제공합니다. 현재 Graviton 프로세서는 Intel 프로세서보다 20% 저렴할 뿐만 아니라 20% 이상의 성능 향상을 제공합

니다. 차세대 Graviton 프로세서는 이러한 성능 차이를 더욱 확장할 것으로 예상되며 테스트 결과 성능이 25% 더 향상되었습니다.

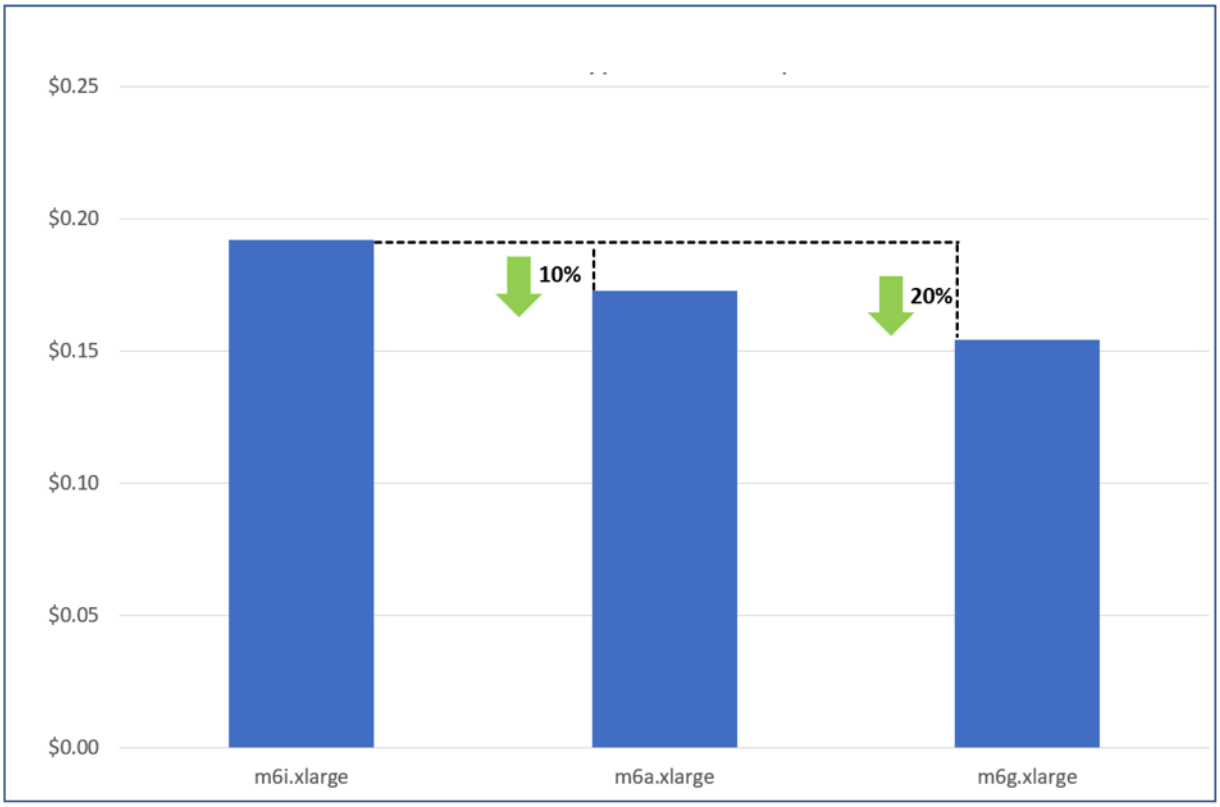
ARM 아키텍처를 기반으로 하는 Graviton 프로세서에서는 Windows Server를 실행할 수 없습니다. 실제로 Windows Server는 x86 프로세서에서만 작동합니다. Windows Server용 Graviton 기반 인스턴스를 사용하면 40%의 가격 대비 성능 향상을 달성할 수 없지만 특정 Microsoft 워크로드와 함께 Graviton 프로세서를 계속 사용할 수 있습니다. 예를 들어 Linux에서 최신 버전의 .NET을 실행할 수 있습니다. 즉, 이러한 워크로드는 ARM 프로세서를 사용할 수 있으며 더 빠르고 저렴한 Graviton EC2 인스턴스의 이점을 누릴 수 있습니다.

다음 예제에서는 `us-east-1` 리전에서 실행 중인 Graviton 인스턴스의 시간당 요금을 보여줍니다.

인텔 인스턴스	시간당 가격	Graviton 인스턴스	시간당 가격	% 차이
c6i.xlarge	0.17 USD	c6g.xlarge	0.136 USD	20%
m6i.xlarge	0.192 USD	m6g.xlarge	0.154 USD	20%
r6i.xlarge	0.252 USD	r6g.xlarge	$0.2016	20%

> **Note**
>
> 요금은 `us-east-1` 리전의 온디맨드 시간당 요금을 기준으로 합니다.

다음 차트에서는 M 시리즈 인스턴스의 가격을 비교합니다.



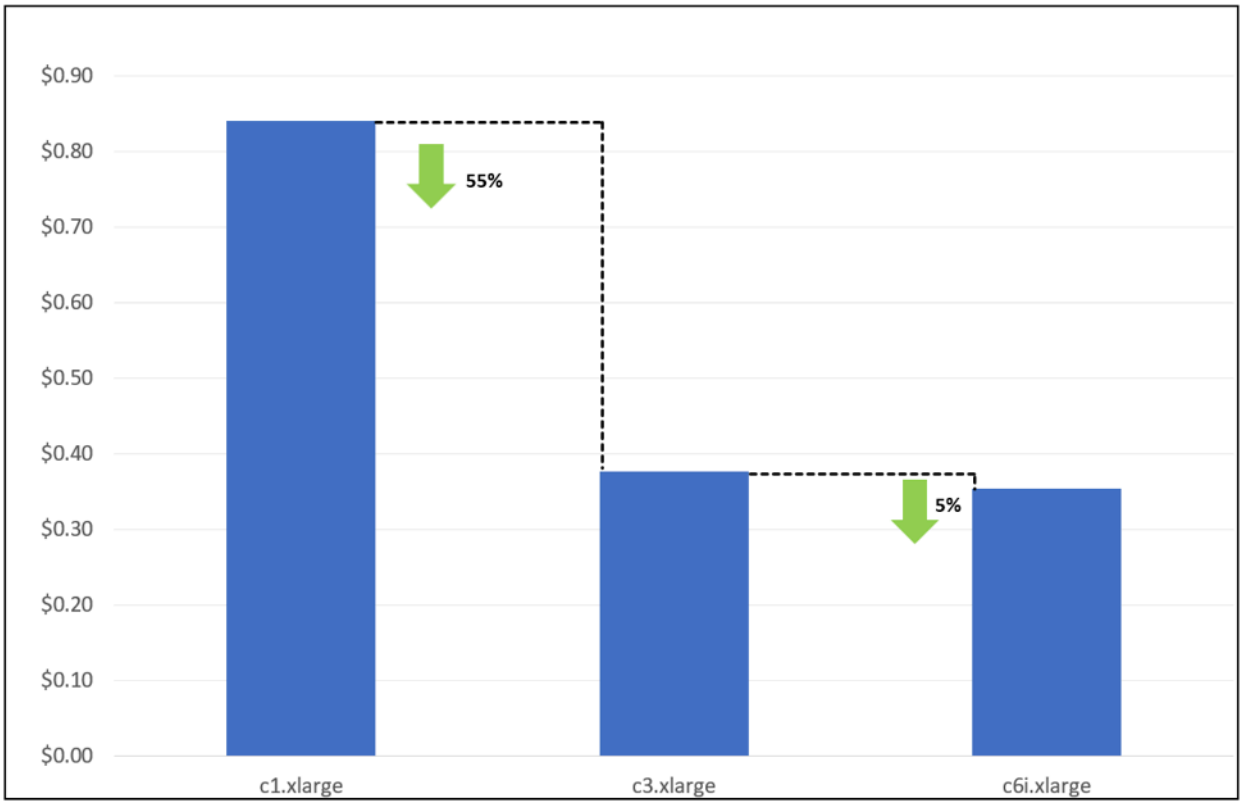
EC2 세대 간 가격 성능 차이 이해

Amazon EC2의 가장 일관된 특성 중 하나는 각 새 세대가 이전 세대보다 더 나은 가격 대비 성능을 제공한다는 것입니다. 다음 표에서 볼 수 있듯이 이후 릴리스마다 최신 세대 EC2 인스턴스의 가격이 감소합니다.

컴퓨팅 최적화 인스턴스	시간당 가격	범용 인스턴스	시간당 가격	메모리 최적화 인스턴스	시간당 가격
C1.xlarge	0.52 USD	M1.xlarge	0.35 USD	r1.xlarge	해당 사항 없음
C3.xlarge	0.21 USD	M3.xlarge	0.266 USD	r3.xlarge	0.333 USD
C5.xlarge	0.17 USD	M5.xlarge	0.192 USD	r5.xlarge	0.252 USD

> **Note**
>
> 요금은 us-east-1 리전의 온디맨드 시간당 요금을 기준으로 합니다.

다음 차트에서는 다양한 세대의 C 시리즈 인스턴스 비용을 비교합니다.



그러나 다음 표와 같이 6세대 인스턴스는 5세대 인스턴스와 가격이 동일합니다.

컴퓨팅 최적화 인스턴스	시간당 가격	범용 인스턴스	시간당 가격	메모리 최적화 인스턴스	시간당 가격
C5.xlarge	0.17 USD	M5.xlarge	0.192 USD	r5.xlarge	0.252 USD
C6i.xlarge	0.17 USD	M6i.xlarge	0.192 USD	r6i.xlarge	0.252 USD

> **Note**
>
> 요금은 us-east-1 리전의 온디맨드 시간당 요금을 기준으로 합니다.

비용은 동일하지만 최신 세대는 더 빠른 프로세서, 향상된 네트워킹 처리량, Amazon Elastic Block Store(Amazon EBS) 처리량 및 IOPS 증가로 인해 우수한 가격 대비 성능을 제공합니다.

가장 중요한 가격 대비 성능 개선 사항 중 하나는 X2i 인스턴스의 개선 사항입니다. 이 세대의 인스턴스는 이전 세대보다 최대 55% 더 높은 가격 대비 성능을 제공합니다. 다음 표에서 볼 수 있듯이 x2iedn은 모든 성능 측면(모두 이전 세대와 동일한 가격)의 개선을 보여줍니다.

Instance	시간당 가격	vCPU	RAM	프로세서 속도	인스턴스 스토리지	네트워킹	Amazon EBS 처리량	EBS IOPS
x1e.2xlarge	1.66 USD	8	244	2.3GHz	237GB SSD	10Gbps	125MB/s	7400
x1iedn.2xlarge	1.66 USD	8	256	3.5GHz	240GB NVMe SSD	25Gbps	2,500MB/s	65000

> **Note**
>
> 요금은 `us-east-1` 리전의 온디맨드 시간당 요금을 기준으로 합니다.

예제 시나리오

배송 차량을 추적하고 SQL Server 성능을 개선하려는 분석 회사의 예를 생각해 보세요. MACO SME 가이 회사의 성능 병목 현상을 검토한 후 회사는 x1e.2xlarge 인스턴스에서 x2iedn.xlarge 인스턴스로 전환합니다. 새 인스턴스 크기는 더 작지만 x2 인스턴스의 개선 사항을 통해 버퍼 풀 확장을 사용하여 SQL Server 성능과 최적화를 높일 수 있습니다. 이를 통해 회사는 SQL Server Enterprise Edition에서 SQL Server Standard Edition으로 다운그레이드할 수 있습니다. 또한 회사는 SQL Server 라이선스를 8개의 vCPUs에서 4개의 vCPUs.

최적화 전:

Server	EC2 인스턴스	SQL Server 에디션	월별 비용
ProdDB1	x1e.2xlarge	엔터프라이즈	3,918.64 USD

Server	EC2 인스턴스	SQL Server 에디션	월별 비용
ProdDB2	x1e.2xlarge	엔터프라이즈	3,918.64 USD
합계			7,837.28 USD

최적화 후:

Server	EC2 인스턴스	SQL Server 에디션	월별 비용
ProdDB1	x2iedn.xlarge	표준	1,215.00 USD
ProdDB2	x2iedn.xlarge	표준	1,215.00 USD
합계			2,430.00 USD

x1e.2xlarge 인스턴스에서 x2iedn.xlarge 인스턴스로 모두 합쳐지면 예제 시나리오에서 회사는 프로덕션 데이터베이스 서버에 매월 5,407 USD를 절감할 수 있습니다. 이렇게 하면 워크로드의 총 비용이 69% 감소합니다.

> **Note**
>
> 요금은 `us-east-1` 리전의 온디맨드 시간당 요금을 기준으로 합니다.

최신 인스턴스로 마이그레이션

이전 세대의 Amazon EC2는 Xen 하이퍼바이저에서 실행되는 반면, 최신 세대는 AWS Nitro 시스템에서 작동합니다. Nitro 시스템은 호스트 하드웨어의 거의 모든 컴퓨팅 및 메모리 리소스를 인스턴스에 전달합니다. 이로 인해 전반적인 성능이 향상됩니다. Xen에서 Nitro 기반 인스턴스로 마이그레이션할 때는 특별한 고려 사항이 있습니다. 예를 들어 AWS Windows AMIs는 Microsoft 설치 미디어에서 사용하는 기본 설정 및 사용자 지정으로 구성됩니다. 사용자 지정에는 최신 세대 인스턴스 유형(Nitro 시스템에 구축된 인스턴스)을 지원하는 드라이버 및 구성이 포함됩니다.

2018년 8월 이전에 생성된 사용자 지정 Windows AMIs 또는 Amazon에서 제공한 Windows AMIs에서 인스턴스를 시작하는 경우 Amazon EC2 설명서의 최신 세대 인스턴스 유형으로 마이그레이션 단계를 완료하는 것이 좋습니다.

버스트 가능 인스턴스 사용

버스트 가능 인스턴스는 컴퓨팅 비용을 절감하는 좋은 방법이지만 다음 시나리오에서는 사용하지 않는 것이 좋습니다.

- 데스크톱 환경을 사용하는 Windows Server의 최소 사양에는 2GB의 RAM이 필요합니다. 최소 RAM 양이 부족하므로 Windows Server에서 t3.micro 또는 t3.nano 인스턴스를 사용하지 마세요.
- 워크로드가 급증하지만 버스트 크레딧을 빌드할 만큼 유휴 상태를 오래 유지하지 않는 경우, 일반 EC2 인스턴스를 사용하는 것이 버스트 가능 인스턴스를 사용하는 것보다 더 효율적입니다. CPU 크레딧을 모니터링하여 이를 확인하는 것이 좋습니다.
- 대부분의 시나리오에서 SQL Server와 함께 버스트 가능 인스턴스를 사용하지 않는 것이 좋습니다. SQL Server에 대한 라이선스는 인스턴스에 할당된 vCPUs 수를 기반으로 합니다. SQL Server가 대부분의 시간 동안 유휴 상태인 경우 완전히 활용하지 않는 SQL 라이선스에 대한 비용을 지불하게 됩니다. 이러한 시나리오에서는 여러 SQL Server 인스턴스를 더 큰 서버로 통합하는 것이 좋습니다.

다음 단계

Amazon EC2 Windows 인스턴스에 대한 비용을 최적화하려면 다음 단계를 수행하는 것이 좋습니다.

- 최고 가격 성능을 위해 최신 세대 EC2 인스턴스를 사용합니다.
- EC2 인스턴스를 AMD 프로세서와 함께 사용하면 컴퓨팅 비용을 10% 절감할 수 있습니다.
- 워크로드와 일치하는 EC2 인스턴스 유형을 선택하여 리소스 사용률을 극대화합니다.

다음 표에는 Windows 워크로드의 일반적인 시작점 예제가 나와 있습니다. SQL Server 워크로드를 개선하기 위한 인스턴스 스토리지 볼륨 또는 훨씬 더 큰 vCPU-to-RAM 비율로 EC2 인스턴스와 같은 추가 옵션을 사용할 수 있습니다. 워크로드를 철저히 테스트하고와 같은 모니터링 도구를 사용하여 필요한 조정 AWS Compute Optimizer 을 수행하는 것이 좋습니다.

워크로드	일반적인	선택 사항
Active Directory	T3, M6i	R6i
파일 서버	T3, M6i	C6i
웹 서버	T3, C6i	M6i, R6i
SQL Server	R6i	x2iedn, X2iezn

EC2 인스턴스 유형을 변경해야 하는 경우 프로세스에는 일반적으로 간단한 서버 재부팅만 포함됩니다. 자세한 내용은 Amazon EC2 설명서의 인스턴스 유형 변경을 참조하세요.

인스턴스 유형을 변경하기 전에 다음 사항을 고려하는 것이 좋습니다.

- 인스턴스 유형을 변경하려면 먼저 Amazon EBS에서 지원하는 인스턴스를 중지해야 합니다. 인스턴스가 중지된 동안 가동 중지 시간을 계획해야 합니다. 인스턴스 중단하고 인스턴스 유형을 변경하는 것은 몇 분이 걸릴 수 있으며, 인스턴스를 다시 시작하는 시간은 애플리케이션의 시작 스크립트에 따라 달라질 수 있습니다. 자세한 내용은 Amazon EC2 설명서의 인스턴스 중지 및 시작을 참조하세요.
- 인스턴스를 중지하고 시작하면가 인스턴스를 새 하드웨어로 AWS 이동합니다. 인스턴스에 퍼블릭 IPv4 주소가 있는 경우는 주소를 AWS 해제하고 인스턴스에 새 퍼블릭 IPv4 주소를 부여합니다. 변경되지 않는 퍼블릭 IPv4 주소가 필요한 경우 탄력적 IP 주소를 사용합니다.
- 인스턴스에서 최대 절전 모드가 활성화된 경우 인스턴스 유형을 변경할 수 없습니다.
- 스팟 인스턴스의 인스턴스 유형은 변경할 수 없습니다.
- 인스턴스가 Auto Scaling 그룹에 있는 경우 Amazon EC2 Auto Scaling은 중지된 인스턴스를 비정상으로 표시하고 인스턴스를 종료하고 대체 인스턴스를 시작할 수 있습니다. 이를 방지하기 위해서는 인스턴스 유형을 변경하는 동안 그룹에 대한 조정 프로세스를 일시 중지할 수 있습니다. 자세한 내용은 Amazon EC2 Auto Scaling 설명서의 Auto Scaling 그룹에 대한 프로세스 일시 중지 및 재개를 참조하세요. Auto Scaling
- NVMe 인스턴스 스토어 볼륨이 있는 인스턴스의 인스턴스 유형을 변경하면 Amazon Machine Image(AMI) 또는 인스턴스 블록 디바이스 매핑에 지정되지 않은 경우에도 모든 NVMe 인스턴스 스토어 볼륨을 사용할 수 있으므로 업데이트된 인스턴스에 추가 인스턴스 스토어 볼륨이 있을 수 있습니다. 그렇지 않으면 업데이트한 인스턴스는 원본 인스턴스를 시작할 때 지정한 것과 동일한 수의 인스턴스 스토어 볼륨을 갖습니다.

추가 리소스

- Amazon EC2 인스턴스 유형(AWS 문서)
- AWS 최적화 및 라이선스 평가(AWS 설명서)

Windows 및 SQL Server 워크로드에 대한 라이선스 가져오기

개요

Microsoft 워크로드 및 기존 엔터프라이즈 라이선스 계약에 상당한 투자를 하는 경우 포함된 라이선스(에서 제공 AWS) 및 기존 보유 라이선스 사용(BYOL) AWS 옵션을 포함하여 이러한 워크로드를 지

원하는 여러 옵션 중에서 선택할 수 있습니다. https://aws.amazon.com/windows/faq/#byol Amazon EC2 전용 호스트를 사용하여 기존 Microsoft 라이선스 계약을 완전히 활용하고 Windows Server를 로 가져올 수 있습니다 AWS. 이렇게 하면 Amazon EC2 인스턴스 비용을 최대 50% 절감할 수 있습니다. Windows 라이선스는 인스턴스 비용의 약 절반을 차지하므로 전용 호스트 AWS 의 로 Windows Server 를 가져오면 상당한 비용 절감 효과를 얻을 수 있습니다. Windows Server를 기본(공유) 테넌시로 가져올 수 없으므로 전용 호스트는 Windows Server에 대한 기존 라이선스를 사용하려는 경우에 적합합니다 AWS.

전용 호스트는 Windows Server BYOL 인스턴스용이 아닙니다. 또한 기존 SQL Server 워크로드에 대한 온프레미스 라이선스를 유연하게 일치시킬 수 있습니다. 전용 호스트는 기본 서버의 물리적 코어를 노출하고 물리적 코어 수준에서 SQL Server 라이선스를 부여할 수 있습니다. SQL Server 라이선스가 인스턴스에 할당된 가상 CPUs 수를 기반으로 하는 기본(공유) 테넌시에서는 이것이 불가능합니다. 이 기능을 사용하면 온프레미스 라이선스 전략과 일치하는 방식으로에서 SQL Server 워크로드 AWS 에 라이선스를 부여할 수 있습니다. 따라서 적격 Windows 라이선스를 사용하면 인스턴스 비용 절감 외에도 기본(공유) 테넌시와 비교하여 SQL Server 라이선스 비용을 최대 50% 절감할 수 있습니다. 이 시나리오에 대한 자세한 내용은이 가이드의 SQL Server 라이선스 이해 섹션을 참조하세요.

Amazon EC2 전용 호스트

Amazon EC2 전용 호스트는 기본적으로가 EC2 컴퓨팅 오퍼링을 실행하는 데 AWS 사용하는 것과 동일한 EC2 호스트입니다. 차이점은 이러한 호스트가 단일 고객 전용이며 기본 물리적 인프라에 대한 독점적 액세스를 제공한다는 것입니다. 전용 호스트를 사용하면 다른 AWS 고객과 리소스를 공유하는 대신 전적으로 전용 하드웨어에서 인스턴스를 실행할 수 있습니다. 이를 통해 클라우드 리소스를 더 잘 제어할 수 있으며 Windows Server 및 SQL Server와 같은 자체 소프트웨어 라이선스를에 가져와 비용을 절감할 수 있습니다 AWS.

다음 사항에 유의하세요.

- 전용 호스트는 단일 고객 전용 물리적 서버입니다. 전용 호스트의 소켓 및 물리적 코어를 볼 수 있으므로 소켓별, 코어별 또는 VM별 소프트웨어 라이선스 계약과 같은 라이선스 규정 준수 요구 사항을 해결할 수 있습니다.
- 동일한 인스턴스 패밀리의 여러 인스턴스 크기를 지원할 수 있는 전용 호스트를 이기종 전용 호스트라고 합니다. 이러한 인스턴스 패밀리에는 T3, A1, C5, M5, R5, C5n, R5n 및 M5n이 포함됩니다. 반면 다른 인스턴스 패밀리는 동일한 전용 호스트에서 하나의 인스턴스 크기만 지원합니다. 이를 동종 전용 호스트라고 합니다.
- 전용 호스트는 호스트별로 요금이 청구됩니다. 즉, 실행 중인 인스턴스 수에 관계없이 전용 호스트당 요금이 부과됩니다. 전용 호스트 요금은 선택한 인스턴스 패밀리, 리전 및 결제 옵션에 따라 다릅니다. 워크로드에 대한 최적의 구성을 선택하여 원하는 성능 및 비용 결과를 달성할 수 있습니다.

이 다이어그램은 공유 테넌시 인스턴스와 전용 호스트 간의 차이점을 보여줍니다.

동종 전용 호스트

M6i 전용 호스트가 사용되는 시나리오를 생각해 보세요. M6i 및 R6i 전용 호스트에는 두 개의 소켓, 64개의 물리적 코어가 있으며 동일한 크기의 인스턴스 유형을 지원합니다. 이를 동종 전용 호스트라고 합니다. 즉, 단일 M6i 전용 호스트에서 시작할 수 있는 인스턴스 수는 인스턴스 크기에 따라 달라집니다.

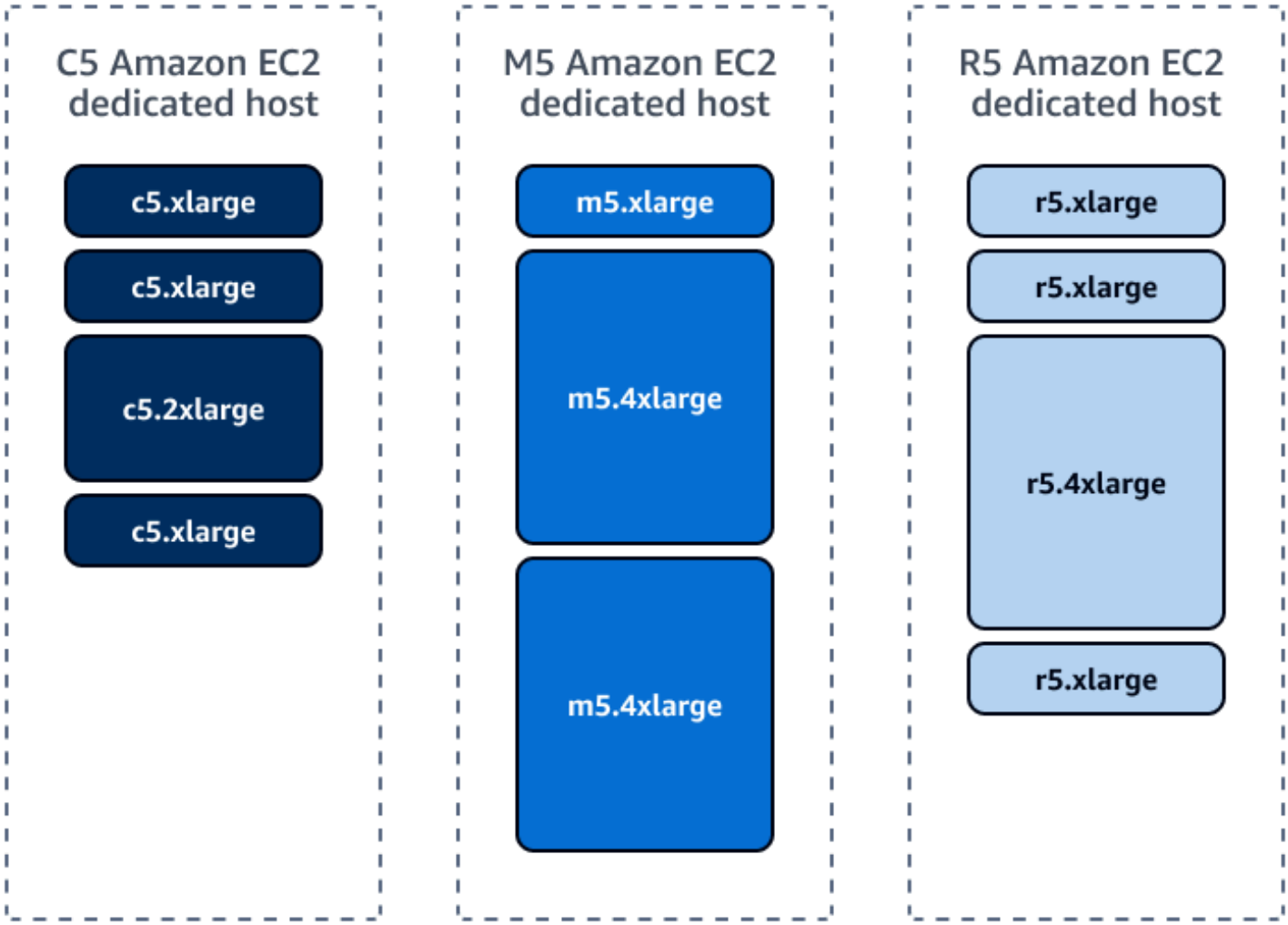
예시:

- xlarge(vCPUs)의 경우이 전용 호스트에서 최대 32개의 m6i.xlarge 인스턴스를 시작할 수 있습니다.
- 8xlarge(vCPUs)의 경우이 전용 호스트에서 최대 4개의 m6i.8xlarge 인스턴스를 시작할 수 있습니다.
- 메탈(vCPUs)의 경우이 전용 호스트에서 최대 1개의 m6i.metal 인스턴스를 시작할 수 있습니다.

다음 다이어그램은 M6 인스턴스에 대한 전용 호스트 옵션을 보여줍니다.

이종 전용 호스트

동일한 호스트에서 여러 인스턴스 크기를 지원하는 전용 호스트를 이종 Amazon EC2 전용 호스트라고 합니다. 다음 다이어그램은 2xlarge, xlarge 및 4xlarge와 같은 다양한 인스턴스 크기를 가진 C5, M5 및 R5 전용 호스트의 예를 보여줍니다.



전용 호스트 관리

Amazon EC2 전용 호스트 관리와 관련하여 다음 사항을 고려하는 것이 좋습니다.

- 전용 호스트를 최대한 활용하려면 조직 내 여러 계정 간에 단일 호스트를 공유할 수 있습니다. 호스트 공유를 사용하면 리소스 최적화가 가능하며 호스트에서 사용 가능한 모든 슬롯을 사용하여 비용을 절감할 수 있습니다. 사업부 간에 전용 호스트를 공유하면 워크로드 간의 분리를 유지하면서 IT 인프라를 중앙 집중화하고 리소스 사용률을 개선할 수 있습니다. 의 조직에 속 AWS Organizations 해 있고 조직 내에서 공유가 활성화된 경우 조직의 소비자에게 공유된 전용 호스트에 대한 액세스 권

한이 자동으로 부여됩니다. 그렇게 하지 않으면 소비자는 리소스 공유에 가입하라는 초대장을 받고 초대를 수락한 후 공유된 전용 호스트의 액세스 권한을 받습니다.

- Windows Server 2019는 BYOL이 가능한 최신 버전이므로 라이선스 포함 모델에서 전용 호스트에서 Windows Server 2022를 실행할 수 있습니다. 전용 호스트에서 Windows Server 2022를 사용하려면 Windows Server 2022 라이선스 포함 인스턴스를 사용해야 합니다.
- AWS License Manager는 AWS 및 온프레미스 환경 전반의 다양한 공급업체에서 소프트웨어 라이선스를 관리하기 위한 포괄적인 솔루션입니다. License Manager를 사용하면 소프트웨어 라이선스 사용 방법을 더 잘 파악하고 제어할 수 있으므로 비용을 절감하고 규정 준수를 개선할 수 있습니다. License Manager를 사용하여 고유한 라이선스 조건을 에뮬레이션하는 규칙을 설정할 수 있습니다. 이를 통해 이러한 규칙을 적용하고 라이선스 오용을 방지할 수 있습니다. 이렇게 하면 규정 미준수 위험을 줄이고 라이선스 관리 프로세스를 개선할 수 있습니다.
- License Manager를 사용하면 호스트 리소스 그룹을 사용하여 호스트의 배치, 릴리스 및 복구를 자동화할 수 있습니다. 이를 통해 생산성을 높이고 관리 오버헤드를 줄일 수 있습니다. 또한 License Manager는 라이선스 규칙에 따라 AWS 및 온프레미스 환경 전반의 라이선스 사용량을 중앙 집중식으로 볼 수 있으므로 조직 전체에서 증분 라이선스 구매, 규정 준수 및 공급업체 감사를 쉽게 관리할 수 있습니다. 또한 License Manager는 AWS Organizations 및 AWS Resource Access Manager (AWS RAM)와 통합되어 계정 및 리전 간에 라이선스 구성을 공유합니다. 이를 통해 일정에 따라 전체 환경에 대한 보고서를 생성하고 라이선스 규칙을 중앙에서 하나의 로 관리할 수 있습니다 AWS 계정. 궁극적으로 거버넌스를 개선하고 복잡성을 줄일 수 있습니다.
- 단일 리전 내에서 전용 호스트의 고가용성을 설계할 때는 프로덕션 크리티컬 워크로드를 위해 최소 2개의 가용 영역에 최소 2개의 전용 호스트를 할당했는지 확인합니다. 자세한 내용은 참조 배포의 Microsoft Windows용 Amazon EC2 전용 호스트를 참조하세요 AWS.
- 각 전용 호스트 인스턴스 패밀리에는 각 인스턴스 크기에 대해 실행할 수 있는 인스턴스 수에 제한이 있습니다. 자세한 내용은 Amazon EC2 설명서의 전용 호스트 구성 테이블을 참조하세요.

AWS 라이선스 옵션

라이선스는 다음과 같은 기본 범주로 분류됩니다.

- 라이선스 포함 -이 라이선스 옵션을 사용하면 온디맨드로 라이선스를 구매하고 사용할 수 있으며 사용한 만큼만 비용을 지불할 수 있습니다. 라이선스 사용에 유연성을 찾고 선결제 비용을 피하려는 사용 사례에 적합합니다. 다양한 Windows Server, SQL Server 및 기타 Microsoft 제품 중에서 선택할 수 있습니다.
- 라이선스 이동성이 있는 BYOL 제품 - 기존 라이선스가 이미 있고 클라우드에서 사용하려는 경우이 라이선스 옵션을 사용하면 Microsoft 라이선스 이동 프로그램을 통해 자체 라이선스를 클라우드로

가져올 수 있습니다. SQL Server with Software Assurance(SA)와 같이 라이선스 이동성이 있는 제품은 공유 또는 전용 테넌시로 가져올 수 있습니다. 이렇게 하면 AWS 인스턴스 비용이 절감됩니다.

- 라이선스 이동성이 없는 BYOL 제품 - 라이선스 이동성이 없는 Windows Server와 같은 Microsoft 제품의 경우는 클라우드에서 이러한 제품을 사용할 수 있는 전용 옵션을 AWS 제공합니다. 또한 전용 호스트를 사용하면 물리적 코어 수준에서 라이선스를 사용할 수 있으므로 워크로드를 실행하는 데 필요한 라이선스를 50% 이상 절약할 수 있습니다. 전용 호스트는 대부분 실행되는 안정적이고 예측 가능한 워크로드에 매우 적합합니다.

Windows Server 라이선스 사용

자체 Windows 라이선스를 사용하는 것은 기존 투자를 활용하고 AWS 비용을 절감할 수 있으므로 라이선스 최적화를 위한 가장 효과적인 전략 중 하나입니다. 특정 BYOL 시나리오에는 SA 또는 라이선스 이동의 이점이 필요하지 않지만 Amazon EC2 전용 인프라는 항상 필요합니다. 자격을 갖추려면 2019년 10월 1일 이전에 영구 라이선스를 구매했거나 2019년 10월 1일 이전에 유효한 기업 등록에 따라 트루업으로 추가해야 합니다. 이러한 특정 BYOL 시나리오에서는 라이선스만 2019년 10월 1일 이전에 사용 가능한 버전으로 업그레이드할 수 있습니다. 예를 들어 2017년에 SA를 삭제한 경우 2019가 아닌 Windows Server 2016까지만 배포할 수 있는 권한이 있습니다. 그러나 2019는 BYOL을 사용할 수 있는 마지막 버전입니다 AWS. 자세한 내용은 AWS 설명서의 라이선싱 – Windows Server를 참조하세요.

라이선스를 가져오면 Microsoft 워크로드 실행 비용에 상당한 영향을 미칠 수 있습니다 AWS. 자체 라이선스를 가져올 때 클라우드에서 실행되는 인스턴스에 대한 추가 라이선스 비용을 지불할 필요가 없으므로 상당한 비용 절감이 발생할 수 있습니다.

다음 표는 다양한 구성에서 단일 c5.xlarge 인스턴스를 연중무휴로 실행하는 데 드는 온디맨드 월별 비용을 보여줍니다.

구성	월별 비용(USD)
Windows Server + SQL Server Enterprise Edition	1,353.00 USD(LI)
Windows Server + SQL Server Standard 에디션	609.00 USD(LI)
Windows Server만 해당	259.00 USD(LI)
컴퓨팅 전용(Linux)	127.00 USD

기존 라이선스를 사용하여 라이선스 비용을 절감하고 전체 AWS 청구서 비용을 절감할 수 있습니다.

Amazon EC2 전용 호스트에서 BYOL을 사용하려면 Windows Server 및 SQL Server용와 같은 자체 소프트웨어 라이선스를 가져와야 합니다. BYOL을 사용하면에서 기존 라이선스를 사용할 수 AWS 있으며 비용을 절감할 수 있습니다. 자체 라이선스를 가져오려면 소프트웨어 공급업체의 라이선스 권한이 있어야 하며 소프트웨어에 대한 설치 미디어 또는 이미지도 제공해야 합니다. 설치 미디어 또는 이미지를 사용하여 전용 호스트에서 인스턴스를 시작할 수 있습니다. BYOL AMI 생성에 대한 자세한 내용은 AWS 블로그의 Microsoft 워크로드에서 VM Import/Export를 사용하여 온프레미스에서 Windows Server Bring-Your-Own-License AMIs를 생성하는 방법을 참조하세요.

> **Note**
>
> Auto로 설정된 라이선스 유형은 AWS 라이선스 포함 옵션과 동일합니다. 이 옵션을 사용하면 원치 않는 온디맨드 지출이 발생할 수 있습니다. 라이선스 유형을 전환해야 합니다.

비용 최적화 시나리오

적절한 라이선스 크기 조정 및 최적화는에서 비용 최적화의 핵심 구성 요소입니다 AWS. 올바른 전략을 구현하면 Amazon EC2 전용 호스트 및 BYOL 옵션을 사용하여 라이선스 비용을 줄이고, 규정 준수를 유지하고, 라이선스 투자에서 가능한 최상의 가치를 얻을 수 있습니다.

이 섹션에서는 다음 예제 시나리오를 다룹니다.

- T3 전용 호스트를 통한 비용 절감
- SQL Server BYOL을 사용하여 공유 테넌시와 전용 호스트 비교
- 고가용성 SQL Server 배포

T3 전용 호스트를 통한 비용 절감

T3 전용 호스트는 전통적으로 고정 CPU 리소스를 제공하는 다른 Amazon EC2 전용 호스트와 다릅니다. 반면 T3 전용 호스트는 CPU 리소스를 공유하고, 기준 CPU 성능을 제공하고, 필요할 때 버스팅할 수 있는 버스트 가능 인스턴스를 지원합니다. 과다 구독이라고도 하는 CPU 리소스를 공유하면 단일 T3 전용 호스트가 유사한 범용 전용 호스트보다 최대 4배 많은 인스턴스를 지원할 수 있습니다.

T3 전용 호스트는 다른 Amazon EC2 전용 호스트보다 더 높은 인스턴스 밀도를 제공하여 TCO를 낮춥니다. 버스트 가능한 T3 인스턴스를 사용하면 이전보다 적은 호스트에서 평균 CPU 사용률이 low-to-moderate 인스턴스 수를 더 많이 통합할 수 있습니다. 또한 T3 전용 호스트는 다른 Amazon EC2 전용 호스트보다 더 많은 수의 vCPU 및 메모리 조합에서 더 작은 인스턴스 크기를 제공합니다. 인스턴스 크

기가 작을수록 TCO가 낮아지고 온프레미스 호스트와 같거나 더 큰 통합 비율을 제공하는 데 도움이 될 수 있습니다.

T3 전용 호스트는 Microsoft Windows 데스크톱, Windows Server, SQL Server, Oracle Databases를 포함하여 low-to-moderate CPU 사용률과 적합한 소켓당, 코어당 또는 VM당 소프트웨어 라이선스를 갖춘 BYOL 소프트웨어를 실행하는 데 가장 적합합니다.

T3 전용 호스트를 사용하여 Windows Server Datacenter 라이선스 축소(코어당)

온프레미스 환경에서는 VMware 호스트의 물리적 CPUs 쉽게 오버구독하고 높은 수준의 통합을 달성할 수 있다는 사실을 활용하고 있습니다.

다음 예제를 살펴보세요. 현재 온프레미스 환경에서 10x36 코어, 384GB RAM VMware 호스트를 사용하고 있습니다. 또한 각 호스트는 평균 CPU 사용률이 낮은 96x2 vCPU, 4GB RAM Windows Server 가상 머신을 실행하고 있습니다.

이제 현재 온프레미스 VMware 호스트에 비해 RAM 양이 두 배 더 많은 T3 전용 호스트로 가상 머신을 이동하여 훨씬 더 높은 수준의 통합을 달성할 수 있습니다. 호스트 비용을 50% 절감하면서 T3 전용 호스트에서 동일한 수의 서버를 실행할 수 있습니다. 이를 통해 Windows Server 라이선스 비용을 33% 줄일 수 있습니다. 다음 표에서는 T3 전용 호스트 사용의 절감 효과를 강조합니다.

	온프레미스 VMware 호스트	T3 전용 호스트	절감
물리적 서버	10	5	
호스트당 물리적 코어 수	36	48	
호스트당 RAM(GB)	384	768	
vCPU 2개, 호스트당 RAM VMs 4GB	96	192	
총 VMs 수	960	960	
총 Windows Server Datacenter 라이선스(코어당) = (서버 수 * 물리적 코어 수)	10 * 36 = 360	5 * 48 = 240	33%

SQL Server BYOL을 사용하여 공유 테넌시와 전용 호스트 비교

Amazon EC2 전용 호스트의 가치를 보여주는 실제 예를 생각해 보세요. 이 시나리오에서 조직은 240개의 코어가 있는 온프레미스 환경에서 SQL Server 워크로드를 실행하고 동일한 워크로드를 비용 효율적으로 배포하려고 합니다 AWS. 이 조직이 자체 라이선스(BYOL)를 제공하는 경우 SA에 대한 비용을 계속 지불하고 코어 수를 줄이면 비용에 직접적인 영향을 미칩니다.

다음 다이어그램은 Microsoft 권한 부여와 SQL Server 간의 AWS 비용 절감을 비교합니다.

Microsoft entitlements (Enterprise Agreements)	
	Number of cores
SQL Server Enterprise edition	208
SQL Server Standard edition	32
Total SA cost	**$341,000**

SQL Server savings with AWS	
AWS shared vCPUs	AWS BYOL/Dedicated Hosts cores
120	96
20	-
$197,418	**$151,355**

AWS 공유 테넌시의 인스턴스 크기를 올바르게 조정하면 SQL Server 라이선스를 140개 코어로 줄일 수 있습니다. 이로 인해 SA 비용은 197,000 USD가 됩니다.

Amazon EC2 전용 호스트를 사용하면 물리적 코어 수준에서 SQL Server에 라이선스를 부여할 수 있습니다. SQL Server 라이선스가 인스턴스에 할당된 vCPUs 수를 기반으로 하는 공유 테넌시에서는 이것이 불가능합니다. 따라서 각각 48개의 코어가 있는 2개의 R5 전용 호스트를 사용하면 공유 테넌시에 필요한 140개의 vCPUs 대신 96개의 코어만 포함하면 됩니다. R5 전용 호스트를 배포하고 물리적 수준에서 워크로드에 라이선스를 부여하면 필요한 수의 SQL Server Enterprise 에디션 라이선스를 96개의 코어로 줄일 수 있습니다. 즉, 라이선스 요구 사항을 충족하고 상당한 비용 절감을 달성하면서 SQL Server 워크로드의 코어(하이퍼 스레딩 고려)를 최대 192개까지 배포할 수 있습니다.

이 경우 조직은 SA 비용으로 연간 약 341,000 USD를 지불합니다. 공유 테넌시의 크기를 조정하면 140개의 vCPUs. Amazon EC2 전용 호스트는 비용을 151,000 USD로 추가로 절감합니다(약 56% 감소).

고가용성 SQL Server 배포

이 예제에서는 다양한 라이선스 고려 사항을 통해에서 AWS SQL Server 배포에 비용이 미치는 영향을 분석합니다. 조직이 3개의 애플리케이션을 지원하기 위해에 6개의 SQL Server Enterprise 서버를 배포해야 AWS 한다고 가정해 보겠습니다. 이러한 서버에는 고가용성이 필요하며 각각 16vCPUs와 256GB의 RAM이 있습니다. 다음 시나리오 세부 정보를 참조하세요.

- 서버 - SQL Server
- 운영 체제 에디션 - Windows Server Datacenter 2019
- SQL Server 에디션 – SQL Server Enterprise 2019
- vCPU – 16

- 메모리(GB) – 256
- 수량 - 6

성능을 저하 AWS 시키지 않고의 비용을 최적화하려면 CPU, 메모리, 네트워크 및 디스크(IOPS/BW) 사용률을 기반으로 적절한 크기의 인스턴스를 사용하는 것이 좋습니다. 워크로드의 크기를 오른쪽으로 조정한 후 16vCPUs를 제공하는 x2iedn.4xlarge 인스턴스 유형에 배치합니다. 그러나이 인스턴스 유형에는 워크로드에 필요한 메모리의 두 배가 포함됩니다. 추가 최적화는 여전히 가능합니다.

시나리오 1

조직은 Windows 및 SQL Server 모두에 대해 라이선스 포함 옵션을 사용하여 AWS 공유 테넌시에 6개의 SQL Server Enterprise 서버를 배포합니다. 이 옵션을 사용하면 Windows 및 SQL Server 라이선스 비용이 인스턴스 가격에 통합됩니다. 다음 시나리오 세부 정보를 참조하세요.

- 공유 테넌시(인스턴스) – x2iedn.4xlarge
- 시간당 비용(USD) – 10.0705 USD
- 단위당 월별 비용(USD) - 7,351.47 USD
- 서버 수 - 6
- CPU – 16
- 메모리 - 512
- 서버 6대의 월별 비용 - 44,108 USD

시나리오 2

조직은 공유 테넌시에서 SQL Server용 SA 및 BYOL을 가집니다. 즉, 조직은 Windows용 라이선스 포함 옵션을 사용하지만 인스턴스에 할당된 vCPUs 수에 따라 자체 SQL Server 라이선스를 제공합니다. 조직에 각각 16개의 vCPUs가 있는 6개의 SQL Server Enterprise 서버가 있으므로 총 96vCPUs가 필요합니다. 다음 시나리오 세부 정보를 참조하세요.

- 공유 테넌시(인스턴스) – x2iedn.4xlarge
- 시간당 비용(USD) – 4.0705 USD
- 단위당 월별 비용(USD) – 2971.47 USD
- 서버 수 - 6
- CPU – 16
- 메모리 - 512

- BYOL 코어 – 96
- 서버 6대의 월별 비용 - 17,828 USD

이 시나리오의 조직은 자체 SQL Server 라이선스를 SA로 가져와 SQL Server에 라이선스 포함 옵션을 사용하는 것보다 비용을 절감할 수 있습니다. 정확한 비용 절감은 특정 라이선스 계약의 요금 및 조건에 따라 달라집니다. 이 시나리오에서는 SQL Server Enterprise 라이선스를에 가져올 때 매월 26,280 USD씩 AWS 비용이 절감됩니다 AWS.

시나리오 3

조직은 Amazon EC2 전용 호스트의 Windows 및 SQL Server 모두에 대해 BYOL을 사용합니다. 즉, 조직은 물리적 코어 수준에서 라이선스를 할당하여 호스트의 물리적 코어에만 라이선스를 부여할 수 있습니다. 물리적 코어 수준에서 라이선스를 사용하면 필요한 라이선스에 영향을 주지 않고 최대 인스턴스 수를 배포할 수 있습니다. 이 라이선스 모델은 일반적으로 Windows Server Datacenter 및 SQL Server Enterprise 에디션과 함께 사용됩니다.

이 시나리오에서는 두 개의 X2iezn Amazon EC2 전용 호스트를 사용합니다. 각 호스트에는 24개의 물리적 코어와 48vCPUs가 있습니다. 이렇게 하면 각각 16개의 vCPUs와 256GB의 RAM이 있는 6개의 SQL Server Enterprise 서버에 충분한 용량이 제공됩니다. 다음 시나리오 세부 정보를 참조하세요.

- 전용 호스트 수 - 2
- 인스턴스 패밀리 - x2iezn
- 시간당 비용(USD) – 11.009 USD
- 단위당 월별 비용(USD) - 8,036 USD
- 물리적 코어 - 48
- 사용 가능한 vCPU – 96
- Windows Server 코어 라이선스 필요 - 24
- SQL Server Enterprise 코어에 필요한 라이선스 - 24
- 월별 비용 - 16,073

X2iezn 패밀리 Amazon EC2 전용 호스트 2개의 총 비용은 매월 16,073 USD입니다. 요금에 대한 자세한 내용은이 시나리오의 AWS Pricing Calculator 견적을 참조하세요. 이 시나리오의 조직은 Windows 라이선스를 가져와서 매월 1,755.65 USD를 절감할 수 있습니다. Amazon EC2 전용 호스트를 사용하는 경우 필요한 SQL Server 라이선스 수를 줄일 수도 있습니다. 공유 테넌시에서는 각각 16개의 vCPUs로 6개의 SQL Server Enterprise 서버를 덮기 위해 96개의 SQL Server Enterprise 라이선스가

필요합니다. 그러나 Amazon EC2 전용 호스트를 사용하고 물리적 코어 수준에서 라이선스를 부여하면 필요한 라이선스 수를 48개 코어로 줄일 수 있습니다.

다음 세부 정보는 예제 3의 비용을 비교하고 다른 시나리오와 비교하여 BYOL 옵션을 사용하여 Amazon EC2 전용 호스트에 워크로드를 배포하여 절감할 수 있는 금액을 보여줍니다.

- 온프레미스 서버 – SQL Server
- vCPU – 16
- 메모리 - 256
- 서버 수 - 6
- 시나리오 1의 월별 비용: Windows(LI) + SQL Server Enterprise(LI) – 44,108 USD
- 시나리오 2의 월별 비용: Windows(LI) + SQL Server Enterprise(BYOL) – 17,828 USD
- 시나리오 3의 월별 비용: Amazon EC2 전용 호스트의 Windows(LI) + SQL Server Enterprise(BYOL) - 16,073 USD

> **Note**
>
> 비용은 온디맨드 요금을 기준으로 합니다. Savings Plans 또는 전용 예약 인스턴스를 사용하여 비용을 더욱 절감할 수 있습니다. 이러한 옵션은 온디맨드 요금에 비해 상당한 비용 절감 효과를 제공하는 유연한 요금 모델을 제공합니다. 이러한 플랜을 사용하면 1년 또는 3년 약정을 할 수 있습니다. 자세한 내용은이 가이드의 Amazon EC2에서 Windows에 대한 지출 최적화 섹션을 참조하세요.

Amazon EC2 전용 호스트에 대해 다음 결제 옵션을 고려하세요.

- 전용 호스트(Amazon EC2 설명서)
- 전용 호스트 예약(Amazon EC2 설명서)
- Savings Plans(Amazon EC2 설명서)

는 AWS Pricing Calculator 이제 전용 호스트 요금을 지원합니다. 이렇게 하면 적절한 기본 전용 호스트를 선택하는 데 도움이 될 수 있습니다.

비용 최적화 권장 사항

를 사용하여 비용을 최적화하려면 AWS Cost Explorer다음 단계를 수행하는 것이 좋습니다.

1. Cost Explorer를 사용 설정합니다.
2. Cost Explorer를 사용하여 Amazon EC2 전용 호스트 배포의 비용 및 사용량을 보고 분석할 수 있습니다.
3. BYOL을 실행 중인지 확인합니다. Amazon EC2 콘솔의 인스턴스 또는 AMI 페이지 또는 `describe-images` 또는 `describe-instances` 명령에서 반환된 응답에 다음 플랫폼 세부 정보 및 사용 작업 값을 표시할 수 있습니다.
   - 플랫폼 세부 정보: Windows , 사용 작업: RunInstances:0002(라이선스 포함)
   - 플랫폼 세부 정보: Windows BYOL, 사용 작업: RunInstances:0800

추가 리소스

- 라이선스 유형 변환에 적합한 라이선스 유형(AWS License Manager 설명서)
- AWS License Manager 및 전용 호스트 워크숍(AWS License Manager 워크숍)
- Amazon EC2 전용 호스트 FAQs(AWS 설명서)
- VM Import/Export를 사용하여 온프레미스에서 Windows Server Bring-Your-Own-License AMIs를 생성하는 방법(블로그의 AWS Microsoft 워크로드)
- VM Import/Export(AWS 문서)
- Amazon Web Services 및 Microsoft: 자주 묻는 질문(AWS 문서)
- License Manager의 라이선스 유형 변환(AWS License Manager 문서)
- Amazon EC2 전용 호스트에 가용성이 높은 SQL Server 배포(AWS 클라우드 운영 및 마이그레이션 블로그)

Amazon EC2에서 Windows에 대한 지출 최적화

개요

서버를 로 마이그레이션할 때 가장 우려되는 사항 중 하나는 인프라 비용 AWS 입니다. 클라우드의 이점 중 하나는 온디맨드 리소스에 대한 비용을 지불하지만 연중무휴 24시간 사용할 수 있어야 하는 프로덕션 워크로드가 있다는 것입니다. Savings Plans은 EC2 인스턴스 AWS Lambda, 및에서 안정 상태 AWS 사용량에 대한 비용을 절감하도록 설계되었습니다 AWS Fargate.

Savings Plans 유연한 요금 모델을 제공하며 일관된 사용량(예: 시간당 10 USD)에 대한 약정의 대가로 Amazon EC2, Fargate, Lambda 및 Amazon SageMaker AI 사용량에 대한 요금을 줄이는 데 도움이 될

수 있습니다. 1년 또는 3년 동안 일정한 양의 시간당 컴퓨팅 지출을 약정하고 그 대가로 해당 사용량에 대한 할인을 받습니다.

Savings Plans

- 선결제 없음 옵션은 선결제할 필요가 없으며 약정은 순전히 월별로 청구됩니다.
- 부분 선결제 옵션은 Savings Plans에 더 저렴한 가격을 제공합니다. 약정의 최소 절반이 선결제되고 나머지는 월별로 청구됩니다.
- 전체 선결제 옵션은 최저 가격을 제공하며 전체 약정은 한 번의 결제로 청구됩니다.

Savings Plans 만료 및 향후 대기 Savings Plans 있습니다 AWS Cost Explorer. Savings Plans 알림을 사용하여 플랜 만료 날짜 1, 7, 30 또는 60일 전 또는 약정이 구매를 위해 대기열에 추가될 때 사전 이메일 알림을 받을 수 있습니다. 이러한 알림은 만료 날짜에도 알림을 제공합니다. 최대 10명의 이메일 수신자에게 알림을 보낼 수 있습니다.

절감형 플랜 이해

모든 유형의 컴퓨팅 사용량에는 온디맨드 요금과 Savings Plans. 시간당 10 USD의 컴퓨팅 사용량을 약정하는 경우 Savings Plans 요금으로 최대 10 USD의 모든 사용량에 대해 Savings Plans 요금을 받습니다. 컴퓨팅 지출 약정을 초과하는 모든 사용량은 일반 온디맨드 요금으로 청구됩니다. 에서 Cost ExplorerSavings Plans을 시작할 수 있습니다 AWS Management Console.

Cost Explorer에 제공된 권장 사항을 사용하여 Savings Plans을 쉽게 약정하여 가장 큰 절감 효과를 실현할 수 있습니다. 권장되는 시간당 약정은 과거 온디맨드 사용량과 선택한 플랜 유형, 기간 및 결제 옵션을 기반으로 합니다. Savings Plans은 먼저 플랜을 구매한 계정에 적용된 다음 통합 결제 패밀리의 다른 계정과 공유됩니다.

> **Note**
>
> 의 Savings Plans 공유 옵션은 기본적으로 활성화 AWS Organizations 되어 있습니다. 지급인 계정의 AWS Billing 콘솔에서이 옵션을 거부할 수 있습니다. 권장 사항 페이지를 방문하여 적격 사용량을 절약하는 데 도움이 되는 Savings Plans AWS 을 확인할 수 있습니다. 이러한 권장 사항은 언제든지 새로 고쳐 최적의 Savings Plans.

컴퓨팅 절감형 플랜

Compute Savings Plans 유연성을 극대화하고 비용을 절감하는 데 도움이 됩니다. 이러한 플랜은 인스턴스 패밀리, 크기, 가용 영역, 리전, 운영 체제 또는 테넌시와 관계없이 EC2 인스턴스 사용량에 자

동으로 적용됩니다. 또한 Fargate 또는 Lambda 사용량에도 적용됩니다. 예를 들어 Compute Savings Plans을 사용하면 언제든지 C4에서 M5 인스턴스로 변경하거나, EU(아일랜드)에서 EU(런던)로 워크로드를 이동하거나, EC2에서 Fargate 또는 Lambda로 워크로드를 이동할 수 있습니다. Savings Plans 가격을 계속 지불합니다.

EC2 인스턴스 절감형 플랜

EC2 Instance Savings Plans 리전에서 개별 인스턴스 패밀리 사용에 대한 약정(예: 버지니아 북부에서 일관된 수준의 M5 사용에 약정)의 대가로 가장 심층적인 할인을 제공합니다. 그러면 가용 영역, 크기, 운영 체제 또는 테넌시와 관계없이 해당 리전에서 선택한 인스턴스 패밀리의 온디맨드 요금에 대한 할인이 자동으로 제공됩니다. EC2 Instance Savings Plans 사용하면 해당 리전의 패밀리 내 인스턴스 간에 사용량을 변경할 수 있습니다. 예를 들어 Windows를 실행하는 c5.xlarge에서 Linux를 실행하는 c5.2xlarge로 전환하고 Savings Plans 가격을 자동으로 활용할 수 있습니다.

컴퓨팅 및 EC2 Instance Savings Plans은 모두 Amazon EMR, Amazon Elastic Kubernetes Service(Amazon EKS) 및 Amazon Elastic Container Service(Amazon ECS) 클러스터의 일부인 EC2 인스턴스에 적용됩니다. Amazon EMR, Amazon EKS 및 Amazon ECS 요금은 Savings Plans에서 다루지 않지만 기본 EC2 인스턴스는 적용됩니다. 컴퓨팅 절감형 플랜은 적용 범위가 더 넓기 때문에 EC2 인스턴스 절감형 플랜이 컴퓨팅 절감형 플랜보다 우선 적용됩니다.

> **Note**
>
> 약정을 한 후에는 Savings Plan을 쉽게 변경할 수 없습니다. Savings Plans 옵션 중 하나를 약정하기 전에 신중하게 계획하는 것이 좋습니다. Savings Plans 약정에 대한 대가로 온디맨드 요금에 비해 더 낮은 가격을 제공하며 계약 기간 동안에는 취소할 수 없습니다.

시간당 약정 예제

Savings Plan을 구매하는 경우 플랜 기간에 시간당 금전적 약정을 체결합니다. 시간당 10 USD의 컴퓨팅 사용량을 약정하는 경우 Savings Plan 요금은 시간당 최대 10 USD의 모든 사용량에 자동으로 적용됩니다. 약정을 초과하는 모든 사용량은 일반 온디맨드 요금으로 청구됩니다. Cost Explorer의 Savings Plans 구매 권장 도구를 사용하여 절감 효과를 극대화할 수 있는 권장 약정을 얻을 수 있습니다. 특정 플랜의 시간당 재무 약정은 플랜 기간 동안 수정할 수 없습니다. 사용량 분석 후 약정을 늘리려면 추가 Savings Plan하여 초과 사용량을 처리할 수 있습니다.

Savings Plans의 이점

예약 인스턴스에 비해 Savings Plans Savings Plans형 플랜에서 제공하는 광범위한 컴퓨팅 옵션을 활용하면서 비용을 절감할 수 있는 보다 유연한 요금 모델을 제공합니다. Savings Plans 컴퓨팅 요구 사항이 변경되더라도 할인을 제공합니다. 이렇게 하면 추가 관리 오버헤드가 발생하지 않고 끊임없이 변화하는 동적 환경을 따라잡을 수 있습니다. Savings Plans.

- 사용하기 쉬움 - 금전적 약정에 대한 대가로 자동 할인을 받을 수 있습니다.
- 유연성 - 여러 사용 유형에 적용되는 단일 약정입니다.
- 잠재적 절감 - 다양한 방법으로 절감할 수 있습니다. 다음 예제를 살펴보세요.
  - Compute Savings Plans을 사용하여 Windows Server 워크로드에서 60% 절감(d2.8xlarge, 3년, 모두 선결제, 창, 공유 테넌시, us-east-2)
  - EC2 Instance Savings Plans을 사용하여 Windows Server 워크로드에서 73% 절감(d2.8xlarge, 3년, 모두 선결제, 창, 공유 테넌시, us-east-2)
  - 이국적이지 않은 인스턴스 유형(t3 패밀리, 3년, 모든 선결제, 기간, 공유 테넌시, us-east-2)에 대한 28~41% 절감
  - Windows Server의 평균 절감액 25~40%

> **Note**
>
> EC2 Instance Savings Plans 유연성이 저하되어 Compute Savings Plans보다 더 큰 할인을 제공합니다. 할인된 가격으로 사용을 약정합니다.

모든 유형의 컴퓨팅 사용량에는 Savings Plan 요금과 온디맨드 요금이 있습니다. 다음 표에는 모든 운영 체제 유형에 대한 Savings Plans 및 온디맨드 요금이 나와 있습니다. 약정 사용량에 대해 Savings Plans 요금이 청구되며 약정을 초과하는 사용량은 일반 온디맨드 요금으로 청구됩니다.

인스턴스 이름	Savings Plans 요금	온디맨드 절감	온디맨드 속도	운영 체제	리전	결제 옵션	기간 길이
x2iedn.xlarge	0.32 USD	61%	0.83 USD	Linux	미국 동부(버지니아 북부)	선수금 없음	3

인스턴스 이름	Savings Plans 요금	온디맨드 절감	온디맨드 속도	운영 체제	리전	결제 옵션	기간 길이
x2iedn.xlarge	2.01 USD	50%	1.02 USD	Windows	미국 동부(버지니아 북부)	선수금 없음	3
x2iedn.xlarge	1.02 USD	20%	2.52 USD	Windows 라이선스 포함 + SQL Server Enterprise Edition	미국 동부(버지니아 북부)	선수금 없음	3
x2iedn.xlarge	0.32 USD	61%	0.83 USD	BYOL	미국 동부(버지니아 북부)	선수금 없음	3

Savings Plans에는 운영 체제가 포함되며 BYOL에 대해 별도의 할인이 적용됩니다. 컴퓨팅 Compute Savings Plans 계산기에서 모두 분류됩니다.

예약 인스턴스 요금 모델

AWS 에는 예약 인스턴스라고 하는 약정을 기반으로 하는 또 다른 요금 모델이 있습니다. 이미 커밋한 후 컴퓨팅이 변경되어 예약 인스턴스가 사용되지 않는 경우이 모델은 문제가 될 수 있습니다. Savings Plans은 표준 및 전환형 예약 인스턴스와 유사한 비용 절감 효과를 제공하지만 유연성은 훨씬 더 높입니다. Compute Savings Plans 인스턴스 패밀리, 크기, 운영 체제, 테넌시 또는 리전에 관계없이 EC2 인스턴스 사용량에 대해 더 저렴한 가격을 제공합니다. 또한 유연성을 극대화합니다.

다음 표는 Savings Plans 또는 예약 인스턴스 중에서 선택하는 데 도움이 될 수 있습니다.

	Reserved Instance	EC2 인스턴스 절감형 플랜	컴퓨팅 절감형 플랜
평균 1년 할인	최대 38%	최대 29%	최대 29%

	Reserved Instance	EC2 인스턴스 절감형 플랜	컴퓨팅 절감형 플랜
평균 3년 할인	최대 58%	최대 73%	최대 60%
인스턴스 패밀리	고정	고정	유연성
인스턴스 크기	고정(Linux 아님)	유연성	유연성
Geography	리전 1개	리전 1개	유연성
운영 체제	고정	유연성	유연성
Service	Amazon EC2 또는 Amazon RDS	Amazon EC2	Amazon EC2, Fargate, Lambda
결제 옵션	모두, 부분, 선결제 없음	모두, 부분, 선결제 없음	모두, 부분, 선결제 없음
인스턴스 제한	가용 영역당 20개	제한 없음	제한 없음

Note

Savings Plans 시간당 금전적 약정을 기준으로 할인을 제공하는 방식으로 작동합니다. 시간당 재무 약정은 플랜 기간 동안 취소하거나 변경할 수 없지만 추가 Savings Plans하여 추가 사용량을 충당할 수 있습니다. 이를 통해 플릿이 증가함에 따라 일관된 시간당 약정을 유지할 수 있습니다.

AWS Cost Explorer 또는 AWS 클라우드 Intelligence Dashboards와 같은 도구를 사용하여 약정을 추적할 수 있습니다. Cost Explorer는 조직이 Savings Plans 적용 범위 전략을 계획하는 데 도움이 될 수 있는 적용 범위 대상 라인을 제공합니다. 워크로드의 75%가 정상 상태인 경우 75%가 좋은 목표입니다. 이렇게 하면 동적 워크로드를 기반으로 온디맨드/변동 지출의 25%가 남습니다. 이를 85%의 적용 범위로 늘려야 하는 경우 시간당 금전적 약정을 늘리기 위해 Savings Plans 약정을 구매할 수 있습니다.

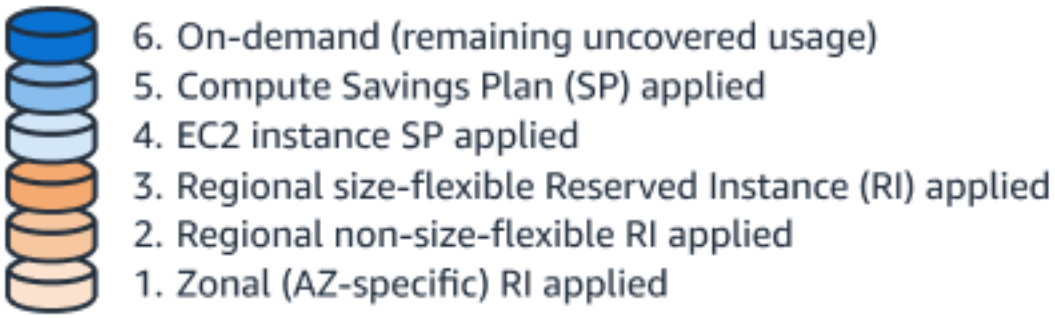
> **Note**
>
> 예약 인스턴스 대신 Savings Plans 구매하는 것이 좋지만 예약 인스턴스를 이미 구매한 경우 두 커밋 모델이 함께 작동할 수 있습니다.

예약 인스턴스를 구매했지만 Savings Plans 옵션 시도를 시작하려는 경우를 예로 들어 보겠습니다. 이 조합이 최종 결제에 적용되는 로직이 있습니다. 다음은에 적용할 수 있는 계층 구조입니다. AWS 계정

1. 영역 예약 인스턴스는 해당 인스턴스를 소유한 계정에 적용됩니다. 예약 인스턴스가 몇 시간 남았으면 조직의 나머지 부분에 적용됩니다.
2. Windows용 유연한 크기 조정이 불가능한 리전 예약 인스턴스는 Windows를 소유한 계정의 일치하는 사용량에 적용됩니다. 남아 있는 모든 것은 조직의 나머지 부분으로 롤아웃됩니다.
3. 유연한 크기의 리전 예약 인스턴스는 해당 인스턴스를 소유한 계정(패밀리 내에서 가장 작은 인스턴스부터 더 큰 인스턴스까지)과 나머지 조직에 적용됩니다.
4. 리전 예약 인스턴스는 미사용 온디맨드 용량 예약에 적용됩니다.
5. EC2 Instance Savings Plans 해당 플랜을 구매한 계정 내에서 적용됩니다.
6. Compute Savings Plans 해당 플랜을 구매한 계정 내에 적용됩니다.

> **Note**
>
> 할인은 할인이 가장 높은 사용량부터 시작하여 할인이 가장 적은 사용량까지 적용됩니다. Windows 인스턴스는 일반적으로 대부분의 일반적인 인스턴스 유형(예: T3, M6, C5)에 대해 Linux보다 할인 가능성이 낮습니다. 즉, Linux 인스턴스는 대부분의 경우 Windows 인스턴스보다 더 많은 이점을 제공합니다.

다음 그림은 예약 인스턴스를 Savings Plans과 나눈 후의 가격을 보여줍니다. 컴퓨팅 및 EC2 Instance Savings Plans 모두 먼저 인스턴스를 실행한 다음 미사용 온디맨드 용량 예약에 적용됩니다.



비용 최적화 시나리오

이 섹션에서는 라이선스 포함 결제 모델을 사용하는 Amazon EC2 전용 호스트 및 Amazon EC2 인스턴스에 대한 비용 최적화 시나리오를 다룹니다.

Amazon EC2 전용 호스트

온프레미스 Windows 워크로드를 마이그레이션하려는 시나리오를 생각해 보세요 AWS. 데이터 센터에는 다음과 같은 서버가 있습니다.

- 16 vCPU 및 128GB RAM이 있는 서버 2개
- 32 vCPU 및 164GB RAM이 있는 서버 2개
- vCPU 8개 및 RAM 64GB가 있는 서버 1개
- vCPU 및 32GB RAM이 있는 서버 16개

또한에 자체 라이선스를 가져올 수 있는 라이선스가 충분하기 AWS 때문에에 자체 라이선스를 가져올 수 있다고 가정합니다. 다음 표에는 사용할 수 있는 서버 인스턴스가 나와 있습니다 AWS.

인스턴스 유형	CPU	RAM	Amount
r5.4xlarge	16	128	2
r5.8xlarge	32	256	2
r5.2xlarge	8	64	1
r5.xlarge	4	32	16
			21

분석에 따르면 이러한 가상 머신 21개를 R5 인스턴스 패밀리 호스트가 있는 전용 호스트 2개에 배포할 수 있습니다. 다음 표에는이 두 전용 호스트의 비용이 나와 있습니다.

전용 호스트 온디맨드 시나리오	선결제	한 달	1년	3년	AWS Pricing Calculator
온디맨드	없음	10,123 USD	121,475 USD	364,392 USD	AWS Pricing Calculator 추정
1년 Savings Plan	없음	7,447 USD	89,362 USD	–	AWS Pricing Calculator 추정
3년 Savings Plan	없음	5,476 USD	65,712 USD	197,128 USD	AWS Pricing Calculator 추정
선결제 포함 3년 Savings Plan	84,438 USD	2,755 USD	117,499 USD	183,618 USD	AWS Pricing Calculator 추정

마이그레이션하려는 서버가 있는 경우 AWS 1년 Savings Plan의 최종 가격은 온디맨드 가격의 121,475 USD가 아닌 89,362 USD입니다. 이는 1년 후 26.5% 할인을 나타냅니다. 더 오랜 기간 AWS 동안에 머무르고 싶다면 3년 Savings Plan을 선택하여 비용 절감 효과를 더욱 높일 수 있습니다. 3년이 끝나면 364,392 USD가 아닌 197,128 USD를 지불합니다. 이로 인해 3년 후 총 금액의 46%가 절감됩니다.

라이선스가 포함된 Amazon EC2 인스턴스

단일 3계층 애플리케이션을 마이그레이션하고에서 제공하는 라이선스를 사용 AWS하려는 시나리오를 생각해 보세요 AWS. 또한 애플리케이션이 다음 서버에서 작동한다고 가정합니다.

- vCPUs 2개와 4GB RAM이 있는 웹 서버 2개
- vCPUs개와 16GB RAM이 있는 애플리케이션 서버 2개
- vCPUs개와 RAM 64GB가 있는 데이터베이스 서버 2개(SQL Server Standard Edition 사용)

다음 표에는에서 사용할 수 있는 서버 인스턴스가 나와 있습니다 AWS.

인스턴스 유형	CPU	RAM	Amount
c5.large	2	4	2
c5.2xlarge	8	16	2
r5.2xlarge	8	64	2
			서버 6개

다음 표에는 이러한 서버의 비용이 나와 있습니다 AWS.

에 포함된 라이선스 AWS	선결제	한 달	1년	3년	AWS Pricing Calculator
온디맨드	없음	3,912 USD	46,950 USD	140,849 USD	AWS Pricing Calculator 추정
1년 Savings Plan	없음	3,466 USD	41,952 USD		AWS Pricing Calculator 추정
선결제 없는 3년 Savings Plan	없음	3,189 USD	38,264 USD	114,804 USD	AWS Pricing Calculator 추정
선결제 포함 3년 Savings Plan	112,110 USD	없음	없음	없음	AWS Pricing Calculator 추정

온디맨드 요금으로 프로덕션 환경(24/7)에서 이러한 서버를 실행하려는 경우 월 3,912 USD의 비용을 지불합니다. 이 월별 비용을 지불하면 1년 후 46,950 USD, 3년 후 총 140,849 USD에 해당합니다.

선결제 없이 1년 Savings Plan 선택하면 월별 비용이 3,466 USD로 감소합니다. 첫 해가 끝날 때 41,952 USD를 지불합니다. 총 할인율은 11%입니다. 선결제 없이 3년 Savings Plan 선택하면 월별 비

용이 3,189 USD로 감소합니다. 3년이 지나면 114,804 USD를 지불합니다. 이렇게 하면 18.5%를 절감할 수 있습니다.

비용 최적화 권장 사항

두 시나리오 모두에서 워크로드를 계획하고 예측할 때 비용을 절감하는 데 도움이 됩니다 AWS. 두 번째 시나리오의 할인이 첫 번째 시나리오에 비해 낮다는 점을 인식하는 것이 중요합니다. 두 번째 시나리오에서는 라이선스 가격이 클라우드 서버의 요금에 포함됩니다. AWS 는 라이선스 요금에 대한 할인을 제공하지 않지만 언제든지 라이선스를 가져올 수 있으며(특정 시나리오에서) 항상 최상의 컴퓨팅/인스턴스 가격을 보증 AWS 할 수 있습니다.

컴퓨팅 및 인스턴스 리소스에 대한 AWS 지출을 제어하려면 다음을 수행하는 것이 좋습니다.

- 액세스 권장 사항
- 필요에 따라 권장 사항 사용자 지정
- 시간당 약정 검토

액세스 권장 사항

Amazon EC2 콘솔을 사용하여 Savings Plan에 대한 권장 사항에 액세스할 수 있습니다. 권장 사항을 다운로드하여 나중에 CSV 형식으로 검토할 수도 있습니다. 자세한 내용은 Savings Plans 설명서의 절감형 플랜 모니터링을 참조하세요. Savings Plans

필요에 따라 권장 사항 사용자 지정

Amazon EC2 콘솔을 열고 인스턴스 섹션을 확장한 다음 Savings Plans 선택합니다. 이 페이지에는 권장 사항 적용 전후의 인스턴스 및 컴퓨팅 요금이 표시됩니다. 권장 사항에 따라 다음 요소를 조정할 수도 있습니다.

- 기간 - 예: 1~3년
- 결제 옵션 - 예: 선결제, 부분 선결제 또는 선결제 없음
- 기록 - 예를 들어 지난 7, 30 또는 60일

시간당 약정 검토

동일한 예제를 사용하여 연중무휴로 실행되는 인스턴스가 있다고 가정합니다. Savings Plan을 사용하는 것이 좋습니다. 크기에 따라 온디맨드 가격은 시간당 120 USD입니다. 시간당 90 USD를 약정할 수 있지만 이는 리전, 인스턴스 및 구매 옵션에 따라 다를 수 있습니다. 이 예제에서는 온디맨드 비용에 비

해 25%를 절감할 수 있습니다. 또한 사용률과 적용 범위가 정의한 임계값 미만인 경우 이를 추적하고 예산이 종료될 때 알림을 구성할 수 있습니다.

권장 사항 검토

Savings Plan 권장 사항을 주의 깊게 검토하는 것이 좋습니다. AWS 는 사용자의 권한 없이는 아무것도 변경하지 않습니다. 이는 권장 사항일 뿐이며 적용 여부는 사용자에게 달려 있습니다.

플랜 구매

Amazon EC2 콘솔을 열고 인스턴스 섹션을 확장한 다음 Savings Plans 선택합니다. 그런 다음 Savings Plans 구매를 선택합니다. 요구 사항에 따라 기간, 리전, 인스턴스 패밀리, 시간당 약정, 결제 옵션, 시작 날짜 등의 옵션을 선택할 수 있습니다. Compute Savings Plans, EC2 Instance Savings Plans 및 SageMaker AI Savings Plans. 자세한 내용은 Savings Plans 설명서의 절감형 플랜 구매를 참조하세요. Savings Plans

사용률 보고서 가져오기

Savings Plan을 구매한 후 사용률 보고서를 얻을 수 있습니다. 보고서는 사용률을 확인하고, 구매한 플랜이 할인을 적용하고 극대화하기에 충분한지 확인하고, 새 할인을 취소하거나 추가하는 데 도움이 됩니다. 이 보고서는 CSV와 같은 다른 형식으로 내보낼 수 있습니다. 자세한 내용은 절감형 플랜 설명서의 사용률 보고서 사용을 참조하세요. Savings Plans

구매 모범 사례 준수

Savings Plans 다음 모범 사례를 따르는 것이 좋습니다.

- AWS Trusted Advisor를 사용하여 유휴 EC2 리소스를 제거합니다.
- Savings Plans 수행합니다.
- 30~60일 동안 지속적으로 유지하는 시간당 요금을 설정합니다.
- 조직이 편한 만큼 일관된 시간당 요금을 충당하기 위한 약정을 구매합니다. 수요 또는 계절의 변동을 고려합니다.
- 분기별 Savings Plans 예산 검토를 선택하여 일관된 비율을 유지합니다(예: Savings Plans 적용 범위의 70% 적용 범위 목표). 비율이 원하는 적용 범위 아래로 떨어지면 적용 범위 목표를 달성하기 위해 Savings Plan을 트루업으로 구입합니다.

추가 리소스

- Amazon EC2 예약 인스턴스Savings Plans(AWS 백서)

- Savings Plans AWS 사용량에 적용되는 방식 이해(Savings Plans 설명서)
- EC2 Windows Server 및 SQL Server 인스턴스에 대한 초당 결제 발표(AWS 문서)
- AWS 비용 최적화 시리즈: Savings Plans 비디오 | Amazon Web Services(YouTube)

AWS 도구를 사용하여 비용 모니터링

개요

비용 가시성은에서 비용을 최적화하는 주요 요소입니다 AWS. AWS 에는 비용을 시각화하고 해당 비용에 대한 대응으로 알림을 생성하는 데 사용할 수 있는 여러 도구가 있습니다. 여기에는 지출을 추적하고 보고하는 데 도움이 AWS Budgets되는 같은 도구가 포함됩니다. 이 섹션에서는 예산 요구 사항에 따라 추적하고 대응할 수 있도록 Windows에서 AWS 지출을 모니터링하는 특정 방법을 다룹니다. 여기에는 Windows EC2 리소스에 필요한 태그 추가가 포함됩니다. 이러한 태그를 사용하면를 사용하여 Windows EC2 및 기타 Microsoft 서비스를 올바르게 모니터링할 수 있습니다 AWS Budgets.

지출을 모니터링하고 AWS 도구를 사용하여 알림을 생성하면 현재 지출, 예상 지출 및 지출 이상에 대한 정보를 더 많이 얻을 수 있습니다. Savings Plans 사용하여 시간당 EC2 인스턴스 요금을 줄이는 경우 Savings Plan의 전체 사용률 및 적용 범위를 보는 것이 좋습니다. 이렇게 하면 절감 효과를 지속적으로 실현할 수 있습니다. AWS Cost Explorer 를 사용하여 Savings Plan 보고 이전 사용량을 기반으로 추가 Savings Plans에 대한 권장 사항을 얻을 수 있습니다. AWS Budgets를 사용하고를 설정하여 특정 지출을 추적할 수도 있습니다AWS Cost Anomaly Detection.

비용 최적화 권장 사항

AWS Budgets Cost Explorer 및 이상 탐지를 사용하여 비용을 최적화하려면 다음 단계를 수행하는 것이 좋습니다.

- Windows EC2 리소스에 태그 지정
- 를 사용하여 알림 설정 AWS Budgets
- 비용 이상 탐지 활성화
- 실시간 지출 분석 가져오기
- Cost Explorer를 사용하여 Windows에 대한 라이선스 포함 지출 보기

Windows EC2 리소스에 태그 지정

AWS 지출을 효과적으로 모니터링하려면 모니터링하려는 워크로드에 대한 태그 지정 전략을 설정해야 합니다. 이는 일반적인 사용 지출과 달리 리소스를 범주별로 그룹화하고 특정 지출에 대한 알림을 받을 수 있도록 하기 위해 중요합니다. 비용에 도움이 될 뿐만 아니라 AWS Systems Manager 자동화와 같은 다른 용도로도 사용할 수 있는 태그 지정 리소스를 사용할 수 있습니다. 또한 필수 태그에 대한 일부 관리를 구현하는 것이 좋습니다.

AWS Budgets Cost Explorer 및 Cost Anomaly Detection에서 지출을 추적하려면 적절한 태그가 있는지 확인해야 합니다. 태그를 사용하여 해당 태그와 일치하는 항목에 대한 특정 예산을 설정하여 지출이 증가할 때 알림을 받을 수 있습니다.

예를 들어 Key=OS Value=Windows와 같은 간단한 태그를 사용할 수 있습니다. 이렇게 하면 모든 Windows 인스턴스가 하나의 그룹으로 결합되어 지출을 추적할 수 있습니다. Systems Manager와 같은 다른 항목에도 태그를 사용할 수 있습니다. 태그를 생성한 후에는 비용 추적을 위해 태그를 활성화해야 합니다. 특정 리소스AWS Config 에 연결된 태그를 모니터링하는 규칙을 추가하는 것이 좋습니다.는 적절한 태그가 포함되지 않은 실행 중인 리소스가 있는 경우 AWS Config 알림을 보내 Windows EC2 지출을 정확하게 표현할 수 있습니다.

태그를 배치한 후에서 사용자 지정 예산을 생성할 수 있습니다 AWS Billing. 이를 통해 Windows EC2 지출을 파악할 수 있습니다. 일일 예산 또는 월별 예산을 설정할 수 있습니다.

를 사용하여 알림 설정 AWS Budgets

이 예제 시나리오에서는 Windows EC2에 대한 일일 예산을 생성합니다. 자동 조정 옵션을 사용하여 지출을 추적하고 그에 따라 예산을 조정하는 반복 예산입니다. 정적 환경이 있는 경우 대신 고정 예산을 사용할 수 있습니다. 기준 시간 범위(예: 30일)를 선택해야 합니다.

1. 에 로그인 AWS Management Console 하고 AWS Cost Management 콘솔을 엽니다.
2. 탐색 창에서 예산을 선택합니다.
3. 페이지 상단에서 예산 생성을 선택합니다.
4. 예산 설정에서 사용자 지정(고급)을 선택합니다.
5. 예산 유형에서 비용 예산을 선택합니다. 그리고 다음을 선택합니다.
6. 세부 정보에서 예산 이름에 예산 이름을 입력합니다. 예를 들어 Windows EC2 지출입니다.
7. 예산 금액 설정에서 기간에서 일별을 선택합니다.
8. 예산 갱신 유형의 경우 예산 기간 이후에 재설정되는 예산에 대해 반복 예산을 선택합니다.
9. 시작 날짜에서 시작 날짜 또는 기간을 선택하여 예산 금액에 대한 추적을 시작합니다.

10.예산 방법에서 자동 조정(신규)을 선택합니다.

11.기준 시간 범위에서 사용자 지정 범위를 선택한 다음 30일을 입력합니다.

12.Next(다음)를 선택합니다.

13.예산 범위 섹션에서 특정 AWS 비용 차원 필터링을 선택합니다. 여기에서 태그를 사용하여 적절한 차원을 생성합니다. AWS Budgets 는 필터에서 플랫폼 유형을 옵션으로 지원하지 않습니다. 따라서 OS 태그를 적용해야 합니다.

14.필터 추가를 선택한 다음 차원에서 태그 옵션을 선택합니다.

15.OS 태그를 선택한 다음 이에 대한 Windows 값을 선택하여 태그에 대한 예산을 생성합니다.

16.Next(다음)를 선택합니다.

17.알림 구성 페이지에서 알림 임계값 추가를 선택합니다. 여기서는 두 개의 알림을 설정합니다. 하나는 50% 임계값에 대한 것이고 다른 하나는 100% 임계값에 대한 것입니다. 50% 임계값 알림이 해당 월의 중간 지점 이전에 위반되면 경고가 표시됩니다. 이렇게 하면 지출이 예상보다 많은지 확인하고 월말에 도달하기 전에 대응할 수 있습니다.

18.임계값에 50을 입력하고 예산 금액의 %를 선택합니다.

19.트리거에서 실제를 선택합니다.

20.이메일 수신자의 경우 이메일 주소를 입력합니다. 임계값 100에 대해 다른 알림을 추가합니다.

> **Note**
>
> 이 예제에서는 알림에 이메일 알림을 사용하지만와 같은 다른 접근 방식도 사용할 수 있습니다Slack.

비용 이상 탐지 활성화

비용 태그를 사용하여 이상 항목인 지출 알림을 설정할 수 있습니다. 예를 들어 AWS Cost Anomaly Detection를 사용하여 지출에 대한 모니터를 생성하고 시스템에서 계정에서 비정상적인 지출을 감지하면 알림을 받을 수 있습니다.

이전에 생성한 Key=OS 및 Value=Windows 태그에 대한 모니터 및 알림을 설정하려면 다음을 수행합니다.

1. 에 로그인 AWS Management Console 하고 AWS Cost Management 콘솔을 엽니다.
2. 탐색 창에서 비용 이상 탐지(Cost Anomaly Detection)를 선택합니다.
3. 비용 모니터 탭을 선택한 다음 모니터 생성을 선택합니다.

