

대규모 컴퓨팅 환경을 구축하려면 상당한 시간과 에너지가 필요합니다. 다음과 같은 몇 가지 사항을 고려해야 합니다.

- 설계 또는 구현, 어디에 노력을 투입할 것인가? 또한 수동 구현의 위험은 무엇인가?
- 어떻게 프로덕션 서버를 업데이트할 것인가? 어떻게 배포를 여러 지리적 리전에 둘아웃할 것인가? 장애가 발생할 경우 어떻게 롤백을 관리할 것인가?
- 배포에 대한 디버깅은 어떻게 할 것인가? 어떻게 오류를 발견하고 수정하며 수정을 확인할 것인가?
- 어떻게 다양한 시스템과 및 하위 시스템에 대한 종속성을 관리할 것인가?
- 마지막으로, 이 모든 작업을 수동으로 할 수 있는가?



**수동으로 개별 구성 요소를 생성하여 환경에 추가할 경우 확장이 불가능합니다.**  
여러분이 대규모 기업 애플리케이션을 담당하고 있는 경우 수동으로 이러한 작업을 처리할 인력은 충분하지 않습니다.

아키텍처 및 애플리케이션을 처음부터 생성하면 내재된 버전 관리가 없습니다. 비상시, 프로덕션 스택을 이전 버전으로 롤백하는 것이 유용하지만, 수동으로 환경을 생성할 경우 이것이 불가능합니다.

감사 추적 기능은 많은 규정 준수 및 보안 상황에서 매우 중요합니다. 사람들이 수동으로 환경을 제어하고 편집하도록 허용하는 것은 위험합니다.

마지막으로, 위험을 최소화하려면 일관성이 매우 중요합니다. 자동화는 일관성을 유지할 수 있게 해줍니다.

## 일반적으로



프로덕션 환경에서 뭔가를 **수동으로**  
변경해야 하는 경우,  
**위험**이 따릅니다.

수동 프로세스는 보상 없는 **위험**입니다.

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



## 인프라 자동화

**aws training and certification**

The diagram shows the AWS CloudFormation logo (a green cube icon) inside a green-bordered box. To its right, a document icon is associated with the text "AWS 인프라를 설명하는 공통 언어를 제공합니다". Below it, a compass icon is associated with the text "설명된 리소스를 자동화된 방식으로 생성하고 구축합니다". A large watermark reading "zlagusdb.com" diagonally across the slide.

AWS CloudFormation은 리소스를 안전하고 반복 가능한 방식으로 프로비저닝하므로, 수동 작업을 수행하거나 사용자 지정 스크립트를 작성할 필요 없이 인프라와 애플리케이션을 구축 및 재구축할 수 있습니다.

AWS CloudFormation을 사용하면 인프라를 코드로 취급할 수 있습니다. 원하는 코드 편집기를 사용하여 코드를 작성하고, GitHub 또는 AWS CodeCommit과 같은 버전 관리 시스템에 체크인하고, 적절한 환경에 배포하기 전에 팀원들과 파일을 검토할 수 있습니다.

# 어떻게 작동합니까?

**AWS CloudFormation 템플릿**

- 생성할 리소스를 설명하는 **JSON/YAML** 형식 파일
- 소스 코드로 취급**: 리포지토리에 저장

```
graph LR; CodeEditor[코드 편집기] --- JSONBox[JSON]; CodeEditor --- YAMLBox[YAML]
```

JSON

```
{ "Resources" : { "HelloBucket" : { "Type" : "AWS::S3::Bucket" } } }
```

