Amazon Web Services를 이용한
백업 및 복구 접근방식

2012년 12월

Simon Elisha

요약

기존 기업의 백업 및 복원 전략은 local area network (LAN) 또는 storage area network (SAN)에 걸쳐 서버의 전체 콘텐츠를 백업하는 전형적인 에이전트 기반의 접근 방식을 취합니다. 기존의 설계는 고장난 구성 요소를 교체하는 것이 복잡하고, 시간 소모적이며, 운영하기도 버거웠기 때문에 이러한 방식을 요구했습니다. 이로 인해 관리가 복잡하고 운영하기가 자원 소모적인 백업 환경이 되었으며, 급증하는 워크로드를 처리하기 위한 데이터 중복 제거 및 가상 테이프 라이브러리 등의 기술이 요구되었습니다.

AWS 플랫폼은 백업 및 복원에 대해 다음의 특성을 갖는 훨씬 더 가벼운 접근 방식을 제공합니다.

- 컴퓨터는 이제 하드웨어 기반보다는 코드를 통해 가상 추상 리소스를 시작합니다.
- 선행 투자 비용보다는 증분 비용으로 용량을 사용할 수 있습니다.
- 리소스 프로비저닝은 단 몇 분만에 실행되며, 실시간 구성에 자체 렌딩합니다.
- 서버 "이미지"를 온 디맨드할 수 있고, 조직에서 유지 관리할 수 있으며, 즉시 활성화시킬 수 있습니다.

이러한 특성 덕분에 적은 인프라 비용으로 삭제되거나 손상된 데이터를 복원할 수 있습니다.

본 문서는 쉽고 가벼운 데이터 백업 및 복원 성능을 활용할 수 있는 몇 가지 높은 수준의 개념을 설명하기 위해 작성되었습니다.

서버보다는 구성을 보호

Amazon Elastic Compute Cloud(Amazon EC2[1]) 서비스는 웹 서버 또는 애플리케이션 서버와 같은 표준 서버의 백업 및 복원을 가능하게 하여 사용자가 서버보다는 정적 데이터 및 구성 보호에 온전히 집중할 수 있습니다. 이 데이터 세트는 전형적으로 다양한 애플리케이션 파일, 운영 시스템 파일, 임시 파일 등에 포함된 서버 데이터 모음보다 훨씬 더 작습니다. 이러한 접근 방식의 변화는 정기적인 야간 증분 백업 또는 주말 전체 백업이 이제는 훨씬 더 적은 시간 및 스토리지 공간을 소비함을 의미합니다.

Amazon EC2에서 컴퓨팅 인스턴스가 시작될 때, 이는 Amazon Machine Image(AMI)에 기초하며[2], 기존의 스토리지 볼륨에도 연결할 수 있습니다 (예. Amazon Elastic Block Store(Amazon EBS))[3]. 또한, 새로운 인스턴스를 시작할 때, "사용자 데이터"[4]를 동적 구성 매개변수로 내부에 액세스할 수 있는 인스턴스로 전달하는 것이 가능합니다.

워크플로의 예는 다음과 같습니다.

- 웹 서버의 새로운 인스턴스를 시작하고 초기 설정에 요구되는 보안 자격 증명 및 웹 서버의 "identity"를 통과합니다. 인스턴스는 운영 시스템 및 관련 웹 서버 애플리케이션 (예. Apache 또는 IIS)을 포함하는 사전 구축된 AMI에 기반합니다.
- 시작 하자마자, 부트 스크립트는 지정된 안전한 Amazon Simple Storage Service (Amazon S3)[5] 버킷에 액세스하는데, 버킷은 특정 구성 파일을 담고 있습니다.
- 구성 파일은 서버 설치에 대한 다양한 지침 (예. 웹 서버 매개변수, 관련 서버의 위치, 설치할 추가적 소프트웨어, 패치 업데이트)을 담고 있습니다.
- 서버는 특정 구성을 실행하여 서비스를 준비합니다. 이 프로세스를 실행하기 위한 오픈 소스 도구를 cloud-init이라 하는데[6], 이미 Amazon Linux AMI에 설치되어 있으며 여러 다른 Linux 배포 버전에서도 사용 가능합니다.

---

[1] http://aws.amazon.com/ec2/
[2] http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/index.html?AMIs.html
[3] http://aws.amazon.com/ebs/
[4] http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/index.html?AESDG-chapter-instancedata.html
[5] http://aws.amazon.com/s3/
[6] https://launchpad.net/cloud-init