4. 1단계에서 모니터 유형으로 비용 할당 태그를 선택합니다.
5. 비용 할당 태그 키에서 Windows EC2 지출을 선택합니다.
6. 비용 할당 태그 값에서 Windows를 선택합니다.
7. 모니터 이름 지정에 Windows EC2 지출을 입력합니다.
8. Next(다음)를 선택합니다.
9. 알림에 대한 구독을 생성하려면 새 구독 생성을 선택합니다. 기존 구독이 있는 경우 기존 구독 선택(Choose an existing subscription)을 선택합니다.
10. 구독 이름에 Windows EC2 지출 이상을 입력합니다.
11. 알림 빈도에서 일별 요약을 선택합니다.
12. 알림 수신자에 이메일 주소를 입력합니다.
13. 임계치 추가를 선택하세요. 임계값에 10을 입력한 다음 예상 속도보다 높은 비율을 선택합니다.
14. 모니터 생성(Create monitor)을 선택합니다.

지출에 대한 실시간 보기

알림은 Windows EC2 지출을 모니터링하는 데 유용한 도구이지만 지출을 실시간으로 보려면 Cost Explorer를 사용해야 합니다. 이 비디오를 보고 Cost Explorer를 통해 EC2 비용을 분석하고 절감하는 방법을 알아보세요. 자세한 내용은 YouTube에서 AWS Supports You | Understanding and reduceing Your EC2 Costs 비디오를 시청하세요.

Windows에 대한 라이선스 포함 지출 보기

Cost Explorer를 사용하여 계정의 EC2 Windows 지출을 볼 수 있습니다. Windows에 대한 라이선스 포함 지출을 보려면 Cost Explorer에서 다음과 같은 올바른 필터를 설정해야 합니다.

- 플랫폼에서 Windows(Amazon VPC)를 선택합니다. API 작업에서 RunInstance:0002를 선택합니다. 이는 라이선스 포함 Windows EC2 인스턴스의 AWS Billing 코드입니다.
- BYOL 인스턴스 지출을 보려면 RunInstance:0002를 RunInstance:0800으로 변경합니다. Windows EC2 BYOL의 결제 코드입니다.

Cost Explorer에서 이러한 가시성을 통해 비용을 Windows EC2에 지출하는 만큼 빠르게 필터링할 수 있습니다. AWS 지출을 더 자세히 알아보려면 AWS Cost and Usage Report 를 사용하여 개별 인스턴스 수준에서 지출로 필터링할 수 있습니다. Amazon QuickSight에서 시각화할 수 있는 보고서를 생성하고 사용자 지정 대시보드를 구축할 수도 있습니다.

자세한 내용은 YouTube에서 AWS Supports You - Visualizing Your Cost and Usage Reports 비디오를 시청하세요.

추가 리소스

- 를 사용하여 필수 태그 설정 AWS Config(AWS Config 문서)
- AWS Budgets 자습서 -에 대한 알림 설정 AWS Billing | Amazon Web Services(YouTube)
- AWS Cost and Usage Report 쿼리 라이브러리(AWS Well-Architected Labs)

SQL Server

고객은 다른 클라우드 공급자보다 15년 이상에서 Microsoft 워크로드 AWS 를 실행해 왔습니다. 이는 주로 AWS 가 클라우드에서 Microsoft 애플리케이션에 대한 경험이 가장 많고 다음 영역에서 Windows Server 및 Microsoft SQL Server에 가장 적합한 플랫폼을 제공하기 때문입니다.

- 성능 및 안정성 향상
- 보안 및 자격 증명 서비스 향상
- 추가 마이그레이션 지원
- 가장 광범위하고 심층적인 기능
- 총 소유 비용(TCO) 절감
- 유연한 라이선스 옵션

AWS 는 Active Directory, .NET, SQL Server, 서비스형 Windows 데스크톱, 지원되는 모든 Windows Server 버전을 포함하여 SQL Server에 의존하는 Windows 애플리케이션을 구축하고 실행하는 데 필요한 모든 것을 지원합니다. 검증된 전문 지식을 갖춘 AWS 는 Windows 워크로드를 쉽게 리프트 앤 시프트, 리팩터링 또는 현대화하는 데 도움이 될 수 있습니다.

가이드의이 섹션에서는 다음 주제를 다룹니다.

- 고가용성 및 재해 복구 솔루션 선택
- SQL Server 라이선스 이해
- SQL Server 워크로드에 적합한 EC2 인스턴스 선택
- 인스턴스 통합
- SQL Server 에디션 비교
- SQL Server 개발자 에디션 평가
- Linux에서 SQL Server 평가
- SQL Server 백업 전략 최적화
- SQL Server 데이터베이스 현대화
- SQL Server용 스토리지 최적화
- Compute Optimizer를 사용하여 SQL Server 라이선스 최적화
- Compute Optimizer를 사용하여 SQL Server 크기 최적화
- SQL Server 워크로드에 대한 Trusted Advisor 권장 사항 검토

고가용성 및 재해 복구 솔루션 선택

개요

복구 시간 목표(RTO) 및 복구 시점 목표(RPO)를 포함하여 재해 복구(DR) 목표를 충족하면서 비즈니스 요구 사항을 충족하는에서 SQL Server 배포를 위한 아키텍처를 설계 AWS 하는 것이 좋습니다. 다음 솔루션은 SQL Server 워크로드 비용을 최적화하는 동시에 Amazon Elastic Compute Cloud(Amazon EC2)의 SQL Server에 적합한 아키텍처를 설계하는 데 도움이 될 수 있습니다.

- SQL Server Always On 가용성 그룹 - SQL Server Always On 가용성 그룹은 SQL Server 데이터베이스를 위한 고가용성 및 재해 복구(HA/DR) 솔루션을 제공합니다. 가용성 그룹은 함께 장애 조치되는 사용자 데이터베이스 집합으로 구성됩니다. Always On 가용성 그룹은 데이터베이스 수준에서 중복성을 제공하지만 공유 스토리지가 필요하지 않습니다. 각 복제본에는 자체 로컬 스토리지가 있습니다. 이 기능을 HA/DR 솔루션으로 배포할 수 있습니다. 자세한 내용은 Microsoft 설명서의 Always On 가용성 그룹이란 무엇입니까?를 참조하세요.
- SQL Server Always On 장애 조치 클러스터 인스턴스(FCI) - SQL Server Always On FCIs Windows Server 장애 조치 클러스터링(WSFC)을 사용하여 SQL Server 인스턴스 수준에서 HA를 제공합니다. FCIs 데이터베이스를 호스팅하기 위해 공유 스토리지가 필요합니다. 공유 블록 스토리지 또는 공유 파일 스토리지를 사용할 수 있습니다. 예를 들어 Amazon FSx for Windows File Server 또는 Amazon FSx for NetApp ONTAP을 여러 가용 영역이 있는 공유 스토리지 솔루션으로 사용할 수 있습니다. 자세한 내용은 Microsoft 설명서의 Always On 장애 조치 클러스터 인스턴스(SQL Server)를 참조하세요.
- SIOS DataKeeper - SIOS DataKeeper는 가용 영역과 모두에 걸쳐 있는 SQL Server FCI를 활성화하여 HA 및 DR 요구 사항을 모두 충족하는 데 도움이 될 수 있습니다 AWS 리전. SIOS DataKeeper는 로컬 Amazon Elastic Block Store(Amazon EBS) 볼륨을 사용하여 클러스터링된 가상 SAN을 생성하고 HA용 가용 영역 간의 동기식 복제를 사용하는 동시에 재해 복구를 위해 리전과 간의 비동기식 복제를 사용합니다. 자세한 내용은 SIOS 설명서의 Windows 애플리케이션에 대한 고가용성 보호를 참조하세요.
- 분산 가용성 그룹 - 분산 가용성 그룹은 두 개의 개별 Always On 가용성 그룹에 걸쳐 있는 특수한 유형의 가용성 그룹입니다. 가용성 그룹은 두 개의 개별 리전(예: `us-east-1` 및 `us-west-1`)에 상주할 수 있습니다. 기본 Always On 가용성 그룹은 서로 다른 두 WSFC 클러스터에 구성되므로 분산 가용성 그룹을 가용성 그룹의 가용성 그룹으로 생각할 수 있습니다. 분산 가용성 그룹을 배포하려면 SQL Server Enterprise 에디션이 필요합니다. 자세한 내용은 Microsoft 설명서의 분산 가용성 그룹을 참조하세요.
- 로그 전송 - 드물지만 리전이 영향을 받아 사용할 수 없게 되는 경우 로그 전송을 구현하여 여러 리전에서 데이터베이스를 보호할 수 있습니다. 트랜잭션 및 로그 배송 빈도에 따라 몇 분 내에 RPO 및

RTO를 달성할 수 있습니다. 자세한 내용은 Microsoft 설명서의 로그 전송(SQL Server) 정보를 참조하세요.

- AWS Elastic Disaster Recovery - Elastic Disaster Recovery는 AWS DR 목적으로 모든 인프라에서 로 서버 복제를 관리하는 서비스형 소프트웨어(SaaS) 애플리케이션입니다. Elastic Disaster Recovery를 사용하여 리전 간에 SQL Server를 복제할 수도 있습니다. Elastic Disaster Recovery는 운영 체제, 설치된 모든 애플리케이션 및 모든 데이터베이스를 포함한 전체 가상 머신을 스테이징 영역으로 복제하는 에이전트 기반 솔루션입니다. 자세한 내용은 Elastic Disaster Recovery 설명서의 Elastic Disaster Recovery란 무엇입니까?를 참조하세요.
- AWS Database Migration Service (AWS DMS) -는 다른 리전을 AWS포함하여 데이터를 주고받는 실시간 마이그레이션을 AWS DMS 지원합니다. 이 기능을 사용하여 재해 복구 데이터베이스 역할을 하도록 다른 리전에 별도의 SQL Server 인스턴스를 설정할 수 있습니다. 자세한 내용은 AWS DMS 설명서의 What is AWS Database Migration Service?를 참조하세요.

SQL Server Always On 가용성 그룹

고가용성 Always On 가용성 그룹을 위해서만 SQL Server Enterprise 에디션을 사용하는 경우 기본 가용성 그룹을 활용하여 SQL Server Standard 에디션으로 다운그레이드할 수 있습니다. Always On 가용성 그룹 대신 기본 가용성 그룹을 사용하여 비용을 65~75% 절감할 수 있습니다.

> **Note**
>
> 여러 SQL Server 에디션 간의 비용 차이에 대한 자세한 내용은이 가이드의 SQL Server 에디션 비교 섹션을 참조하세요.

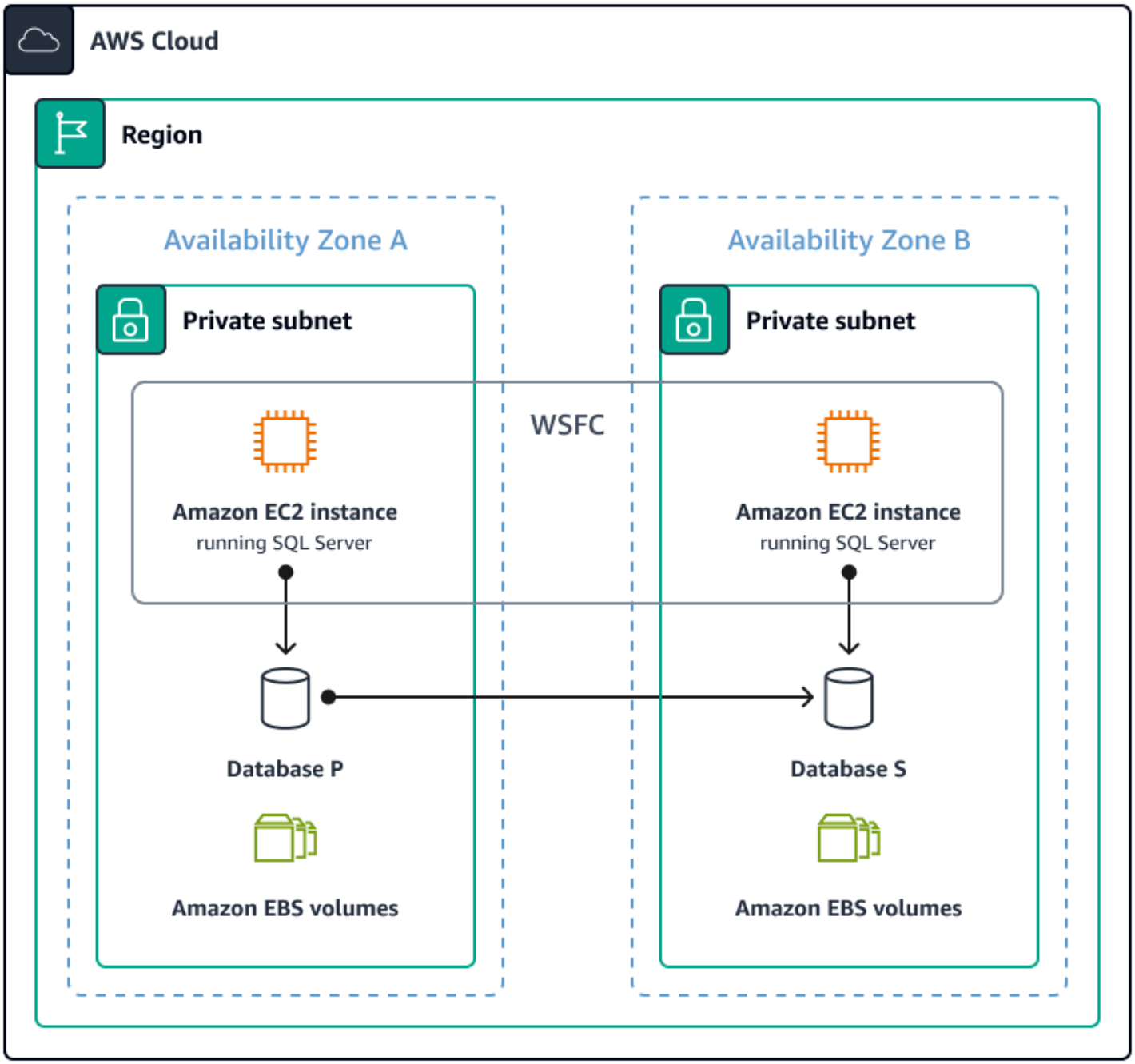
Features

- SQL Server Standard 에디션에서 사용 가능
- 복제본 2개 제한(기본 및 보조)
- 보조 복제본에 대한 읽기 액세스 없음
- 보조 복제본에 대한 무결성 검사 없음

제한 사항

- 가용성 그룹당 하나의 가용성 데이터베이스만 지원
- 기본 가용성 그룹은 분산 가용성 그룹의 일부가 될 수 없습니다.

다음 다이어그램은 Windows Server 장애 조치 클러스터 솔루션의 아키텍처 예제를 보여줍니다.



SQL Server Always On 장애 조치 클러스터 인스턴스

장애 조치 클러스터 인스턴스(FCIs)를 사용하여 가동 중지 시간을 최소화하고 데이터 손실 위험을 줄이면서 지속적인 데이터베이스 운영을 보장할 수 있습니다. FCIs 읽기 전용 복제본 구성 없이 SQL Server 데이터베이스의 고가용성을 원하는 경우 신뢰할 수 있는 솔루션을 제공합니다.

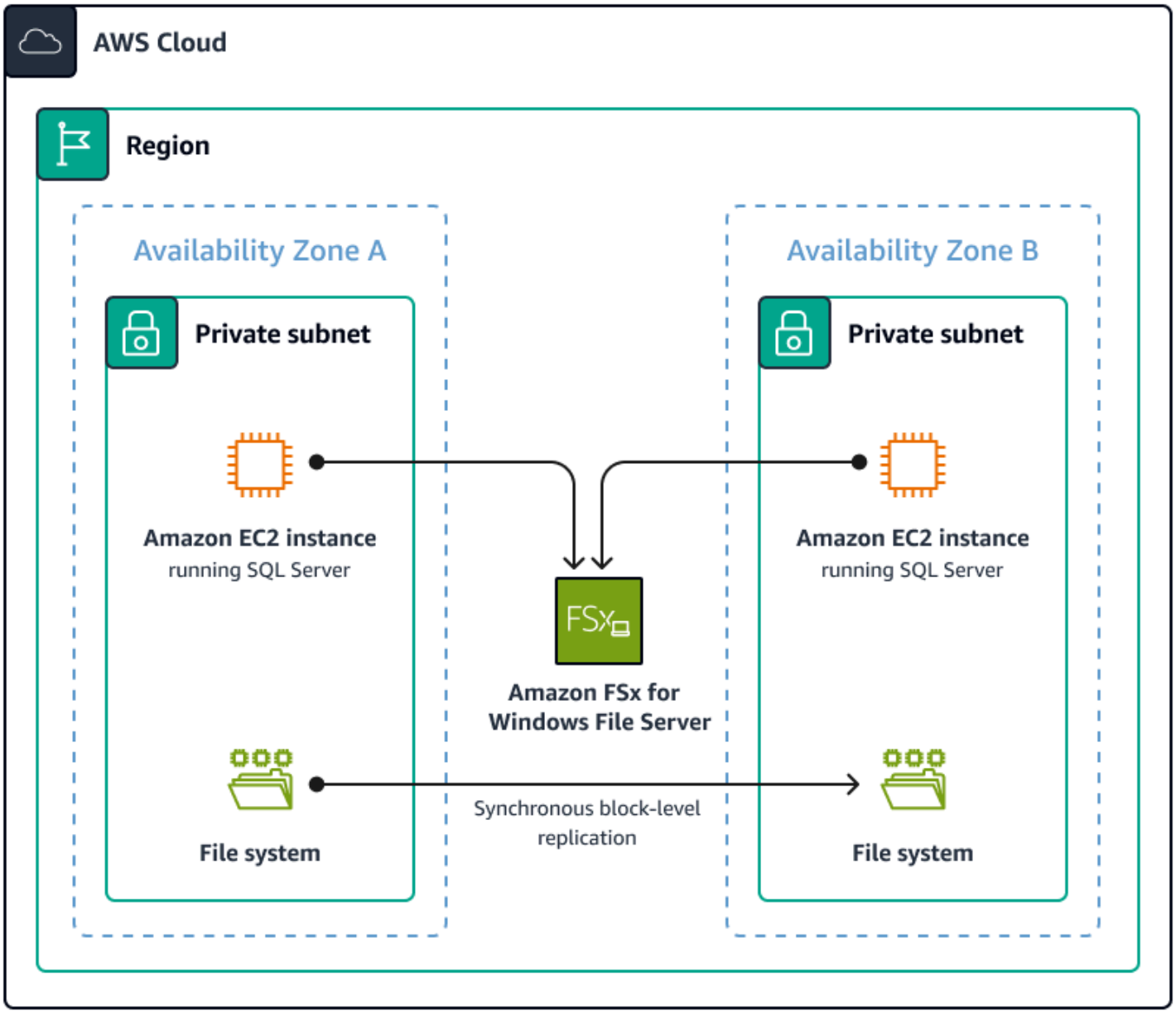
가용성 그룹과 달리 FCIs SQL Server Enterprise Edition 없이도 신뢰할 수 있는 장애 조치 솔루션을 제공할 수 있습니다. 대신 FCIs SQL Server Standard Edition 라이선스만 필요합니다. FCIs 사용하여 SQL Server 라이선스 비용을 65~75% 줄일 수 있습니다.

> **Note**
>
> SQL Server 버전 간의 비용 차이에 대한 자세한 내용은이 가이드의 SQL Server 버전 비교 섹션을 참조하세요.

다음을 고려하세요.

- Amazon FSx for Windows File Server는 SQL Server FCI 공유 스토리지 요구 사항을 충족하기 위한 강력한 솔루션을 제공합니다. FSx for Windows File Server를 사용하면 스토리지 복제 솔루션용 라이선스를 구매하고 공유 스토리지를 직접 관리할 필요가 없습니다. 이로 인해 30~40%의 상당한 비용 절감 효과를 얻을 수 있습니다. 자세한 내용은 AWS 스토리지 블로그의 Amazon FSx for Windows File Server를 사용하여 Microsoft SQL Server 고가용성 배포 간소화 게시물을 참조하세요.
- Software Assurance 혜택 요약(다운로드 가능한 PDF) 및 기존 보유 라이선스 사용(BYOL) 모델을 사용하면 보조 서버가 패시브인 한 패시브 장애 조치 혜택을 활용할 수 있습니다. 따라서 클러스터의 패시브 노드에 라이선스를 제공할 필요가 없으므로 SQL 라이선스 비용이 절감됩니다.

다음 다이어그램은 FSx for Windows File Server를 사용한 SQL Server FCI의 예제 아키텍처를 보여줍니다.



SIOS DataKeeper

SQL Server FCIs를 배포할 계획이라면 공유 스토리지 요구 사항을 고려하는 것이 좋습니다 AWS. 기존 온프레미스 설치는 일반적으로 공유 스토리지 요구 사항을 충족하기 위해 스토리지 영역 네트워크(SAN)를 사용하지만 실행 가능한 옵션은 아닙니다 AWS. Amazon FSx for Windows File Server는의 SQL Server FCI에 권장되는 스토리지 솔루션 AWS이지만 다른에 클러스터 서버를 추가하지 못하게 하는 제한 사항이 있습니다 AWS 리전.

SIOS DataKeeper를 사용하여 가용 영역과 리전을 모두 포함하는 SQL Server FCI를 생성하는 동시에 비용을 58~71% 절감할 수 있습니다. SIOS DataKeeper는 FCI의 고가용성 이점을 달성하는 데 도움이

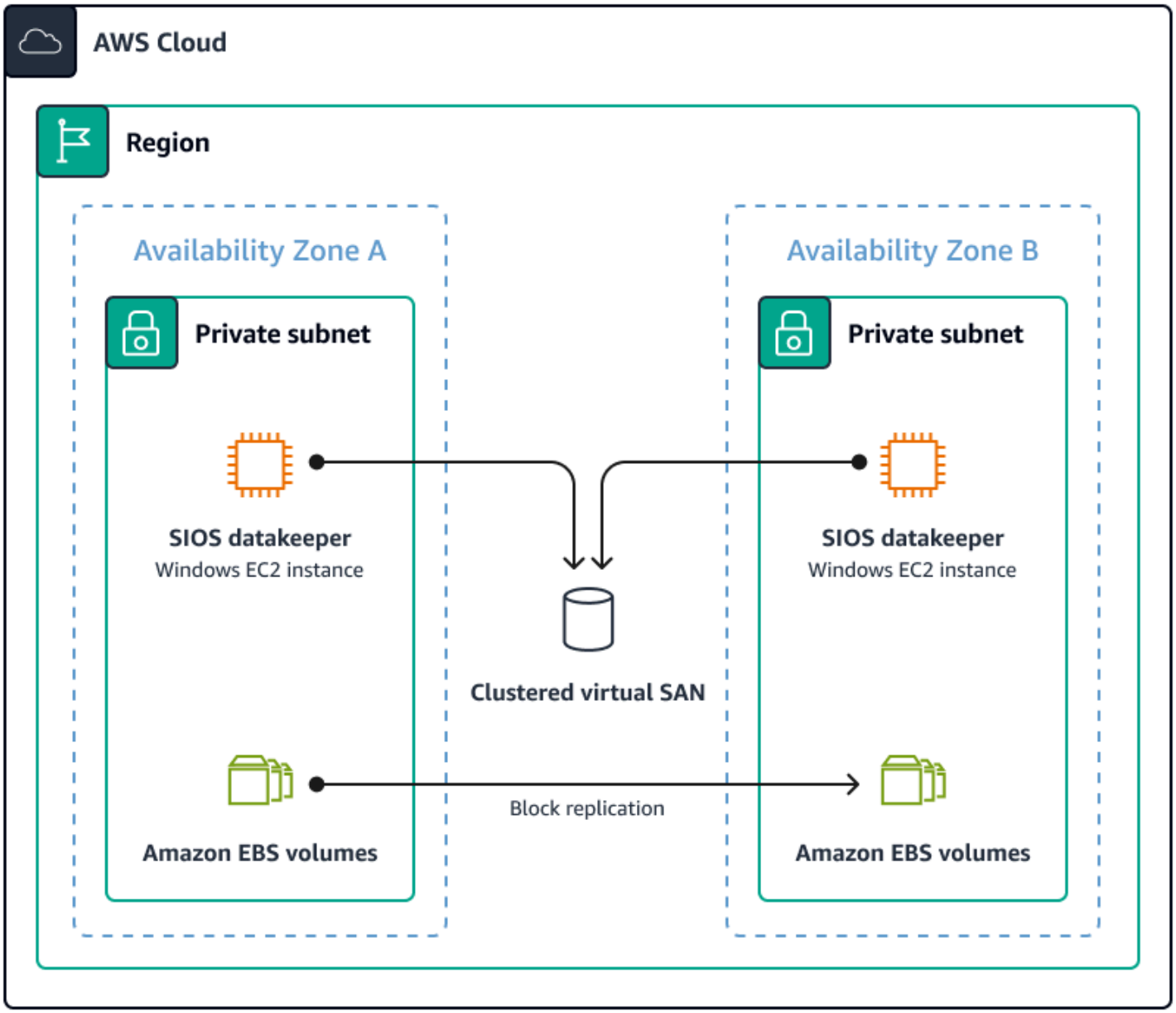
될 수 있습니다. 따라서 SIOS DataKeeper는 조직을 위한 비용 효율적이고 신뢰할 수 있는 솔루션입니다.

SIOS DataKeeper 사용 시 다음과 같은 추가 이점을 고려하세요.

- SIOS DataKeeper는 로컬 EBS 볼륨을 사용하여 클러스터링된 가상 SAN을 생성하고 고가용성을 위해 가용 영역 간의 동기식 복제를 사용합니다. 재해 복구를 위해 SIOS DataKeeper는 리전 간 비동기 복제를 사용합니다.
- SIOS DataKeeper는 SQL Server Standard 에디션을 사용하여 엔터프라이즈급 클러스터링 기능을 제공합니다. 이렇게 하면 SQL Server Enterprise 에디션을 사용하는 SQL Server Always On 가용성 그룹을 사용하여 고가용성을 구현하는 것에 비해 SQL Server 라이선스 비용이 65~75% 절감됩니다. SIOS DataKeeper를 사용하면 조직의 요구 사항을 충족하는 가용성, 유연성 및 비용 효율적인 SQL Server 환경을 만들 수 있습니다.

> **Note**
>
> SQL Server 에디션 간의 비용 차이에 대한 자세한 내용은이 가이드의 SQL Server 에디션 비교 섹션을 참조하세요.

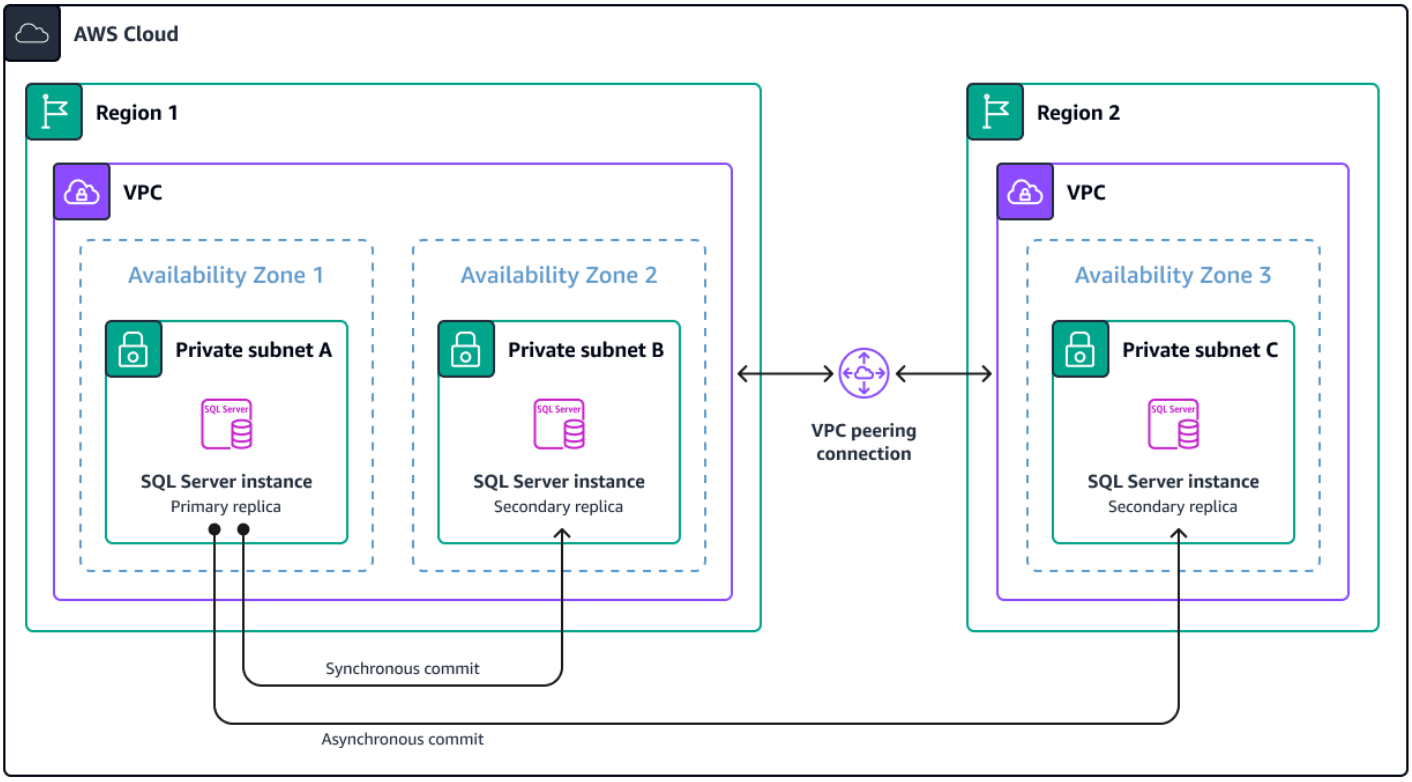
다음 다이어그램은 클러스터링된 가상 SAN 솔루션을 사용하는 SQL Server FCI의 아키텍처 예제를 보여줍니다.



Always On 가용성 그룹

고가용성 및 재해 복구를 위해 Always On 가용성 그룹을 사용할 수 있습니다. 한 리전의 두 가용 영역에 SQL Server를 배포하여 고가용성을 달성할 수 있습니다. 리전 간에 가용성 그룹을 확장하여 재해 복구를 달성할 수 있습니다.

다음 다이어그램은 Always On 가용성 그룹을 기반으로 하는 솔루션의 아키텍처 예제를 보여줍니다. 다이어그램의 리전 1에 있는 복제본은 가용성 그룹의 자동 장애 조치를 제공하는 동기 커밋을 사용합니다. 리전 2의 복제본은 가용성 그룹의 수동 장애 조치가 필요한 비동기 커밋을 사용하고 있습니다.

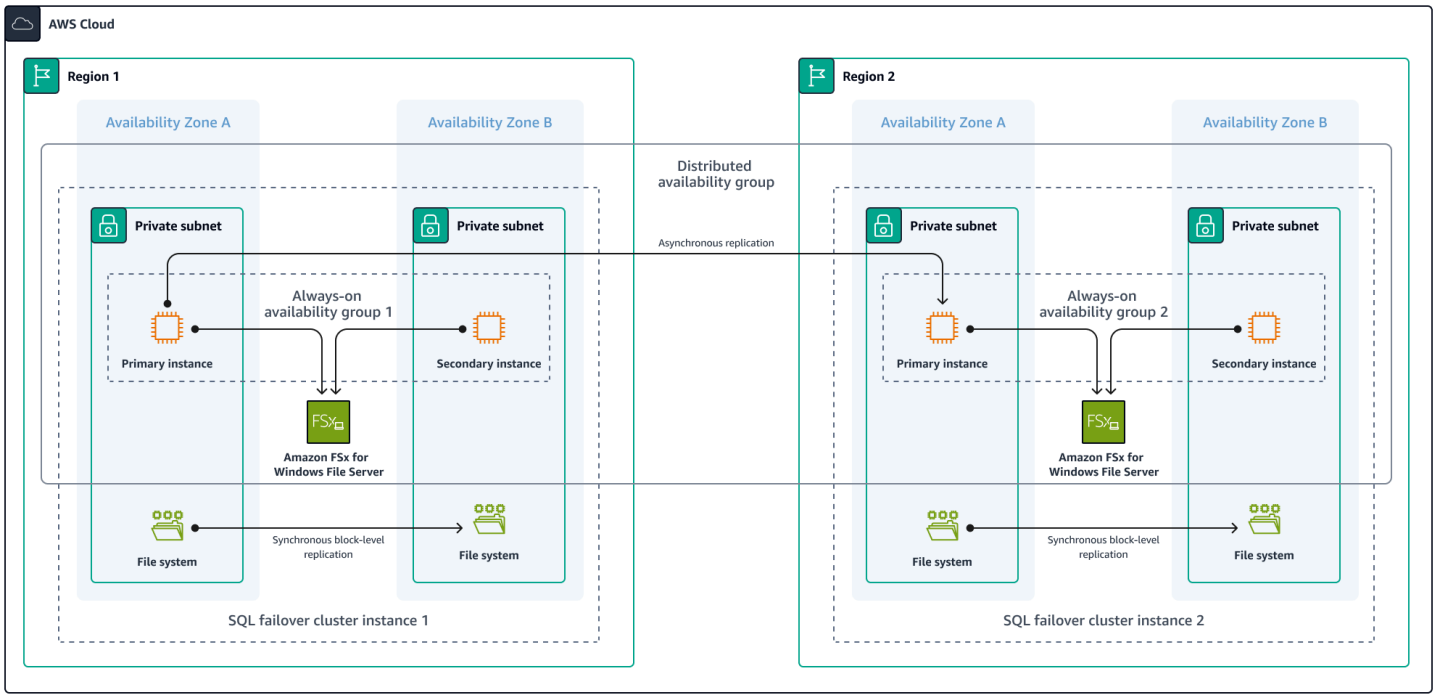


분산 가용성 그룹

신뢰성 또는 재해 복구를 위태롭게 할 수 없는 미션 크리티컬 SQL Server 배포의 경우 다중 리전 접근 방식을 사용하는 것이 좋습니다. 여러 리전에 걸쳐 가용성 그룹을 배포하는 것은 비즈니스 연속성을 유지하고 가동 중지 시간을 최소화하기 위한 가장 복원력이 뛰어난 솔루션입니다.

이 아키텍처는 공유 스토리지, 동기식 블록 수준 복제, SQL Server FCIs 등 Amazon FSx for Windows File Server의 기능을 최대한 활용합니다. 이러한 기능을 사용하면 여러 가용 영역에 걸쳐 가용성이 높은 SQL Server 환경을 생성할 수 있습니다. 다른 리전에서이 설정을 복제하면 가장 심각한 중단도 처리할 수 있는 완전 중복 시스템을 얻을 수 있습니다. 이 솔루션을 차별화하는 요소는 유연성과 보안 수준입니다. 분산 가용성 그룹의 도메인 독립적 아키텍처를 사용하면 기본 Windows 클러스터 서버가 서로 다른 Active Directory 도메인에 조인할 수 있으며, 인증서 기반 인증은 SQL Server 환경을 최대한 보호하고 다중 리전 DR 전략에 대한 높은 RTO 및 RPO 요구 사항을 제공합니다. 다중 리전 아키텍처 구축에 대한 자세한 내용은 아키텍처 블로그의 필드 노트: FCI 및 분산 가용성 그룹을 사용하여 SQL Server용 다중 리전 아키텍처 구축을 참조하세요. AWS

다음 다이어그램은 분산 가용성 그룹을 사용하는 다중 리전 솔루션의 아키텍처 예제를 보여줍니다.



로그 전달

로그 전송은 예상치 못한 중단이 발생할 경우 여러 리전에서 데이터베이스를 보호할 수 있는 검증되고 안정적이며 비용 효율적인 방법입니다. 조직은 수십 년 동안 로그 전송을 사용하여 데이터를 보호해 왔습니다.

로그 배송을 구현하면 트랜잭션 빈도와 로그 배송 작업에 따라 몇 분 안에 RPO 및 RTO를 달성할 AWS 수 있습니다. 드물지만 리전에 액세스할 수 없게 되는 경우 로그 전송을 통해 데이터를 안전하게 복구할 수 있습니다.

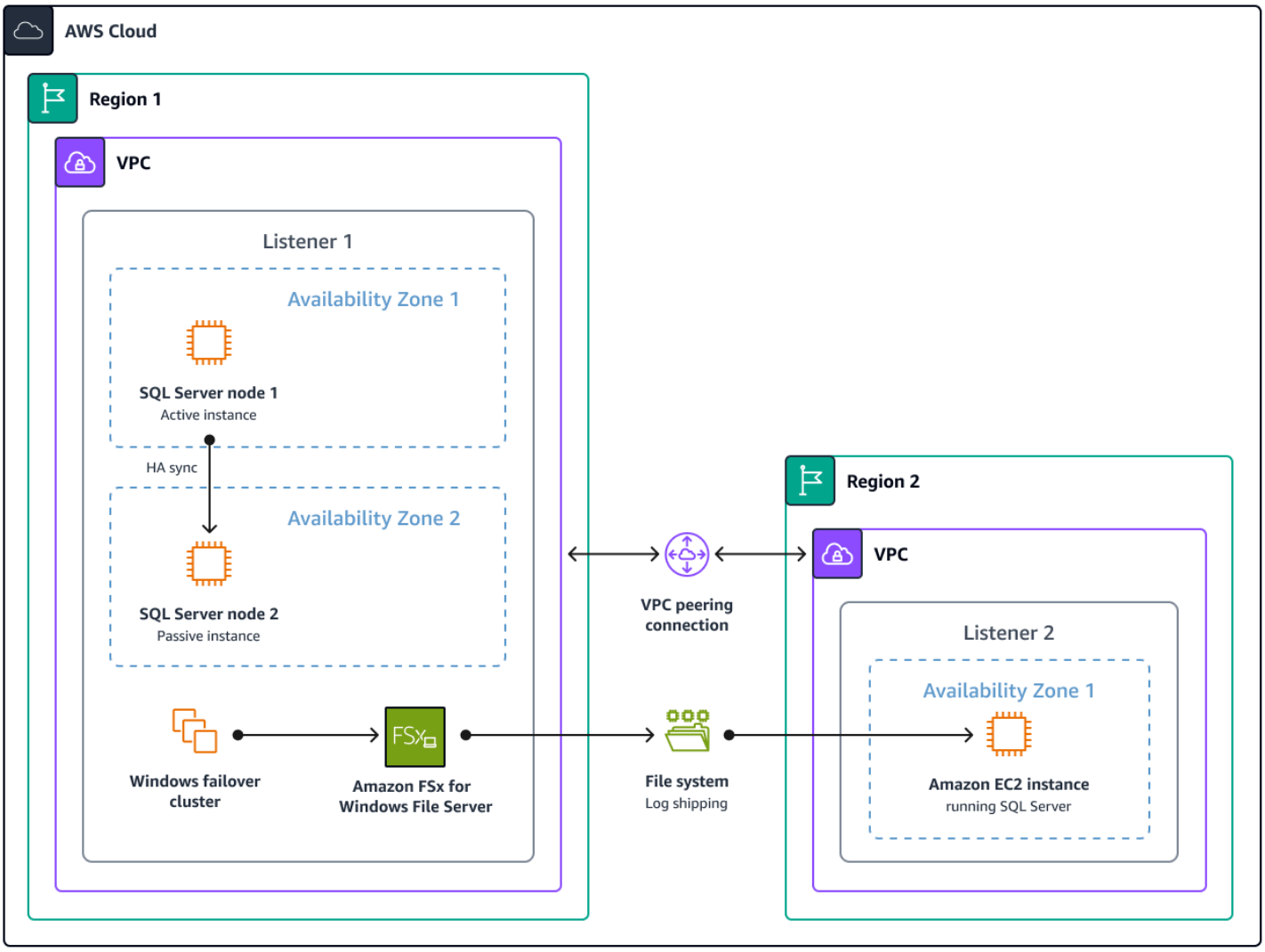
로그 전송을 사용하면 다음과 같은 추가 이점을 고려할 수 있습니다.

- 리전 간 재해 복구 복원성을 위한 로그 전송을 사용하여 비용을 절감하고 비즈니스 요구 사항을 충족합니다. 로그 전송은 SQL Server Standard Edition 또는 SQL Server Web Edition 라이선스만 필요하기 때문에 TCO를 줄입니다.
- 활성 Software Assurance와 함께 로그 전송을 사용하여 재해 복구/패시브 서버에서 라이선스 비용을 제거합니다. Software Assurance에서 로그 배송을 사용하는 경우 기본/활성 SQL Server에 대해서만 라이선스를 부여해야 합니다.
- SQL Server Enterprise 에디션이 리전 간에 분산 가용성 그룹을 설정할 필요가 없으므로 SQL Server 라이선스 비용을 65~75% 절감합니다. 재해 복구 요구 사항을 충족하기 위해 로그 전송과 결합된 SQL Server Standard 에디션 및 SQL Server FCIs 사용하여이 작업을 수행할 수 있습니다.

> **Note**
>
> SQL Server 버전 간의 비용 차이에 대한 자세한 내용은이 가이드의 SQL Server 버전 비교 섹션을 참조하세요.

자세한 내용은 AWS 아키텍처 블로그의 Amazon FSx for Windows 구성에서 SQL Server FCI에 대한 로그 전송을 사용하여 SQL Server DR 확장을 참조하세요.

다음 다이어그램은 로그 배송 솔루션의 아키텍처 예제를 보여줍니다.

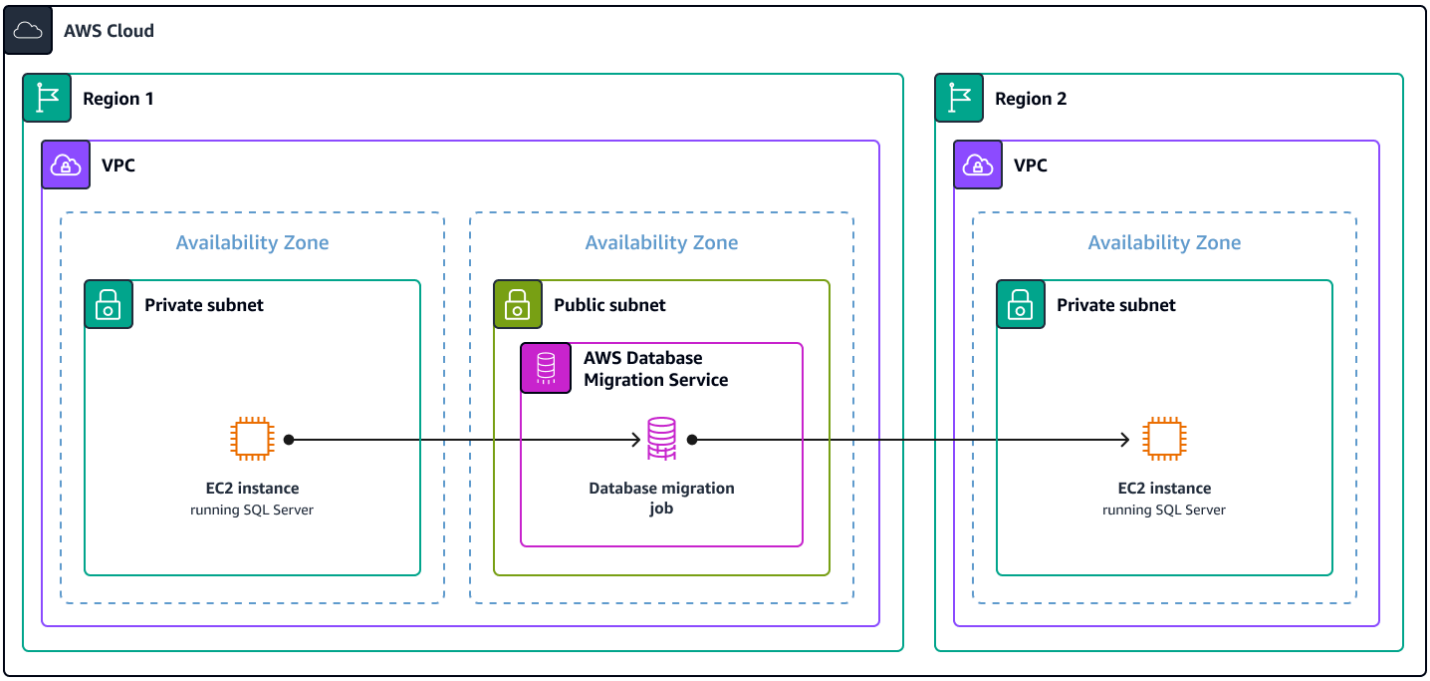


AWS Database Migration Service

AWS Database Migration Service (AWS DMS)를 사용하여 애플리케이션 요구 사항에 따라 HA/DR 솔루션을 설계할 수 있습니다.를 AWS DMS 사용하면 데이터를 동일한 리전(HA) 또는 여러 리전(DR)의 보조 SQL Server 데이터베이스에 쉽게 복사할 수 있습니다. 이 접근 방식은 기술적으로 적합하며 리소스 사용을 최적화하면서 인프라에 대한 AWS 투자를 극대화할 수 있습니다.

AWS DMS 는 비용 효율적인 서비스입니다. 전송 프로세스 및 추가 로그 스토리지 중에 사용된 CPU 리소스에 대해서만 요금이 부과됩니다. 즉, 상당한 추가 비용을 발생시키지 않고도이 솔루션의 이점을 누릴 수 있습니다. AWS DMS 를 사용하여 데이터를 사용할 수 있고 액세스할 수 있는지 확인하는 동시에 라이선스 및 리소스 사용과 관련된 비용을 최소화할 수 있습니다.

다음 다이어그램은 기반 솔루션의 아키텍처 예제를 보여줍니다 AWS DMS.



AWS Elastic Disaster Recovery

일부 조직은 모든 중요한 비즈니스 애플리케이션에 재해 복구 계획이 있는지 확인해야 합니다. 과거에는 이러한 조직 중 다수가 기존 재해 복구 솔루션에 많은 투자를 했기 때문에 전체 중복 인프라를 사전 구축하고 유지 관리해야 했습니다. 이 접근 방식은 비용이 많이 들고 시간이 많이 걸리며 규모 조정이 어렵습니다.

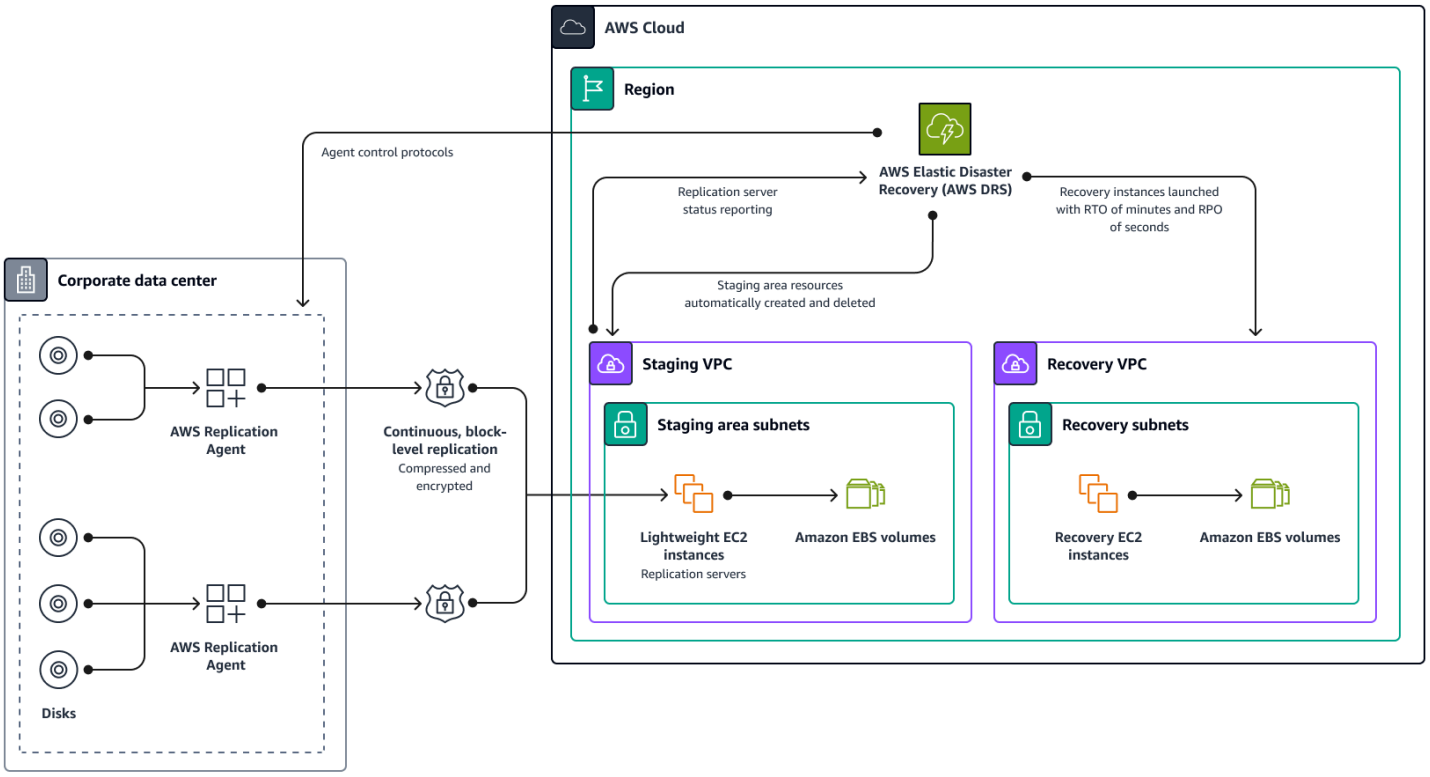
이제 AWS Elastic Disaster Recovery 를 사용하여 재해 복구 인프라를 사전 빌드할 필요가 없습니다. 재해 복구 시스템은 필요할 때까지 Elastic Disaster Recovery에서 시작되지 않으므로 필요할 때 사용

한 만큼만 비용을 지불하면 됩니다. 즉, 소프트웨어 라이선스와 고성능 컴퓨팅 비용을 크게 줄일 수 있습니다.

또한 재해 복구 솔루션의 스테이징 영역에는 저렴한 Amazon Elastic Block Store(Amazon EBS) 볼륨이 포함되어 있습니다. EBS 볼륨은 중복 리소스 프로비저닝 비용을 더욱 줄입니다. 이를 통해 비즈니스 요구 사항을 충족하는 강력하고 안정적인 재해 복구 솔루션을 유지하면서 전체 재해 복구 비용을 절감할 수 있습니다. Elastic Disaster Recovery를 사용하여 핵심 비즈니스 활동에 집중할 수 있으며, AWS 는 재해 복구 솔루션의 기본 인프라를 관리합니다.

SQL Server의 경우 Elastic Disaster Recovery를 비용 효율적인 재해 복구 옵션으로 사용할 수 있습니다. 활성 Software Assurance를 사용하는 경우 내결함성이 뛰어나고 가용성이 높은 SQL Server 아키텍처의 패시브 노드에 대한 라이선스가 적용됩니다. 하지만 패시브 서버가 온라인 상태가 되도록 컴퓨팅 비용을 계속 지불하고 있습니다. Elastic Disaster Recovery를 사용하면 기본 서버는 활성 Software Assurance를 유지 관리하고 재해 복구 컴퓨팅 비용을 지불할 필요 없이 DR 환경에 복제할 수 있습니다. 이러한 비용 절감을 결합하면 SQL Server 재해 복구 비용을 50% 이상 줄일 수 있습니다.

다음 다이어그램은 Elastic Disaster Recovery를 기반으로 하는 솔루션의 아키텍처 예제를 보여줍니다.



자세한 내용은 AWS 블로그의 Microsoft 워크로드에서 [를 사용하여 복원된 DR 사이트에서 SQL Server의 고가용성을 설정하는 방법을 AWS Elastic Disaster Recovery] 참조하세요.

비용 비교

다음 표에서는이 섹션에서 다루는 HA/DR 솔루션의 비용을 비교합니다. 이 비교를 위해 다음과 같은 가정이 이루어집니다.

- 인스턴스 유형 - r5d.xlarge
- 라이선스 유형 - Windows 및 SQL Server 모두에 포함된 라이선스
- 리전 – `us-east-1`

Solution	높은 가용성	재해 복구	엔터프라이즈 에디션	스탠다드 에디션	비용
로그 전달	아니요	예	예	예	SQL Server Enterprise Edition: 32,674.8 USD(노드 2개) SQL Server Standard 에디션: 14,804.4 USD(노드 2개)
Always On 가용성 그룹	예	예	예	예, 하지만 기본 가용성 그룹(노드 2개)	SQL Server Enterprise Edition: 32,674.8 USD(노드 2개) SQL Server Standard 에디션: 14,804.4

Solution	높은 가용성	재해 복구	엔터프라이즈 에디션	스탠다드 에디션	비용
					USD(노드 2개)
상시 작동 FCIs	예	아니요	예	예(노드 2개)	SQL Server Standard 에디션: 14,804.4 USD
분산 가용성 그룹	예	예	예	아니요	SQL Server Enterprise Edition: 65,349.6 USD(노드 4개)

Solution	높은 가용성	재해 복구	엔터프라이즈 에디션	스탠다드 에디션	비용
Elastic Disaster Recovery	아니요	예	예	예	근사치입니다. 인스턴스 1개와 스토리지 1TB 복제에 대해 월 107.48 USD 참고: Elastic Disaster Recovery는 복제 서버당 시간당 요금이 청구됩니다. 비용은 디스크 수, 스토리지 크기, 드릴 또는 복구 시작 횟수 또는 복제하려는 리전에 관계없이 동일합니다.

Solution	높은 가용성	재해 복구	엔터프라이즈 에디션	스탠다드 에디션	비용
SIOS 데이터 키퍼	예	예	예	예	Software Assurance를 사용하는 Always On 가용성 그룹(노드 2개, 코어 24개): 213,480 USD SIOS DataKeeper 및 Software Assurance가 포함된 SQL Server Standard 에디션에서 실행되는 2노드 SQL Server 클러스터: 61,530 USD(2노드)
AWS DMS	아니요	예	예	예	r5.xlarge 인스턴스 및 1TB 스토리지의 경우 월 745.38 USD

비용 최적화 권장 사항

조직의 요구 사항을 충족하는 HA/DR 솔루션을 선택하려면 다음 단계를 수행하는 것이 좋습니다.

- 이 가이드의 SQL Server 워크로드에 적합한 EC2 인스턴스 선택 섹션을 검토하세요.

- 피크 워크로드 중에 성능 카운터를 실행하여 워크로드의 IOPS 및 처리량 요구 사항을 결정합니다.
  - IOPS = 디스크 읽기/초 + 디스크 쓰기/초
  - 처리량 = 디스크 읽기 바이트/초 + 디스크 쓰기 바이트/초
- 성능 향상 및 비용 절감을 위해 다음 스토리지 볼륨 유형을 사용합니다.
  - `tempdb` 및 버퍼 풀 확장을 위한 NVMe 인스턴스 스토리지
  - 데이터베이스 파일의 io2 볼륨
- Amazon EC2의 SQL Server에 대한 비용 최적화 권장 사항에 AWS Trusted Advisor를 사용합니다. SQL Server 최적화 검사를 수행하기 Trusted Advisor 위해 용 에이전트를 설치할 필요가 없습니다. 는 가상 CPUs(vCPUs), 버전 및 에디션과 같은 Amazon EC2 SQL Server 라이선스 포함 인스턴스 구성을 Trusted Advisor 검사합니다. 그런 다음 모범 사례를 기반으로 추천을 Trusted Advisor 합니다.
- Amazon EC2 인스턴스와 Amazon EBS 올바른 크기 조정 권장 사항에 AWS Compute Optimizer 모두 사용합니다.
- AWS Pricing Calculator를 사용하여 비용 추정을 위한 HA/DR 전략을 설계합니다.
- SQL Server Enterprise Edition에서 SQL Server Standard Edition으로의 다운그레이드가 가능한 옵션인지 확인하려면 sys dm_db_persisted_sku_features 동적 관리 보기를 사용하여 현재 데이터베이스에서 활성 상태인 에디션별 기능을 식별합니다.

> **Note**
>
> 라이선스 포함 EC2 인스턴스를 사용할 때 SQL Server 에디션을 변경하려면 Side-by-side 마이그레이션해야 합니다.

- 정의된 RTO 및 RPO로 데이터베이스를 복구할 수 있는 설계를 더 잘 설계하려면 반년 또는 연간 재해 복구 훈련을 수행합니다. 또한 아키텍처 약점을 식별하는 데 도움이 될 수 있습니다.

추가 리소스

- Amazon FSx for Windows File Server를 사용하여 Microsoft SQL Server 고가용성 배포 간소화(AWS 스토리지 블로그)
- 필드 노트: FCI 및 분산 가용성 그룹을 사용하여 SQL Server용 다중 리전 아키텍처 구축(AWS 아키텍처 블로그)
- SQL Server에 대한 재해 복구 설계 AWS: 1부(AWS 데이터베이스 블로그)
- Amazon FSx for Windows(YouTube)를 사용한 Microsoft SQL 고가용성 YouTube

- Amazon EBS를 사용하여 Microsoft SQL Server 성능 극대화(AWS 스토리지 블로그)
- 온프레미스 스토리지 패턴과 AWS 스토리지 서비스 비교(AWS 스토리지 블로그)
- 데이터 센터 NAS를 Amazon FSx File Gateway로 대체할 계획(AWS 스토리지 블로그)
- 의 고가용성 SQL Server 배포 비용 최적화 AWS(AWS 스토리지 블로그)
- (Microsoft Workloads on AWS)를 사용하여 SQL Server Always On 가용성 그룹에 대한 재해 복구를 설정하는 방법 AWS Elastic Disaster Recovery
- 를 사용하여 복원된 DR 사이트에서 SQL Server의 고가용성을 설정하는 방법 AWS Elastic Disaster Recovery(Microsoft Workloads on AWS)

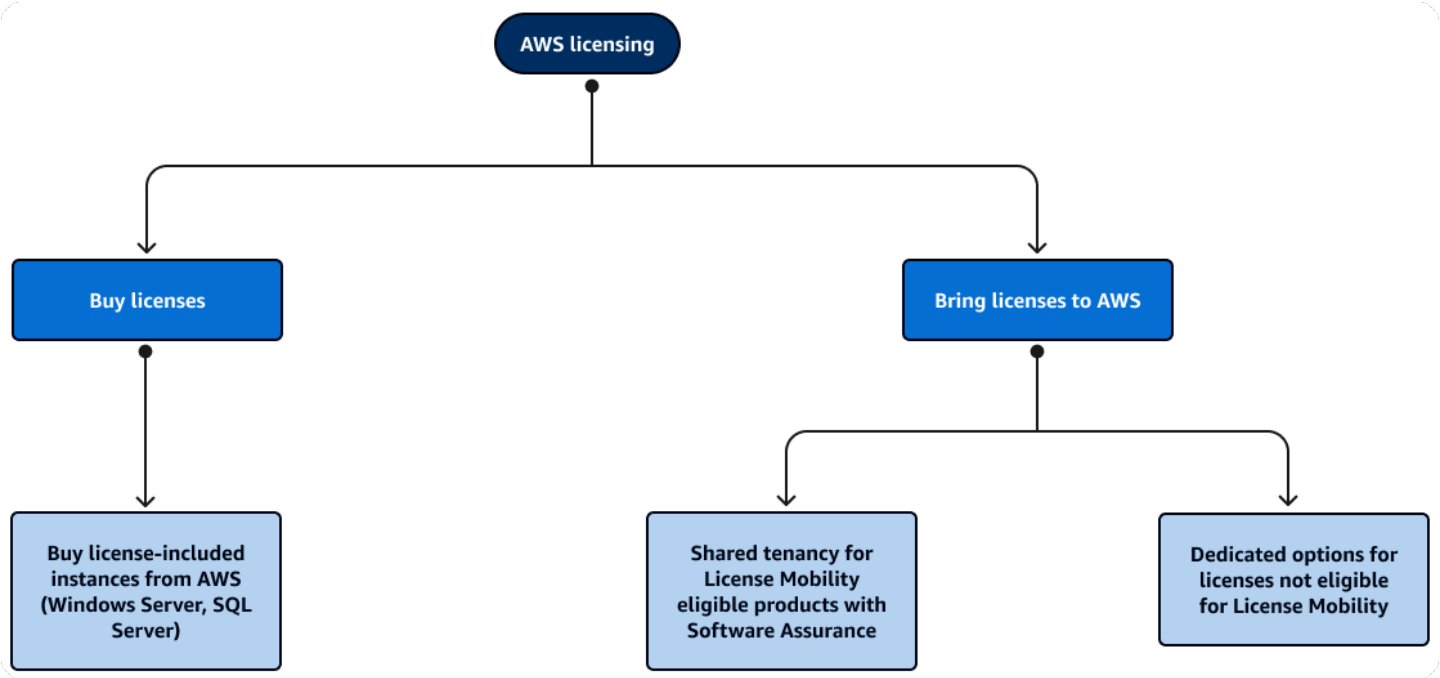
SQL Server 라이선스 이해

개요

점점 더 많은 기업이 워크로드를 클라우드로 이전함에 따라 클라우드 플랫폼의 비용 최적화가 최우선 순위가 되었습니다. 라이선싱은에서 Microsoft 워크로드 실행과 관련된 가장 중요한 비용 중 하나입니다 AWS. 이 섹션에서는 SQL Server용 Microsoft 라이선스를 최적화 AWS 하여의 비용을 최적화하는 방법을 설명합니다.

AWS 라이선스 옵션

AWS 는 라이선스를 위한 다양한 유연한 비용 최적화 옵션을 제공합니다. 이러한 라이선스 옵션은 비용을 절감하고 규정 준수를 유지하며 비즈니스 요구 사항을 충족하는 데 도움이 되도록 설계되었습니다.



AWS 는 라이선스를 세 가지 주요 유형으로 분류합니다.

1. 라이선스 포함 -이 라이선스 옵션을 사용하면 온디맨드 방식으로 라이선스를 구매하고 사용할 수 있으며 사용한 만큼만 비용을 지불할 수 있습니다. 라이선스 포함 옵션은 라이선스 사용에 유연성이 필요하고 선결제 비용을 피하려는 시나리오에 적합합니다. Windows Server, SQL Server 및 기타 Microsoft 제품 중에서 선택할 수 있습니다.
2. 라이선스 이동성이 있는 기존 라이선스 사용(BYOL) 제품 -이 라이선스 옵션은 기존 라이선스가 이미 있고 클라우드에서 사용하려는 시나리오를 위해 설계되었습니다.는 고객이 Microsoft의 라이선스 이동 프로그램을 통해 자체 라이선스를 클라우드로 가져올 수 있도록 AWS 허용합니다. SQL Server with Software Assurance(SA)와 같이 라이선스 이동성이 있는 제품을 공유 또는 전용 테넌시로 가져와 AWS 인스턴스 비용을 줄일 수 있습니다.
3. 라이선스 이동성이 없는 BYOL 제품 - Windows Server와 같이 라이선스 이동성이 없는 Microsoft 제품의 경우 클라우드에서 이러한 제품을 사용할 수 있는 전용 옵션을 AWS 제공합니다. 또한 전용 호스트는 물리적 코어 수준에서 라이선스를 부여할 수 있는 기회를 제공합니다. 이렇게 하면 워크로드를 실행하는 데 필요한 라이선스를 50% 이상 절약할 수 있습니다. 전용 호스트는 대부분의 시간에 실행되는 안정적이고 예측 가능한 워크로드에 적합한 옵션입니다.

라이선스 가져오기로 인한 비용 영향

라이선스를 가져오면 Microsoft 워크로드 실행 비용에 상당한 영향을 미칠 수 있습니다 AWS. 자체 라이선스를 사용하는 경우 클라우드에서 실행되는 인스턴스에 대한 추가 라이선스 비용을 지불할 필요가 없습니다. 이로 인해 상당한 비용 절감이 발생할 수 있습니다.

다음 비교는 24/7 단일 c5.xlarge 인스턴스를 실행하는 온디맨드 월별 비용을 보여줍니다.

- Windows Server + SQL Server Enterprise Edition: $1353/월(라이선스 포함)
- Windows Server + SQL Server Standard Edition: $609/월(라이선스 포함)
- Windows Server만 해당: 월 259 USD(라이선스 포함)
- 컴퓨팅 전용(Linux): $127/월

궁극적으로 자체 라이선스를 가져오면 Microsoft 워크로드 실행 비용에 상당한 영향을 미칠 수 있습니다 AWS. 기존 라이선스를 사용하는 경우 라이선스 비용을 절감하고 전체 AWS 청구서 비용을 절감할 수 있습니다.

라이선스 최적화

AWS 최적화 및 라이선스 평가(AWS OLA)는 컴퓨팅 및 라이선스 비용을 줄여 라이선스를 최적화하는 데 도움이 될 수 있습니다. AWS OLA는에서 실행되는 워크로드 AWS 또는 마이그레이션이 계획된 워크로드에 대한 라이선스 요구 사항을 평가하도록 설계되었습니다. AWS OLA는 라이선스 사용 최적화에 대한 권장 사항을 제공합니다.

라이선스 사용을 최적화하기 위한 주요 전략 중 하나는 적절한 크기의 인스턴스입니다. 크기를 올바르게 조정하려면 CPU, 메모리 및 스토리지 요구 사항에 따라 워크로드에 적합한 인스턴스 유형을 선택해야 합니다. 적절한 인스턴스 크기를 선택하면 비용 효율적인 방식으로 리소스를 사용하고 있는지 확인할 수 있습니다. 이로 인해 상당한 비용 절감이 발생할 수 있습니다.

Microsoft 소프트웨어 라이선스를 사용하면 소프트웨어가 실행되는 코어 수가 라이선스 비용을 결정하는 데 중요한 요소입니다. 예를 들어 Windows Server 및 SQL Server 라이선스는 일반적으로 코어 수에 따라 라이선스가 부여됩니다. 인스턴스 크기를 적절하게 조정하면 Microsoft 소프트웨어가 실행되는 코어 수를 줄이고 인스턴스 비용과 필요한 라이선스 수를 모두 줄일 수 있습니다.

비용 최적화 권장 사항

라이선스 최적화는 비용 최적화의 핵심 구성 요소입니다 AWS. 올바른 전략을 구현하면 라이선스 비용을 줄이고, 규정 준수를 유지하고, 라이선스 투자에서 가능한 최상의 가치를 얻을 수 있습니다. 이 섹션에서는 라이선스 최적화를 위한 몇 가지 전략을 간략하게 설명합니다.

적격 Windows Server 라이선스 사용

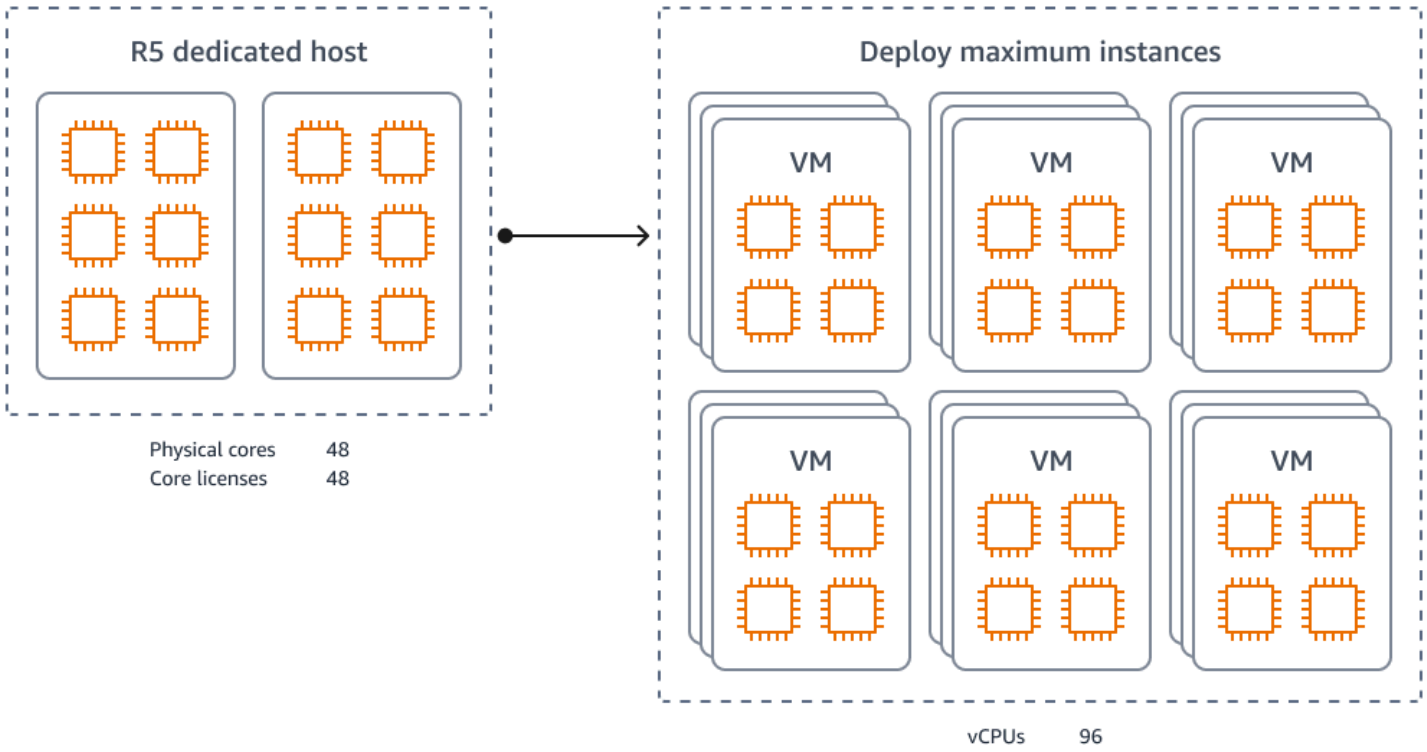
자체 Windows Server 라이선스를 가져오는 것은 라이선스 최적화를 위한 가장 효과적인 전략 중 하나입니다. 이 전략을 사용하면 기존 투자를 활용하여 AWS 지출을 줄일 수 있습니다.

예를 들어 1/10/2019 이전에 라이선스를 구매했거나 해당 날짜 이전에 서명된 활성 엔터프라이즈 계약에 따라 라이선스를 트루업으로 구매한 경우 Amazon EC2 전용 호스트에 Windows Server 2019 및 이전 버전을 배포할 수 있습니다. 이 규칙은 Microsoft가 2019년에 나열된 공급자(예: Alibaba AWS또는 Google Cloud)에 배포할 때 Windows Server와 같이 라이선스 이동이 없는 제품에 대한 라이선스 이용 약관을 변경한 것을 기반으로 합니다. 새 약관에 따라 자체 Windows Server 라이선스를 로 가져올 수 AWS 없지만 대신 라이선스 포함 인스턴스를 사용해야 합니다. 그러나 해당 날짜 이전에 영구 라이선스를 구매한 경우에도 Amazon EC2 전용 호스트에 해당 Windows Server 라이선스를 배포할 수 있습니다.

물리적 수준 라이선스

물리적 코어 수준에서 라이선스를 부여하면 호스트의 물리적 코어만 라이선스를 부여할 수 있으므로 필요한 라이선스 수에 영향을 주지 않고 최대 인스턴스 수를 배포할 수 있습니다. 이는 일반적으로 Windows Server Datacenter 및 SQL Server Enterprise 에디션을 사용하여 수행됩니다.

예를 들어 코어가 48개이고 vCPUs가 96개로 변환되는 R5 전용 호스트를 고려해 보세요. Windows Server Datacenter 에디션을 사용하는 경우 48개의 라이선스만 있으면 됩니다. 이렇게 하면 다음 다이어그램과 같이 최대 96개의 vCPUs가 있는 인스턴스 조합을 배포할 수 있습니다.



이 접근 방식은 호스트에서 실행할 수 있는 인스턴스 수를 최대화할 수 있는 워크로드가 충분한 경우 특히 비용 효율적일 수 있습니다. 물리적 코어 수준에서 라이선스를 부여하면 각 인스턴스에 대한 추가 라이선스 비용을 방지하고 라이선스 투자에 대해 가능한 최상의 가치를 얻을 수 있습니다.

SQL Server의 물리적 코어 수준의 라이선스

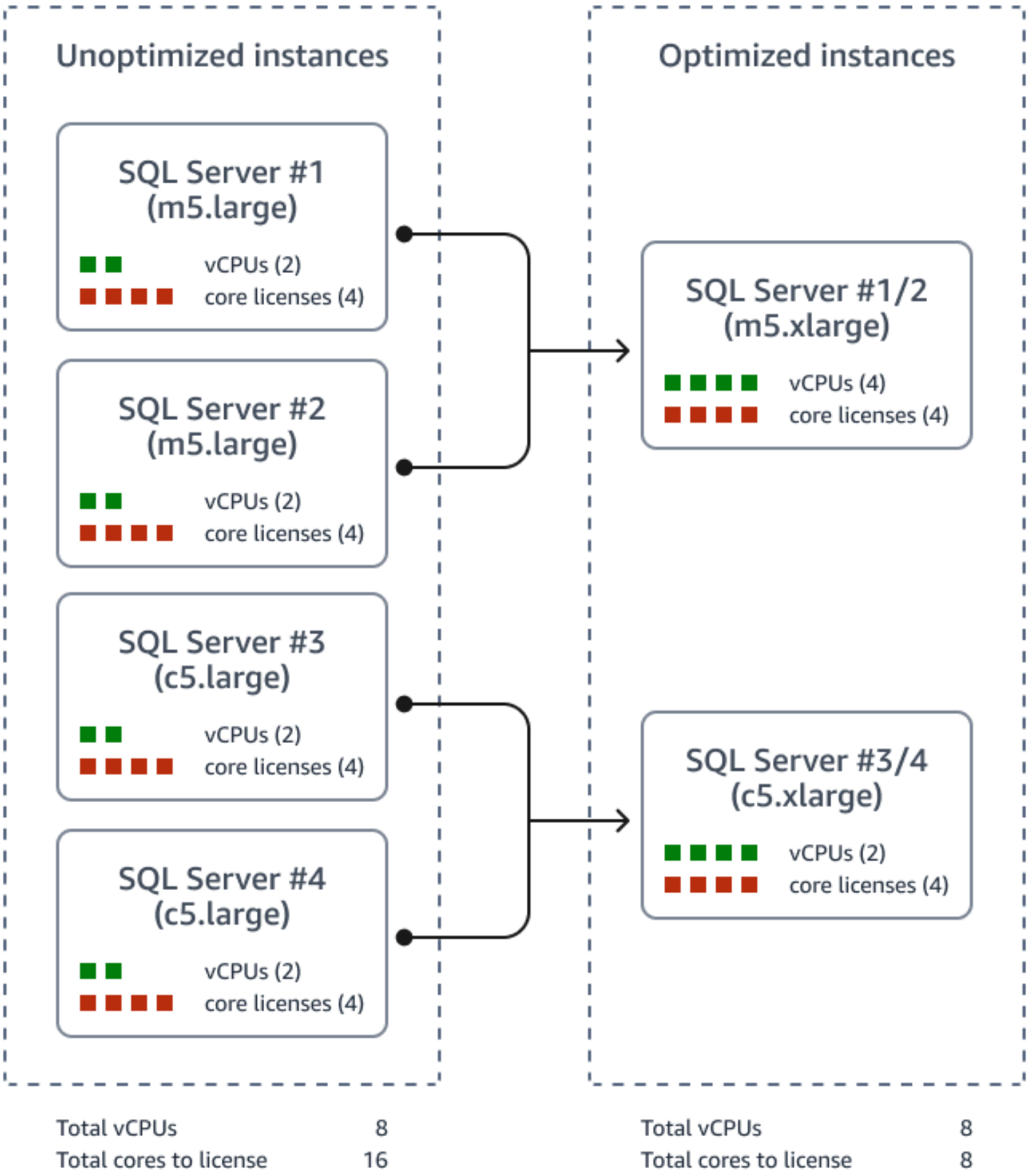
공유 테넌시에서 SQL Server 라이선스는 인스턴스에 할당된 vCPUs 수를 기반으로 합니다. 반면 전용 호스트를 사용하면 물리적 코어 수준 또는 vCPU 수준에서 SQL Server Enterprise 에디션에 라이선스를 부여할 수 있습니다.

R5 전용 호스트의 이전 예제와 마찬가지로 물리적 코어 수준에서 SQL Server Enterprise 에디션에 라이선스를 부여하면 호스트에 라이선스를 부여하려면 48개의 SQL Server Enterprise 에디션 라이선스만 필요합니다. 반면 vCPU로 라이선스를 부여하는 것이 유일한 옵션인 공유 테넌시에서는 동일한 워크로드에 대해 96개의 SQL Server Enterprise 에디션 라이선스가 있어야 합니다. 따라서 전용 호스트는 공유 테넌시에 비해 SQL Server 라이선스 비용을 최대 50% 절감할 수 있습니다. 이는 적격 Windows 라이선스를 가져와 인스턴스 비용을 절감하는 것 외에도 가능합니다.

SQL Server 인스턴스 통합

SQL Server 통합은 여러 SQL Server 인스턴스를 하나의 서버로 결합하는 프로세스입니다. 인스턴스에 vCPUs가 2개뿐인 경우에도 SQL Server에는 인스턴스당 최소 4개의 코어 라이선스가 필요합니다.

즉, 코어가 4개 미만인 서버에서 SQL Server를 실행하면 이러한 인스턴스의 라이선스를 과도하게 부여하고 필요한 것보다 더 많은 라이선스를 사용할 수 있습니다.



예를 들어 vCPUs 2개가 있는 인스턴스 2개를 vCPUs 4개가 있는 단일 인스턴스로 통합하면 라이선스 요구 사항을 50% 줄일 수 있습니다. 코어 라이선스는 8개가 아니라 4개만 필요하기 때문입니다.

통합에 대한 자세한 내용은이 가이드의 SQL Server 통합 섹션을 참조하세요.

SQL Server 에디션 다운그레이드

SQL Server 에디션 변경은 라이선스 사용을 최적화하고 비용을 절감하기 위한 주요 전략일 수 있습니다. SQL Server의 Enterprise 에디션은 Standard 에디션보다 비용이 많이 들기 때문에 다운그레이드를 통해 비용을 크게 절감할 수 있습니다.

투명한 데이터 암호화(TDE) 및 Always On 가용성 그룹은 SQL Server Enterprise 에디션에서 널리 사용되는 두 가지 기능입니다. 그러나 SQL Server Enterprise 에디션의 전체 기능 세트가 필요하지 않은 경우 이러한 기능에 대해 고려할 수 있는 비용 효율적인 대안이 있습니다. 예를 들어 SQL Server 2019부터 SQL Server Standard 에디션에서 TDE를 가져올 수 있습니다. Always On 가용성 그룹 대신 SQL Server Standard 에디션의 고가용성을 위해 FSx for Windows File Server의 공유 스토리지와 함께 장애 조치 클러스터링을 사용할 수 있습니다.

SQL Server Enterprise Edition에서 SQL Server Standard Edition으로 다운그레이드하면 라이선스 비용을 크게 줄일 수 있습니다. 자세한 내용은 AWS 스토리지 블로그의 게시물에서 고가용성 SQL Server 배포 비용 최적화 AWS를 참조하세요.

라이선스 비용을 줄이는 것 외에도 SQL Server 에디션을 다운그레이드하면 Software Assurance 지출을 줄이고 향후 트루업을 방지하는 데 도움이 될 수 있습니다. 사용하지 않은 라이선스를 선반에 반환하면 추가 라이선스 비용을 방지하고 라이선스 투자에서 가능한 최상의 가치를 얻을 수 있습니다.

SQL Server 워크로드를 신중하게 평가하고 비즈니스 요구 사항에 중요한 기능을 결정하는 것이 중요합니다. 자세한 내용은 AWS 권장 가이드의 환경 평가를 참조하고 Microsoft SQL Server 데이터베이스가 SQL Server Enterprise 에디션별 기능을 사용하는지 여부를 확인하세요.

올바른 버전의 SQL Server를 선택하고 SQL Server Enterprise Edition 기능에 대한 대안을 사용하는 경우 규정 준수를 유지하고 비즈니스 요구 사항을 충족하면서 상당한 비용을 절감할 수 있습니다. 다운그레이드 옵션에 대한 자세한 내용은이 가이드의 SQL Server 에디션 비교 섹션을 참조하세요.

비프로덕션 환경에서 SQL Server 개발자 에디션 사용

비프로덕션 환경에서는 온프레미스 환경에서 MSDN 구독을 사용하여 Enterprise 또는 Standard 에디션과 같은 라이선스 가능한 SQL Server 에디션을 배포할 수 있습니다. 그러나 MSDN 구독에는 라이선스 이동이 없습니다. 따라서 로 마이그레이션하면 해당 라이선스를 가져올 AWS수 없습니다. 대신 SQL Server 개발자 에디션을 사용해야 합니다.

SQL Server 개발자 에디션은 무료로 사용할 수 있는 SQL Server의 전체 기능 에디션입니다. 이 에디션은 SQL Server 버전 2016 이상에서 사용할 수 있습니다. Microsoft 웹 사이트에서 다운로드할 수 있습니다. SQL Server 개발자 에디션은 라이브 프로덕션 데이터에 연결되지 않는 한 개발, 테스트 및 스테이징과 같은 모든 비프로덕션 환경에서 사용하기 위한 것입니다.

비프로덕션 환경에서 SQL Server 개발자 에디션을 사용하는 경우 추가 라이선스 비용을 피할 수 있습니다. 자세한 내용은이 가이드의 SQL Server 개발자 에디션 평가 섹션을 참조하세요.

SQL Server 워크로드를 위한 CPU 최적화

경우에 따라 RAM 또는 네트워킹 제한과 같은 다른 요인으로 인해 워크로드에 필요한 것보다 많은 CPUs가 있는 인스턴스 유형을 선택해야 할 수 있습니다. 그러나는 이러한 상황에서 라이선스 비용을 최적화하는 데 도움이 되는 솔루션을 AWS 제공합니다.

SQL Server 코어 라이선스를 사용하는 대부분의 고객과 마찬가지로 EC2 인스턴스에서 하이퍼스레딩을 비활성화하거나 CPUs를 꺼 호스트로 사용 가능한 CPUs 수를 제한할 수 있습니다. 이 옵션을 사용하면 RAM과 같은 다른 인스턴스 기능을 활용하면서도 추가 라이선스 구매 비용을 절감할 수 있습니다.

예를 들어 워크로드에 128GB의 메모리가 필요하지만 SQL Server 코어가 8개만 필요하기 때문에 r5.4xlarge 인스턴스를 배포하는 경우 활성 CPUs. 이렇게 하면 적극적으로 사용 중인 8개의 코어에 대해서만 라이선스를 부여하면 되므로 필요한 SQL Server 라이선스에 대해 50%를 절약할 수 있습니다.

인스턴스 유형	총 vCPUs	CPUs 최적화 기능이 있는 활성 vCPU	SQL Server 라이선스 절감액
r5.4xlarge	16	8	50%
r5.12xlarge	48	8	83%

인스턴스 크기를 적절하게 조정하는 경우 워크로드에 가장 비용 효율적인 인스턴스 유형을 사용하고 있는지 확인할 수 있습니다. 는 새 인스턴스 유형을 AWS 도입하므로 이러한 새 인스턴스가 더 적은 코어로 워크로드 요구 사항을 충족할 수 있는지 평가하는 것이 중요합니다.

추가 리소스

- Amazon Web Services 및 Microsoft: 자주 묻는 질문(AWS 문서)

SQL Server 워크로드에 적합한 EC2 인스턴스 선택

> **Important**
>
> 이 섹션을 읽기 전에 먼저이 가이드의 SQL Server 라이선스 이해 및 Windows 워크로드에 적합한 인스턴스 유형 선택 섹션을 읽는 것이 좋습니다.

개요

Microsoft SQL Server는 15년 이상 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 실행되어 왔습니다. AWS 는 이러한 경험을 바탕으로 최소 사양에서 고성능 다중 리전 클러스터까지 실행되는 SQL Server 워크로드에 맞게 Amazon EC2 인스턴스를 개발하는 데 이를 사용했습니다.

SQL Server에 대해 올바른 EC2 인스턴스를 선택하는 것은 워크로드에 따라 크게 달라집니다. SQL Server의 라이선스 방식, 메모리 사용 방식, SQL Server 기능이 Amazon EC2 제품과 일치하는 방식을 이해하면 애플리케이션에 가장 적합한 EC2 인스턴스로 안내할 수 있습니다.

이 섹션에서는 다양한 SQL Server 워크로드와 이를 특정 EC2 인스턴스와 페어링하여 라이선스 및 컴퓨팅 비용을 최소화하는 방법을 설명합니다.

비용 비교

Amazon EC2를 사용하면 Windows Server 및 SQL Server 라이선스에 따라 기존 보유 라이선스 사용(BYOL) 또는 요금을 지불할 수 있습니다. pay-as-you-go 라이선스의 경우 Windows Server 및 SQL Server 라이선스에 대한 라이선스 비용은 EC2 인스턴스의 시간당 비용으로 베이크됩니다. 예를 들어 가격이 다른 AMIs 가질 수 있습니다. AMI의 가격은 AMI가 실행되는 SQL Server 에디션에 따라 달라집니다.