YAML

```
Resources: HelloBucket: Type: AWS::S3::Bucket
```

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

### AWS CloudFormation 템플릿에 대한 추가 설명:

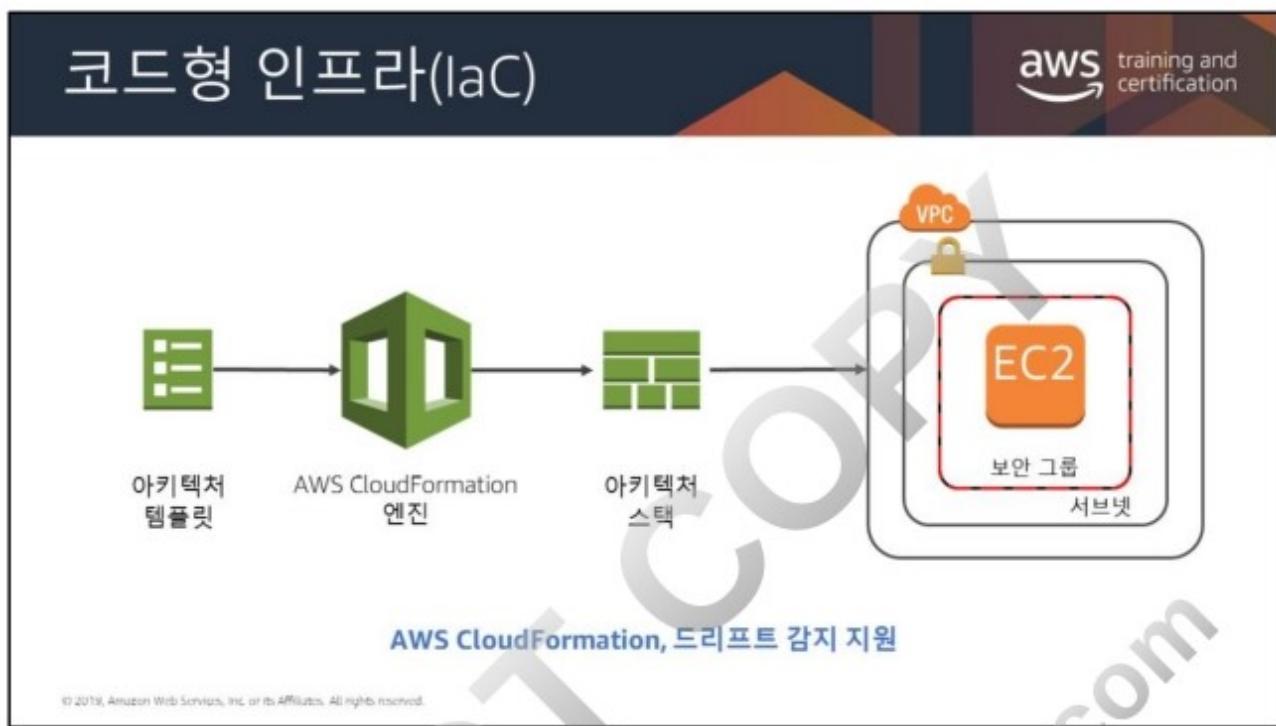
- 템플릿을 코드로 취급하고, 원하는 버전 제어 방법(예: Git, SVN 등)으로 이를 관리합니다.
- JSON 템플릿 파일 내에 전체 애플리케이션 스택(고객 애플리케이션에 필요한 모든 리소스)을 정의합니다.
- 템플릿에 대한 런타임 파라미터를 정의합니다(Amazon EC2 인스턴스 크기, Amazon EC2 키 페어 등).

이제 YAML 형식 템플릿을 생성하여 AWS CloudFormation에서 AWS 리소스 및 속성을 설명할 수 있습니다. 이제, YAML 형식 템플릿 또는 JSON 형식 템플릿을 사용해 AWS 인프라를 모델링하고 설명할 수 있는 옵션이 있습니다. YAML 형식의 AWS CloudFormation 템플릿은 기존의 JSON 형식 템플릿과 동일한 구조를 따르고 동일한 기능을 모두 지원합니다.

또한 한 스택의 출력을 다른 스택과 공유할 수 있는 교차 스택 참조를 만들 수도 있습니다. 이렇게 하면 IAM 역할, VPC 정보, 보안 그룹 등을 공유할 수 있습니다. 이전에는 이렇게 하려면 AWS CloudFormation 사용자 지정 리소스를 사용해야 했습니다. 이제 새로운 ImportValue 내장 함수를 사용해 간단히 한 스택에서 값을 내보내고 다른 스택에서 값을 가져올 수 있습니다.

교차 스택 참조는 AWS 인프라를 스택(예: 네트워크 스택, 애플리케이션 스택 등) 기준으로 그룹화된 논리적 구성 요소로 분리하는 고객, 그리고 중첩 스택의 대안으로 스택을 느슨하게 결합해야 하는 고객에게 유용합니다.