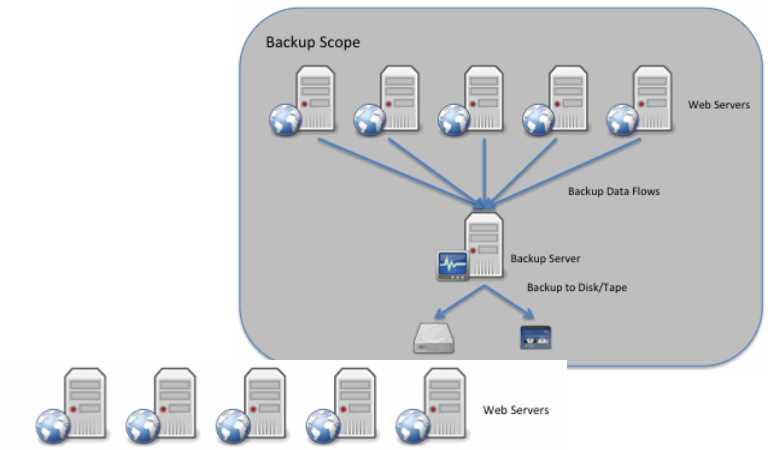


그림 1: 기존 백업 접근방식

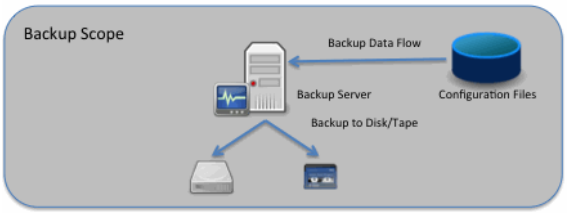


그림 2: Amazon EC2 백업 접근방식

이러한 경우, 서버 자체를 백업할 필요가 없습니다. AMI 조합 및 구성 파일은 관련 구성을 포함하고 있습니다. 따라서 백업 및 복구가 필요한 유일한 구성 요소는 AMI 및 구성 파일입니다.

10개의 서버를 갖춘 웹 팜을 생각해보십시오. 각각의 5GB 서버는 운영 시스템과 관련 구성 파일을 담고 있으며, 전체 백업을 실행하기 위해 50GB의 스토리지 및 네트워크 용량을 요구합니다. (웹 콘텐츠는 전형적으로 독립적으로 백업되는 별도의 저장소에 저장됩니다.) 이를 AMI (5GB로 가정) 및 관련 구성 파일 (일반적으로 수십 KB)만 보호해야 하는 AWS 접근 방식과 비교합니다. 이는 백업 및 복원 비용을 엄청나게 줄이고, "백업 윈도우"를 제거하며, 환경에 대한 효율적인 버전 통제를 가능하게 합니다.

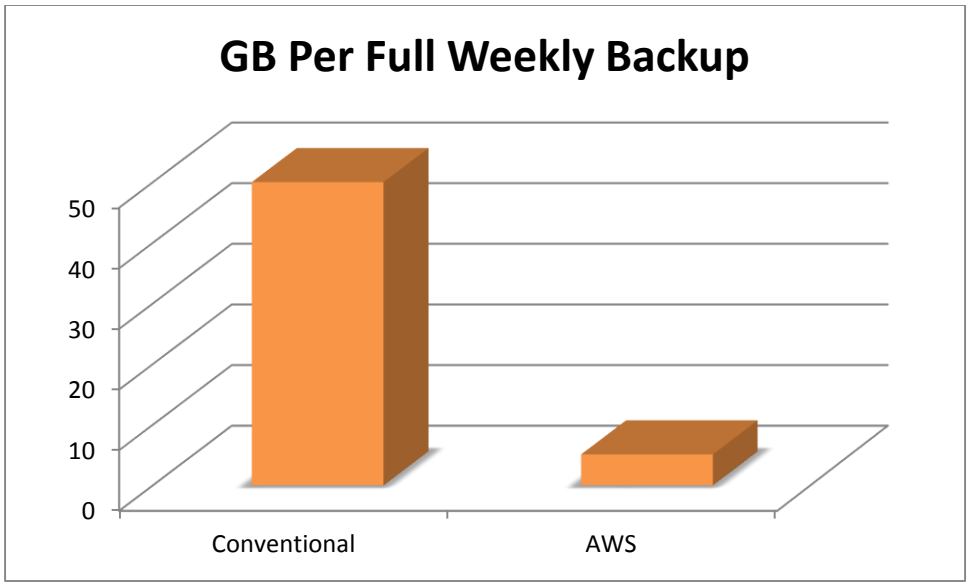


그림 3: 백업 데이터 볼륨 감소 사례

자체 구성 인스턴스 - 유연성 및 개발 옵션의 배포

원하는 대로 인스턴스를 시작 및 종료할 수 있고 동시에 실행되는 다른 버전의 애플리케이션이 있기 때문에 보다 정교하고 유연한 배포 옵션을 활용할 수 있습니다. 자체 구성 인스턴스는 롤링 업그레이드 및 해당 환경에서의 A/B 테스팅 등의 기술을 도입할 수 있도록 합니다.

예를 들어, 아키텍처에 새로운 버전의 애플리케이션 서버를 도입하려면 다음의 접근 방식을 따라야 합니다.

1. 올바른 AMI 버전 및 관련 구성 파일에 근거한 애플리케이션의 새로운 인스턴스를 생성합니다. 이 예에서는 이를 "애플리케이션 버전 2.0"이라 합니다.
2. 관련 로드 밸런서에 "애플리케이션 버전 2.0"을 맵핑하고 이제 "로테이션 중"인 서버는 고객의 요청에 대한 서비스를 제공합니다.
3. "애플리케이션 버전 2.0"이 생산 중임을 확인하면 기존의 "애플리케이션 버전 1.0" 인스턴스를 중단하거나 종료할 수 있습니다.
4. 이 지점에서는 전체 애플리케이션이 중단 없이 버전 2.0 모드에서 중단된 인스턴스를 사용하여 버전 1.0으로 간단히 롤백하는 기능과 함께 운영됩니다.

이 예를 좀 더 설명하면 새로운 애플리케이션 성능 또는 기능의 A/B 테스팅을 활용하고자할 수 있습니다. 애플리케이션 서버의 새로운 버전을 롤링 업그레이드 아키텍처에 도입했던 것과 동일한 방식으로, 지정된 고객을 특정 (새로운 버전) 애플리케이션 인스턴스로 디렉션하도록 로드 밸런서를 사용할 수 있으며, 나머지 고객이 기존 버전의 애플리케이션을 계속 사용하는 동안 이를 실행할 수 있습니다.