Windows Server 및 SQL Server 요금은 항목별로 구분되지 않습니다. 와 같은 도구에서는 항목별 요금을 찾을 수 없습니다AWS Pricing Calculator. 라이선스 포함 상품의 다양한 조합을 선택하면 다음 표와 같이 라이선스 비용을 추론할 수 있습니다.

EC2 인스턴스	AMI	컴퓨팅 가격	Windows 라이선스 가격	SQL 라이선스 가격	총 가격
r5.xlarge	Linux(컴퓨팅 요금)	183.96 USD	-	-	183.96 USD

EC2 인스턴스	AMI	컴퓨팅 가격	Windows 라이선스 가격	SQL 라이선스 가격	총 가격
r5.xlarge	Linux + SQL 개발자	183.96 USD	$0	$0	183.96 USD
r5.xlarge	Windows Server(LI)	183.96 USD	134.32 USD	-	318.28 USD
r5.xlarge	Windows + SQL 개발자	183.96 USD	134.32 USD	$0	318.28 USD
r5.xlarge	Windows + SQL Web(LI)	183.96 USD	134.32 USD	49.64 USD	367.92 USD
r5.xlarge	Windows + SQL Standard(LI)	183.96 USD	134.32 USD	350.4 USD	668.68 USD
r5.xlarge	Windows + SQL Enterprise(LI)	183.96 USD	134.32 USD	1,095 USD	$1413.28

> **Note**
>
> 위 표의 요금은 `us-east-1` 리전의 온디맨드 요금을 기준으로 합니다.

SQL Server를 실행하는 가장 비용 효율적인 방법은 상위 수준 에디션의 기능이 필요할 때까지 하위 수준 에디션을 유지하는 것입니다. 자세한 내용은이 가이드의 SQL Server 버전 비교 섹션을 참조하세요. SQL Server Web Edition에서 SQL Server Standard Edition으로 업그레이드하는 것은 SQL Server 라이선스 비용의 7배 이상이며 Standard Edition에서 Enterprise Edition으로 전환하는 비용의 3배 이상입니다. 라이선스 비용의 차이는 고려해야 할 주요 요소이며이 섹션의 나머지 부분에서 살펴봅니다.

비용 최적화 시나리오

배송 차량을 추적하는 분석 회사가 SQL Server 성능을 개선하고자 하는 예제 시나리오를 생각해 보세요. MACO 전문가가 회사의 성능 병목 현상을 검토한 후 회사는 x1e.2xlarge 인스턴스에서

x2iedn.xlarge 인스턴스로 전환합니다. 인스턴스 크기는 더 작지만 x2 인스턴스의 개선 사항은 버퍼 풀 확장을 사용하여 SQL Server 성능과 최적화를 개선합니다. 이를 통해 회사는 SQL Server Enterprise Edition에서 SQL Server Standard Edition으로 다운그레이드하고 SQL Server 라이선스를 vCPUsvCPUs.

최적화 전:

Server	EC2 인스턴스	SQL Server 에디션	월별 비용
ProdDB1	x1e.2xlarge	엔터프라이즈	3,918.64 USD
ProdDB2	x1e.2xlarge	엔터프라이즈	3,918.64 USD
합계			7,837.28 USD

최적화 후:

Server	EC2 인스턴스	SQL Server 에디션	월별 비용
ProdDB1	x2iedn.xlarge	표준	1,215.00 USD
ProdDB2	x2iedn.xlarge	표준	1,215.00 USD
합계			2,430.00 USD

x1e.2xlarge 인스턴스에서 x2iedn.xlarge 인스턴스로의 통합 변경으로 인해 예제 고객은 프로덕션 데이터베이스 서버에 매월 5,407 USD를 절약할 수 있었습니다. 이렇게 하면 워크로드의 총 비용이 69% 절감되었습니다.

> **Note**
>
> 위 표의 요금은 `us-east-1` 리전의 온디맨드 요금을 기준으로 합니다.

비용 최적화 권장 사항

메모리 최적화 인스턴스

SQL Server의 가장 중요한 측면 중 하나는 메모리에 대한 의존도를 이해하는 것입니다. SQL Server는 운영 체제에서 사용하지 않는 사용 가능한 모든 RAM을 사용하려고 시도합니다(기본 설치의 경우 최대 2TB). 성능상의 이유로이 작업을 수행합니다. 메모리의 데이터 작업은 디스크에서 데이터를 지속적으로 가져와서 변경한 다음 디스크에 다시 쓸 때보다 훨씬 더 효과적입니다. 대신 SQL Server는 연결된 데이터베이스에서 최대한 많은 데이터를 로드하려고 시도하고 해당 데이터를 RAM에 유지합니다. 데이터에 대한 변경 사항은 메모리에서 발생하며 나중에 디스크로 강화됩니다.

> **Note**
>
> SQL Server가 변경 사항을 작성하는 방법에 대한 자세한 설명은 Microsoft 설명서의 페이지 작성을 참조하세요.

SQL Server는 더 많은 양의 RAM에서 더 나은 성능을 제공하므로 일반적으로 Amazon EC2 메모리 최적화 인스턴스 유형으로 시작하는 것이 좋습니다. 메모리 최적화 인스턴스는 다목적이며 다양한 옵션을 제공합니다. R 패밀리는 vCPU-to-RAM 비율이 1:8이며 Intel 프로세서, AMD 프로세서, 향상된 네트워킹, 향상된 EBS 성능, 인스턴스 스토리지 및 향상된 프로세서 속도에 대한 옵션이 있습니다. 메모리 사용량이 많은 워크로드의 경우 많은 동일한 옵션을 결합하고 vCPU-to-RAM 비율을 1에서 32로 확장하는 X 패밀리도 있습니다. 메모리 최적화 인스턴스의 다목적성으로 인해 모든 모양과 크기의 SQL Server 워크로드에 적용할 수 있습니다.

최소 리소스 미만의 워크로드(4vCPUs 미만)

일부 사용 사례는 버스트 가능(T3) 인스턴스에서 잘 작동하지만 일반적으로 SQL Server 워크로드에 버스트 가능 인스턴스를 사용하지 않는 것이 좋습니다. SQL Server에 대한 라이선스는 인스턴스에 할당된 vCPUs 수를 기반으로 합니다. SQL Server가 대부분의 하루 동안 유휴 상태이고 버스트 크레딧을 획득하는 경우 완전히 활용하지 않는 SQL 라이선스에 대해 비용을 지불합니다. 또한 SQL Server의 최소 라이선스 요구 사항은 서버당 코어 4개입니다. 즉, 4개의 vCPUs에 상당하는 컴퓨팅 성능이 필요하지 않은 SQL Server 워크로드가 있는 경우 사용하지 않는 SQL Server 라이선스를 지불하게 됩니다. 이러한 시나리오에서는 여러 SQL Server 인스턴스를 더 큰 서버로 통합하는 것이 가장 좋습니다.

최소 리소스를 사용하는 워크로드(64GB RAM 미만)

64GB RAM 미만의 많은 SQL Server 워크로드는 고성능 또는 고가용성의 우선 순위를 지정하지 않습니다. 이러한 유형의 워크로드의 경우 애플리케이션이 Microsoft의 라이선스 제한의 적용을 받는 경우 SQL Server 웹 에디션이 적합할 수 있습니다.

> **Important**
>
> SQL Server 웹 에디션은 Microsoft의 라이선스 조건에 따라 사용 사례가 제한됩니다. SQL Server Web 에디션은 퍼블릭 및 인터넷으로 액세스 가능한 웹 페이지, 웹 사이트, 웹 애플리케이션 및 웹 서비스를 지원하는 데에만 사용할 수 있습니다. LOB(Line of Business) 애플리케이션(예: 고객 관계 관리, 전사적 자원 관리 및 기타 유사한 애플리케이션)을 지원하는 데는 사용할 수 없습니다.

SQL Server 웹 에디션은 최대 32vCPUs와 64GB RAM까지 확장되며 SQL Server Standard 에디션보다 86% 저렴합니다. 리소스가 적은 워크로드의 경우 Intel에 비해 컴퓨팅 가격이 10% 저렴한 r6a와 같은 AMD 메모리 최적화 인스턴스를 사용하는 것도 컴퓨팅 및 SQL 라이선스 비용을 최소화하는 좋은 방법입니다.

평균 리소스가 있는 워크로드(128GB RAM 미만)

SQL Server Standard 에디션은 최대 128GB RAM의 대부분의 SQL Server 워크로드에 사용됩니다. SQL Server Standard 에디션은 SQL Server Enterprise 에디션보다 65~75% 저렴하며 최대 48vCPUs와 128GB RAM까지 확장할 수 있습니다. 128GB RAM 제한은 일반적으로 48개의 vCPU 제한 이전에 적용되므로 SQL Server Enterprise Edition으로 업그레이드하지 않으려는 대부분의 고객은이 제한에 중점을 둡니다.

SQL Server에는 버퍼 풀 확장이라는 기능이 있습니다. 이 기능을 사용하면 SQL Server가 디스크의 일부를 사용하여 RAM의 확장 역할을 할 수 있습니다. 버퍼 풀 확장은 Amazon EC2 인스턴스 스토리지에 사용되는 NVMe SSDs와 같이 초고속 스토리지와 결합할 때 잘 작동합니다. Amazon EC2 인스턴스 스토리지가 포함된 Amazon EC2 인스턴스는 인스턴스 이름에 "d"로 표시됩니다(예: r5d, r6id, x2iedn).

버퍼 풀 확장은 일반 RAM을 대체하지 않습니다. 그러나 128GB 이상의 RAM이 필요한 경우 r6id.4xlarge 및 x2iedn.xlarge와 같은 EC2 인스턴스에서 버퍼 풀 확장을 사용하여 Enterprise Edition 라이선스 업그레이드를 지연할 수 있습니다.

고성능 워크로드(128GB RAM 이상)

고성능이 필요한 SQL Server 워크로드는 많은 리소스에 의존하기 때문에 비용 최적화가 어렵습니다. 그러나 EC2 인스턴스의 차이점을 이해하면 잘못된 선택을 할 수 없습니다.

다음 표에는 다양한 메모리 최적화 EC2 인스턴스와 해당 성능 제한이 나와 있습니다.

	r5b	r6idn	r7iz	x2iedn	x2iezn
처리자	3.1GHz 2세대 인텔 제온 프로세서	3.5GHz 3세대 인텔 제온 프로세서	3.9GHz 4세대 인텔 제온 스케일러블 프로세서	3.5GHz 3세대 인텔 제온 프로세서	4.5GHz 2세대 인텔 제온 프로세서
CPU:RAM 비율	1:8	1:8	1:8	1:32	1:32
최대 vCPU	96	128	128	128	48
최대 RAM	768GB	1,024GB	1,024GB	4,096GB	1,536GB
인스턴스 스토리지	–	NVMe SSD (4x 1900GB)	–	NVMe SSD (2x 1900GB)	–
io2 Block Express	지원	지원	지원	지원	–
최대 EBS IOPS	260,000	350,000	160,000	260,000	80,000
최대 EBS 처리량	60Gbps	80Gbps	40Gbps	80Gbps	19Gbps
최대 네트워크 대역폭	25Gbps	200Gbps	50Gbps	100Gbps	100Gbps

각 인스턴스는 다른 용도로 사용됩니다. SQL Server 워크로드를 이해하면 가장 적합한 인스턴스 유형을 선택하는 데 도움이 될 수 있습니다.

속성에 대한 세부 정보:

- r5b - r5b의 "b" 속성은이 인스턴스 유형이 높은 EBS 성능에 초점을 맞추고 있음을 의미합니다. 5세대 메모리 최적화 인스턴스에서는 r5b가 선호되었습니다. io2 Block Express 볼륨을 활용하고 최대 스토리지 IOPS 260,000에 도달한 최초의 인스턴스 유형이었습니다. r5b 인스턴스 유형은 여전히 높은 EBS 성능 요구 사항에 대한 비용 효율적인 대안입니다.
- r6idn - 6세대 메모리 최적화 인스턴스는 이전 세대에 비해 상당히 개선되었습니다. r5b의 EBS 성능 향상은 r6idn으로 한 단계 더 나아가 최대 IOPS를 350,000으로 높입니다. 또한 r6idn에는 SQL Server 성능을 더욱 높이기 위해 tempdb 및 버퍼 풀 확장을 위한 인스턴스 스토어 볼륨이 있습니다.
- x2iedn - x2iedn은 r6idn과 유사합니다. 비슷한 수준의 향상된 EBS, 향상된 네트워킹 및 NVMe SSD 인스턴스 스토리지를 제공하지만 높은 메모리 워크로드와 낮은 CPU 수량(더 낮은 SQL Server 라이선스 비용)에 대해 1:32 vCPU-to-RAM 비율을 제공합니다.
- x2iezn - x2iezn의 "z" 속성은이 인스턴스 유형이 높은 프로세서 성능에 초점을 맞추고 있음을 나타냅니다. Cascade Lake 프로세서의 전체 코어 터보 주파수는 최대 4.5GHz입니다. vCPU-to-RAM 비율과 함께이 EC2 인스턴스를 사용하는 것이 좋습니다. 따라서 SQL Server 라이선스 비용을 낮게 유지할 수 있습니다.
- r7iz - r7iz의 "z" 속성은이 인스턴스 유형이 높은 프로세서 성능에 초점을 맞추고 있음을 나타냅니다. Sapphire Rapids 프로세서의 전체 코어 터보 주파수는 최대 3.9GHz입니다. x2iezn 인스턴스와 마찬가지로 r7iz는 고주파 프로세서 성능을 우선시하지만 vCPU-to-RAM 비율은 1:8입니다.

추가 리소스

- 범용 Amazon EC2 인스턴스(AWS 문서)
- 비교 도구(Vantage)
- 라이선스 - SQL Server(AWS 문서)

인스턴스 통합

이 섹션에서는 여러 SQL Server 인스턴스를 동일한 서버에 결합하여 라이선스 비용을 최소화하고 리소스 사용률을 극대화하는 비용 최적화 기술에 중점을 둡니다.

개요

인스턴스 생성은 SQL Server 데이터베이스 엔진 설치 프로세스의 일부입니다. SQL Server 인스턴스는 자체 서버 파일, 보안 로그인 및 시스템 데이터베이스(마스터, 모델, msdb 및 tempdb)를 포함하는 완전한 설치입니다. 인스턴스에는 자체 파일과 서비스가 모두 있으므로 인스턴스가 서로 간섭하지 않

고 동일한 운영 체제에 여러 SQL Server 인스턴스를 설치할 수 있습니다. 그러나 인스턴스는 모두 동일한 서버에 설치되므로 모두 컴퓨팅, 메모리 및 네트워킹과 같은 동일한 하드웨어 리소스를 공유합니다.

"사용 중" 인스턴스가 공유 하드웨어 리소스를 과도하게 사용하지 않도록 프로덕션 환경에서 서버당 단일 SQL Server 인스턴스만 사용하는 것이 일반적입니다. 각 SQL Server 인스턴스에 자체 리소스를 갖춘 자체 운영 체제를 제공하는 것이 리소스 거버넌스에 의존하는 것보다 더 나은 경계입니다. 이는 대용량 RAM 및 CPU 리소스가 필요한 고성능 SQL Server 워크로드의 경우 특히 그렇습니다.

그러나 모든 SQL Server 워크로드가 많은 양의 리소스를 사용하는 것은 아닙니다. 예를 들어 일부 조직에서는 규정 준수 또는 보안 목적으로 각 고객에게 전용 SQL Server 인스턴스를 할당합니다. 일반적으로 활성화되지 않는 소규모 클라이언트 또는 클라이언트의 경우 최소한의 리소스로 SQL Server 인스턴스를 실행해야 합니다.

Microsoft SQL Server 2019: 라이선싱 가이드에 명시된 대로 SQL Server를 실행하는 각 서버는 최소 4개의 CPU 라이선스를 고려해야 합니다. 즉, vCPUs가 2개뿐인 서버를 실행하더라도 4개의 vCPUs에 대해 SQL Server 라이선스를 부여해야 합니다. SQL Server Standard 에디션을 사용하는 경우 Microsoft의 퍼블릭 SQL Server 요금에 따라 3,945 USD가 다릅니다. 최소한의 리소스를 사용하여 단일 SQL Server 인스턴스로 여러 서버를 실행하는 조직의 경우 사용하지 않는 리소스에 라이선스를 부여하는 데 드는 총 비용이 상당할 수 있습니다.

비용 최적화 시나리오

이 섹션에서는 각각 단일 SQL Server 인스턴스가 있는 4개의 Windows Server 서버를 실행하는 것과 여러 SQL Server 인스턴스를 동시에 실행하는 하나의 대형 Windows Server 서버의 차이를 비교하는 예제 시나리오를 살펴봅니다.

각 SQL Server 인스턴스에 vCPUs2개와 8GB RAM만 필요한 경우 시간당 컴퓨팅 비용인 0.096 USD 외에 SQL Server 라이선스에 대한 서버당 총 비용은 7,890 USD입니다.

EC2 인스턴스	vCPU	RAM	가격	라이선스를 부여할 vCPUs	총 SQL Server 라이선스 비용
m6i.large	2	8	0.096	4	7,890 USD

이를 서버 4개로 확장하면 시간당 컴퓨팅 비용이 0.384 USD인 SQL Server 라이선스의 총 비용은 31,560 USD입니다.

EC2 인스턴스	vCPU	RAM	가격	라이선스를 부여할 vCPUs	총 SQL Server 라이선스 비용
m6i.large 4개	2	32	0.384	16	31,560 USD

4개의 SQL Server 인스턴스를 모두 단일 EC2 인스턴스에 결합하면 컴퓨팅 리소스와 컴퓨팅의 총량은 동일하게 유지됩니다. 그러나 불필요한 SQL Server 라이선스 비용을 제거하면 워크로드를 실행하는 데 드는 총 비용을 15,780 USD 줄일 수 있습니다.

EC2 인스턴스	vCPU	RAM	가격	라이선스를 부여할 vCPUs	총 SQL Server 라이선스 비용
m6i.2xlarge	8	32	0.384	8	15,780 USD

> **Note**
>
> 앞의 표에서 컴퓨팅 비용은 `us-east-1` 리전에서 Windows Server를 실행하는 Amazon EC2 서버의 시간당 온디맨드 요금을 보여줍니다. SQL Server Standard Edition 라이선스 비용은 Microsoft의 퍼블릭 SQL Server 요금을 참조합니다.

비용 최적화 권장 사항

SQL Server 인스턴스 통합을 고려하는 경우 가장 큰 문제는 통합하려는 각 인스턴스의 리소스 사용량입니다. 각 서버의 워크로드 패턴을 더 잘 이해하려면 장기간에 걸쳐 성능 지표를 가져오는 것이 중요합니다. 리소스 소비 모니터링을 위한 몇 가지 일반적인 도구는 Amazon CloudWatch, Windows 성능 모니터(퍼프몬) 및 SQL Server의 기본 모니터링 도구입니다.

SQL Server 워크로드가 서로 간섭하지 않고 동일한 서버 리소스를 사용하도록 결합할 수 있는지 여부를 분석할 때는 다음 질문을 고려하는 것이 좋습니다.

- 안정 상태에서 사용되는 리소스(CPU, 메모리 및 네트워크 대역폭)는 무엇입니까?
- 스파이크 중에 사용되는 리소스(CPU, 메모리 및 네트워크 대역폭)는 무엇입니까?

- 스파이크는 얼마나 자주 발생하나요? 스파이크가 일관되나요?
- 한 서버의 리소스 스파이크가 다른 서버의 리소스 스파이크와 일치하나요?
- SQL Server에서 사용하는 스토리지 IOPS 및 처리량은 얼마입니까?

SQL Server 인스턴스를 결합하려는 경우 클라우드 운영 및 마이그레이션 블로그의 Amazon EC2 인스턴스 하나에서 여러 SQL Server 인스턴스 실행 게시물을 AWS 참조하세요. 이 게시물에서는 SQL Server에서 구성을 변경하여 인스턴스를 추가하는 방법에 대한 지침을 제공합니다. 시작하기 전에 동일한 서버에 여러 인스턴스가 설치될 때의 사소한 차이점을 고려하세요.

- 기본 SQL Server 데이터베이스 인스턴스의 이름은 MSSQLSERVER 이며 포트 1433을 사용합니다.
- 동일한 서버에 설치된 각 추가 인스턴스는 "이름" 데이터베이스 인스턴스입니다.
- 이름이 지정된 각 인스턴스에는 고유한 인스턴스 이름과 고유한 포트가 있습니다.
- 명명된 인스턴스에 대한 트래픽을 조정하려면 SQL Server 브라우저를 실행해야 합니다.
- 각 인스턴스는 데이터베이스 데이터 파일 및 별도의 로그인에 별도의 위치를 사용할 수 있습니다.
- SQL Server 최대 서버 메모리 설정은 각 인스턴스의 성능 요구 사항에 따라 구성해야 하며, 합계는 기본 운영 체제에 충분한 메모리를 남겨둡니다.
- SQL Server 기본 백업 및 복원 기능을 사용하거나 마이그레이션 또는 통합AWS DMS에 사용할 수 있습니다.

추가 리소스

- SQL Server 라이선싱 데이터시트(AWS 클라우드 운영 및 마이그레이션 블로그)
- SQL Server 다중 인스턴스 설정 블로그 게시물(AWS 클라우드 운영 및 마이그레이션 블로그)
- SQL Server 모범 사례 가이드(AWS 권장 가이드 설명서)

SQL Server 에디션 비교

개요

Microsoft SQL Server 라이선스는 Windows 워크로드 환경에서 가장 큰 비용 중 하나입니다. SQL Server의 라이선스 비용은 워크로드를 실행하는 컴퓨팅 비용 이상으로 쉽게 확장할 수 있습니다. 잘못된 에디션을 선택하면 사용하지 않거나 필요하지 않은 기능에 대한 비용을 지불할 수 있습니다. 이 섹션에서는 기능 및 상대 비용을 포함하여 다음 SQL Server 에디션을 비교합니다.

- 엔터프라이즈 - SQL Server 엔터프라이즈 에디션은 고성능, 무제한 가상화 및 여러 비즈니스 인텔리전스(BI) 도구를 통해 데이터 센터 기능을 제공합니다.
- 표준 - SQL Server Standard 에디션은 소규모 조직 및 부서에 기본 데이터 관리 및 비즈니스 인텔리전스를 제공합니다.
- 웹 - SQL Server 웹 에디션은 웹 호스트 또는 웹 부가가치 제공업체(VAPs)인 기업에 적합합니다. 이 에디션은 낮은 총 소유 비용을 제공하며 소규모 및 대규모 웹 속성을 위한 확장성 및 관리 기능을 제공합니다.

> **Important**
>
> SQL Server 웹 에디션을 사용하여 퍼블릭 및 인터넷 액세스 가능 웹 페이지, 웹 사이트, 웹 애플리케이션 및 웹 서비스만 지원할 수 있습니다. SQL Server 웹 에디션을 사용하여 line-of-business 애플리케이션(예: 고객 관계 관리 또는 엔터프라이즈 리소스 관리 애플리케이션)을 지원할 수 없습니다.

- 개발자 - SQL Server 개발자 에디션에는 엔터프라이즈 에디션의 모든 기능이 포함되어 있지만 개발 목적으로만 사용됩니다.
- Express – SQL Server Express 에디션은 무료 데이터베이스이며 데스크톱 애플리케이션을 학습하거나 구축하는 데 사용할 수 있습니다. Express 에디션을 다른 에디션으로 업데이트할 수 있습니다.

> **Note**
>
> SQL Server 평가 에디션은 180일 평가판 기간 동안 사용할 수 있습니다.

비용 영향

Microsoft 리셀러로부터 SQL Server 라이선스를 구입하여 Software Assurance AWS 를 통해 로 가져올 수 있습니다. 또는 라이선스에 Amazon EC2 AMI가 포함된 pay-as-you-go 모델과 함께 SQL Server 라이선스를 사용할 수 있습니다. AMIs

Microsoft 리셀러로부터 SQL Server 라이선스를 구매하는 경우 코어 라이선스는 2개 팩으로 판매되며 서버당 최소 4개의 코어에 라이선스를 부여해야 합니다. 다음 표는 엔터프라이즈 에디션과 스탠다드 에디션 간의 비용 비교를 보여줍니다.

버전	SQL Server Enterprise Edition(코어 2개 팩)	SQL Server Standard Edition(코어 2개 팩)	절감
2022	15,123 USD	3,945 USD	74%
2019	13,748 USD	3,586 USD	74%

> **Note**
>
> 위 표의 요금은 SQL Server 2022 및 SQL Server 2019에 대한 Microsoft의 공개 요금을 기준으로 합니다.

다음 비용 비교는 라이선스 포함 Amazon EC2 AMIs를 사용하여 다양한 버전의 SQL Server를 호스팅하는 것을 보여줍니다. 이 비교에서 SQL Server는 `us-east-1` 리전의 r6i.xlarge(4 vCPU)에서 호스팅됩니다.

Instance	컴퓨팅 비용	Windows 라이선스 비용	SQL Server 라이선스 비용	합계
R6i.xlarge(Linux)	183.96 USD	–	–	183.96 USD
R6i.xlarge + Windows	183.96 USD	134.32 USD	–	$318.28
R6i.xlarge + SQL Server 웹 에디션	183.96 USD	134.32 USD	49.35 USD	367.63 USD
R6i.xlarge + SQL Server Standard 에디션	183.96 USD	134.32 USD	350.4 USD	668.68 USD
R6i.xlarge + SQL 엔터프라이즈 에디션	183.96 USD	134.32 USD	1,095 USD	1,413.28 USD

워크로드에 적합한 SQL Server 에디션을 선택하여 SQL Server 라이선스 비용을 최대 95% 절감할 수 있습니다. 다음 표에서는 r6i.xlarge 인스턴스의 SQL Server 라이선스 비용을 비교합니다.

Edition	% 절감
Enterprise와 비교한 표준	68%
웹과 표준 비교	86%
웹과 엔터프라이즈 비교	95%

대부분의 시나리오에서 조직은 Enterprise에서 Standard Edition으로 전환하지만 Standard 또는 Enterprise Edition에서 Web Edition으로 전환할 수 있는 경우가 있습니다.

비용 최적화 권장 사항

규모 조정 제한, 고가용성, 성능 및 보안을 기반으로 워크로드에 가장 적합한 에디션을 선택할 수 있습니다. 다음 표에는 SQL Server 에디션에서 지원되는 기능이 나와 있습니다. 이렇게 하면 사용할 에디션을 결정하는 데 도움이 될 수 있습니다. 이 비교는 SQL Server 2016 SP1 이상 버전에 적용됩니다.

규모 조정 제한

다음 표에서는 다양한 SQL Server 에디션의 조정 제한을 비교합니다.

Feature	엔터프라이즈 에디션	스탠다드 에디션	웹 에디션	Express 에디션
SQL Server Database Engine, SQL Server Analysis Services(SSAS) 또는 SQL Server Reporting Services(SSRS)의 단일 인스턴스에서 사용하는 최대 컴퓨팅 용량	운영 체제 최대값	소켓 4개 또는 코어 24개 중 더 작은 값으로 제한	소켓 4개 또는 코어 16개 중 작은 값으로 제한	소켓 4개 또는 코어 4개 중 더 작은 값으로 제한

Feature	엔터프라이즈 에디션	스탠다드 에디션	웹 에디션	Express 에디션
SQL Server Database Engine 인스턴스당 버퍼 풀의 최대 메모리	운영 체제 최대값	128GB	64GB	1410MB
SQL Server Database Engine 인스턴스당 버퍼 풀 확장의 최대 용량	최대 메모리 구성의 32배	구성된 최대 메모리의 4배	N/A	N/A
최대 관계형 데이터베이스 크기	524PB	524PB	524PB	10GB
Columnstore 캐시 또는 메모리 최적화 데이터의 최대 메모리	운영 체제 최대값	32GB	16 GB	352MB

애플리케이션에 16개 미만의 코어(vCPUs)와 64GB의 RAM이 필요한 경우 SQL Server 웹 에디션에서 평가를 시작할 수 있습니다. 워크로드에 64GB 이상의 메모리 또는 기타 고가용성 옵션이 필요한 경우 SQL Server Standard 에디션으로 업그레이드해야 합니다.

SQL Server 웹 에디션을 사용하여 퍼블릭 및 인터넷 액세스 웹 페이지, 웹 사이트, 웹 애플리케이션 및 웹 서비스를 지원할 수 있지만 SQL Server 웹 에디션을 사용하여 비즈니스 애플리케이션을 지원할 수는 없습니다. SQL Server 웹 에디션의 사용 사례에 대한 자세한 내용은 Microsoft 라이선싱 지원 또는 Microsoft 리셀러에게 문의하십시오.

SQL Server Standard 에디션은 최대 24개의 코어(48개의 vCPUs) 및 128GB 메모리 워크로드에 사용할 수 있습니다. 그러나 버퍼 풀 확장을 사용하여 SQL Server Standard 에디션이 r6id EC2 인스턴스에 있는 것과 같은 로컬 인스턴스 스토리지를 활용할 수 있도록 할 수 있습니다. 이렇게 하면 메모리가 최대 메모리 구성의 4배 크기까지 확장됩니다. 이러한 기능 조합은 메모리 요구 사항이 증가하기 시작할 때 서버가 Enterprise Edition으로 업그레이드하는 것을 지연시킬 수 있습니다.

버퍼 풀 및 페이지 수명 예상 카운터에서 데이터베이스 페이지를 찾아 메모리 사용률을 식별할 수 있습니다. 페이지 수명은 디스크로 다시 플러시되기 전에 페이지가 메모리에 있는 시간을 알려줍니다. 이 카운터 기본값은 300입니다. 페이지가 몇 시간 또는 며칠 동안 메모리에 있는 경우 할당된 메모리를 줄일 가능성이 있습니다.

높은 가용성

다음 표에서는 다양한 SQL Server 에디션의 고가용성 기능을 비교합니다.

Feature	엔터프라이즈 에디션	스탠다드 에디션	웹 에디션	Express 에디션
서버 코어 지원 1	예	예	예	예
로그 전달	예	예	예	아니요
데이터베이스 미러링	예	전체 안전 모드	감시인으로만	감시인으로만
백업 압축	예	예	아니요	아니요
Always On 장애 조치 클러스터 인스턴스	노드 16개	노드 2개	아니요	아니요
Always On 가용성 그룹	동기식 보조 복제본 2개를 포함하여 최대 8개의 보조 복제본	아니요	아니요	아니요
기본 가용성 그룹	아니요	노드 2개	아니요	아니요
온라인 페이지 및 파일 복원	예	아니요	아니요	아니요
온라인 인덱싱	예	아니요	아니요	아니요
온라인 스키마 변경	예	아니요	아니요	아니요

Feature	엔터프라이즈 에디션	스탠다드 에디션	웹 에디션	Express 에디션
빠른 복구	예	아니요	아니요	아니요
미러링된 백업	예	아니요	아니요	아니요
핫 추가 메모리 및 CPU	예	아니요	아니요	아니요
암호화된 백업	예	예	아니요	아니요
Microsoft Azure로 하이브리드 백업(URL로 백업)	예	예	아니요	아니요
재해 복구를 위한 장애 조치 서버	예	예	아니요	아니요
고가용성을 위한 장애 조치 서버	예	예	아니요	아니요

기타 일반적인 기능

다음 표에서는 다양한 SQL Server 에디션의 가장 일반적인 기능을 비교합니다. 광범위한 기능 목록은 Microsoft 설명서의 SQL Server 2019 버전 및 지원되는 기능을 참조하세요.

Feature	엔터프라이즈 에디션	스탠다드 에디션	웹 에디션	Express 에디션
(성능) 리소스 어드벤처	예	아니요	아니요	아니요
(보안) 투명한 데이터베이스 암호화(TDE)	예	예	아니요	아니요

Feature	엔터프라이즈 에디션	스탠다드 에디션	웹 에디션	Express 에디션
(보안) 확장 가능한 키 관리(EKM)	예	아니요	아니요	아니요
(복제) Oracle 게시	예	아니요	아니요	아니요
(복제) 피어 투 피어 트랜잭션 복제	예	아니요	아니요	아니요
변경 데이터 캡처	예	예	아니요	아니요

SQL Server 개발자 에디션

개발, QA, 테스트, 스테이징 및 UAT 환경과 같은 모든 비프로덕션 워크로드는 SQL Server 개발자 에디션을 사용하여 SQL Server 라이선스 비용을 100% 절감할 수 있습니다. SQL Server를 다운로드한 후 공유 테넌시를 사용하여 EC2 인스턴스에 SQL Server 개발자 에디션을 설치할 수 있습니다. SQL Server 개발자 에디션에는 전용 인프라가 필요하지 않습니다. 자세한 내용은이 설명서의 SQL Server 개발자 버전 권장 사항을 참조하세요.

에디션 전환

기존 워크로드의 경우 한 에디션에서 다른 에디션으로 전환하려면 광범위한 테스트가 필요합니다. Enterprise 또는 Standard 에디션에서 실행되는 워크로드를 확인하여 에디션별 기능이 사용되는지 여부와 해당 기능에 대한 대체 솔루션이 있는지 확인하는 것이 가장 좋습니다. 예를 들어 데이터베이스가 엔터프라이즈 수준 기능을 사용하고 있는지 확인하려면 다음 예제 명령과 같이 모든 데이터베이스에서 동적 관리 뷰(DMV)를 실행할 수 있습니다.

```
SELECT feature_name FROM sys.dm_db_persisted_sku_features; GO
```

SQL 유지 관리 작업의 일부로 온라인 재인덱싱과 같이 T-SQL에서 캡처할 수 없는 일부 Enterprise Edition 기능이 있습니다. 수동으로 확인해야 합니다.

마이그레이션 고려 사항

SQL Server에 라이선스를 부여하는 방법에 따라 에디션 전환 옵션이 결정됩니다. SQL Server AMIs를 포함한 AMIs에는 EC2 인스턴스 가격에 라이선스 비용이 포함되며, 라이선스 비용은 AMI에 적용됩니

다. AWS 결제 코드를 사용하여 AMI에 포함된 SQL Server 버전을 확인할 수 있습니다. AWS 라이선스 포함 인스턴스의 경우 운영 체제 내에서 SQL Server 에디션을 변경해도 AMI와 연결된 청구는 변경되지 않습니다. SQL Server의 새 버전을 실행하는 AMI를 사용하여 데이터베이스를 새 EC2 인스턴스로 마이그레이션해야 합니다.

자체 라이선스를 가져오면 더 유연하게 사용할 수 있습니다. 일반적으로 새 버전을 실행하는 다른 EC2 인스턴스로 마이그레이션하는 것이 좋습니다. 이렇게 하면 무언가 계획대로 진행되지 않는 경우 쉽게 페일백할 수 있습니다. 그러나 기존 서버를 사용해야 하는 경우에도 SQL Server를 side-by-side 설치하고 인스턴스 간에 데이터베이스를 마이그레이션할 수 있습니다. 버전 다운그레이드side-by-side 대한 자세한 단계는 MSSQLTips 웹 사이트의 SQL Server에서 버전 업그레이드 및 다운그레이드를 참조하세요.

추가 리소스

- SQL Server 2022(Microsoft Learn)의 버전 및 지원되는 기능
- sys.dm_db_persisted_sku_features(Transact-SQL)(Microsoft Learn)
- 어떤 버전의 SQL Server를 사용해야 합니까? (브렌트 오자르 무제한)
- AWS Pricing Calculator (AWS)

SQL Server 개발자 에디션 평가

개요

SQL Server 개발자 에디션은 Enterprise 에디션의 모든 기능을 포함하고 비프로덕션 환경에서 사용할 수 있는 SQL Server의 무료 에디션입니다. Microsoft Developer Network(MSDN) 라이선스를 사용할 수 없는 클라우드에서 SQL Server 개발자 에디션은 개발 및 테스트 워크로드에 라이선스를 제공할 필요 없이 비용을 절감할 수 있는 좋은 방법입니다. 이는 대규모 개발 및 테스트 환경을 운영하고 불필요한 비용을 줄이려는 팀에 특히 적용됩니다.

프로덕션 환경은 애플리케이션의 최종 사용자(예: 인터넷 웹 사이트)가 액세스하는 환경으로 정의되며 해당 애플리케이션의 피드백 수집 또는 수락 테스트 이상의 용도로 사용됩니다. 프로덕션 환경을 구성하는 기타 시나리오는 다음과 같습니다.

- 프로덕션 데이터베이스에 연결하는 환경
- 프로덕션 환경에 대한 재해 복구 또는 백업을 지원하는 환경
- 피크 활동 기간 동안 프로덕션으로 교체되는 서버와 같이 적어도 일정 기간 동안 프로덕션에 사용되는 환경

라이선스에 대한 자세한 내용은 설명서의 Amazon Web Services 및 Microsoft: 자주 묻는 질문을 참조하세요 AWS .

비용 영향

비프로덕션 워크로드에 SQL Server 개발자 에디션을 사용하는 경우 개발 및 테스트 환경에 대한 현재 SQL Server 라이선스 비용의 100%를 절감할 수 있습니다.

SQL Server 버전	SQL Server Enterprise Edition(코어 2개 팩)	SQL Server Standard Edition(코어 2개 팩)	SQL Server 개발자 에디션
2022	15,123 USD	3,945 USD	무료
2019	13,748 USD	3,586 USD	무료

> **Note**
>
> 위 표의 요금은 SQL Server 2022 및 SQL Server 2019에 대한 Microsoft의 공개 요금을 기준으로 합니다.

다음 표에서는 4개의 vCPUs되고 `us-east-2` 리전에서 온디맨드 요금을 사용하는 다양한 SQL Server 에디션의 비용을 비교합니다. 이는의 라이선스 포함 인스턴스에 의존하는 시나리오에 적용됩니다 AWS.

EC2 인스턴스	AMI	컴퓨팅 가격	Windows 라이선스 가격	SQL Server 라이선스 가격	총 가격
r5.xlarge	Linux(컴퓨팅 요금)	183.96 USD	–	–	183.96 USD
r5.xlarge	Linux + SQL Server 개발자 에디션	183.96 USD	$0	$0	183.96 USD

EC2 인스턴스	AMI	컴퓨팅 가격	Windows 라이선스 가격	SQL Server 라이선스 가격	총 가격
r5.xlarge	Windows Server(LI)	183.96 USD	134.32 USD	–	318.28 USD
r5.xlarge	Windows + SQL Server 개발자 에디션	183.96 USD	134.32 USD	$0	318.28 USD
r5.xlarge	Windows + SQL Server 웹 에디션(LI)	183.96 USD	134.32 USD	49.64 USD	367.92 USD
r5.xlarge	Windows + SQL Server Standard Edition(LI)	183.96 USD	134.32 USD	350.4 USD	668.68 USD
r5.xlarge	Windows + SQL Server Enterprise Edition(LI)	183.96 USD	134.32 USD	1,095 USD	$1413.28

비용 최적화 시나리오

데이터 무결성 회사가 새로 인수한 후 관리형 호스팅 공급자의 현재 위치에서 새로 획득한 워크로드를 마이그레이션하여의 다른 워크로드와 통합하려고 했습니다 AWS 클라우드. 초기 요금에서는 회사의 SQL Server 워크로드가 현재 관리형 서비스 공급자 AWS 보다에서 60% 더 많이 실행되는 것으로 나타났습니다. MACO SME가 추정을 평가하여 고객이 개발 및 테스트 환경에 대해 관리형 호스팅 공급자의 SQL Server 라이선스 비용을 실제로 지불하고 있음을 발견했습니다. 마이그레이션 중에 비프로덕션 워크로드를 SQL Server 개발자 에디션으로 전환하여 SQL Server 라이선스를 40% 줄였습니다.

Amazon EC2에 포함된 SQL Server 라이선스

라이선스 포함 AMIs를 사용하는 EC2 인스턴스에 SQL Server가 있는 경우 Enterprise Edition에서 Developer Edition으로 직접 변환할 수 없습니다. 라이선스 포함 인스턴스의 라이선스 비용은 AMI에 적용됩니다. SQL Server가 운영 체제 내에서 제거되더라도 EC2 인스턴스에는 여전히 라이선스 비용이 청구됩니다.

Developer Edition으로 변환하려면 SQL Server Developer Edition을 다운로드하여 새 EC2 인스턴스에 설치한 다음 데이터베이스를 마이그레이션해야 합니다. 다양한 방법을 사용하여 EC2 인스턴스 간에 SQL Server 데이터베이스를 마이그레이션할 수 있습니다. 자세한 내용은 Microsoft SQL Server 데이터베이스를 설명서로 마이그레이션의 SQL Server 데이터베이스 마이그레이션 방법을 참조하세요 AWS 클라우드. 또한 자동 SQL Server 개발자 솔루션을 사용하여 마이그레이션하려는 새 인스턴스를 준비할 수 있습니다.

Amazon EC2의 SQL Server BYOL

BYOL을 사용하는 SQL Server 인스턴스가 있는 경우 다음 인플레이스 변환 또는 side-by-side 다운그레이드 옵션 중에서 선택할 수 있습니다.

- Microsoft 웹 사이트에서 SQL Server 개발자 에디션을 다운로드합니다. 수동 또는 자동 설치 지침은 AWS 블로그의 SQL Server 개발자 배포 자동화 게시물을 참조하세요.
- SQL Server 기본 백업 및 복원을 사용하여 데이터베이스를 마이그레이션하거나 한 SQL 인스턴스에서 다른 SQL 인스턴스로 데이터베이스를 분리/연결할 수 있습니다.
- 대량 배포에 자동화 도구를 사용합니다.

> **Note**
>
> SQL Server 개발자 에디션은 비프로덕션 환경 전용입니다.

추가 리소스

- EC2에 SQL Server Developer Edition을 배포하기 위한 SQL Server Developer 배포 자동화(AWS 블로그)
- SQL 2022 요금(Microsoft)
- SQL 2019 요금(Microsoft)
- 라이선스 옵션(Amazon EC2의 SQL Server)

- AWS Pricing Calculator (Amazon EC2의 SQL Server 설명서)
- Microsoft SQL Server 2019 라이선싱 가이드(Microsoft에서 다운로드)
- SQL Server 2022 개발자 에디션(Microsoft에서 다운로드)

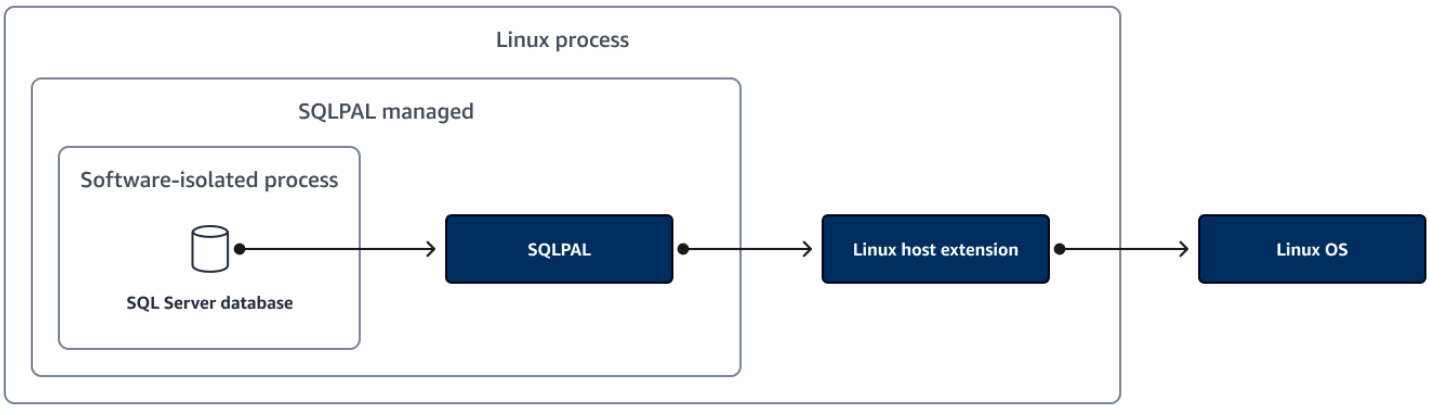
Linux에서 SQL Server 평가

개요

SQL Server 2017부터 Linux 운영 체제에 SQL Server를 설치할 수 있었습니다. Linux 기반 SQL Server는 엔터프라이즈에 바로 사용할 수 있으며 유연성, 고성능, 보안 기능, TCO 절감, HA/DR 기능 및 우수한 사용자 경험을 제공합니다. Windows Server의 SQL Server에서 Linux의 SQL Server로 전환하여 Windows Server 라이선스 비용을 절감할 수 있습니다.

Linux의 경우 SQL Server를 Red Hat Enterprise Linux(RHEL), SUSE Linux Enterprise Server(SLES), Ubuntu 및 Amazon Linux 2에 배포할 수 있습니다. SQL Server 데이터베이스 엔진은 Windows Server와 Linux 모두에서 동일한 방식으로 실행되지만 Linux를 사용할 때 특정 작업에 몇 가지 기본적인 변경 사항이 있습니다. Linux와 Windows에서 SQL Server Always On 애플리케이션을 실행하는 것의 한 가지 주요 차이점은 장애 조치 클러스터링과 관련이 있습니다. Windows Server 호스트에 Always On 가용성 그룹을 배포하는 경우 장애 조치 클러스터링을 지원하는 내장 기능으로 Windows Server 장애 조치 클러스터링(WSFC) 및 Active Directory를 활용할 수 있습니다. 그러나 Linux에서 장애 조치 클러스터링을 지원하는 데 WSFC와 Active Directory를 사용할 수 없습니다. Linux 기반 SQL Server에 대한 장애 조치 클러스터링을 시작하려면 AWS Launch Wizard를 사용하여 ClusterLabs Pacemaker를 사용하여 Linux 인스턴스에서 클러스터 설정 및 SQL 설치를 간소화할 수 있습니다.

Windows 및 Linux의 SQL Server는 공통 코드 기반을 공유합니다. 즉, SQL Server 코어 엔진이 Linux에서 실행되도록 변경되지 않았습니다. 다음 다이어그램과 같이 SQL Server는 플랫폼 추상화 계층(SQLPAL)을 도입했습니다.



SQLPAL은 SQL Server와 기본 운영 체제 간의 호출 및 통신을 추상화하는 역할을 합니다. 호스트 확장은 단순히 네이티브 Linux 애플리케이션입니다. 하위 수준 운영 체제 함수는 I/O, 메모리 및 CPU 사용량을 최적화하기 위한 기본 호출입니다. 호스트 확장이 시작되면 SQLPAL을 로드하고 초기화하여 SQL Server를 불러옵니다. SQLPAL은 나머지 코드에 필요한 번역을 제공하는 격리된 소프트웨어 프로세스를 시작합니다. SQL Server 아키텍처에이 새 계층을 추가하면 운영 체제에 관계없이 SQL Server를 Windows에서 매우 강력하게 만든 것과 동일한 엔터프라이즈급 핵심 기능과 이점을 사용할 수 있습니다.

비용 영향

r5.2xlarge 인스턴스의 경우 Windows Server 라이선스 비용 절감은 각 시나리오에서 약 268 USD입니다. 더 저렴한 SQL Server 에디션을 사용할 때보다 총 서버 비용의 비율이 더 높습니다. 다음 표에는 비용 절감이 나와 있습니다.

Instance	Edition	Windows 기반 SQL Server의 월별 비용	Linux 기반 SQL Server의 월별 비용	절감
r5.2xlarge	웹	$735	466 USD	37%
r5.2xlarge	표준	1,337 USD	1,068 USD	20%
r5.2xlarge	엔터프라이즈	2,826 USD	2,558 USD	10%

> **Note**
>
> 이전 표의 요금 추정은 `us-east-1` 리전의 온디맨드 요금을 기반으로 하며에서 직접 볼 수 있습니다AWS Pricing Calculator.

SMB 세그먼트의 ISV 고객이 개발 환경에서 비용을 절감하려는 예제 시나리오를 생각해 보세요. 이미 Windows 서버 세트에서 SQL Server Developer 에디션을 사용하고 있습니다. Windows with SQL Server Developer Edition에서 Linux with SQL Server Developer Edition으로 전환하면 ISV 고객은 개발 워크로드를 33% 절감할 수 있습니다. 다음 표에는이 시나리오에 대한 다음과 같은 예상 비용이 나와 있습니다.

Estimate	월별 비용
Windows + SQL Server	9,307.72 USD
Linux + SQL Server	6,218.36 USD
예상 비용 절감	$3,089.36(33%)

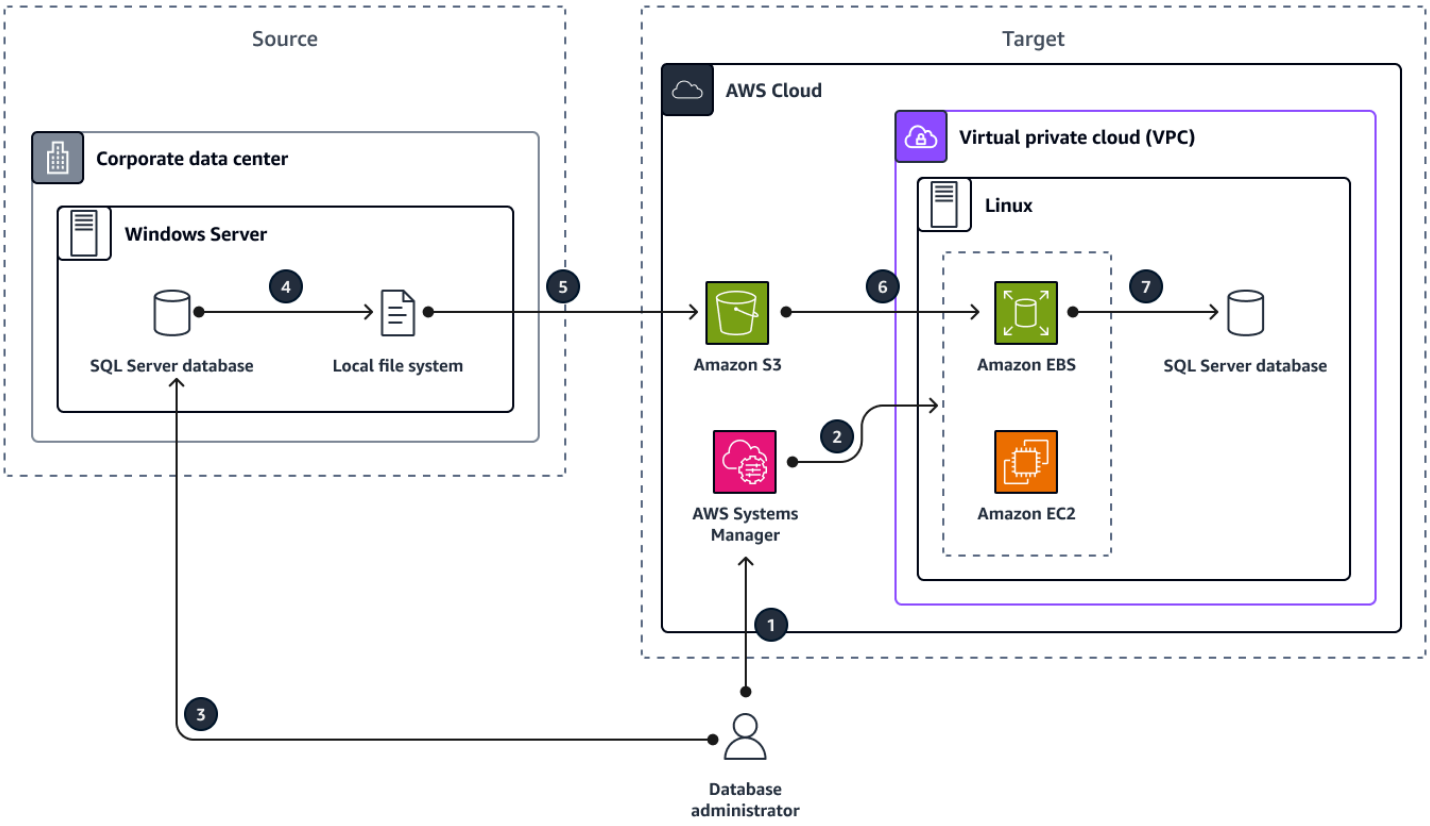
또 다른 예제 시나리오에서는 회사가 라이선스 포함 SQL Server EC2 인스턴스를 Windows에서 Linux로 마이그레이션합니다. 이 회사는 Windows Server 라이선스 비용을 연간 총 300,000 USD 절감하며, 이는 총 AWS 청구서의 약 20%에 해당합니다.

비용 최적화 권장 사항

다음 사항을 고려하는 것이 좋습니다.

- Linux 기반 SQL Server는 SQL Server 2017부터 지원됩니다.
- Windows에서 Linux로 전환하는 데 도움이 되도록 Microsoft SQL Server Databases용 리플랫포밍 어시스턴트를 사용할 수 있습니다. 리플랫포밍 어시스턴트는 일반적인 비호환성을 확인하고, Windows 호스트에서 데이터베이스를 내보낸 다음 Ubuntu 16.04에서 Microsoft SQL Server 2017을 실행하는 EC2 인스턴스로 데이터베이스를 가져와 기존 SQL Server 워크로드를 Windows에서 Linux 운영 체제로 이동하는 데 도움이 되는 스크립팅 도구입니다.
- SQL Server의 백업 및 복원 기능을 사용하여 Windows의 SQL Server에서 Linux로 전환할 수도 있습니다.
- 를 사용하여 Linux 또는 Ubuntu의 SQL Server에 쉽고 빠르게 배포할 수 있습니다AWS Launch Wizard. Launch Wizard는 애플리케이션 요구 사항에 따라 독립 실행형 시나리오와 고가용성 시나리오 모두에서 Linux 또는 Ubuntu에 SQL Server를 배포할 수 있습니다. 자세한 내용은 AWS 블로그의 Microsoft 워크로드에서 Linux 기반 SQL Server Always에 배포 AWS Launch Wizard 게시물을 참조하세요.

다음 다이어그램은 Microsoft SQL Server Databases용 Windows에서 Linux로의 리플랫포밍 어시스턴트를 사용하는 솔루션의 아키텍처를 보여줍니다.



추가 리소스

- Linux 기반 SQL Server 개요(Microsoft Learn)
- Linux 기반 SQL Server 설치 가이드(Microsoft Learn)
- 를 사용하여 Linux에서 SQL Server Always에 배포 AWS Launch Wizard(AWS 블로그의 Microsoft 워크로드)
- Linux 기반 고가용성 SQL Server(AWS 오픈 소스 블로그)

SQL Server 백업 전략 최적화

개요

대부분의 조직은 복구 시점 목표(RPO), 마지막 백업 이후 허용되는 최대 시간, 복구 시간 목표(RTO), 서비스 중단과 서비스 복원 사이의 허용되는 최대 지연에 대한 현재 요구 사항을 충족하기 위해 Amazon EC2의 SQL Server에서 데이터를 보호할 수 있는 적절한 솔루션을 찾고 있습니다. EC2 인스턴스에서 SQL Server를 실행하는 경우 데이터 백업을 생성하고 데이터를 복원하는 여러 옵션이 있습니다. SQL Server on Amazon EC2의 데이터를 보호하기 위한 백업 전략에는 다음이 포함됩니다.

- Windows Volume Shadow Copy Service(VSS) 지원 Amazon Elastic Block Store(Amazon EBS) 스냅샷 또는 AWS Backup을 사용한 서버 수준 백업
- SQL Server에서 기본 백업 및 복원을 사용한 데이터베이스 수준 백업

데이터베이스 수준 기본 백업에는 다음과 같은 스토리지 옵션이 있습니다.

- Amazon EBS 볼륨을 사용한 로컬 백업
- Amazon FSx for Windows File Server 또는 Amazon FSx for NetApp ONTAP을 사용한 네트워크 파일 시스템 백업 FSx NetApp
- 를 사용하여 Amazon Simple Storage Service(Amazon S3)에 네트워크 백업 AWS Storage Gateway
- Amazon S3 for SQL Server 2022로 직접 백업

이 단원에서는 다음을 수행합니다.

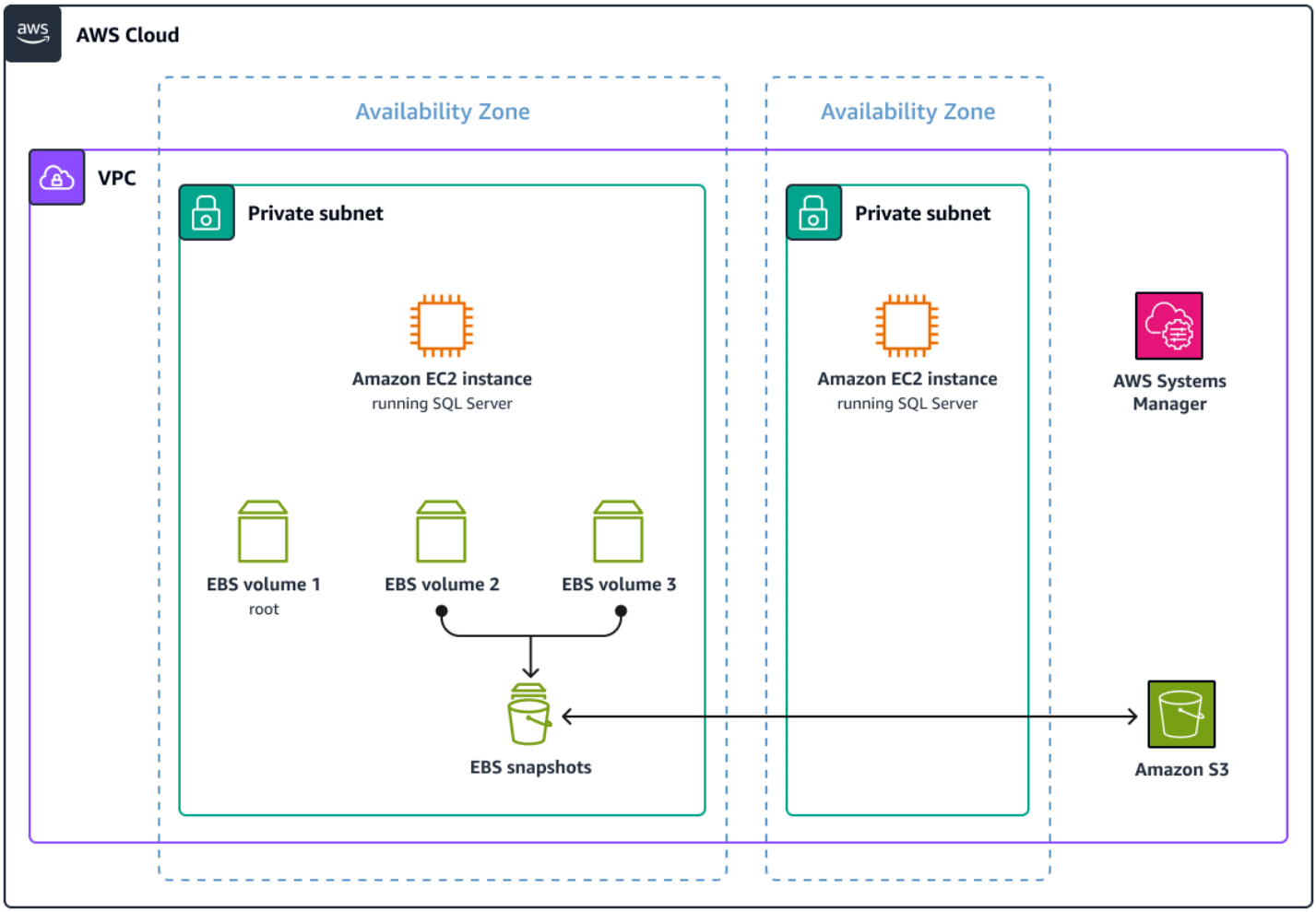
- 스토리지 공간을 절약하는 데 도움이 되는 기능을 강조합니다.
- 다양한 백엔드 스토리지 옵션 간의 비용을 비교합니다.
- 이러한 권장 사항을 구현하는 데 도움이 되는 심층 설명서 링크를 제공합니다.

VSS 지원 스냅샷을 사용한 서버 수준 백업

VSS 지원 스냅샷 아키텍처는 AWS Systems Manager Run Command를 사용하여 SQL Server 인스턴스에 VSS 에이전트를 설치합니다. 또한 Run Command를 사용하여 운영 체제 및 애플리케이션 버퍼를 디스크에 플러시하고, I/O 작업을 일시 중지하고, EBS 볼륨의 특정 시점 스냅샷을 생성한 다음, I/O를 재개하는 전체 워크플로를 간접적으로 호출할 수 있습니다.

이 Run Command는 대상 인스턴스에 연결된 모든 EBS 볼륨의 자동 스냅샷을 생성합니다. 사용자 데이터베이스 파일은 일반적으로 다른 볼륨에 저장되므로 루트 볼륨을 제외하는 옵션도 있습니다. 여러 EBS 볼륨을 스트라이핑하여 SQL Server 파일용 단일 파일 시스템을 생성하는 경우 Amazon EBS는 단일 API 명령을 사용하여 중단 일관성 다중 볼륨 스냅샷도 지원합니다. 애플리케이션 일치 VSS 지원 EBS 스냅샷에 대한 자세한 내용은 Amazon EC2 설명서의 VSS 애플리케이션 일치 스냅샷 생성을 참조하세요.

다음 다이어그램은 VSS 지원 스냅샷을 사용한 서버 수준 백업을 위한 아키텍처를 보여줍니다.



VSS 지원 스냅샷을 사용하면 다음과 같은 이점이 있습니다.

- DB 인스턴스의 첫 번째 스냅샷에는 전체 DB 인스턴스에 대한 데이터가 포함됩니다. 동일한 DB 인스턴스의 후속 스냅샷은 증분식이며, 마지막 스냅샷 이후 변경된 데이터만 저장됩니다.
- EBS 스냅샷은 point-in-time 복구를 제공합니다.
- 스냅샷에서 새 SQL Server EC2 인스턴스로 복원할 수 있습니다.
- Amazon EBS를 사용하여 인스턴스를 암호화하거나 TDE를 사용하여 인스턴스에서 데이터베이스를 암호화하는 경우 해당 인스턴스 또는 데이터베이스는 동일한 암호화로 자동으로 복원됩니다.
- 자동화된 크로스 리전 백업을 복사할 수 있습니다.
- 스냅샷에서 EBS 볼륨을 복원하면 애플리케이션이 즉시 EBS 볼륨에 액세스할 수 있게 됩니다. 즉, 스냅샷에서 하나 이상의 기본 EBS 볼륨을 복원한 후 SQL Server를 즉시 온라인 상태로 전환할 수 있습니다.
- 기본적으로 복원된 볼륨은 애플리케이션이 처음 읽기를 시도할 때 Amazon S3에서 기본 블록을 가져옵니다. 이는 스냅샷에서 EBS 볼륨이 복원된 후 성능이 지연될 수 있음을 의미합니다. 볼륨은 결

국 공칭 성능을 따라잡습니다. 그러나 빠른 스냅샷 복원(FSR) 스냅샷을 사용하면 이러한 지연을 방지할 수 있습니다.

- EBS 스냅샷에 수명 주기 관리를 사용할 수 있습니다.

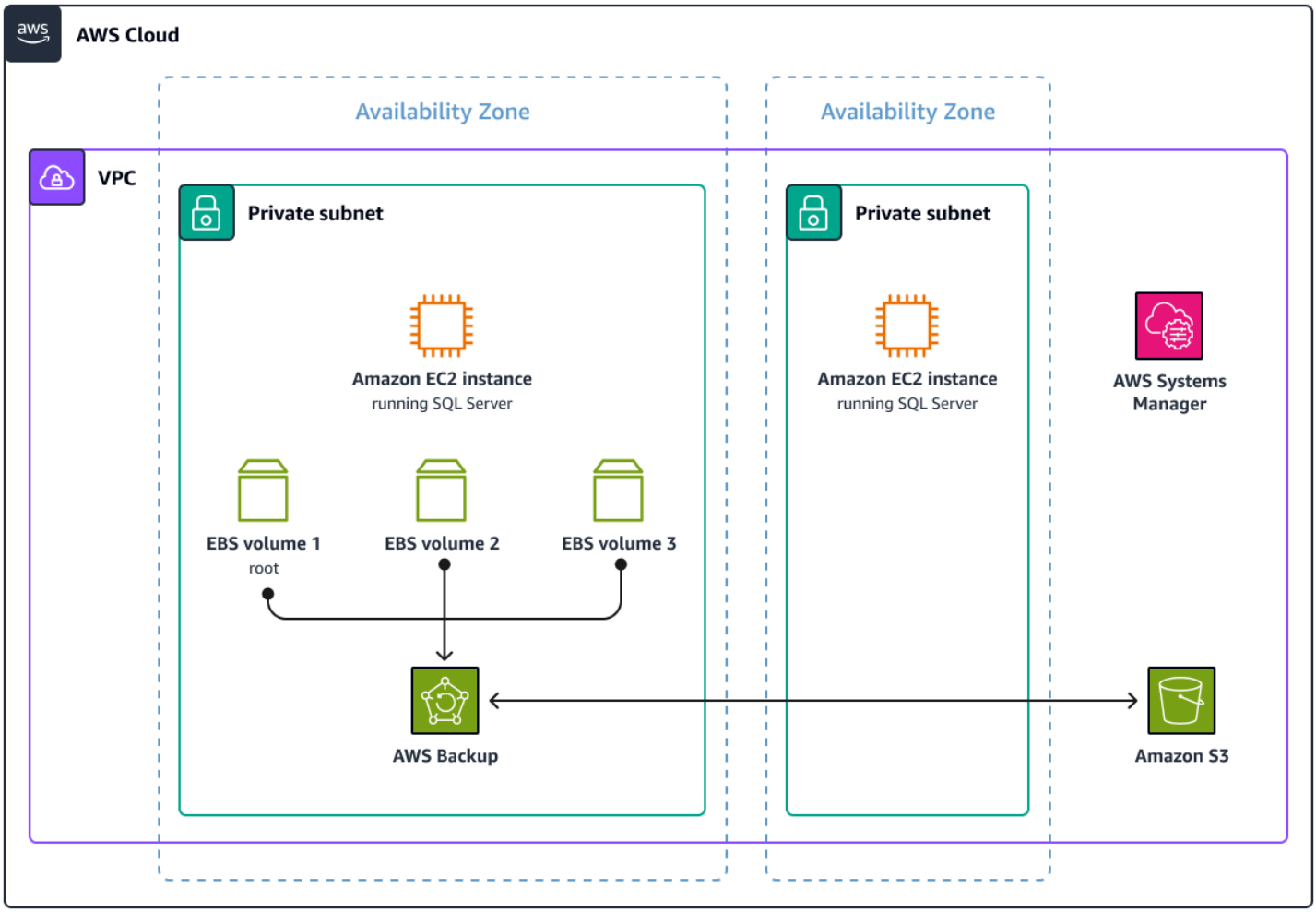
VSS 지원 스냅샷 사용에 대한 다음 제한 사항을 고려하세요.

- SQL Server 인스턴스의 암호화된 스냅샷으로는 리전 간 point-in-time 복구를 수행할 수 없습니다.
- 암호화되지 않은 인스턴스의 암호화된 스냅샷은 생성할 수 없습니다.
- 스냅샷은 EBS 볼륨 수준에서 생성되므로 개별 데이터베이스를 복원할 수 없습니다.
- 인스턴스를 자체로 복원할 수 없습니다.
- DB 인스턴스의 스냅샷은 DB 인스턴스와 동일한 AWS Key Management Service (AWS KMS) 키를 사용하여 암호화해야 합니다.
- 스냅샷 백업 프로세스 중에는 스토리지 I/O가 1초 미만(약 10밀리초) 동안 일시 중지됩니다.

를 사용한 SQL Server 백업 AWS Backup

AWS Backup를 사용하여 데이터 보호를 중앙 집중화하고 자동화할 수 있습니다 AWS 서비스.는 대규모로 데이터 보호를 간소화하는 비용 효율적인 완전 관리형 정책 기반 솔루션을 AWS Backup 제공합니다. AWS Backup 또한는 규정 준수 의무를 지원하고 비즈니스 연속성 목표를 달성하는 데 도움이 됩니다. 와 함께 데이터 보호(백업) 정책을 AWS Organizations AWS Backup 중앙에서 배포하여 조직의 AWS 계정 및 리소스 전반에 걸쳐 백업 활동을 구성, 관리 및 관리할 수 있습니다.

다음 다이어그램은를 사용하여 EC2의 SQL Server에 대한 백업 및 복원 솔루션의 아키텍처를 보여줍니다 AWS Backup.



를 사용하여 SQL Server를 백업하면 AWS Backup다음과 같은 이점이 있습니다.

- 백업 일정, 보존 관리 및 수명 주기 관리를 자동화할 수 있습니다.
- 여러 계정 및에 걸쳐 조직 전체에서 백업 전략을 중앙 집중화할 수 있습니다 AWS 리전.
- 에서 백업 활동 모니터링 및 알림을 중앙 집중화할 수 있습니다 AWS 서비스.
- 재해 복구 계획을 위해 크로스 리전 백업을 구현할 수 있습니다.
- 크로스 계정 백업을 지원합니다.
- 보조 백업 암호화로 보안 백업을 수행할 수 있습니다.
- 모든 백업은 암호화 키를 사용하여 AWS KMS 암호화를 지원합니다.
- 이 솔루션은 TDE와 함께 작동합니다.
- AWS Backup 콘솔에서 특정 복구 지점으로 복원할 수 있습니다.
- 모든 SQL Server 데이터베이스를 포함하는 전체 SQL Server 인스턴스를 백업할 수 있습니다.

데이터베이스 수준 백업

이러한 접근 방식은 네이티브 Microsoft SQL Server 백업 기능을 사용합니다. SQL Server 인스턴스에서 개별 데이터베이스를 백업하고 복원할 수 있습니다.

네이티브 SQL Server 백업 및 복원에 대한 이러한 각 옵션은 다음과 같은 기능도 지원합니다.

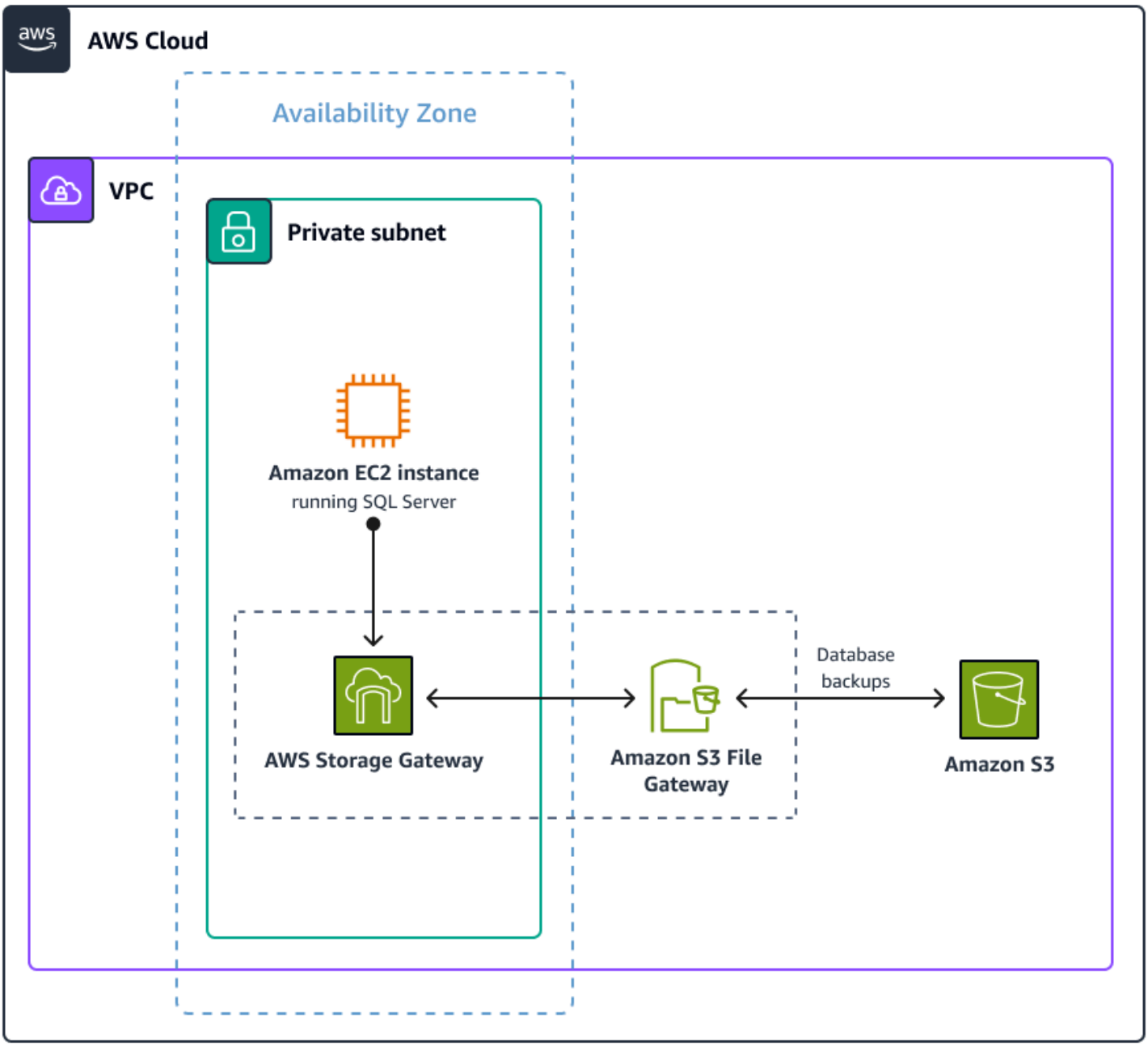
- 압축 및 다중 파일 백업
- 전체, 차등 및 T-로그 백업
- TDE 암호화 데이터베이스

SQL Server 기본 백업 및 Amazon S3로 복원

Amazon EC2의 SQL Server는 SQL Server 데이터베이스의 기본 백업 및 복원을 지원합니다. SQL Server 데이터베이스를 백업한 다음 기존 데이터베이스나 새 SQL Server EC2 인스턴스, Amazon RDS for SQL Server 또는 온프레미스 서버로 백업 파일을 복원할 수 있습니다.

Storage Gateway는 온프레미스 애플리케이션에 사실상 무제한의 클라우드 스토리지에 대한 액세스를 제공하는 하이브리드 클라우드 스토리지 서비스입니다. Storage Gateway를 사용하여 Microsoft SQL Server 데이터베이스를 Amazon S3에 직접 백업하여 온프레미스 스토리지 공간을 줄이고 Amazon S3를 사용하여 내구성, 확장성 및 비용 효율적인 스토리지를 제공할 수 있습니다.

다음 다이어그램은 Storage Gateway 및 Amazon S3를 사용하는 기본 백업 및 복원 솔루션의 아키텍처를 보여줍니다.



Storage Gateway에서 기본 SQL Server 백업을 사용하면 다음과 같은 이점이 있습니다.

- 스토리지 게이트웨이를 EC2 인스턴스의 서버 메시지 블록(SMB) 파일 공유로 매핑하고 백업을 Amazon S3로 전송할 수 있습니다.
- 백업은 S3 버킷으로 직접 이동하거나 Storage Gateway 파일 캐시를 통해 이동합니다.
- 다중 파일 백업이 지원됩니다.

Storage Gateway를 사용한 기본 백업의 제한 사항은 다음과 같습니다.

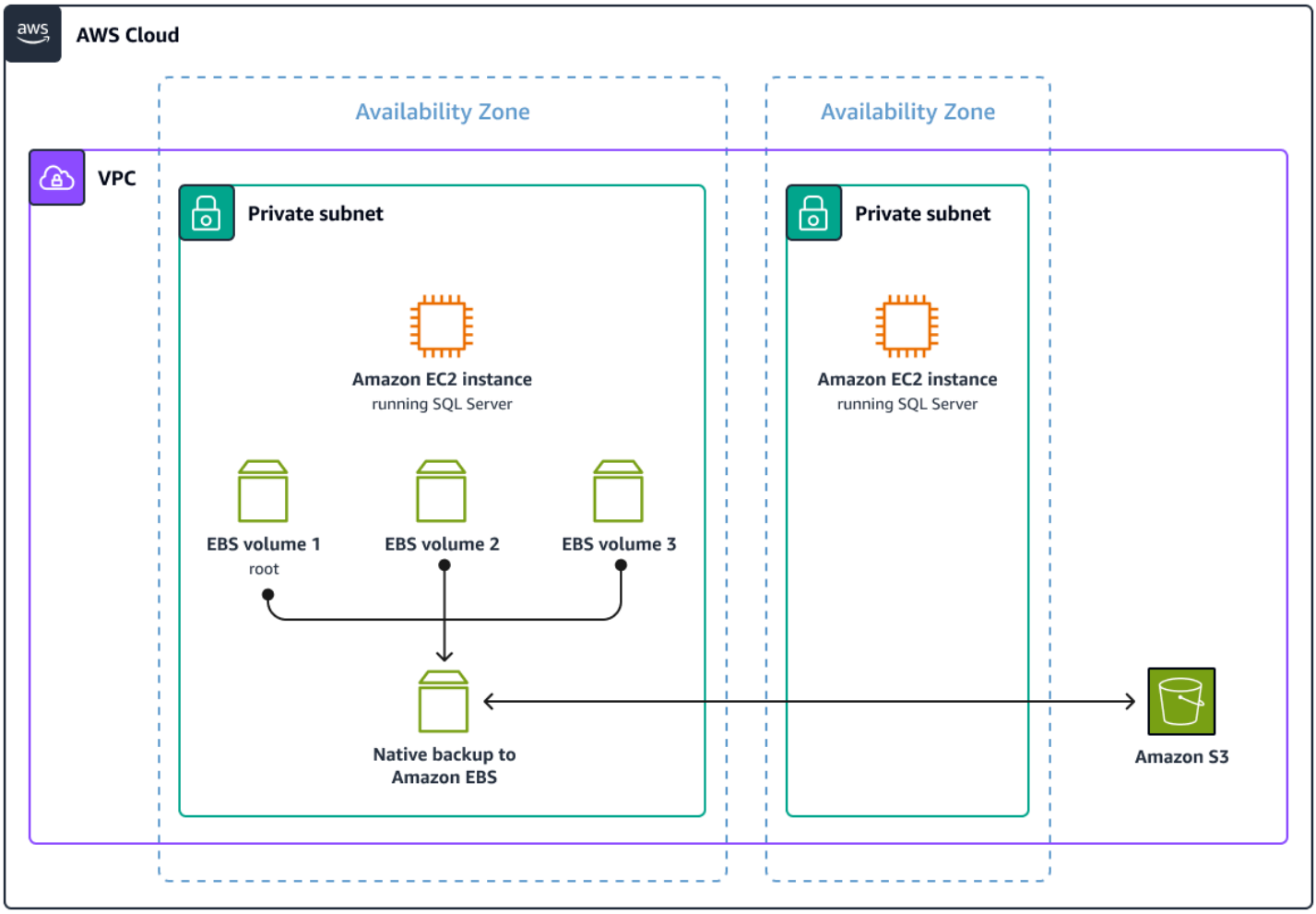
- 각 개별 데이터베이스에 대해 백업 및 복원을 설정해야 합니다.
- 백업 파일에 대한 Amazon S3 수명 주기 정책을 관리해야 합니다.

Storage Gateway를 설정하는 방법에 대한 자세한 내용은 AWS 블로그의 Store SQL Server backups in Amazon S3 using AWS Storage Gateway post를 참조하세요.

EBS 볼륨에 대한 SQL Server 네이티브 백업

SQL Server 데이터베이스의 기본 백업을 가져와 Amazon EBS 볼륨에 파일을 저장할 수 있습니다. Amazon EBS는 고성능 블록 스토리지 서비스입니다. EBS 볼륨은 암호화를 지원하는 탄력적입니다. EC2 인스턴스에서 분리하여 연결할 수 있습니다. 동일한 EBS 볼륨 유형 또는 다른 EBS 볼륨 유형의 EC2 인스턴스에 SQL Server를 백업할 수 있습니다. 다른 EBS 볼륨에 백업할 때의 이점 중 하나는 비용 절감입니다.

다음 다이어그램에서는 EBS 볼륨에 네이티브 백업의 아키텍처를 보여줍니다.



EBS 볼륨에 SQL Server 기본 백업을 사용하면 다음과 같은 이점이 있습니다.

- SQL Server EC2 인스턴스에서 개별 데이터베이스의 백업을 수행하고 전체 인스턴스를 복원하는 대신 개별 데이터베이스를 복원할 수 있습니다.
- 다중 파일 백업이 지원됩니다.
- SQL Server 에이전트와 SQL Server 작업 엔진을 사용하여 백업 작업을 예약할 수 있습니다.
- 하드웨어 선택을 통해 성능상의 이점을 얻을 수 있습니다. 예를 들어, st1 스토리지 볼륨을 사용하여 처리량을 높일 수 있습니다.

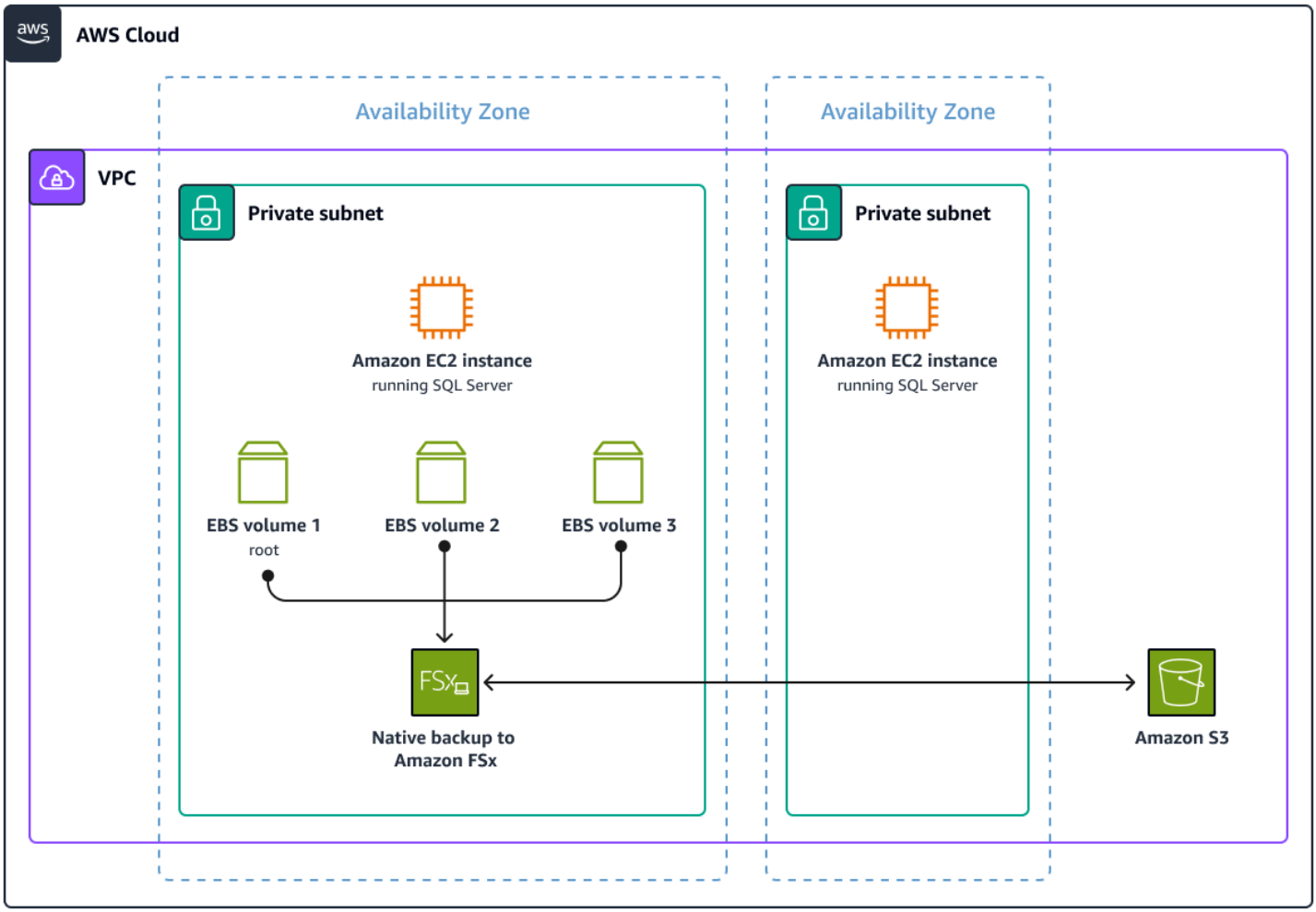
EBS 볼륨에 기본 백업을 사용할 때 다음과 같은 제한 사항을 고려하세요.

- EBS 볼륨에서 Amazon S3로 백업을 수동으로 이동해야 합니다.
- 대규모 백업의 경우 Amazon EC2에서 디스크 공간을 관리해야 합니다.
- EC2 인스턴스에서는 Amazon EBS 처리량이 병목 현상을 일으킬 수 있습니다.
- Amazon EBS에 백업을 저장하려면 추가 스토리지가 필요합니다.