DO NOT COPY  
zlagusdbs@gmail.com



AWS CloudFormation을 사용하면 Amazon Web Services (AWS) 리소스 세트를 템플릿을 제출하는 것만큼 간단하게 배포할 수 있습니다.

템플릿은 특정 환경에 배포될 리소스를 설명 및 정의하는 텍스트 파일입니다.

AWS CloudFormation은 템플릿을 처리하는 엔진입니다. AWS CloudFormation의 출력은 스택이라고 합니다.

스택은 그룹으로 함께 배포되는 AWS 리소스의 모음입니다.

AWS CloudFormation 템플릿에 대한 추가 설명:

- 이를 코드로 취급하고 버전 관리 시스템을 사용하여 관리할 수 있습니다.
- JSON 또는 YAML 템플릿 파일 내에 전체 애플리케이션 스택(고객 애플리케이션에 필요한 모든 리소스)을 정의합니다.
- 템플릿의 런타임 파라미터를 정의합니다(Amazon EC2 인스턴스 크기, Amazon EC2 키 페어 등).

AWS CloudFormation이 지원하는 리소스 목록은

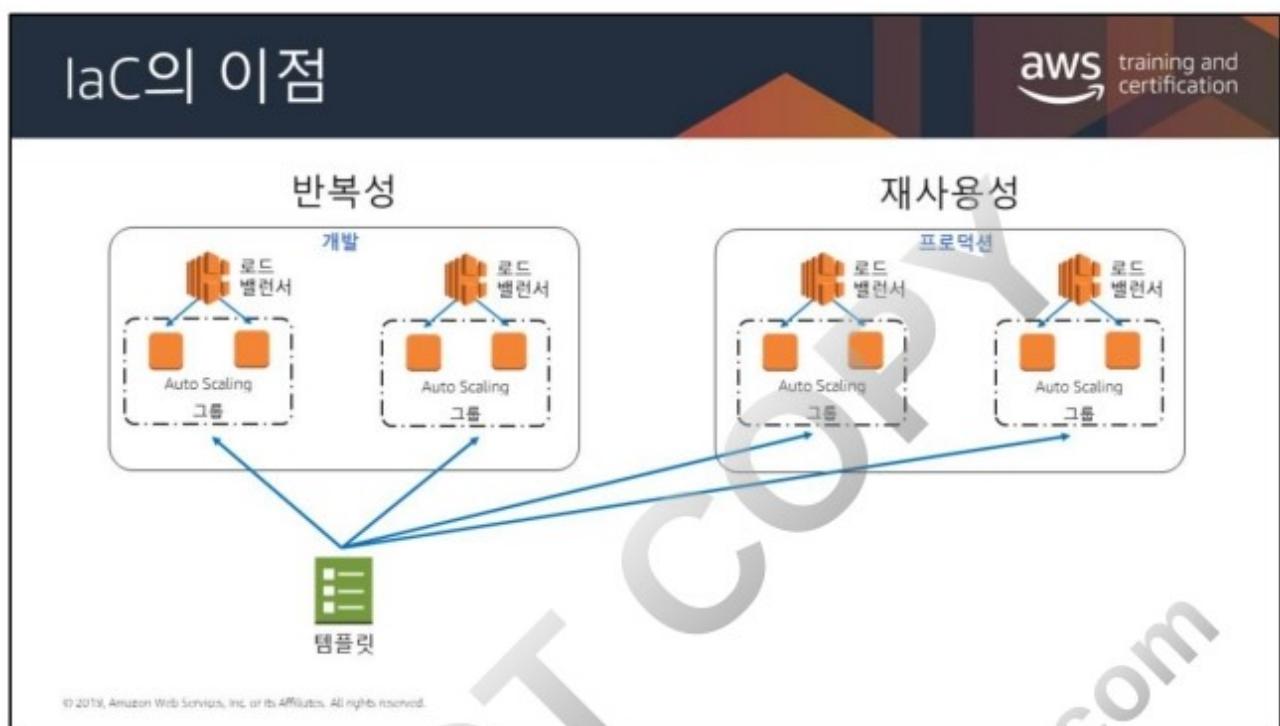
<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html>을 참조하십시오.