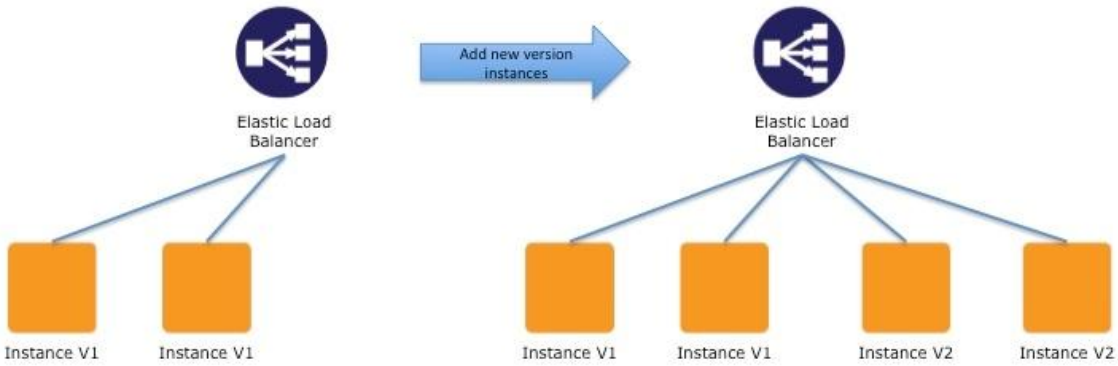


그림 4: 롤링 업그레이드 - 새로운 버전 인스턴스 추가

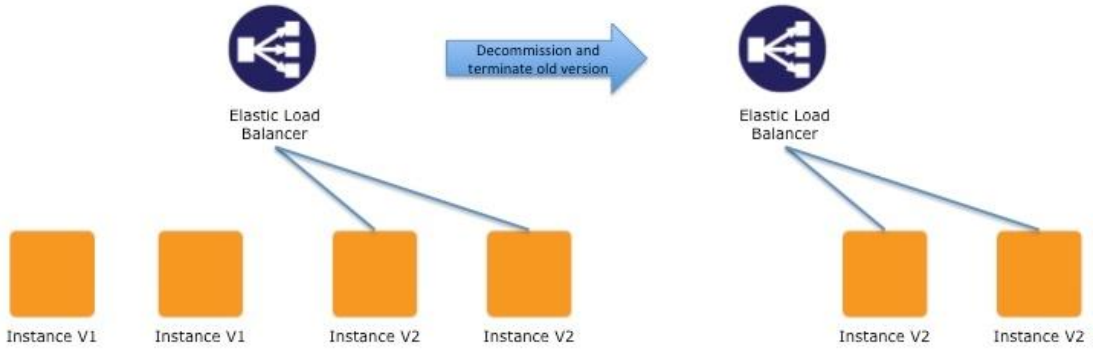


그림 5: 롤링 업그레이드 - 오래된 버전 사용 중지 및 종료

고려할 다른 중요한 측면은 안전성 그리고 타협의 개선입니다. 인스턴스는 쉽게 교체되기 때문에 "수리" 보다는 "교체" 전략에 집중할 수 있습니다. 이 전략은 반응 속도 및 복잡성을 현저히 줄입니다.

그 예로, 인터넷 서비스를 호스팅하는 콘텐트 관리 시스템(CMS)을 생각해 보겠습니다. 어떠한 이유로 코드의 최신 버전이 배포되지 않았고, 해커는 이를 알고 해당 사이트에 보안 위반을 사용합니다. 인스턴스가 타협된 범죄 과학적 분석은 시간 소모적이며, 100% 확신을 가지고 각각에 대해 "sanitize"하는 것은 대개 불가능합니다. 그 대신, 타협된 인스턴스를 간단히 종료하고 "새 것"으로 교체합니다. 그러면 이러한 새로운 인스턴스는 소프트웨어의 최신 패치 버전이 항상 배포되어 있음을 확인하기 위해 업데이트된 구성 파일을 활용할 것입니다. 이 접근 방식을 취해 완전히 개선되지 않은 보안 위반 리스크를 제거하고 새로운 인스턴스가 타협되지 않도록 보장합니다.

이 접근 방식은 규모에 따라 분산 시스템을 배포할 때 사용하는 핵심 설계 패턴인 "architect for failure"에 대해 효율적인 방법도 제공합니다. 원하는 대로 구성 요소를 자동으로 교체할 수 있기 때문에 예상하지 못한 실패로 서비스 제공에 영향받을 일은 없습니다.

Amazon Machine Image(AMI)의 백업 및 복구

등록한 AMI은 Amazon EBS 스냅샷을 사용하여 자동으로 계정에 저장됩니다. 이러한 스냅샷은 Amazon S3 내에 있으며 내구성이 강합니다[7]. 이는 AMI에 대한 기본 스토리지 매커니즘이 여러 장애 상황으로부터 보호됨을 의미합니다.

별도의 AWS 계정 간의 AMI 공유 또한 가능합니다. 다음의 방법으로 결과적으로 전적으로 독립적인 AMI 카피를 생성할 수 있습니다.

- 원본 AMI를 제어하는 또 다른 지정된 AWS 계정에 공유
- 공유 AMI에 근거하여 새로운 인스턴스 시작
- 실행 중인 이 인스턴스로부터 새로운 AMI 시작

그러면 새로운 AMI가 2차 계정에 저장되고 원본 AMI의 독립적인 카피가 됨. 물론 동일한 계정 내에서 여러 AMI 카피를 생성할 수도 있습니다.