Amazon FSx for Windows File Server에 SQL Server 네이티브 백업

Amazon FSx for Windows File Server는 빠르고 예측 가능하며 일관된 성능을 제공하도록 설계된 최대 64TB의 스토리지를 제공하는 완전관리형 네이티브 Windows 파일 시스템입니다. AWS 는 FSx for Windows File Server에서 다중 AZ 파일 시스템 배포에 대한 네이티브 지원을 도입했습니다. 기본 지원을 사용하면 여러 가용 영역에서 고가용성 및 중복 AWS 성을 사용하여에 Windows 파일 스토리지를 더 쉽게 배포할 수 있습니다. AWS 또한는 SMB 지속적으로 사용 가능(CA) 파일 공유에 대한 지원을 도입했습니다. FSx for Windows File Server를 SQL Server 데이터베이스의 백업 스토리지로 사용할 수 있습니다.

다음 다이어그램에서는 FSx for Windows File Server에 대한 기본 SQL Server 백업의 아키텍처를 보여줍니다.



FSx for Windows File Server에 기본 SQL Server 백업을 사용하면 다음과 같은 이점이 있습니다.

- Amazon FSx 파일 공유에 SQL Server 데이터베이스를 백업할 수 있습니다.
- SQL Server 인스턴스에서 개별 데이터베이스의 백업을 수행하고 전체 인스턴스를 복원하는 대신 개별 데이터베이스를 복원할 수 있습니다.
- 멀티파트 백업이 지원됩니다.
- SQL Server 에이전트와 작업 엔진을 사용하여 백업 작업을 예약할 수 있습니다.
- 인스턴스는 Amazon EBS에 비해 네트워크 대역폭이 더 높습니다.

FSx for Windows File Server에 기본 SQL Server 백업을 사용할 때 다음과 같은 제한 사항을 고려하세요.

- AWS Backup 또는를 사용하여 Amazon FSx에서 Amazon S3로 백업을 수동으로 이동해야 합니다 AWS DataSync.

- 대규모 백업에는 Amazon FSx의 디스크 공간 관리를 위한 추가 오버헤드가 필요할 수 있습니다.
- EC2 인스턴스 네트워크 처리량이 병목 현상을 일으킬 수 있습니다.
- FSx for Windows File Server에 백업을 저장하려면 추가 스토리지가 필요합니다.

Amazon FSx for NetApp ONTAP으로 SQL Server 백업

FSx for ONTAP을 사용하는 스냅샷은 항상 충돌이 일관되지만 애플리케이션에 일관되게 적용되는 스냅샷을 생성하려면 데이터베이스를 중지(또는 데이터베이스의 I/O 일시 중지)해야 합니다. FSx for ONTAP과 함께 NetApp SnapCenter(SQL Server를 비롯한 특정 애플리케이션의 플러그인이 포함된 오케스트레이션 도구)를 사용하여 추가 비용 없이 애플리케이션 일관성 스냅샷을 생성하고 데이터베이스를 보호, 복제 및 복제할 수 있습니다.

NetApp SnapCenter

NetApp SnapCenter는 애플리케이션 일관성 데이터 보호를 위한 통합 플랫폼입니다. SnapCenter는 스냅샷을 백업이라고 합니다. 이 가이드에서는 동일한 이름 지정 규칙을 채택합니다. SnapCenter는 애플리케이션 일관성 백업, 복원 및 복제를 관리하기 위한 단일 창을 제공합니다. 특정 데이터베이스 애플리케이션에 대한 SnapCenter 플러그인을 추가하여 애플리케이션 일관성 백업을 생성합니다. SQL Server용 SnapCenter 플러그인은 데이터 보호 워크플로를 간소화하는 다음과 같은 기능을 제공합니다.

- 전체 및 로그 백업을 위한 세분화된 백업 및 복원 옵션
- 현재 위치 복원 및 대체 위치로 복원

SnapCenter에 대한 자세한 내용은 AWS 스토리지 블로그의 Amazon FSx for NetApp ONTAP과 함께 NetApp SnapCenter를 사용하여 SQL Server 워크로드 보호를 참조하세요.

백업을 위한 비용 최적화

다음 옵션은 SQL Server 백업 저장 비용을 줄이는 데 도움이 될 수 있습니다 AWS.

- 백업 파일을 생성하는 동안 SQL Server 압축을 활성화하고 가능한 가장 작은 파일을 스토리지로 전송합니다. 예를 들어 3:1 압축률은 디스크 공간을 약 66% 절약하고 있음을 나타냅니다. 이러한 열을 쿼리하려면 Transact-SQL 문을 사용할 수 있습니다`SELECT backup_size/compressed_backup_size FROM msdb..backupset;`.
- S3 버킷으로 이동하는 백업의 경우 Amazon S3 Intelligent-Tiering 스토리지 클래스를 활성화하여 스토리지 비용을 30% 절감합니다.

- FSx for Windows File Server 또는 FSx for ONTAP으로 백업하는 경우 단일 가용 영역을 사용하여 비용을 50% 절감합니다(여러 가용 영역을 사용하는 경우와 비교). 요금 정보는 Amazon FSx for Windows File Server 요금 및 Amazon FSx for NetApp ONTAP 요금을 참조하세요.
- SQL Server 2022의 가장 효율적인 옵션은 Amazon S3에 직접 백업하는 것입니다. Storage Gateway를 피하면 추가 비용을 절감할 수 있습니다.

백업에 대한 테스트 결과 벤치마크

이 섹션에서는이 가이드에서 다루는 백업 솔루션에 대한 성능 벤치마크 테스트 결과를 기반으로 샘플 1TB 데이터베이스의 비용 및 성능 관점에서 다음 옵션을 비교합니다.

- EC2 인스턴스 사양 - Windows Server 2019 및 SQL Server 2019 개발자 에디션의 r5d.8xlarge
- 데이터베이스 사양 - TDE가 비활성화된 상태에서 1TB 크기

테스트는 r5d.8xlarge 인스턴스와 1TB SQL Server 데이터베이스를 소스로 사용하여 수행되었습니다. 소스 시스템은 모범 사례에 따라 구성되었으며, 소스 데이터베이스에는 별도의 gp3 볼륨에 분산된 4개의 데이터 파일(각각 250GB)과 1개의 로그 파일(50GB)이 포함되었습니다. SQL Server 네이티브 BACKUP 명령에는 10개의 백업 파일에 쓰기, 압축을 사용하여 백업 성능을 최적화하고 네트워크를 통해 전송되어 대상에 기록되는 데이터의 양을 줄이는 작업이 포함됩니다. 모든 테스트 사례에서 스토리지 성능은 병목 현상이었습니다.

이러한 유형의 테스트에 사용할 수 있는 구성은 거의 무궁무진합니다. 이 테스트는 성능, 비용, 확장성 및 실제 사용 사례를 최적화하는 데 중점을 두었습니다. 다음 표에는 백업 대상 옵션에 대해 캡처된 성능 지표가 나와 있습니다.

백업 옵션	수준	실행 기간(Appx)	백업 속도	월별 비용 USD*
로컬 EBS st1 HDD에 대한 기본 백업, 2TB	데이터베이스	00:30:46분	554.7Mbps	92.16 USD
로컬 EBS SSD gp3에 대한 기본 백업, 2TB	데이터베이스	00:22:00분	512Mbps	193.84 USD
FSx for Windows File Server	데이터베이스	00:20:58분	814.0Mbps	1,146 USD

백업 옵션	수준	실행 기간(Appx)	백업 속도	월별 비용 USD*
HDD에 대한 기본 백업, 2TB @512Mbps 처리량				
FSx for Windows File Server SSD에 대한 기본 백업, 2TB @512Mbps 처리량	데이터베이스	00:20:00분	814.0Mbps	1,326 USD
2TB gp3를 사용하여 S3 File Gateway m6i.4xlarge(16 vCPU, 64GB)에 기본 백업	데이터베이스	00:23:20분	731.5Mbps	470.42 USD
EBS VSS 스냅샷	EBS 볼륨	00:00:02초 00:00:53초	해당 사항 없음 스냅샷	51 USD
AWS Backup (AMI 백업)	AMI	00:00:04초 00:08:00분	해당 사항 없음 스냅샷	75 USD
Amazon S3에 직접 기본 SQL Server 백업(SQL Server 2022)	데이터베이스	00:12:00분	731.5Mbps	첫 50TB/월, GB 당 0.023 USD/월 23.55 USD
FSx for ONTAP에 대한 기본 백업(SnapCenter 사용)	데이터베이스	–	–	440.20 USD

위 표에서는 다음을 가정합니다.

- 데이터 전송 및 Amazon S3 비용은 포함되지 않습니다.
- 스토리지 가격은 인스턴스 요금에 포함됩니다.
- 비용은 `us-east-1` 리전을 기준으로 합니다.
- 처리량과 IOPS는 한 달 동안 전체 변화율이 10%인 여러 백업에서 10% 증가합니다.

테스트 결과에 따르면 가장 빠른 옵션은 FSx for Windows File Server에 대한 기본 SQL Server 데이터베이스 백업입니다. Storage Gateway 및 로컬로 연결된 EBS 볼륨에 대한 백업은 더 비용 효율적인 옵션이지만 성능이 더 느립니다. 서버 수준 백업(AMI)의 경우 최적의 성능, 비용 및 관리 용이성을 AWS Backup 위해를 사용하는 것이 좋습니다.

비용 최적화 권장 사항

Amazon EC2에서 SQL Server를 백업하기 위한 가능한 솔루션을 이해하는 것은 데이터를 보호하고, 백업 요구 사항을 충족하고, 중요한 이벤트로부터 복구하기 위한 계획을 수립하는 데 중요합니다. 이 섹션에서 살펴보는 SQL Server 인스턴스 및 데이터베이스를 백업하고 복원하는 다양한 방법은 데이터를 보호하고 조직의 요구 사항을 충족하는 백업 및 복원 전략을 고안하는 데 도움이 될 수 있습니다.

이 섹션에서는 다음 백업 옵션을 다룹니다.

- 압축
- Amazon S3 Intelligent-Tiering
- 단일 가용 영역
- URL로 백업

이러한 각 옵션에 대해 제공되는 지침은 상위 수준입니다. 조직에서 이러한 권장 사항을 구현하려면 계정 팀에 문의하는 것이 좋습니다. 그런 다음 팀은 Microsoft Specialist SA와 협력하여 대화를 주도할 수 있습니다. optimize-microsoft@amazon.com으로 이메일을 보내 문의할 수도 있습니다.

요약하면 다음을 권장합니다.

- SQL Server 2022를 사용하는 경우 Amazon S3에 백업하는 것이 가장 비용 효율적인 옵션입니다.
- SQL Server 2019 및 이전 SQL Server 에디션을 사용하는 경우 가장 비용 효율적인 옵션으로 Amazon S3에서 지원하는 Storage Gateway에 백업하는 것이 좋습니다.

압축

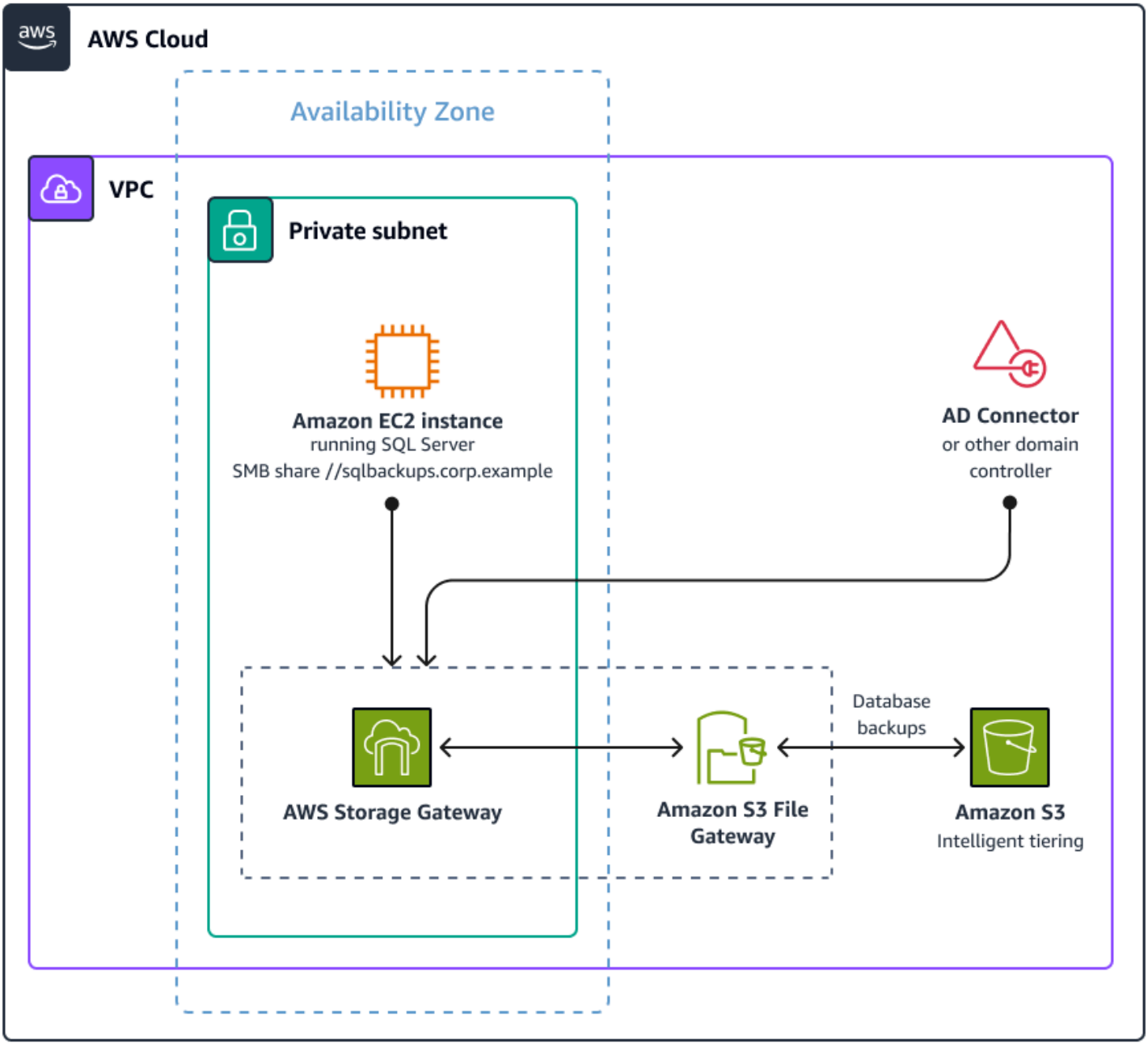
압축의 목표는 각 백업에서 사용하는 스토리지를 줄이는 것이며, 이는 다양한 스토리지 옵션에 유용합니다. SQL Server 인스턴스 수준에서 SQL Server 백업에 대한 압축을 활성화해야 합니다. 다음 예제에서는 백업 데이터베이스에 압축 키워드를 추가하는 방법을 보여줍니다.

```
BACKUP DATABASE <database_name> TO DISK WITH COMPRESSION (ALGORITHM =
QAT_DEFLATE)
```

Amazon S3 Intelligent-Tiering

Amazon S3 버킷으로 이동하는 백업의 경우 Amazon S3 Intelligent-Tiering을 Amazon S3 File Gateway 스토리지 클래스로 활성화할 수 있습니다. 이렇게 하면 스토리지 비용을 최대 30% 절감할 수 있습니다. 그런 다음 Active Directory 도메인과 통합할 수 있는 SMB 파일 공유를 사용하여 S3 File Gateway를 SQL 서버에 탑재합니다. 이를 통해 공유에 대한 액세스 제어, 기존 서비스 계정을 활용할 수 있는 기능, 일반적인 Microsoft 집중 파일 프로토콜을 사용하여 Amazon S3에 대한 액세스를 제공할 수 있습니다. 도메인 컨트롤러에 직접 연결되지 않은 계정의 경우 Active Directory 커넥터를 사용하여 온프레미스 또는 클라우드에서 Active Directory와의 통신을 용이하게 할 수 있습니다. 게이트웨이에서 Active Directory 설정을 구성하려면 도메인 컨트롤러가 Active Directory에 대한 요청을 프록시할 Active Directory 커넥터 IPs를 지정해야 합니다.

다음 다이어그램은 S3 Intelligent-Tiering 기반 솔루션의 아키텍처를 보여줍니다.



기본적으로 S3 버킷에 작성된 백업 파일은 표준 계층을 사용합니다. 백업 파일을 표준 계층에서 S3 Intelligent-Tiering으로 변환하려면 수명 주기 규칙을 생성해야 합니다. 를 사용하여 S3 Intelligent-Tiering을 활성화AWS Management Console할 수도 있습니다. 자세한 내용은 설명서의 Amazon S3 Intelligent-Tiering 사용 시작하기를 AWS 참조하세요.

단일 가용 영역

단일 가용 영역 파일 시스템을 생성하려면 FSx for Windows File Server 파일 시스템을 생성할 때 단일 AZ 옵션을 선택합니다. 또한 Amazon FSx는 Windows Volume Shadow Copy Service를 사용하여 매

일 파일 시스템의 내구성이 뛰어난 백업(Amazon S3에 저장)을 수행하며 언제든지 추가 백업을 수행할 수 있습니다. 단일 가용 영역 사용과 관련된 몇 가지 문제에 유의하세요. 예를 들어 파일 시스템이 프로비저닝된 영향을 받는 가용 영역이 한 번에 몇 시간 동안 다운되면 SMB 파일 공유에 액세스할 수 없게 됩니다. 데이터에 액세스해야 하는 경우 소스 리전 내의 사용 가능한 가용 영역에 있는 백업에서 데이터를 복원해야 합니다. 자세한 내용은이 가이드의 단일 가용 영역 사용 섹션을 참조하세요.

URL로 백업

SQL Server 2022의 경우 URL로 백업 기능을 사용하면 Amazon S3로 직접 백업할 수 있습니다. 이는 스토리지 계층에서 Amazon S3의 전체 기능 세트를 가져오고이 기능을 용이하게 하기 위해 이전 버전에 필요한 어플라이언스 비용을 제거 AWS 하므로에서 실행되는 SQL Server 2022에 AWS Storage Gateway 이상적인 백업 접근 방식입니다. 이 기능을 구현할 때 고려해야 할 두 가지 기본 비용은 데이터 전송 비용과 선택한 S3 스토리지 클래스입니다. Amazon S3의 기본 재해 복구 기능을 원하는 경우 리전 간 복제에 리전 간 데이터 송신 비용이 발생한다는 점을 고려해야 합니다. 이 옵션을 구성하는 방법에 대한 자세한 내용은 블로그의 Microsoft 워크로드에서 Amazon S3로 SQL Server 데이터베이스 백업 게시물을 AWS 참조하세요.

추가 리소스

- Amazon EC2의 SQL Server에 대한 백업 및 복원 옵션(AWS 권장 가이드)
- 를 사용한 Amazon RDS의 Point-in-time 복구 및 연속 백업 AWS Backup(AWS 스토리지 블로그)
- Amazon FSx for NetApp ONTAP과 함께 NetApp SnapCenter를 사용하여 SQL Server 워크로드 보호(AWS 스토리지 블로그)
- Amazon S3 Intelligent-Tiering 사용 시작하기(AWS 리소스 센터 시작하기)
- Amazon RDS for SQL Server의 백업 및 복원 전략(AWS 데이터베이스 블로그)
- 온프레미스 Microsoft SQL Server 데이터베이스를 Amazon EC2로 마이그레이션(AWS 권고 가이드)
- Amazon EC2에 Microsoft SQL Server를 배포하는 모범 사례(AWS 백서)

SQL Server 데이터베이스 현대화

개요

확장성, 성능 및 비용 최적화를 위해 레거시 데이터베이스를 현대화하는 여정을 시작하는 경우 SQL Server와 같은 상용 데이터베이스와 관련된 문제에 직면할 수 있습니다. 상용 데이터베이스는 비용이

많이 들고 고객을 잠그며 징벌적 라이선스 조건을 제공합니다. 이 섹션에서는 SQL Server에서 오픈 소스 데이터베이스로 마이그레이션하고 현대화하는 옵션과 워크로드에 가장 적합한 옵션을 선택하는 방법에 대한 간략한 개요를 제공합니다.

SQL Server 데이터베이스를 Amazon Aurora PostgreSQL과 같은 오픈 소스 데이터베이스로 리팩터링하여 Windows 및 SQL Server 라이선스 비용을 절감할 수 있습니다. Aurora와 같은 클라우드 네이티브 최신 데이터베이스는 오픈 소스 데이터베이스의 유연성과 저렴한 비용을 상용 데이터베이스의 강력한 엔터프라이즈급 기능과 병합합니다. 가변 워크로드 또는 다중 테넌트 워크로드가 있는 경우 Aurora 서버리스 V2로 마이그레이션할 수도 있습니다. 이렇게 하면 워크로드 특성에 따라 비용을 90%까지 줄일 수 있습니다. 또한는 Babelfish for Aurora PostgreSQL과 같은 기능, AWS Schema Conversion Tool (AWS SCT)와 같은 도구, AWS Database Migration Service (AWS DMS)와 같은 서비스를 AWS 제공하여 SQL Server 데이터베이스의 마이그레이션 및 현대화를 간소화합니다 AWS.

데이터베이스 제품

Windows의 SQL Server에서 Amazon Aurora, Amazon RDS for MySQL 또는 Amazon RDS for PostgreSQL과 같은 오픈 소스 데이터베이스로 마이그레이션하면 성능이나 기능을 손상시키지 않고도 상당한 비용 절감 효과를 얻을 수 있습니다. 다음을 고려하세요.

- Amazon EC2의 SQL Server Enterprise Edition에서 Amazon RDS for PostgreSQL 또는 Amazon RDS for MySQL로 전환하면 비용을 최대 80% 절감할 수 있습니다.
- Amazon EC2의 SQL Server Enterprise Edition에서 Amazon Aurora PostgreSQL 호환 버전 또는 Amazon Aurora MySQL 호환 버전으로 전환하면 비용을 최대 70% 절감할 수 있습니다.

기존 데이터베이스 워크로드의 경우 Amazon RDS for PostgreSQL 및 Amazon RDS for MySQL은 요구 사항을 해결하고 관계형 데이터베이스를 위한 비용 효율적인 솔루션을 제공합니다. Aurora는 이전에 비용이 많이 드는 상용 공급업체로 제한된 수많은 가용성 및 성능 기능을 추가합니다. Aurora의 복원력 기능은 추가 비용입니다. 그러나 다른 상용 공급업체의 유사한 기능에 비해 Aurora의 복원력 비용은 동일한 유형의 기능에 대해 상용 소프트웨어가 부과하는 것보다 여전히 저렴합니다. Aurora 아키텍처는 표준 MySQL 및 PostgreSQL 배포에 비해 성능이 크게 향상되도록 최적화되었습니다.

Aurora는 오픈 소스 PostgreSQL 및 MySQL 데이터베이스와 호환되므로 이식성의 추가 이점이 있습니다. 최상의 옵션이 Amazon RDS for PostgreSQL, Amazon RDS for MySQL 또는 Aurora인지 여부는 비즈니스 요구 사항을 이해하고 필요한 기능을 최상의 옵션에 매핑하는 데 달려 있습니다.

Amazon RDS와 Aurora 비교

다음 표에는 Amazon RDS와 Amazon Aurora의 주요 차이점이 요약되어 있습니다.

범주	Amazon RDS for PostgreSQL 또는 Amazon RDS for MySQL	Aurora PostgreSQL 또는 Aurora MySQL
성능	우수한 성능	3배 이상의 성능
장애 조치	일반적으로 60~120초*	일반적으로 30초
확장성	최대 5개의 읽기 전용 복제본 초 단위 지연	최대 15개의 읽기 전용 복제본 밀리초 단위 지연
스토리지	최대 64TB	최대 128TB
스토리지 HA	각각 데이터베이스 복사본이 있는 1개 또는 2개의 대기가 있는 다중 AZ	기본적으로 3개의 가용 영역에 걸쳐 데이터 사본 6개
백업	일일 스냅샷 및 로그 백업	Amazon S3에 대한 지속적인 비동기식 백업
Aurora를 사용한 혁신	NA	100GB 빠른 데이터베이스 복제
	자동 크기 조정 읽기 전용 복제본	
	쿼리 계획 관리	
	Aurora 서버리스	
	글로벌 데이터베이스를 사용한 리전 간 복제본	
	클러스터 캐시 관리**	
	병렬 쿼리	
	데이터베이스 활동 스트림	

*대규모 트랜잭션은 장애 조치 시간을 늘릴 수 있습니다.

**Aurora PostgreSQL에서 사용 가능

다음 표에는이 섹션에서 다루는 다양한 데이터베이스 서비스의 예상 월별 비용이 나와 있습니다.

데이터베이스 서비스	월별 비용 USD*	AWS Pricing Calculator (필수 AWS 계정)
Amazon RDS for SQL Server Enterprise 에디션	3,750 USD	Estimate
Amazon RDS for SQL Server Standard 에디션	2,318 USD	Estimate
Amazon EC2의 SQL Server Enterprise Edition	2,835 USD	Estimate
Amazon EC2의 SQL Server Standard 에디션	1,345 USD	Estimate
Amazon RDS for PostgreSQL	742 USD	Estimate
Amazon RDS for MySQL	712 USD	Estimate
Aurora PostgreSQL	1,032 USD	Estimate
Aurora MySQL	1,031 USD	Estimate

* 스토리지 가격은 인스턴스 요금에 포함됩니다. 비용은 `us-east-1` 리전을 기준으로 합니다. 처리량과 IOPS는 가정입니다. 계산은 r6i.2xlarge 및 r6g.2xlarge 인스턴스에 대한 것입니다.

비용 최적화 권장 사항

이기종 데이터베이스 마이그레이션은 일반적으로 데이터베이스 스키마를 소스에서 대상 데이터베이스 엔진으로 변환하고 데이터를 소스에서 대상 데이터베이스로 마이그레이션해야 합니다. 마이그레이션의 첫 번째 단계는 SQL Server 스키마 및 코드 객체를 평가하여 대상 데이터베이스 엔진으로 변환하는 것입니다.

AWS Schema Conversion Tool (AWS SCT)를 사용하여 데이터베이스가 Amazon RDS for MySQL 또는 Amazon RDS for PostgreSQL, Aurora MySQL MySQL 및 PostgreSQL. Babelfish Compass 도구를 사용하여 Babelfish for Aurora PostgreSQL과의 호환성을 평가할 수도 있습니다. 이렇게 하면 AWS SCT 및 Compass 강력한 도구가 마이그레이션 전략을 결정하기 전에 관련된 선결제 작업을 이해할 수 있습니다. 계속 진행하기로 결정하면가 스키마에 필요한 변경 사항을 AWS SCT 자동화합니다. Babelfish Compass의 핵심 철학은 SQL 데이터베이스가 수정 없이 또는 거의 없이 Aurora로 이동할 수 있도록 하는 것입니다. Compass는 기존 SQL 데이터베이스를 평가하여 이를 수행할 수 있는지 확인합니다. 이렇게 하면 SQL Server에서 Aurora로 데이터를 마이그레이션하는 데 노력하기 전에 결과를 알 수 있습니다.

AWS SCT 는 데이터베이스 스키마 및 코드를 대상 데이터베이스 엔진으로의 변환 및 마이그레이션을 자동화합니다. Babelfish for Aurora PostgreSQL을 사용하여 스키마 변경 없이 데이터베이스와 애플리케이션을 SQL Server에서 Aurora PostgreSQL로 마이그레이션할 수 있습니다. 이렇게 하면 마이그레이션이 가속화될 수 있습니다.

스키마가 마이그레이션된 후 AWS DMS 를 사용하여 데이터를 마이그레이션할 수 있습니다.는 전체 데이터 로드를 AWS DMS 수행하고 변경 사항을 복제하여 가동 중지 시간을 최소화하면서 마이그레이션을 수행할 수 있습니다.

이 섹션에서는 다음 도구를 자세히 살펴봅니다.

- AWS Schema Conversion Tool
- Babelfish for Aurora PostgreSQL
- Babelfish Compass
- AWS Database Migration Service

AWS Schema Conversion Tool

AWS SCT 를 사용하여 기존 SQL Server 데이터베이스를 평가하고 Amazon RDS 또는 Aurora와의 호환성을 평가할 수 있습니다. 마이그레이션 프로세스를 간소화하기 위해 AWS SCT 를 사용하여 이기종 데이터베이스 마이그레이션에서 스키마를 한 데이터베이스 엔진에서 다른 데이터베이스 엔진으로 변환할 수도 있습니다. AWS SCT 를 사용하여 애플리케이션을 평가하고 C#, C++, Java 및 기타 언어로 작성된 애플리케이션의 임베디드 애플리케이션 코드를 변환할 수 있습니다. 자세한 내용은 설명서의 를 사용하여 애플리케이션 SQL 변환 AWS SCT을 AWS SCT 참조하세요.

AWS SCT 는 많은 데이터베이스 소스를 지원하는 무료 AWS 도구입니다. 를 사용하려면 소스 데이터베이스를 AWS SCT가리킨 다음 평가를 실행합니다. 그런 다음는 스키마를 AWS SCT 평가하고 평가 보고서를 생성합니다. 평가 보고서에는 요약, 복잡성 및 마이그레이션 작업, 적절한 대상 데이터베이

스 엔진, 변환 권장 사항이 포함됩니다. 다운로드하려면 설명서의 설치, 확인 및 업데이트를 AWS SCT AWS SCT참조하세요 AWS SCT .

다음 표에는 데이터베이스를 다른 대상 플랫폼으로 변경하는 것과 관련된 복잡성을 보여주기 위해에서 AWS SCT 생성된 실행 요약의 예가 나와 있습니다.

대상 플랫폼	자동 또는 최소 변경 사항			복잡한 작업			
	스토리지 객체	코드 객체	변환 작업	스토리지 객체		코드 객체	
Amazon RDS for MySQL	60(98%)	8(35%)	42	1(2%)	1	15(65%)	56
Amazon Aurora MySQL 호환 버전	60(98%)	8(35%)	42	1(2%)	1	15(65%)	56
Amazon RDS for PostgreSQL	60(98%)	12(52%)	54	1(2%)	1	11(48%)	26
Amazon Aurora PostgreSQL 호환 에디션	60(98%)	12(52%)	54	1(2%)	1	11(48%)	26
Amazon RDS for MariaDB	60(98%)	7(30%)	42	1(2%)	1	16(70%)	58
Amazon Redshift	61(100%)	9(39%)	124	0(0%)	0	14(61%)	25

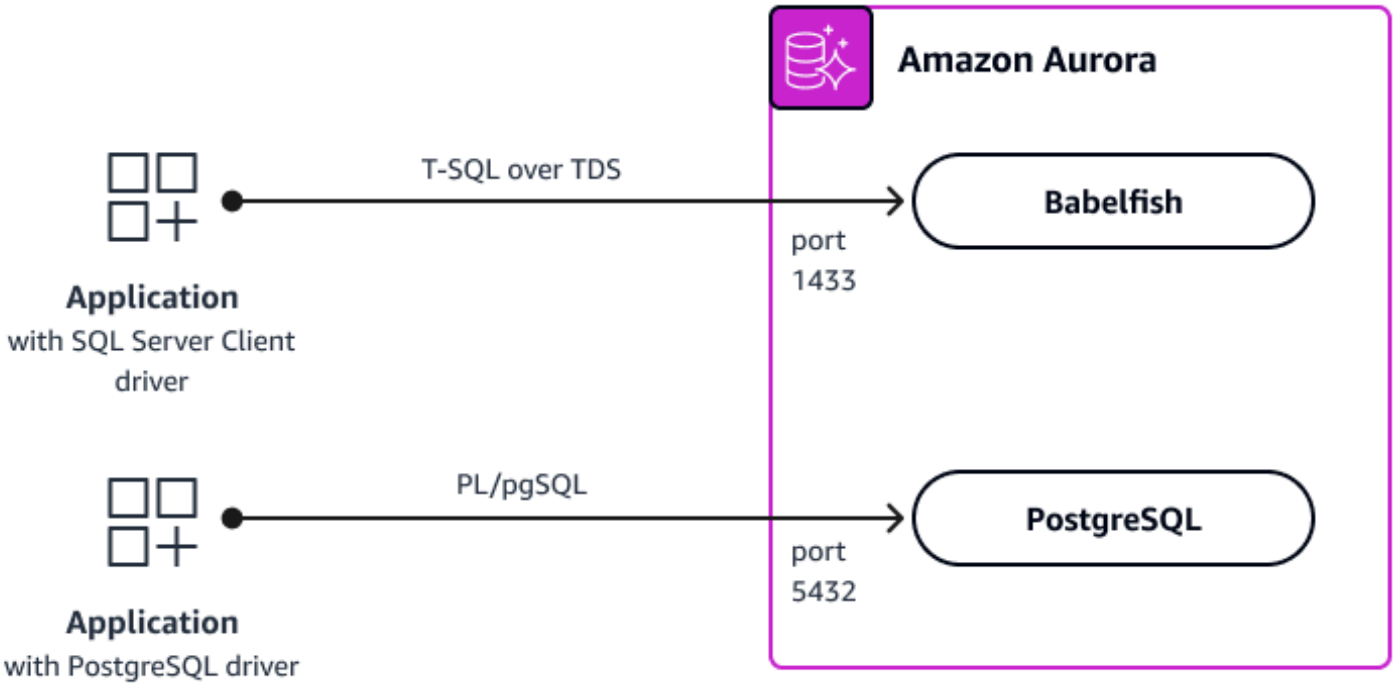
AWS Glue	0(0%)	17(100%)	0	0(0%)	0	0(0%)	0
Babelfish	59(97%)	10(45%)	20	2(3%)	2	12(55%)	30

AWS SCT 또한 보고서는 자동으로 변환할 수 없는 스키마 요소에 대한 세부 정보를 제공합니다. 마이그레이션 플레이북을 참조하여 AWS SCT 변환 격차를 줄이고 대상 스키마를 최적화할 수 있습니다. AWS 이기종 마이그레이션을 지원하는 데이터베이스 마이그레이션 플레이북이 많이 있습니다.

Babelfish for Aurora PostgreSQL

Babelfish for Aurora PostgreSQL은 SQL Server 클라이언트의 데이터베이스 연결을 허용하는 기능으로 Aurora PostgreSQL을 확장합니다. Babelfish를 사용하면 원래 SQL Server용으로 구축된 애플리케이션이 코드 변경이 거의 없고 데이터베이스 드라이버를 변경하지 않고도 Aurora PostgreSQL에서 직접 작동할 수 있습니다. Babelfish는 Aurora PostgreSQL이 T-SQL 및 PL/pgSQL 언어 모두에서 작동할 수 있도록 Aurora PostgreSQL 이중 언어를 바꿉니다.pgSQL Babelfish는 SQL Server에서 Aurora PostgreSQL로 마이그레이션하는 작업을 최소화합니다. 이를 통해 마이그레이션을 가속화하고 위험을 최소화하며 마이그레이션 비용을 크게 절감할 수 있습니다. T-SQL 사후 마이그레이션을 계속 사용할 수 있지만 PostgreSQL 네이티브 도구를 개발에 사용하는 옵션도 있습니다.

다음 다이어그램은 T-SQL을 사용하는 애플리케이션이 SQL Server의 기본 포트 1433에 연결하고 Babelfish 변환기를 사용하여 Aurora PostgreSQL 데이터베이스와 통신하는 방법을 보여주며, PL/pgSQL을 사용하는 애플리케이션은 Aurora PostgreSQL의 기본 포트 5432를 사용하여 Aurora PostgreSQL 데이터베이스에 직접 동시에 연결할 수 있습니다.



Babelfish는 특정 SQL Server T-SQL 기능을 지원하지 않습니다. 이러한 이유로 Amazon은 SQL 문에 대한 line-by-line 분석을 수행하고 Babelfish에서 지원되지 않는 평가 도구가 있는지 확인할 수 있는 평가 도구를 제공합니다.

Babelfish 평가에는 두 가지 옵션이 있습니다.는 SQL Server 데이터베이스와 Babelfish의 호환성을 평가할 AWS SCT 수 있습니다. 또 다른 옵션은 Compass 도구가 Babelfish for Aurora PostgreSQL의 새 릴리스에 따라 업데이트되므로 권장되는 솔루션인 Babelfish Compass 도구입니다.

Babelfish Compass

Babelfish Compass는 Babelfish for Aurora PostgreSQL의 최신 릴리스에 맞춰 무료로 다운로드할 수 있는 도구입니다. 반면 AWS SCT 는 잠시 후 최신 Babelfish 버전을 지원합니다. Babelfish Compass는 SQL Server 데이터베이스 스키마에 대해 실행됩니다. SQL Server Management Studio(SSMS)와 같은 도구를 사용하여 소스 SQL Server 데이터베이스 스키마를 추출할 수도 있습니다. 그런 다음 Babelfish Compass를 통해 스키마를 실행할 수 있습니다. 이렇게 하면 SQL Server 스키마와 Babelfish의 호환성과 마이그레이션 전에 변경이 필요한지 여부를 자세히 설명하는 보고서가 생성됩니다. 또한 Babelfish Compass 도구는 이러한 많은 변경 사항을 자동화하고 궁극적으로 마이그레이션을 가속화할 수 있습니다.

평가 및 변경이 완료되면 SSMS 또는 sqlcmd와 같은 SQL Server 네이티브 도구를 사용하여 스키마를 Aurora PostgreSQL로 마이그레이션할 수 있습니다. 지침은 AWS 데이터베이스 블로그의 Babelfish를 사용하여 SQL Server에서 Amazon Aurora로 마이그레이션 게시물을 참조하세요.

AWS Database Migration Service

스키마가 마이그레이션된 후 AWS Database Migration Service (AWS DMS)를 사용하여 가동 중지 시간을 최소화하면서 데이터를 AWS 로 마이그레이션할 수 있습니다.는 전체 데이터 로드를 수행할 AWS DMS 뿐만 아니라 소스 시스템이 가동되고 실행되는 동안 소스에서 대상으로 변경 사항을 복제합니다. 소스 데이터베이스와 대상 데이터베이스가 모두 동기화된 후 애플리케이션이 마이그레이션을 완료하는 대상 데이터베이스를 가리키는 전환 활동이 발생할 수 있습니다. AWS DMS 현재는 Aurora PostgreSQL 대상에 대해 Babelfish를 사용하여 전체 데이터 로드만 수행하고 변경 사항을 복제하지 않습니다. 자세한 내용은 AWS DMS 설명서의 의 대상으로 Babelfish 사용을 AWS Database Migration Service 참조하세요.

AWS DMS 는 동종(동일한 데이터베이스 엔진에서) 마이그레이션과 이기종(다른 데이터베이스 엔진에서) 마이그레이션을 모두 수행할 수 있습니다.는 많은 소스 및 대상 데이터베이스 엔진을 AWS DMS 지원합니다. 자세한 내용은 데이터베이스 블로그의 Amazon RDS for SQL Server로 SQL Server 데이터베이스 마이그레이션 AWS DMS 게시물을 참조하세요. AWS

추가 리소스

- Goodbye Microsoft SQL Server, Hello Babelfish(AWS 뉴스 블로그)
- CLI(데이터베이스 블로그)를 AWS Schema Conversion Tool 사용하여 데이터베이스 스키마 및 애플리케이션 SQL 변환AWS
- 필드에서 학습한 모범 사례와 교훈을 사용하여 SQL Server를 Amazon Aurora PostgreSQL로 마이그레이션(AWS 데이터베이스 블로그)
- Microsoft SQL Server에서 Amazon RDS for PostgreSQL 및 Amazon Aurora PostgreSQL로 마이그레이션 후 데이터베이스 객체 검증(AWS 데이터베이스 블로그)

SQL Server용 스토리지 최적화

개요

이 섹션에서는 EC2 워크로드의 SQL Server용 Amazon Elastic Block Store(Amazon EBS) SSD 스토리지에 대한 비용 최적화에 중점을 둡니다.

SQL Server 워크로드를 배포하고 실행하기 위한 다양한 스토리지 옵션이 있습니다 AWS. 올바른 스토리지 선택은 목적, 아키텍처, 내구성, 성능, 용량 및 비용을 기반으로 해야 합니다. SQL Server 워크로드를 실행하는 AWS 고객은 일반적으로 Amazon EBS, NVMe, Amazon FSx 및 Amazon Simple Storage Service(Amazon S3) 스토리지의 조합을 사용합니다.

Amazon EBS는 EC2 컴퓨팅 인스턴스에 연결된 네트워크 연결 스토리지로, 일반 운영 체제, 애플리케이션, 데이터베이스 및 백업 파일을 저장하고 처리하는 데 사용됩니다. Amazon EBS SSD(Solid State Drive) 스토리지에는 범용 SSD(gp2 및 gp3) 및 프로비저닝된 IOPS SSD(io1, io2 및 io2BX)가 포함됩니다. 다음을 고려하세요.

- r5d와 같은 일부 EC2 인스턴스에는 호스트 인스턴스에 물리적으로 연결된 로컬 NVMe SSDs가 있습니다. 이러한 볼륨은 SQL Server tempdb 또는 버퍼 풀 확장에 일반적으로 사용되는 블록 수준 스토리지를 제공합니다.
- Amazon FSx for Windows File Server는 완전 관리형 파일 스토리지 서비스인 반면, Amazon FSx for NetApp ONTAP은 NetApp의 인기 있는 ONTAP 파일 시스템을 기반으로 구축된 완전 관리형 공유 스토리지입니다. Amazon FSx는 고가용성 SQL Server 장애 조치 클러스터링된 인스턴스(FCI) 구성에서 SQL Server 워크로드를 실행하는 데 자주 사용됩니다. 이 솔루션은 SQL Server 데이터 및 로그 파일을 호스팅하므로 EC2 인스턴스의 EBS 성능 요구 사항이 줄어듭니다.
- Amazon S3는 업계 최고의 확장성, 데이터 가용성, 보안 및 성능을 제공하는 객체 스토리지 서비스입니다. SQL Server 기본 백업 파일, AMIs, EBS 스냅샷, 애플리케이션 로그 등을 Amazon S3에 저장할 수 있습니다.

Amazon EBS의 SSD 스토리지 유형, 성능 및 비용

Amazon EBS의 SSD 스토리지 비용은 일반적으로 내구성과 성능이 증가함에 따라 증가합니다. 현재 스토리지는 각각 고유한 성능 지표가 있는 5가지 볼륨 유형으로 제공됩니다. SSD 지원 볼륨의 사용 사례 및 특성에 대한 요약은 Amazon EBS 설명서의 SSD(Solid State Drive) 볼륨 섹션에 있는 표를 참조하세요.

Amazon CloudWatch를 사용하여 SSD 성능을 모니터링하고, 추세 데이터를 캡처하고, 특정 임계값이 충족될 때 경보를 설정할 수 있습니다. 에서 SQL Server 워크로드를 실행하는 경우 세부 모니터링을 활성화하고 CloudWatch 사용자 지정 지표를 배포하여 디스크 지연 시간 AWS, IOPS, 처리량, 디스크 대기열 길이, 사용량 및 여유 용량 등과 같은 세부 볼륨 성능 지표를 캡처하는 것이 좋습니다. 이러한 CloudWatch 성능 지표를 사용하여 과소 프로비저닝된 스토리지와 과다 프로비저닝된 스토리지를 식별하고 기록 데이터 포인트를 제공하여 스토리지 요구 사항을 정확하게 정의할 수 있습니다.

Amazon EBS의 SSD 스토리지 비용도 할당된 용량에 따라 달라집니다. 아래 표에는 다양한 볼륨 유형의 비교가 나와 있습니다. 모든 볼륨 유형에는 1TB의 용량과 유사한 성능 구성이 있습니다.

볼륨 유형	최대 IOPS(16KiB I/O)	최대 처리량 (128KiB I/O)	1TB당 요금	비용 절감률
gp2	3,000	250	102.40 USD	
gp3	3,000	250	86.92 USD	15%
io1	16,000	500	1,168 USD	
io2	16,000	500	1,168 USD	
gp3	16,000	500	146.92 USD	87%
io2bx	16,000	4,000	1,168 USD	
gp3	16,000	1,000	181.92 USD	84%

> **Note**
>
> 이전 표의 성능 및 비용 지표는의 추정치를 기반으로 볼륨당입니다 AWS Pricing Calculator. AWS 계정 에서 견적에 액세스하려면가 필요합니다 AWS Pricing Calculator.

Amazon EBS SSD gp3 볼륨은 저렴한 비용으로 우수한 성능을 제공합니다. 16,000 IOPS 미만 및 500MiBps 처리량이 필요한 워크로드에 대해 io1 또는 io2 볼륨보다 gp3 볼륨을 선택하면 최대 87%까지 절약할 수 있습니다.

io2 Block Express(io2BX) 볼륨은 일반 io2 볼륨에 비해 향상된 성능을 제공합니다. 16,000IOPS에서 io1 또는 io2 볼륨은 500MiBps 처리량만 가능하며 io2BX 볼륨은 최대 4,000MiBps 처리량으로 구성할 수 있습니다. io1 및 io2 볼륨과 비교하여 io2BX 볼륨은 정확히 동일한 가격으로 16,000~64,000 IOPS 사이의 처리량의 4배 이상을 제공합니다. 일반 io2 볼륨은 io2BX 지원 EC2 인스턴스에 연결하여 io2BX-supported 볼륨으로 변환할 수 있습니다. io2BX-supported EC2 인스턴스 목록은 Amazon EBS 설명서의 프로비저닝된 IOPS SSD 볼륨을 참조하세요. 새 스토리지를 배포하기 전에 AWS Pricing Calculator를 사용하여 월별 비용을 추정하고 내구성, 성능 및 용량 간의 장단점을 기반으로 비용에 미치는 영향을 이해할 수 있습니다.

Amazon EBS에 대한 일반 SSD 비용 최적화

저장하는 항목을 평가하고 올바른 스토리지 유형 및 클래스를 사용하고 있는지 확인하는 것이 좋습니다. 예를 들어 Amazon S3는 SQL Server 백업에 적합한 가격대, 기본 제공 수명 주기 정책 및 복제 옵션을 제공합니다. SQL Server 2022에는 Amazon S3에 직접 백업할 수 있는 기능이 있지만 이전 버전의 SQL Server는 기본 로컬 백업을 사용합니다. 이전 버전의 SQL Server를 실행하는 경우 Amazon EBS HDD 볼륨에 백업한 다음 백업을 Amazon S3에 복사하는 것이 좋습니다. 이 솔루션은 백업에 gp3 볼륨을 사용하는 대신 53%를 절약할 수 있습니다.

다음 표에는 Amazon EBS gp3, Amazon EBS HDD st1 및 Amazon S3의 1TB 스토리지 요금 차이가 나와 있습니다.

스토리지 유형	Capacity	가격 오후
EBS gp3 500MiBps	1TB	96.92 USD
EBS st1 버스트 500MiBps		46.08 USD
S3 Standard		23.55 USD
S3 Standard(빈번하지 않은 액세스)		12.80 USD
S3 Glacier Deep Archive		1.03 USD

> **Note**
>
> 위 표의 비용 지표는의 추정치를 기반으로 합니다 AWS Pricing Calculator. AWS 계정 에서 견적에 액세스하려면가 필요합니다 AWS Pricing Calculator.

다음 사항을 고려하는 것이 좋습니다.

- 자세한 모니터링을 활성화하고 CloudWatch 사용자 지정 지표를 배포하여 스토리지 성능 요구 사항을 정확하게 캡처합니다.
- Amazon EBS 스토리지를 gp2에서 gp3로 업그레이드하여 비용을 절감하고 유연성을 높이며 성능을 개선합니다.
- 내구성과 성능 유연성을 높이기 위해 Amazon EBS 스토리지를 io1에서 io2로 업그레이드합니다.

- 내구성과 성능을 높이려면 가능하면 io1 또는 io2 대신 io2BX를 사용합니다.
- 스토리지를 선택할 때는 용량 요구 사항과 고성능 볼륨 비용을 줄이는 데 도움이 되는 mix-and-match 접근 방식을 고려하세요. 예를 들어 루트 볼륨(운영 체제), SQL Server 설치, 시스템 데이터베이스(tempdb 제외) 및 성능이 낮은 사용자 데이터베이스에 저비용 gp3 볼륨을 사용할 수 있습니다. 이를 통해 고성능 사용자 데이터베이스 전용으로 사용할 수 있는 io2 볼륨의 용량과 비용을 줄일 수 있습니다.
- SQL Server 데이터베이스를 호스팅하는 경우 데이터베이스당 여러 SQL Server 데이터 파일을 사용하는 AWS것이 좋습니다. 이를 통해 읽기/쓰기 워크로드를 여러 볼륨에 분산하여 볼륨당 성능 및 용량 요구 사항을 줄이고 결과적으로 비용을 절감할 수 있습니다.
- 프로덕션 워크로드에 io1 또는 io2/io2BX와 같은 고성능 스토리지가 필요한 경우에도 비용 절감을 위해 비프로덕션 워크로드의 경우 gp3 볼륨을 고려하세요.
- 시간 경과에 따른 스토리지 사용률을 추적하고 추세를 파악하여 사용량 급증과 예상치 못한 비용을 쉽게 식별할 수 있습니다.
- 실제 사용률에 따라 EBS 볼륨을 늘리거나 줄이는 권장 AWS Compute Optimizer 사항에를 사용합니다.
- 의 탄력성을 사용하여 Amazon EBS용 SSD 볼륨의 성능 및 용량 요구 사항을 AWS 조정합니다. 온프레미스 환경과 달리 향후 워크로드를 위해 스토리지 성능과 용량을 오버프로비저닝할 필요가 없습니다. 데이터베이스를 온라인 상태로 유지하면서 기존 SQL Server 워크로드를 로 마이그레이션 AWS 하고 필요에 따라 성능 또는 용량을 조정할 수 있습니다.

추가 리소스

- Amazon EBS 볼륨 유형(Amazon EBS 설명서)
- Amazon Elastic Block Store(Amazon EBS)(Amazon EBS 설명서)
- 프로비저닝된 IOPS SSD 볼륨(Amazon EBS 설명서)
- SSD 인스턴스 스토어 볼륨(Amazon EC2 설명서)
- Amazon EBS에 대한 Amazon CloudWatch 지표(Amazon EBS 설명서)
- Amazon EC2 스토리지 최적화 인스턴스 사양(Amazon EC2 설명서)
- Amazon FSx for NetApp ONTAP과 함께 NetApp SnapCenter를 사용하여 SQL Server 워크로드 보호(AWS 스토리지 블로그)
- Amazon EC2 FAQ(AWS 제품 페이지)

Compute Optimizer를 사용하여 SQL Server 라이선스 최적화

를 사용하여 SQL Server 라이선스를 최적화하는 방법에 대한 지침입니다 AWS Compute Optimizer.

개요

AWS Compute Optimizer는 Amazon Elastic Compute Cloud(Amazon EC2)의 Microsoft SQL Server 워크로드에 대한 라이선스 최적화 기회를 추천할 수 있습니다. Compute Optimizer는 라이선스 비용을 줄이기 위한 자동화된 권장 사항을 제공할 수 있습니다. Compute Optimizer의 권장 사항은 Microsoft SQL Server 라이선스가 있는 각 EC2 인스턴스 옆에 나열되어 있습니다. 제공되는 정보에는 권장 비용 절감 기회, EC2 인스턴스 온디맨드 요금, 시간당 기존 보유 라이선스 사용(BYOL) 요금 등이 포함됩니다. 이 정보는 라이선스 에디션을 다운그레이드할지 여부를 결정하는 데 도움이 될 수 있습니다.

Compute Optimizer는 추론된 워크로드 유형별로 Amazon EC2에서 SQL Server 인스턴스를 자동으로 검색합니다. 라이선스 권장 사항을 보려면 Compute Optimizer에서 SQL Server 인스턴스를 선택한 다음 읽기 전용 데이터베이스 자격 증명을 사용하여 Amazon CloudWatch Application Insights로 인증할 수 있습니다. Compute Optimizer는 SQL Server Enterprise Edition 기능을 사용하는지 분석합니다. Enterprise Edition 기능을 사용하지 않는 경우 Compute Optimizer는 라이선스 비용을 줄이기 위해 Standard Edition으로 다운그레이드할 것을 권장합니다.

Compute Optimizer를 사용하여 SQL Server 워크로드를 실행하는 Amazon EC2 인스턴스에 대한 크기 조정 권장 사항을 만들 수도 있습니다. 자세한 내용은이 안내서의 Compute Optimizer를 사용하여 SQL Server 크기 조정 최적화를 참조하세요.

비용 최적화 권장 사항

Compute Optimizer의 라이선스 권장 사항은 Microsoft SQL Server에서 사용 중인 기능을 평가하고 워크로드에 가장 비용 효율적인 에디션을 선택하는 데 도움이 될 수 있습니다. SQL Server Enterprise Edition은 Standard Edition보다 훨씬 더 비쌉니다. 자세한 내용은이 가이드의 SQL Server 버전 비교를 참조하고 Microsoft 웹 사이트의 SQL Server 2022 요금을 참조하세요. SQL Server 플릿을 평가하고 권장 사항을 제공하도록 Compute Optimizer를 구성하는 데 시간을 투자하면 라이선스 비용을 크게 줄일 수 있습니다.

라이선스 세부 정보 페이지에는 다음 정보가 제공됩니다.

- 테이블을 사용하여 현재 라이선스 설정(예: 에디션, 모델 및 인스턴스 코어 수)을 Compute Optimizer 권장 사항과 비교합니다.
- 사용률 그래프를 사용하여 분석 기간 동안 사용된 Enterprise Edition 기능의 수를 검토합니다.

자세한 내용은 Compute Optimizer 설명서의 상용 소프트웨어 라이선스 권장 사항의 세부 정보 보기를 참조하세요.

Compute Optimizer 구성

Compute Optimizer는 `mssql_enterprise_features_used` 지표를 사용하여 상용 소프트웨어 라이선스를 분석합니다. 이 지표에 대한 자세한 내용은 상용 소프트웨어 라이선스에 대한 지표를 참조하세요.

1. Compute Optimizer에 옵트인할 수 있는 적절한 권한이 있는지 확인합니다. 자세한 내용은 다음 자료를 참조하세요.
   - Compute Optimizer 옵트인 정책
   - 독립형 Compute Optimizer에 대한 액세스 권한을 부여하는 정책 AWS 계정
   - 조직의 관리 계정에 대한 Compute Optimizer 액세스 권한을 부여하는 정책
2. CloudWatch Application Insights에 필요한 인스턴스 역할 및 정책을 연결합니다. 지침은 상용 소프트웨어 라이선스 권장 사항을 활성화하는 정책을 참조하세요.
3. Microsoft SQL Server 데이터베이스 자격 증명을 사용하여 CloudWatch Application Insights를 활성화합니다. 지침은 CloudWatch 설명서의 모니터링을 위한 애플리케이션 설정을 참조하세요.

   > **Note**
   >
   > 상용 소프트웨어 라이선스에 대한 권장 사항을 생성하려면 최소 30시간의 연속 CloudWatch 지표 데이터가 필요합니다. 자세한 내용은 CloudWatch 지표 요구 사항을 참조하세요.

4. 다음 SQL 쿼리를 사용하여 CloudWatch Application Insights에 대한 최소 권한 액세스를 구성합니다.

   ```
   GRANT VIEW SERVER STATE TO [LOGIN];
   GRANT VIEW ANY DEFINITION TO [LOGIN];
   ```

   이렇게 하면 새 서비스인 PrometheusSqlExporterSQL이 활성화됩니다.
5. 대상 AWS 계정 또는 조직 관리 계정에서 Compute Optimizer를 선택합니다. 지침은 계정에서 옵트인을 참조하세요.

> **Note**
>
> 옵트인하고 나면 결과 및 최적화 권장 사항이 생성되는 데 최대 24시간이 걸릴 수 있습니다.

6. Compute Optimizer 콘솔의 탐색 창에서 라이선스를 선택합니다.
7. 조사 결과 열에서 지표 조사 결과가 충분하지 않은 인스턴스를 검색합니다. Compute Optimizer는 CloudWatch Application Insights가 활성화되지 않았거나 권한이 충분하지 않은 것을 감지하면이 결과를 반환합니다. 자세한 내용은 결과 이유를 참조하세요. 다음을 수행하여 이러한 결과를 해결합니다.
   a. 인스턴스를 선택합니다.
   b. 보안 암호를 추가합니다.
   c. 인스턴스 역할과 정책이 연결되어 있는지 확인합니다.
   d. 라이선스 권장 사항 활성화를 선택합니다.
8. 조사 결과 열에서 최적화되지 않은 조사 결과가 있는 인스턴스를 검색합니다. Compute Optimizer는 Amazon EC2 인프라가 사용자가 비용을 지불하는 Microsoft SQL Server 라이선스 기능을 사용하지 않는 것을 감지하면이 결과를 반환합니다. 자세한 내용은 결과 이유를 참조하세요. 다음을 수행하여 이러한 결과를 해결합니다.
   a. 인스턴스를 선택합니다.
   b. 현재 라이선스 에디션을 권장 에디션과 비교합니다.
   c. 현재 라이선스 사용률 그래프를 검토합니다.
   d. 라이선스를 다운그레이드하려면 권장 사항 구현을 선택합니다.
   e. 요구 사항을 검토하고 지침에 따라 라이선스를 다운그레이드합니다. 프로세스를 자동화하려면 문서를 사용하여 AWS Systems Manager SQL Server 엔터프라이즈 에디션 다운그레이드를 참조하여 비용 절감(AWS 블로그)을 참조하세요.

추가 리소스

- 를 사용하여 Microsoft SQL Server 라이선스 비용 절감 AWS Compute Optimizer(AWS 블로그)
- 란 무엇입니까 AWS Compute Optimizer? (AWS 문서)
- 상용 소프트웨어 라이선스 권장 사항 보기(AWS 설명서)
- Microsoft SQL Server 에디션 다운그레이드(AWS 설명서)
- (AWS)의 Microsoft SQL Server AWS

- Microsoft Licensing on AWS (AWS)
- Microsoft SQL Server 2019 요금(Microsoft)
- Microsoft SQL Server 2022 요금(Microsoft)

Compute Optimizer를 사용하여 SQL Server 크기 최적화

개요

AWS Compute Optimizer는 데이터베이스 관리자(DBAs)가 Amazon Elastic Compute Cloud(Amazon EC2)에서 Microsoft SQL Server 워크로드를 검색하고 EC2 인스턴스 크기를 조정하여 라이선스 비용을 최대 25% 절감할 수 있도록 지원합니다. Compute Optimizer의 추론된 워크로드 유형 기능은 기계 학습(ML)을 사용하고 AWS 리소스에서 실행 중일 수 있는 애플리케이션을 자동으로 감지합니다. Compute Optimizer에는 SQL Server에 대한 지원이 추론된 워크로드 유형으로 포함되어 있습니다. 추론된 워크로드 유형 기능을 사용하면 Amazon EC2 인스턴스에서 실행되는 특정 워크로드를 기반으로 비용 절감 기회를 정확히 파악할 수 있습니다.

이 기능을 사용하면 SQL Server와 같이 지원되는 추론된 워크로드 유형별로 비용 절감 기회를 분류할 수 있습니다. Compute Optimizer는 과다 프로비저닝된 SQL Server EC2 인스턴스를 자동으로 검색할 수 있습니다. EC2 콘솔로 전환하여 인스턴스의 크기를 줄여 라이선스 및 인프라 비용을 줄일 수 있습니다.

Compute Optimizer를 사용하여 SQL Server 라이선스 권장 사항을 만들 수도 있습니다. 자세한 내용은 이 설명서의 Compute Optimizer를 사용하여 SQL Server 라이선스 최적화를 참조하세요.

Compute Optimizer 구성

SQL Server 추론 워크로드에서 Compute Optimizer를 사용하는 방법에 대한 지침은 성능 최적화 및 라이선스 비용 절감: Amazon EC2 SQL Server 인스턴스 활용(블로그)을 참조 AWS Compute Optimizer하세요.AWS 독립 실행형 계정, 조직의 멤버인 계정 및 조직의 관리 계정에 옵트인할 수 있습니다. 독립 실행형 및 멤버 계정의 경우 옵트인하면 해당 계정에 대해서만 Compute Optimizer가 활성화됩니다. 조직 관리 계정의 경우 해당 계정에서만 Compute Optimizer를 활성화할지 아니면 조직의 모든 멤버 계정에 대해 Compute Optimizer를 활성화할지 선택할 수 있습니다.

Compute Optimizer 옵트인 프로세스는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 자동으로 생성합니다. 자세한 내용은 AWS Compute Optimizer에 대한 서비스 연결 역할 사용을 참조하세요.

Compute Optimizer는 CPU, I/O, 네트워크 및 Amazon Elastic Block Store(Amazon EBS) 사용량과 같은 Amazon CloudWatch 지표를 기반으로 리소스를 분석합니다. 권장 사항을 생성하려면 지난 14일 동안 연속 30시간 이상의 CloudWatch 지표 데이터가 필요합니다. 향상된 인프라 지표 기능을 활성화하면 사용률 지표가 93일로 확장됩니다. 자세한 내용은 Compute Optimizer 설명서의 CloudWatch 지표 요구 사항 및 향상된 인프라 지표를 참조하세요.

Compute Optimizer는 vCPU, 메모리, 스토리지, 네트워크, 위험 및 마이그레이션 노력을 기반으로 각 옵션과 관련된 옵션과 비용 절감을 제공합니다. CloudWatch 지표 대시보드를 사용하여 권장 사항에 사용되는 데이터를 분석할 수 있습니다. 이 데이터를 사용하면 SQL Server 워크로드를 실행하는 EC2 인스턴스의 크기를 조정할 수 있습니다. 인스턴스 유형을 변경하는 방법에 대한 자세한 내용은 Amazon EC2 설명서의 인스턴스 유형 변경을 참조하세요.

추가 리소스

- AWS Compute Optimizer Microsoft SQL Server 워크로드 식별 및 필터링(AWS)
- 성능 최적화 및 라이선스 비용 절감: Amazon EC2 SQL Server 인스턴스 AWS Compute Optimizer 활용(AWS 블로그)
- 란 무엇입니까 AWS Compute Optimizer? (AWS 문서)
- EC2 인스턴스 권장 사항 보기(AWS 문서)

SQL Server 워크로드에 대한 Trusted Advisor 권장 사항 검토

개요

AWS Trusted Advisor는 모범 사례를 따르는 AWS 데 도움이 되는 권장 사항을 제공합니다. 사용량, 구성 및 지출을 분석하여 비용을 절감하거나, 시스템 가용성 및 성능을 개선하거나, 보안 격차를 줄이는 데 도움이 되는 실행 가능한 권장 사항을 Trusted Advisor 제공합니다. 이 섹션에서는에서 SQL Server 워크로드 운영 비용을 줄이는 데 도움이 되는 Trusted Advisor 검사에 중점을 둡니다 AWS 클라우드.

비용 최적화 권장 사항

Trusted Advisor 는 Amazon Elastic Compute Cloud(Amazon EC2)에서 SQL Server 워크로드를 최적화하는 데 도움이 되는 권장 사항을 제공합니다. 검사는 SQL Server 워크로드를 검사하고 최적화가 필요한 인스턴스를 자동으로 나열합니다. Trusted Advisor 권장 사항을 운영하면 비용을 절감하고 조직의 보안 태세를 개선할 수 있습니다.

다음은 Microsoft SQL Server에 초점을 맞춘 Trusted Advisor 검사입니다.

- Microsoft SQL Server용으로 과다 프로비저닝된 Amazon EC2 인스턴스 -이 검사는 SQL Server를 실행 중인 Amazon EC2 인스턴스를 분석하고 인스턴스가 SQL Server 소프트웨어 vCPU 제한을 초과하는 경우 알려줍니다. 예를 들어 SQL Server Standard 에디션이 있는 인스턴스는 최대 48개의 vCPUs. SQL Server Web을 포함하는 인스턴스는 최대 32개의 vCPU를 사용할 수 있습니다.

Edition	vCPU 최소	vCPU 최대
웹	4	32
표준	4	48
엔터프라이즈	4	OS 제한

- Microsoft SQL Server용 Amazon EC2 인스턴스 통합 -이 검사는 Amazon EC2 인스턴스를 분석하고 인스턴스에 최소 SQL Server 라이선스 수 미만인 경우 알려줍니다. 작은 SQL Server 인스턴스를 통합하면 비용을 절감할 수 있습니다. 라이선스에 포함된 작은 SQL Server 인스턴스가 많은 경우 통합을 고려하세요. Microsoft SQL Server 2019 라이선스 가이드에 따라 SQL Server에는 인스턴스당 최소 4개의 vCPU 라이선스가 필요합니다. 이러한 데이터베이스를 통합하면 라이선스 비용을 절감할 수 있습니다. 인스턴스의 데이터베이스 수, 최대 데이터베이스 크기, 데이터베이스의 총 크기에 따라 결정을 내릴 수 있습니다. 통합은 SQL Server의 웹, 표준 및 엔터프라이즈 에디션에서 지원됩니다. 자세한 내용은 SQL Server 데이터베이스 통합(Microsoft 블로그 게시물)을 참조하세요.

  AWS 는 하나의 서버에만 대규모 프로덕션 데이터베이스를 배치하는 것을 권장하지 않습니다. 그러나 개발, 테스트 및 스테이징과 같은 비프로덕션 환경에 사용되는 더 작은 것을 통합할 수 있습니다. 이는 현재 SQL Server 사용량에 따라 달라집니다. 사용량이 적은 데이터베이스가 있는 경우 하나의 서버에 통합할 수 있습니다.

Trusted Advisor구성

다음을 수행하여 SQL Server 집중 검사를 평가합니다 Trusted Advisor.

1. AWS Management Console에 로그인합니다.
2. AWS Trusted Advisor 콘솔을 엽니다.
3. 탐색 창의 권장 사항에서 비용 최적화를 선택합니다.
4. 비용 최적화 검사 목록에서 Microsoft SQL Server용 Amazon EC2 인스턴스 통합 및 Microsoft SQL Server 검사용으로 과다 프로비저닝된 Amazon EC2 인스턴스의 상태를 검토합니다.
   - 녹색 확인 기호는 Amazon EC2 인스턴스가 최적으로 구성되었음을 나타냅니다.

- 주황색 알림 기호는 개선 기회가 있음을 나타냅니다.

5. 검사를 선택하여 세부 정보 및 권장 사항을 확인합니다.
6. 점검에서 제공하는 지침에 따라 SQL Server 워크로드를 실행하는 Amazon EC2 인스턴스를 최적화합니다.
7. 인스턴스를 정기적으로 모니터링하고 검사를 주기적으로 새로 고칩니다.

추가 리소스

- Trusted Advisor 참조 확인(AWS 문서)
- (AWS)의 Microsoft SQL Server AWS
- Microsoft Licensing on AWS (AWS)
- SQL Server 2019 요금(Microsoft)
- AWS Launch Wizard SQL Server용(AWS 문서)

컨테이너

현대화는 모놀리스를 마이크로서비스로 분해하고, 서버리스 함수(AWS Lambda)를 사용하여 이벤트 기반 애플리케이션을 다시 설계하고, SQL Server에서 Amazon Aurora 또는 특별히 구축된 관리형 데이터베이스로 데이터베이스를 용도 변경하는 등 다양한 옵션을 제공하는 혁신적인 여정입니다. .NET Framework 애플리케이션을 Linux 및 Windows 컨테이너로 리플랫포밍하는 현대화 경로는 다른 현대화 옵션보다 적은 노력이 필요합니다. 컨테이너는 다음과 같은 이점을 제공합니다.

- 혁신 가속화 - 컨테이너로 이동하면 애플리케이션 구축, 테스트 및 배포를 포함한 개발 수명 주기의 단계를 더 쉽게 자동화할 수 있습니다. 이러한 프로세스를 자동화하면 개발 및 운영 팀이 혁신에 더 많은 시간을 할애할 수 있습니다.
- 총 소유 비용(TCO) 절감 - 컨테이너로 전환하면 라이선스 관리 및 엔드포인트 보호 도구에 대한 의존도 줄일 수 있습니다. 컨테이너는 임시 컴퓨팅 단위이므로 패치 적용, 조정, 백업 및 복원과 같은 관리 작업을 자동화하고 간소화할 수 있습니다. 이렇게 하면 컨테이너 기반 워크로드 관리 및 운영의 TCO가 줄어듭니다. 마지막으로 컨테이너는 가상 머신에 비해 더 효율적입니다. 컨테이너를 사용하여 더 나은 격리를 제공하여 애플리케이션 배치를 극대화할 수 있기 때문입니다. 이렇게 하면 애플리케이션의 인프라 리소스 사용률이 증가합니다.
- 리소스 사용률 개선 - 컨테이너를 사용하여 애플리케이션 배치를 극대화할 수 있으므로 컨테이너는 가상 머신에 비해 더 효율적입니다. 이렇게 하면 더 나은 격리를 제공하여 애플리케이션 인프라 리소스의 활용도가 높아집니다.
- 기술 격차 해소 -는 컨테이너 기술 및 DevOps 관행에 대한 개발 팀의 기술을 강화할 수 있는 몰입의 날을 AWS 제공합니다.

이 섹션은 다음 주제를 포함합니다.

- Windows 애플리케이션을 컨테이너로 이동
- Amazon ECS에서 AWS Fargate 작업 비용 최적화
- Amazon EKS 비용에 대한 가시성 확보
- App2Container를 사용하여 Windows 애플리케이션 리플랫폼

라이선스 정보는 Amazon Web Services 및 Microsoft: 자주 묻는 질문의 라이선스 섹션을 참조하거나 microsoft@amazon.com으로 질문을 이메일로 보내세요.

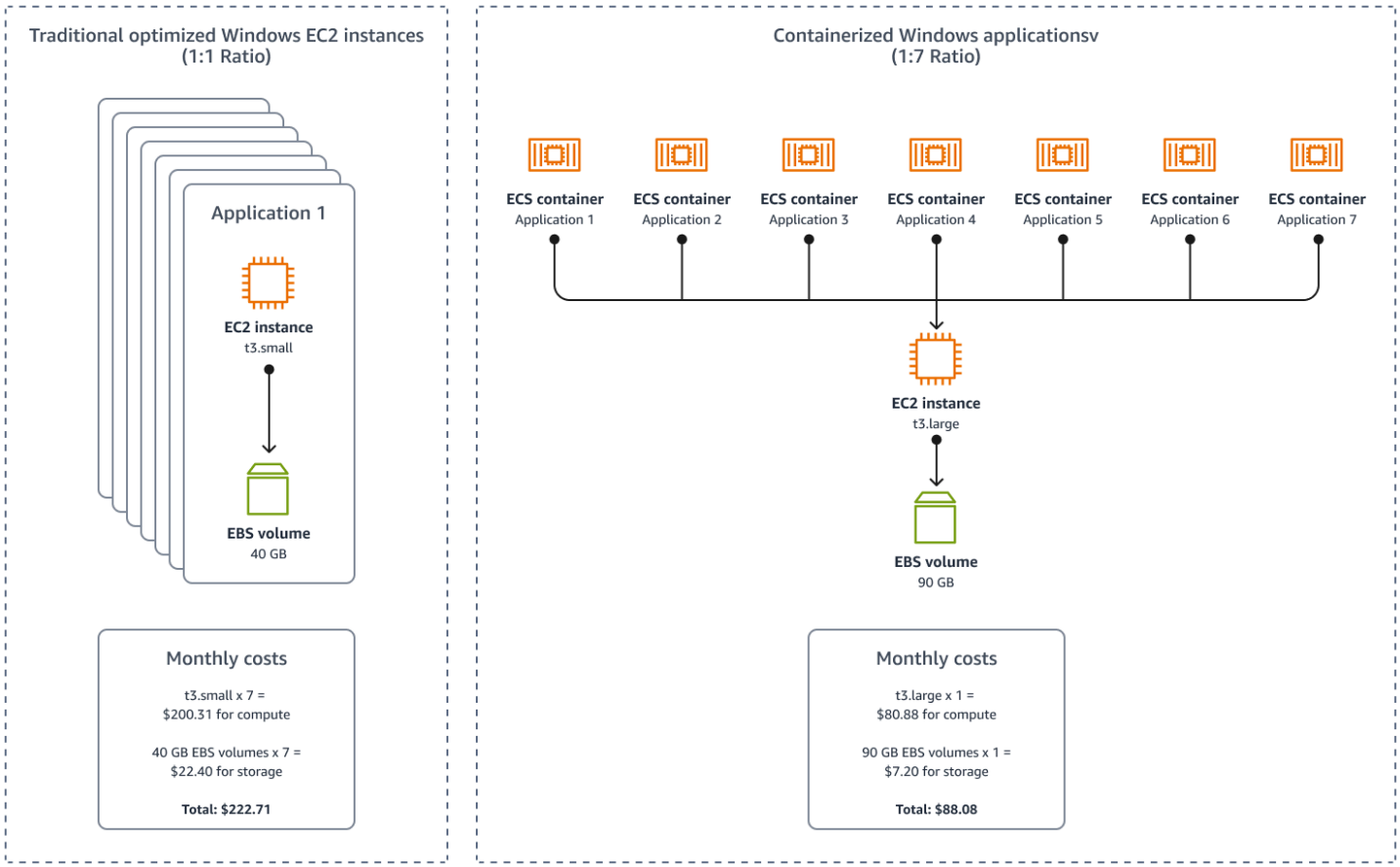
Windows 애플리케이션을 컨테이너로 이동

개요

2021년 CNCF 연간 설문 조사에 따르면 조직의 96%가 컨테이너를 사용하거나 평가하여 인프라를 현대화하고 있습니다. 이는 컨테이너가 조직의 위험을 줄이고 운영 효율성과 속도를 높이며 민첩성을 활성화하는 데 도움이 될 수 있기 때문입니다. 컨테이너를 사용하여 애플리케이션 실행 비용을 줄일 수도 있습니다. 이 섹션에서는 Amazon Elastic Container Service(Amazon ECS), Amazon Elastic Kubernetes Service(Amazon EKS) 및를 포함한 AWS 컨테이너 서비스에서 컨테이너를 비용 효율적으로 실행하는 방법에 대한 권장 사항을 제공합니다AWS Fargate.

비용 이점

다음 인포그래픽은 AWS 최적화 및 라이선스 평가(AWS OLA) 권장 사항을 기반으로 ASP.NET 프레임워크 애플리케이션을 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 통합하여 기업이 달성할 수 있는 비용 절감 효과를 보여줍니다. 다음 인포그래픽은 애플리케이션을 Windows 컨테이너로 이동하여 얻을 수 있는 추가 비용 절감 효과를 보여줍니다.



AWS OLA는 사업부가 개별 t3.small 인스턴스로 리프트 앤 시프트를 수행할 것을 권장했습니다. 비즈니스는 다음 성능 사용률 분석에서 알 수 있듯이 온프레미스 서버에서 7개의 ASP.NET 애플리케이션을 실행하여 이러한 비용 절감을 달성할 수 있습니다.

		Not sharable	Nearing end of life	Approximately 15% overhead		The OS accounts for 1.2 GB per instance			Optimized for EC2		
	Server name	Storage	Operating system	On-premises CPU AVG utilization	On-premises CPU peak utilization	On-premises RAM (GB)	On-premises RAM AVG utilization (GB)	On-premises RAM peak utilization (GB)	Instance size	vCPU	RAM (GB)
1	AppServer01	60	Windows Server 2012	7.00%	17.00%	8	13.50%	17.10%	t3.small	2	2
2	AppServer02	39	Windows Server 2012	20.07%	22.00%	16	7.50%	12.40%	t3.small	2	2
3	AppServer03	39	Windows Server 2012	24.00%	25.50%	16	8.80%	11.90%	t3.small	2	2
4	AppServer04	4	Windows Server 2012	21.40%	24.00%	16	7.80%	10.70%	t3.small	2	2
5	AppServer05	40	Windows Server 2012	21.30%	23.00%	16	8.20%	12.00%	t3.small	2	2
6	AppServer06	39	Windows Server 2012	21.50%	23.50%	16	7.90%	10.90%	t3.small	2	2
7	AppServer07	39	Windows Server 2012	21.60%	22.90%	16	8.40%	11.50%	t3.small	2	2

추가 분석에서는 컨테이너에서 워크로드를 실행하여 비즈니스 비용을 더 많이 절감할 수 있는 것으로 나타났습니다. 컨테이너는 CPU, RAM 및 디스크 사용량과 같은 시스템 리소스의 운영 체제 오버헤드를 줄입니다(다음 단원에서 설명). 이 시나리오에서 비즈니스는 7개의 애플리케이션을 모두 하나의 t3.large 인스턴스로 통합할 수 있으며 여전히 3GB RAM을 절약할 수 있습니다. 컨테이너로 마이그레이션하면 Amazon EC2 대신 컨테이너를 사용하여 컴퓨팅 및 스토리지 전반에서 평균 64%의 비용 절감을 달성할 수 있습니다.

비용 최적화 권장 사항

다음 섹션에서는 애플리케이션을 통합하고 컨테이너를 사용하여 비용을 최적화하기 위한 권장 사항을 제공합니다.

Amazon EC2의 Windows 설치 공간 축소

Windows 컨테이너를 사용하면 더 많은 애플리케이션을 더 적은 수의 Amazon EC2 인스턴스로 통합할 수 있으므로 Amazon EC2 기반 Windows 설치 공간을 줄일 수 있습니다. 예를 들어 ASP.NET 애플리케이션이 500개 있다고 가정해 보겠습니다. Amazon EC2에서 Windows용 애플리케이션 하나당 코어 하나를 실행하는 경우 이는 500개의 Windows 인스턴스(t3.small)와 같습니다. Windows 컨테이너(t3.large 포함)를 사용하기 위해 1:7 비율(EC2 인스턴스 유형/크기에 따라 크게 증가할 수 있음)을 가정하면 약 71개의 Windows 인스턴스만 필요합니다. 이는 Amazon EC2 풋프린트에서 Windows가 85.8% 감소했음을 나타냅니다.

Windows 라이선스 비용 절감

Windows 인스턴스에 라이선스를 부여하는 경우 해당 인스턴스에서 실행되는 컨테이너에 라이선스를 부여할 필요가 없습니다. 따라서 Windows 컨테이너를 사용하여 ASP.NET 애플리케이션을 통합하면 Windows 라이선스 비용을 크게 줄일 수 있습니다.

스토리지 공간 감소

새 EC2 인스턴스를 시작할 때마다 운영 체제를 수용할 새 Amazon Elastic Block Store(Amazon EBS) 볼륨을 생성하고 비용을 지불합니다. 이렇게 조정하면 비용이 그에 따라 조정됩니다. 컨테이너를 사용하는 경우 모든 컨테이너가 동일한 기본 운영 체제를 공유하므로 스토리지 비용을 절감할 수 있습니다. 또한 컨테이너는 계층 개념을 사용하여 해당 이미지를 기반으로 실행 중인 모든 컨테이너에 대해 컨테이너 이미지의 변경할 수 없는 부분을 재사용합니다. 앞의 예제 시나리오에서 모든 컨테이너는 .NET Framework를 실행하므로 모두 중간 및 변경 불가능한 ASP.NET 프레임워크 계층을 공유합니다.

end-of-support 서버를 컨테이너로 마이그레이션

Windows Server 2012 및 Windows Server 2012 R2에 대한 지원은 2023년 10월 10일에 종료되었습니다. Windows Server 2012 또는 이전 버전에서 실행되는 애플리케이션을 컨테이너화하여 새 운영 체제에서 실행되도록 마이그레이션할 수 있습니다. 이렇게 하면 컨테이너가 제공하는 비용 효율성, 위험 감소, 운영 효율성, 속도 및 민첩성을 활용하면서 규정을 준수하지 않는 운영 체제에서 애플리케이션을 실행하지 않아도 됩니다.

이 접근 방식에서 고려해야 할 주의 사항은 애플리케이션에 현재 사용 중인 운영 체제 버전과 관련된 특정 APIs가 필요한 경우입니다(예: COM Interop). 이 경우 애플리케이션을 최신 Windows 버전으로 이동하는 방법을 테스트해야 합니다. Windows 컨테이너는 기본 컨테이너 이미지(예: Windows Server 2019)를 컨테이너 호스트의 운영 체제(예: Windows Server 2019)와 정렬합니다. 테스트하고 컨테이너로 이동하면 Dockerfile 내의 기본 이미지를 변경하고 최신 버전의 Windows를 실행하는 새로운 호스트 세트에 배포하여 향후 운영 체제 업그레이드를 더 쉽게 할 수 있습니다.

타사 관리 도구 및 라이선스 제거

서버 플릿을 관리하려면 패치 및 구성 관리를 위해 여러 타사 시스템 작업 도구를 사용해야 합니다. 이로 인해 인프라 관리가 복잡해지고 타사 라이선스 비용이 발생하는 경우가 많습니다. 에서 컨테이너를 사용하는 경우 운영 체제 측에서 아무것도 관리할 필요가 AWS없습니다. 컨테이너 런타임은 컨테이너를 관리합니다. 즉, 기본 호스트는 임시 호스트이며 쉽게 교체할 수 있습니다. 컨테이너 호스트를 직접 관리할 필요 없이 컨테이너를 실행할 수 있습니다. 또한 AWS Systems Manager Session Manager 와 같은 무료 도구를 사용하여 호스트에 쉽게 액세스하고 문제를 해결할 수 있습니다.

제어 및 이식성 개선

컨테이너를 사용하면 EC2 인스턴스보다 CPU 및 RAM과 같은 서버 리소스를 더 세밀하게 제어할 수 있습니다. EC2 인스턴스의 경우 인스턴스 패밀리, 인스턴스 유형 및 CPU 옵션을 선택하여 CPU 및 RAM을 제어할 수 있습니다. 그러나 컨테이너를 사용하면 ECS 작업 정의의 컨테이너 또는 Amazon EKS의 포드에 할당할 CPU 또는 RAM의 양을 정확히 정의할 수 있습니다. 실제로 Windows 컨테이너에 대한 컨테이너 수준 CPU 및 메모리를 지정하는 것이 좋습니다. 이러한 수준의 세분화는 비용 이점을 제공합니다. 다음 예제 코드를 고려하세요.

```
json
{
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/demo-
service:1",
    "containerDefinitions": [
        {
            "name": "demo-service",
            "image": "mcr.microsoft.com/dotnet/framework/samples:aspnetapp-
windowsservercore-ltsc2019",
            "cpu": 512,
            "memory": 512,
            "links": [],
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
```

혁신 가속화

컨테이너로 이동하면 애플리케이션 구축, 테스트 및 배포를 포함한 개발 수명 주기의 단계를 더 쉽게 자동화할 수 있습니다. 이러한 프로세스를 자동화하면 개발 및 운영 팀이 혁신에 집중할 수 있는 더 많은 시간을 확보할 수 있습니다.

TCO 절감

컨테이너로 전환하면 라이선스 관리 및 엔드포인트 보호 도구에 대한 의존도가 감소하는 경우가 많습니다. 컨테이너는 임시 컴퓨팅 단위이므로 패치 적용, 조정, 백업 및 복원과 같은 관리 작업을 자동화하고 간소화할 수 있습니다. 이렇게 하면 컨테이너 기반 워크로드 관리 및 운영의 TCO를 줄일 수 있습니

다. 컨테이너는 애플리케이션 배치를 극대화하여 애플리케이션의 인프라 리소스 사용률을 높일 수 있으므로 가상 머신에 비해 더 효율적입니다.

기술 격차 해소

AWS 는 컨테이너 및 DevOps 기술에 대한 고객 개발 팀의 역량을 높일 수 있는 프로그램과 몰입의 날을 제공합니다. 여기에는 실습 컨설팅 및 활성화가 포함됩니다.

.NET 5+로 리팩터링 및 Linux 컨테이너 사용

.NET Framework 애플리케이션을 컨테이너로 이동하여 비용을 절감할 수 있지만, 레거시 .NET 애플리케이션을 클라우드 네이티브 대안으로 리팩터링하면 비용을 더욱 절감할 수 있습니다 AWS.

라이선스 비용 제거

애플리케이션을 Windows의 .NET Framework에서 Linux의 .NET Core로 리팩터링하면 약 45%의 비용 절감 효과가 발생합니다.

최신 개선 사항 액세스

애플리케이션을 Windows의 .NET Framework에서 Linux의 .NET Core로 리팩터링하면 Graviton2와 같은 최신 개선 사항에 액세스할 수 있습니다. Graviton2는 유사한 인스턴스에 비해 40% 더 나은 성능 가격을 제공합니다.

보안 및 성능 개선

애플리케이션을 Windows의 .NET Framework에서 Linux 컨테이너의 .NET Core로 리팩터링하면 보안 및 성능이 향상됩니다. 이는 최신 보안 패치를 받고, 컨테이너 격리의 이점을 누리고, 새로운 기능에 액세스할 수 있기 때문입니다.

IIS의 한 인스턴스에서 많은 애플리케이션을 실행하는 대신 Windows 컨테이너 사용

인터넷 정보 서비스(IIS)가 있는 하나의 EC2 Windows 인스턴스에서 여러 애플리케이션을 실행하는 대신 Windows 컨테이너를 사용하는 경우 다음과 같은 이점이 있습니다.

- 보안 - 컨테이너는 IIS 수준에서 격리를 통해 달성되지 않는 즉시 사용 가능한 보안 수준을 제공합니다. 한 IIS 웹 사이트 또는 애플리케이션이 손상되면 다른 모든 호스팅 사이트가 노출되고 취약해집니다. 컨테이너 이스케이프는 드물며 웹 취약성을 통해 서버를 제어하는 것보다 악용하기가 더 어렵습니다.
- 유연성 - 프로세스 격리 상태에서 컨테이너를 실행하고 자체 인스턴스를 보유하는 기능을 통해 보다 세분화된 네트워킹 옵션을 사용할 수 있습니다. 또한 컨테이너는 많은 EC2 인스턴스에서 복잡한 배

포 방법을 제공합니다. 단일 IIS 인스턴스에서 애플리케이션을 통합할 때 이러한 이점을 얻을 수 없습니다.

- 관리 오버헤드 - 서버 이름 표시(SNI)는 관리 및 자동화가 필요한 오버헤드를 생성합니다. 또한 패치 적용, BSOD 문제 해결(자동 조정이 없는 경우), 엔드포인트 보호 등과 같은 일반적인 운영 체제 관리 작업을 수행해야 합니다. 보안 모범 사례에 따라 IIS 사이트를 구성하는 것은 시간이 많이 걸리고 지속적인 활동입니다. 신뢰 수준을 설정해야 할 수도 있으며, 이는 관리 오버헤드에도 추가됩니다. 컨테이너는 상태 비저장 및 변경 불가능하도록 설계되었습니다. 궁극적으로 Windows 컨테이너를 대신 사용하면 배포가 더 빠르고 안전하며 반복 가능합니다.

다음 단계

레거시 워크로드를 실행하기 위해 최신 인프라에 투자하면 조직에 엄청난 이점이 있습니다. AWS 컨테이너 서비스를 사용하면 온프레미스에서든 클라우드에서든 기본 인프라를 더 쉽게 관리할 수 있으므로 혁신과 비즈니스 요구 사항에 집중할 수 있습니다. 클라우드의 모든 컨테이너 중 거의 80%가 AWS 현재에서 실행됩니다.는 거의 모든 사용 사례에 대해 풍부한 컨테이너 서비스 세트를 AWS 제공합니다. 시작하려면 의 컨테이너를 참조하세요 AWS.

추가 리소스

- ECS 용량 공급자 및 EC2 스팟 인스턴스를 사용하여 컨테이너 워크로드 비용 최적화(AWS 블로그)
- Amazon ECS 및 (블로그)에 대한 비용 최적화 체크리스트 AWS FargateAWS
- Amazon EKS on AWS Graviton2 정식 출시: 다중 아키텍처 앱 고려 사항(AWS 블로그)
- 의 Kubernetes에 대한 비용 최적화 AWS(AWS 블로그)
- Karpenter 통합을 통한 Kubernetes 컴퓨팅 비용 최적화(AWS 블로그)

Amazon ECS에서 AWS Fargate 작업 비용 최적화

개요

올바른 크기 조정 AWS Fargate 작업은 비용 최적화에 중요한 단계입니다. 애플리케이션은 Fargate 작업에 대한 임의 크기 조정으로 빌드되는 경우가 많으며 다시 검토하지 않습니다. 이로 인해 Fargate 작업이 과다 프로비저닝되고 불필요한 지출이 발생할 수 있습니다. 이 섹션에서는를 사용하여 Fargate에서 실행되는 Amazon Elastic Container Service(Amazon ECS) 서비스에 대한 작업 CPU 및 메모리를 최적화AWS Compute Optimizer할 수 있도록 실행 가능한 권장 사항을 제공하는 방법을 보여줍니다. Compute Optimizer는 이러한 권장 사항 채택으로 인한 비용 영향도 정량화합니다. 이를 통해 절감 기

회의 크기에 따라 최적화 작업의 우선순위를 지정할 수 있습니다. Compute Optimizer 권장 사항은 태스크 크기를 줄이기 위한 컨테이너 수준 CPU 및 메모리 구성을 제공합니다.

비용 이점

Fargate에서 Amazon ECS 작업의 크기를 올바르게 조정하면 장기 실행 작업의 비용을 30~70% 절감할 수 있습니다. 애플리케이션 성능 지표를 검토하여 작업 크기를 조정하지 않고 EC2 컴퓨팅 인스턴스에 사용된 것과 동일한 사고 방식을 컨테이너 크기 조정에 적용할 수 있습니다. 이로 인해 Fargate 작업이 너무 커서 유휴 리소스에 대한 비용이 증가합니다. Compute Optimizer를 사용하여 적절한 크기 조정 기회를 사후적으로 표시할 수 있습니다. 애플리케이션 소유자는 특정 애플리케이션 성능 지표를 검토하고 운영 체제 오버헤드를 제거하여 적절한 작업 크기가 지정되도록 하는 것이 가장 좋습니다. 자세한 내용은이 가이드의 Windows 애플리케이션을 컨테이너로 이동 섹션을 참조하세요.

비용 최적화 권장 사항

이 단원에서는 Fargate 작업에서 Compute Optimizer를 사용하여 Amazon ECS의 크기를 올바르게 조정하기 위한 권장 사항을 제공합니다.

비용 최적화 프로세스의 일환으로 다음을 수행하는 것이 좋습니다.

- Compute Optimizer 활성화
- Compute Optimizer 결과 사용
- 적절한 크기로 태스크에 태그 지정
- 비용 할당 태그가 AWS 결제 도구에서 작동하도록 활성화
- 올바른 크기 조정 권장 사항 구현
- Cost Explorer에서 비용 전후 검토

Compute Optimizer 활성화

AWS Compute Optimizer의 조직 또는 단일 계정 수준에서를 활성화할 수 있습니다 AWS Organizations. 조직 전체 구성은 모든 멤버 계정의 전체 플릿에서 신규 및 기존 인스턴스에 대한 지속적인 보고서를 제공합니다. 이를 통해 올바른 크기 조정은 point-in-time 활동으로 사용할 수 있습니다.

조직 수준

대부분의 조직에서 Compute Optimizer를 사용하는 가장 효율적인 방법은 조직 수준에서입니다. 이를 통해 조직에 대한 다중 계정 및 다중 리전 가시성을 제공하고 검토를 위해 데이터를 하나의 소스로 중앙 집중화할 수 있습니다. 조직 수준에서 이를 활성화하려면 다음을 수행합니다.

1. 필요한 권한이 있는 역할로 AWS Organizations 관리 계정에 로그인하고이 조직 내의 모든 계정에 대해 옵트인하도록 선택합니다. 조직의 모든 기능을 활성화해야 합니다.
2. 관리 계정을 활성화한 후 계정에 로그인하고, 다른 모든 멤버 계정을 확인하고, 추천 사항을 찾아볼 수 있습니다.

> **Note**
>
> Compute Optimizer에 대한 위임된 관리자 계정을 구성하는 것이 가장 좋습니다. 이렇게 하면 최소 권한 원칙을 적용하여 AWS Organizations 관리 계정에 대한 액세스를 최소화하는 동시에 조직 전체 서비스에 대한 액세스를 제공할 수 있습니다.

단일 계정 수준

비용이 높지만에 액세스할 수 없는 계정을 대상으로 하는 경우에도 해당 계정 및 리전에 대해 Compute Optimizer를 활성화 AWS Organizations할 수 있습니다. 옵트인 프로세스에 대한 자세한 내용은 시작하기를 참조하세요 AWS Compute Optimizer.

> **Note**
>
> 권장 사항은 매일 업데이트되며 생성하는 데 최대 12시간이 걸릴 수 있습니다. Compute Optimizer에서 Fargate의 Amazon ECS에 대한 권장 사항을 생성하려면 지난 14일 동안 24시간의 지표가 필요하다는 점에 유의하세요. 자세한 내용은 Compute Optimizer 설명서의 Fargate의 Amazon ECS 서비스 요구 사항을 참조하세요.

Compute Optimizer는 Fargate의 Amazon ECS 서비스에 대해 다음과 같은 Amazon CloudWatch 및 Amazon ECS 사용률 지표를 자동으로 분석합니다.

- `CPUUtilization` - 서비스에 사용되는 CPU 용량의 백분율입니다.
- `MemoryUtilization` - 서비스에 사용되는 메모리의 백분율입니다.

Compute Optimizer 결과 사용

단일 계정 및 단일 리전 내에서 올바른 크기 조정을 변경하는 데 초점을 맞춘 예를 생각해 보세요. 이 예제에서는 Compute Optimizer가 모든 계정의 조직 수준에서 활성화됩니다. 올바른 크기 조정은 대부분

의 경우 몇 주에 걸쳐 예약된 유지 관리 기간 동안 애플리케이션 소유자가 정밀도로 수행하는 파괴적인 프로세스입니다.

조직의 관리 계정 내에서 Compute Optimizer로 이동하는 경우(다음 단계에 표시됨) 조사하려는 계정을 선택할 수 있습니다. 이 예제에서는 한 작업이에서 과다 프로비저닝된 단일 계정에서 실행됩니다`us-east-1`. Amazon ECS 서비스의 권장 크기로 크기를 조정하는 데 중점을 둡니다.

1. Compute Optimizer 콘솔을 엽니다.
2. 대시보드 페이지에서 Findings=Over-provisioned로 필터링하여 Fargate의 모든 Amazon ECS 서비스를 확인합니다.
3. Fargate에서 과다 프로비저닝된 ECS 서비스에 대한 자세한 권장 사항을 검토하려면 아래로 스크롤한 다음 권장 사항 보기를 선택합니다.
4. 내보내기를 선택하고 나중에 사용할 수 있도록 파일을 저장합니다.

> **Note**
>
> 향후 검토를 위해 권장 사항을 저장하려면 Compute Optimizer가 각 리전에서에 쓸 수 있는 S3 버킷이 있어야 합니다. 자세한 내용은 Compute Optimizer 설명서의에 대한 Amazon S3 버킷 정책을 AWS Compute Optimizer 참조하세요.

Compute Optimizer의 권장 사항을 보려면 다음을 수행합니다.

1. Compute Optimizer 콘솔에서 권장 사항 내보내기 페이지로 이동합니다.
2. S3 버킷 대상에서 S3 버킷을 선택합니다.
3. 필터 내보내기 섹션의 리소스 유형에서 Fargate의 ECS 서비스를 선택합니다.
4. Fargate의 ECS 서비스에 대한 권장 사항 페이지에서 Fargate의 ECS 서비스 중 하나를 드릴하고 Compute Optimizer의 CPU 및 메모리 권장 사항을 확인합니다. 예를 들어 현재 설정과 권장 작업 크기 비교 및 현재 설정과 권장 컨테이너 크기 비교 섹션의 권장 사항을 검토합니다.

적절한 크기로 Fargate에 대한 ECS 서비스 목록을 가져오려면 다음을 수행합니다.

1. Amazon S3 콘솔을 엽니다.
2. 탐색 창에서 버킷을 선택한 다음 결과를 내보낸 버킷을 선택합니다.
3. 객체 탭에서 객체를 선택하고 다운로드를 선택합니다.

4. 다운로드한 결과에서 결과 열을 필터링하여 Fargate의 OVER_PROVISIONED Amazon ECS 서비스만 표시합니다. 올바른 크기 조정을 위해 대상으로 지정하려는 Amazon ECS 서비스를 보여줍니다.
5. 나중에 사용할 수 있도록 작업 정의를 텍스트 편집기에 저장합니다.

오른쪽 크기 조정 태그 작업

워크로드에 태그를 지정하는 것은에서 리소스를 구성하기 위한 강력한 도구입니다 AWS. 태그를 사용하여 비용에 대한 세분화된 가시성을 확보하고 차지백을 활성화할 수 있습니다. 리소스에 태그를 AWS 추가하여 차지백 및 자동화를 처리하는 방법에는 여러 가지가 있습니다. 자세한 내용은 리소스 태그 지정을 위한 AWS 백서 모범 사례를 참조하세요. AWS 다음 예제에서는 AWS CloudShell를 사용하여 대상 계정 및 내 Amazon ECS 서비스의 일부인 모든 작업에 태그를 지정합니다 AWS 리전.

```
#!/bin/bash
# Set variables
TAG_KEY="rightsizing"
TAG_VALUE="enabled"
# Get a list of ECS Clusters
ClustersArns=$( w secs list-clusters -query 'clusterArns' -output text)
for ClustersArn in $ClustersArns; do
 ServiceArns=$( w secs list-services -cluster $ClustersArn -query 'serviceArns' -output
 text)
 for ServiceArn in $ServiceArns; do
  TasksArns=$( w secs list-tasks -cluster $ClustersArn -service-name $ServiceArn -query
 'taskArns' -output text)
  for TasksArn in $TasksArns; do
    w secs tag-resource -resource-arn $TasksArn -tags key=$TAG_KEY,value=$TAG_VALUE
  done
 done
done
```

다음 코드 예제는 모든 Amazon ECS 서비스에 대한 태그 전파를 활성화하는 방법을 보여줍니다.

```
#!/bin/bash
# Set variables
TAG_KEY="rightsizing"
TAG_VALUE="enabled"
# Get a list of ECS Clusters
ClustersArns=$(aws ecs list-clusters --query 'clusterArns' --output text)
for ClustersArn in $ClustersArns; do
```

```
 ServiceArns=$(aws ecs list-services --cluster $ClustersArn --query 'serviceArns' --
output text)
 for ServiceArn in $ServiceArns; do
  aws ecs update-service --cluster $ClustersArn --service $ServiceArn --propagate-tags
 SERVICE &>/dev/null
  aws ecs tag-resource --resource-arn $ServiceArn --tags key=$TAG_KEY,value=$TAG_VALUE
 done
done
```

비용 할당 태그가 AWS 결제 도구에서 작동하도록 활성화

사용자 정의 비용 할당 태그를 활성화하는 것이 좋습니다. 이렇게 하면 AWS 결제 도구(예: AWS Cost Explorer)에서 Rightsizing 태그를 인식하고 필터링할 수 있습니다 AWS Cost and Usage Report. 이를 활성화하지 않으면 태그 필터링 옵션과 데이터를 사용할 수 없습니다. 비용 할당 태그 사용에 대한 자세한 내용은 설명서의 사용자 정의 비용 할당 태그 활성화를 AWS 결제 및 비용 관리 참조하세요.

24시간을 기다린 후 다음 섹션에서 올바른 크기 조정 권장 사항을 구현하기 전에 Cost Explorer에서 태그를 볼 수 있습니다. 이렇게 하려면 Cost Explorer에서 크기 조정 태그를 검색합니다.

올바른 크기 조정 권장 사항 구현

Compute Optimizer는 작업 또는 컨테이너 크기 권장 사항을 제공합니다. 올바른 크기 조정 권장 사항을 구현하려면 다음을 수행합니다.

1. Amazon ECS 콘솔을 엽니다.
2. 탐색 모음에서 태스크 정의가 들어 있는 리전을 선택합니다.
3. 탐색 창에서 태스크 정의(Task definitions)를 선택합니다.
4. 태스크 정의(Task definitions) 페이지에서 태스크를 선택한 다음 새 개정 생성(Create new revision)을 선택합니다.
5. 새 태스크 정의 개정 생성(Create new task definition revision) 페이지에서 변경합니다. 컨테이너 크기 권장 사항을 업데이트하려면 ECS 작업 정의memory의 containerDefinitions 블록에서 cpu 및를 업데이트합니다. 예시:

```
"containerDefinitions": [
 {
  "name": "your-container-name",
  "image": "your-image",
  "cpu": 1024,
  "memory": 2048,
 }
```

```
],
```

6. 정보를 확인하고 생성(Create)을 선택합니다.

Amazon ECS 서비스를 업데이트하려면 다음을 수행합니다.

1. Amazon ECS 콘솔을 엽니다.
2. 클러스터(Clusters) 페이지에서 클러스터를 선택합니다.
3. Cluster overview(클러스터 개요) 페이지에서 서비스를 선택한 다음 Update(업데이트)를 선택합니다.
4. 태스크 정의(Task definition)에서 사용할 태스크 정의 패밀리 및 개정을 선택합니다.

고급 연산자의 경우 CloudShell을 사용하여 Amazon ECS 서비스를 업데이트할 수 있습니다. 예시:

```
bash
#!/bin/bash
# Set variables
ClustersName="workshop-cluster"
ServiceName="lab7-fargate-service"
TaskDefinition="lab7-fargate-demo:3"
# update the service
aws ecs update-service --cluster $ClustersName --service $ServiceName --task-definition
 $TaskDefinition
```

비용 전후 검토

리소스의 크기를 올바르게 조정한 후에는 Cost Explorer를 사용하여 Rightsizing 태그를 사용하여 비용 전후를 표시할 수 있습니다. 리소스 태그를 사용하여 비용을 추적할 수 있다는 점을 기억하세요. 여러 계층의 태그를 사용하면 비용에 대한 세분화된 가시성을 얻을 수 있습니다. 이 가이드에서 다루는 예제에서 Rightsizing 태그는 모든 대상 인스턴스에 일반 태그를 적용하는 데 사용됩니다. 그런 다음 팀 태그를 사용하여 리소스를 추가로 구성합니다. 다음 단계는 애플리케이션 태그를 도입하여 특정 애플리케이션 운영에 따른 비용 영향을 추가로 보여주는 것입니다.

단일 계정 수준에 대해 권한 부여 태그를 사용하여 달성할 수 있는 비용 절감의 예를 생각해 보세요. 이 예제에서 운영 비용은 일일 30.26 USD에서 일일 7.56 USD로 변경됩니다. 매월 744시간을 가정할 때 오른쪽 크기 조정 전 연간 비용은 11,044.9 USD입니다. 크기 조정 후 연간 비용은 2,759.4 USD로 떨어집니다. 이는이 계정의 컴퓨팅 비용이 75% 감소한다는 의미입니다. 대규모 조직 전체에서 이러한 영향이 미치는 것을 상상해 보세요.

올바른 크기 조정 여정을 시작하기 전에 다음 사항을 고려하세요.

- AWS 는 비용 절감을 위한 다양한 옵션을 제공합니다. 여기에는가 이동하기 전에 온프레미스 인스턴스를 AWS 검토하는 AWS OLA가 포함됩니다 AWS. 또한 AWS OLA는 적절한 크기 조정 권장 사항 및 라이선스 지침을 제공합니다.
- Savings Plans을 구매하기 전에 적절한 크기 조정을 모두 완료합니다. 이렇게 하면 Savings Plans 약정에서 초과 구매를 방지하는 데 도움이 될 수 있습니다.

다음 단계

다음 단계를 수행하는 것이 좋습니다.

1. 기존 환경을 검토하고 Amazon EBS gp2 볼륨을 gp3 볼륨으로 변환하는 것을 고려합니다.
2. Savings Plans 검토합니다.

추가 리소스

- Compute Optimizer 시작하기(AWS 문서)
- AWS 리소스 태그 지정 모범 사례(AWS 백서)
- 의 Windows 컨테이너 AWS(AWS Workshop Studio)

Amazon EKS 비용에 대한 가시성 확보

개요

Kubernetes 배포 비용을 효과적으로 모니터링하려면 전체적인 보기가 필요합니다. 유일하게 고정되고 알려진 비용은 Amazon Elastic Kubernetes Service(Amazon EKS) 컨트롤 플레인입니다. 여기에는 컴퓨팅 및 스토리지에서 네트워킹에 이르기까지 배포를 구성하는 다른 모든 구성 요소가 포함되며, 이는 애플리케이션 요구 사항에 따라 가변적인 양입니다.

Kubecost를 사용하여 네임스페이스 및 서비스에서 개별 포드까지 Kubernetes 인프라 비용을 분석한 다음 대시보드에 데이터를 표시할 수 있습니다. Kubecost는 컴퓨팅 및 스토리지와 같은 클러스터 내 비용과 Amazon Simple Storage Service(Amazon S3) 버킷 및 Amazon Relational Database Service(Amazon RDS) 인스턴스와 같은 out-of-cluster 비용을 표시합니다. Kubecost는이 데이터를 기반으로 적절한 크기 조정을 권장하고 시스템에 영향을 미칠 수 있는 중요한 알림을 표시합니다.

Kubecost는와 통합하여 Compute Savings Plans, 예약 인스턴스 및 기타 할인 프로그램의 절감액을 AWS Cost and Usage Report 표시할 수 있습니다.

비용 이점

Kubecost는 Amazon EKS 배포 비용을 시각화하는 보고서와 대시보드를 제공합니다. 이를 통해 클러스터에서 컨트롤러, 서비스, 노드, 포드 및 볼륨과 같은 다양한 구성 요소 각각으로 드릴다운할 수 있습니다. 이렇게 하면 Amazon EKS 환경에서 실행되는 애플리케이션을 전체적으로 볼 수 있습니다. 이러한 가시성을 활성화하면 Kubecost 권장 사항에 따라 조치를 취하거나 세분화된 수준에서 각 애플리케이션의 비용을 볼 수 있습니다. Amazon EKS 노드 그룹의 크기를 올바르게 조정하면 표준 EC2 인스턴스와 동일한 잠재적 절감 효과를 얻을 수 있습니다. 컨테이너와 노드의 크기를 조정할 수 있는 경우 컨테이너를 실행하는 데 필요한 인스턴스의 크기와 Auto Scaling 그룹에 필요한 EC2 인스턴스 수에서 컴퓨팅 부풀림을 제거할 수 있습니다.

비용 최적화 권장 사항

Kubecost를 활용하려면 다음을 수행하는 것이 좋습니다.

1. 환경에 Kubecost 배포
2. Windows 애플리케이션의 세분화된 비용 분석
3. 적절한 크기의 클러스터 노드
4. 적절한 크기의 컨테이너 요청
5. 활용도가 낮은 노드 관리
6. 중단된 워크로드 해결
7. 추천에 대한 조치
8. 자체 관리형 노드 업데이트

환경에 Kubecost 배포

Amazon EKS Finhack 워크숍에서는 AWS 소유 계정에서 Kubecost를 사용하도록 구성된 Amazon EKS 환경을 배포하는 방법을 설명합니다. 이를 통해 기술을 직접 체험할 수 있습니다. 조직에서이 워크숍을 실행하는 데 관심이 있는 경우 계정 팀에 문의하세요.

Helm을 사용하여 Amazon EKS 클러스터에 Kubecost를 배포하려면 AWS 및 Kubecost 공동 작업을 참조하여 블로그에 게시된 EKS 고객을 위한 비용 모니터링을 제공합니다. AWS 또는 Kubecost 설치 및

구성에 대한 지침은 공식 Kubecost 설명서를 참조할 수 있습니다. Windows 노드에 대한 Kubecost 지원에 대한 자세한 내용은 Kubecost 설명서의 Windows 노드 지원을 참조하세요.

Windows 애플리케이션의 세분화된 비용 분석

Amazon EC2 스팟 인스턴스를 사용하여 상당한 비용 절감을 달성할 수 있지만 Windows 워크로드가 상태 저장되는 경향이 있다는 점에서 이점을 얻을 수도 있습니다. 스팟 인스턴스 사용은 애플리케이션에 따라 다르므로 사용 사례에 적용할 수 있는지 확인하는 것이 좋습니다.

Windows 애플리케이션의 세분화된 비용 분석을 얻으려면 Kubecost에 로그인합니다. 탐색 페이지에서 절감액을 선택합니다.

적절한 크기의 클러스터 노드

Kubecost의 탐색 모음에서 절감액을 선택한 다음 클러스터 노드의 크기 조정을 선택합니다.

Kubecost가 클러스터가 vCPU 및 RAM 측면에서 과도하게 프로비저닝되었다고 보고하는 예를 생각해 보세요. 다음 표에는 Kubecost의 세부 정보 및 권장 사항이 나와 있습니다.

	현재	권장 사항: 간편	권장 사항: 복합
총 개수	매월 US $3462.57	매월 US $137.24	매월 US $303.68
노드 수	4	5	4
CPU	VCPUs	VCPUs	VCPUs
RAM	152GB	20GB	18GB
인스턴스 분석	c5.xlarge 2개 + 추가 2개	t3a.medium 5개	c5n.large 2개 + 추가 1개

Kubecost 블로그 게시물 Kubernetes 클러스터에 대한 최적의 노드 집합 찾기에서 설명한 대로 간단한 옵션은 단일 노드 그룹을 사용하는 반면 복잡한 노드 그룹은 다중 노드 그룹 접근 방식을 사용합니다. 채택 방법 알아보기 버튼은 원클릭 클러스터 크기 조정을 수행할 수 있습니다. Kubecost 클러스터 컨트롤러를 설치해야 합니다.

eksctl에서 생성하지 않은 자체 관리형 Windows 노드를 사용하는 경우 기존 자체 관리형 노드 그룹 업데이트를 참조하세요. 이 지침은 Auto Scaling 그룹에서 사용하는 Amazon EC2 시작 템플릿에서 인스턴스 유형을 변경하는 방법을 보여줍니다.

적절한 크기의 컨테이너 요청

Kubecost의 탐색 모음에서 절감액을 선택하고를 오른쪽 크기 추천 요청 페이지로 이동합니다. 이 페이지에는 포드의 효율성, 올바른 크기 조정 권장 사항 및 예상 비용 절감이 표시됩니다. 사용자 지정 버튼을 사용하여 클러스터, 노드, 네임스페이스\컨트롤러 등을 기준으로 필터링할 수 있습니다.

예를 들어, Kubecost가 CPU 및 RAM(메모리) 측면에서 일부 포드가 과도하게 프로비저닝되었다고 계산했다고 가정해 보겠습니다. 그런 다음 Kubecost는 예상 월별 절감액을 달성하기 위해 새 CPU 및 RAM 값에 맞게 조정할 것을 권장합니다. CPU 및 RAM 값을 변경하려면 배포 매니페스트 파일을 업데이트해야 합니다.

활용도가 낮은 노드 관리

Kubecost의 탐색 모음에서 절감액을 선택한 다음 활용도가 낮은 노드 관리를 선택합니다.

클러스터의 노드 하나가 CPU 및 RAM(메모리) 측면에서 활용되지 않아 드레이닝하고 종료하거나 크기를 조정할 수 있음을 보여주는 예를 생각해 보세요. 노드 및 포드 검사를 통과하지 않는 노드를 선택하면 드레이닝할 수 없는 이유에 대한 자세한 정보가 제공됩니다.

중단된 워크로드 해결

Kubecost의 탐색 모음에서 절감액을 선택한 다음 중단된 워크로드 페이지를 선택합니다. 이 예제에서는 창이라는 네임스페이스를 기준으로 필터링합니다. 이 페이지에는 트래픽 임계값을 충족하지 않고 중단된 것으로 간주되는 포드가 표시됩니다. 포드는 정의된 기간 동안 일정량의 네트워크 트래픽을 보내거나 받아야 합니다.

하나 이상의 포드가 중단되었음을 신중하게 고려한 후 복제본 수를 축소하거나, 배포를 삭제하거나, 리소스를 더 적게 소비하도록 크기를 조정하거나, 배포가 중단되었다고 생각되는 애플리케이션 소유자에게 알림으로써 비용을 절감할 수 있습니다.

권장 사항에 대한 조치

클러스터 노드의 적절한 크기 섹션에서 Kubecost는 클러스터의 작업자 노드 사용량을 분석하고 노드의 적절한 크기 조정에 대한 권장 사항을 제공하여 비용을 절감합니다. Amazon EKS와 함께 사용할 수 있는 노드 그룹에는 자체 관리형 및 관리형이라는 두 가지 유형이 있습니다.

자체 관리형 노드 업데이트

자체 관리형 노드 업데이트에 대한 자세한 내용은 Amazon EKS 설명서의 자체 관리형 노드 업데이트를 참조하세요. 로 생성된 노드 그룹은 업데이트할 `eksctl` 수 없으며 새 구성을 사용하여 새 노드 그룹으로 마이그레이션해야 한다고 명시되어 있습니다.

예를 들어 라는 Windows 노드 그룹ng-windows-m5-2xlarge (m5.2xlarge EC2 인스턴스 사용)이 있고 포드를 라는 새 노드 그룹ng-windows-t3-large (비용 절감을 위해 t3.large EC2 인스턴스 지원)으로 마이그레이션하려는 경우

에서 배포한 노드 그룹을 사용할 때 새 노드 그룹으로 마이그레이션하려면 다음을 eksctl수행합니다.

1. 포드가 현재 있는 노드를 찾으려면 kubectl describe pod <pod_name> -n <namespace> 명령을 실행합니다.
2. kubectl describe node <node_name> 명령을 실행합니다. 출력은 노드가 m5.2xlarge 인스턴스에서 실행 중임을 보여줍니다. 또한 노드 그룹 이름()과 일치합니다ng-windows-m5-2xlarge.
3. 노드 그룹를 사용하도록 배포를 변경하려면 노드 그룹을 ng-windows-t3-large삭제ng-windows-m5-2xlarge하고를 실행합니다kubectl describe svc,deploy,pod -n windows. 노드 그룹이 삭제되었으므로 배포가 즉시 다시 배포되기 시작합니다.

   > **Note**
   >
   > 노드 그룹을 삭제하면 서비스가 중단됩니다.

4. 몇 분 후에 kubectl describe svc,deploy,pod -n windows 명령을 다시 실행합니다. 출력은 포드가 모두 다시 실행 중 상태임을 보여줍니다.
5. 포드가 노드 그룹에서 실행 중임을 표시하려면 kubectl describe pod <pod_name> -n <namespace> 및 kubectl describe node <node_name> 명령을 다시 ng-windows-t3-large실행합니다.

대체 크기 조정 방법

이 방법은 자체 관리형 또는 관리형 노드 그룹의 모든 조합에 적용됩니다. EKS 자체 관리형 노드 그룹에서 EKS 관리형 노드 그룹으로 워크로드를 원활하게 마이그레이션 블로그 게시물에서는 워크로드를 대규모 인스턴스 유형의 한 노드 그룹에서 가동 중지 없이 적절한 크기의 노드 그룹으로 마이그레이션하는 방법에 대한 지침을 제공합니다.

다음 단계

Kubecost를 사용하면 Amazon EKS 환경의 비용을 쉽게 시각화할 수 있습니다. Kubecost를 Kubernetes 및 AWS APIs와 심층적으로 통합하면 잠재적 비용 절감을 찾는 데 도움이 될 수 있습니다. Kubecost의 절감형 대시보드에서 이를 권장 사항으로 볼 수 있습니다. Kubecost는 클러스터 컨트롤러 기능을 통해 이러한 권장 사항 중 일부를 구현할 수도 있습니다.

및 Kubecost 공동 작업의 step-by-step 배포를 검토하여 AWS Containers 블로그의 EKS 고객 블로그 게시물에 대한 비용 모니터링을 제공하는 것이 좋습니다. AWS

추가 리소스

- Amazon EKS 워크숍(Amazon EKS 워크숍)
- AWS 및 Kubecost가 협력하여 EKS 고객을 위한 비용 모니터링 제공(AWS 블로그)
- Amazon EKS Finhack 워크숍(AWS 워크숍 스튜디오)
- 의 Windows 컨테이너 AWS(AWS Workshop Studio)

App2Container를 사용하여 Windows 애플리케이션 리플랫폼

개요

AWS App2Container는 Java 및 .NET 웹 애플리케이션을 컨테이너로 마이그레이션하고 현대화하기 위한 명령줄 도구입니다. App2Container는 베어 메탈, 가상 머신, Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 또는 기타 클라우드 공급자에서 실행되는 모든 애플리케이션의 인벤토리를 분석하고 빌드합니다. 컨테이너화할 애플리케이션을 선택합니다. App2Container는 애플리케이션 아티팩트와 종속성을 컨테이너 이미지로 패키징하고, 네트워크 포트를 구성하고, 필요한 Amazon Elastic Container Service(Amazon ECS) 및 Amazon Elastic Kubernetes Service(Amazon EKS) 배포 아티팩트를 생성합니다.이 아티팩트는 코드형 인프라(IaC) 템플릿입니다. App2Container는 컨테이너화된 애플리케이션을 프로덕션 환경에 배포하는 데 필요한 클라우드 인프라 및 CI\CD 파이프라인을 프로비저닝합니다. 자세한 내용은 App2Container 설명서의 App2Container 작동 방식을 참조하세요. App2Container

App2Container를 사용하면 애플리케이션을 컨테이너로 마이그레이션 AWS 하고 현대화하는 동시에 애플리케이션의 배포 및 작업을 표준화할 수 있습니다. App2Container를 사용하면 개념 증명(PoC)을 빠르게 구축하거나 컨테이너에 프로덕션 워크로드 배포를 가속화할 수 있습니다.

Windows 애플리케이션 작업 시 유의해야 할 몇 가지 사항이 있습니다. App2Container는 Windows Server 2016, Windows Server 2019 또는 Windows Server Core 2004에서 실행되는 IIS 호스팅 Windows Communication Foundation(WCF) 애플리케이션을 포함하여 Microsoft Internet Information Services(IIS)에 배포된 ASP.NET 애플리케이션의 컨테이너화를 지원합니다. 자세한 내용은 App2Container 설명서의 Windows용 지원 애플리케이션을 참조하세요. App2Container는 Windows Server Core 컨테이너 버전을 컨테이너화 명령을 실행하는 서버의 운영 체제(OS) 버전과 일치시키는 Windows Server Core를 컨테이너 아티팩트의 기본 이미지로 사용합니다. 이 접근 방식은 애플리케이

션을 기본 OS에서 분리하므로 기존 마이그레이션을 수행하지 않고도 OS를 업그레이드할 수 있습니다.

작업자 시스템을 사용하여 애플리케이션을 컨테이너화하는 경우 Windows Server 2019 장기 서비스 채널(LTSC)과 같은 컨테이너 기본 이미지는 Windows Server 2019와 같은 작업자 시스템 OS와 일치합니다. 애플리케이션 서버에서 직접 컨테이너화를 실행하는 경우 버전은 애플리케이션 서버 OS와 일치합니다. 애플리케이션이 Windows Server 2008 또는 2012 R2에서 실행 중인 경우에도 컨테이너화 및 배포 단계를 위해 작업자 시스템을 설정하여 App2Container를 계속 사용할 수 있습니다. App2Container는 Windows 7 또는 Windows 10과 같은 Windows 클라이언트 운영 체제에서 실행되는 애플리케이션을 지원하지 않습니다. App2Container는 Java 프로세스에 Tomcat, TomEE 및 JBoss(독립 실행형 모드) 프레임워크를 지원합니다. 자세한 내용은 App2Container 호환성을 참조하세요.

비용 이점

애플리케이션을 컨테이너화하고 통합하면 one-application-to-one-server 배포 설계 패턴과 비교할 때 최대 60%의 컴퓨팅 비용을 절감할 수 있습니다. App2Container는 애플리케이션 컨테이너화 프로세스를 신속하게 처리하는 데 도움이 됩니다. 다음은 현대화 요구 사항에 App2Container를 사용할 때 얻을 수 있는 몇 가지 이점입니다.

- App2Container는 추가 비용 없이 제공됩니다.
- App2Container는 컨테이너 이미지에서 여러 애플리케이션을 지원합니다.
- App2Container를 사용하여 레거시 .NET 애플리케이션을 컨테이너로 이동하여 지원 종료에 근접하는 운영 체제를 해결합니다. 최신 운영 체제로 전환하고, 추가 지원 비용을 지불하지 않고, 보안 위험을 줄일 수 있습니다.
- 컨테이너는 .NET 애플리케이션을 패키징하는 효율적이고 비용 효율적인 방법입니다. MACO 권장 사항 - 컨테이너로 이동에서 컨테이너의 이점을 검토합니다.
- 애플리케이션 통합 및 컨테이너화는 컴퓨팅 리소스를 보다 효율적으로 사용하여 컴퓨팅, 스토리지 및 라이선스 공간을 줄이는 데 도움이 됩니다.
- 컨테이너로 전환하면 운영 오버헤드와 인프라 비용을 줄이고 개발 이동성과 배포 민첩성을 높일 수 있습니다.

비용 최적화 권장 사항

App2Container 사용 방법에 대한 지침은 시작하기를 참조하세요 AWS App2Container. App2Container 명령에 대한 자세한 내용은 App2Container 명령 참조를 참조하세요.

다음 단계

App2Container는 애플리케이션을 컨테이너화하고 Amazon EKS 또는 Amazon ECS에 배포하는 프로세스를 가속화할 수 있습니다. 컨테이너에 애플리케이션을 배포하면 컴퓨팅, 네트워킹 및 스토리지 비용이 절감되고 애플리케이션 운영자의 운영 오버헤드가 줄어듭니다.

App2Container에 대한 실습 경험은 AWS App2Container 워크숍을 통한 현대화를 참조하세요. 심층 학습 경험을 원한다면 AWS 계정 팀에 App2Container 몰입의 날을 설정하도록 요청하세요.

추가 리소스

- (AWS 블로그 게시물)를 사용하여 복잡한 다계층 Windows 애플리케이션 컨테이너화 AWS App2Container
- (AWS 블로그 게시물)를 사용하여 레거시 ASP.NET 애플리케이션 컨테이너화 AWS App2Container
- App2Container 지원 애플리케이션(AWS 문서)
- AWS App2Container 워크숍으로 현대화(AWS 워크숍 스튜디오)
- AWS App2Container FAQs(AWS 웹 사이트)

스토리지

Microsoft 워크로드에 적합한 스토리지를 선택하는 것은 중요한 아키텍처 결정입니다. 의사 결정 프로세스의 일환으로 스토리지 계획을 개발하고 애플리케이션 및 서비스에 대한 기능 요구 사항을 결정하는 것이 좋습니다. 이 장에서는 계획에 반영될 수 있는 다음 스토리지 옵션에 대한 개요를 제공합니다.

섹션:

- Amazon EBS
- Amazon FSx
- AWS Storage Gateway

Amazon EBS

Amazon Elastic Block Store(Amazon EBS)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 함께 사용할 수 있는 영구 블록 수준 스토리지 볼륨을 저장할 수 있는 완전 관리형 블록 스토리지 서비스입니다. Amazon EBS의 여러 기능을 활용하여 클라우드의 Windows 워크로드에 대한 스토리지 리소스를 효과적으로 관리하고 최적화할 수 있습니다. 예를 들어 Amazon EBS를 사용하여 워크로드에 필요한 정확한 IOPS 양과 처리량을 프로비저닝하고, 워크로드 요구 사항에 맞게 다양한 볼륨 유형 중에서 선택하고, 도구를 사용하여 낭비되는 스토리지 리소스를 식별하고 제거할 수 있습니다. 스토리지 성능 및 사용량을 세밀하게 제어하면 불필요한 비용을 방지하면서 스토리지 리소스를 최적화할 수 있습니다.

이 섹션은 다음 주제를 포함합니다.

- Amazon EBS 볼륨을 gp2에서 gp3로 마이그레이션
- Amazon EBS 스냅샷 수정
- 연결되지 않은 Amazon EBS 볼륨 삭제

Amazon EBS 볼륨을 gp2에서 gp3로 마이그레이션

개요

솔리드 스테이트 드라이브(SSD)는 프로덕션 및 고성능 워크로드를 위한 표준 스토리지 옵션입니다. Amazon EBS는 중고성능 워크로드를 위한 범용 SSD 볼륨을 제공합니다. 여러 AWS 서비스 (Amazon EC2 포함)의 표준은 이러한 범용 SSD 볼륨의 2세대인 gp2입니다. gp3라는 3세대 범용 SSDs는 2020년 12월에 출시되었습니다.

gp3 제품은 이전 세대에 비해 성능 사용자 지정 측면을 크게 개선했습니다. Amazon EBS gp2 볼륨의 경우 성능은 볼륨 크기와 밀접하게 결합됩니다. gp2 볼륨은 1GB의 용량마다 3IOPS의 성능을 제공합니다. 즉, 2,000GB gp2 볼륨은 6,000IOPS를 지원합니다. gp3 볼륨의 경우 스토리지 용량과 독립적으로 성능을 사용자 지정할 수 있습니다. 이를 통해 작은 용량 볼륨도 최대 16,000 IOPS 및 1,000Mb/s 처리량의 성능 기능을 달성할 수 있습니다.

gp3 볼륨의 또 다른 주요 변경 사항은 기준 IOPS 성능입니다. gp3 볼륨은 3,000 IOPS에서 시작합니다. 이에 비해 gp2 볼륨은 크기가 1TiB에 도달해야 동일한 성능 기능에 도달할 수 있습니다. 일반적으로 C: 드라이브가 1TiB보다 훨씬 작은 Windows Server의 경우 gp2에서 gp3로 업그레이드하는 것이 상당한 성능 개선입니다.

마지막으로 gp3 볼륨의 가격은 gp2 볼륨에 비해 가장 큰 개선 사항 중 하나입니다. gp3 볼륨은 gp2 볼륨보다 20% 저렴한 비용으로 향상된 성능 기능을 제공합니다.

비용 영향

용량과 독립적으로 성능을 확장할 수 있으므로 추가 IOPS 및 처리량 추가의 요금 측면을 이해하는 것이 중요합니다. gp2 볼륨의 경우 요금은 GiB-월당 0.10 USD의 프로비저닝된 용량을 기준으로 합니다. gp3 볼륨의 경우 요금은 고성능 프로비저닝된 IOPS SSD 볼륨과 비슷합니다.이 볼륨의 경우 용량에 대한 비용이 1개이고 추가 IOPS 및 처리량에 대한 별도의 비용이 있습니다.

다음 표에 나와 있듯이 gp3 볼륨의 용량 가격은 GiB-월당 0.08 USD(gp2보다 20% 저렴)이고 처리량에 대해 프로비저닝된 IOPS-월당 3,000 USD 및 프로비저닝된 MiBs-월당 0.04 USD의 별도 IOPS 비용은 125MiBs.

	gp3	gp2
볼륨 크기	1GiB – 16TiB	1GiB – 16TiB
기준 IOPS	3,000	3 IOPS/GiB(최소 100 IOPS)~최대 16,000 IOPS 1TiB 미만의 볼륨은 최대 3,000IOPS까지 버스트할 수 있습니다.
볼륨당 최대 IOPS	16,000	16,000
기준 처리량	125MiBs	처리량 한도는 볼륨 크기에 따라 128MiBs~250MiBs입니다.

	gp3	gp2
볼륨당 최대 처리량	1,000MiBs	250MiBs
가격	0.08 USD/GiB-월 3,000 IOPS 무료 및 3,000개 이상의 0.005 USD/프로비저닝된 IOPS-월 125MiBs 무료 및 125MiBs 이상 월 0.04 USD/프로비저닝된 MiBs	매월 GiB당 0.10 USD

> **Important**
>
> gp3 볼륨은 용량과 성능에 대해 별도의 비용이 발생하지만 동일한 성능 수준으로 구성된 경우 gp3 볼륨은 항상 gp2 볼륨보다 저렴합니다.

다음 표에는 다양한 용량 및 성능 구성에서 gp2를 gp3 볼륨으로 변환하여 달성할 수 있는 비용 절감의 예가 나와 있습니다.

gp2 구성의 예

볼륨 크기(GiB)	최대 IOPS	처리량(MiBs)	비용(USD/월)
30	3000	128	3.00 USD
100	3000	128	10.00 USD
500	3000	250	50.00 USD
1000	3000	250	100.00 USD
2000	6000	250	200.00 USD
6000	16000	250	600.00 USD

gp3(기준) 구성의 예

최대 IOPS	처리량(MiBs)	비용(USD/월)	비용 절감(gp2 대비)
3000	125	2.40 USD	20%
3000	125	8.00 USD	20%
3000	125	40.00 USD	20%
3000	125	80.00 USD	20%
3000	125	160.00 USD	20%
3000	125	480.00 USD	20%

gp3(gp2 일치) 구성의 예

최대 IOPS	처리량(MiBs)	비용(USD/월)	비용 절감(gp2 대비)
3000	128	2.52 USD	16%
3000	128	8.12 USD	19%
3000	250	45.00 USD	10%
3000	250	85.00 USD	15%
6000	250	180.00 USD	10%
16000	250	550.00 USD	8%

비용 분석은 Amazon EBS 리소스의 EBS gp2에서 gp3로의 마이그레이션 비용 절감 계산기 섹션을 참조하세요. 계산기를 다운로드하여 gp2 볼륨을 gp3로 마이그레이션하여 얼마나 절약할 수 있는지 확인할 수 있습니다.

비용 최적화 권장 사항

마이그레이션 프로세스를 완료하는 방법에 대한 지침은 AWS 스토리지 블로그의 gp2에서 gp3로 Amazon EBS 볼륨 마이그레이션 및 비용 최대 20% 절감 게시물을 참조하세요.

추가 리소스

- Amazon EBS 볼륨을 gp2에서 gp3로 마이그레이션하고 비용을 최대 20% 절감(AWS 스토리지 블로그)
- Amazon EBS 볼륨 유형을 최적화하는 AWS Config 사용자 지정 규칙 구축(AWS 클라우드 운영 및 마이그레이션 블로그)
- 미사용 Amazon EBS 볼륨을 삭제하여 AWS 비용 제어(AWS 클라우드 운영 및 마이그레이션 블로그)
- Amazon EBS 마이그레이션 유틸리티(GitHub)
- 2020 re:Invent 발표(Cloud Financial Management)에서 절감 효과 확인AWS
- 비용 최적화 워크숍(AWS Well-Architected Labs)
- gp2에서 gp3로의 마이그레이션 비용 절감 계산기(다운로드)

Amazon EBS 스냅샷 수정

개요

EBS 볼륨을 삭제하고 스냅샷의 보존 및 보관을 관리하는 것은 처음부터 비용을 제어하는 데 중요한 측면입니다. point-in-time 스냅샷을 생성하여 EBS 볼륨의 데이터를 Amazon Simple Storage Service(Amazon S3)에 백업할 수 있습니다. 스냅샷은 증분 백업이므로 가장 최근 스냅샷 이후에 변경된 디바이스에 블록만 저장합니다. 그러면 스냅샷을 만드는 데 필요한 시간이 최소화되며 데이터를 복제하지 않으므로 스토리지 비용이 절약됩니다. 각 스냅샷에는 (스냅샷이 생성된 시점부터) 데이터를 새 EBS 볼륨으로 복원하는 데 필요한 모든 정보가 포함되어 있습니다.

EBS 스냅샷 요금은 기가바이트-월 단위로 계산됩니다. 스냅샷의 크기와 스냅샷을 유지하는 기간에 대한 요금이 청구됩니다. 요금은 스토리지 계층에 따라 다릅니다. Standard 티어의 경우 저장된 변경된 블록에 대해서만 요금이 청구됩니다. 아카이브 계층의 경우 저장된 모든 스냅샷 블록에 대해 요금이 청구됩니다. 또한 아카이브 계층에서 스냅샷을 검색하는 데 따른 요금이 청구됩니다. 다음은 각 스토리지 계층에 대한 예제 시나리오입니다.

- 표준 티어 - 100GB의 데이터를 저장하는 볼륨이 있습니다. 첫 번째 스냅샷(스냅 A)의 전체 100GB 데이터에 대한 요금이 청구됩니다. 다음 스냅샷(스냅 B) 시 105GB의 데이터가 있습니다. 그런 다음 증분 스냅 B에 대해 추가 5GB 스토리지에 대해서만 요금이 청구됩니다.
- 아카이브 계층 - 스냅 B를 아카이브합니다. 그러면 스냅샷이 아카이브 계층으로 이동하고 전체 105GB 스냅샷 블록에 대한 요금이 청구됩니다.

Amazon Data Lifecycle Manager를 사용하면 일정에 따라 스냅샷을 유지하고 관리할 수 있는 수명 주기를 설정할 수 있습니다.

비용 영향

EBS 볼륨 및 스냅샷에 대한 요금은 별도로 관리됩니다. EBS 스냅샷은 활성 EBS 볼륨보다 낮은 요금으로 청구됩니다. 인스턴스가 종료되면 연결된 각 EBS 볼륨에 대한 DeleteOnTermination 속성의 값에 따라 볼륨을 보존할지 삭제할지 여부가 결정됩니다. 기본적으로 `DeleteOnTermination` 속성은 루트 볼륨에 `True` 대해 로 설정됩니다. 다른 모든 볼륨 유형에 `False` 대해 로 설정됩니다. 이렇게 하면 연산자가 EC2 인스턴스를 삭제하려고 하지만 루트 볼륨 외에도 인스턴스에 추가된 볼륨을 남겨 두는 상황이 발생합니다. 더 이상 필요하지 않은 볼륨(및 관련 스냅샷)을 확인하는 방법에 대한 지침은 Amazon EBS 설명서의 Amazon EBS 볼륨에 대한 정보 보기를 참조하세요.

기본적으로 스냅샷을 생성하면 스냅샷이 Amazon EBS 스냅샷 표준 계층(표준 계층)에 저장됩니다. 표준 계층에 저장된 스냅샷은 증분적입니다. 가장 최근의 스냅샷 이후에 변경된 볼륨의 블록만 저장됨을 의미합니다. Amazon EBS 스냅샷 아카이브는 자주 또는 빠르게 검색할 필요가 없는 거의 액세스하지 않는 스냅샷의 저렴한 장기 스토리지에 사용할 수 있는 새로운 스토리지 계층입니다. 표준과 아카이브의 요금 차이는 중요하며 스냅샷 전략을 설정할 때 중요한 고려 사항이어야 합니다. Amazon EBS 스냅샷 아카이브는 90일 이상 저장할 계획이고 액세스할 필요가 거의 없는 스냅샷에 대해 최대 75% 더 낮은 스냅샷 스토리지 비용을 제공합니다.

Amazon EBS 스냅샷 스토리지	비용
표준	0.05 USD/GB/월
아카이브	0.0125 USD/GB/월

소규모 환경에서는 비용 절감이 중요하지 않을 수 있습니다. EBS 볼륨이 삭제된 경우에도 TBs의 EBS 스냅샷이 있는 여러 계정과 수천 개의 EC2 인스턴스가 있는 대규모 규모에서 절감 효과가 더 큽니다.

다음 표에서는 사용량이 50TB에 불과한 월별 표준 계층과 아카이브 계층을 비교합니다. 이 규모가 낮더라도 연간 수천 달러의 비용 절감 효과가 있습니다.

Amazon EBS 스냅샷 스토리지	월별 비용	연간 비용
표준 50TB	312.50 USD	3,750 USD
아카이브 50TB	78.13 USD	937.60 USD
	연간 절감액	2,812.40 USD

비용 최적화 권장 사항

스냅샷을 삭제해도 조직의 데이터 스토리지 비용이 줄어들지 않을 수 있습니다. 다른 스냅샷은 해당 스냅샷의 데이터를 참조할 수 있으며, 참조된 데이터는 항상 보존됩니다. 예를 들어 10GiB의 데이터가 포함된 볼륨의 첫 번째 스냅샷을 생성하면 스냅샷의 크기도 10GiB입니다. 스냅샷은 증분식이기 때문에 동일한 볼륨으로부터 생성하는 두 번째 스냅샷에는 첫 번째 스냅샷이 생성된 이후 변경된 데이터 블록만 포함됩니다. 두 번째 스냅샷은 첫 번째 스냅샷의 데이터도 참조합니다. 4GiB의 데이터를 변경하고 두 번째 스냅샷을 생성하는 경우 두 번째 스냅샷의 크기는 4GiB입니다. 또한 두 번째 스냅샷은 첫 번째 스냅샷의 변경되지 않은 6GiB도 참조합니다. 자세한 내용은 AWS 지식 센터의 EBS 볼륨의 스냅샷을 삭제한 후 볼륨 자체를 삭제한 후 스토리지 비용이 절감되지 않은 이유는 무엇입니까?를 참조하세요.

다음을 고려하세요.

- 다른 사용자가 AWS 계정 소유하고 계정과 공유하는 스냅샷에 대해서는 요금이 청구되지 않습니다. 공유 스냅샷을 계정에 복사하는 경우에만 요금이 청구됩니다. 공유 스냅샷에서 생성한 EBS 볼륨에 대해서도 요금이 청구됩니다.
- 스냅샷(스냅 A)이 다른 스냅샷(스냅 B)에서 참조되는 경우 스냅 B를 삭제해도 스토리지 비용이 절감되지 않을 수 있습니다. 스냅샷을 삭제하면 해당 스냅샷에 고유한 데이터만 제거됩니다. 다른 스냅샷에서 참조하는 데이터는 그대로 유지되며이 참조된 데이터에 대한 요금이 청구됩니다. 증분 스냅샷을 삭제하려면 Amazon EBS 설명서의 증분 스냅샷 삭제를 참조하세요.

스냅샷 안정화는 워크로드를 실행할 때 표준 운영 관행입니다 AWS. 시간이 지남에 따라 스냅샷은 필요하지 않은 데이터에 대해 비용이 많이 드는 요금을 더할 수 있습니다.

추가 리소스

- 미사용 Amazon EBS 볼륨을 삭제하여 AWS 비용 제어(AWS 클라우드 운영 및 마이그레이션 블로그)
- Amazon EBS 스냅샷 삭제(Amazon EBS 설명서)
- 비용 최적화 워크숍(AWS Well-Architected Labs)
- Amazon Data Lifecycle Manager를 사용하여 Amazon EBS 스냅샷 자동 아카이브(AWS 스토리지 블로그)

연결되지 않은 Amazon EBS 볼륨 삭제

개요

연결되지 않은(고립된) EBS 볼륨으로 인해 AWS 환경에서 불필요한 스토리지 비용이 발생할 수 있습니다. 사용하지 않은 EBS 볼륨과 사용하지 않은 EBS 볼륨의 정기 검토 및 삭제를 AWS 환경 위생의 일부로 통합하는 것이 중요합니다. EBS 볼륨 사용을 지속적으로 검토하는 프로세스를 마련하는 것이 가장 좋습니다. 를 사용하여 사용률이 낮은 인스턴스AWS Compute Optimizer를 검토할 수 있습니다. 이 섹션에서는 연결되지 않았거나 활용도가 낮은 EBS 볼륨을 식별, 관리 및 삭제하는 데 도움이 됩니다.

Amazon EBS

Amazon Elastic Block Store(Amazon EBS)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 스토리지 볼륨을 제공하는 블록 수준 디바이스입니다. EBS는 EC2 인스턴스를 연결하고 분리할 수 있는 유연성과 함께 영구 스토리지를 제공합니다. 즉, EC2 인스턴스가 종료되더라도 EBS 볼륨의 수명 주기가 유지됩니다. DeleteOnTermination 속성은 인스턴스 종료 시 연결된 EBS 볼륨을 보존 또는 삭제할지 여부를 제어하는 기능입니다. 기본적으로 속성은 루트 볼륨에 `True` 대해 로 설정되어 삭제됩니다. 다른 볼륨의 `False` 경우 로 설정되어 보존됩니다.

비용 영향

미사용 또는 분리된 볼륨이라고도 하는 연결되지 않은 EBS 볼륨에는 프로비저닝된 스토리지 크기 및 스토리지 유형에 따라 연결된 볼륨과 동일한 요금이 발생합니다. Amazon EBS 요금의 평균 비용은 GB/월 0.10 USD로 최소화될 수 있지만 미사용 EBS 볼륨이 누적되면 시간이 지남에 따라 상당한 비용이 발생할 수 있다는 점을 인식하는 것이 중요합니다.

예를 들어, 다음 표와 같이 각각 스토리지 크기가 100GB인 프로비저닝된 미사용 EBS 볼륨 50개를 유지하는 것의 영향을 고려합니다.

스토리지 볼륨 수	볼륨 유형	크기	총 월별 비용
볼륨 50개	gp2(0.10 USD)	100GB	100GB 50.00 EBS 볼륨 월 $0.10 USD = $500.00 USD

위 표의 시나리오를 통해 매월 약 500 USD 또는 연간 6,000 USD의 비용이 절감됩니다. 이는 비용 절감을 위한 효과적인 단계입니다. 연결되지 않은 EBS 볼륨의 삭제를 AWS 환경 위생에 정기적으로 포함해야 합니다.

비용 최적화 권장 사항

AWS 를 사용하여 연결되지 않은 EBS 볼륨의 삭제를 쉽게 자동화할 수 있습니다. 예를 들어 AWS Lambda AWS Config, Amazon CloudWatch 및 AWS Systems Manager 를 사용하여 수명, 태그 및 기타 사양을 기반으로 연결되지 않은 볼륨을 삭제하기 위한 기준을 정의할 수 있습니다. 이를 사용하여 대규모로 정리 프로세스를 자동화 AWS 서비스 할 수도 있습니다.

의도하지 않은 결과를 방지하려면 연결되지 않은 EBS 볼륨을 삭제하기 전에 실사를 수행하는 것이 좋습니다.

연결되지 않은 EBS 볼륨 관리

다음 모범 사례를 고려하는 것이 좋습니다.

- 규정 준수 요구 사항 충족 - 연결되지 않은 EBS 볼륨의 삭제가 조직의 거버넌스 및 규정 준수 요구 사항을 준수하는지 확인합니다.
- 데이터 백업 및 보존 정책 설정 - 연결되지 않은 EBS 볼륨을 삭제하기 전에 중요한 데이터를 다른 스토리지 리포지토리(예: Amazon S3)에 백업합니다. 데이터 보존의 경우 Amazon EBS 스냅샷은 EBS 볼륨보다 데이터를 더 비용 효율적으로 보존하는 방법이며 향후 필요한 경우 볼륨을 복원할 수 있습니다. 스냅샷을 효과적으로 관리하는 방법에 대한 자세한 내용은이 가이드의 Amazon EBS 스냅샷 수정 섹션을 참조하세요.
- 종속성 확인 - 연결되지 않은 EBS 볼륨과 기타 AWS 리소스 간의 종속성이 있는지 확인합니다. AWS Management Console 또는 API를 사용하여 크기, 상태 및 관련 리소스와 같은 EBS 볼륨에 대한 설명 정보를 수집할 수 있습니다. 이는 일시적으로 연결되지 않은 리소스를 삭제하지 않도록 보호하는 중요한 단계입니다.
- 보존 정책 생성 - 연결되지 않은 EBS 볼륨에 대한 보존 기간을 설정합니다. 이렇게 하면 연결되지 않은 볼륨을 삭제할 적절한 시간을 식별하여 AWS 환경을 최적화할 수 있습니다. 예를 들어, Amazon

EventBridge 규칙을 생성하여 Lambda 함수를 일정에 따라 시작할 수 있습니다. Lambda 함수는 AWS SDK를 사용하여 연결되지 않은 EBS 볼륨을 능동적으로 식별하고, 쉽게 추적할 수 있도록 태그 지정 메커니즘을 적용하고, 연결되지 않은 EBS 볼륨이 정의된 임계값에 도달하거나 초과할 때 알림을 보낼 수 있습니다.

- 연결되지 않은 EBS 볼륨 태그 지정 - EBS 볼륨에 태그를 지정하는 것은 환경, 애플리케이션 또는 소유자와 같은 속성을 기반으로 볼륨을 구성하고 식별하는 데 도움이 될 수 있는 유용한 방법입니다. 이렇게 하면 태그를 기반으로 더 이상 필요하지 않은 볼륨을 빠르게 식별할 수 있으므로 연결할 수 없는 볼륨을 결정할 때 특히 유용할 수 있습니다.
- 안전한 삭제 보장 - EBS 볼륨이 마지막으로 연결된 시기를 검토하면 볼륨을 삭제해도 안전한지 확인하는 데 도움이 될 수 있습니다. 자세한 내용은 AWS 지식 센터에서 AWS CLI 명령을 사용하여 특정 Amazon EBS 볼륨의 연결 또는 분리 기록을 나열하려면 어떻게 해야 합니까?를 참조하세요.
- 활용도가 낮은 EBS 볼륨 식별 - 활용도가 낮은 EBS 볼륨을 식별 및 제거하는 것은 스토리지 비용을 절감하고 최적화된 AWS 환경을 유지하는 데 매우 권장되는 방법입니다. AWS Trusted Advisor는 활용도가 낮은 EBS 볼륨을 식별하고 비용을 절감하고 효율성을 개선하기 위한 권장 사항을 제공하는 데 도움이 될 AWS Compute Optimizer 수 있습니다. 예를 들어, (GitHub)를 사용하여 EBS 볼륨을 최적화하기 위한 자동화 설정 AWS Trusted Advisor, Trusted Advisor 조직 설정(TAO) 대시보드(AWS Workshop Studio) 및 (AWS 스토리지 블로그)를 사용하여 Amazon EBS 볼륨 비용 최적화 AWS Compute Optimizer를 참조하세요.

연결되지 않은 EBS 볼륨의 정리 자동화

연결되지 않은 EBS 볼륨의 정리를 자동화하려면 다음 도구를 고려하는 것이 좋습니다.

- AWS APIs(DescribeVolumes) - AWS SDKs 또는 AWS Command Line Interface ()를 사용하여 연결되지 않은 EBS 볼륨을 필터링하고 찾을 수 있습니다AWS CLI. 일정에 따라 실행되는 스크립트 또는 Lambda 함수를 사용하여이 프로세스를 자동화하여 시간과 노력을 절약할 수 있습니다. GitHub의 샘플 스크립트는 작동 방식을 보여줍니다. 스크립트는 Lambda를 사용하여 AWS CloudTrail 로그를 분석하고 연결되지 않은 EBS 볼륨을 식별합니다.
- AWS Systems Manager 자동화 - 이를 통해 인프라에서 일상적인 유지 관리 및 문제 해결 작업을 자동화할 수 있습니다. 시작하려면 특정 순서로 실행할 일련의 단계를 정의하는 자동화 실행서를 생성합니다. 예를 들어 연결되지 않은 EBS 볼륨의 스냅샷을 먼저 생성한 다음 볼륨 자체를 삭제하는 실행서를 생성할 수 있습니다. 이렇게 하면 수동으로 수행할 경우 시간이 많이 걸리고 오류가 발생하기 쉬운 작업을 자동화하는 데 도움이 될 수 있습니다.
- AWS Config - 이를 통해 시간 경과에 따른 AWS 리소스 변경 사항을 평가, 감사 및 추적할 수 있습니다. 구성 변경 사항을 캡처하여 AWS Config 를 사용하여 환경의 규정 준수, 거버넌스 및 리소스 사용률을 평가할 수 있습니다. 예를 들어는 미사용 EBS 볼륨을 식별할 AWS Config 수 있습니다. 또한

AWS Systems Manager Automation AWS Config 을와 연결하여 미사용 EBS 볼륨의 삭제를 자동으로 해결할 수 있습니다.

추가 리소스

- AWS Config 및를 사용하여 미사용 Amazon Elastic Block Store(Amazon EBS) 볼륨 삭제 AWS Systems Manager(AWS 권고 가이드)
- 미사용 Amazon EBS 볼륨을 삭제하여 AWS 비용 제어(AWS 클라우드 운영 및 마이그레이션 블로그)
- AWSConfigRemediation-DeleteUnusedEBSVolume(AWS Systems Manager Automation 실행서 참조)

Amazon FSx

Amazon FSx for Windows File Server는 Windows 워크로드에 최적화된 완전 관리형 파일 스토리지 서비스입니다. 복잡한 스토리지 인프라 관리 없이 Windows 기반 애플리케이션 및 워크로드를 실행할 수 있는 간단하고 확장 가능한 솔루션을 제공합니다. FSx for Windows File Server를 사용하면 Microsoft SQL Server, Microsoft SharePoint 및 사용자 지정 .NET 애플리케이션을 포함하여 Windows 애플리케이션을 기본적으로 지원하는 공유 파일 스토리지를 쉽게 프로비저닝하고 액세스할 수 있습니다. 또한 FSx for Windows File Server는 pay-as-you-go 및 스토리지 할당량, 스토리지 공간을 줄이고 성능과 비용을 최적화하는 자동 데이터 중복 제거와 같은 유연한 요금 옵션을 제공하여 비용을 관리하는 데 도움이 됩니다.

이 섹션은 다음 주제를 포함합니다.

- 올바른 SMB 파일 스토리지 선택
- Amazon FSx에서 데이터 중복 제거 활성화
- FSx for Windows File Server의 데이터 샤딩 이해
- Amazon FSx의 HDD 볼륨 사용량 이해
- 단일 가용 영역 사용

올바른 SMB 파일 스토리지 선택

개요

AWS 는 최신 AWS 인프라 혁신과 보안을 결합하면서 업계 최고의 파일 서비스의 풍부한 기능을 제공하는 다양한 완전 관리형 스토리지 서비스를 제공합니다. 서비스를 코드형 인프라(IaC) 워크플로에 통합 AWS 하고 AWS 컴퓨팅, 모니터링 및 데이터 보호 서비스와 통합할 수 있습니다. Windows 워크로드의 경우 애플리케이션 요구 사항에 맞게 사용할 수 있는 두 가지 완전 관리형 파일 서비스인 FSx for Windows File Server와 Amazon FSx for NetApp ONTAP 중에서 선택할 수 있습니다.

FSx for Windows File Server

Amazon FSx for Windows File Server는 Windows Server에 구축된 완전관리형 공유 스토리지를 제공하며 광범위한 데이터 액세스, 데이터 관리 및 관리 기능을 제공합니다. FSx for Windows File Server는 Windows 네이티브 서비스이므로 Windows 환경과 쉽게 통합됩니다. 사용자 및 그룹 공유에는 FSx for Windows File Server를 사용하고, SQL Server, Windows 애플리케이션 및 가상 데스크톱 인프라(VDI)에는 Always On 장애 조치 클러스터 인스턴스를 사용하는 것이 좋습니다. FSx for Windows File Server는 Amazon FSx File Gateway, Amazon Kendra, Amazon S3에 대한 감사 로그 및 Amazon Data Firehose와도 잘 통합됩니다.

FSx for ONTAP

FSx for ONTAP은 NetApp의 독점 ONTAP 파일 시스템을 기반으로 합니다. 어느 정도의 기술 향상이 필요하며 대부분 기존 온프레미스 NetApp 사용자에게 권장됩니다. 일반적인 사용 사례에는 사용자 및 그룹 공유, SQL Server용 Always On 장애 조치 클러스터 인스턴스, Windows 애플리케이션이 포함됩니다. FSx for ONTAP은 64TB를 초과하는 파일 시스템(DFS 네임스페이스 서버 없이 PB 규모 조정), 복제, 복제, 스냅샷, 압축(스토리지 효율성) 및 데이터의 지능형 계층화를 지원하는 여러 프로토콜을 지원합니다.

비용 영향

FSx for Windows File Server

FSx for Windows File Server는 SQL Server AWS 용 장애 조치 클러스터 인스턴스를 배포하기 위한 의 첫 번째 공유 스토리지 솔루션입니다. FSx for Windows File Server를 사용하면 SQL Standard 버전 라이선스를 사용하여 장애 조치 클러스터 인스턴스를 시작할 수 있습니다. 그러나 이렇게 하면 SQL Server Enterprise Edition 라이선스가 필요한 Always On 가용성 그룹에 의존할 수 없습니다. SQL Server Enterprise Standard 에디션에서 SQL Server Standard 에디션으로 전환하면 SQL Server 라이선스에 65~75%를 절약할 수 있습니다.

장애 조치 클러스터 인스턴스용 FSx for Windows File Server를 사용하여 일반적인 EBS 스토리지에서 스토리지 I/O를 오프로드할 수 있습니다. I/O를 FSx for Windows File Server로 오프로드하면 스토리지 처리량에 영향을 주지 않고 높은 Amazon EBS 처리량과 IOPS에 의존하는 EC2 인스턴스를 스케일 다운할 수 있습니다.

FSx for ONTAP

FSx for ONTAP을 사용하여 블록 프로토콜 iSCSI에서 Microsoft 장애 조치 클러스터를 실행하고 SQL Server 인스턴트 파일 초기화, SnapMirror를 사용한 리전 간 복제, 바이러스 백신 지원 및 복제의 이점을 누릴 수 있습니다. 테스트를 위해 여러 데이터베이스 복사본을 생성하는 경우 복제는 공간 소비와 이러한 데이터베이스 복사본을 얼마나 빨리 생성할 수 있는지에 상당한 차이를 만들 수 있습니다. 또한 NetApp SnapCenter를 사용하여 FSx for ONTAP을 사용하여 SQL Server용 EC2 인스턴스의 백업, 복원 및 복제 기능을 관리할 수 있습니다. 또한 FSx for ONTAP은 SSD에서 저비용 용량 풀 스토리지로의 자동 계층화를 제공하여 성능과 비용 효율성을 혼합합니다.

FSx for ONTAP은 Windows 기본 NTFS 파일 시스템을 지원하는 FSx for Windows File Server와 달리 NetApp의 파일 시스템(ONTAP)을 지원합니다. FSx for ONTAP의 최소 크기는 1,024GB이고 FSx for Windows File Server는 32GB부터 시작할 수 있습니다.

Microsoft 분산 파일 시스템과의 통합

FSx for Windows File Server 및 FSx for ONTAP은 Microsoft의 분산 파일 시스템(DFS)과 통합되어 기존 배포에 원활하게 통합됩니다. 아키텍처를 계획할 때는 다음 사항에 유의하세요.

- FSx for Windows File Server 및 FSx for ONTAP은 두 배포 유형(다중 가용 영역 및 단일 가용 영역) 모두에서 DFS 네임스페이스(DFSN)를 지원합니다.
- FSx for Windows File Server만 DFS 복제(DFSR)를 지원하며, 단일 가용 영역을 사용하는 경우에만 지원됩니다.

비용 최적화 권장 사항

FSx for Windows File Server와 FSx for ONTAP의 성능은 요금과 마찬가지로 구성에 따라 매우 달라집니다. FSx for Windows File Server 요금은 주로 스토리지 용량 및 스토리지 유형, 처리량 용량, 백업 및 전송된 데이터에 따라 달라집니다. FSx for ONTAP을 사용하면 SSD 스토리지, SSD IOPS, 용량 풀 사용량, 처리량 용량 및 백업 비용을 지불합니다.

파일 서비스	5TB 스토리지 비용	구성	리전
FSx for Windows File Server	982.78 USD	단일 가용 영역 SSD(15,000IOPS) 32MBps 5TB 백업(중복 제거 비용 절감 없음)	미국 동부(버지니아 북부)
FSx for ONTAP	979.28 USD	단일 가용 영역 100% SSD 15,000개의 읽기-쓰기 용량 계층 15,000 SSD IOPS 128MBps 5TB 백업(중복 제거 비용 절감 없음)	미국 동부(버지니아 북부)

다음 사항에 유의하세요.

- 중복 제거 및 압축을 사용하면 데이터 크기를 줄여 물리적 디바이스에 더 많은 데이터를 저장할 수 있지만 프로비저닝된 솔리드 스테이트 드라이브(SSD) 또는 하드 디스크 드라이브(HDD) 스토리지에 대한 비용을 지불하면 됩니다.
- FSx for ONTAP을 사용하여 데이터를 계층화할 수 있습니다. 데이터의 100%를 정기적으로 액세스하고 SSD 스토리지가 필요한 경우는 매우 드뭅니다. 콜드 데이터 및 자주 액세스하지 않는 데이터를 용량 계층으로 이동하여 비용을 절감할 수 있습니다.
- 여기에 언급된 요금은 SSD 계층의 100% 데이터와 SSD 계층의 15,000IOPS로 계산됩니다.

백업

기본적으로 FSx for ONTAP 및 FSx for Windows File Server는 모두 Amazon S3에 완전 관리형 백업을 저장합니다. 그러나 FSx for ONTAP에는 SnapVault를 사용한 백업을 위한 추가 옵션이 있으며,이

옵션은 용량 계층에 상주하도록 백업을 구성할 수 있습니다. SnapVault를 사용한 백업은 기본 완전 관리형 백업 옵션보다 비용 효율적인 자체 관리형 메커니즘입니다. 완전 관리형 백업 옵션은 GB/월 0.05 USD입니다. FSx for ONTAP(10:1 SSD에서 용량 풀 스토리지로)의 SnapVault 백업은 0.03221 USD(0.9x0.0219+0.1x0.125)입니다.

다음 사항에 유의하세요.

- AWS 관리형 백업은 1시간의 세부 수준을 제공합니다. SnapVault를 사용하면 최소 5분까지 진행할 수 있습니다.
- NetApp의 도구(예: CLI 및 API)를 사용하여 SnapVault 관계 및 스냅샷 복제를 구성할 수 있습니다.
- SnapVault 볼륨에서 `all` 계층화 정책을 활성화하여 용량 계층을 백업 데이터의 스토리지로 사용합니다.
- SnapVault 대상은 동일한 AWS 리전리전 간 또는 온프레미스에 있을 수 있습니다. 이는 일반적으로 단일 가용 영역 또는 다중 가용 영역 파일 시스템 백업 대상에 해당합니다. 이에 비해 AWS Backup는 Amazon S3의 리전 복원력을 기반으로 합니다.

오른쪽 크기 조정

또한 적절한 크기 조정 및 과다 프로비저닝 방지를 통해 비용을 절감하고 파일 시스템을 최대한 활용할 수 있습니다.

적절한 크기로 다음을 수행합니다.

1. 데이터를 기반으로 현재 요구 사항을 식별합니다. 일반적인 Windows 워크로드의 경우 성능 모니터와 같은 기본 제공 운영 체제 도구를 사용할 수 있습니다.
2. 성능 모니터에서 다음 카운터를 사용하여 현재 성능 요구 사항을 측정합니다. 캡처 간격은 1초로 설정되며 최대 로그 크기는 1,000MB이고 덮어쓰기가 활성화됩니다.

```
Logman.exe create counter PerfLog-Short -o "c:\perflogs\PerfLog-Long.blg" -f bincirc
 -v mmddhhmm -max 1024 -c "\LogicalDisk(*)\*" "\Memory\*" "\.NET CLR Memory(*)\*"
 "\Cache\*" "\Network Interface(*)\*" "\Paging File(*)\*" "\PhysicalDisk(*)\*"
 "\Processor(*)\*" "\Processor Information(*)\*" "\Process(*)\*" "\Thread(*)\*"
 "\Redirector\*" "\Server\*" "\System\*" "\Server Work Queues(*)\*" "\Terminal
 Services\*" -si 00:00:01
```

3. 로그 캡처를 시작하려면 `logman start PerfLog-Short` 명령을 실행합니다. 로그 캡처를 중지하려면 `logman stop PerfLog-Short` 명령을 실행합니다.

> **Note**
>
> 캡처를 실행하는 서버의 c:\perflogs에서 성능 로그 파일을 찾을 수 있습니다. 자세한 내용은 Microsoft 설명서의 Windows 성능 모니터 개요를 참조하세요.

4. 올바른 구성을 식별한 후 Microsoft DISKSPD와 같은 디스크 스트레스 도구를 사용하여 Amazon FSx 파일 시스템에서 추정치가 올바른지 테스트합니다.
5. 성능에 만족하는 경우 파일 공유로 전환합니다.

스토리지 용량은 확장만 가능하므로 보수적인 방식으로 스토리지 용량을 사용하는 것이 좋습니다. 필요에 따라 처리량 용량을 확장 및 축소할 수 있습니다.

추가 리소스

- Amazon FSx for NetApp ONTAP FAQs(AWS 웹 사이트)
- 새 지표로 Amazon FSx for Windows File Server 성능 최적화(AWS 스토리지 블로그)

Amazon FSx에서 데이터 중복 제거 활성화

개요

데이터 중복 제거는 더 적은 용량 요구 사항으로 데이터를 더 효율적으로 저장할 수 있는 기능입니다. 여기에는 충실도 또는 무결성을 손상시키지 않고 데이터 내에서 중복을 찾아 제거하는 작업이 포함됩니다. 데이터 중복 제거는 하위 파일 변수 크기 청킹 및 압축을 사용합니다.이 청킹 및 압축은 일반 파일 서버의 경우 2:1, 가상화 데이터의 경우 최대 20:1의 최적화 비율을 제공합니다. 데이터 중복 제거는 NTFS 압축보다 훨씬 더 효과적입니다. 중복 제거 아키텍처에 내재된는 하드웨어 장애 시 복원력이 뛰어나며 메타데이터 및 가장 많이 액세스된 데이터 청크에 대한 중복성을 포함하여 데이터 및 메타데이터에 대한 전체 체크섬 검증을 제공합니다.

FSx for Windows File Server는 데이터 중복 제거를 완벽하게 지원합니다. 이를 사용하면 범용 파일 공유의 평균 50~60%가 절감될 수 있습니다. 공유 내에서 절감액 범위는 사용자 문서의 경우 30~50%, 소프트웨어 개발 데이터 세트의 경우 최대 70~80%입니다. 데이터 중복 제거를 통해 얻을 수 있는 스토리지 절감액은 파일 간에 중복이 존재하는 정도를 포함하여 데이터 세트의 특성에 따라 달라집니다. 저장된 데이터가 본질적으로 동적인 경우 중복 제거는 좋은 옵션이 아닙니다.

비용 영향

엔터프라이즈의 데이터 스토리지 성장에 대처하기 위해 관리자는 서버를 통합하고 용량 조정 및 데이터 최적화 주요 목표를 설정합니다. 데이터 중복 제거의 기본 설정은 즉시 비용을 절감하거나 관리자가 설정을 미세 조정하여 추가 이점을 확인할 수 있습니다. 예를 들어 특정 파일 유형에서만 중복 제거가 실행되도록 구성하거나 사용자 지정 작업 일정을 만들 수 있습니다.

높은 수준에서 중복 제거에는 최적화, 폐영역 회수, 스크러빙이라는 세 가지 유형의 작업이 있습니다. 최적화 후 가비지 수집 작업을 실행할 때까지 공간이 확보되지 않습니다. 작업을 예약하거나 수동으로 실행할 수 있습니다. 데이터 중복 제거 작업을 예약할 때 사용할 수 있는 모든 설정은 작업을 수동으로 시작할 때도 사용할 수 있습니다(예약별 설정은 제외).

중복 제거를 통한 유효 비용 절감 효과가 25%에 불과하더라도 FSx for Windows File Server의 경우 상당한 비용 절감 효과가 있습니다. 이러한 예상 절감액은의 추정치를 기반으로 합니다 AWS Pricing Calculator.

비용 최적화 권장 사항

FSx for Windows File Server 파일 시스템의 중복 제거는 기본적으로 활성화되어 있지 않습니다. PowerShell에서 원격 관리를 사용하여 중복 제거를 활성화하려면 `Enable-FSxDedup` 명령을 실행한 다음 `Set-FSxDedupConfiguration` 명령을 사용하여 구성을 설정해야 합니다. 자세한 내용은 FSx for Windows File Server 설명서의 파일 시스템 관리를 참조하세요.

중복 제거를 활성화하려면 다음 명령을 실행합니다.

```
PS C:\Users\Admin> Invoke-Command -ComputerName amznfsxzzzzzzzz.corp.example.com -
ConfigurationName FSxRemoteAdmin -ScriptBlock {Enable-FsxDedup }
```

중복 제거 구성을 확인하려면 다음 명령을 실행합니다.

```
Invoke-Command -ComputerName amznfsxzzzzzzzz.corp.example.com -ConfigurationName
 FSxRemoteAdmin -ScriptBlock {
Set-FSxDedupSchedule -Name "CustomOptimization" -Type Optimization -Days
 Mon,Tues,Wed,Sat -Start 09:00 -DurationHours 7
}
```

PowerShell `Measure-DedupFileMetadata` cmdlet을 실행하여 폴더 그룹, 단일 폴더 또는 단일 파일을 삭제한 다음 가비지 수집 작업을 실행할 경우 볼륨에서 회수할 수 있는 잠재적 디스크 공간을 확인할 수 있습니다. 특히이 `DedupDistinctSize` 값은 파일을 삭제할 경우 반환되는 공간의 양을 알려

줍니다. 파일에는 다른 폴더 간에 공유되는 청크가 있는 경우가 많으므로 중복 제거 엔진은 고유하고 가비지 수집 작업 후에 삭제될 청크를 계산합니다.

기본 데이터 중복 제거 작업 일정은 권장 워크로드에 적합하고 최대한 비침입적이 되도록 설계되었습니다(백업 사용 유형에 대해 활성화된 우선 순위 최적화 작업 제외). 워크로드에 많은 리소스 요구 사항이 있는 경우 유휴 시간에만 작업을 실행하도록 예약하거나 데이터 중복 제거 작업이 사용할 수 있는 시스템 리소스의 양을 줄이거나 늘리는 것이 좋습니다.

기본적으로 데이터 중복 제거는 사용 가능한 메모리의 25%를 사용합니다. 그러나를 사용하여 이를 늘릴 수 있습니다`-memory switch`. 최적화 작업의 경우 범위를 15~50으로 설정하는 것이 좋습니다. 예약된 작업의 경우 더 많은 메모리 소비를 사용할 수 있습니다. 예를 들어, 가비지 수집 및 스크러빙 작업(일반적으로 휴지 시간에 실행하도록 예약)을 사용하면 메모리 사용량을 높일 수 있습니다(예: 50).

데이터 중복 제거 설정에 대한 자세한 내용은 FSx for Windows File Server 설명서의 데이터 중복 제거를 통한 스토리지 비용 절감을 참조하세요.

추가 리소스

- 데이터 중복 제거 이해(Microsoft 설명서)
- 데이터 중복 제거를 통한 스토리지 비용 절감(FSx for Windows File Server 설명서)

FSx for Windows File Server의 데이터 샤딩 이해

개요

FSx for Windows File Server 성능은 구성에 따라 다릅니다. 주로 스토리지 유형, 스토리지 용량 및 처리량 구성을 기반으로 합니다. 선택한 처리량 용량은 네트워크 I/O 제한, CPU 및 메모리, 파일 서버에서 부과하는 디스크 I/O 제한을 포함하여 파일 서버에 사용할 수 있는 성능 리소스를 결정합니다. 선택한 스토리지 용량 및 스토리지 유형에 따라 스토리지 볼륨에 사용할 수 있는 성능 리소스, 즉 스토리지 디스크에서 부과하는 디스크 I/O 제한이 결정됩니다. 성능 외에도 구성 선택 사항도 비용에 영향을 미칩니다. FSx for Windows File Server 요금은 주로 스토리지 용량 및 스토리지 유형, 처리량 용량, 백업 및 전송된 데이터에 따라 달라집니다.

파일 스토리지 및 성능 요구 사항이 비교적 큰 경우 데이터 샤딩의 이점을 누릴 수 있습니다. 데이터 샤딩에는 파일 데이터를 더 작은 데이터 세트(샤드)로 나누고 여러 파일 시스템에 저장하는 작업이 포함됩니다. 여러 인스턴스에서 데이터에 액세스하는 애플리케이션은 이러한 샤드에 대한 읽기 및 쓰기를 병렬로 수행하여 높은 수준의 성능을 달성할 수 있습니다. 동시에 공통 네임스페이스를 사용하여 애플리케이션에 통합된 뷰를 제공할 수도 있습니다. 또한 파일 데이터 스토리지를 각 파일 시스템이 대용량

파일 데이터 세트에 대해 지원하는 것(64TB) 이상으로 최대 수백 페타바이트까지 확장하는 데 도움이 될 수 있습니다.

비용 영향

대규모 데이터 세트의 경우 일반적으로 동일한 수준의 성능을 달성하기 위해 하나의 대용량 SSD 공유가 아닌 여러 개의 작은 FSx for Windows File Server 파일 시스템을 배포하는 것이 더 효과적입니다. FSx for Windows File Server HDD와 SSD 스토리지 유형을 함께 사용하면 비용을 절감할 수 있으며 워크로드를 최상의 기본 디스크 하위 시스템과 일치시킬 수 있습니다. 다음 표에서는 단일 17TB 파일 시스템의 차이점을 확인하고 동일한 용량에를 추가하는 여러 개의 작은 파일 시스템과 비교할 수 있습니다.

워크로드가 여러 개인 대용량 SSD 파일 시스템

[서버 이름]	비용	구성	리전
Amazon FSx for Windows File Server	5,716 USD	17TB SSD 30% 중복 제거 256Mbps 17TB 백업	미국 동부(버지니아 북부)

DFSN을 사용하여 분할된 워크로드

[서버 이름]	비용	구성	리전	공유
Amazon FSx for Windows File Server	1,024 USD	2TB SSD 20% 중복 제거 128Mbps 2TB 백업 Multi-AZ	미국 동부(버지니아 북부)	공유 1

[서버 이름]	비용	구성	리전	공유
Amazon FSx for Windows File Server	2,132 USD	5TB SSD 30% 중복 제거 256Mbps 5TB 백업 Multi-AZ	미국 동부(버지니아 북부)	공유 2
Amazon FSx for Windows File Server	1,036 USD	10TB HDD 40% 중복 제거 128Mbps 10TB 백업 Multi-AZ	미국 동부(버지니아 북부)	공유 3
DFSN Windows EC2 인스턴스	27 USD	t3a.medium vCPU 2개 4GiB 메모리	미국 동부(버지니아 북부)	DFSN 인스턴스

대형 SSD 파일 시스템의 연간 비용은 68,592 USD입니다. 분할된 워크로드의 연간 비용은 50,640 USD입니다. 이 예제에서는 워크로드를 적절한 백엔드 스토리지에 매칭하면서 26% 절감 효과를 얻을 수 있습니다. 요금 추정에 대한 자세한 내용은 AWS Pricing Calculator 견적을 참조하세요.