구성 파일의 백업 및 복구

고객은 구성 파일에 대해 다양한 버전의 관리 접근 방식을 사용하며, 사용자는 Amazon EC2를 구성하는데 사용했던 파일과 동일한 체제를 따를 수 있습니다. 예를 들어, 다른 버전의 구성 파일을 지정된 장소에 저장하고 다른 코드와 마찬가지로 안전하게 통제할 수 있을 것입니다. 그리고 나서 이러한 코드 저장소를 적합한 백업 주기 (에. 매일, 매주, 매달)와 스냅샷을 이용하여 안전한 위치에 백업합니다.

또한, 정기적으로 다른 위치에 파일을 백업함과 동시에 서비스 내구성의 이점을 활용하며 구성 파일을 저장하도록 Amazon S3를 사용할 수 있을 것입니다. 부트스트래핑 접근 방식은 오직 상상력에 의해서만 제한됩니다. 사용자가 AWS 리소스, 그리고 간단한 JSON 파일의 모든 관련 종속성 또는 런타임 매개 변수를 설명할 수 있는 만큼 AWS CloudFormation 템플릿을 사용할 것을 제시합니다.

데이터베이스 및 파일 서버의 백업

데이터베이스 데이터의 백업과 파일 서버의 백업은 웹과 애플리케이션 layer부터 다릅니다. 대체로, 데이터베이스와 파일 서버는 많은 양의 비즈니스 테이터 (수 십 GB에서 수 십 TB까지)를 담고 있으며, 항상

---

[7] http://aws.amazon.com/s3/ - protecting

보존되고 보로되어야 합니다. 이러한 경우에 빠르고 안정적이며 공간 효율적인 백업을 생성할 수 있는 스냅샷과 같은 효율적인 데이터 이전 기술을 활용할 수 있습니다.

Amazon EBS 볼륨의 RAID 세트를 기반으로 설계된 데이터베이스를 위한 또 다른 백업 접근 방식은 단일 Amazon EBS 볼륨을 사용하여 구축된 다른 데이터베이스 인스턴스에 비동기식으로 데이터를 복제하는 것입니다. 대상 Amazon EBS 볼륨이 느린 성능을 갖는 동안에도 데이터 액세스를 사용하지 않으며, Amazon EBS 스냅샷 기능 (Amazon EBS 스냅샷 옵션 섹션 참조)을 이용하여 Amazon S3로 쉽게 스냅샷할 수있습니다.

정적 콘텐츠 백업 대안

정적 정보의 대용량 데이터 세트 (예. 맵 타일 또는 웹사이트 그래픽)를 관리할 경우, 해당 데이터를 99.999999999%의 객체 스토리지 내구성을 제공하도록 설계된 Amazon S3로 이전하기를 선택할 수 있습니다. 이는 데이터가 웹 서버를 통하기 보다는 Amazon S3로부터 직접 제공되고 높은 내구성을 갖도록하므로 애플리케이션 성능이 개선될 수 있습니다.

논리적 손상을 보호하기 위해 객체 버전 관리[8], MFA 삭제[9] 및 또 다른 Amazon S3 버킷으로의 간단한 데이터 복사 등의 기술을 사용할 수 있습니다.

Amazon EBS용 스냅샷 옵션

Amazon EC2 볼륨은 블록 기반 데이터를 저장하기 위해 Amazon EBS를 사용합니다. 이러한 예로는 파일 시스템과 데이터베이스가 있습니다. Amazon EBS는 단순히 AWS Management Console, command line interface(CLI) 또는 API를 사용하여 Amazon S3에 볼륨의 스냅샷을 생성할 수 있도록 합니다. 콘솔을 사용한 **스냅샷 생성** 옵션의 클릭은 Amazon S3에 스냅샷 생성을 시작합니다.

[8] http://docs.amazonwebservices.com/AmazonS3/latest/dev/Versioning.html
[9] http://docs.amazonwebservices.com/AmazonS3/latest/dev/UsingMFADelete.html