비용 최적화 권장 사항

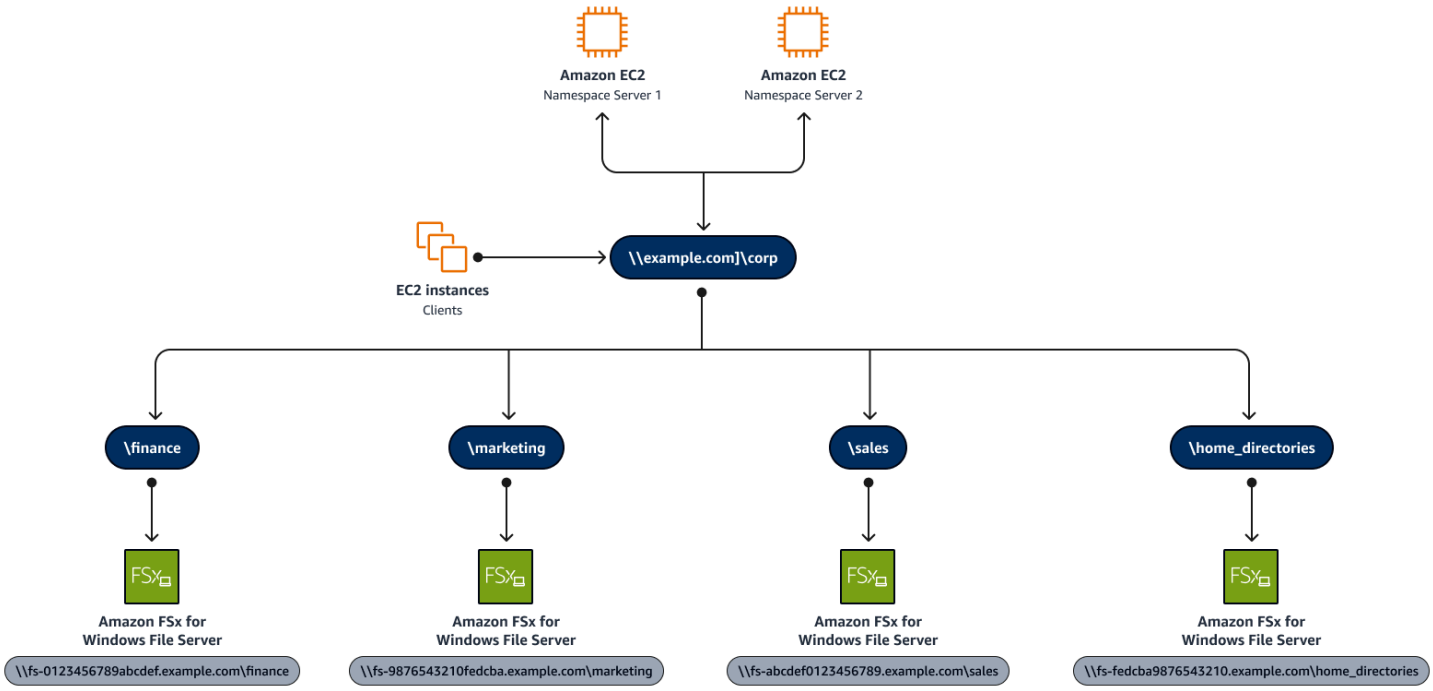
데이터 중복 제거 솔루션을 배포하려면 데이터 유형, I/O 크기 및 I/O 액세스 패턴을 기반으로 Microsoft DFS 네임스페이스를 설정해야 합니다. 각 네임스페이스는 최대 50,000개의 파일 공유와 총 수백 페타바이트의 스토리지 용량을 지원합니다.

사용하려는 모든 파일 시스템에 I/O를 균등하게 분산하는 샤딩 규칙을 선택하는 것이 가장 효율적입니다. 워크로드를 모니터링하면 추가 최적화 또는 비용 절감에 도움이 됩니다. Amazon FSx 파일 시

스템의 성능 정보를 측정하는 데 도움이 필요한 경우 FSx for Windows File Server 설명서의 FSx for Windows File Server 성능을 참조하세요. FSx

샤딩 전략을 선택한 후 DFS 네임스페이스를 사용하여 공유에 쉽게 액세스할 수 있도록 파일 시스템을 그룹화할 수 있습니다. 이를 통해 사용자는 실제로 용도에 맞게 구축된 사용 사례를 통해 다양한 파일 시스템에 액세스할 때 하나의 동종 파일 시스템을 볼 수 있습니다. 최종 사용자가 공유가 설계된 워크로드를 쉽게 확인할 수 있도록 적절한 이름 지정 규칙을 사용하여 공유를 생성하는 것이 중요합니다. 프로덕션 공유와 비프로덕션 공유에 레이블을 지정하는 것도 중요하므로 최종 사용자는 실수로 잘못된 파일 시스템에 파일을 배치하지 않습니다.

다음 다이어그램은 단일 DFS 네임스페이스를 여러 Amazon FSx 파일 시스템의 액세스 포인트로 사용하는 방법을 보여줍니다.



다음 사항에 유의하세요.

- 기존 FSx for Windows File Server 공유를 DFS 트리에 추가할 수 있습니다.
- Amazon FSx는 DFS 공유 경로의 루트에 추가할 수 없습니다. 하위 폴더는 하나뿐입니다.
- DFS 네임스페이스 구성을 제공하려면 EC2 인스턴스를 배포해야 합니다.

DFS-N 구성에 대한 자세한 내용은 Microsoft 설명서의 DFS 네임스페이스 개요를 참조하세요. DFS 네임스페이스 사용에 대한 자세한 내용은 YouTube의 Amazon FSx for Windows File Server에서 DFS 네임스페이스 사용 비디오를 참조하세요.

추가 리소스

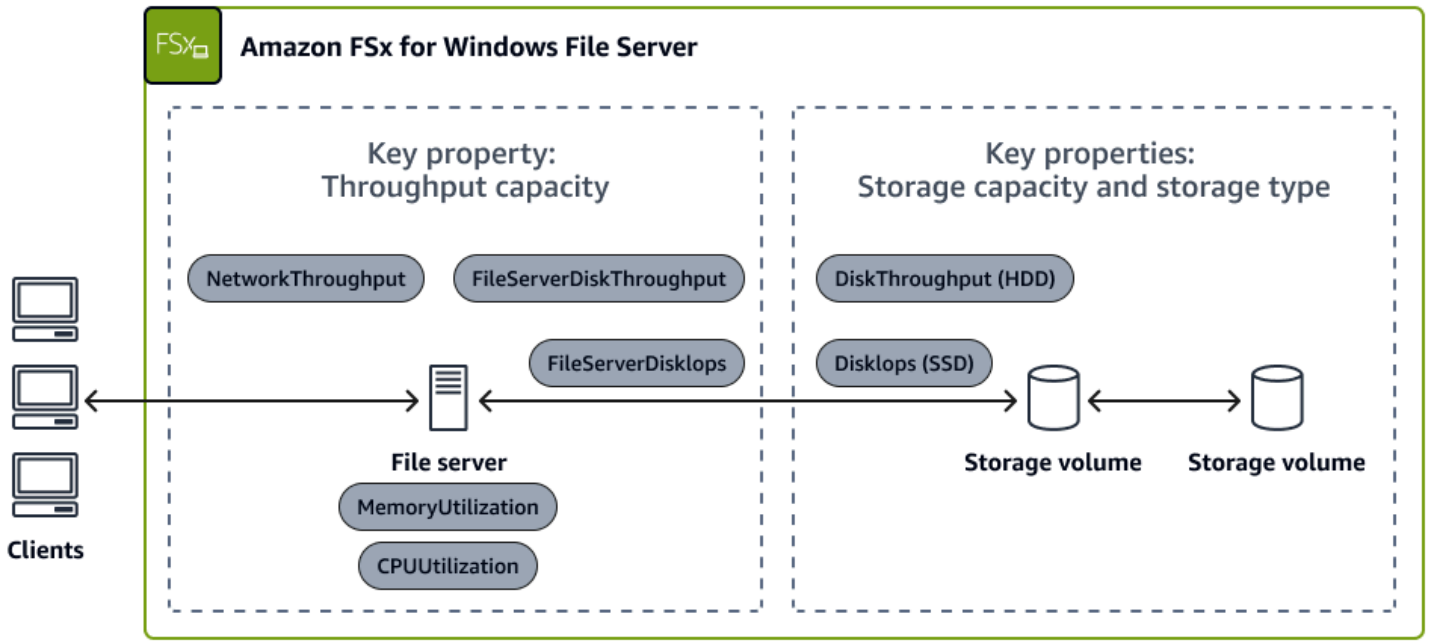
- DFS 네임스페이스를 사용하여 여러 파일 시스템 그룹화(Amazon FSx 설명서)
- 연습 6: 샤드를 사용하여 성능 확장(Amazon FSx 설명서)
- Amazon FSx for Windows File Server에서 DFS 네임스페이스 사용(AWS Labs)

Amazon FSx의 HDD 볼륨 사용량 이해

개요

Amazon FSx for Windows File Server는 파일 시스템 용량과 독립적으로 처리량을 선택할 수 있는 유연성을 제공합니다. HDD(하드 디스크 드라이브)와 SSD(솔리드 스테이트 드라이브)의 두 가지 용량 설정을 사용할 수 있습니다. EBS st1 드라이브는 HDD의 파일 시스템 스토리지에 사용됩니다. EBS io1 드라이브는 SSD에 사용됩니다.

다음 다이어그램은 처리량과 스토리지 설정 간의 관계를 보여줍니다.



HDD 기반 스토리지를 사용하면 80 버스트 디스크 IOPS(스토리지의 TiB당 IOPs)가 포함된 12 IOPS 기준과 80 버스트 메가바이트/초(스토리지의 TiB당)가 포함된 12MB/초 기준의 처리량을 얻을 수 있습니다. 예를 들어 공유의 크기가 50TB인 경우 처리량과 IOPS 모두에 대한 기준으로 50 * 12 = 600이 됩니다.

Amazon FSx for Windows File Server는 80 버스트 IOPS를 제공합니다. 버스트 크레딧은 사용률이 기준 속도보다 낮으면 자동으로 다시 채워지고 사용률이 기준 속도보다 높으면 자동으로 사용됩니다. 예

를 들어 워크로드가 한 시간 동안 10 IOPS/TB(기준 속도보다 2 IOPS/TB 낮음)만 사용하는 경우 버스트 크레딧이 다시 부족해지기 전에 다음 시간 동안 14 IOPS/TB(기준보다 2 IOPS/TB 높음)를 활용할 수 있습니다.

파일 작업의 경우 Amazon FSx for Windows File Server는 SSD 스토리지에서 밀리초 미만의 일관된 지연 시간을 제공하고 HDD 스토리지에서 10밀리초 미만의 지연 시간을 제공합니다. HDD 스토리지를 포함한 모든 파일 시스템에 대해 Amazon FSx for Windows File Server는 파일 서버에 빠른(메모리 내) 캐시를 제공하므로 스토리지 유형에 관계없이 활발하게 액세스되는 데이터에 대해 밀리초 미만의 지연 시간과 고성능을 얻을 수 있습니다.

적절한 경우 HDD 스토리지를 사용하면 전체 스토리지 용량의 비용을 절감하고 필요에 맞는 안정적인 스토리지 플랫폼을 제공하는 데 도움이 될 수 있습니다.

비용 영향

Amazon FSx for Windows File Server 성능은 스토리지 용량, 스토리지 유형 및 처리량의 세 가지 요인에 따라 달라집니다. 네트워크 I/O 성능 및 인 메모리 캐시 크기는 처리량 용량에 따라 단독으로 결정되지만 디스크 I/O 성능은 처리량 용량, 스토리지 유형 및 스토리지 용량의 조합에 따라 결정됩니다.

SSD는 I/O 집약적 워크로드에 권장되지만 HDD 성능 사양으로 요구 사항을 충족할 수 있는 다양한 워크로드가 있습니다. HDD 스토리지는 홈 디렉터리, 사용자 및 부서별 공유, 콘텐츠 관리 시스템 등 광범위한 워크로드에 맞게 설계되었습니다. 예를 들어 사용자에게 현재 프로젝트를 지원하는 데이터에 대한 짧은 지연 시간 액세스만 필요한 경우 저장 중인 대부분의 데이터에 자주 액세스하지 않습니다.

AWS Pricing Calculator를 사용하여의 HDD 파일 시스템과 20TB SSD를 비교할 수 있습니다`us-east-1`. 다음 표에서 볼 수 있듯이 중복 제거 비용 절감이 없더라도 HDD 파일 시스템을 SSD 파일 시스템과 비교할 때 비용 차이는 중요합니다.

Amazon FSx 파일 시스템 구성	월별 비용
20TB 다중 AZ SSD(`us-east-1`)	4,699.30 USD
20TB 다중 AZ HDD(`us-east-1`)	542.88 USD
예상 월별 절감액	4,156.42 USD

> **Note**
>
> 추가 FSx for Windows File Server 절감액은이 가이드의 Amazon FSx에서 데이터 중복 제거 활성화 섹션을 참조하세요.

성능 요구 사항을 올바르게 식별하면 워크로드에 적합한 스토리지를 선택하고 비용을 절감할 수 있습니다.

비용 최적화 권장 사항

HDD 스토리지를 사용하기로 결정한 경우 파일 시스템을 테스트하여 성능 요구 사항을 충족할 수 있는지 확인합니다. HDD 스토리지는 SSD 스토리지에 비해 비용이 저렴하지만 스토리지 단위당 디스크 처리량 및 디스크 IOPS 수준이 낮습니다. I/O 요구 사항이 낮은 범용 사용자 공유 및 홈 디렉터리, 데이터를 자주 검색하지 않는 대규모 콘텐츠 관리 시스템 또는 대용량 파일이 적은 데이터 세트에 적합할 수 있습니다.

기존 파일 시스템의 스토리지 유형은 변경할 수 없습니다. Amazon FSx for Windows File Server 파일 시스템의 스토리지 유형을 변환하려면 기존 파일 시스템을 백업하고 원하는 스토리지 유형의 새 파일 시스템으로 복원해야 합니다. 기존 SSD 파일 시스템을 HDD 파일 시스템으로 변환하려는 경우 HDD의 최소 용량이 2TB로 훨씬 더 높다는 점에 유의하세요.

다른 스토리지 유형의 백업을 복원하려면 다음을 수행합니다.

1. 기존 파일 시스템을 백업합니다.
2. HDD 스토리지 유형을 사용하여 새 Amazon FSx 파일 시스템을 생성합니다.
3. 원하는 스토리지 유형을 사용하여 백업을 새 파일 시스템으로 복원합니다.
4. 새 파일 시스템의 스토리지 유형이 올바르고 데이터가 손상되지 않았는지 확인합니다.

변경 사항을 프로덕션으로 이전하기 전에 Amazon FSx 파일 시스템의 성능을 분석하고 변경 사항이 허용되는지 확인하는 것이 좋습니다. 자세한 지침은 AWS 스토리지 블로그의 새 지표로 Amazon FSx for Windows File Server 성능 최적화 게시물을 참조하세요.

추가 리소스

- Amazon FSx로 비용 최적화(Amazon FSx 설명서)

단일 가용 영역 사용

개요

이 섹션에서는 Amazon FSx for Windows File Server의 단일 가용 영역 구현을 사용하는 것이 더 유용한 경우를 설명합니다. 단일 가용 영역으로 이동하면 비용이 절감되는 동시에 Amazon FSx for Windows File Server를 관리형 파일 스토리지 서비스로 사용할 수 있는 시나리오를 다룹니다. 프로덕션 워크로드용 Amazon FSx에 대해 단일 가용 영역을 구현하는 것이 좋습니다. 이렇게 하면 여러 가용 영역을 중복할 수 있습니다.

비용 영향

단일 가용 영역 파일 시스템은 여러 가용 영역 구현에 비해 약 40%의 비용 절감 효과를 제공합니다. 다중 가용 영역 파일 시스템을 사용하면 단일 가용 영역 파일 시스템에서 SSD의 경우 GB-월 0.230 USD, HDD의 경우 GB-월 0.025 USD를 지불하고, SSD의 경우 GB-월 0.130 USD, HDD의 경우 GB-월 0.013 USD를 지불합니다. 를 사용하여 비용 비교를 확인하고 자체 견적을 생성할 수 있습니다AWS Pricing Calculator.

10TB 파일 시스템의 경우 여러 가용 영역에 대해 매월 약 1,200 USD를 지불하거나 단일 가용 영역에 대해 매월 680 USD를 지불하는 차이가 있을 수 있습니다. 이 예제에서는 SSD와 함께 10TB FSx for Windows File Server 파일 시스템을 사용합니다. 중복 제거에 대한 예상 절감액은 50%입니다. 전반적으로 단일 가용 영역은 진입 비용이 낮지만 다음 섹션에서 다루는 몇 가지 주의 사항이 있습니다.

비용 최적화 권장 사항

단일 가용 영역 배포

단일 가용 영역이 적합한지 확인하려면 FSx for Windows File Server에 저장되는 데이터에 대한 자체 내부 SLAs를 고려하세요. 이를 위해서는 고객에게 제공할 SLAs(내부 및 외부)가 있는지, Amazon FSx 단일 가용 영역의 9999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999 SLAs 9 단일 가용 영역이 있는 FSx for Windows File Server의 가동 시간은 여전히 99.9%입니다. 여러 가용 영역에 대한 Amazon FSx의 SLA는 99.99%를 초과합니다. 미션 크리티컬 워크로드의 경우 추가 비용이 발생하더라도 단일 가용 영역에서 여러 가용 영역을 사용하는 것이 좋습니다.

단일 가용 영역 배포는 SQL Server 데이터베이스 백업과 같은 워크로드에 적합합니다. HDD 계층을 통해 저렴한 스토리지를 제공하는 동시에 일관된 가동 시간을 제공할 수 있습니다. 가용성이 높은 SQL 서버 또는 프로덕션 애플리케이션 액세스와 같이 프로덕션 워크로드에 더 높은 수준의 가용성이 필요한 경우 단일 가용 영역이 워크로드에 적합하지 않습니다. 백업, 비프로덕션 테스트 및 개발 환경의 경우 Amazon FSx 단일 가용 영역 구현을 통해 운영 비용을 절감할 수 있습니다.

Amazon FSx 단일 가용 영역 파일 시스템이 잘 작동하는 한 가지 사용 사례는 Always On 가용성 그룹을 사용하는 고가용성 SQL Server 클러스터의 서버당 스토리지로 여러 Amazon FSx 단일 가용 영역 파일 시스템이 사용되는 프로덕션 상황입니다. 자세한 내용은 AWS 스토리지 블로그의 게시물에서 고가용성 SQL Server 배포 비용 최적화 AWS를 참조하세요.

다중 리전 복제

Amazon FSx를 사용한 다중 리전 복제를 활용하려는 경우 단일 가용 영역 파일 시스템(단일 가용 영역 파일 시스템만 작동하는 시스템)으로 비용을 절감할 수 있는 잠재적 옵션은 입니다. 네이티브 Microsoft DFS-R에서의 사용을 지원하는 단일 AZ 파일 시스템을 배포할 수 있습니다. DFS-R에는 리전 및 여러 사이트에 데이터를 자동으로 복제할 수 있는 기능이 있습니다. Amazon FSx를 사용하여 DFS-R을 구성하는 방법에 대한 자세한 내용은 Amazon FSx 설명서의 Microsoft 분산 파일 시스템 복제 사용을 참조하세요.

다중 리전 비용 절감을 위한 또 다른 대안은를 사용하는 것입니다 AWS Storage Gateway. 이렇게 하면 Amazon FSx의 다중 리전 액세스를 위해 다른 리전에서 Amazon FSx File Gateway를 구현할 수 있습니다. FSx 자세한 내용은 이 설명서의 AWS Storage Gateway 단원을 참조하십시오.

여러 리전에서 작업하는 경우 리전 간 데이터 트래픽에 대한 데이터 전송 비용을 고려해야 합니다. 리전 간 트래픽 이동 시 0.02 USD/Gb의 요금이 발생합니다. 따라서 대량으로 데이터를 일관되게 변경하면 전체 비용이 추가됩니다. 예를 들어 1TB의 데이터 전송은 약 20.48 USD에 해당합니다.

유지보수 윈도우

Amazon FSx에서 단일 가용 영역을 사용하는 경우 유지 관리 기간은 주요 고려 사항입니다. 기본 Windows Server에 대한 일상적인 소프트웨어 패치로 인해 유지 관리 기간 동안 Amazon FSx 파일 시스템을 약 20분 동안 사용할 수 없습니다. 야간 백업에 파일 시스템을 사용하는 경우 백업 중에 중단이 발생하지 않도록 Amazon FSx 유지 관리 기간을 적절히 조정합니다. Amazon FSx 파일 시스템을 생성한 후 유지 관리 기간을 조정할 수 있습니다.

추가 리소스

- 가용성 및 내구성: 단일 AZ 및 다중 AZ 파일 시스템(Amazon FSx 설명서)
- Amazon FSx for Windows File Server 요금(AWS 웹 사이트)

AWS Storage Gateway

AWS Storage Gateway 는 온프레미스 환경을 클라우드 스토리지와 연결하는 하이브리드 AWS 클라우드 스토리지 서비스입니다. 이를 통해 기존 온프레미스 인프라를와 원활하게 통합하여 클라우

드에서 데이터를 AWS저장 및 검색하고 하이브리드 환경에서 애플리케이션을 실행할 수 있습니다. Windows 워크로드의 경우 Storage Gateway를 사용하여 SMB 및 NFS와 같은 기본 Windows 프로토콜을 사용하여 데이터를 저장하고 액세스할 수 있습니다. Storage Gateway를 사용하면 온프레미스 하드웨어 및 소프트웨어를 클라우드에 대한 브리지로 AWS 사용하여에서 Windows 워크로드 실행과 관련된 비용을 줄일 수 있습니다. 이를 통해 기존 인프라를 크게 변경 AWS 하지 않고도의 확장성과 비용 효율성을 활용할 수 있습니다.

Storage Gateway에서 Amazon S3 File Gateway, Amazon FSx File Gateway, Tape Gateway 및 Volume Gateway를 가져옵니다. S3 File Gateway 및 FSx File Gateway는 Microsoft 워크로드에서 가장 일반적으로 사용됩니다.

Amazon S3 파일 게이트웨이

Amazon S3 File Gateway를 사용하면 기존 SMB 공유를 사용하여 사용자에게 액세스 권한을 제공하면서 Amazon S3에 파일을 저장할 수 있습니다. 이를 통해 친숙한 사용자 인터페이스를 제공하고 Amazon S3에 데이터를 저장하고 다양한 Amazon S3 스토리지 계층을 활용하여 비용을 절감할 수 있습니다. S3 Intelligent Tiering을 통해 Storage Gateway를 구현하면 수명 주기 파일을 가장 저렴한 스토리지 계층으로 자동으로 이동하여 비용을 더욱 절감할 수 있습니다. 확장, 읽기 전용 액세스, 빠른 반복 읽기(캐시에서) 및 데이터베이스 덤프에는 S3 File Gateway를 사용하는 것이 좋습니다. 일반적으로 고성능 또는 고가용성 쓰기, 편집 파일 또는 부서 공유에는 권장되지 않습니다.

Amazon FSx File Gateway

Amazon FSx File Gateway는 Amazon FSx Windows 파일 시스템을 사용할 때도 비용 절감 효과를 제공할 수 있습니다. FSx File Gateway를 설치하여 다른 리전의 Amazon FSx 파일 시스템에 대한 현지화된 액세스를 제공하여 두 개의 독립 파일 시스템을 보유하는 비용을 방지할 수 있습니다. 이는 온프레미스 파일 서버가 여러 개 있고 여러 하드웨어 디바이스에 대한 비용을 지불하지 않도록 통합하려는 경우에도 유용할 수 있습니다.

비용 영향

Amazon S3 파일 게이트웨이

Storage Gateway의 시작 마법사를 사용할 수 있으므로 S3 File Gateway를 쉽게 설정할 수 있습니다. Storage Gateway AWS 환경에서 EC2 인스턴스를 사용하여 몇 분 만에 게이트웨이를 배포할 수 있습니다. 게이트웨이가 설정된 후 SMB 및 NFS 프로토콜을 통해 액세스할 수 있도록 Storage Gateway 공유를 구성할 수 있습니다. 일반적인 Windows 워크로드의 경우이 설정을 사용하여 Active Directory 환경을 활용하고 파일 공유에 대한 권한을 설정할 수도 있습니다. Storage Gateway는 일반

적인 Windows 파일 공유로 작동하므로 일반적인 사용량에 효과적으로 통합할 수 있습니다. 파일 및 폴더는 객체로 저장되고 NTFS 액세스 제어 목록(ACLs 메타데이터로 저장됩니다.

다음 표에서는 10TB 스토리지 비용을 사용 가능한 세 가지 스토리지 옵션과 비교합니다.

- FSx for Windows File Server
- Amazon S3 파일 게이트웨이
- Amazon Elastic Block Store(Amazon EBS)

Amazon S3를 사용하면 데이터를 다양한 사용 계층으로 분할할 수 있으므로 10TB의 스토리지를 훨씬 저렴한 비용으로 저장할 수 있는 요금입니다. 요금 견적에서 S3 Intelligent Tiering은 요금 유연성에 사용됩니다. 여기에는 S3 Standard의 80%, Infrequent Access의 10%, Amazon S3 Glacier의 10%가 포함됩니다. S3 Glacier를 사용할 수 있지만 S3 Glacier로 이동된 파일에 즉시 액세스할 필요가 없도록 적절한 수명 주기 규칙을 설정하는 것이 중요합니다. S3 Glacier는 일반적인 액세스 사용량이 아니라 순전히 아카이브 사용량을 위한 것입니다.

스토리지 시스템	10TB 스토리지 비용	리전
FSx for Windows File Server(중복 제거 비용 50% 절감)	683.20 USD SSD	미국 동부(버지니아 북부)
Amazon S3 파일 게이트웨이	449.51 USD Intelligent Tiering	미국 동부(버지니아 북부)
Amazon EBS	1,335.69 USD GP3	미국 동부(버지니아 북부)

다음을 고려하세요.

- S3 Glacier에서는 RestoreObject API를 사용하여 객체를 Amazon S3로 복원하지 않는 한 일반 I/O 오류가 발생합니다. Amazon CloudWatch Events를 사용하여이 I/O 오류에 대한 알림을 사용하는 것이 좋습니다. 이렇게 하면 운영 팀이 액세스해야 할 수 있는 파일에 대해이 오류를 받는 사용자에게 대응할 수 있습니다. 이러한 오류에 대한 자세한 내용은 Amazon S3 File Gateway 설명서의 Error: InaccessibleStorageClass를 참조하세요.
- 액세스에 대한 S3 Glacier 제한 외에도 Storage Gateway에서 객체/폴더당 허용되는 ACLs은 10개뿐입니다. Storage Gateway를 사용하기 전에 ACL 항목이 10개 이상 필요하지 않은지 확인합니다.

Amazon FSx File Gateway

Amazon S3 File Gateway와 마찬가지로 FSx File Gateway는 데이터를 장기간 보관하는 파일 시스템에 대한 액세스를 제공합니다. Amazon S3 File Gateway에서 데이터는 Amazon S3에 상주합니다. FSx File Gateway의 경우 데이터는 FSx for Windows File Server에 상주합니다. FSx for Windows File Server에 다중 AZ 옵션을 사용할 수 있지만 다중 리전 옵션은 없습니다. 글로벌 회사 또는 원격 사무실이 있는 경우 지연 시간을 방지하기 위해 최종 사용자에게 지리적으로 더 가까운 공유 스토리지 플랫폼을 제공해야 할 수 있습니다. 다른 Amazon FSx 파일 시스템을 배포하는 경우 완전히 새로운 Amazon FSx for Windows File Server 파일 시스템과 필요한 스토리지 비용이 추가됩니다. 완전히 새로운 파일 시스템을 생성하고 비용을 중복하지 않도록 보조 리전에 FSx File Gateway를 배포할 수 있습니다. 이를 통해 사용자에게 파일에 대한 현지화된 액세스를 제공하는 동시에 전체 비용을 절감할 수 있습니다.

스토리지 시스템	10TB 스토리지 비용	리전
Amazon FSx for Windows File Server	683.20 USD SSD	미국 동부(버지니아 북부)
Amazon FSx File Gateway	$503.70/단일 게이트웨이	미국 동부(버지니아 북부)

> **Note**
>
> 위 표의 가격은 Storage Gateway 요금을 기준으로 합니다.

다음 사항에 유의하세요.

- FSx File Gateway를 사용하면 다중 리전 워크로드에 대해 매월 약 180 USD(또는 연간 2,100 USD)를 절약할 수 있습니다.
- FSx File Gateway를 사용하면 데이터 전송 요금이 훨씬 저렴합니다. 액세스하는 파일을 정기적으로 캐싱하기만 하면 되며 전체 보조 복사본이 아니기 때문입니다.
- 서로 다른 리전에 FSx for Windows File Server를 두 개 배포하여 AWS Backup 또는 로 업데이트할 수 있지만 AWS DataSync두 옵션 모두 거의 실시간으로 제공되지 않습니다.

비용 최적화 권장 사항

Amazon S3 파일 게이트웨이

S3 File Gateway는 파일 저장을 위한 저렴한 옵션을 제공하지만 파일 시스템을 구현하고 사용하는 방법과 관련하여 고려해야 할 몇 가지 문제가 있습니다. 예를 들어 S3 File Gateway는 Storage Gateway 소프트웨어를 실행하기 위해 가상 머신을 사용해야 합니다. 에서 AWS Storage Gateway는 기본적으로 m5.xlarge 인스턴스를 사용하여 Amazon EC2에 배포됩니다. 온프레미스 스토리지 비용을 줄이려면 Storage Gateway를 VMware 및 Hyper-V와 같은 가상화 플랫폼에 가상 어플라이언스로 배포할 수 있습니다.

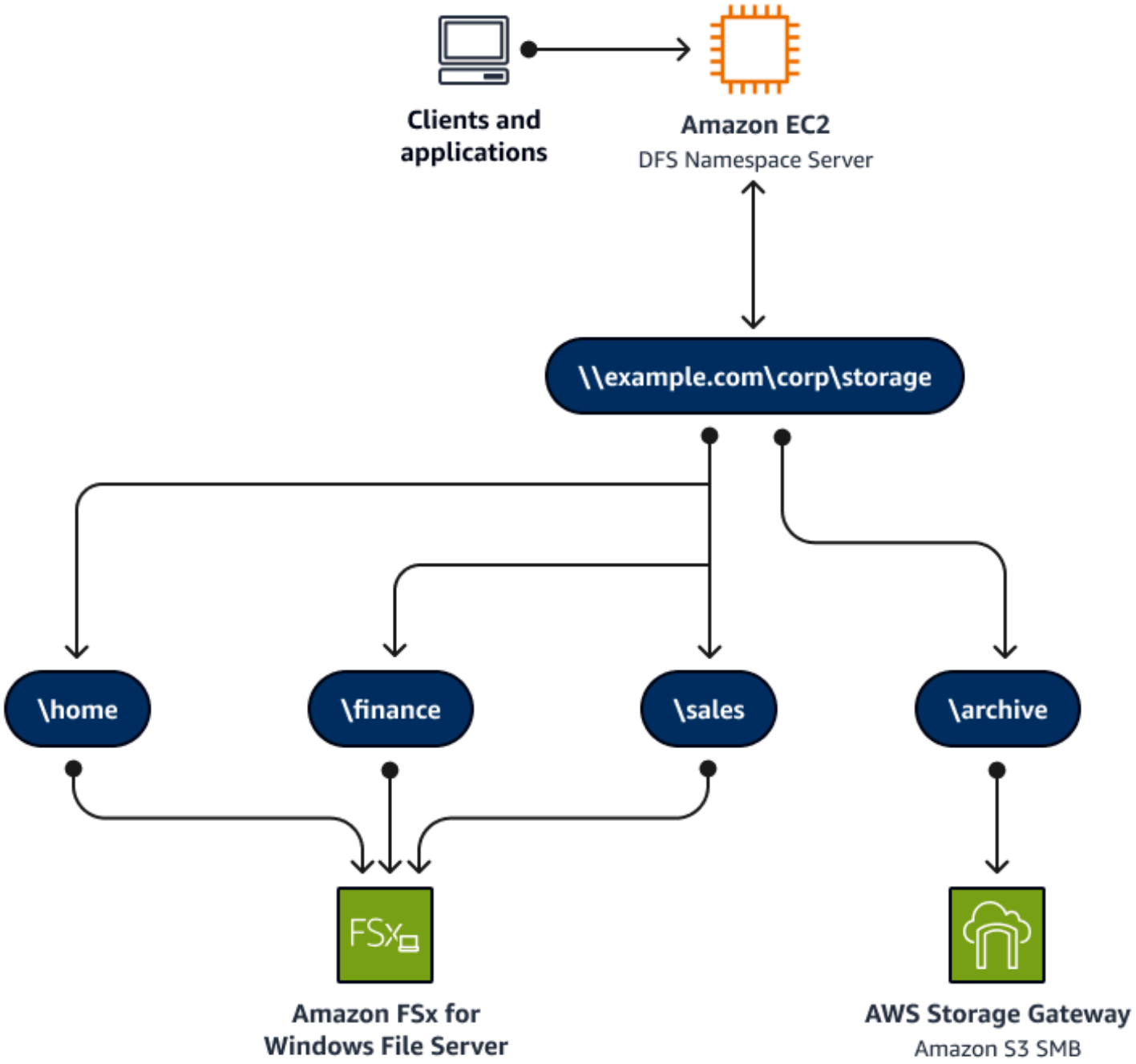
고가용성 고려 사항

Storage Gateway를 실행하는 것은 파일에 대한 액세스에 대한 단일 장애 지점입니다. 불필요한 가동 중지를 방지하려면 사용자가 Storage Gateway 인스턴스를 변경하거나 중지 및 시작할 수 있도록 엄격한 액세스 제어를 구현하는 것이 좋습니다. 또한의 배포의 경우 Amazon Data Lifecycle Manager를 사용하여 라우팅 스냅샷을 생성하여 Storage Gateway 구현을 빠르게 복구하는 AWS것이 좋습니다. VMware를 사용하여 온프레미스에서 Storage Gateway를 실행하는 경우 고가용성을 위해 구성할 수 있습니다.

여러 파일 시스템 실행

일일 사용 파일 워크로드를 아카이브 워크로드와 분리하면 불필요한 스토리지 비용을 피할 수 있습니다. Storage Gateway는 FSx for Windows File Server 파일 시스템과 함께 배포할 수 있습니다. DFS 네임스페이스를 사용하면 FSx for Windows File Server에서 실행되는 기본 일일 사용 스토리지와 Amazon S3(Storage Gateway를 통해 액세스)에서 실행되는 스토리지를 제공할 수 있습니다.

다음 다이어그램은 단일 DFS 네임스페이스를 다양한 백엔드 스토리지 옵션의 프런트엔드 액세스 포인트로 사용하는 방법을 보여줍니다.



클라이언트는 \\example.com\storage와 같은 폴더 구조로 이동합니다. 이 기본 디렉터리에는 하위 디렉터리가 포함되어 있습니다. FSx for Windows File Server 파일 시스템에는 일반적으로 액세스되는 파일 공유가 포함되어 있습니다. Storage Gateway에서 생성된 파일 공유를 아카이브 데이터에 사용할 수 있습니다. 사용자는 항목을 아카이브 폴더에 수동으로 아카이브하거나 일부 파일을 일반 파일 공유에서 아카이브 폴더로 이동하는 프로세스를 구축할 수 있습니다.

다음을 고려하세요.

- 스토리지 요구 사항을 검토하고 캐시에 적합한 스토리지를 제공합니다.
- 게이트웨이를 Active Directory 구성에 추가하고 파일에 액세스하기 위해 표준 Windows ACLs을 사용합니다.

FSx File Gateway

FSx File Gateway의 배포는 S3 File Gateway의 배포와 비슷하지만 시작 마법사를 사용하면 훨씬 더 쉽습니다. 자세한 지침은 Amazon FSx File Gateway 설명서의 3단계: Amazon FSx File Gateway 생성 및 활성화를 참조하세요. FSx 환경에 FSx File Gateway를 배포한 후 이를 기존 Amazon FSx 파일 시스템에 연결하고 파일에 액세스할 수 있습니다.

FSx File Gateway를 배포할 때 스토리지가 가장 중요한 고려 사항입니다. 기본 스토리지는 150GB를 제공하며, 이는 파일을 캐싱하기 위한 적절한 공간입니다. 여유 공간이 적을 때 모니터링 알림을 생성하면 과다 할당 없이 스토리지 크기를 적절하게 조정하는 데 도움이 될 수 있습니다.

추가 리소스

- AWS Storage Gateway 리소스(AWS 문서)

Active Directory

Windows Server를 실행하는 Amazon Elastic Compute Cloud(Amazon EC2)는 Windows 기반 애플리케이션 및 워크로드를 배포하기 위한 안전하고 신뢰할 수 있는 고성능 환경입니다. 인스턴스를 빠르게 프로비저닝하고 필요에 따라 확장하거나 축소할 수 있으며 사용한 만큼만 비용을 지불할 수 있습니다. Active Directory 서비스는 Windows Server 환경에서 자격 증명 관리의 기본 소스로 사용됩니다.

이 섹션은 다음 주제를 포함합니다.

- Amazon EC2의 자체 관리형 Active Directory
- AWS Managed Microsoft AD
- AD Connector

Amazon EC2의 자체 관리형 Active Directory

개요

이 섹션에서는 Amazon Elastic Compute Cloud(Amazon EC2)에서 Active Directory를 실행하는 비용을 줄이기 위한 권장 사항을 제공합니다. 주요 초점은 Active Directory 도메인 컨트롤러의 크기를 적절하게 조정하고의 유연성을 사용하여 환경에 맞게 AWS 클라우드 조정할 수 있도록 하는 것입니다.는 변화하는 요구 사항에 맞게 인스턴스를 쉽게 중지하고 크기를 조정하거나 너무 빠르게 스케일 업하는 경우 인스턴스의 크기를 줄이는 데 AWS 도움이 될 수 있습니다. 올바른 인스턴스 크기와 유형을 선택하면 상당한 비용 절감 효과를 얻을 수 있습니다.

비용 영향

다음 표는 범용 인스턴스보다 버스트 가능한 인스턴스 패밀리 인스턴스를 선택하는 차이점을 보여줍니다. 이 선택을 통해 매달 상당한 금액을 절약할 수 있습니다. 인스턴스를 적절하게 계획하고 크기를 조정하면 비용을 관리하는 데 도움이 될 수 있습니다.

인스턴스 유형	인스턴스 개수	vCPU	메모리	비용
t3a.medium	2	2	8	$81.76/월
m5a.large	2	2	8	$259.88/월

비용에 대한 자세한 내용은 AWS Pricing Calculator 견적을 참조하세요.

매월 178.12 USD의 절감액은 도메인 컨트롤러에 대해 연간 2,000 USD 이상의 절감액이 됩니다. 는 한 계정에서 도메인 컨트롤러가 두 개뿐인 작은 설치 공간용이라는 점에 유의하세요. 여러 계정 및 추가 도메인 컨트롤러를 사용하여 규모에 따라 이러한 절감액은 상당한 비용 절감 효과를 가져올 수 있습니다.

비용 최적화 권장 사항

Microsoft는 Active Directory 환경을 배포할 때에 대한 용량 계획 권장 사항을 제공합니다. Active Directory 환경을 계획하거나 확장할 때 다음 주요 구성 요소를 고려하는 것이 좋습니다.

- 메모리
- Network
- 스토리지
- 처리자

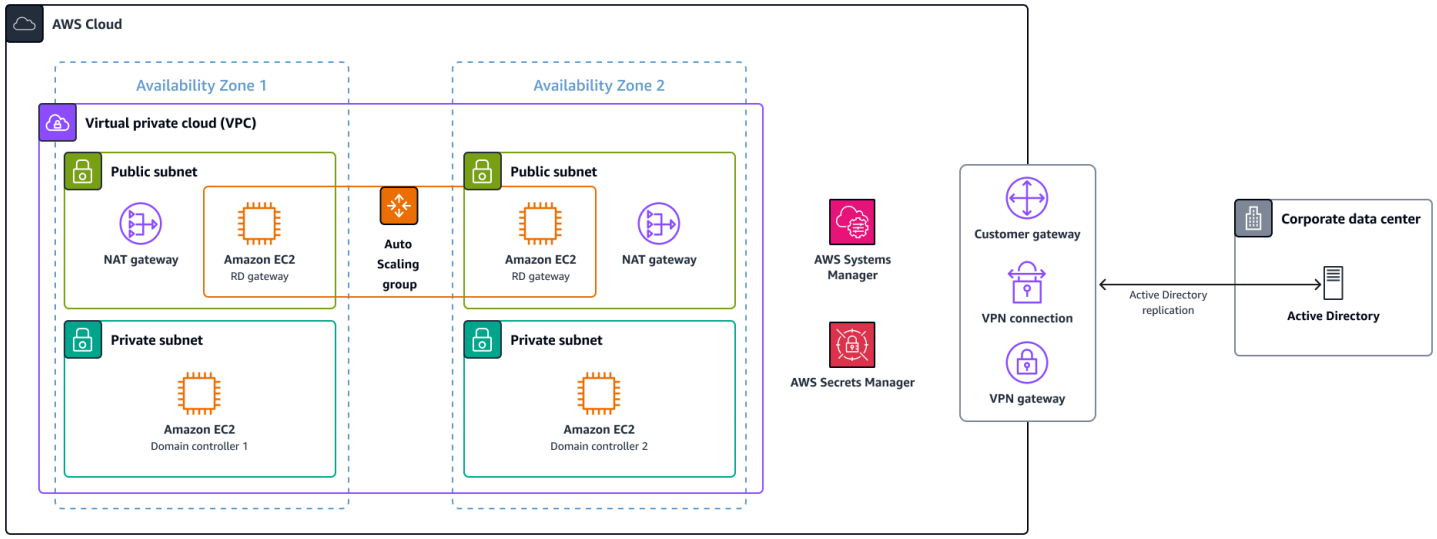
이러한 주요 구성 요소를 염두에 두고 Active Directory 환경에 적합한 인스턴스 유형을 선택할 수 있습니다 AWS. 이 섹션에서는 몇 가지 Active Directory에서 AWS 배포 시나리오에 대한 예제를 다룹니다. 이러한 시나리오를 사용하면 온프레미스 환경에서와 동일한 수의 사용자와 컴퓨터를 처리할 계획이 없는 AWS경우에서 온프레미스 환경을 복제할 필요가 없다는 점을 명확히 할 수 있습니다.

다음 표에서는 설치 AWS 공간의 vCPU, 메모리 및 디스크와 관련된 중요한 구성 요소를 강조합니다.

구성 요소	견적
스토리지/데이터베이스 크기	사용자당 40~60KB
RAM	데이터베이스 크기 기본 운영 체제 권장 사항 타사 애플리케이션
Network	1GB
CPU	코어당 동시 사용자 1,000명

하이브리드 배포 시나리오

다음 다이어그램은 Active Directory의 하이브리드 배포를 위한 아키텍처 예제를 보여줍니다.



다이어그램에서 볼 수 있듯이 일반적으로 온프레미스 풋프린트가 있고 이를 로 확장합니다 AWS 클라우드. 마이그레이션의 초기 단계에서는 일반적으로 모든 사용자와 서버가 배포되지 않습니다 AWS. 따라서 처음에는 더 작은 크기의 공간을 배포하여 마이그레이션 작업 비용을 절감하는 것이 중요합니다.

온프레미스에서 인증하는 서버와 사용자가 있는 온프레미스 공간을 유지하려는 경우 도메인 컨트롤러에 대해 동일한 공간이 필요하지 않습니다 AWS. Active Directory 모범 사례에 따라 적절한 Active Directory 사이트 및 서비스를 구현하여 사용자와 컴퓨터를 온프레미스 풋프린트에 인증하는 동시에의 도메인 컨트롤러에 대한 풋프린트만 인증할 수 있습니다 AWS AWS. 이렇게 하면 모든 온프레미스 인프라가 아닌 AWS 리소스로만 사용을 제한 AWS 하여 Active Directory 설치 공간의 크기를 과도하게 조정하지 않도록 할 수 있습니다. 하이브리드 설정 설계에 대한 지침은 Microsoft 설명서의 도메인 컨트롤러의 적절한 배치 및 사이트 고려 사항을 참조하세요.

적절한 크기 조정을 통한 AWS 마이그레이션 최적화

사용자를 위해 Active Directory의 새 인스턴스를 배포하거나 Active Directory 인프라를 AWS 위해 로 완전히 마이그레이션할 계획인 경우 앞의 표에서 선택한 인스턴스에 대한 vCPU, 메모리 및 디스크 공간에 대한 Microsoft의 권장 사항에 따라 크기 조정을 계획하는 것이 좋습니다.

새로운 풋프린트인 경우 작은 크기로 시작하고 인스턴스 유형을 쉽게 변경하여 환경이 확장될 때 크기를 조정할 수 있는 기능을 활용할 수 있습니다 AWS. 이 설명서의 Amazon EC2의 Windows 섹션에서는 CPU 및 메모리 사용률을 모니터링하고 검토하는 방법을 보여줍니다 AWS. 이렇게 하면 EC2 인스턴스의 크기를 늘려야 하는 시점을 알 수 있습니다.

온프레미스 Active Directory 환경을 로 완전히 마이그레이션 AWS하는 경우 동일한 크기 조정 계획을 구현하여 적절한 성능을 보장할 수 있습니다. 의 온프레미스에 있는 내용을 복제하기 전에 Active Directory 환경을 철저히 검토하는 AWS것이 좋습니다. 이렇게 하면 과다 프로비저닝을 방지하는 데 도움이 될 수 있습니다. 성능 모니터를 사용하여 기존 도메인 컨트롤러의 트래픽 양 및 사용률에 대한 정보를 수집해야 합니다. 이를 통해 전체 사용량을 이해할 수 있으므로 적절한 크기를 설정하고 궁극적으로 비용을 절감할 수 있습니다.

에서 Active Directory 최적화 AWS

Active Directory를 실행하는 경우 사용률을 지속적으로 모니터링하고 필요에 따라 인스턴스 크기를 변경하여 지출을 줄이는 AWS것이 중요합니다. AWS Compute Optimizer 를 사용하여 실행 중인 리소스에 대한 정보를 가져올 수 있습니다 AWS. Compute Optimizer를 사용하여 Windows 워크로드의 크기를 조정하는 방법에 대한 자세한 내용은이 가이드의 Amazon EC2의 Windows 섹션을 참조하세요. 보다 포괄적인 심층 분석을 위해 성능 모니터를 사용하여 Active Directory 도메인 컨트롤러의 사용률을 모니터링하고 성능을 평가한 다음 그에 따라 크기를 조정할 수 있습니다.

CloudWatch를 사용하여 도메인 컨트롤러의 성능을 모니터링할 수도 있습니다. 도메인 컨트롤러를 최적화(확장 또는 축소)하려면 CloudWatch에서 사용할 수 있는 지표를 사용하여 올바른 결정을 내릴 수 있습니다. CloudWatch 에이전트를 사용하여 데이터 수집을 위해 전송할 사용자 지정 성능 모니터 지표를 구성할 수 있습니다. 지침은 AWS 지식 센터의 CloudWatch 에이전트를 사용하여 Windows 서버의 성능 모니터에 대한 지표를 보려면 어떻게 해야 합니까?를 참조하세요.

CloudWatch 에이전트를 배포한 후의 에이전트 구성 파일 내에서 `metrics_collected`다음 지표를 구성할 수 있습니다.

지표 범주	메트릭 이름
데이터베이스에서 인스턴스로(NTDSA)	데이터베이스 캐시 % 적중
I/O 데이터베이스 읽기 평균 지연 시간	
I/O 데이터베이스 읽기/초	
I/O 로그 쓰기 평균 지연 시간	
디렉터리 서비스(NTDS)	LDAP 바인딩 시간
DRA 보류 복제 작업	
DRA 보류 복제 동기화	

지표 범주	메트릭 이름
DNS	재귀 쿼리/초
재귀 쿼리 실패/초	
TCP 쿼리 수신/초	
수신된 총 쿼리/초	
전송된 총 응답/초	
UDP 쿼리 수신/초	
LogicalDisk	평균 디스크 대기열 길이
% 여유 공간	
메모리	사용 중인 커밋 바이트 비율
장기 평균 대기 캐시 수명(들)	
네트워크 인터페이스	전송된 바이트/초
Bytes Received/sec	
현재 대역폭	
NTDS	ATQ 예상 대기열 지연
ATQ 요청 지연 시간	
DS 디렉터리 읽기/초	
DS 디렉터리 검색/초	
DS 디렉터리 쓰기/초	
LDAP 클라이언트 세션	
LDAP 검색/초	

지표 범주	메트릭 이름
LDAP 바인딩 성공/초	
처리자	프로세서 시간 %
보안 시스템 전체 통계	Kerberos 인증
NTLM 인증	

추가 리소스

- 의 Active Directory 도메인 서비스 AWS: 파트너 솔루션 배포 가이드(AWS 문서)
- Active Directory 도메인 서비스에 대한 용량 계획(Microsoft 설명서)
- EC2 인스턴스에서 Active Directory를 실행하기 위한 설계 고려 사항(AWS 백서)

AWS Managed Microsoft AD

개요

AWS Directory Service for Microsoft Active Directory라고도 하는 AWS Managed Microsoft AD는 Windows Server Active Directory에서 구동되며에서 관리합니다 AWS. AWS Managed Microsoft AD 를 사용하여 광범위한 Active Directory 인식 애플리케이션을 로 마이그레이션할 수 있습니다 AWS 클라우드. AWS Managed Microsoft AD 는 다양한 기본 Active Directory 애플리케이션 및 서비스와 함께 작동합니다. 또한 AWS 관리형 애플리케이션 및 서비스를 지원합니다. 서비스 및 결제 메커니즘 AWS Managed Microsoft AD 으로 인해에 대한 비용 최적화 레버가 많지는 않지만 비용을 최소화하는 데 도움이 되는 몇 가지 설계 원칙이 있습니다.

비용 영향

AWS Managed Microsoft AD 는 현재 SKUs를 기반으로 하는 관리형 서비스이므로 크기 조정은 비교적 간단한 프로세스입니다. 현재 사용 가능한 크기 조정 SKUs는 Standard 및 Enterprise 에디션입니다. 다른 SKUs에는 디렉터리 공유, 추가 도메인 컨트롤러(추가 리전 포함) 추가, 리전 간 데이터 전송이 포함됩니다.

비용 최적화 권장 사항

AWS Managed Microsoft AD Standard Edition과 AWS Managed Microsoft AD Enterprise Edition에는 차이가 있습니다. Enterprise Edition은 최대 500,000개의 Active Directory 객체, 125개의 계정 공유(소프트 제한)를 지원하며 다중 리전을 지원합니다. Standard Edition은 최대 30,000개의 Active Directory 객체, 5개의 계정 공유(최대 약 30개로 소프트 제한)를 지원하며 다중 리전을 지원하지 않습니다.

디렉터리 유형을 선택하기 전에 고려해야 할 질문은 다음과 같습니다.

- 다중 리전 지원이 필요합니까?
- 디렉터리가 30개 이상의 계정과 공유됩니까?
- Active Directory 객체 수가 30,000개를 초과합니까?

위의 질문 중 하나에 대한 답변이 예인 경우 Enterprise Edition이 필요합니다. 모든 질문에 대한 답변이 아니요인 경우 Standard Edition으로 시작하는 것이 좋습니다.

> **Note**
>
> 디렉터리를 Standard Edition에서 Enterprise Edition으로 업그레이드할 수 있지만 디렉터리는 다운그레이드할 수 없습니다. Standard Edition 배포는 단방향 문을 통과하지 않습니다. 디렉터리를 Enterprise Edition으로 업그레이드하려면에 문의하세요 AWS.

Enterprise Edition에서 AWS Managed Microsoft AD 디렉터리를 공유할 때 각 공유에 대한 비용이 발생합니다. 이는 각 계정에 디렉터리를 배포하는 비용보다 적지만 선택하지 않으면 공유 비용이 증가할 수 있습니다. Amazon Relational Database Service(Amazon RDS) 및 Amazon FSx for Windows File Server가 포함된 계정과만 디렉터리를 공유하는 것이 좋습니다. 이러한 서비스만이 기능을 지원하기 때문입니다. 를 포함하여 FSx for Windows File Server를 자체 관리형 Active Directory와 통합할 수 있습니다 AWS Managed Microsoft AD. 다른 계정에 Amazon FSx만 필요한 경우 디렉터리를 공유할 필요 없이에 AWS Managed Microsoft AD 대해 자체 관리형 Amazon FSx 배포를 수행할 수 있습니다.

추가 도메인 컨트롤러를 배포할 시기를 결정할 때는가 동일한 VPC의 별도의 가용 영역에 있는 두 개의 서브넷만 AWS Managed Microsoft AD 지원한다는 점에 유의하세요. 도메인 컨트롤러를 추가해도 서브넷을 추가할 수 없습니다. 성능 문제로 인해 도메인 컨트롤러를 추가해야 하는지 확인하려면 CloudWatch에서 도메인 컨트롤러 성능 지표를 검토하세요. 이렇게 하면 하나 또는 모든 도메인 컨트롤러가 압도당하고 있는지 알 수 있습니다. 하나의 도메인 컨트롤러만 압도당하고 있다고 판단되면 추가 도메인 컨트롤러를 추가해도 로드가 완화되지 않으므로 현재 사용 가능한 도메인 컨트롤러에서 로

드 밸런싱이 이루어지지 않는 애플리케이션을 더 자세히 살펴보아야 합니다. 모든 도메인 컨트롤러가 많이 사용되는 경우 추가 도메인 컨트롤러를 추가하면 기존 도메인 컨트롤러의 부하가 감소할 수 있습니다. 조정을 자동화하는 방법에 대한 지침은 AWS 보안 블로그의 사용률 지표를 기반으로 AWS Managed Microsoft AD 조정을 자동화하는 방법을 참조하세요.

디렉터리를 여러 리전으로 확장한 경우 파일 스토리지에 디렉터리 NETLOGON 또는 SYSVOL 공유를 사용하지 않는 것이 좋습니다. 모든 도메인 컨트롤러는 해당 공유의 콘텐츠를 복제합니다. 파일 스토리지에 공유를 사용하지 않으면 데이터 전송 비용이 최소화됩니다.

또한 와의 엔터프라이즈 계약에 등록할 수 있는 옵션도 있습니다 AWS. 엔터프라이즈 계약은 필요에 가장 적합한 계약을 조정할 수 있는 옵션을 제공합니다. 자세한 내용은 엔터프라이즈 고객을 참조하세요.

추가 리소스

- AWS Managed Microsoft AD 할당량(AWS Directory Service 문서화)
- AWS Directory Service 요금(AWS 웹 사이트)
- 의 Active Directory 도메인 서비스 AWS(AWS 백서)

AD Connector

개요

AD Connector는 클라우드에서 정보를 캐싱하지 않고도 Amazon WorkSpaces, Amazon QuickSight, Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 원활한 도메인 조인과 같은 호환 AWS 애플리케이션에 기존 온프레미스 Microsoft Active Directory를 쉽게 연결할 수 있는 프록시 서비스입니다. AD Connector를 사용하여 Active Directory에 하나의 서비스 계정을 추가할 수 있습니다. AD Connector를 사용하면 디렉터리 동기화 또는 페더레이션 인프라 호스팅의 비용과 복잡성이 필요하지 않습니다. 서비스의 특성과 결제 메커니즘으로 인해 AD Connector에 대한 비용 최적화 레버는 많지 않지만이 섹션의 설계 권장 사항에 따라 비용을 최소화할 수 있습니다.

비용 영향

AD Connector는 사전 설정된 SKUs. 따라서 크기 조정이 간단합니다. 사용 가능한 크기 조정 SKUs는 작은 크기와 큰 크기 두 가지입니다. AD Connector와 관련된 비용 추정AWS Pricing Calculator에를 사용할 수 있습니다.

비용 최적화 권장 사항

백엔드 컴퓨팅 리소스 외에는 작은 커넥터 크기와 큰 커넥터 크기 간에 차이가 없습니다.

디렉터리 유형을 선택하기 전에 고려해야 할 질문은 다음과 같습니다.

- AD Connector와 통합된 AWS 애플리케이션을 사용하는 활성 사용자가 많습니까(10,000명 이상)?
- 사용자가 여러 중첩 그룹, 심층 중첩 그룹 또는 순환 중첩 그룹의 멤버입니까?

두 질문 모두에 대한 답변이 아니요인 경우 작은 크기로 시작하는 것이 좋습니다. 위의 질문 중 하나에 예라고 답한 경우 큰 크기를 고려할 필요가 있을 수 있습니다. 작은 크기의 AD 커넥터로 시작할 수 있으며 성능으로 인해 디렉터리가 손상된 경우 디렉터리를 큰 크기로 업그레이드하도록 요청할 수 있습니다.

> **Note**
>
> AD 커넥터를 스몰에서 라지로 업그레이드할 수 있지만 AD 커넥터는 다운그레이드할 수 없습니다.

대부분의 성능 문제는 AD Connector와 관련이 없지만 많은 사용자가 많은, 깊은 또는 순환 중첩 그룹의 멤버이기 때문에 온프레미스 Active Directory 도메인 컨트롤러가 압도됩니다.

또한 와의 엔터프라이즈 계약에 등록할 수 있는 옵션도 있습니다 AWS. 엔터프라이즈 계약은 필요에 가장 적합한 계약을 조정할 수 있는 옵션을 제공합니다. 자세한 내용은 엔터프라이즈 고객을 참조하세요.

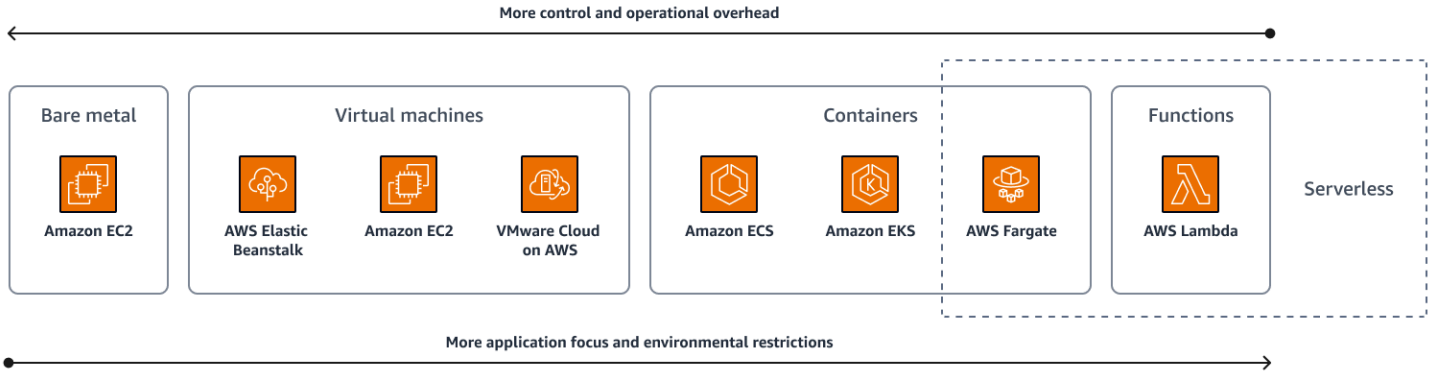
추가 리소스

- AD Connector 할당량(AWS Directory Service 문서화)
- 기타 디렉터리 유형 요금(AWS 웹 사이트)
- 의 Active Directory 도메인 서비스 AWS(AWS 백서)

.NET

.NET 애플리케이션 개발 및 배포는 클라우드 컴퓨팅에서 제공하는 규모와 민첩성을 달성하는 데 도움이 되는 중요한 키입니다. 많은 레거시 .NET 애플리케이션의 경우에서 애플리케이션을 실행하는 데 가장 적합한 컴퓨팅 선택 AWS 은 AWS Elastic Beanstalk 또는 Amazon Elastic Compute Cloud(Amazon EC2)를 통해 가상 머신을 사용하는 것입니다. Windows 및 Linux 컨테이너에서 .NET 애플리케이션을 실행할 수도 있습니다.

.NET 코어를 도입하면 모든 클라우드 이점을 활용하는 최신 .NET 애플리케이션을 설계할 수 있습니다. 최신 애플리케이션은 기존의 컴퓨팅 선택 세트를 사용할 수 있으며 AWS Fargate 또는를 비롯한 다양한 유형의 서버리스 환경을 대상으로 할 수도 있습니다 AWS Lambda. 이제 .NET 6+는 Graviton2 EC2 패밀리와 같은 ARM64 EC2 인스턴스에서 워크로드의 고성능 호스팅을 제공합니다. 이렇게 하면 Amazon EC2에서 사용할 수 있는 최신 세대의 프로세서에 액세스할 수 있습니다. 즉, 비디오 인코딩, 웹 서버 및 고성능 컴퓨팅(HPC)과 같은 워크로드 유형에 맞는 컴퓨팅에서 애플리케이션을 호스팅할 수 있습니다.



이 섹션에서는 비용 효율성에 중점을 두고 클라우드의 이점을 활용하도록 .NET 애플리케이션을 조정하는 데 도움이 되는 권장 사항을 제공합니다.

이 섹션은 다음 주제를 포함합니다.

- 최신 .NET으로 리팩터링하고 Linux로 전환
- .NET 앱 컨테이너화
- Graviton 인스턴스 및 컨테이너 사용
- 정적 .NET Framework 앱에 대한 동적 조정 지원
- 캐싱을 사용하여 데이터베이스 수요 감소
- 서버리스 .NET 고려
- 특별히 구축된 데이터베이스 고려

최신 .NET으로 리팩터링하고 Linux로 전환

개요

레거시 .NET Framework 앱을 현대화하면 보안, 성능 및 확장성을 개선하는 데 도움이 될 수 있습니다. .NET Framework 앱을 현대화하는 효과적인 방법은 앱을 최신 .NET 버전(6+)으로 마이그레이션하는 것입니다. 다음은 이러한 애플리케이션을 오픈 소스 .NET으로 이동할 때 얻을 수 있는 몇 가지 주요 이점입니다.

- Linux 운영 체제에서 실행하여 Windows 라이선스 비용을 줄이려면
- 최신 언어의 가용성 활용
- Linux에서 실행하도록 최적화된 성능 확보

많은 조직에서 여전히 이전 버전의 .NET Framework를 실행하고 있습니다. 이는 이전 버전의 취약성이 더 이상 Microsoft에서 해결되지 않기 때문에 보안 위험을 초래할 수 있습니다. Microsoft는 최신 버전의 .NET Framework 4.5.2, 4.6 및 4.6.1에 대한 지원을 종료했습니다. 이전 버전의 프레임워크를 계속 실행할 경우의 위험과 이점을 평가하는 것이 매우 중요합니다. 위험을 줄이고 비용을 절감하려면 .NET의 최신 버전으로 리팩터링하는 데 시간과 노력을 투자하는 것이 좋습니다.

비용 영향

컴퓨팅, 메모리 및 네트워킹 리소스의 균형을 제공하는 범용 EC2 인스턴스 유형(m5)을 고려합니다. 이러한 인스턴스는 웹 서버, 중간 규모의 데이터베이스 및 소스 코드 리포지토리와 같은 다양한 애플리케이션에 적합합니다.

예를 들어 미국 동부(버지니아 북부)의 Windows Server(라이선스 포함)에 vCPUs 4개와 16GB 메모리가 있는 온디맨드 m5.xlarge 인스턴스의 경우 매월 274.48 USD가 부과됩니다. Linux 서버의 동일한 리소스는 매월 140.16 USD입니다. 이 예제에서는 애플리케이션을 .NET Framework에서 최신 버전의 .NET으로 마이그레이션하고 Linux 서버에서 애플리케이션을 실행할 때 비용이 49% 절감됩니다. 비용은 EC2 인스턴스를 선택할 때 선택하는 옵션(예: 인스턴스 유형, 운영 체제, 스토리지)에 따라 달라질 수 있습니다. Savings Plans 또는 예약 인스턴스를 사용하여 비용을 추가로 최적화할 수 있습니다. 자세한 내용은를 사용하여 비용 견적AWS Pricing Calculator을 실행합니다. Windows 포함 인스턴스의 경우 요금 모델에 관계없이 vCPU당 시간당 0.046 USD의 라이선스 비용이 부과됩니다.

이러한 .NET Framework 애플리케이션을 최신 .NET으로 이식하려면 개발자의 노력이 필요합니다. 애플리케이션과 해당 종속성을 평가하여 대상 플랫폼 버전과 호환되는지 확인해야 합니다. AWS Porting Assistant for .NET은 .NET Framework 애플리케이션을 스캔하고 .NET 호환성 평가를 생성하여 Linux와 더 빠르게 호환되도록 애플리케이션을 이식하는 데 도움이 되는 보조 도구입니다. Porting Assistant

for .NET은 .NET과의 비호환성을 식별하고, 알려진 대체 항목을 찾고, 자세한 호환성 평가를 생성합니다. 솔루션을 이식한 후 프로젝트를 종속성으로 성공적으로 컴파일하려면 수동으로 코드를 변경해야 합니다. 이렇게 하면 애플리케이션을 Linux로 현대화하는 데 드는 수작업이 줄어듭니다. 애플리케이션이 ARM 프로세서를 지원하는 경우 Linux로 이동하면 Graviton 인스턴스를 사용할 수 있습니다. 이를 통해 추가 비용 절감을 20% 더 달성할 수 있습니다. 자세한 내용은 AWS 컴퓨팅 블로그의 Powering .NET 5 with AWS Graviton2: Benchmarks를 참조하세요.

레거시 AWS .NET 프레임워크 애플리케이션을 최신 .NET으로 이식하는 데 도움이 되는 Toolkit for .NET Refactoring 및 .NET Upgrade Assistant와 같은 다른 도구가 있습니다.

비용 최적화 권장 사항

.NET Framework 앱을 마이그레이션하려면 다음을 수행합니다.

1. 사전 조건 - .NET용 Porting Assistant를 사용하려면 애플리케이션 소스 코드를 분석하려는 시스템에 .NET 5+를 설치해야 합니다. 시스템의 리소스는 최소 1.8GHz 처리 속도, 4GB 메모리 및 5Gb 스토리지 공간을 가져야 합니다. 자세한 내용은 Porting Assistant for .NET 설명서의 사전 조건을 참조하세요.
2. 평가 - Porting Assistant for .NET을 실행 파일(다운로드)로 다운로드합니다. 시스템에 도구를 다운로드하여 설치하여 애플리케이션 평가를 시작할 수 있습니다. 평가 페이지에는 최신 .NET과 호환되지 않는 이식된 프로젝트, 패키지 및 APIs가 포함되어 있습니다. 따라서 평가 후 솔루션에서 빌드 오류가 발생합니다. 평가 결과를 보거나 CSV 파일로 다운로드할 수 있습니다. 자세한 내용은 Porting Assistant for .NET 설명서의 Porting a solution을 참조하세요.
3. 리팩터링 - 애플리케이션을 평가한 후 프로젝트를 대상 프레임워크 버전으로 포팅할 수 있습니다. 솔루션을 이식하면 프로젝트 파일과 일부 코드가 이식 도우미에 의해 수정됩니다. 로그를 확인하여 소스 코드의 변경 사항을 검토할 수 있습니다. 대부분의 경우 코드를 프로덕션에 준비하기 위해 마이그레이션 및 테스트를 완료하는 데 추가 노력이 필요합니다. 애플리케이션에 따라 일부 변경 사항에는 개체 프레임워크, 자격 증명 및 인증이 포함될 수 있습니다. 자세한 내용은 Porting Assistant for .NET 설명서의 Porting a solution을 참조하세요.

이는 애플리케이션을 컨테이너로 현대화하는 첫 번째 단계입니다. .NET Framework 앱을 Linux 컨테이너로 현대화하기 위한 여러 비즈니스 및 기술 동인이 있을 수 있습니다. 중요한 동인 중 하나는 Windows 운영 체제에서 Linux로 전환하여 총 소유 비용을 줄이는 것입니다. 이렇게 하면 애플리케이션을 .NET의 교차 플랫폼 버전과 컨테이너로 마이그레이션하여 리소스 사용률을 최적화할 때 라이선스 비용이 절감됩니다.

애플리케이션을 Linux로 이식한 후 AWS App2Container를 사용하여 애플리케이션을 컨테이너화할 수 있습니다. App2Container는 Amazon ECS 또는 Amazon EKS를 직접 배포할 수 있는 엔드포인트 서비

스로 사용합니다. App2Container는 애플리케이션을 반복적으로 컨테이너화하는 데 필요한 모든 코드형 인프라(IaC) 배포 아티팩트를 제공합니다.

추가 고려 사항 및 리소스

- https(2002년 레거시 프레임워크)를 기반으로 애플리케이션을 빌드하고 .NET 6으로 이식하려는 경우 AWS 블로그의 Microsoft 워크로드에서 Porting Assistant for VB.NET .NET을 사용하여 .NET 6.0으로 레거시 https 애플리케이션 이식 게시물을 참조하세요.
- Windows Communication Foundation(WCF)에 레거시 애플리케이션이 있고 최신 .NET에서 애플리케이션을 실행하려는 경우 CoreWCF를 채택할 수 있습니다. 자세한 내용은 AWS 블로그의 Microsoft 워크로드에서 Porting Assistant for .NET을 사용하여 레거시 WCF 애플리케이션을 CoreWCF로 현대화 게시물을 참조하세요.
- 이식 도우미를 Visual Studio IDE의 확장으로 추가할 수 있습니다. 이를 통해 IDE와 Porting Assistant for .NET 도구 간에 전환할 필요 없이 코드를 변환하는 데 필요한 모든 작업을 수행할 수 있습니다. 자세한 내용은 AWS 블로그의 Microsoft Workloads에서 Porting Assistant for .NET Visual Studio IDE 확장을 통한 .NET 애플리케이션 현대화 가속화 게시물을 참조하세요.
- AWS Porting Assistant for .NET은 이제 평가의 소스 코드 및 호환성 분석 구성 요소가 포함된 오픈 소스 도구입니다. 이렇게 하면 개발자가 .NET 이식 지식과 모범 사례를 사용하고 공유하도록 장려할 수 있습니다.
- AWS Toolkit for .NET Refactoring을 사용하여 Linux의 최신 .NET으로 .NET 프레임워크 애플리케이션을 이식할 수 있습니다. 자세한 내용은 블로그의 Microsoft 워크로드에서 AWS Toolkit for .NET 리팩터링을 사용한 .NET 현대화 가속화 게시물을 참조하세요 AWS .
- 를 AWS 사용하여 로 ASP.NET Core 애플리케이션의 컨테이너화 및 마이그레이션을 가속화할 AWS App2Container 수 있습니다.

.NET 앱 컨테이너화

개요

컨테이너는 일관되고 재현 가능한 방식으로 애플리케이션을 패키징하고 배포하는 가볍고 효율적인 방법입니다. 이 섹션에서는 서버리스 컨테이너 서비스 AWS Fargate인를 사용하여 확장 가능하고 안정적인 인프라를 제공하면서 .NET 애플리케이션의 비용을 절감하는 방법을 설명합니다.

비용 영향

비용 절감을 위해 컨테이너를 사용하는 효과에 영향을 미치는 몇 가지 요인으로는 애플리케이션의 크기와 복잡성, 배포해야 하는 애플리케이션 수, 애플리케이션의 트래픽 및 수요 수준이 있습니다. 소규모 또는 단순 애플리케이션의 경우 컨테이너 및 관련 서비스 관리로 인한 오버헤드로 인해 실제로 비용이 증가할 수 있으므로 컨테이너는 기존 인프라 접근 방식에 비해 상당한 비용 절감 효과를 제공하지 못할 수 있습니다. 그러나 규모가 크거나 복잡한 애플리케이션의 경우 컨테이너를 사용하면 리소스 사용률을 개선하고 필요한 인스턴스 수를 줄여 비용을 절감할 수 있습니다.

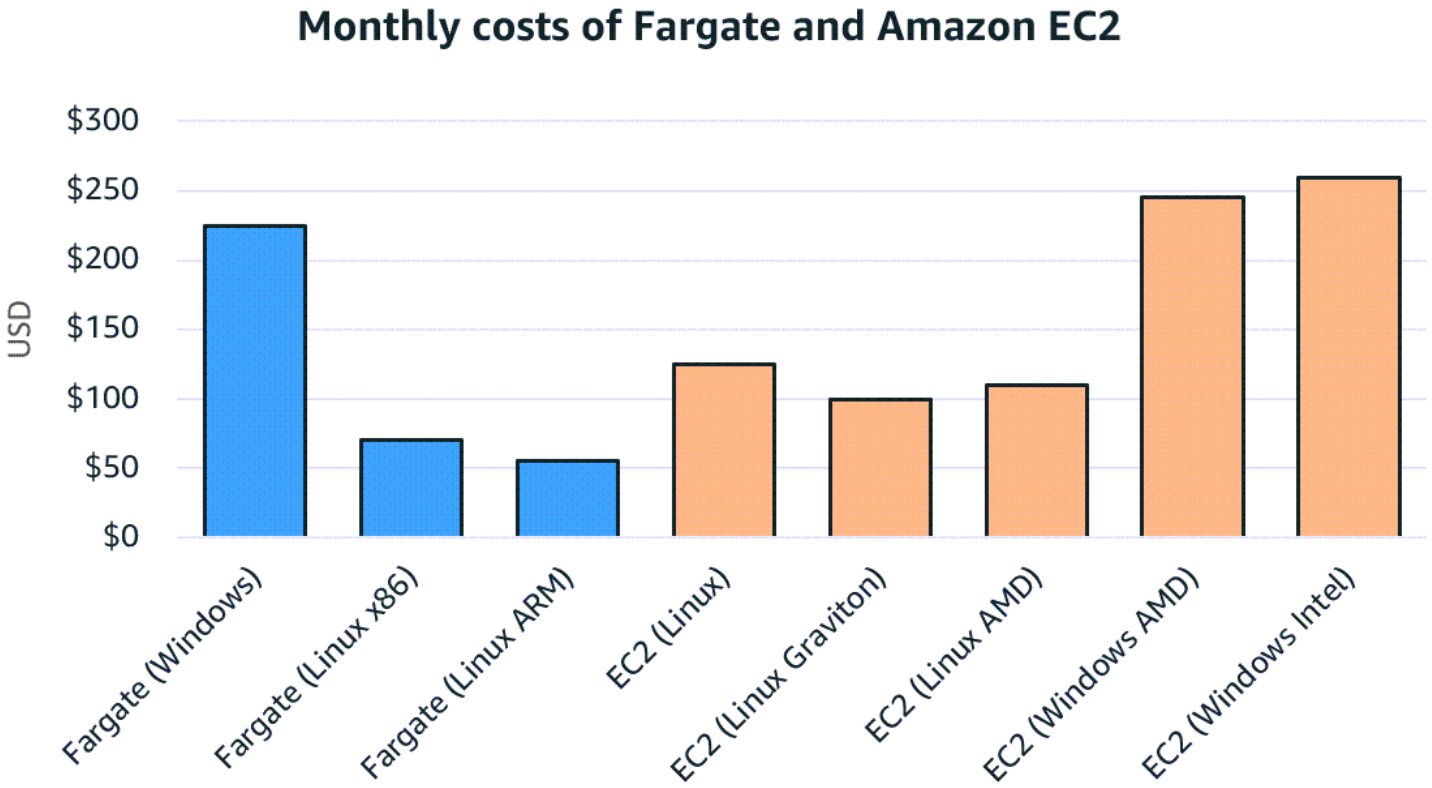
비용 절감을 위해 컨테이너를 사용할 때는 다음 사항에 유의하는 것이 좋습니다.

- 애플리케이션 크기 및 복잡성 - 더 크고 복잡한 애플리케이션은 더 많은 리소스가 필요한 경향이 있고 리소스 사용률 개선의 이점을 더 많이 누릴 수 있으므로 컨테이너화에 더 적합합니다.
- 애플리케이션 수 - 조직이 배포해야 하는 애플리케이션이 많을수록 컨테이너화를 통해 비용을 더 많이 절감할 수 있습니다.
- 트래픽 및 수요 - 트래픽과 수요가 높은 애플리케이션은 컨테이너가 제공하는 확장성과 탄력성의 이점을 누릴 수 있습니다. 이로 인해 비용이 절감될 수 있습니다.

아키텍처와 운영 체제가 다르면 컨테이너 비용에 영향을 미칩니다. Windows 컨테이너를 사용하는 경우 라이선스 고려 사항으로 인해 비용이 감소하지 않을 수 있습니다. Linux 컨테이너의 경우 라이선스 비용이 더 낮거나 없습니다. 다음 차트는 미국 동부(오하이오) 리전 AWS Fargate 의에서 기본 구성을 사용합니다. vCPUs 4개와 메모리 8GB가 할당된 상태에서 12시간 동안 각각 실행되는 매월 작업 30개.

EC2-based 컨테이너 호스트와 서버리스 또는 AWS의 두 가지 기본 컴퓨팅 플랫폼에서 컨테이너를 실행할 수 있습니다AWS Fargate. Fargate 대신 Amazon Elastic Container Service(Amazon ECS)를 사용하는 경우 필요한 경우 배치 엔진이 컨테이너를 인스턴스화할 수 있도록 실행 중인 컴퓨팅(인스턴스)을 유지해야 합니다. 대신 Fargate를 사용하는 경우 필요한 컴퓨팅 용량만 프로비저닝됩니다.

다음 차트는 Fargate를 사용하는 컨테이너와 Amazon EC2를 사용하는 컨테이너의 차이를 보여줍니다. Fargate의 유연성으로 인해 애플리케이션의 작업은 하루 12시간 실행될 수 있으며, 업무 외 시간에는 사용률이 전혀 없습니다. 그러나 Amazon ECS의 경우 EC2 인스턴스의 Auto Scaling 그룹을 사용하여 컴퓨팅 용량을 제어해야 합니다. 이로 인해 하루 24시간 용량이 실행되어 결국 비용이 증가할 수 있습니다.



비용 최적화 권장 사항

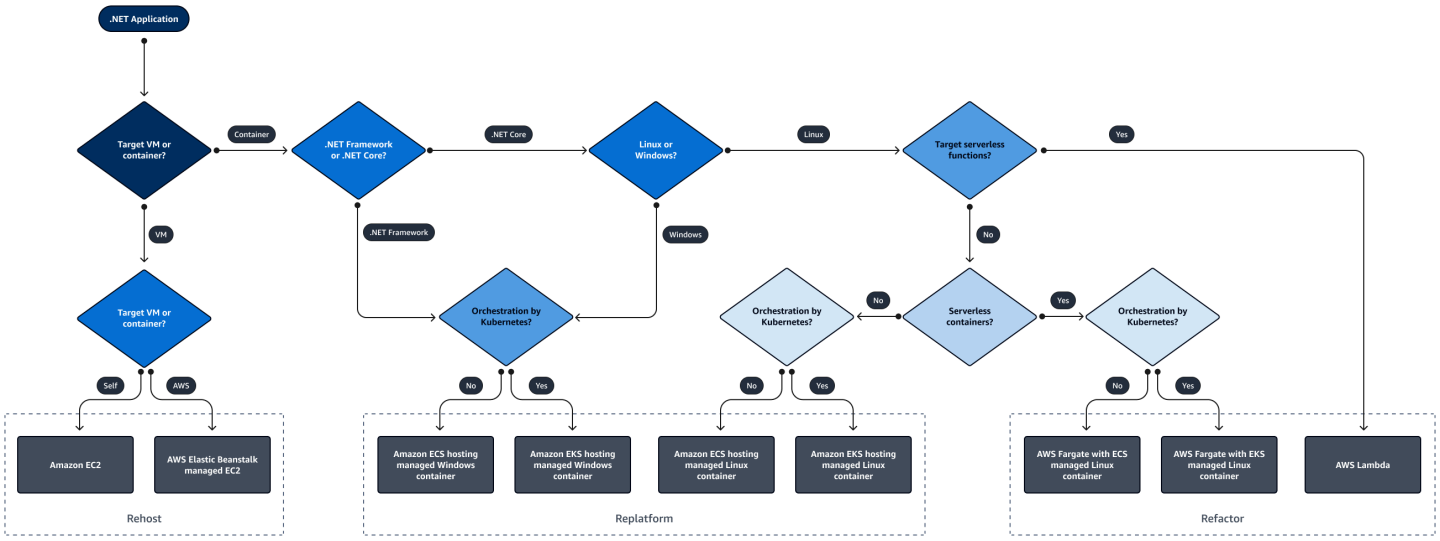
Windows 대신 Linux 컨테이너 사용

Windows 컨테이너 대신 Linux 컨테이너를 사용하면 비용을 크게 절감할 수 있습니다. 예를 들어 EC2 EC2 Windows에서 .NET Framework를 실행하는 대신 EC2 Linux에서 .NET Core를 실행하는 경우 컴퓨팅 비용을 약 45% 절감할 수 있습니다. x86 대신 ARM 아키텍처(AWS Graviton)를 사용하면 40% 추가 절감 효과를 얻을 수 있습니다.

기존 .NET Framework 애플리케이션에 대해 Linux 기반 컨테이너를 실행하려는 경우 Linux 컨테이너를 사용하려면 이러한 애플리케이션을 .NET의 최신 교차 플랫폼 버전(예: .NET 6.0)으로 이식해야 합니다. 주요 고려 사항은 리팩터링 비용을 Linux 컨테이너의 비용 절감을 통해 얻는 비용 절감액과 비교하는 것입니다. 애플리케이션을 최신 .NET으로 이식하는 방법에 대한 자세한 내용은 AWS 설명서의 Porting Assistant for .NET을 참조하세요.

최신 .NET(즉, .NET Framework에서 벗어남)으로 전환할 때 얻을 수 있는 또 다른 이점은 추가 현대화 기회를 사용할 수 있다는 것입니다. 예를 들어, 확장 가능하고 민첩하며 비용 효율적인 마이크로서비스 기반 아키텍처로 애플리케이션을 재설계하는 것을 고려할 수 있습니다.

다음 다이어그램은 현대화 기회를 탐색하기 위한 의사 결정 프로세스를 보여줍니다.



Savings Plans 활용

컨테이너를 사용하면 Compute Savings Plans을 활용하여 Fargate 비용을 절감할 수 있습니다. 유연한 할인 모델은 전환형 예약 인스턴스와 동일한 할인을 제공합니다. Fargate 요금은 컨테이너 이미지를 다운로드하기 시작한 시점부터 Amazon ECS 작업이 종료될 때까지(가장 가까운 초로 반올림) 사용된 vCPU 및 메모리 리소스를 기반으로 합니다. Fargate용 Savings Plans은 1년 또는 3년 기간 동안 특정 양의 컴퓨팅 사용량(시간당 달러로 측정)을 사용하겠다는 약정의 대가로 Fargate 사용량을 최대 50% 절감합니다. AWS Cost Explorer를 사용하여 Savings Plan.

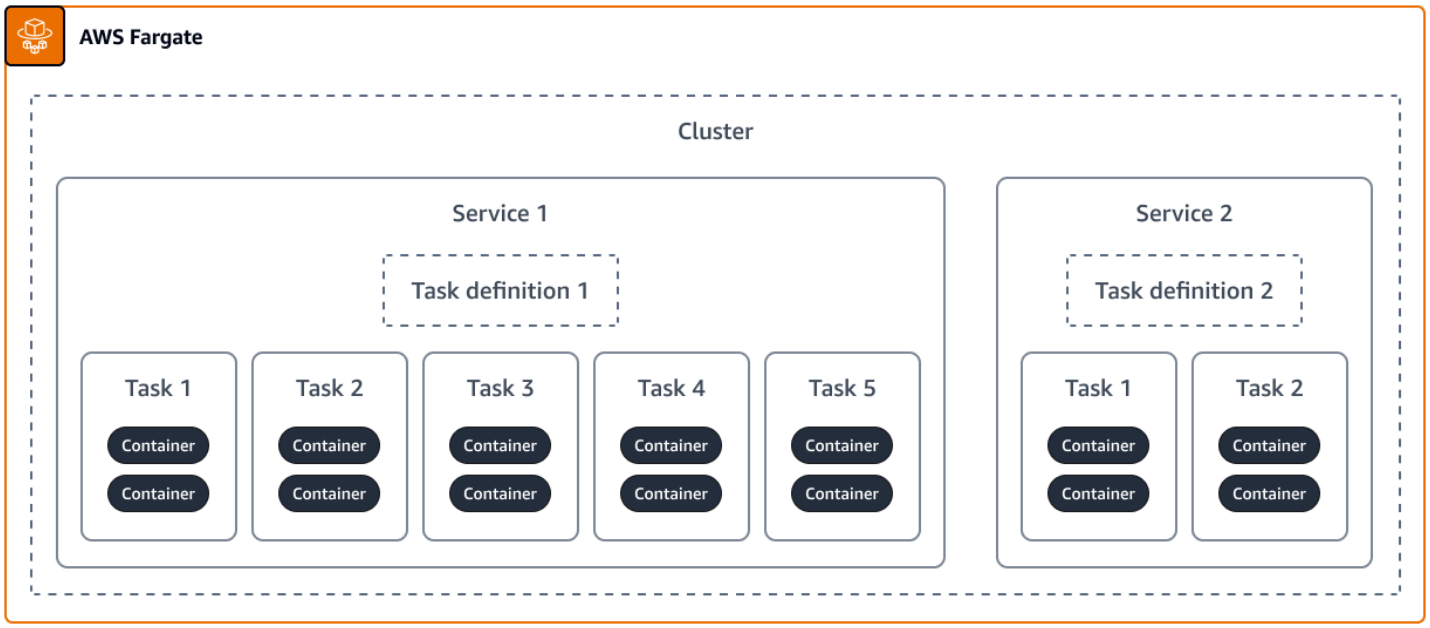
Compute Savings Plans은 가장 큰 절감을 가장 먼저 얻을 수 있는 사용량에 적용된다는 점을 이해하는 것이 중요합니다. 예를 들어에서 t3.medium Linux 인스턴스를 실행us-east-2하고 동일한 Windows t3.medium 인스턴스를 실행하는 경우 Linux 인스턴스는 먼저 Savings Plan 혜택을 받습니다. 이는 Linux 인스턴스의 절감 가능성이 50%인 반면 동일한 Windows 인스턴스의 절감 가능성이 35%이기 때문입니다. Amazon EC2 또는 Lambda AWS 계정와 같이에서 실행 중인 다른 Savings Plan 적격 리소스가 있는 경우 먼저 Savings Plan을 Fargate에 적용할 필요가 없습니다. 자세한 내용은 Savings Plans 설명서의 절감형 플랜이 AWS 사용량에 적용되는 방식 이해 및이 가이드의 Amazon EC2에서 Windows에 대한 지출 최적화 섹션을 참조하세요. Savings Plans

적절한 크기의 Fargate 작업

Fargate 태스크의 크기가 최대 비용 최적화를 달성할 수 있도록 올바르게 조정되었는지 확인하는 것이 중요합니다. 일반적으로 개발자는 애플리케이션에 사용되는 Fargate 작업에 대한 구성을 처음 결정할 때 필요한 사용 정보를 모두 가지고 있지는 않습니다. 이로 인해 작업이 과다 프로비저닝되고 불필요한 지출이 발생할 수 있습니다. 이를 방지하려면 Fargate에서 실행되는 테스트 애플리케이션을 로드하여 다양한 사용 시나리오에서 특정 작업 구성이 어떻게 작동하는지 이해하는 것이 좋습니다. 로드 테스트

결과, vCPU, 작업의 메모리 할당 및 Auto Scaling 정책을 사용하여 성능과 비용 간의 적절한 균형을 찾을 수 있습니다.

다음 다이어그램은 Compute Optimizer가 최적의 작업 및 컨테이너 크기에 대한 권장 사항을 생성하는 방법을 보여줍니다.



한 가지 접근 방식은의 분산 로드 테스트에 설명된 것과 같은 로드 테스트 AWS 도구를 사용하여 vCPU 및 메모리 사용률의 기준을 설정하는 것입니다. 로드 테스트를 실행하여 일반적인 애플리케이션 로드를 시뮬레이션한 후 기준 사용률이 달성될 때까지 작업에 대한 vCPU 및 메모리 구성을 미세 조정할 수 있습니다.

추가 리소스

- Amazon ECS 및 (Containers 블로그 게시물)에 대한 비용 최적화 체크리스트 AWS FargateAWS
- Amazon ECS 시작 유형별 이론적 비용 최적화: Fargate vs EC2(AWS 컨테이너 블로그 게시물)
- Porting Assistant for .NET(AWS 문서화)
- 의 분산 로드 테스트 AWS(AWS 솔루션 라이브러리)
- AWS Compute Optimizer ,에서 Amazon ECS 서비스에 대한 지원 시작 AWS Fargate(AWS 클라우드 재무 관리 블로그 게시물)

Graviton 인스턴스 및 컨테이너 사용

개요

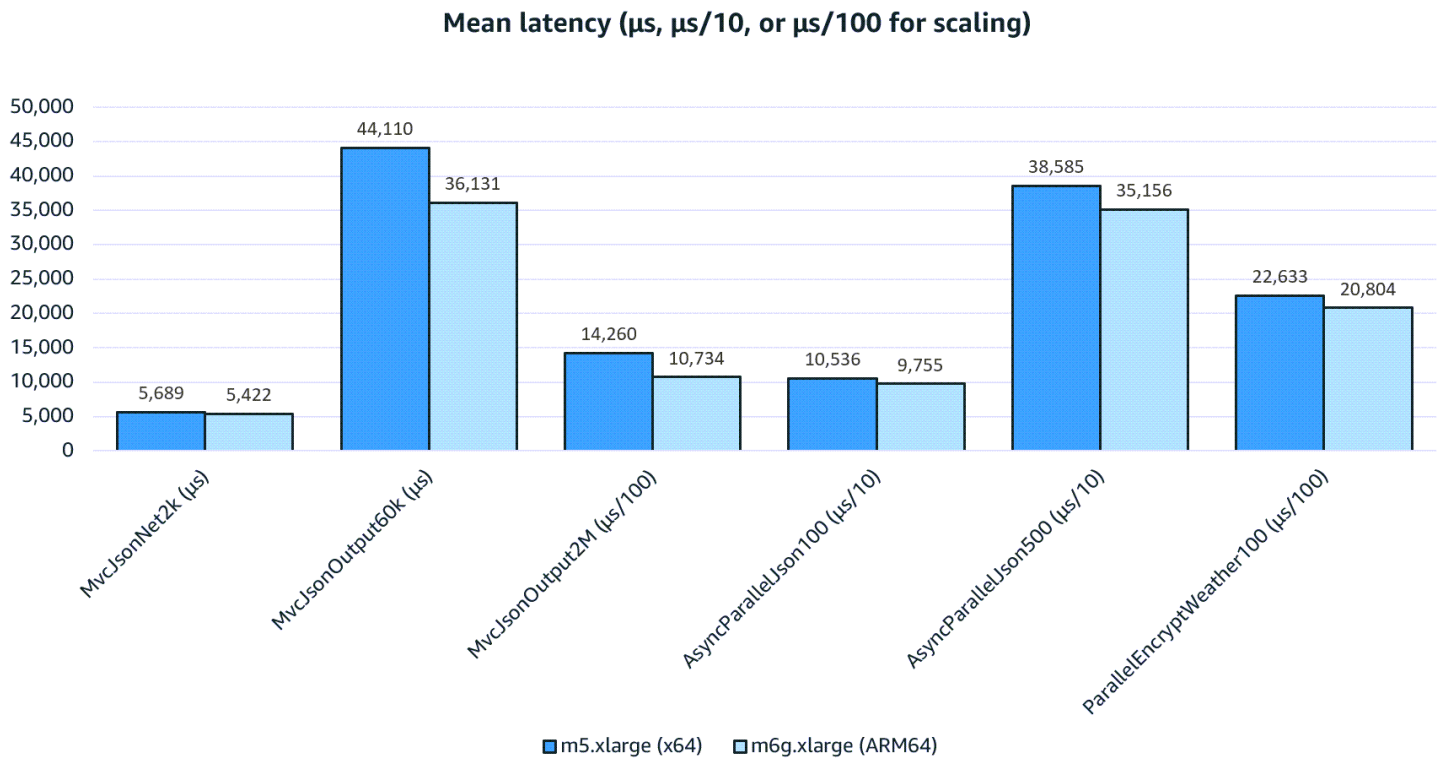
AWS Graviton 인스턴스는에서 실행되는 컨테이너 AWS 를 포함하여 Amazon Elastic Compute Cloud(Amazon EC2)에서 실행되는 클라우드 워크로드에 최상의 가격 대비 성능을 제공하도록에서 설계된 ARM 프로세서로 구동됩니다 AWS. 현재 Amazon EC2에서 사용할 수 있는 Graviton은 3세대입니다. 이 가이드는 최신 버전의 Graviton을 사용할 때 상당한 비용 절감 효과가 있기 때문에 .NET 애플리케이션에서 Graviton 2 및 3을 사용하는 데 중점을 둡니다. Graviton 인스턴스는 Linux 운영 체제만 실행한다는 점에 유의하세요. 따라서 Graviton 인스턴스는 Linux에서 실행되는 .NET용 강력한 제품이지만 Windows 운영 체제 또는 레거시 .NET Framework 애플리케이션에는 옵션이 아닙니다.

Graviton 3은 유사한 EC2 인스턴스보다 60% 더 효율적이며 최대 40% 더 나은 성능을 제공합니다. 이 가이드는 Graviton 사용의 비용 이점에 중점을 두지만 Graviton은 성능 개선 및 환경 지속 가능성 개선의 추가 이점을 제공한다는 점에 유의해야 합니다.

비용 영향

Graviton으로 전환하면 최대 45%까지 절감할 수 있습니다. 레거시 .NET Framework 애플리케이션을 최신 .NET 버전으로 리팩터링한 후에는 Graviton 인스턴스 사용 기능을 잠금 해제합니다. Graviton으로 전환하는 것은 .NET 개발자에게 효과적인 비용 최적화 기법입니다.

다음 표의 예제는 Graviton 인스턴스로 마이그레이션하여 달성할 수 있는 성능 개선 가능성을 보여줍니다.



이전 다이어그램에서 결과를 생성하는 데 사용되는 벤치마킹 접근 방식에 대한 전체 분석 및 설명은 AWS 컴퓨팅 블로그의 Powering .NET 5 with AWS Graviton2: Benchmarks를 참조하세요.

효율성이 향상되는 이유 중 하나는 x86과 Graviton 간의 vCPU 의미 차이입니다. x86 아키텍처에서 vCPU는 하이퍼스레딩으로 달성되는 논리적 코어입니다. Graviton에서 vCPU는 vCPU가 워크로드에 완전히 커밋되도록 하는 물리적 코어와 같습니다.

Graviton2를 사용하면 유사한 x86/x64 인스턴스에 비해 가격 대비 40% 더 나은 성능을 얻을 수 있습니다. Graviton3은 Graviton2를 통해 다음을 제공합니다.

- 최대 25% 향상된 성능으로 성능 프로파일 향상
- 최대 2배 더 높은 부동 소수점 성능
- 최대 두 배 더 빠른 암호화 워크로드 성능
- 최대 3배 향상된 기계 학습 성능

또한 Graviton3는 클라우드에서 최초로 DDR5 메모리를 탑재한 인스턴스입니다.

다음 표는 Graviton 기반 인스턴스와 이에 상응하는 x86 기반 인스턴스 간의 비용 절감 차이를 보여줍니다.

이 표에는 19.20%의 Graviton 절감액이 나와 있습니다.

인스턴스 유형	아키텍처	vCPU	메모리(GB)	시간당 비용(온디맨드)
t4g.xlarge	ARM	4	16	$0.1344
t3.xlarge	x86	4	16	$0.1664

이 표에는 14.99%의 Graviton 절감액이 나와 있습니다.

인스턴스 유형	아키텍처	vCPU	메모리(GB)	시간당 비용(온디맨드)
c7g.4xlarge	ARM	16	32	$0.5781
c6i.4xlarge	x86	16	32	0.6800 USD

Graviton을 고려할 때는 애플리케이션의 성능 프로파일을 테스트하는 것이 중요합니다. Graviton은 견고한 소프트웨어 개발 사례를 대체하지 않습니다. 테스트를 사용하여 기본 컴퓨팅 리소스를 최대한 활용하고 있는지 확인할 수 있습니다.

비용 최적화 권장 사항

Graviton 프로세서/인스턴스를 활용하는 방법에는 여러 가지가 있습니다. 이 섹션에서는 x86 아키텍처 머신 사용에서 Graviton(ARM) 인스턴스로 이동하는 데 필요한 변경 사항을 안내합니다.

Lambda에서 런타임 설정 변경

런타임 설정을 전환하는 것이 좋습니다 AWS Lambda. 자세한 내용은 Lambda 설명서의 런타임 환경 수정을 참조하세요. .NET은 컴파일된 언어이므로 빌드 프로세스에 따라이 작업을 수행해야 합니다. 이를 수행하는 방법의 예는 GitHub의 Graviton에서 .NET을 참조하세요.

컨테이너

컨테이너화된 워크로드의 경우 다중 아키텍처 컨테이너 이미지를 생성합니다. Docker 빌드 명령에서 여러 아키텍처를 지정하여이 작업을 수행할 수 있습니다. 예시:

```
docker buildx build -t "myImageName:latest" --platform linux/amd64,linux/arm64 --push .
```

와 같은 도구를 사용하여 빌드를 오케스트레이션 AWS 클라우드 개발 키트 (AWS CDK) 할 수도 있습니다. https://aws.amazon.com/blogs/devops/build-and-deploy-net-web-applications-to-arm-powered-aws-graviton-2-amazon-ecs-clusters-using-aws-cdk/ Docker의 예제는 Docker 설명서의 Docker Desktops를 사용하여 Arm 및 x86용 다중 아치 이미지 빌드를 참조하세요.

Amazon EC2

x86/x64에서 ARM으로 마이그레이션하려면 컴파일 단계에서 ARM 아키텍처를 대상으로 지정합니다. 비주얼 스튜디오에서 ARM64 CPU를 생성할 수 있습니다. 지침은 Microsoft 설명서의 Arm64 및 기타 플랫폼을 대상으로 프로젝트를 구성하려면을 참조하세요.

.NET CLI를 사용하는 경우 ARM 시스템에서 빌드를 실행하면 Graviton 호환 빌드가 생성됩니다. 데모를 보려면 YouTube에서 Arm64 onGraviton2로 .NET 6 성능 가속화 AWS Graviton2를 시청하세요. 종속성 문제로 인해 컴파일 시간 오류가 발생하여 개별적으로 해결할 수 있습니다. 종속성에 대한 ARM 라이브러리가 있는 한 전환은 비교적 간단해야 합니다.

추가 리소스

- ARM용 컨테이너를 빌드하고 Amazon ECS에서 Graviton 및 스팟 인스턴스를 사용하여 저장하는 방법(AWS 블로그)
- AWS LambdaAWS Graviton2 프로세서로 구동되는 함수 - Arm에서 함수를 실행하고 최대 34% 더 나은 가격 대비 성능을 얻을 수 있습니다(AWS 블로그).
- Arm-based AWS Graviton2 프로세서로 AWS Lambda 함수 마이그레이션(AWS 블로그)
- 를 사용하여 ARM 기반 AWS Graviton 2 Amazon ECS 클러스터에 .NET 웹 애플리케이션 구축 및 배포 AWS CDK(AWS 블로그)
- Graviton Fast Start – 워크로드를 AWS Graviton으로 이전하는 데 도움이 되는 새로운 프로그램(AWS 블로그)
- AWS Graviton2를 사용하여 .NET 5에 전원 공급: 벤치마크(AWS 블로그)

정적 .NET Framework 앱에 대한 동적 조정 지원

개요

애플리케이션에 클라우드를 사용할 때 얻을 수 있는 주요 이점 중 하나는 탄력성 또는 수요에 따라 컴퓨팅을 스케일 인 또는 스케일 아웃하는 기능입니다. 이를 통해 최대 사용량에 맞게 프로비저닝하는 대신 필요한 컴퓨팅 용량에 대해서만 비용을 지불할 수 있습니다. 온라인 소매업체가 평소보다 많은 트래픽을 빠르게 가져올 수 있는 Cyber Monday(예: 몇 분 내에 수천 퍼센트)는 탄력성의 좋은 예입니다.

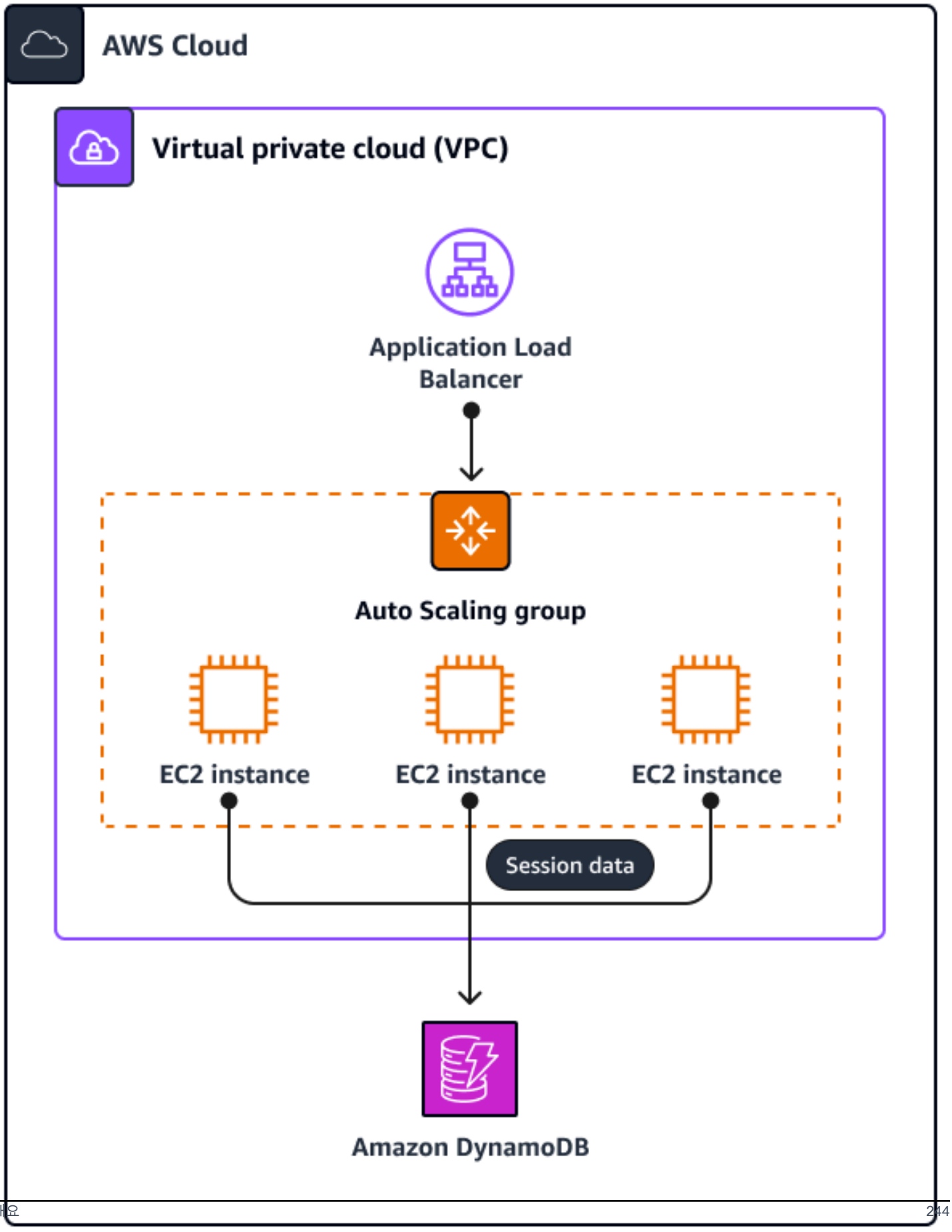
레거시 .NET 웹 애플리케이션을 클라우드로 가져오는 경우(예: IIS에서 실행되는 ASP.NET Framework 애플리케이션) 애플리케이션의 상태 저장 특성으로 인해 로드 밸런싱된 서버 팜을 빠르게 확장하는 기능이 어렵거나 불가능할 수 있습니다. 사용자 세션 데이터는 일반적으로 지속되어야 하는 교차 요청 데이터를 포함하는 ASP.NET 세션 상태 또는 정적 변수와 함께 애플리케이션의 메모리 내에 저장됩니다. 사용자 세션 선호도는 일반적으로 로드 밸런서 고정 세션을 통해 유지됩니다.

이는 운영상 어려운 것으로 증명됩니다. 용량을 늘려야 하는 경우 서버를 의도적으로 프로비저닝하고 추가해야 합니다. 이는 느린 프로세스일 수 있습니다. 패치가 적용되거나 예기치 않은 장애가 발생할 경우 노드를 서비스에서 제외하는 것은 최종 사용자 경험에 문제가 될 수 있으며 영향을 받는 노드와 연결된 모든 사용자의 상태가 손실될 수 있습니다. 이 경우 사용자가 다시 로그인해야 합니다.

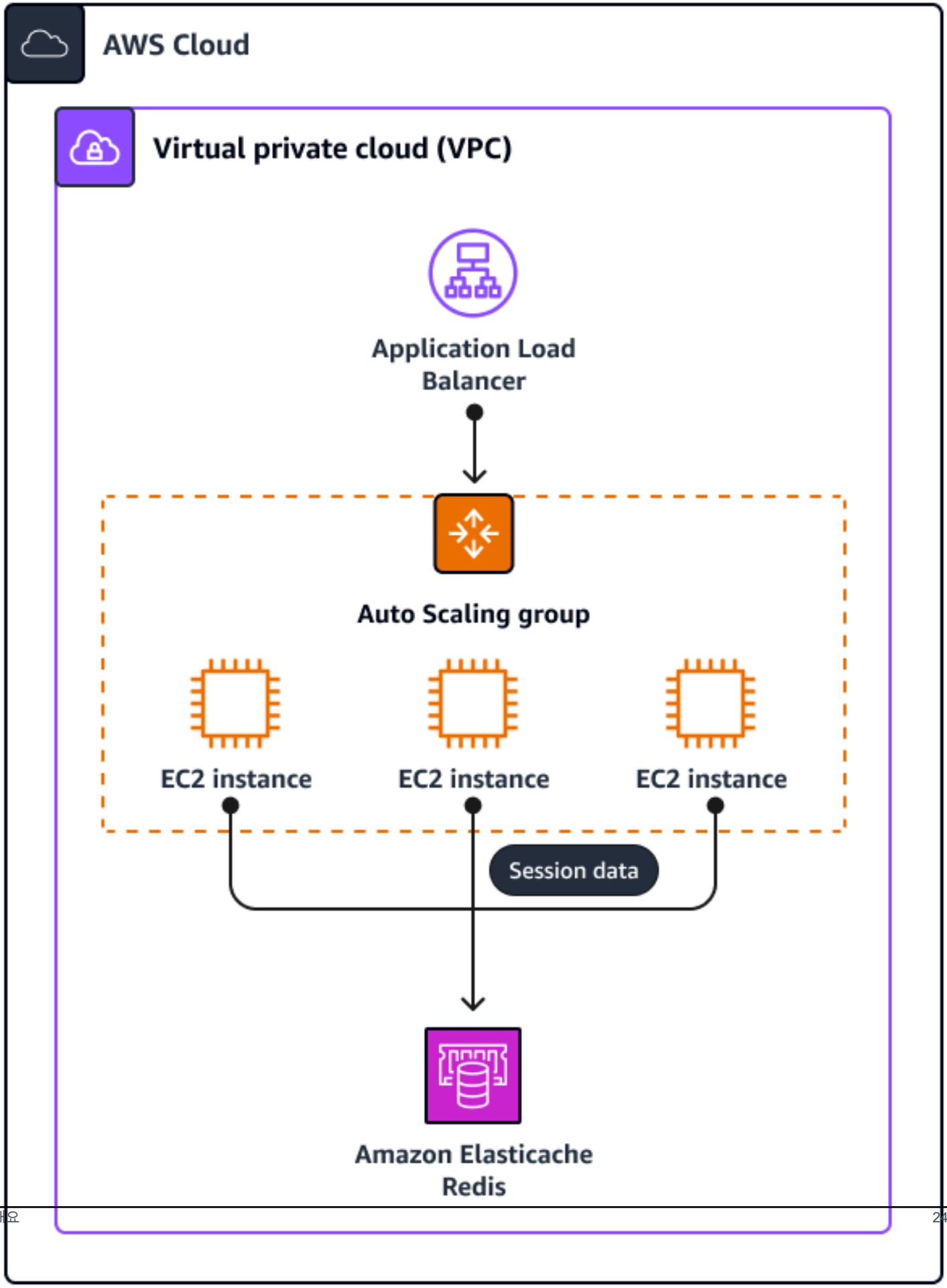
ASP.NET 애플리케이션의 세션 상태를 중앙 집중화하고 레거시 ASP.NET 애플리케이션에 Auto Scaling 규칙을 적용하면 클라우드의 탄력성을 활용하고 애플리케이션을 실행할 때 비용 절감을 활용할 수 있습니다. 예를 들어 컴퓨팅 확장성을 통해 비용을 절감할 수 있지만 예약 인스턴스 사용량 감소 및 Amazon 스팟 인스턴스 요금 사용과 같은 다양한 요금 모델 중에서 선택할 수도 있습니다.

두 가지 일반적인 기법으로는 Amazon DynamoDB를 세션 상태 공급자로 사용하고 Amazon ElastiCache(Redis OSS)를 ASP.NET 세션 스토어로 사용하는 것이 있습니다.

다음 다이어그램은 DynamoDB를 세션 상태 공급자로 사용하는 아키텍처를 보여줍니다.


AWS Cloud
Virtual private cloud (VPC)
Application Load Balancer
Auto Scaling group
EC2 instance
EC2 instance
EC2 instance
Session data
Amazon DynamoDB

다음 다이어그램은 ElastiCache(Redis OSS)를 세션 상태 공급자로 사용하는 아키텍처를 보여줍니다.

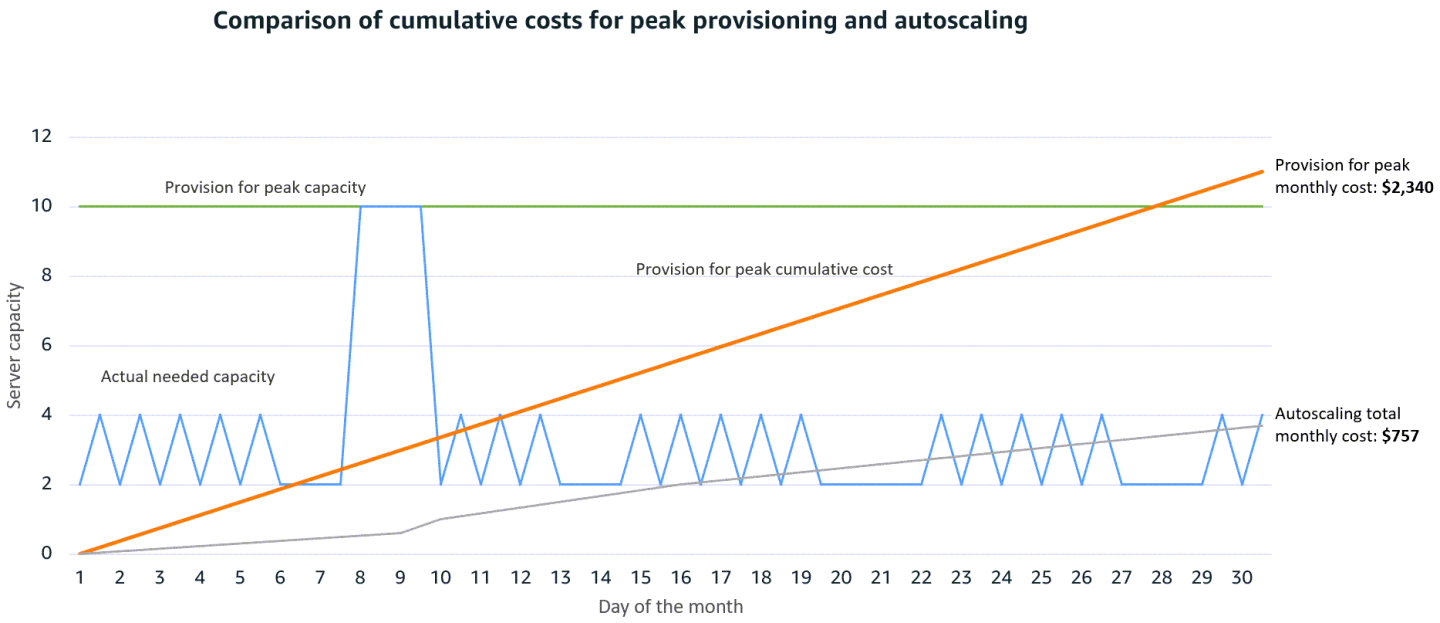

AWS Cloud
Virtual private cloud (VPC)
Application Load Balancer
Auto Scaling group
EC2 instance
EC2 instance
EC2 instance
Session data
Amazon Elasticache Redis

비용 영향

프로덕션 애플리케이션의 규모 조정 이점을 확인하려면 실제 수요를 모델링하는 것이 좋습니다. 이 섹션에서는 샘플 애플리케이션을 모델링하기 위해 다음과 같이 가정합니다.

- 교체에서 추가 및 제거되는 인스턴스는 동일하며 인스턴스 크기 변형이 도입되지 않습니다.
- 애플리케이션의 고가용성을 유지하기 위해 서버 사용률이 두 개의 활성 서버 아래로 절대 떨어지지 않습니다.
- 서버의 양은 트래픽에 따라 선형적으로 확장됩니다(즉, 트래픽의 두 배에 필요한 컴퓨팅의 두 배).
- 트래픽은 한 달 동안 6시간 단위로 모델링되며, 10배 트래픽의 1일 동안 일중 변동과 비정상적인 트래픽 피크(예: 프로모션 판매) 1개가 발생합니다. 주말 트래픽은 기본 사용률로 모델링됩니다.
- 야간 트래픽은 기본 사용률로 모델링되는 반면, 평일 트래픽은 4배 사용률로 모델링됩니다.
- 예약 인스턴스 요금은 1년 선결제 없는 요금을 사용합니다. 일반 주간 요금은 온디맨드 요금을 사용하는 반면 버스트 수요는 스팟 인스턴스 요금을 사용합니다.

다음 다이어그램은이 모델이 최대 사용량을 프로비저닝하는 대신 .NET 애플리케이션에서 탄력성을 활용하는 방법을 보여줍니다. 이로 인해 약 68%가 절감됩니다.



DynamoDB를 세션 상태 스토리지 메커니즘으로 사용하는 경우 다음 파라미터를 사용합니다.

```
Storage: 20GB
Session Reads: 40 million
Session Writes: 20 million
```

```
Pricing Model: On demand
```

이 서비스의 예상 월별 비용은 매월 약 35.00 USD입니다.

ElastiCache(Redis OSS)를 세션 상태 스토리지 메커니즘으로 사용하는 경우 다음 파라미터를 사용합니다.

```
Number of Nodes: 3
Node size: cache.t4g.medium
Pricing Model: 1y reserved
```

이 서비스의 예상 월별 비용은 매월 약 91.00 USD입니다.

비용 최적화 권장 사항

첫 번째 단계는 레거시 .NET 애플리케이션에서 세션 상태를 구현하는 것입니다. ElastiCache를 상태 스토리지 메커니즘으로 사용하는 경우 AWS 개발자 도구 블로그의 ElastiCache를 ASP.NET세션 스토어'로 사용하는 지침을 따르세요. DynamoDB를 사용하는 경우 SDK for .NET 설명서의 What is the AWS SDK for .NET 지침을 따르세요.

애플리케이션에서 InProc 세션을 사용하여 로 시작하는 경우 세션에 저장하려는 모든 객체를 직렬화할 수 있는지 확인합니다. 이렇게 하려면 `SerializableAttribute` 속성을 사용하여 인스턴스가 세션에 저장될 클래스를 장식합니다. 예시:

```
[Serializable()]
public class TestSimpleObject {
public string SessionProperty {get;set;}
}
```

또한 .NET은 사용 중인 모든 서버 간에 동일해야 `MachineKey` 합니다. 이는 일반적으로 공통 Amazon Machine Image(AMI)에서 인스턴스가 생성되는 경우입니다. 예시:

```
<machineKey
validationKey="some long hashed value"
decryptionKey="another long hashed value"
validation="SHA1"/>
```

그러나 기본 이미지가 변경되면 동일한 .NET 머신 이미지로 구성해야 합니다(IIS 또는 서버 수준에서 구성 가능). 자세한 내용은 Microsoft 설명서의 SystemWebSectionGroup.MachineKey 속성을 참조하세요.

마지막으로 조정 이벤트에 대한 응답으로 Auto Scaling 그룹에 서버를 추가하는 메커니즘을 결정해야 합니다. 이를 수행하는 방법에는 여러 가지가 있습니다. Auto Scaling 그룹의 EC2 인스턴스에 .NET Framework 애플리케이션을 원활하게 배포하려면 다음 방법을 사용하는 것이 좋습니다.

- EC2 Image Builder를 사용하여 완전히 구성된 서버 및 애플리케이션이 포함된 AMI를 구성합니다. 그런 다음이 AMI를 사용하여 Auto Scaling 그룹의 시작 템플릿을 구성할 수 있습니다.
- AWS CodeDeploy을 사용하여 귀하의 애플리케이션을 배포합니다. CodeDeploy를 사용하면 Amazon EC2 Auto Scaling과 직접 통합할 수 있습니다. 이는 애플리케이션의 각 릴리스에 대해 새 AMI를 생성하는 대안을 제공합니다.

추가 리소스

- EC2 Image Builder를 사용하여 이미지 생성(EC2 Image Builder 설명서)
- Visual Studio Team Services와 AWS CodeDeploy 함께를 사용하여 .NET 웹 애플리케이션 배포(AWS 개발자 도구 블로그)

캐싱을 사용하여 데이터베이스 수요 감소

개요

캐싱을 효과적인 전략으로 사용하여 .NET 애플리케이션의 비용을 절감할 수 있습니다. 많은 애플리케이션은 애플리케이션에 데이터에 대한 빈번한 액세스가 필요한 경우 SQL Server와 같은 백엔드 데이터베이스를 사용합니다. 이러한 백엔드 서비스를 유지하여 수요에 대처하는 데 드는 비용은 높을 수 있지만 효과적인 캐싱 전략을 사용하여 크기 조정 및 크기 조정 요구 사항을 줄여 백엔드 데이터베이스의 부하를 줄일 수 있습니다. 이를 통해 비용을 절감하고 애플리케이션의 성능을 개선할 수 있습니다.

캐싱은 SQL Server와 같이 비용이 많이 드는 리소스를 사용하는 과중한 워크로드 읽기와 관련된 비용을 절감하는 데 유용한 기술입니다. 워크로드에 적합한 기술을 사용하는 것이 중요합니다. 예를 들어 로컬 캐싱은 확장할 수 없으며 애플리케이션의 각 인스턴스에 대해 로컬 캐시를 유지해야 합니다. 기본 데이터 소스의 비용이 낮을수록 캐싱 메커니즘과 관련된 추가 비용이 상쇄되도록 성능 영향을 잠재적 비용과 비교해야 합니다.

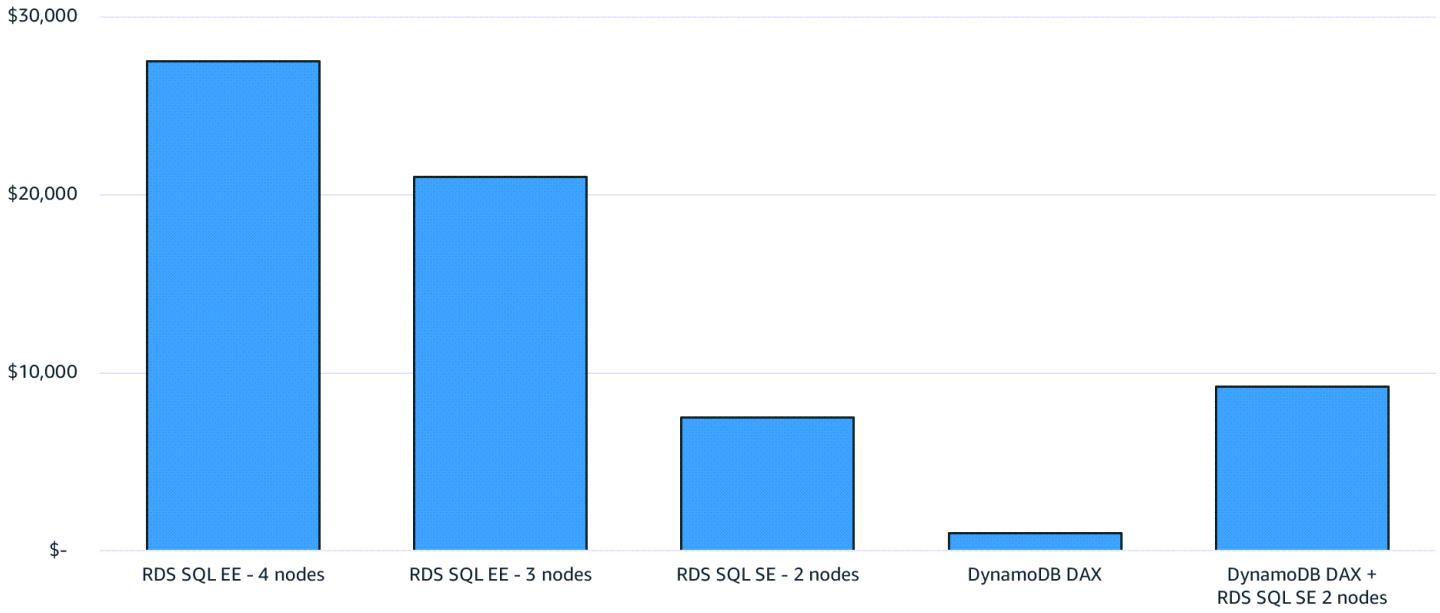
비용 영향

SQL Server를 사용하려면 데이터베이스 크기를 조정할 때 읽기 요청을 고려해야 합니다. 이는 로드에 맞게 읽기 전용 복제본을 도입해야 할 수 있으므로 비용에 영향을 미칠 수 있습니다. 읽기 전용 복제본

을 사용하는 경우 SQL Server Enterprise 에디션에서만 사용할 수 있다는 점을 이해하는 것이 중요합니다. 이 에디션에는 SQL Server Standard 에디션보다 더 비싼 라이선스가 필요합니다.

다음 다이어그램은 캐싱 효과를 이해하는 데 도움이 되도록 설계되었습니다. SQL Server Enterprise 에디션을 실행하는 db.m4.2xlarge 노드 4개가 있는 Amazon RDS for SQL Server를 보여줍니다. 하나의 읽기 전용 복제본을 사용하여 다중 AZ 구성에 배포됩니다. 독점 읽기 트래픽(예: SELECT 쿼리)은 읽기 전용 복제본으로 전달됩니다. 이에 비해 Amazon DynamoDB는 r4.2xlarge 2노드 DynamoDB Accelerator(DAX) 클러스터를 사용합니다.

다음 차트는 높은 읽기 트래픽을 처리하는 전용 읽기 전용 복제본이 필요하지 않은 결과를 보여줍니다.



읽기 전용 복제본이 없는 로컬 캐싱을 사용하거나 Amazon RDS의 SQL Server를 캐싱 계층으로 사용하여 DAX를 나란히 도입하여 비용을 크게 절감할 수 있습니다. 이 계층은 SQL Server에서 오프로드하고 데이터베이스를 실행하는 데 필요한 SQL Server의 크기를 줄입니다.

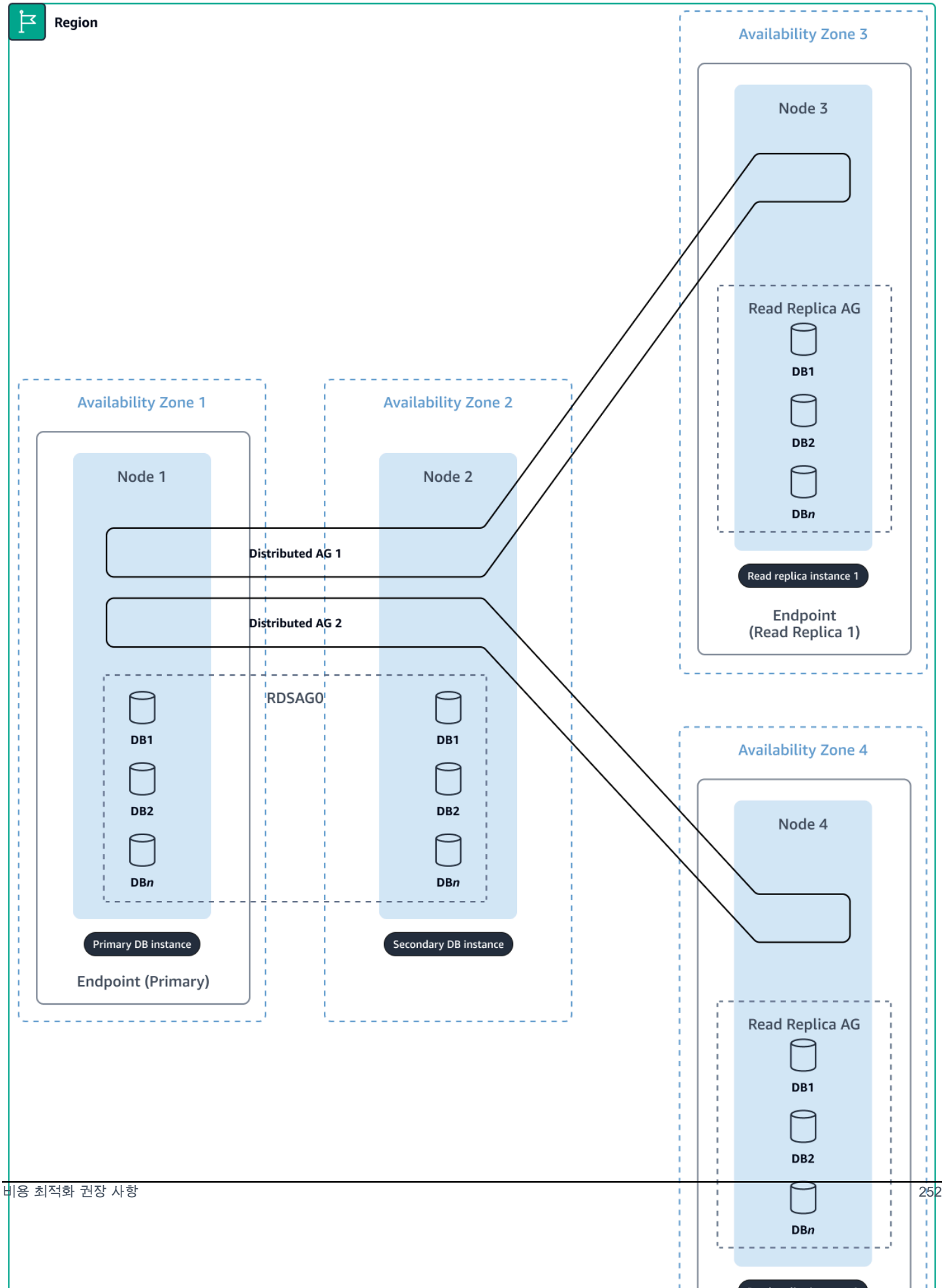
비용 최적화 권장 사항

로컬 캐싱

로컬 캐싱은 온프레미스 환경 또는 클라우드에서 호스팅되는 애플리케이션의 콘텐츠를 캐싱하는 가장 일반적으로 사용되는 방법 중 하나입니다. 이는 구현이 비교적 쉽고 직관적이기 때문입니다. 로컬 캐싱에는 데이터베이스 또는 기타 소스에서 콘텐츠를 가져와 더 빠른 액세스를 위해 메모리 또는 디스크에서 로컬로 캐싱하는 작업이 포함됩니다. 이 접근 방식은 구현하기 쉽지만 일부 사용 사례에는 적합하지 않습니다. 예를 들어, 여기에는 애플리케이션 상태 또는 사용자 상태 보존과 같이 시간이 지남에 따라

캐싱 콘텐츠를 유지해야 하는 사용 사례가 포함됩니다. 또 다른 사용 사례는 캐시된 콘텐츠에 다른 애플리케이션 인스턴스에서 액세스해야 하는 경우입니다.

아래 다이어그램은 노드 4개와 읽기 전용 복제본 2개가 있는 고가용성 SQL Server 클러스터를 보여줍니다.


Region
Availability Zone 3
Node 3
Read Replica AG
DB1
DB2
DBn
Read replica instance 1
Endpoint
(Read Replica 1)
Availability Zone 1
Availability Zone 2
Node 1
Node 2
Distributed AG 1
Distributed AG 2
RDSAG0
DB1
DB2
DBn
DB1
DB2
DBn
Primary DB instance
Secondary DB instance
Endpoint (Primary)
Availability Zone 4
Node 4
Read Replica AG
DB1
DB2
DBn

로컬 캐싱을 사용하면 여러 EC2 인스턴스에서 트래픽을 로드 밸런싱해야 할 수 있습니다. 각 인스턴스는 자체 로컬 캐시를 유지해야 합니다. 캐시에 상태 저장 정보가 저장되어 있는 경우 데이터베이스에 대한 정기적인 커밋이 있어야 하며, 사용자는 각 후속 요청(스티키 세션)에 대해 동일한 인스턴스로 전달되어야 할 수 있습니다. 이로 인해 일부 인스턴스는 과도하게 사용될 수 있지만 일부 인스턴스는 트래픽이 고르지 않게 분산되어 활용도가 낮기 때문에 애플리케이션을 확장하려고 할 때 문제가 발생합니다.

.NET 애플리케이션에 대해 인 메모리 또는 로컬 스토리지를 사용하는 로컬 캐싱을 사용할 수 있습니다. 이렇게 하려면 디스크에 객체를 저장하고 필요할 때 검색하는 기능을 추가하거나 데이터베이스에서 데이터를 쿼리하여 메모리에 유지할 수 있습니다. 예를 들어 C#의 SQL Server에서 로컬 캐싱 인 메모리 및 로컬 데이터 스토리지를 수행하려면 MemoryCache 및 LiteDB 라이브러리의 조합을 사용할 수 있습니다.는 인 메모리 캐싱을 MemoryCache 제공하는 반면 LiteDB는 빠르고 가벼운 임베디드 NoSQL 디스크 기반 데이터베이스입니다.

인 메모리 캐싱을 수행하려면 .NET 라이브러리를 사용합니다System.Runtime.MemoryCache. 다음 코드 예제에서는 System.Runtime.Caching.MemoryCache 클래스를 사용하여 인 메모리 데이터를 캐싱하는 방법을 보여줍니다. 이 클래스는 애플리케이션의 메모리에 데이터를 일시적으로 저장하는 방법을 제공합니다. 이렇게 하면 데이터베이스 또는 API와 같이 더 비싼 리소스에서 데이터를 가져올 필요성을 줄여 애플리케이션의 성능을 개선하는 데 도움이 될 수 있습니다.

코드는 다음과 같이 작동합니다.

1. MemoryCache 라는 프라이빗 정적 인스턴스_memoryCache가 생성됩니다. 캐시에는 식별을 위한 이름(dataCache)이 부여됩니다. 그런 다음 캐시는 데이터를 저장하고 검색합니다.
2. GetData 메서드는 string 키와 라는 Func<T> 위임이라는 두 가지 인수를 사용하는 일반 메서드입니다getData. 키는 캐시된 데이터를 식별하는 데 사용되는 반면, getData 대리인은 데이터가 캐시에 없을 때 실행되는 데이터 검색 로직을 나타냅니다.
3. 메서드는 먼저 _memoryCache.Contains(key) 메서드를 사용하여 캐시에 데이터가 있는지 확인합니다. 데이터가 캐시에 있는 경우 메서드는를 사용하여 데이터를 검색_memoryCache.Get(key)하고 예상 유형 T로 캐스팅합니다.
4. 데이터가 캐시에 없는 경우 메서드는 getData 대리인을 호출하여 데이터를 가져옵니다. 그런 다음를 사용하여 캐시에 데이터를 추가합니다_memoryCache.Add(key, data, DateTimeOffset.Now.AddMinutes(10)). 이 호출은 캐시 항목이 10분 후에 만료되도록 지정하며,이 시점에서 데이터가 캐시에서 자동으로 제거됩니다.
5. ClearCache 메서드는 string 키를 인수로 사용하고를 사용하여 해당 키와 연결된 데이터를 캐시에서 제거합니다_memoryCache.Remove(key).

```
using System;
using System.Runtime.Caching;

public class InMemoryCache
{
    private static MemoryCache _memoryCache = new MemoryCache("dataCache");

    public static T GetData<T>(string key, Func<T> getData)
    {
        if (_memoryCache.Contains(key))
        {
            return (T)_memoryCache.Get(key);
        }

        T data = getData();
        _memoryCache.Add(key, data, DateTimeOffset.Now.AddMinutes(10));

        return data;
    }

    public static void ClearCache(string key)
    {
        _memoryCache.Remove(key);
    }
}
```

다음 코드를 사용할 수 있습니다.

```
public class Program
{
    public static void Main()
    {
        string cacheKey = "sample_data";

        Func<string> getSampleData = () =>
        {
            // Replace this with your data retrieval logic
            return "Sample data";
        };

        string data = InMemoryCache.GetData(cacheKey, getSampleData);
        Console.WriteLine("Data: " + data);
    }
```

```
}
```

다음 예제에서는 LiteDB를 사용하여 로컬 스토리지에 데이터를 캐싱하는 방법을 보여줍니다. LiteDB를 인 메모리 캐싱의 대안 또는 보완으로 사용할 수 있습니다. 다음 코드는 LiteDB 라이브러리를 사용하여 로컬 스토리지에 데이터를 캐싱하는 방법을 보여줍니다. `LocalStorageCache` 클래스에는 캐시를 관리하기 위한 주요 함수가 포함되어 있습니다.

```
using System;
using LiteDB;

public class LocalStorageCache
{
    private static string _liteDbPath = @"Filename=LocalCache.db";

    public static T GetData<T>(string key, Func<T> getData)
    {
        using (var db = new LiteDatabase(_liteDbPath))
        {
            var collection = db.GetCollection<T>("cache");
            var item = collection.FindOne(Query.EQ("_id", key));

            if (item != null)
            {
                return item;
            }
        }

        T data = getData();

        using (var db = new LiteDatabase(_liteDbPath))
        {
            var collection = db.GetCollection<T>("cache");
            collection.Upsert(new BsonValue(key), data);
        }

        return data;
    }

    public static void ClearCache(string key)
    {
        using (var db = new LiteDatabase(_liteDbPath))
        {
            var collection = db.GetCollection("cache");
```

```
            collection.Delete(key);
        }
    }
}

public class Program
{
    public static void Main()
    {
        string cacheKey = "sample_data";

        Func<string> getSampleData = () =>
        {
            // Replace this with your data retrieval logic
            return "Sample data";
        };

        string data = LocalStorageCache.GetData(cacheKey, getSampleData);
        Console.WriteLine("Data: " + data);
    }
}
```

자주 변경되지 않는 정적 캐시 또는 정적 파일이 있는 경우 이러한 파일을 Amazon Simple Storage Service(Amazon S3) 객체 스토리지에 저장할 수도 있습니다. 애플리케이션은 시작 시 정적 캐시 파일을 검색하여 로컬에서 사용할 수 있습니다. .NET을 사용하여 Amazon S3에서 파일을 검색하는 방법에 대한 자세한 내용은 Amazon S3 설명서의 객체 다운로드를 참조하세요.

DAX를 사용한 캐싱

모든 애플리케이션 인스턴스에서 공유할 수 있는 캐싱 계층을 사용할 수 있습니다. DynamoDB Accelerator(DAX)는 DynamoDB용 완전 관리형 고가용성 인 메모리 캐시로, 10배의 성능 개선을 제공할 수 있습니다. DAX를 사용하면 DynamoDB 테이블에서 읽기 용량 단위를 오버프로비저닝할 필요성을 줄여 비용을 절감할 수 있습니다. 이는 읽기 작업이 많고 개별 키에 대해 반복 읽기가 필요한 워크로드에 특히 유용합니다.

DynamoDB는 온디맨드 또는 프로비저닝된 용량으로 가격이 책정되므로 월별 읽기 및 쓰기 수가 비용에 영향을 미칩니다. 많은 워크로드를 읽은 경우 DAX 클러스터는 DynamoDB 테이블의 읽기 수를 줄여 비용을 절감하는 데 도움이 될 수 있습니다. DAX를 설정하는 방법에 대한 지침은 DynamoDB 설명서의 DynamoDB Accelerator(DAX)를 사용한 인 메모리 가속화를 참조하세요. DynamoDB .NET 애플리케이션 통합에 대한 자세한 내용은 YouTube에서 'Amazon DynamoDB DAX를 '내 ASP.NET애플리케이션에 통합'을 참조하세요.

추가 리소스

- DynamoDB Accelerator(DAX)를 사용한 인 메모리 가속화 - Amazon DynamoDB(DynamoDB 설명서)
- Amazon DynamoDB DAX를 ASP.NET 애플리케이션에 통합(YouTube)
- 객체 다운로드(Amazon S3 설명서)

서버리스 .NET 고려

개요

서버리스 컴퓨팅은 애플리케이션을 구축하고 배포하는 데 널리 사용되는 접근 방식이 되었습니다. 이는 주로 서버리스 접근 방식이 최신 아키텍처를 구축할 때 제공하는 확장성과 민첩성 때문입니다. 그러나 일부 시나리오에서는 서버리스 컴퓨팅이 비용에 미치는 영향을 고려하는 것이 중요합니다.

Lambda는 개발자가 전용 서버 없이 코드를 실행할 수 있는 서버리스 컴퓨팅 플랫폼입니다. Lambda는 인프라 비용을 절감하려는 .NET 개발자에게 특히 매력적인 옵션입니다. Lambda를 사용하면 .NET 개발자는 확장성이 뛰어나고 비용 효율적일 수 있는 애플리케이션을 개발하고 배포할 수 있습니다. 서버리스 접근 방식을 사용하면 개발자는 더 이상 애플리케이션 요청을 처리하기 위해 서버를 프로비저닝하지 않습니다. 대신 개발자는 온디맨드 방식으로 실행되는 함수를 생성할 수 있습니다. 따라서 가상 머신을 실행, 관리 및 확장하는 것보다 서버리스 접근 방식이 더 확장 가능하고 관리 가능하며 비용 효율적일 수 있습니다. 따라서 사용량이 적은 리소스나 서버 유지 관리 비용에 대해 걱정할 필요 없이 애플리케이션에서 사용하는 리소스에 대해서만 비용을 지불하면 됩니다.

개발자는 최신 교차 플랫폼 .NET 버전을 사용하여 빠르고 효율적이며 비용 효율적인 서버리스 애플리케이션을 구축할 수 있습니다. .NET Core 이상 버전은 이전 .NET Framework 버전보다 서버리스 플랫폼에서 실행하는 데 더 적합한 무료 오픈 소스 프레임워크입니다. 이를 통해 개발자는 개발 시간을 줄이고 애플리케이션 성능을 높일 수 있습니다. 또한 최신 .NET은 C# 및 F#을 비롯한 다양한 프로그래밍 언어를 지원합니다. 이러한 이유로 클라우드에서 최신 아키텍처를 구축하려는 개발자에게 매력적인 옵션입니다.

이 섹션에서는 Lambda를 서버리스 옵션으로 사용하여 비용을 절감하는 방법을 설명합니다. Lambda 함수의 실행 프로파일을 미세 조정하고, Lambda 함수의 메모리 할당 크기를 올바르게 조정하고, 네이티브 AOT를 사용하고, Graviton 기반 함수로 이동하여 비용을 추가로 최적화할 수 있습니다.

비용 영향

비용을 절감할 수 있는 정도는 할당된 메모리 양과 각 함수의 기간 외에도 서버리스 함수가 실행할 실행 수를 비롯한 여러 요인에 따라 달라집니다.는 매월 100만 개의 무료 요청과 매월 400,000GB 초의 컴퓨팅 시간을 포함하는 프리 티어를 AWS Lambda 제공합니다. 이러한 프리 티어 한도에 도달하거나 그 즈음에 있는 워크로드에 대한 월별 비용을 크게 줄일 수 있습니다.

Lambda 함수를 대상으로 하는 로드 밸런서를 사용할 때 추가 비용이 발생할 수도 있습니다. 이는 Lambda 대상에 대해 로드 밸런서가 처리하는 데이터의 양으로 계산됩니다.

비용 최적화 권장 사항

Lambda 함수의 적절한 크기

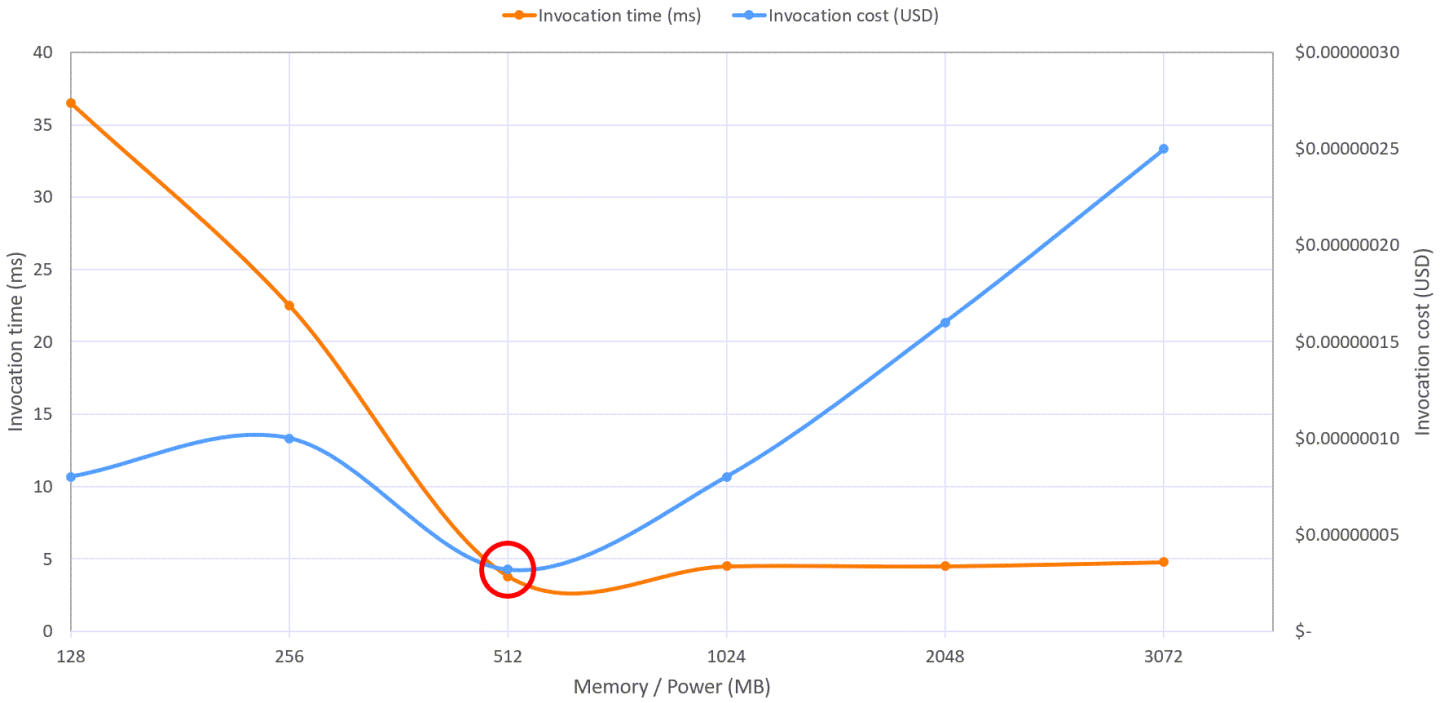
올바른 크기 조정은 .NET 기반 Lambda 함수의 비용 최적화를 위한 필수 사례입니다. 이 프로세스에는 코드를 변경할 필요 없이 성능과 비용 효율성의 균형을 맞추는 최적의 메모리 구성을 식별하는 작업이 포함됩니다.

128MB에서 최대 10,240MB 범위의 Lambda 함수에 대한 메모리를 구성하여 호출 중에 사용할 수 있는 vCPU의 양도 조정합니다. 이렇게 하면 메모리 또는 CPU 바운드 애플리케이션이 실행 중에 추가 리소스에 액세스할 수 있으므로 호출 기간과 전체 비용이 잠재적으로 감소합니다.

그러나 .NET 기반 Lambda 함수에 대한 최적의 구성을 식별하는 것은 수동적이고 시간이 많이 걸리는 프로세스일 수 있으며, 특히 변경이 빈번한 경우 더욱 그렇습니다. AWS Lambda Power Tuning 도구를 사용하면 예제 페이로드에 대해 메모리 구성 세트를 분석하여 적절한 구성을 식별할 수 있습니다.

예를 들어 .NET 기반 Lambda 함수의 메모리를 늘리면 성능에 영향을 주지 않고 총 호출 시간이 향상되고 비용이 절감될 수 있습니다. 함수에 대한 최적의 메모리 구성은 다를 수 있습니다. AWS Lambda Power Tuning 도구는 각 함수에 대해 가장 비용 효율적인 구성을 식별하는 데 도움이 될 수 있습니다.

다음 예제 차트에서는이 Lambda 함수의 메모리가 증가함에 따라 총 호출 시간이 향상됩니다. 이로 인해 함수의 원래 성능에 영향을 주지 않고 총 실행 비용이 절감됩니다. 이 함수의 경우 함수의 최적 메모리 구성은 512MB입니다. 리소스 사용률이 각 간접 호출의 총 비용에 가장 효율적이기 때문입니다. 이는 함수마다 다르며 Lambda 함수에서 도구를 사용하면 적절한 크기 조정의 이점을 얻을 수 있는지 식별할 수 있습니다.



새 업데이트가 릴리스될 때 통합 테스트의 일환으로이 연습을 정기적으로 완료하는 것이 좋습니다. 자주 업데이트되지 않는 경우이 연습을 주기적으로 수행하여 함수가 조정되고 크기가 올바른지 확인합니다. Lambda 함수에 적합한 메모리 설정을 식별한 후 프로세스에 적절한 크기 조정을 추가할 수 있습니다. AWS Lambda Power Tuning 도구는 새 코드를 릴리스하는 동안 CI/CD 워크플로에서 사용할 수 있는 프로그래밍 방식 출력을 생성합니다. 이렇게 하면 메모리 구성을 자동화할 수 있습니다.

AWS Lambda Power Tuning 도구를 무료로 다운로드할 수 있습니다. 도구 사용 방법에 대한 지침은 GitHub에서 상태 시스템을 실행하는 방법을 참조하세요.

Lambda는 .NET 애플리케이션을 사전 컴파일할 수 있는 네이티브 AOT도 지원합니다. 이렇게 하면 .NET 함수의 실행 시간을 줄여 비용을 절감할 수 있습니다. 네이티브 AOT 함수 생성에 대한 자세한 내용은 Lambda 설명서의 네이티브 AOT 컴파일이 있는 .NET 함수를 참조하세요.

유휴 대기 시간 방지

Lambda 함수 기간은 결제 계산에 사용되는 하나의 차원입니다. 함수 코드가 차단 호출을 수행하면 응답 수신을 기다리는 시간에 대한 요금이 청구됩니다. 이 대기 시간은 Lambda 함수가 함께 연결되거나 함수가 다른 함수의 오케스트레이터 역할을 할 때 증가할 수 있습니다. 배치 작업 또는 주문 전달 시스템과 같은 워크플로가 있는 경우 관리 오버헤드가 추가됩니다. 또한 최대 Lambda 제한 시간인 15분 내에 모든 워크플로 로직 및 오류 처리를 완료하지 못할 수 있습니다.

함수 코드에서이 로직을 처리하는 대신 워크플로의 오케스트레이터AWS Step Functions로 사용할 솔루션을 다시 설계하는 것이 좋습니다. 표준 워크플로를 사용하는 경우 워크플로의 총 기간이 아닌 워크

플로 내의 각 상태 전환에 대해 요금이 청구됩니다. 또한 재시도, 대기 조건, 오류 워크플로 및 콜백에 대한 지원을 상태 조건으로 이동하여 Lambda 함수가 비즈니스 로직에 집중할 수 있도록 할 수 있습니다. 자세한 내용은 AWS 컴퓨팅 블로그의 AWS Lambda 비용 최적화 - 2부를 참조하세요.

Graviton 기반 함수로 이동

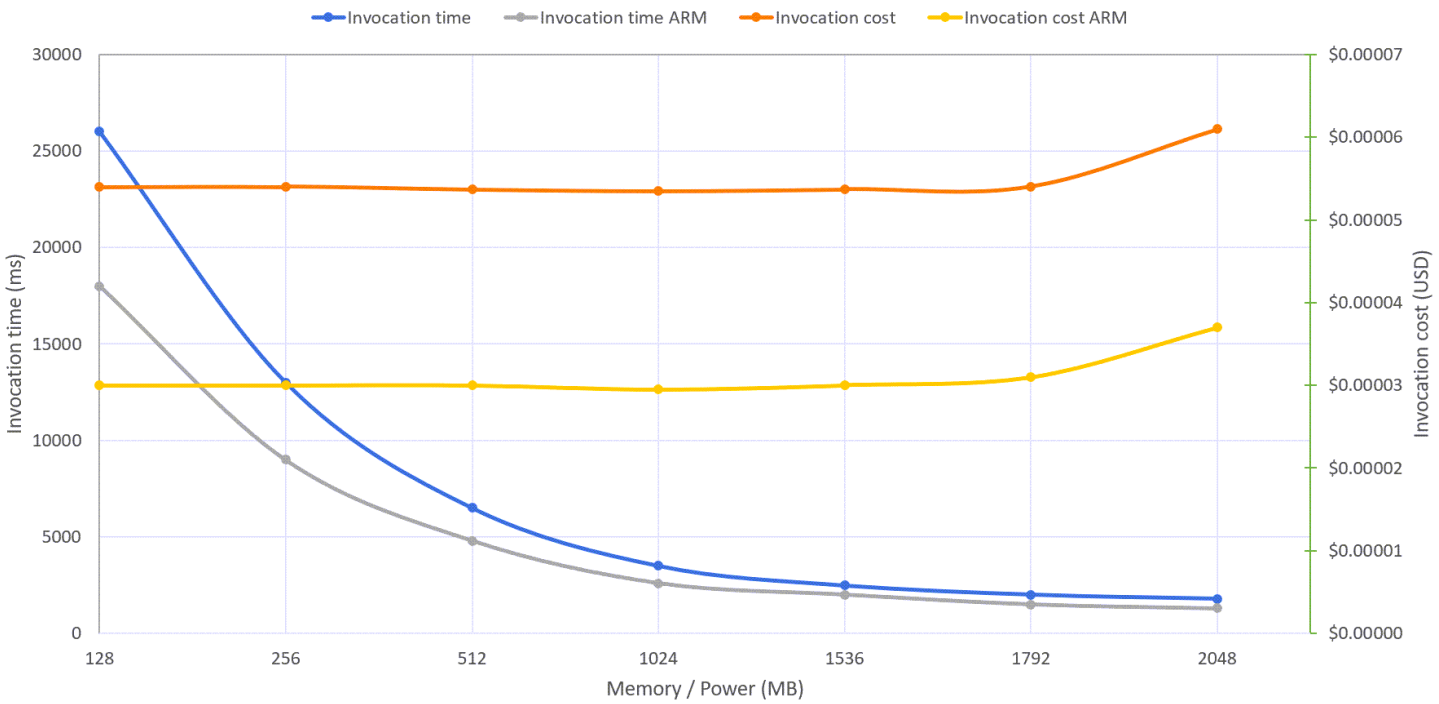
이제 차세대 Graviton2 프로세서로 구동되는 Lambda 함수를 정식 버전으로 사용할 수 있습니다. ARM 기반 프로세서 아키텍처를 사용하는 Graviton2 함수는 다양한 서버리스 워크로드에 대해 20% 저렴한 비용으로 최대 19% 더 나은 성능을 제공하도록 설계되었습니다. 지연 시간이 짧고 성능이 더 좋으면 Graviton2 프로세서로 구동되는 함수는 미션 크리티컬 서버리스 애플리케이션을 구동하는 데 적합합니다.

Graviton 기반 Lambda 함수로 마이그레이션하는 것은 Lambda 비용을 최적화하려는 .NET 개발자에게 비용 효율적인 옵션일 수 있습니다. Graviton 기반 함수는 기존 x86 프로세서 대신 ARM 기반 프로세서를 사용합니다. 이로 인해 성능을 저하시키지 않고 비용을 크게 절감할 수 있습니다.

Graviton 기반 함수로 전환하면 몇 가지 이점이 있지만 고려해야 할 몇 가지 과제와 고려 사항도 있습니다. 예를 들어 Graviton 기반 함수는 Amazon Linux 2를 사용해야 하며, 이는 일부 .NET 애플리케이션과 호환되지 않을 수 있습니다. 또한 ARM 기반 프로세서와 호환되지 않는 타사 라이브러리 또는 종속성과의 호환성 문제가 있을 수 있습니다.

.NET Framework 애플리케이션을 실행 중이고 Lambda를 통해 서버리스를 활용하려는 경우 Porting Assistant for .NET을 사용하여 애플리케이션을 최신 .NET으로 이식하는 것을 고려할 수 있습니다. 이를 통해 레거시 .NET 애플리케이션을 최신 .NET으로 이식하는 속도를 높여 Linux에서 애플리케이션을 실행할 수 있습니다.

다음 차트는 주요 숫자를 계산하는 함수에 대한 x86 및 ARM/Graviton2 아키텍처 결과를 비교합니다.



함수가 단일 스레드를 사용하고 있습니다. 메모리가 1.8GB로 구성된 경우 두 아키텍처의 최소 지속 시간이 보고됩니다. 그 이상으로 Lambda 함수는 1개 이상의 vCPU에 액세스할 수 있지만이 경우 함수는 추가 전원을 사용할 수 없습니다. 동일한 이유로 비용은 최대 1.8GB의 메모리로 안정적입니다. 메모리가 많을수록이 워크로드에 대한 추가 성능 이점이 없기 때문에 비용이 증가합니다. Graviton2 프로세서는이 컴퓨팅 집약적 함수에 대해 더 나은 성능과 더 낮은 비용을 명확하게 제공합니다.

Graviton에서 및 ARM 기반 프로세서를 사용하도록 함수를 구성하려면 다음을 수행합니다.

1. 에 로그인한 AWS Management Console 다음 Lambda 콘솔을 엽니다.
2. 함수 생성(Create function)을 선택합니다.
3. 함수 이름에 이름을 입력합니다.
4. 런타임에서 .NET 6(C#/PowerShell)을 선택합니다.
5. 아키텍처에서 arm64를 선택합니다.
6. 필요한 추가 구성을 수행한 다음 함수 생성을 선택합니다.

추가 리소스

- 대상으로서 Lambda 함수(AWS 문서)
- 를 사용하여 AWS Lambda 비용 및 성능 최적화 AWS Compute Optimizer(AWS 컴퓨팅 블로그)
- AWS Lambda 비용 최적화 - 1부(AWS 컴퓨팅 블로그)

- AWS Lambda 비용 최적화 - 2부(AWS 컴퓨팅 블로그)
- .NET 7을 AWS Lambda 사용하여에서 서버리스 .NET 애플리케이션 빌드(AWS 컴퓨팅 블로그)

특별히 구축된 데이터베이스 고려

개요

Microsoft 기반 워크로드를 실행할 때 가장 비용이 많이 드는 측면 중 하나는 SQL Server와 같은 상용 데이터베이스의 라이선스에서 비롯됩니다. 기업은 SQL Server를 선택한 데이터베이스 플랫폼으로 표준화하는 경우가 많으며 조직의 개발 문화에 정체됩니다. 개발자는 일반적으로 사용 사례에 관계없이 관계형 SQL Server 기반 모델을 선택합니다. 이유는 다음과 같습니다.

- 비즈니스에 이미 사용 가능한 SQL Server 인스턴스 및/또는 라이선스가 있습니다.
- 팀은 공유 라이브러리, ORMs 및 비즈니스 로직을 사용하여 SQL 프로그래밍 모델에 적응했습니다.
- 경영진은 대안을 인식하지 못합니다.
- 개발자는 대안을 알지 못합니다.

특별히 구축된 데이터베이스는 사용 사례의 데이터 액세스 패턴을 수용할 수 있습니다. 이러한 데이터베이스는 보다 최신 아키텍처(예: 마이크로서비스)를 채택하고 개별 애플리케이션의 범위가 좁아짐에 따라 기업에서 점점 더 많이 채택되고 있습니다.

특별히 구축된 데이터베이스는 관계형 모델을 제외하거나 NoSQL(비관계형) 모델이 필요하지 않습니다. 실제로 관계형 데이터베이스는 워크로드의 특정 요구 사항에 따라 선택할 때 특별히 빌드된 것으로 간주됩니다. 특별히 구축된 데이터베이스를 사용하면 팀이 .NET 애플리케이션과 관련된 데이터베이스 비용을 절감하는 동시에 확장성, 복원력, 차별화되지 않은 과중한 작업 감소와 같은 표준 클라우드 이점을 얻을 수 있습니다.

다음 표에는에서 제공하는 목적별 데이터베이스가 나와 있습니다 AWS.

데이터베이스	유형	특성
Amazon Aurora PostgreSQL 또는 Amazon Aurora MySQL	관계형	데이터가 고정된 구조를 갖는 사용 사례 관계형 데이터베이스는 ACID 트랜잭션을 통해 데이터 일관성을 자연스럽게 유지합니다.

데이터베이스	유형	특성
Amazon DynamoDB	키-값 페어	해시 테이블 데이터 구조를 사용하여 데이터를 저장하는 NoSQL 데이터베이스 비정형 데이터의 고성능 스토리지 및 검색 사용 사례에는 사용자 프로필, 세션 상태 및 장바구니 데이터가 포함됩니다.
Amazon ElastiCache	인 메모리	밀리초 미만의 액세스 시간으로 비정형 데이터를 메모리에 저장하는 고성능 NoSQL 데이터베이스 사용자 세션과 같은 자주 액세스하는 임시 데이터와 다른 느린 데이터 스토어 앞의 캐싱 계층으로 사용됩니다. ElastiCache(Redis OSS) 및 ElastiCache(Memcached) 모두에 대한 지원 포함
Amazon MemoryDB	내구성이 뛰어난 인 메모리	내구성이 뛰어난 스토리지를 갖춘 Redis 호환 목적별 데이터베이스
Amazon Timestream	시계열	시간순으로 처리량이 많은 데이터를 수집하도록 설계된 데이터베이스 사용 사례에는 사물 인터넷(IoT) 애플리케이션과 지표 또는 원격 측정 데이터 저장이 포함됩니다.

데이터베이스	유형	특성
Amazon DocumentDB	문서	규정된 구조 또는 다른 데이터에 대한 강제 관계 없이 데이터를 저장하는 NoSQL 데이터베이스 제품 카탈로그와 같은 읽기 집약적인 워크로드에 자주 사용됩니다.
Amazon Neptune	그래프	데이터와 데이터 항목 간 연결 표현을 모두 포함하는 NoSQL 데이터베이스 사용 사례에는 사기 탐지, 추천 엔진 및 소셜 애플리케이션이 포함됩니다.
Amazon Keyspaces	와이드 열	Apache Cassandra 기반 고성능 분산 데이터베이스 사용 사례에는 IoT 애플리케이션, 이벤트 처리 및 게임 애플리케이션이 포함됩니다.

특별히 구축된 데이터베이스 채택의 중요한 동인은 상용 라이선스 제거에서 비롯될 수 있습니다. 그러나 DynamoDB(온디맨드 모드 포함), Aurora, Amazon Neptune, Amazon Keyspaces와 같은 데이터베이스의 자동 크기 조정 기능을 사용하면 최대 사용량이 아닌 평균 사례에 맞게 용량을 프로비저닝할 수 있습니다. Timestream과 같은 목적별 데이터베이스는 서버리스이며 사전 프로비저닝 없이 수요에 맞게 자동으로 확장됩니다.

AWS 는 특별히 구축된 오픈 소스 호환 관계형 데이터베이스를 사용하려는 경우 Babelfish for Aurora PostgreSQL을 제공하지만 애플리케이션에 중요한 코드 변경을 수행할 수 없거나 수행할 의향이 없는 경우 Babelfish for Aurora PostgreSQL을 제공합니다. 경우에 따라 Babelfish를 사용하면 변경 없이 기존 SQL Server 액세스 코드를 사용할 수 있습니다.

애플리케이션을 위해 특별히 구축된 관계형 데이터베이스를 선택할 때는 애플리케이션에 필요한 것과 동일한(또는 기능적으로 동등한) 기능을 유지하는 것이 중요합니다. 이 권장 사항은 용도에 맞게 구축된 데이터베이스를 애플리케이션의 기본 데이터 스토어로 다룹니다. 특정 애플리케이션(예: 캐싱)은 다른 권장 사항에서 다룹니다.

비용 영향

.NET 워크로드에 특별히 구축된 데이터베이스를 채택하면 컴퓨팅 소비/비용에 직접적인 영향을 미칠 가능성은 낮지만 .NET 애플리케이션에서 사용하는 데이터베이스 서비스의 비용에 직접적인 영향을 미칠 수 있습니다. 실제로 민첩성, 확장성, 복원력 및 데이터 내구성의 추가 이점과 비교할 때 비용 절감이 보조 목표일 수 있습니다.

애플리케이션을 위해 특별히 구축된 데이터베이스를 선택하고 이를 효과적으로 사용하도록 데이터 전략을 재설계하는 전체 프로세스를 설명하는 것은이 가이드의 범위를 벗어납니다. 자세한 내용은 AWS 자습서 디렉터리의 목적별 데이터베이스를 참조하세요.

다음 표에는 SQL Server를 특별히 구축된 데이터베이스로 대체하여 애플리케이션 비용을 변경하는 방법의 몇 가지 예가 나와 있습니다. 이는 단순히 대략적인 추정치입니다. 정확한 프로덕션 비용을 계산하려면 실제 워크로드의 벤치마크와 최적화가 필요합니다.

다음은 온디맨드 컴퓨팅과의 단일 인스턴스 데이터베이스인 100GB SSD를 포함하여 일반적으로 사용되는 목적별 데이터베이스 추정치입니다`us-east-1`. 라이선스 비용에는 SQL Server 라이선스와 소프트웨어 보증이 포함됩니다.

다음 표에는 상용 데이터베이스 예제의 예상 비용이 나와 있습니다.

데이터베이스 엔진	라이선싱 모델	인스턴스 유형/사양	AWS 컴퓨팅 + 스토리지 비용	라이선스 비용	총 월별 비용
Amazon EC2의 SQL Server Standard 에디션	라이선스 포함	r6i.2xlarge(CPU 8개/64GB RAM)	1,345.36 USD	$0.00	1,345.36 USD
Amazon EC2의 SQL Server	라이선스 포함	r6i.2xlarge(CPU 8	2,834.56 USD	$0.00	2,834.56 USD

데이터베이스 엔진	라이선싱 모델	인스턴스 유형/사양	AWS 컴퓨팅 + 스토리지 비용	라이선스 비용	총 월별 비용
Enterprise 에디션		개/64GB RAM)			
Amazon EC2의 SQL Server Standard 에디션	BYOL	r6i.2xlarge(CPU 8개/64GB RAM)	644.56 USD	456.00 USD	1,100.56 USD
Amazon EC2의 SQL Server Enterprise 에디션	BYOL	r6i.2xlarge(CPU 8개/64GB RAM)	644.56 USD	1,750.00 USD	2,394.56 USD
Amazon RDS의 SQL Server Standard 에디션		db.r6i.2xlarge(CPU 8개/64GB RAM)	2,318.30 USD	$0.00	2,318.30 USD
Amazon RDS의 SQL Server Enterprise Edition		db.r6i.2xlarge(CPU 8개/64GB RAM)	3,750.56 USD	0.00 USD	3,750.56 USD

다음 표에는 특별히 구축된 예제의 예상 비용이 나와 있습니다.

데이터베이스 엔진	인스턴스 유형/사양	AWS 컴퓨팅 + 스토리지 비용	라이선스 비용	총 월별 비용
Amazon Aurora PostgreSQL	r6g.2xlarge(CPU 8개/64GB RAM)	855.87 USD	0.00 USD	855.87 USD
DynamoDB	프로비저닝된 기본 100WCU/400RCU	72.00 USD		72.00 USD
Amazon DocumentDB	db.r6i.2x large(CPU 8개/64GB RAM)	$778.60		$778.60

> **Important**
>
> 이 표는 처음 3년 구매 기간 동안 소프트웨어 보증이 포함된 SQL Server의 예상 라이선스 비용을 기준으로 합니다. SQL Server Standard 에디션의 경우: 4,100 USD, 코어 팩 2개, 3년. SQL Server Enterprise 에디션의 경우: 15,700 USD, 코어 팩 2개, 3년.

목적별 데이터베이스를 채택하기 전에 비용 영향을 고려하는 것이 좋습니다. 예를 들어, 용도가 지정된 데이터베이스를 사용하도록 애플리케이션을 업데이트하는 비용은 애플리케이션 및 소스 데이터베이스의 복잡성과 관련이 있습니다. 이 아키텍처 전환을 계획할 때는 총 소유 비용을 고려해야 합니다. 여기에는 애플리케이션 리팩터링, 새로운 기술에 대한 직원 기술 향상, 각 워크로드에 예상되는 성능 및 소비에 대한 신중한 계획이 포함됩니다. 거기에서 투자가 비용 절감의 가치가 있는지 결정할 수 있습니다. 대부분의 경우 end-of-support 제품을 유지 관리하는 것은 보안 및 규정 준수 위험이며, 이를 해결하는 데 드는 비용은 초기 투자와 노력의 가치가 있습니다.

비용 최적화 권장 사항

SQL Server에 액세스하는 .NET 애플리케이션의 경우 특별히 구축된 관계형 데이터베이스를 위한 대체 라이브러리가 있습니다. 애플리케이션에서 이러한 라이브러리를 구현하여 유사한 SQL Server 애플리케이션 기능을 대체할 수 있습니다.

다음 표에서는 많은 일반적인 시나리오에서 사용할 수 있는 일부 라이브러리를 강조합니다.

라이브러리	데이터베이스	에 대한 대체	프레임워크 호환성
Npgsql 엔터티 프레임워크 코어 공급자	Amazon Aurora PostgreSQL	Entity Framework Core SQL Server 공급자	최신 .NET
Npgsql Entity Framework 6 공급자	Amazon Aurora PostgreSQL	Entity Framework 6.0 SQL Server 공급자	.NET Framework
Npgsql(ADO.NET:// 호환 PostgreSQL 라이브러리)	Amazon Aurora PostgreSQL	ADO.NET	.NET Framework/현대 .NET
MySQL 개체 프레임워크 코어 공급자	Amazon Aurora MySQL	Entity Framework Core SQL Server 공급자	최신 .NET
Pomelo.EntityFrameworkCore.MySql	Amazon Aurora MySQL	Entity Framework Core SQL Server 공급자	최신 .NET

Babelfish를 사용하여 Amazon Aurora PostgreSQL에 연결하려면 특별한 코딩이 필요하지 않습니다. 그러나 모든 코드는 사용 전에 철저하게 테스트해야 합니다.

다른 용도에 맞게 구축된 데이터베이스에는 용도에 맞게 구축된 데이터베이스에 액세스할 수 있는 .NET 호환 라이브러리에 액세스하기 위한 라이브러리가 있습니다. 그러한 예는 다음과 같습니다.

- Amazon DynamoDB NoSQL 데이터베이스 사용(AWS SDK for .NET 문서)
- MongoDB C# 드라이버(MongoDB 설명서)
- .NET(Timestream 설명서)
- Cassandra .NET Core 클라이언트 드라이버를 사용하여 프로그래밍 방식으로 Amazon Keyspaces에 액세스(Amazon Keyspaces 설명서)
- .NET을 사용하여 Neptune DB 인스턴스에 연결(Neptune 설명서)

전용 빌드 데이터베이스로 마이그레이션하는 경우에서 다음 도구를 사용하여 마이그레이션 프로세스를 AWS 지원할 수 있습니다.

- AWS Schema Conversion Tool (AWS SCT)를 사용하면 SQL Server 스키마를 Amazon Aurora 및 Amazon DynamoDB로 변환할 수 있습니다.
- AWS Database Migration Service (AWS DMS)를 사용하면 SQL Server에서 Aurora 또는 DynamoDB로 데이터를 한 번 또는 지속적으로 마이그레이션할 수 있습니다.
- Babelfish Compass는 Babelfish for Aurora PostgreSQL과 함께 사용할 SQL Server 데이터베이스의 호환성을 확인하는 데 도움이 될 수 있습니다.

추가 리소스

- SQL Server를 Amazon Aurora PostgreSQL로 마이그레이션하기 위한 지침(AWS 데이터베이스 블로그)
- .NET Modernization 워크숍(AWS Workshop Studio)
- Babelfish APP Modernization Immersion Day(AWS Workshop Studio)
- .NET Immersion Day(AWS 워크숍 스튜디오)
- .NET(GitHub)을 사용하여 Amazon Timestream 시작하기 GitHub
- (AWS 프레젠테이션)의 최신 .NET 애플리케이션을 위해 특별히 구축된 데이터베이스 AWS

다음 단계

이 가이드 검토를 완료한 후 다음 단계를 수행하여 MACO를 구현하는 것이 좋습니다.

1. MACO 전문가에게 문의하세요. MACO 전문가가 질문에 답변하고 문제를 해결할 수 있습니다. 이미 AWS 계정 팀과 협력하고 있는 경우 팀에 문의하여 MACO 전문가에게 도움을 요청하세요. 계정 팀이 없는 경우 optimize-microsoft@amazon.com에 문의하세요.
2. 권장 사항을 적용합니다. 이 가이드와 MACO 전문가와의 대화에서 학습한 권장 사항, 모범 사례 및 전략을 적용합니다.
3. 비용 변경을 추적합니다. 워크로드에 태그를 지정하고 AWS Cost Explorer 및와 같은 서비스를 사용하여 세부 비용 추적, 모니터링 및 제어를 AWS Budgets 수행합니다.

문서 기록

아래 표에 이 가이드의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
SQL Server 업데이트	SQL Server 워크로드에 CPU 최적화 섹션을 업데이트했습니다.	2024년 10월 25일
SQL Server 및 컨테이너 업데이트	Compute Optimizer를 사용하여 SQL Server 크기 최적화, SQL Server 워크로드에 대한 Trusted Advisor 권장 사항 검토, App2Container를 사용하여 Windows 애플리케이션 리플랫폼 섹션을 추가했습니다.	2024년 6월 29일
SQL Server 라이선스 최적화	Compute Optimizer를 사용하여 SQL Server 라이선스 최적화 섹션을 추가했습니다.	2024년 5월 22일
최초 게시	—	2023년 12월 21일

AWS 권장 가이드 용어집

다음은 AWS 권장 가이드에서 제공하는 전략, 가이드 및 패턴에서 일반적으로 사용되는 용어입니다. 용어집 항목을 제안하려면 용어집 끝에 있는 피드백 제공 링크를 사용하십시오.