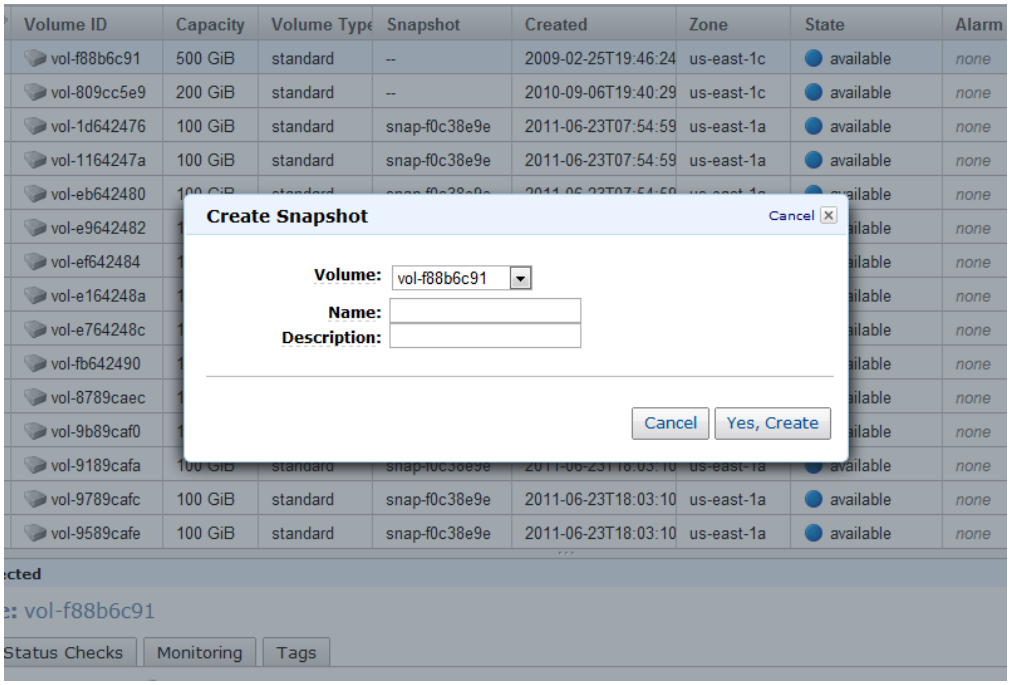


그림 3 - Amazon EBS의 콘솔을 사용한 스냅샷 생성.

ec2-create-snapshot 명령을 사용하여 스냅샷을 생성할 수도 있습니다.

이러한 명령을 백업 전략에 적용할 경우에는 데이터를 직접 내구성이 강한 디스크 기반 스토리지에 보관합니다. 일정을 조정하여 규칙적으로 명령을 내릴 수 있으며, Amazon S3의 경제적인 비용 덕분에 많은 세대의 데이터를 보관할 수 있습니다. 또한, 스냅샷은 블록 기반이기 때문에 초기 스냅샷이 생성된 후에 변경된 데이터에 대해서만 공간을 소비합니다.

스냅샷으로부터 데이터를 복원하기 위해서는 콘솔이나 CLI 명령 ec2-create-volume을 사용하여 기존의 스냅샷으로부터 새로운 볼륨을 생성합니다. 예를 들어, 볼륨을 이전 point-in-time 백업으로 복원하기 위해서는 다음의 절차를 따를 수 있습니다.

1. 다음의 명령을 사용하여 백업 스냅샷으로부터 새로운 볼륨을 생성합니다.

   ec2-create-volume –z us-west-1b –snapshot *MySnapshotName*
2. Amazon EC2 인스턴스 내에서 기존 볼륨을 마운트 해제 (예. Linux의 umount 또는 Windows의 Logical Volume Manager 사용)합니다.
3. 다음의 명령을 사용하여 인스턴스로부터 기존 볼륨을 분리합니다.

   ec2-detach-volume *OldVolume*

4. 다음의 명령을 사용하여 스냅샷으로부터 생성된 새로운 볼륨을 부착합니다.

   ec2-attach-volume *VolumeID* –I *InstanceID* –d *Device*
5. 실행 중인 인스턴스에 볼륨을 리마운트합니다.

이 프로세스는 필요에 따라 전체 볼륨 데이터를 복원하는 빠르고 안정적인 방법입니다. 부분적 복원만 필요한 경우에는 다른 디바이스 이름 하에서 실행 중인 인스턴스에 볼륨을 부착하고 마운트한 후에, 운영 시스템 복사 명령을 사용하여 백업 볼륨으로부터 생산 볼륨으로 데이터를 복사할 수 있습니다.

Amazon EBS 스냅샷은 콘솔, API 또는 GUI를 통한 Amazon EBS 스냅샷 복사 기능을 사용하여AWS 지역 간에도 복사할 수 있습니다[10]. 이로 인해 기본 복제 기술을 관리하지 않고도 지역 외에서 데이터를 보호할 수 있습니다.

지속적인 또는 "핫" 백업의 생성

시스템을 백업할 경우, 시스템을 어떠한 처리도 실행하지 않는 "조용한" 상태로 유지하는 것이 이상적입니다. 백업의 관점에서 볼 때 "이상적"인 상태는 컴퓨터가 트래픽을 수용하지 않는 상태이지만, 24/7 IT 운영이 표준이므로 이러한 이상적인 상태는 매우 드뭅니다.

따라서 "clean" 백업을 위해서는 파일 시스템 또는 데이터베이스를 "quiesce"하는 것이 필요합니다. 이를 수행하는 방법은 데이터베이스 및/또는 파일 시스템에 따라 다르며, 실사도 필요합니다. 데이터베이스에 대한 프로세스를 요약하자면 다음과 같습니다.

- 가능하다면 데이터베이스를 "hot backup mode"로 둡니다. 또는, 데이터베이스 "읽기 전용 복제본" 사본을 생성합니다. 이것은 최신 데이터베이스 사본이지만 별도의 인스턴스에서 실행합니다. AWS 상에서 백업을 수행하는 필요한 시간 동안 이 인스턴스를 실행할 수 있으며 이후 닫아 리소스를 저장해야 함에 유의합니다. 또한, 추가적인 복제 워크로드 때문에 읽기 전용 복제본이 존재하는 동안 주 데이터베이스에 대한 성능에 영향을 미칠 수 있습니다.
- 관련 Amazon EBS 스냅샷 명령.
- 데이터베이스를 핫 백업 모드에서 꺼낸 후에, 또는 읽기 전용 복제본을 사용하는 경우에 읽기 전용 복제본 인스턴스를 종료합니다.