숫자

7가지 전략

애플리케이션을 클라우드로 이전하기 위한 7가지 일반적인 마이그레이션 전략 이러한 전략은 Gartner가 2011년에 파악한 5가지 전략을 기반으로 하며 다음으로 구성됩니다.

- 리팩터링/리아키텍트 - 클라우드 네이티브 기능을 최대한 활용하여 애플리케이션을 이동하고 해당 아키텍처를 수정함으로써 민첩성, 성능 및 확장성을 개선합니다. 여기에는 일반적으로 운영 체제와 데이터베이스 이식이 포함됩니다. 예: 온프레미스 Oracle 데이터베이스를 Amazon Aurora PostgreSQL 호환 버전으로 마이그레이션합니다.
- 리플랫포밍(리프트 앤드 리셰이프) - 애플리케이션을 클라우드로 이동하고 일정 수준의 최적화를 도입하여 클라우드 기능을 활용합니다. 예:에서 온프레미스 Oracle 데이터베이스를 Oracle용 Amazon Relational Database Service(RDS)로 마이그레이션합니다 AWS 클라우드.
- 재구매(드롭 앤드 숍) - 일반적으로 기존 라이선스에서 SaaS 모델로 전환하여 다른 제품으로 전환합니다. 예: 고객 관계 관리(CRM) 시스템을 Salesforce.com 마이그레이션합니다.
- 리호스팅(리프트 앤드 시프트) - 애플리케이션을 변경하지 않고 클라우드로 이동하여 클라우드 기능을 활용합니다. 예:의 EC2 인스턴스에서 온프레미스 Oracle 데이터베이스를 Oracle로 마이그레이션합니다 AWS 클라우드.
- 재배치(하이퍼바이저 수준의 리프트 앤 시프트) - 새 하드웨어를 구매하거나, 애플리케이션을 다시 작성하거나, 기존 운영을 수정하지 않고도 인프라를 클라우드로 이동합니다. 온프레미스 플랫폼에서 동일한 플랫폼의 클라우드 서비스로 서버를 마이그레이션합니다. 예: Microsoft Hyper-V 애플리케이션을 로 마이그레이션합니다 AWS.
- 유지(보관) - 소스 환경에 애플리케이션을 유지합니다. 대규모 리팩터링이 필요하고 해당 작업을 나중으로 연기하려는 애플리케이션과 비즈니스 차원에서 마이그레이션할 이유가 없어 유지하려는 레거시 애플리케이션이 여기에 포함될 수 있습니다.
- 사용 중지 - 소스 환경에서 더 이상 필요하지 않은 애플리케이션을 폐기하거나 제거합니다.

A

ABAC

속성 기반 액세스 제어를 참조하세요.

추상화된 서비스

관리형 서비스를 참조하세요.

ACID

원자성, 일관성, 격리, 내구성을 참조하세요.

능동-능동 마이그레이션

양방향 복제 도구 또는 이중 쓰기 작업을 사용하여 소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되고, 두 데이터베이스 모두 마이그레이션 중 연결 애플리케이션의 트랜잭션을 처리하는 데이터베이스 마이그레이션 방법입니다. 이 방법은 일회성 전환이 필요한 대신 소규모의 제어된 배치로 마이그레이션을 지원합니다. 더 유연하지만 액티브-패시브 마이그레이션보다 더 많은 작업이 필요합니다.

능동-수동 마이그레이션

소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되지만 소스 데이터베이스만 연결 애플리케이션의 트랜잭션을 처리하고 데이터는 대상 데이터베이스로 복제되는 데이터베이스 마이그레이션 방법입니다. 대상 데이터베이스는 마이그레이션 중 어떤 트랜잭션도 허용하지 않습니다.

집계 함수

행 그룹에서 작동하고 그룹에 대한 단일 반환 값을 계산하는 SQL 함수입니다. 집계 함수의 예로는 `SUM` 및 `MAX`가 있습니다.

AI

인공 지능을 참조하세요.

AIOps

인공 지능 작업을 참조하세요.

익명화

데이터세트에서 개인 정보를 영구적으로 삭제하는 프로세스입니다. 익명화는 개인 정보 보호에 도움이 될 수 있습니다. 익명화된 데이터는 더 이상 개인 데이터로 간주되지 않습니다.

안티 패턴

솔루션이 다른 솔루션보다 비생산적이거나 비효율적이거나 덜 효과적이어서 반복되는 문제에 자주 사용되는 솔루션입니다.

애플리케이션 제어

맬웨어로부터 시스템을 보호하기 위해 승인된 애플리케이션만 사용할 수 있는 보안 접근 방식입니다.

애플리케이션 포트폴리오

애플리케이션 구축 및 유지 관리 비용과 애플리케이션의 비즈니스 가치를 비롯하여 조직에서 사용하는 각 애플리케이션에 대한 세부 정보 모음입니다. 이 정보는 포트폴리오 검색 및 분석 프로세스의 핵심이며 마이그레이션, 현대화 및 최적화할 애플리케이션을 식별하고 우선순위를 정하는 데 도움이 됩니다.

인공 지능

컴퓨터 기술을 사용하여 학습, 문제 해결, 패턴 인식 등 일반적으로 인간과 관련된 인지 기능을 수행하는 것을 전문으로 하는 컴퓨터 과학 분야입니다. 자세한 내용은 What is Artificial Intelligence?를 참조하십시오.

인공 지능 운영(AIOps)

기계 학습 기법을 사용하여 운영 문제를 해결하고, 운영 인시던트 및 사용자 개입을 줄이고, 서비스 품질을 높이는 프로세스입니다. AWS 마이그레이션 전략에서 AIOps가 사용되는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

비대칭 암호화

한 쌍의 키, 즉 암호화를 위한 퍼블릭 키와 복호화를 위한 프라이빗 키를 사용하는 암호화 알고리즘입니다. 퍼블릭 키는 복호화에 사용되지 않으므로 공유할 수 있지만 프라이빗 키에 대한 액세스는 엄격히 제한되어야 합니다.

원자성, 일관성, 격리성, 내구성(ACID)

오류, 정전 또는 기타 문제가 발생한 경우에도 데이터베이스의 데이터 유효성과 운영 신뢰성을 보장하는 소프트웨어 속성 세트입니다.

ABAC(속성 기반 액세스 제어)

부서, 직무, 팀 이름 등의 사용자 속성을 기반으로 세분화된 권한을 생성하는 방식입니다. 자세한 내용은 AWS Identity and Access Management (IAM) 설명서의 용 ABAC AWS를 참조하세요.

신뢰할 수 있는 데이터 소스

가장 신뢰할 수 있는 정보 소스로 간주되는 기본 버전의 데이터를 저장하는 위치입니다. 익명화, 편집 또는 가명화와 같은 데이터 처리 또는 수정의 목적으로 신뢰할 수 있는 데이터 소스의 데이터를 다른 위치로 복사할 수 있습니다.

가용 영역

다른 가용 영역의 장애로부터 격리 AWS 리전 되고 동일한 리전의 다른 가용 영역에 저렴하고 지연 시간이 짧은 네트워크 연결을 제공하는 내의 고유한 위치입니다.

AWS 클라우드 채택 프레임워크(AWS CAF)

조직이 클라우드로 성공적으로 전환 AWS 하기 위한 효율적이고 효과적인 계획을 개발하는 데 도움이 되는의 지침 및 모범 사례 프레임워크입니다. AWS CAF는 지침을 비즈니스, 사람, 거버넌스, 플랫폼, 보안 및 운영이라는 6가지 중점 영역으로 구성합니다. 비즈니스, 사람 및 거버넌스 관점은 비즈니스 기술과 프로세스에 초점을 맞추고, 플랫폼, 보안 및 운영 관점은 전문 기술과 프로세스에 중점을 둡니다. 예를 들어, 사람 관점은 인사(HR), 직원 배치 기능 및 인력 관리를 담당하는 이해관계자를 대상으로 합니다. 이러한 관점에서 AWS CAF는 성공적인 클라우드 채택을 위해 조직을 준비하는 데 도움이 되는 인력 개발, 교육 및 커뮤니케이션에 대한 지침을 제공합니다. 자세한 내용은 AWS CAF 웹 사이트와 AWS CAF 백서를 참조하십시오.

AWS 워크로드 검증 프레임워크(AWS WQF)

데이터베이스 마이그레이션 워크로드를 평가하고, 마이그레이션 전략을 권장하고, 작업 견적을 제공하는 도구입니다. AWS WQF는 AWS Schema Conversion Tool (AWS SCT)에 포함되어 있습니다. 데이터베이스 스키마 및 코드 객체, 애플리케이션 코드, 종속성 및 성능 특성을 분석하고 평가 보고서를 제공합니다.

B

잘못된 봇

개인 또는 조직을 방해하거나 해를 입히기 위한 봇입니다.

BCP

비즈니스 연속성 계획을 참조하세요.

동작 그래프

리소스 동작과 시간 경과에 따른 상호 작용에 대한 통합된 대화형 뷰입니다. Amazon Detective에서 동작 그래프를 사용하여 실패한 로그온 시도, 의심스러운 API 호출 및 유사한 작업을 검사할 수 있습니다. 자세한 내용은 Detective 설명서의 Data in a behavior graph를 참조하십시오.

빅 엔디안 시스템

가장 중요한 바이트를 먼저 저장하는 시스템입니다. Endianness도 참조하세요.

바이너리 분류

바이너리 결과(가능한 두 클래스 중 하나)를 예측하는 프로세스입니다. 예를 들어, ML 모델이 “이 이메일이 스팸인가요, 스팸이 아닌가요?”, '이 제품은 책인가요, 자동차인가요?' 등의 문제를 예측해야 할 수 있습니다.

블룸 필터

요소가 세트의 멤버인지 여부를 테스트하는 데 사용되는 메모리 효율성이 높은 확률론적 데이터 구조입니다.

블루/그린(Blue/Green) 배포

별개의 동일한 두 환경을 생성하는 배포 전략입니다. 현재 애플리케이션 버전은 한 환경(파란색)에서 실행하고 새 애플리케이션 버전은 다른 환경(녹색)에서 실행합니다. 이 전략을 사용하면 영향을 최소화하면서 빠르게 롤백할 수 있습니다.

bot

인터넷을 통해 자동화된 작업을 실행하고 인적 활동 또는 상호 작용을 시뮬레이션하는 소프트웨어 애플리케이션입니다. 인터넷에서 정보를 인덱싱하는 웹 크롤러와 같은 일부 봇은 유용하거나 유용합니다. 잘못된 봇이라고 하는 일부 다른 봇은 개인 또는 조직을 방해하거나 해를 입히기 위한 것입니다.

봇넷

맬웨어에 감염되어 있고 봇 셰이더 또는 봇 운영자라고 하는 단일 당사자가 제어하는 봇 네트워크입니다. Botnet은 봇과 봇의 영향을 확장하는 가장 잘 알려진 메커니즘입니다.

브랜치

코드 리포지토리의 포함된 영역입니다. 리포지토리에 생성되는 첫 번째 브랜치가 기본 브랜치입니다. 기존 브랜치에서 새 브랜치를 생성한 다음 새 브랜치에서 기능을 개발하거나 버그를 수정할 수 있습니다. 기능을 구축하기 위해 생성하는 브랜치를 일반적으로 기능 브랜치라고 합니다. 기능을 출시할 준비가 되면 기능 브랜치를 기본 브랜치에 다시 병합합니다. 자세한 내용은 About branches(GitHub 설명서)를 참조하십시오.

브레이크 글래스 액세스

예외적인 상황에서 승인된 프로세스를 통해 사용자가 일반적으로 액세스할 권한이 없는에 액세스할 수 AWS 계정 있는 빠른 방법입니다. 자세한 내용은 Well-Architected 지침의 깨진 절차 구현 표시기를 AWS 참조하세요.

브라운필드 전략

사용자 환경의 기존 인프라 시스템 아키텍처에 브라운필드 전략을 채택할 때는 현재 시스템 및 인프라의 제약 조건을 중심으로 아키텍처를 설계합니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

버퍼 캐시

가장 자주 액세스하는 데이터가 저장되는 메모리 영역입니다.

사업 역량

기업이 가치를 창출하기 위해 하는 일(예: 영업, 고객 서비스 또는 마케팅)입니다. 마이크로서비스 아키텍처 및 개발 결정은 비즈니스 역량에 따라 이루어질 수 있습니다. 자세한 내용은 백서의 AWS에서 컨테이너화된 마이크로서비스 실행의 비즈니스 역량 중심의 구성화 섹션을 참조하십시오.

비즈니스 연속성 계획(BCP)

대규모 마이그레이션과 같은 중단 이벤트가 운영에 미치는 잠재적 영향을 해결하고 비즈니스가 신속하게 운영을 재개할 수 있도록 지원하는 계획입니다.

C

CAF

AWS 클라우드 채택 프레임워크를 참조하세요.

canary 배포

최종 사용자에게 버전의 느린 증분 릴리스입니다. 확신이 드는 경우 새 버전을 배포하고 현재 버전을 완전히 교체합니다.

CCoE

Cloud Center of Excellence를 참조하세요.

CDC

변경 데이터 캡처를 참조하세요.

변경 데이터 캡처(CDC)

데이터베이스 테이블과 같은 데이터 소스의 변경 내용을 추적하고 변경 사항에 대한 메타데이터를 기록하는 프로세스입니다. 대상 시스템의 변경 내용을 감사하거나 복제하여 동기화를 유지하는 등의 다양한 용도로 CDC를 사용할 수 있습니다.

카오스 엔지니어링

시스템의 복원력을 테스트하기 위해 의도적으로 장애 또는 중단 이벤트를 도입합니다. AWS Fault Injection Service (AWS FIS)를 사용하여 AWS 워크로드에 스트레스를 주고 응답을 평가하는 실험을 수행할 수 있습니다.

CI/CD

지속적 통합 및 지속적 전달을 참조하세요.

분류

예측을 생성하는 데 도움이 되는 분류 프로세스입니다. 분류 문제에 대한 ML 모델은 이산 값을 예측합니다. 이산 값은 항상 서로 다릅니다. 예를 들어, 모델이 이미지에 자동차가 있는지 여부를 평가해야 할 수 있습니다.

클라이언트측 암호화

대상이 데이터를 AWS 서비스 수신하기 전에 로컬에서 데이터를 암호화합니다.

클라우드 혁신 센터(CCoE)

클라우드 모범 사례 개발, 리소스 동원, 마이그레이션 타임라인 설정, 대규모 혁신을 통한 조직 선도 등 조직 전체에서 클라우드 채택 노력을 추진하는 다분야 팀입니다. 자세한 내용은 AWS 클라우드 엔터프라이즈 전략 블로그의 CCoE 게시물을 참조하세요.

클라우드 컴퓨팅

원격 데이터 스토리지와 IoT 디바이스 관리에 일반적으로 사용되는 클라우드 기술 클라우드 컴퓨팅은 일반적으로 엣지 컴퓨팅 기술과 연결됩니다.

클라우드 운영 모델

IT 조직에서 하나 이상의 클라우드 환경을 구축, 성숙화 및 최적화하는 데 사용되는 운영 모델입니다. 자세한 내용은 클라우드 운영 모델 구축을 참조하십시오.

클라우드 채택 단계

조직이 로 마이그레이션할 때 일반적으로 거치는 4단계: AWS 클라우드

- 프로젝트 - 개념 증명 및 학습 목적으로 몇 가지 클라우드 관련 프로젝트 실행
- 기반 – 클라우드 채택 확장을 위한 기초 투자(예: 랜딩 존 생성, CCoE 정의, 운영 모델 구축)
- 마이그레이션 - 개별 애플리케이션 마이그레이션
- Re-invention - 제품 및 서비스 최적화와 클라우드 혁신

이러한 단계는 Stephen Orban이 블로그 게시물 The Journey Toward Cloud-First and the Stages of Adoption on the AWS 클라우드 Enterprise Strategy 블로그에서 정의했습니다. AWS 마이그레이션 전략과 어떤 관련이 있는지에 대한 자세한 내용은 마이그레이션 준비 가이드를 참조하세요.

CMDB

구성 관리 데이터베이스를 참조하세요.

코드 리포지토리

소스 코드와 설명서, 샘플, 스크립트 등의 기타 자산이 버전 관리 프로세스를 통해 저장되고 업데이트되는 위치입니다. 일반적인 클라우드 리포지토리에는 GitHub 또는가 포함됩니다Bitbucket Cloud. 코드의 각 버전을 브랜치라고 합니다. 마이크로서비스 구조에서 각 리포지토리는 단일 기능 전용입니다. 단일 CI/CD 파이프라인은 여러 리포지토리를 사용할 수 있습니다.

콜드 캐시

비어 있거나, 제대로 채워지지 않았거나, 오래되었거나 관련 없는 데이터를 포함하는 버퍼 캐시입니다. 주 메모리나 디스크에서 데이터베이스 인스턴스를 읽어야 하기 때문에 성능에 영향을 미치며, 이는 버퍼 캐시에서 읽는 것보다 느립니다.

콜드 데이터

거의 액세스되지 않고 일반적으로 과거 데이터인 데이터. 이런 종류의 데이터를 쿼리할 때는 일반적으로 느린 쿼리가 허용됩니다. 이 데이터를 성능이 낮고 비용이 저렴한 스토리지 계층 또는 클래스로 옮기면 비용을 절감할 수 있습니다.

컴퓨터 비전(CV)

기계 학습을 사용하여 디지털 이미지 및 비디오와 같은 시각적 형식에서 정보를 분석하고 추출하는 AI 필드입니다. 예를 들어 Amazon SageMaker AI는 CV에 대한 이미지 처리 알고리즘을 제공합니다.

구성 드리프트

워크로드의 경우 구성이 예상 상태에서 변경됩니다. 이로 인해 워크로드가 규정을 준수하지 않을 수 있으며, 일반적으로 점진적이고 의도하지 않습니다.

구성 관리 데이터베이스(CMDB)

하드웨어 및 소프트웨어 구성 요소와 해당 구성을 포함하여 데이터베이스와 해당 IT 환경에 대한 정보를 저장하고 관리하는 리포지토리입니다. 일반적으로 마이그레이션의 포트폴리오 검색 및 분석 단계에서 CMDB의 데이터를 사용합니다.

규정 준수 팩

규정 준수 및 보안 검사를 사용자 지정하기 위해 조합할 수 있는 AWS Config 규칙 및 문제 해결 작업의 모음입니다. YAML 템플릿을 사용하여 적합성 팩을 AWS 계정 및 리전 또는 조직 전체에 단일 엔터티로 배포할 수 있습니다. 자세한 내용은 AWS Config 설명서의 적합성 팩을 참조하세요.

지속적 통합 및 지속적 전달(CI/CD)

소프트웨어 릴리스 프로세스의 소스, 빌드, 테스트, 스테이징 및 프로덕션 단계를 자동화하는 프로세스입니다. CI/CD는 일반적으로 파이프라인으로 설명됩니다. CI/CD를 통해 프로세스를 자동화하고, 생산성을 높이고, 코드 품질을 개선하고, 더 빠르게 제공할 수 있습니다. 자세한 내용은 지속적 전달의 이점을 참조하십시오. CD는 지속적 배포를 의미하기도 합니다. 자세한 내용은 지속적 전달(Continuous Delivery)과 지속적인 개발을 참조하십시오.

CV

컴퓨터 비전을 참조하세요.

D

저장 데이터

스토리지에 있는 데이터와 같이 네트워크에 고정되어 있는 데이터입니다.

데이터 분류

중요도와 민감도를 기준으로 네트워크의 데이터를 식별하고 분류하는 프로세스입니다. 이 프로세스는 데이터에 대한 적절한 보호 및 보존 제어를 결정하는 데 도움이 되므로 사이버 보안 위험 관리 전략의 중요한 구성 요소입니다. 데이터 분류는 AWS Well-Architected Framework에서 보안 원칙의 구성 요소입니다. 자세한 내용은 데이터 분류를 참조하십시오.

데이터 드리프트

프로덕션 데이터와 ML 모델 학습에 사용된 데이터 간의 상당한 차이 또는 시간 경과에 따른 입력 데이터의 의미 있는 변화. 데이터 드리프트는 ML 모델 예측의 전반적인 품질, 정확성 및 공정성을 저하시킬 수 있습니다.

전송 중 데이터

네트워크를 통과하고 있는 데이터입니다. 네트워크 리소스 사이를 이동 중인 데이터를 예로 들 수 있습니다.

데이터 메시

중앙 집중식 관리 및 거버넌스를 통해 분산되고 분산된 데이터 소유권을 제공하는 아키텍처 프레임워크입니다.

데이터 최소화

꼭 필요한 데이터만 수집하고 처리하는 원칙입니다. 에서 데이터를 최소화하면 개인 정보 보호 위험, 비용 및 분석 탄소 발자국을 줄일 AWS 클라우드 수 있습니다.

데이터 경계

신뢰할 수 있는 자격 증명만 예상 네트워크에서 신뢰할 수 있는 리소스에 액세스하도록 하는 데 도움이 되는 AWS 환경의 예방 가드레일 세트입니다. 자세한 내용은 데이터 경계 구축을 참조하세요 AWS.

데이터 사전 처리

원시 데이터를 ML 모델이 쉽게 구문 분석할 수 있는 형식으로 변환하는 것입니다. 데이터를 사전 처리한다는 것은 특정 열이나 행을 제거하고 누락된 값, 일관성이 없는 값 또는 중복 값을 처리함을 의미할 수 있습니다.

데이터 출처

라이프사이클 전반에 걸쳐 데이터의 출처와 기록을 추적하는 프로세스(예: 데이터 생성, 전송, 저장 방법).

데이터 주체

데이터를 수집 및 처리하는 개인입니다.

데이터 웨어하우스

분석과 같은 비즈니스 인텔리전스를 지원하는 데이터 관리 시스템입니다. 데이터 웨어하우스에는 일반적으로 많은 양의 기록 데이터가 포함되며 일반적으로 쿼리 및 분석에 사용됩니다.

데이터 정의 언어(DDL)

데이터베이스에서 테이블 및 객체의 구조를 만들거나 수정하기 위한 명령문 또는 명령입니다.

데이터베이스 조작 언어(DML)

데이터베이스에서 정보를 수정(삽입, 업데이트 및 삭제)하기 위한 명령문 또는 명령입니다.

DDL

데이터베이스 정의 언어를 참조하세요.

딥 앙상블

예측을 위해 여러 딥 러닝 모델을 결합하는 것입니다. 딥 앙상블을 사용하여 더 정확한 예측을 얻거나 예측의 불확실성을 추정할 수 있습니다.

딥 러닝

여러 계층의 인공 신경망을 사용하여 입력 데이터와 관심 대상 변수 간의 매핑을 식별하는 ML 하위 분야입니다.

심층 방어

네트워크와 그 안의 데이터 기밀성, 무결성 및 가용성을 보호하기 위해 컴퓨터 네트워크 전체에 일련의 보안 메커니즘과 제어를 신중하게 계층화하는 정보 보안 접근 방식입니다. 이 전략을 채택하면 AWS Organizations 구조의 여러 계층에 여러 컨트롤을 AWS추가하여 리소스를 보호할 수 있습니다. 예를 들어, 심층 방어 접근 방식은 다단계 인증, 네트워크 세분화 및 암호화를 결합할 수 있습니다.

위임된 관리자

에서 AWS Organizations호환되는 서비스는 AWS 멤버 계정을 등록하여 조직의 계정을 관리하고 해당 서비스에 대한 권한을 관리할 수 있습니다. 이러한 계정을 해당 서비스의 위임된 관리자라고 합니다. 자세한 내용과 호환되는 서비스 목록은 AWS Organizations 설명서의 AWS Organizations와 함께 사용할 수 있는 AWS 서비스를 참조하십시오.

배포

대상 환경에서 애플리케이션, 새 기능 또는 코드 수정 사항을 사용할 수 있도록 하는 프로세스입니다. 배포에는 코드 베이스의 변경 사항을 구현한 다음 애플리케이션 환경에서 해당 코드베이스를 구축하고 실행하는 작업이 포함됩니다.

개발 환경

환경을 참조하세요.

탐지 제어

이벤트 발생 후 탐지, 기록 및 알림을 수행하도록 설계된 보안 제어입니다. 이러한 제어는 기존의 예방적 제어를 우회한 보안 이벤트를 알리는 2차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Detective controls를 참조하십시오.

개발 가치 흐름 매핑 (DVSM)

소프트웨어 개발 라이프사이클에서 속도와 품질에 부정적인 영향을 미치는 제약 조건을 식별하고 우선 순위를 지정하는 데 사용되는 프로세스입니다. DVSM은 원래 린 제조 방식을 위해 설계된 가치 흐름 매핑 프로세스를 확장합니다. 소프트웨어 개발 프로세스를 통해 가치를 창출하고 이동하는 데 필요한 단계와 팀에 중점을 둡니다.

디지털 트윈

건물, 공장, 산업 장비 또는 생산 라인과 같은 실제 시스템을 가상으로 표현한 것입니다. 디지털 트윈은 예측 유지 보수, 원격 모니터링, 생산 최적화를 지원합니다.

차원 테이블

스타 스키마에서는 팩트 테이블의 정량적 데이터에 대한 데이터 속성을 포함하는 더 작은 테이블입니다. 차원 테이블 속성은 일반적으로 텍스트 필드 또는 텍스트처럼 동작하는 개별 숫자입니다. 이러한 속성은 일반적으로 쿼리 제약, 필터링 및 결과 집합 레이블 지정에 사용됩니다.

재해

워크로드 또는 시스템이 기본 배포 위치에서 비즈니스 목표를 달성하지 못하게 방해하는 이벤트입니다. 이러한 이벤트는 자연재해, 기술적 오류, 의도하지 않은 구성 오류 또는 멀웨어 공격과 같은 사람의 행동으로 인한 결과일 수 있습니다.

재해 복구(DR)

재해로 인한 가동 중지 시간과 데이터 손실을 최소화하는 데 사용하는 전략 및 프로세스입니다. 자세한 내용은 AWS Well-Architected Framework의 Disaster Recovery of Workloads on AWS: Recovery in the Cloud를 참조하세요.

DML

데이터베이스 조작 언어를 참조하세요.

도메인 기반 설계

구성 요소를 각 구성 요소가 제공하는 진화하는 도메인 또는 핵심 비즈니스 목표에 연결하여 복잡한 소프트웨어 시스템을 개발하는 접근 방식입니다. 이 개념은 에릭 에반스에 의해 그의 저서인 도메인 기반 디자인: 소프트웨어 중심의 복잡성 해결(Boston: Addison-Wesley Professional, 2003)에서 소개되었습니다. Strangler Fig 패턴과 함께 도메인 기반 설계를 사용하는 방법에 대한 자세한 내용은 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

DR

재해 복구를 참조하세요.

드리프트 감지

기준 구성과의 편차 추적. 예를 들어 AWS CloudFormation 를 사용하여 시스템 리소스의 드리프트를 감지하거나를 사용하여 AWS Control Tower 거버넌스 요구 사항 준수에 영향을 미칠 수 있는 랜딩 존의 변경 사항을 감지할 수 있습니다.

DVSM

개발 값 스트림 매핑을 참조하세요.

E

EDA

탐색 데이터 분석을 참조하세요.

EDI

전자 데이터 교환을 참조하세요.

엣지 컴퓨팅

IoT 네트워크의 엣지에서 스마트 디바이스의 컴퓨팅 성능을 개선하는 기술 클라우드 컴퓨팅과 비교할 때 엣지 컴퓨팅은 통신 지연 시간을 줄이고 응답 시간을 개선할 수 있습니다.

전자 데이터 교환(EDI)

조직 간의 비즈니스 문서 자동 교환. 자세한 내용은 전자 데이터 교환이란 무엇입니까?를 참조하세요.

암호화

사람이 읽을 수 있는 일반 텍스트 데이터를 사이퍼텍스트로 변환하는 컴퓨팅 프로세스입니다.

암호화 키

암호화 알고리즘에 의해 생성되는 무작위 비트의 암호화 문자열입니다. 키의 길이는 다양할 수 있으며 각 키는 예측할 수 없고 고유하게 설계되었습니다.

엔디안

컴퓨터 메모리에 바이트가 저장되는 순서입니다. 빅 엔디안 시스템은 가장 중요한 바이트를 먼저 저장합니다. 리틀 엔디안 시스템은 가장 덜 중요한 바이트를 먼저 저장합니다.

엔드포인트

서비스 엔드포인트를 참조하세요.

엔드포인트 서비스

Virtual Private Cloud(VPC)에서 호스팅하여 다른 사용자와 공유할 수 있는 서비스입니다. 를 사용하여 엔드포인트 서비스를 생성하고 다른 AWS 계정 또는 AWS Identity and Access Management (IAM) 보안 주체에 권한을 AWS PrivateLink 부여할 수 있습니다. 이러한 계정 또는 보안 주체는 인터페이스 VPC 엔드포인트를 생성하여 엔드포인트 서비스에 비공개로 연결할 수 있습니다. 자세한 내용은 Amazon Virtual Private Cloud(VPC) 설명서의 엔드포인트 서비스 생성을 참조하십시오.

엔터프라이즈 리소스 계획(ERP)

엔터프라이즈의 주요 비즈니스 프로세스(예: 회계, MES, 프로젝트 관리)를 자동화하고 관리하는 시스템입니다.

봉투 암호화

암호화 키를 다른 암호화 키로 암호화하는 프로세스입니다. 자세한 내용은 AWS Key Management Service (AWS KMS) 설명서의 봉투 암호화를 참조하세요.

환경

실행 중인 애플리케이션의 인스턴스입니다. 다음은 클라우드 컴퓨팅의 일반적인 환경 유형입니다.

- 개발 환경 - 애플리케이션 유지 관리를 담당하는 핵심 팀만 사용할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. 개발 환경은 변경 사항을 상위 환경으로 승격하기 전에 테스트하는 데 사용됩니다. 이러한 유형의 환경을 테스트 환경이라고도 합니다.
- 하위 환경 - 초기 빌드 및 테스트에 사용되는 환경을 비롯한 애플리케이션의 모든 개발 환경입니다.
- 프로덕션 환경 - 최종 사용자가 액세스할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. CI/CD 파이프라인에서 프로덕션 환경이 마지막 배포 환경입니다.
- 상위 환경 - 핵심 개발 팀 이외의 사용자가 액세스할 수 있는 모든 환경입니다. 프로덕션 환경, 프로덕션 이전 환경 및 사용자 수용 테스트를 위한 환경이 여기에 포함될 수 있습니다.

에픽

애자일 방법론에서 작업을 구성하고 우선순위를 정하는 데 도움이 되는 기능적 범주입니다. 에픽은 요구 사항 및 구현 작업에 대한 개괄적인 설명을 제공합니다. 예를 들어, AWS CAF 보안 에픽에는 ID 및 액세스 관리, 탐지 제어, 인프라 보안, 데이터 보호 및 인시던트 대응이 포함됩니다. AWS 마이그레이션 전략의 에픽에 대한 자세한 내용은 프로그램 구현 가이드를 참조하십시오.

ERP

엔터프라이즈 리소스 계획을 참조하세요.

탐색 데이터 분석(EDA)

데이터 세트를 분석하여 주요 특성을 파악하는 프로세스입니다. 데이터를 수집 또는 집계한 다음 초기 조사를 수행하여 패턴을 찾고, 이상을 탐지하고, 가정을 확인합니다. EDA는 요약 통계를 계산하고 데이터 시각화를 생성하여 수행됩니다.

F

팩트 테이블

별표 스키마의 중앙 테이블입니다. 비즈니스 운영에 대한 정량적 데이터를 저장합니다. 일반적으로 팩트 테이블에는 측정값이 포함된 열과 차원 테이블에 대한 외래 키가 포함된 열의 두 가지 유형이 포함됩니다.

빠른 실패

자주 증분 테스트를 사용하여 개발 수명 주기를 줄이는 철학입니다. 애자일 접근 방식의 중요한 부분입니다.

장애 격리 경계

에서 장애의 영향을 제한하고 워크로드의 복원력을 개선하는 데 도움이 되는 가용 영역, AWS 리전 컨트롤 플레인 또는 데이터 플레인과 같은 AWS 클라우드경계입니다. 자세한 내용은 AWS 장애 격리 경계를 참조하세요.

기능 브랜치

브랜치를 참조하세요.

기능

예측에 사용하는 입력 데이터입니다. 예를 들어, 제조 환경에서 기능은 제조 라인에서 주기적으로 캡처되는 이미지일 수 있습니다.

기능 중요도

모델의 예측에 특성이 얼마나 중요한지를 나타냅니다. 이는 일반적으로 SHAP(Shapley Additive Descriptions) 및 통합 그레디언트와 같은 다양한 기법을 통해 계산할 수 있는 수치 점수로 표현됩니다. 자세한 내용은 기계 학습 모델 해석 가능성을 참조하세요 AWS.

기능 변환

추가 소스로 데이터를 보강하거나, 값을 조정하거나, 단일 데이터 필드에서 여러 정보 세트를 추출하는 등 ML 프로세스를 위해 데이터를 최적화하는 것입니다. 이를 통해 ML 모델이 데이터를 활용

할 수 있습니다. 예를 들어, 날짜 '2021-05-27 00:15:37'을 '2021년', '5월', '목', '15일'로 분류하면 학습 알고리즘이 다양한 데이터 구성 요소와 관련된 미묘한 패턴을 학습하는 데 도움이 됩니다.

몇 장의 샷 프롬프트

유사한 작업을 수행하도록 요청하기 전에 LLM에 작업과 원하는 출력을 보여주는 몇 가지 예제를 제공합니다. 이 기법은 컨텍스트 내 학습을 적용하여 모델이 프롬프트에 포함된 예제(샷)에서 학습합니다. 퓨샷 프롬프트는 특정 형식 지정, 추론 또는 도메인 지식이 필요한 작업에 효과적일 수 있습니다. 제로샷 프롬프트도 참조하세요.

FGAC

세분화된 액세스 제어를 참조하세요.

세분화된 액세스 제어(FGAC)

여러 조건을 사용하여 액세스 요청을 허용하거나 거부합니다.

플래시컷 마이그레이션

단계적 접근 방식을 사용하는 대신 변경 데이터 캡처를 통해 연속 데이터 복제를 사용하여 최대한 짧은 시간 내에 데이터를 마이그레이션하는 데이터베이스 마이그레이션 방법입니다. 목표는 가동 중지 시간을 최소화하는 것입니다.

FM

파운데이션 모델을 참조하세요.

파운데이션 모델(FM)

일반화되고 레이블이 지정되지 않은 데이터의 대규모 데이터 세트에 대해 훈련된 대규모 딥 러닝 신경망입니다. FMs은 언어 이해, 텍스트 및 이미지 생성, 자연어 대화와 같은 다양한 일반 작업을 수행할 수 있습니다. 자세한 내용은 파운데이션 모델이란 무엇입니까?를 참조하세요.

G

생성형 AI

대량의 데이터에 대해 훈련되었으며 간단한 텍스트 프롬프트를 사용하여 이미지, 비디오, 텍스트 및 오디오와 같은 새 콘텐츠 및 아티팩트를 생성할 수 있는 AI 모델의 하위 집합입니다. 자세한 내용은 생성형 AI란 무엇입니까?를 참조하세요.

지리적 차단

지리적 제한을 참조하세요.

지리적 제한(지리적 차단)

Amazon CloudFront에서 특정 국가의 사용자가 콘텐츠 배포에 액세스하지 못하도록 하는 옵션입니다. 허용 목록 또는 차단 목록을 사용하여 승인된 국가와 차단된 국가를 지정할 수 있습니다. 자세한 내용은 CloudFront 설명서의 콘텐츠의 지리적 배포 제한을 참조하십시오.

Gitflow 워크플로

하위 환경과 상위 환경이 소스 코드 리포지토리의 서로 다른 브랜치를 사용하는 방식입니다. Gitflow 워크플로는 레거시로 간주되며 트렁크 기반 워크플로는 현대적이고 선호하는 접근 방식입니다.

골든 이미지

시스템 또는 소프트웨어의 새 인스턴스를 배포하기 위한 템플릿으로 사용되는 시스템 또는 소프트웨어의 스냅샷입니다. 예를 들어 제조업에서는 골든 이미지를 사용하여 여러 디바이스에 소프트웨어를 프로비저닝하고 디바이스 제조 작업의 속도, 확장성 및 생산성을 개선할 수 있습니다.

브라운필드 전략

새로운 환경에서 기존 인프라의 부재 시스템 아키텍처에 대한 그린필드 전략을 채택할 때 브라운필드라고도 하는 기존 인프라와의 호환성 제한 없이 모든 새로운 기술을 선택할 수 있습니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

가드레일

조직 단위(OU) 전체에서 리소스, 정책 및 규정 준수를 관리하는 데 도움이 되는 중요 규칙입니다. 예방 가드레일은 규정 준수 표준에 부합하도록 정책을 시행하며, 서비스 제어 정책과 IAM 권한 경계를 사용하여 구현됩니다. 탐지 가드레일은 정책 위반 및 규정 준수 문제를 감지하고 해결을 위한 알림을 생성하며, 이는 AWS Config,, Amazon GuardDuty AWS Security Hub, , AWS Trusted Advisor Amazon Inspector 및 사용자 지정 AWS Lambda 검사를 사용하여 구현됩니다.

H

HA

고가용성을 참조하세요.

이기종 데이터베이스 마이그레이션

다른 데이터베이스 엔진을 사용하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Oracle에서 Amazon Aurora로) 이기종 마이그레이션은 일반적으로 리아키텍트 작업의 일부이며 스

키마를 변환하는 것은 복잡한 작업일 수 있습니다.AWS 는 스키마 변환에 도움이 되는 AWS SCT를 제공합니다.

높은 가용성(HA)

문제나 재해 발생 시 개입 없이 지속적으로 운영할 수 있는 워크로드의 능력. HA 시스템은 자동으로 장애 조치되고, 지속적으로 고품질 성능을 제공하고, 성능에 미치는 영향을 최소화하면서 다양한 부하와 장애를 처리하도록 설계되었습니다.

히스토리언 현대화

제조 산업의 요구 사항을 더 잘 충족하도록 운영 기술(OT) 시스템을 현대화하고 업그레이드하는 데 사용되는 접근 방식입니다. 히스토리언은 공장의 다양한 출처에서 데이터를 수집하고 저장하는 데 사용되는 일종의 데이터베이스입니다.

홀드아웃 데이터

기계 학습 모델을 훈련하는 데 사용되는 데이터 세트에서 보류된 레이블이 지정된 기록 데이터의 일부입니다. 홀드아웃 데이터를 사용하여 모델 예측을 홀드아웃 데이터와 비교하여 모델 성능을 평가할 수 있습니다.

동종 데이터베이스 마이그레이션

동일한 데이터베이스 엔진을 공유하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Microsoft SQL Server에서 Amazon RDS for SQL Server로) 동종 마이그레이션은 일반적으로 리호스팅 또는 리플랫포밍 작업의 일부입니다. 네이티브 데이터베이스 유틸리티를 사용하여 스키마를 마이그레이션할 수 있습니다.

핫 데이터

자주 액세스하는 데이터(예: 실시간 데이터 또는 최근 번역 데이터). 일반적으로 이 데이터에는 빠른 쿼리 응답을 제공하기 위한 고성능 스토리지 계층 또는 클래스가 필요합니다.

핫픽스

프로덕션 환경의 중요한 문제를 해결하기 위한 긴급 수정입니다. 핫픽스는 긴급하기 때문에 일반적인 DevOps 릴리스 워크플로 외부에서 실행됩니다.

하이퍼케어 기간

전환 직후 마이그레이션 팀이 문제를 해결하기 위해 클라우드에서 마이그레이션된 애플리케이션을 관리하고 모니터링하는 기간입니다. 일반적으로 이 기간은 1~4일입니다. 하이퍼케어 기간이 끝나면 마이그레이션 팀은 일반적으로 애플리케이션에 대한 책임을 클라우드 운영 팀에 넘깁니다.

정보

IaC

코드형 인프라를 참조하세요.

자격 증명 기반 정책

AWS 클라우드 환경 내에서 권한을 정의하는 하나 이상의 IAM 보안 주체에 연결된 정책입니다.

유휴 애플리케이션

90일 동안 평균 CPU 및 메모리 사용량이 5~20%인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하거나 온프레미스에 유지하는 것이 일반적입니다.

IIoT

산업용 사물 인터넷을 참조하십시오.

변경 불가능한 인프라

기존 인프라를 업데이트, 패치 또는 수정하는 대신 프로덕션 워크로드를 위한 새 인프라를 배포하는 모델입니다. 변경 불가능한 인프라는 변경 가능한 인프라보다 본질적으로 더 일관되고 안정적이며 예측 가능합니다. 자세한 내용은 AWS Well-Architected Framework의 변경 불가능한 인프라를 사용한 배포 모범 사례를 참조하세요.

인바운드(수신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 외부에서 네트워크 연결을 수락, 검사 및 라우팅하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

증분 마이그레이션

한 번에 전체 전환을 수행하는 대신 애플리케이션을 조금씩 마이그레이션하는 전환 전략입니다. 예를 들어, 처음에는 소수의 마이크로서비스나 사용자만 새 시스템으로 이동할 수 있습니다. 모든 것이 제대로 작동하는지 확인한 후에는 레거시 시스템을 폐기할 수 있을 때까지 추가 마이크로서비스 또는 사용자를 점진적으로 이동할 수 있습니다. 이 전략을 사용하면 대규모 마이그레이션과 관련된 위험을 줄일 수 있습니다.

Industry 4.0

연결성, 실시간 데이터, 자동화, 분석 및 AI/ML의 발전을 통한 제조 프로세스의 현대화를 참조하기 위해 2016년에 Klaus Schwab에서 도입한 용어입니다.

인프라

애플리케이션의 환경 내에 포함된 모든 리소스와 자산입니다.

코드형 인프라(IaC)

구성 파일 세트를 통해 애플리케이션의 인프라를 프로비저닝하고 관리하는 프로세스입니다. IaC는 새로운 환경의 반복 가능성, 신뢰성 및 일관성을 위해 인프라 관리를 중앙 집중화하고, 리소스를 표준화하고, 빠르게 확장할 수 있도록 설계되었습니다.

산업용 사물 인터넷(IIoT)

제조, 에너지, 자동차, 의료, 생명과학, 농업 등의 산업 부문에서 인터넷에 연결된 센서 및 디바이스의 사용 자세한 내용은 산업용 사물 인터넷(IoT) 디지털 트랜스포메이션 전략 구축을 참조하십시오.

검사 VPC

AWS 다중 계정 아키텍처에서는 VPC(동일하거나 다른 AWS 리전), 인터넷 및 온프레미스 네트워크 간의 네트워크 트래픽 검사를 관리하는 중앙 집중식 VPCs. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

사물 인터넷(IoT)

인터넷이나 로컬 통신 네트워크를 통해 다른 디바이스 및 시스템과 통신하는 센서 또는 프로세서가 내장된 연결된 물리적 객체의 네트워크 자세한 내용은 IoT란?을 참조하십시오.

해석력

모델의 예측이 입력에 따라 어떻게 달라지는지를 사람이 이해할 수 있는 정도를 설명하는 기계 학습 모델의 특성입니다. 자세한 내용은 기계 학습 모델 해석 가능성을 참조하세요 AWS.

IoT

사물 인터넷을 참조하세요.

IT 정보 라이브러리(TIL)

IT 서비스를 제공하고 이러한 서비스를 비즈니스 요구 사항에 맞게 조정하기 위한 일련의 모범 사례 ITIL은 ITSM의 기반을 제공합니다.

IT 서비스 관리(TSM)

조직의 IT 서비스 설계, 구현, 관리 및 지원과 관련된 활동 클라우드 운영을 ITSM 도구와 통합하는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

ITIL

IT 정보 라이브러리를 참조하세요.

ITSM

IT 서비스 관리를 참조하세요.

L

레이블 기반 액세스 제어(LBAC)

사용자 및 데이터 자체에 각각 보안 레이블 값을 명시적으로 할당하는 필수 액세스 제어(MAC)를 구현한 것입니다. 사용자 보안 레이블과 데이터 보안 레이블 간의 교차 부분에 따라 사용자가 볼 수 있는 행과 열이 결정됩니다.

랜딩 존

랜딩 존은 확장 가능하고 안전한 잘 설계된 다중 계정 AWS 환경입니다. 조직은 여기에서부터 보안 및 인프라 환경에 대한 확신을 가지고 워크로드와 애플리케이션을 신속하게 시작하고 배포할 수 있습니다. 랜딩 존에 대한 자세한 내용은 안전하고 확장 가능한 다중 계정 AWS 환경 설정을 참조하십시오.

대규모 언어 모델(LLM)

방대한 양의 데이터를 기반으로 사전 훈련된 딥 러닝 AI 모델입니다. LLM은 질문 답변, 문서 요약, 텍스트를 다른 언어로 변환, 문장 완성과 같은 여러 작업을 수행할 수 있습니다. 자세한 내용은 LLMs 참조하십시오.

대규모 마이그레이션

300대 이상의 서버 마이그레이션입니다.

LBAC

레이블 기반 액세스 제어를 참조하세요.

최소 권한

작업을 수행하는 데 필요한 최소 권한을 부여하는 보안 모범 사례입니다. 자세한 내용은 IAM 설명서의 최소 권한 적용을 참조하십시오.

리프트 앤드 시프트

7R을 참조하세요.

리틀 엔디안 시스템

가장 덜 중요한 바이트를 먼저 저장하는 시스템입니다. Endianness도 참조하세요.

LLM

대규모 언어 모델을 참조하세요.

하위 환경

환경을 참조하세요.

M

기계 학습(ML)

패턴 인식 및 학습에 알고리즘과 기법을 사용하는 인공 지능의 한 유형입니다. ML은 사물 인터넷(IoT) 데이터와 같은 기록된 데이터를 분석하고 학습하여 패턴을 기반으로 통계 모델을 생성합니다. 자세한 내용은 기계 학습을 참조하십시오.

기본 브랜치

브랜치를 참조하세요.

맬웨어

컴퓨터 보안 또는 개인 정보 보호를 손상하도록 설계된 소프트웨어입니다. 맬웨어는 컴퓨터 시스템을 방해하거나, 민감한 정보를 유출하거나, 무단 액세스를 가져올 수 있습니다. 맬웨어의 예로는 바이러스, 웜, 랜섬웨어, 트로이 목마, 스파이웨어, 키로거 등이 있습니다.

관리형 서비스

AWS 서비스 는 인프라 계층, 운영 체제 및 플랫폼을 AWS 운영하고, 사용자는 엔드포인트에 액세스하여 데이터를 저장하고 검색합니다. Amazon Simple Storage Service(Amazon S3) 및 Amazon DynamoDB는 관리형 서비스의 예입니다. 이를 추상화된 서비스라고도 합니다.

제조 실행 시스템(MES)

원재료를 생산 현장의 완성된 제품으로 변환하는 생산 프로세스를 추적, 모니터링, 문서화 및 제어하기 위한 소프트웨어 시스템입니다.

MAP

마이그레이션 가속화 프로그램을 참조하세요.

메커니즘

도구를 생성하고 도구 채택을 유도한 다음 결과를 검사하여 조정하는 전체 프로세스입니다. 메커니즘은 작동 시 자체를 강화하고 개선하는 주기입니다. 자세한 내용은 AWS Well-Architected Framework의 빌드 메커니즘을 참조하세요.

멤버 계정

조직의 일부인 관리 계정을 AWS 계정 제외한 모든 계정. AWS Organizations하나의 계정은 한 번에 하나의 조직 멤버만 될 수 있습니다.

MES

제조 실행 시스템을 참조하세요.

메시지 대기열 원격 측정 전송(MQTT)

리소스가 제한된 IoT 디바이스에 대한 게시/구독 패턴을 기반으로 하는 경량 M2M(machine-to-machine) 통신 프로토콜입니다.

마이크로서비스

잘 정의된 API를 통해 통신하고 일반적으로 소규모 자체 팀이 소유하는 소규모 독립 서비스입니다. 예를 들어, 보험 시스템에는 영업, 마케팅 등의 비즈니스 역량이나 구매, 청구, 분석 등의 하위 영역에 매핑되는 마이크로 서비스가 포함될 수 있습니다. 마이크로서비스의 이점으로 민첩성, 유연한 확장, 손쉬운 배포, 재사용 가능한 코드, 복원력 등이 있습니다. 자세한 내용은 AWS 서버리스 서비스를 사용하여 마이크로서비스 통합을 참조하세요.

마이크로서비스 아키텍처

각 애플리케이션 프로세스를 마이크로서비스로 실행하는 독립 구성 요소를 사용하여 애플리케이션을 구축하는 접근 방식입니다. 이러한 마이크로서비스는 경량 API를 사용하여 잘 정의된 인터페이스를 통해 통신합니다. 애플리케이션의 특정 기능에 대한 수요에 맞게 이 아키텍처의 각 마이크로서비스를 업데이트, 배포 및 조정할 수 있습니다. 자세한 내용은 에서 마이크로서비스 구현을 참조하세요 AWS.

Migration Acceleration Program(MAP)

조직이 클라우드로 전환하기 위한 강력한 운영 기반을 구축하고 초기 마이그레이션 비용을 상쇄하는 데 도움이 되는 컨설팅 지원, 교육 및 서비스를 제공하는 AWS 프로그램입니다. MAP에는 레거시 마이그레이션을 체계적인 방식으로 실행하기 위한 마이그레이션 방법론과 일반적인 마이그레이션 시나리오를 자동화하고 가속화하는 도구 세트가 포함되어 있습니다.

대규모 마이그레이션

애플리케이션 포트폴리오의 대다수를 웨이브를 통해 클라우드로 이동하는 프로세스로, 각 웨이브에서 더 많은 애플리케이션이 더 빠른 속도로 이동합니다. 이 단계에서는 이전 단계에서 배운 모범 사례와 교훈을 사용하여 팀, 도구 및 프로세스의 마이그레이션 팩토리를 구현하여 자동화 및 민첩한 제공을 통해 워크로드 마이그레이션을 간소화합니다. 이것은 AWS 마이그레이션 전략의 세 번째 단계입니다.

마이그레이션 팩토리

자동화되고 민첩한 접근 방식을 통해 워크로드 마이그레이션을 간소화하는 다기능 팀입니다. 마이그레이션 팩토리 팀에는 일반적으로 스프린트에서 일하는 운영, 비즈니스 분석가 및 소유자, 마이그레이션 엔지니어, 개발자, DevOps 전문가가 포함됩니다. 엔터프라이즈 애플리케이션 포트폴리오의 20~50%는 공장 접근 방식으로 최적화할 수 있는 반복되는 패턴으로 구성되어 있습니다. 자세한 내용은 이 콘텐츠 세트의 클라우드 마이그레이션 팩토리 가이드와 마이그레이션 팩토리에 대한 설명을 참조하십시오.

마이그레이션 메타데이터

마이그레이션을 완료하는 데 필요한 애플리케이션 및 서버에 대한 정보 각 마이그레이션 패턴에는 서로 다른 마이그레이션 메타데이터 세트가 필요합니다. 마이그레이션 메타데이터의 예로는 대상 서브넷, 보안 그룹 및 AWS 계정이 있습니다.

마이그레이션 패턴

사용되는 마이그레이션 전략, 마이그레이션 대상, 마이그레이션 애플리케이션 또는 서비스를 자세히 설명하는 반복 가능한 마이그레이션 작업입니다. 예: AWS Application Migration Service를 사용하여 Amazon EC2로 마이그레이션을 리호스팅합니다.

Migration Portfolio Assessment(MPA)

로 마이그레이션하기 위한 비즈니스 사례를 검증하기 위한 정보를 제공하는 온라인 도구입니다 AWS 클라우드. MPA는 상세한 포트폴리오 평가(서버 적정 규모 조정, 가격 책정, TCO 비교, 마이그레이션 비용 분석)와 마이그레이션 계획(애플리케이션 데이터 분석 및 데이터 수집, 애플리케이션 그룹화, 마이그레이션 우선순위 지정, 웨이브 계획)을 제공합니다. MPA 도구(로그인 필요)는 모든 AWS 컨설턴트와 APN 파트너 컨설턴트가 무료로 사용할 수 있습니다.

마이그레이션 준비 상태 평가(MRA)

AWS CAF를 사용하여 조직의 클라우드 준비 상태에 대한 인사이트를 얻고, 강점과 약점을 식별하고, 식별된 격차를 해소하기 위한 실행 계획을 수립하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오. MRA는 AWS 마이그레이션 전략의 첫 번째 단계입니다.

마이그레이션 전략

워크로드를 로 마이그레이션하는 데 사용되는 접근 방식입니다 AWS 클라우드. 자세한 내용은이 용어집의 7R 항목을 참조하고 대규모 마이그레이션을 가속화하기 위해 조직 동원을 참조하세요.

ML

기계 학습을 참조하세요.

현대화

비용을 절감하고 효율성을 높이고 혁신을 활용하기 위해 구식(레거시 또는 모놀리식) 애플리케이션과 해당 인프라를 클라우드의 민첩하고 탄력적이고 가용성이 높은 시스템으로 전환하는 것입니다. 자세한 내용은 의 애플리케이션 현대화 전략을 참조하세요 AWS 클라우드.

현대화 준비 상태 평가

조직 애플리케이션의 현대화 준비 상태를 파악하고, 이점, 위험 및 종속성을 식별하고, 조직이 해당 애플리케이션의 향후 상태를 얼마나 잘 지원할 수 있는지를 확인하는 데 도움이 되는 평가입니다. 평가 결과는 대상 아키텍처의 청사진, 현대화 프로세스의 개발 단계와 마일스톤을 자세히 설명하는 로드맵 및 파악된 격차를 해소하기 위한 실행 계획입니다. 자세한 내용은 의 애플리케이션에 대한 현대화 준비 상태 평가를 참조하세요 AWS 클라우드.

모놀리식 애플리케이션(모놀리식 유형)

긴밀하게 연결된 프로세스를 사용하여 단일 서비스로 실행되는 애플리케이션입니다. 모놀리식 애플리케이션에는 몇 가지 단점이 있습니다. 한 애플리케이션 기능에 대한 수요가 급증하면 전체 아키텍처 규모를 조정해야 합니다. 코드 베이스가 커지면 모놀리식 애플리케이션의 기능을 추가하거나 개선하는 것도 더 복잡해집니다. 이러한 문제를 해결하기 위해 마이크로서비스 아키텍처를 사용할 수 있습니다. 자세한 내용은 마이크로서비스로 모놀리식 유형 분해를 참조하십시오.

MPA

마이그레이션 포트폴리오 평가를 참조하세요.

MQTT

메시지 대기열 원격 측정 전송을 참조하세요.

멀티클래스 분류

여러 클래스에 대한 예측(2개 이상의 결과 중 하나 예측)을 생성하는 데 도움이 되는 프로세스입니다. 예를 들어, ML 모델이 '이 제품은 책인가요, 자동차인가요, 휴대폰인가요?' 또는 '이 고객이 가장 관심을 갖는 제품 범주는 무엇인가요?'라고 물을 수 있습니다.

변경 가능한 인프라

프로덕션 워크로드를 위해 기존 인프라를 업데이트하고 수정하는 모델입니다. 일관성, 신뢰성 및 예측 가능성을 높이기 위해 AWS Well-Architected Framework에서는 변경 불가능한 인프라를 모범 사례로 사용할 것을 권장합니다.

O

OAC

오리진 액세스 제어를 참조하세요.

OAI

오리진 액세스 ID를 참조하세요.

OCM

조직 변경 관리를 참조하세요.

오프라인 마이그레이션

마이그레이션 프로세스 중 소스 워크로드가 중단되는 마이그레이션 방법입니다. 이 방법은 가동 중지 증가를 수반하며 일반적으로 작고 중요하지 않은 워크로드에 사용됩니다.

OI

작업 통합을 참조하세요.

OLA

운영 수준 계약을 참조하세요.

온라인 마이그레이션

소스 워크로드를 오프라인 상태로 전환하지 않고 대상 시스템에 복사하는 마이그레이션 방법입니다. 워크로드에 연결된 애플리케이션은 마이그레이션 중에도 계속 작동할 수 있습니다. 이 방법은 가동 중지 차단 또는 최소화를 수반하며 일반적으로 중요한 프로덕션 워크로드에 사용됩니다.

OPC-UA

Open Process Communications - Unified Architecture를 참조하세요.

Open Process Communications - 통합 아키텍처(OPC-UA)

산업 자동화를 위한 M2M(Machinemachine-to-machine) 통신 프로토콜입니다. OPC-UA는 데이터 암호화, 인증 및 권한 부여 체계와 상호 운용성 표준을 제공합니다.

운영 수준 협약(OLA)

서비스 수준에 관한 계약(SLA)을 지원하기 위해 직무 IT 그룹이 서로에게 제공하기로 약속한 내용을 명확히 하는 계약입니다.

운영 준비 상태 검토(ORR)

인시던트 및 가능한 장애의 범위를 이해, 평가, 예방 또는 줄이는 데 도움이 되는 질문 체크리스트 및 관련 모범 사례입니다. 자세한 내용은 AWS Well-Architected Framework의 운영 준비 검토(ORR)를 참조하세요.

운영 기술(OT)

물리적 환경과 함께 작동하여 산업 운영, 장비 및 인프라를 제어하는 하드웨어 및 소프트웨어 시스템입니다. 제조에서 OT 및 정보 기술(IT) 시스템의 통합은 Industry 4.0 혁신의 핵심 초점입니다.

운영 통합(OI)

클라우드에서 운영을 현대화하는 프로세스로 준비 계획, 자동화 및 통합을 수반합니다. 자세한 내용은 운영 통합 가이드를 참조하십시오.

조직 트레일

조직 내 모든에 대한 모든 이벤트를 로깅 AWS CloudTrail 하는 AWS 계정 에서 생성된 추적입니다 AWS Organizations. 이 트레일은 조직에 속한 각 AWS 계정 에 생성되고 각 계정의 활동을 추적합니다. 자세한 내용은 CloudTrail 설명서의 Creating a trail for an organization을 참조하십시오.

조직 변경 관리(OCM)

사람, 문화 및 리더십 관점에서 중대하고 파괴적인 비즈니스 혁신을 관리하기 위한 프레임워크입니다. OCM은 변화 채택을 가속화하고, 과도기적 문제를 해결하고, 문화 및 조직적 변화를 주도함으로써 조직이 새로운 시스템 및 전략을 준비하고 전환할 수 있도록 지원합니다. AWS 마이그레이션 전략에서는 클라우드 채택 프로젝트에 필요한 변경 속도 때문에이 프레임워크를 인력 가속화라고 합니다. 자세한 내용은 사용 가이드를 참조하십시오.

오리진 액세스 제어(OAC)

CloudFront에서 Amazon Simple Storage Service(S3) 콘텐츠를 보호하기 위해 액세스를 제한하는 고급 옵션입니다. OAC는 AWS KMS (SSE-KMS)를 사용한 모든 서버 측 암호화 AWS 리전와 S3 버킷에 대한 동적 PUT 및 DELETE 요청에서 모든 S3 버킷을 지원합니다.

오리진 액세스 ID(OAI)

CloudFront에서 Amazon S3 콘텐츠를 보호하기 위해 액세스를 제한하는 옵션입니다. OAI를 사용하면 CloudFront는 Amazon S3가 인증할 수 있는 보안 주체를 생성합니다. 인증된 보안 주체는 특

정 CloudFront 배포를 통해서만 S3 버킷의 콘텐츠에 액세스할 수 있습니다. 더 세분화되고 향상된 액세스 제어를 제공하는 OAC도 참조하십시오.

ORR

운영 준비 상태 검토를 참조하세요.

OT

운영 기술을 참조하세요.

아웃바운드(송신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 내에서 시작된 네트워크 연결을 처리하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

P

권한 경계

사용자나 역할이 가질 수 있는 최대 권한을 설정하기 위해 IAM 보안 주체에 연결되는 IAM 관리 정책입니다. 자세한 내용은 IAM 설명서의 권한 경계를 참조하십시오.

개인 식별 정보(PII)

직접 보거나 다른 관련 데이터와 함께 짝을 지을 때 개인의 신원을 합리적으로 추론하는 데 사용할 수 있는 정보입니다. PII의 예로는 이름, 주소, 연락처 정보 등이 있습니다.

PII

개인 식별 정보를 참조하세요.

플레이북

클라우드에서 핵심 운영 기능을 제공하는 등 마이그레이션과 관련된 작업을 캡처하는 일련의 사전 정의된 단계입니다. 플레이북은 스크립트, 자동화된 런북 또는 현대화된 환경을 운영하는 데 필요한 프로세스나 단계 요약의 형태를 취할 수 있습니다.

PLC

프로그래밍 가능한 로직 컨트롤러를 참조하세요.

PLM

제품 수명 주기 관리를 참조하세요.

정책

권한을 정의하거나(자격 증명 기반 정책 참조), 액세스 조건을 지정하거나(리소스 기반 정책 참조), 조직의 모든 계정에 대한 최대 권한을 정의할 수 있는 객체 AWS Organizations 입니다(서비스 제어 정책 참조).

다국어 지속성

데이터 액세스 패턴 및 기타 요구 사항을 기반으로 독립적으로 마이크로서비스의 데이터 스토리지 기술 선택. 마이크로서비스가 동일한 데이터 스토리지 기술을 사용하는 경우 구현 문제가 발생하거나 성능이 저하될 수 있습니다. 요구 사항에 가장 적합한 데이터 스토어를 사용하면 마이크로서비스를 더 쉽게 구현하고 성능과 확장성을 높일 수 있습니다. 자세한 내용은 마이크로서비스에서 데이터 지속성 활성화를 참조하십시오.

포트폴리오 평가

마이그레이션을 계획하기 위해 애플리케이션 포트폴리오를 검색 및 분석하고 우선순위를 정하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 상태 평가를 참조하십시오.

조건자

`WHERE` 절에서 `false`일반적으로 위치한 `true` 또는를 반환하는 쿼리 조건입니다.

조건자 푸시다운

전송 전에 쿼리의 데이터를 필터링하는 데이터베이스 쿼리 최적화 기법입니다. 이렇게 하면 관계형 데이터베이스에서 검색하고 처리해야 하는 데이터의 양이 줄어들고 쿼리 성능이 향상됩니다.

예방적 제어

이벤트 발생을 방지하도록 설계된 보안 제어입니다. 이 제어는 네트워크에 대한 무단 액세스나 원치 않는 변경을 방지하는 데 도움이 되는 1차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Preventative controls를 참조하십시오.

보안 주체

작업을 수행하고 리소스에 액세스할 수 AWS 있는의 엔터티입니다. 이 엔터티는 일반적으로 , AWS 계정 IAM 역할 또는 사용자의 루트 사용자입니다. 자세한 내용은 IAM 설명서의 역할 용어 및 개념의 보안 주체를 참조하십시오.

설계에 따른 개인 정보 보호

전체 개발 프로세스를 통해 개인 정보를 고려하는 시스템 엔지니어링 접근 방식입니다.

프라이빗 호스팅 영역

Amazon Route 53에서 하나 이상의 VPC 내 도메인과 하위 도메인에 대한 DNS 쿼리에 응답하는 방법에 대한 정보가 담긴 컨테이너입니다. 자세한 내용은 Route 53 설명서의 프라이빗 호스팅 영역 작업을 참조하십시오.

사전 예방적 제어

규정 미준수 리소스의 배포를 방지하도록 설계된 보안 제어입니다. 이러한 제어는 리소스가 프로비저닝되기 전에 리소스를 스캔합니다. 리소스가 컨트롤을 준수하지 않으면 프로비저닝되지 않습니다. 자세한 내용은 AWS Control Tower 설명서의 제어 참조 가이드를 참조하고 보안 제어 구현의 사전 예방적 제어를 참조하세요. AWS

제품 수명 주기 관리(PLM)

설계, 개발 및 출시부터 성장 및 성숙도, 거부 및 제거에 이르기까지 전체 수명 주기 동안 제품의 데이터 및 프로세스 관리.

프로덕션 환경

환경을 참조하세요.

프로그래밍 가능한 로직 컨트롤러(PLC)

제조에서 기계를 모니터링하고 제조 프로세스를 자동화하는 매우 안정적이고 적응력이 뛰어난 컴퓨터입니다.

프롬프트 체인

한 LLM 프롬프트의 출력을 다음 프롬프트의 입력으로 사용하여 더 나은 응답을 생성합니다. 이 기법은 복잡한 작업을 하위 작업으로 나누거나 예비 응답을 반복적으로 구체화하거나 확장하는 데 사용됩니다. 이를 통해 모델 응답의 정확성과 관련성을 개선하고 보다 세분화되고 개인화된 결과를 얻을 수 있습니다.

가명화

데이터세트의 개인 식별자를 자리 표시자 값으로 바꾸는 프로세스입니다. 가명화는 개인 정보를 보호하는 데 도움이 될 수 있습니다. 가명화된 데이터는 여전히 개인 데이터로 간주됩니다.

게시/구독(pub/sub)

마이크로서비스 간의 비동기 통신을 지원하여 확장성과 응답성을 개선하는 패턴입니다. 예를 들어 마이크로서비스 기반 MES에서 마이크로서비스는 다른 마이크로서비스가 구독할 수 있는 채널에 이벤트 메시지를 게시할 수 있습니다. 시스템은 게시 서비스를 변경하지 않고도 새 마이크로서비스를 추가할 수 있습니다.

Q

쿼리 계획

SQL 관계형 데이터베이스 시스템의 데이터에 액세스하는 데 사용되는 지침과 같은 일련의 단계입니다.

쿼리 계획 회귀

데이터베이스 서비스 최적화 프로그램이 데이터베이스 환경을 변경하기 전보다 덜 최적의 계획을 선택하는 경우입니다. 통계, 제한 사항, 환경 설정, 쿼리 파라미터 바인딩 및 데이터베이스 엔진 업데이트의 변경으로 인해 발생할 수 있습니다.

R

RACI 매트릭스

책임, 책임, 상담, 정보 제공(RACI)을 참조하세요.

RAG

Retrieval Augmented Generation을 참조하세요.

랜섬웨어

결제가 완료될 때까지 컴퓨터 시스템이나 데이터에 대한 액세스를 차단하도록 설계된 악성 소프트웨어입니다.

RASCI 매트릭스

책임, 책임, 상담, 정보 제공(RACI)을 참조하세요.

RCAC

행 및 열 액세스 제어를 참조하세요.

읽기 전용 복제본

읽기 전용 용도로 사용되는 데이터베이스의 사본입니다. 쿼리를 읽기 전용 복제본으로 라우팅하여 기본 데이터베이스의 로드를 줄일 수 있습니다.

재설계

7R을 참조하세요.

Recovery Point Objective(RPO)

마지막 데이터 복구 시점 이후 허용되는 최대 시간입니다. 이에 따라 마지막 복구 시점과 서비스 중단 사이에 허용되는 데이터 손실로 간주되는 범위가 결정됩니다.

Recovery Time Objective(RTO)

서비스 중단과 서비스 복원 사이의 허용 가능한 지연 시간입니다.

리팩터링

7R을 참조하세요.

리전

지리적 영역의 AWS 리소스 모음입니다. 각 AWS 리전 는 내결함성, 안정성 및 복원력을 제공하기 위해 서로 격리되고 독립적입니다. 자세한 내용은 계정에서 사용할 수 있는 지정을 참조 AWS 리전 하세요.

회귀

숫자 값을 예측하는 ML 기법입니다. 예를 들어, '이 집은 얼마에 팔릴까?'라는 문제를 풀기 위해 ML 모델은 선형 회귀 모델을 사용하여 주택에 대해 알려진 사실(예: 면적)을 기반으로 주택의 매매 가격을 예측할 수 있습니다.

리호스팅

7R을 참조하세요.

release

배포 프로세스에서 변경 사항을 프로덕션 환경으로 승격시키는 행위입니다.

재배치

7R을 참조하세요.

리플랫포밍

7R을 참조하세요.

재구매

7R을 참조하세요.

복원력

중단에 저항하거나 복구할 수 있는 애플리케이션의 기능입니다. 에서 복원력을 계획할 때 고가용성 및 재해 복구가 일반적인 고려 사항입니다 AWS 클라우드. 자세한 내용은 AWS 클라우드 복원력을 참조하세요.

리소스 기반 정책

Amazon S3 버킷, 엔드포인트, 암호화 키 등의 리소스에 연결된 정책입니다. 이 유형의 정책은 액세스가 허용된 보안 주체, 지원되는 작업 및 충족해야 하는 기타 조건을 지정합니다.

RACI(Responsible, Accountable, Consulted, Informed) 매트릭스

마이그레이션 활동 및 클라우드 운영에 참여하는 모든 당사자의 역할과 책임을 정의하는 매트릭스입니다. 매트릭스 이름은 매트릭스에 정의된 책임 유형에서 파생됩니다. 실무 담당자 (R), 의사 결정권자 (A), 업무 수행 조언자 (C), 결과 통보 대상자 (I). 지원자는 (S) 선택사항입니다. 지원자를 포함하면 매트릭스를 RASCI 매트릭스라고 하고, 지원자를 제외하면 RACI 매트릭스라고 합니다.

대응 제어

보안 기준에서 벗어나거나 부정적인 이벤트를 해결하도록 설계된 보안 제어입니다. 자세한 내용은 Implementing security controls on AWS의 Responsive controls를 참조하십시오.

retain

7R을 참조하세요.

사용 중지

7R을 참조하세요.

검색 증강 세대(RAG)

응답을 생성하기 전에 LLM이 훈련 데이터 소스 외부에 있는 신뢰할 수 있는 데이터 소스를 참조하는 생성형 AI 기술입니다. 예를 들어 RAG 모델은 조직의 지식 기반 또는 사용자 지정 데이터에 대한 의미 검색을 수행할 수 있습니다. 자세한 내용은 RAG란 무엇입니까?를 참조하십시오.

교체

공격자가 보안 인증 정보에 액세스하는 것을 더 어렵게 만들기 위해 보안 암호를 주기적으로 업데이트하는 프로세스입니다.

행 및 열 액세스 제어(RCAC)

액세스 규칙이 정의된 기본적이고 유연한 SQL 표현식을 사용합니다. RCAC는 행 권한과 열 마스크로 구성됩니다.

RPO

복구 시점 목표를 참조하세요.

RTO

복구 시간 목표를 참조하세요.

런북

특정 작업을 수행하는 데 필요한 일련의 수동 또는 자동 절차입니다. 일반적으로 오류율이 높은 반복 작업이나 절차를 간소화하기 위해 런북을 만듭니다.

S

SAML 2.0

많은 ID 제공업체(idP)에서 사용하는 개방형 표준입니다. 이 기능을 사용하면 연동 SSO(Single Sign-On)를 AWS Management Console 사용할 수 있으므로 사용자는 조직 내 모든 사용자에 대해 IAM에서 사용자를 생성하지 않고도에 로그인하거나 AWS API 작업을 호출할 수 있습니다. SAML 2.0 기반 페더레이션에 대한 자세한 내용은 IAM 설명서의 SAML 2.0 기반 페더레이션 정보를 참조하십시오.

SCADA

감독 제어 및 데이터 획득을 참조하세요.

SCP

서비스 제어 정책을 참조하세요.

secret

에는 암호 또는 사용자 자격 증명과 같이 암호화된 형식으로 저장하는 AWS Secrets Manager기밀 또는 제한된 정보가 있습니다. 보안 암호 값과 메타데이터로 구성됩니다. 보안 암호 값은 바이너리, 단일 문자열 또는 여러 문자열일 수 있습니다. 자세한 내용은 Secrets Manager 설명서의 Secrets Manager 보안 암호에 무엇이 있습니까?를 참조하세요.

설계별 보안

전체 개발 프로세스를 통해 보안을 고려하는 시스템 엔지니어링 접근 방식입니다.

보안 제어

위협 행위자가 보안 취약성을 악용하는 능력을 방지, 탐지 또는 감소시키는 기술적 또는 관리적 가드레일입니다. 보안 제어에는 예방, 탐지, 대응 및 사전 예방의 네 가지 주요 유형이 있습니다.

보안 강화

공격 표면을 줄여 공격에 대한 저항력을 높이는 프로세스입니다. 더 이상 필요하지 않은 리소스 제거, 최소 권한 부여의 보안 모범 사례 구현, 구성 파일의 불필요한 기능 비활성화 등의 작업이 여기에 포함될 수 있습니다.

보안 정보 및 이벤트 관리(SIEM) 시스템

보안 정보 관리(SIM)와 보안 이벤트 관리(SEM) 시스템을 결합하는 도구 및 서비스입니다. SIEM 시스템은 서버, 네트워크, 디바이스 및 기타 소스에서 데이터를 수집, 모니터링 및 분석하여 위협과 보안 침해를 탐지하고 알림을 생성합니다.

보안 응답 자동화

보안 이벤트에 자동으로 응답하거나 해결하도록 설계된 사전 정의되고 프로그래밍된 작업입니다. 이러한 자동화는 보안 모범 사례를 구현하는 데 도움이 되는 탐지 또는 대응 AWS 보안 제어 역할을 합니다. 자동 응답 작업의 예로는 VPC 보안 그룹 수정, Amazon EC2 인스턴스 패치 적용 또는 자격 증명 교체 등이 있습니다.

서버 측 암호화

데이터를 AWS 서비스 수신하는가 대상에서 데이터를 암호화합니다.

서비스 제어 정책(SCP)

AWS Organizations에 속한 조직의 모든 계정에 대한 권한을 중앙 집중식으로 제어하는 정책입니다. SCP는 관리자가 사용자 또는 역할에 위임할 수 있는 작업에 대해 제한을 설정하거나 가드레일을 정의합니다. SCP를 허용 목록 또는 거부 목록으로 사용하여 허용하거나 금지할 서비스 또는 작업을 지정할 수 있습니다. 자세한 내용은 AWS Organizations 설명서의 서비스 제어 정책을 참조하세요.

서비스 엔드포인트

에 대한 진입점의 URL입니다 AWS 서비스. 엔드포인트를 사용하여 대상 서비스에 프로그래밍 방식으로 연결할 수 있습니다. 자세한 내용은 AWS 일반 참조의 AWS 서비스 엔드포인트를 참조하십시오.

서비스 수준에 관한 계약(SLA)

IT 팀이 고객에게 제공하기로 약속한 내용(예: 서비스 가동 시간 및 성능)을 명시한 계약입니다.

서비스 수준 표시기(SLI)

오류율, 가용성 또는 처리량과 같은 서비스의 성능 측면에 대한 측정입니다.

서비스 수준 목표(SLO)

서비스 수준 지표로 측정되는 서비스의 상태를 나타내는 대상 지표입니다.

공동 책임 모델

클라우드 보안 및 규정 준수를 AWS 위해와 공유하는 책임을 설명하는 모델입니다. AWS 는 클라우드의 보안을 담당하는 반면, 사용자는 클라우드의 보안을 담당합니다. 자세한 내용은 공동 책임 모델을 참조하십시오.

SIEM

보안 정보 및 이벤트 관리 시스템을 참조하세요.

단일 장애 지점(SPOF)

애플리케이션의 중요한 단일 구성 요소에 장애가 발생하여 시스템이 중단될 수 있습니다.

SLA

서비스 수준 계약을 참조하세요.

SLI

서비스 수준 표시기를 참조하세요.

SLO

서비스 수준 목표를 참조하세요.

분할 앤 시드 모델

현대화 프로젝트를 확장하고 가속화하기 위한 패턴입니다. 새로운 기능과 제품 릴리스가 정의되면 핵심 팀이 분할되어 새로운 제품 팀이 만들어집니다. 이를 통해 조직의 역량과 서비스 규모를 조정하고, 개발자 생산성을 개선하고, 신속한 혁신을 지원할 수 있습니다. 자세한 내용은 에서 애플리케이션 현대화에 대한 단계별 접근 방식을 참조하세요 AWS 클라우드.

SPOF

단일 장애 지점을 참조하세요.

스타 스키마

하나의 큰 팩트 테이블을 사용하여 트랜잭션 또는 측정된 데이터를 저장하고 하나 이상의 작은 차원 테이블을 사용하여 데이터 속성을 저장하는 데이터베이스 조직 구조입니다. 이 구조는 데이터 웨어하우스 또는 비즈니스 인텔리전스용으로 설계되었습니다.

Strangler Fig 패턴

레거시 시스템을 폐기할 수 있을 때까지 시스템 기능을 점진적으로 다시 작성하고 교체하여 모놀리식 시스템을 현대화하기 위한 접근 방식. 이 패턴은 무화과 덩굴이 나무로 자라 결국 숙주를 압도

하고 대체하는 것과 비슷합니다. Martin Fowler가 모놀리식 시스템을 다시 작성할 때 위험을 관리하는 방법으로 이 패턴을 도입했습니다. 이 패턴을 적용하는 방법의 예는 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

서브넷

VPC의 IP 주소 범위입니다. 서브넷은 단일 가용 영역에 상주해야 합니다.

감독 제어 및 데이터 획득(SCADA)

제조에서 하드웨어와 소프트웨어를 사용하여 물리적 자산과 프로덕션 작업을 모니터링하는 시스템입니다.

대칭 암호화

동일한 키를 사용하여 데이터를 암호화하고 복호화하는 암호화 알고리즘입니다.

합성 테스트

사용자 상호 작용을 시뮬레이션하여 잠재적 문제를 감지하거나 성능을 모니터링하는 방식으로 시스템을 테스트합니다. Amazon CloudWatch Synthetics를 사용하여 이러한 테스트를 생성할 수 있습니다.

시스템 프롬프트

LLM에 컨텍스트, 지침 또는 지침을 제공하여 동작을 지시하는 기법입니다. 시스템 프롬프트는 컨텍스트를 설정하고 사용자와의 상호 작용을 위한 규칙을 설정하는 데 도움이 됩니다.

T

tags

AWS 리소스를 구성하기 위한 메타데이터 역할을 하는 키-값 페어입니다. 태그를 사용하면 리소스를 손쉽게 관리, 식별, 정리, 검색 및 필터링할 수 있습니다. 자세한 내용은 AWS 리소스에 태그 지정을 참조하십시오.

대상 변수

지도 ML에서 예측하려는 값으로, 결과 변수라고도 합니다. 예를 들어, 제조 설정에서 대상 변수는 제품 결함일 수 있습니다.

작업 목록

런북을 통해 진행 상황을 추적하는 데 사용되는 도구입니다. 작업 목록에는 런북의 개요와 완료해야 할 일반 작업 목록이 포함되어 있습니다. 각 일반 작업에 대한 예상 소요 시간, 소유자 및 진행 상황이 작업 목록에 포함됩니다.

테스트 환경

환경을 참조하세요.

훈련

ML 모델이 학습할 수 있는 데이터를 제공하는 것입니다. 훈련 데이터에는 정답이 포함되어야 합니다. 학습 알고리즘은 훈련 데이터에서 대상(예측하려는 답)에 입력 데이터 속성을 매핑하는 패턴을 찾고, 이러한 패턴을 캡처하는 ML 모델을 출력합니다. 그런 다음 ML 모델을 사용하여 대상을 모르는 새 데이터에 대한 예측을 할 수 있습니다.

전송 게이트웨이

VPC와 온프레미스 네트워크를 상호 연결하는 데 사용할 수 있는 네트워크 전송 허브입니다. 자세한 내용은 AWS Transit Gateway 설명서의 전송 게이트웨이란 무엇입니까?를 참조하세요.

트렁크 기반 워크플로

개발자가 기능 브랜치에서 로컬로 기능을 구축하고 테스트한 다음 해당 변경 사항을 기본 브랜치에 병합하는 접근 방식입니다. 이후 기본 브랜치는 개발, 프로덕션 이전 및 프로덕션 환경에 순차적으로 구축됩니다.

신뢰할 수 있는 액세스

사용자를 대신하여 AWS Organizations 및 계정에서 조직에서 작업을 수행하도록 지정하는 서비스에 대한 권한 부여. 신뢰할 수 있는 서비스는 필요할 때 각 계정에 서비스 연결 역할을 생성하여 관리 작업을 수행합니다. 자세한 내용은 설명서의 다른 AWS 서비스와 AWS Organizations 함께 사용을 참조하세요 AWS Organizations .

튜닝

ML 모델의 정확도를 높이기 위해 훈련 프로세스의 측면을 여러 변경하는 것입니다. 예를 들어, 레이블링 세트를 생성하고 레이블을 추가한 다음 다양한 설정에서 이러한 단계를 여러 번 반복하여 모델을 최적화하는 방식으로 ML 모델을 훈련할 수 있습니다.

피자 두 판 팀

피자 두 판이면 충분한 소규모 DevOps 팀. 피자 두 판 팀 규모는 소프트웨어 개발에 있어 가능한 최상의 공동 작업 기회를 보장합니다.

U

불확실성

예측 ML 모델의 신뢰성을 저해할 수 있는 부정확하거나 불완전하거나 알려지지 않은 정보를 나타내는 개념입니다. 불확실성에는 두 가지 유형이 있습니다. 인식론적 불확실성은 제한적이고 불완전한 데이터에 의해 발생하는 반면, 우연한 불확실성은 데이터에 내재된 노이즈와 무작위성에 의해 발생합니다. 자세한 내용은 Quantifying uncertainty in deep learning systems 가이드를 참조하십시오.

차별화되지 않은 작업

애플리케이션을 만들고 운영하는 데 필요하지만 최종 사용자에게 직접적인 가치를 제공하거나 경쟁 우위를 제공하지 못하는 작업을 헤비 리프팅이라고도 합니다. 차별화되지 않은 작업의 예로는 조달, 유지보수, 용량 계획 등이 있습니다.

상위 환경

환경을 참조하세요.

V

정리

스토리지를 회수하고 성능을 향상시키기 위해 증분 업데이트 후 정리 작업을 수반하는 데이터베이스 유지 관리 작업입니다.

버전 제어

리포지토리의 소스 코드 변경과 같은 변경 사항을 추적하는 프로세스 및 도구입니다.

VPC 피어링

프라이빗 IP 주소를 사용하여 트래픽을 라우팅할 수 있게 하는 두 VPC 간의 연결입니다. 자세한 내용은 Amazon VPC 설명서의 VPC 피어링이란?을 참조하십시오.

취약성

시스템 보안을 손상시키는 소프트웨어 또는 하드웨어 결함입니다.

W

웜 캐시

자주 액세스하는 최신 관련 데이터를 포함하는 버퍼 캐시입니다. 버퍼 캐시에서 데이터베이스 인스턴스를 읽을 수 있기 때문에 주 메모리나 디스크에서 읽는 것보다 빠릅니다.

웜 데이터

자주 액세스하지 않는 데이터입니다. 이런 종류의 데이터를 쿼리할 때는 일반적으로 적절히 느린 쿼리가 허용됩니다.

창 함수

현재 레코드와 어떤 식으로든 관련된 행 그룹에 대해 계산을 수행하는 SQL 함수입니다. 창 함수는 이동 평균을 계산하거나 현재 행의 상대 위치를 기반으로 행 값에 액세스하는 등의 작업을 처리하는 데 유용합니다.

워크로드

고객 대면 애플리케이션이나 백엔드 프로세스 같이 비즈니스 가치를 창출하는 리소스 및 코드 모음입니다.

워크스트림

마이그레이션 프로젝트에서 특정 작업 세트를 담당하는 직무 그룹입니다. 각 워크스트림은 독립적이지만 프로젝트의 다른 워크스트림을 지원합니다. 예를 들어, 포트폴리오 워크스트림은 애플리케이션 우선순위 지정, 웨이브 계획, 마이그레이션 메타데이터 수집을 담당합니다. 포트폴리오 워크스트림은 이러한 자산을 마이그레이션 워크스트림에 전달하고, 마이그레이션 워크스트림은 서버와 애플리케이션을 마이그레이션합니다.

WORM

쓰기를 한 번, 많이 읽기를 참조하세요.

WQF

AWS 워크로드 검증 프레임워크를 참조하세요.

한 번 쓰기, 많이 읽기(WORM)

데이터를 한 번 쓰고 데이터가 삭제되거나 수정되지 않도록 하는 스토리지 모델입니다. 권한 있는 사용자는 필요한 만큼 데이터를 읽을 수 있지만 변경할 수는 없습니다. 이 데이터 스토리지 인프라는 변경할 수 없는 것으로 간주됩니다.

Z

제로데이 익스플로잇

제로데이 취약성을 활용하는 공격, 일반적으로 맬웨어입니다.

제로데이 취약성

프로덕션 시스템의 명백한 결함 또는 취약성입니다. 위협 행위자는 이러한 유형의 취약성을 사용하여 시스템을 공격할 수 있습니다. 개발자는 공격의 결과로 취약성을 인지하는 경우가 많습니다.

제로샷 프롬프트

LLM에 작업을 수행하기 위한 지침을 제공하지만 작업에 도움이 될 수 있는 예제(샷)는 제공하지 않습니다. LLM은 사전 훈련된 지식을 사용하여 작업을 처리해야 합니다. 제로샷 프롬프트의 효과는 작업의 복잡성과 프롬프트의 품질에 따라 달라집니다. 스크린샷이 거의 없는 프롬프트도 참조하세요.

좀비 애플리케이션

평균 CPU 및 메모리 사용량이 5% 미만인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하는 것이 일반적입니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.