파일 시스템의 백업은 유사하게 작동하며, 특정 운영 시스템 또는 파일 시스템의 성능에 크게 의존합니다. 지속적인 백업을 위해 자체 데이터를 플러쉬할 수 있는 파일 시스템의 예는 xfs (xfs_freeze)입니다. 질문의 파일 시스템이 프리즈 기능을 지원하지 않을 경우에는 이를 마운트 해제하고, 스냅샷 명령을 내린 후 파일 시스템을 리마운트 합니다. 또는, I/O의 작동 중지를 지원하는 logical volume manager를 사용하여 이 프로세스를 촉진할 수 있습니다.

[10] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-copy-snapshot.html

스냅샷 프로세스는 빠르게 실행할 수 있고 “point in time”을 캡처할 수 있기 때문에 백업하는 볼륨은 단 몇 초 만에 마운트 해제되어야 합니다. 이는 백업 “window”가 가능한 작고, 중단 시간이 예측 가능하며, 효과적으로 일정을 조정할 수 있도록 보장합니다. 스냅샷 생성의 데이터 복사 프로세스가 길어질 동안 볼륨의 마운트 해제를 요구하는 스냅샷 활동은 매우 빠릅니다. 백업 체계를 구조화할 때 이 두 프로세스를 혼돈하지 마십시오.

Amazon 관계형 데이터베이스 서비스에 대한 백업

Amazon 관계형 데이터베이스 서비스(Amazon RDS)[11]는 자동 백업을 포함합니다. 따라서 데이터베이스에 대한 백업을 생성하는데 특정 명령을 내리지 않아도 됩니다.

Amazon RDS는 DB 인스턴스 백업 및 복구를 위한 두 가지 방법, 즉 자동 백업과 데이터베이스 스냅샷(DB 스냅샷)을 제공합니다.

- 자동 백업은 DB 인스턴스의 point-in-time 복구를 지원합니다. DB 인스턴스의 자동 백업을 활성화하면 Amazon RDS가 매일 자동으로 데이터에 대한 완전한 백업을 실행하고(기본 백업 기간 내에), 트랜잭션 로그를 캡처(DB 인스턴스를 업데이트할 때)합니다. point-in-time 복구를 시작 시, 사용자가 요청한 특정 시간으로 DB 인스턴스를 복구하기 위해 가장 적합한 일일 백업에 트랜잭션 로그가 적용됩니다. Amazon RDS는 사용자가 지정한 일정 기간 (보존 기간) 동안 DB 인스턴스의 백업을 보관합니다. 보존 기관은 기본적으로 1일이지만 최대 35일까지 설정할 수 있습니다. point-in-time 복원을 시작할 수 있으며, 보존 기간을 최대 복구 가능한 최근 시간까지 초 단위로 지정할 수 있습니다. DescribeDBInstances 호출을 사용하여 DB 인스턴스의 복구 가능한 최근 시간 (일반적으로 최근 5분)으로 돌아갈 수 있습니다. 또는, AWS Management Console에서 DB 인스턴스를 선택하고 콘솔의 아래쪽 패널에 있는 **Description** 탭에서 복구 가능한 최근 시간을 찾을 수 있습니다.

- 사용자에 의해 개시되는 DB 스냅샷은 원하는 빈도만큼 DB 인스턴스를 일관되게 백업할 수 있으며 이후 언제든지 해당 상태로 복구할 수 있습니다. DB 스냅샷은 AWS Management Console 또는 CreateDBSnapshot 호출을 통해 작성할 수 있으며, 콘솔 또는 DeleteDBSnapshot 호출로 명시적으로 삭제할 때까지 유지됩니다.

point in time으로 또는 DB 스냅샷으로부터 복원할 때, 새로운 DB 인스턴스는 새로운 endpoint로 작성됩니다. (원할 경우 AWS Management Console 또는 DeleteDBInstance 호출을 사용해서 오래된 DB 인스턴스를 삭제할 수 있습니다.) 이렇게 하면 특정 DB 스냅샷 또는 point in time으로부터 다중 DB 인스턴스를 작성할 수 있습니다.

[11] http://aws.amazon.com/rds/

멀티 볼륨 백업

어떤 경우에는 잠재적 처리량을 향상시키기 위해 logical volume manager를 사용하여 다양한 Amazon EBS 볼륨 전반에 걸쳐 데이터를 스트라이핑할 수도 있습니다. logical volume manager (예. mdadm 또는 LVM)를 이용 시, 기본 디바이스 보다는 volume manager layer로부터 백업을 실행하는 것이 중요합니다. 이는 모든 메타데이터는 일관되며 다양한 서브 구성 요소 볼륨이 일관됨을 보장합니다. 이러한 경우, logical volume manager를 사용하여 이 유형의 백업에 대해 ec2-create-snapshot 명령을 사용할 수 있습니다. 이를 완료하기 위해 여러 접근 방식을 사용할 수 있으며, alestic.com에 의해 작성된 스크립트를 예로 들겠습니다 (http://alestic.com/2009/09/ec2-consistent-snapshot).

logical volume manager 또는 파일 시스템 수준에서 이러한 특성의 백업을 수행할 수도 있습니다. 이러한 경우, "기존" 백업 에이전트를 이용해 네트워크에서 백업할 데이터를 선택합니다. Zmanda, NetBackup 또는 CommVault 등의 도구를 사용할 때, 일관된 서버명/IP 주소를 예상해야 함을 숙지합니다. 결과적으로, Virtual Private Cloud(VPC)[12]에 배포된 인스턴스와 동시에 이러한 도구를 사용하는 것은 안정성을 보장하는 최고의 방법입니다.

다른 접근 방법은 싱글 라지 볼륨에 존재하는 주요 시스템 볼륨의 복제를 생성하는 것입니다. 하나의 라지 볼륨만 백업할 필요가 있으며 주요 시스템에서는 백업이 이루어지지 않으므로 백업 프로세스가 간소화됩니다. 하지만 싱글 볼륨이 백업하는 동안 변화 유지를 충분하게 수행하는지의 여부와 애플리케이션에 대한 최대 볼륨 크기가 적당한지를 확인하는 것은 중요합니다.

기타 백업 및 복구 통합 지점

Amazon S3에 Oracle Secure Backup Cloud Module을 이용한 Oracle Backup

데이터베이스 관리자는 항상 Oracle 데이터베이스의 데이터를 보호할 효율적인 방법을 모색합니다. Oracle은 Oracle 데이터베이스에서 Amazon S3 버킷에 직접 데이터를 백업하는 기능을 가능케 했습니다. 이로 인해 백업은 Amazon S3가 가능케 한 경제적인 그리고 내구성 강한 스토리지의 장점을 활용함과 동시에 Oracle 데이터베이스 프레인워크 및 RMAN을 사용한 운영 절차에 기본적으로 통합되었습니다.

OSB Cloud Module의 설치 및 운영에 대한 자세한 내용은 http://aws.amazon.com/oracle을 참조하십시오.

Oracle 백업에 대한 이 접근 방식은 저렴하고 신뢰할 만한 오프 프레미스 Oracle 데이터베이스 백업을 가능케 했으며, 온 프레미스와 Amazon EC2 모두에 호스트되는 Oracle 데이터베이스에 적용할 수 있습니다.

---

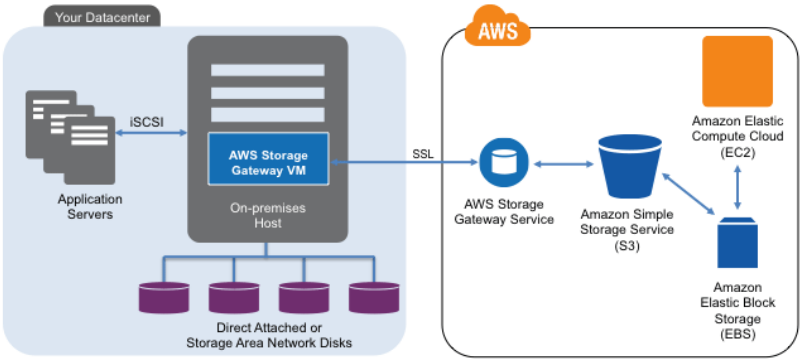
[12] http://aws.amazon.com/vpc/

Amazon S3에 온 프레미스 백업 전송

많은 백업 소프트웨어 벤더는 현재 백업 대상으로 Amazon S3을 지원합니다 (예. CommVault Simpana Software Cloud Storage Connector, SecoBackup, 및 Zmanda). 또한, 많은 스토리지 게이트웨이는 기존 백업 소프트웨어와 Amazon S3 스토리지 간의 통합을 제공합니다 (예. Nasuni 및 Riverbed). 이것은 복잡성 및 오프사이트 테이프 관리의 보안 위험을 제거하며, 내구성이 강한 비용 효율적인 오프사이트 백업을 제공하는데 유용합니다.

데이터가 전송된 Amazon S3로의 전용 링크를 제공하기 위해 AWS Direct Connect[13]를 활용할 수도 있습니다. 이는 잠재성을 높은 대역폭과 사설 연결성의 가능성을 선사합니다.

AWS Storage Gateway[14]는 AWS 클라우드 스토리지와 온 프레미스 애플리케이션 사이에 원활한 데이터 마이그레이션을 가능케하며 백업 콘텐츠를 Amazon S3 에 보내는 유용한 방법을 지원합니다. AWS Storage Gateway는 볼륨 데이터를 사용자의 인프라 및 AWS에 지역별로 저장합니다. 스토리지 복제와 더불어 Amazon EBS Snapshot처럼 데이터를 저장하여 데이터를 복원하는데 사용할 수 있고 Amazon EC2 인스턴스에 나타낼 수 있습니다. 따라서 복원 프로세스가 효율적이며 반복 가능합니다.



백업 생성 관리 및 보안 관리

지속적으로 백업을 실행할 경우, 효율적인 백업 로테이션 전략을 구사하여 스토리지 비용을 줄이고 비즈니스 요구 조건에 따라 올바른 데이터 버전이 유지되도록 하는 것이 중요합니다. 백업 로테이션 프로토콜에 대한 자세한 논의는 이 문서에서 다뤄지지 않습니다.

프로토콜 측면에서, 민감한 데이터의 경우에는 백업 프로세스의 일부로 데이터 보관중 및 전송중에 암호화해야 합니다. 백업 로테이션 및 암호화 필요 요건에 대한 흥미로운 솔루션은 s3napback 도구입니다 http://dev.davidsoergel.com/trac/s3napback/.

Amazon RDS를 이용하면 자동으로 백업을 생성하고 최대 8일까지 유지하며, 단 몇 초만에 과거 5분에 대한 DB Instances의 복원을 가능하게 합니다.

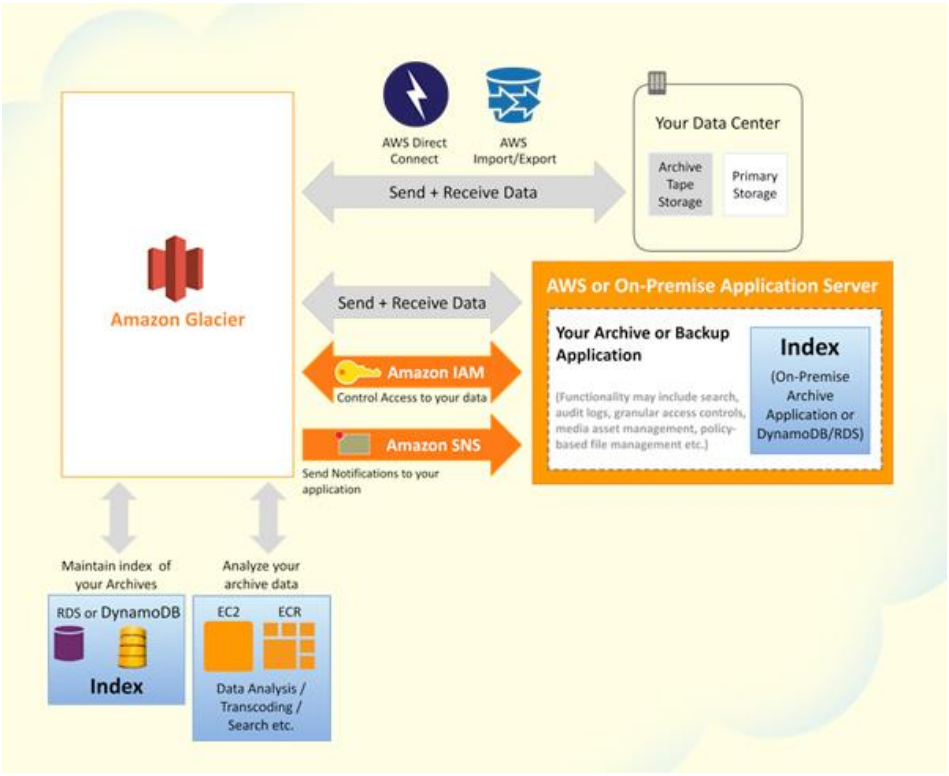
---

[13] http://aws.amazon.com/directconnect/
[14] http://aws.amazon.com/storagegateway

장기적인 데이터 저장

많은 고객들이 디지털 정보를 필요할 때면 언제든지 검색할 수 있는 포맷으로 장시간 (예. 7년, 21년, 환자의 일생 또는 막연한 기간) 저장하고자 합니다. 이는 대규모 (지속적으로 증가하는) 정보의 저장을 내구성, 경제성, 낮은 유지 비용 측면 살펴야 하는 과제를 낳았습니다. Amazon Glacier[15] 서비스는 강한 내구성 (즉, 연평균 99.999999999%의 내구성을 제공하도록 설계)을 바탕으로 고객들이 효율적이고 안정적이며 낮은 비용으로 무제한의 아카이빙 데이터를 장기간 저장할 수 있도록 설계되었습니다. 언제든지 데이터를 검색할 수 있으며, 즉각적이기 보다는 단 3 시간-5 시간 내에 찾도록 선택할 수 있습니다. 이로 인해 비용 효율적인 장기간 스토리지 및 실시간에 가까운 데이터 검색이라는 두가지 목표 (자주 충돌하는)를 모두 달성할 수 있습니다.

Amazon Glacier에서 데이터는 아카이브로 저장되어 Amazon Glacier에 업로드되고 볼트로 정리되며, 고객은 AWS Identity and Access Management(IAM)[16] 서비스를 이용해 액세스를 통제할 수 있습니다. 일정을 정하여 데이터를 검색할 수 있으며, 일반적으로 3 시간-5 시간 내에 완료됩니다.



Amazon Glacier는 Amazon S3, AWS 스토리지 및 데이터베이스 서비스등의 다른 AWS 서비스와 원활히 통합합니다. Amazon S3를 사용하여 Glacier에 자동으로 데이터를 아카이브 (및 검색)할 수명 주기 규정을 생성할 수 있습니다.[17]

고객은 Amazon Glacier를 현재 사용하는 백업 및 아카이브 도구 및 프로세스에 통합할 수 있으며, 어떠한 데이터라도 장시간 보관할 수 있는 새로운 스토리지 계층을 대표합니다. 또한, 기존의 테이프 기반 아카이브를

[15] http://aws.amazon.com/glacier
[16] http://aws.amazon.com/iam
[17] http://docs.aws.amazon.com/AmazonS3/latest/dev/object-archival.html

사용하는 경우, 이를 Amazon cier로 (AWS Import/Export 서비스를 사용)[18] 이전할 수 있으며, 물리적 디바이스는 AWS로 배송하여 관련 Amazon Glacier 볼트로 직접 전송될 수 있습니다.

결론

AWS 플랫폼은 인프라 구성에 대한 새롭고 보다 유연한 옵션을 제공하며, 기업 고객이 훨씬 더 효율적이고 경제적인 백업 및 복원 체제를 갖추도록 지원합니다. 현재의 레거시 접근 방식에서 최첨단 "코드로서의 인프라" 접근 방식으로의 변화에서 특정 프로세스 및 절차가 진화함 따라, 백업 인프라의 비용 및 복잡성을 줄이면서 애플리케이션의 백업 및 복원을 실현할 수 있습니다.

참고 문헌

1. 백업 및 스토리지 웹페이지 - https://aws.amazon.com/backup-storage/

2. Oracle Secure Backup Cloud Module을 이용해 Oracle Databases를 Amazon S3에 백업하는 방법 단계별 동영상 - https://aws.amazon.com/backup-storage/gsg-oracle-rman/

[18] http://aws.amazon.com/importexport