### 샘플 템플릿은

<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-sample-templates.html>을 참조하십시오.

스택에서 드리프트 감지 작업을 수행하면 스택이 소기의(기존) 템플릿 구성에서 드리프트되었는지 확인하고, 드리프트 감지를 지원하는 스택에 있는 각 리소스의 드리프트 상태 관련 세부 정보를 반환합니다. “현재 스택의 드리프트 감지” 대화 상자를 클릭하면 스택에서 드리프트 감지를 활성화할 수 있습니다. 대화 상자를 닫고 드리프트 세부 정보를 나중에 검토하는 경우에도 드리프트 감지 프로세스가 계속됩니다. 자세한 내용은 다음 사이트를 참조하십시오.

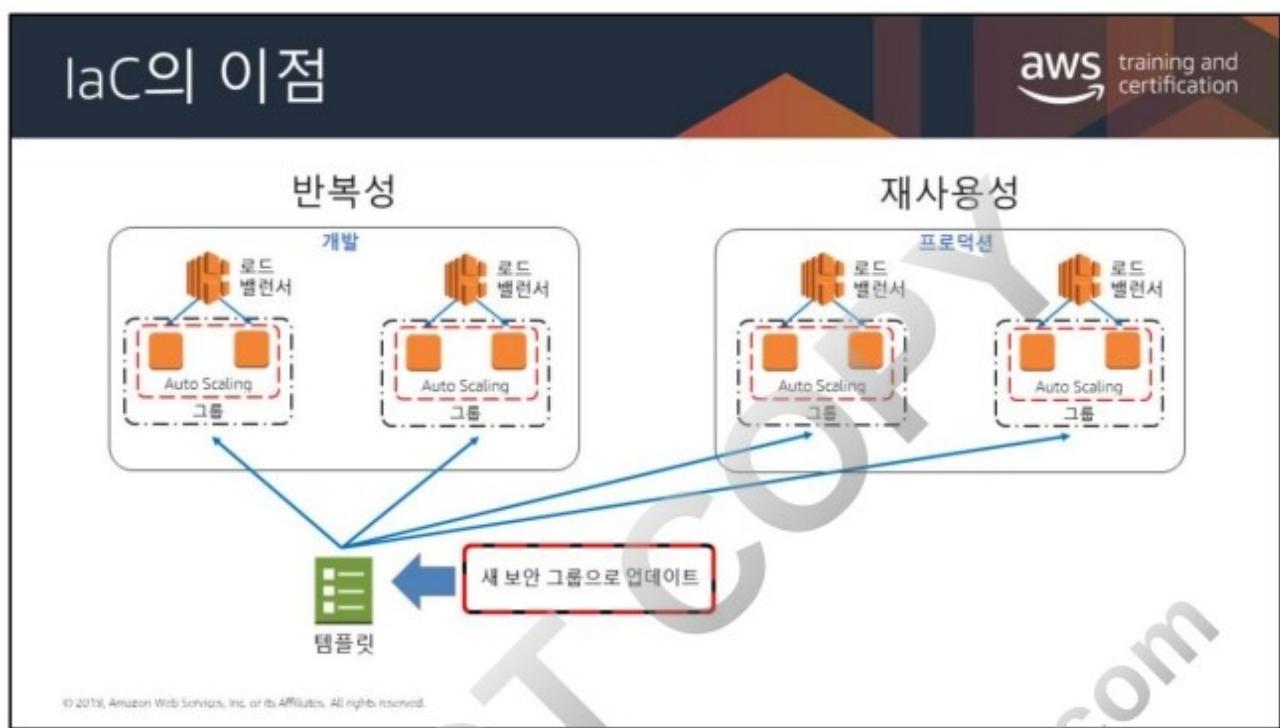
- <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/detect-drift-stack.html>
- <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-stack-drift.html>



인프라를 코드형으로 구축하는 경우, 환경을 구축하면서 반복성과 재사용성의 이점을 활용할 수 있습니다.

템플릿 하나(또는 템플릿의 조합)로 복잡한 동일 환경을 반복해서 구축할 수 있습니다.

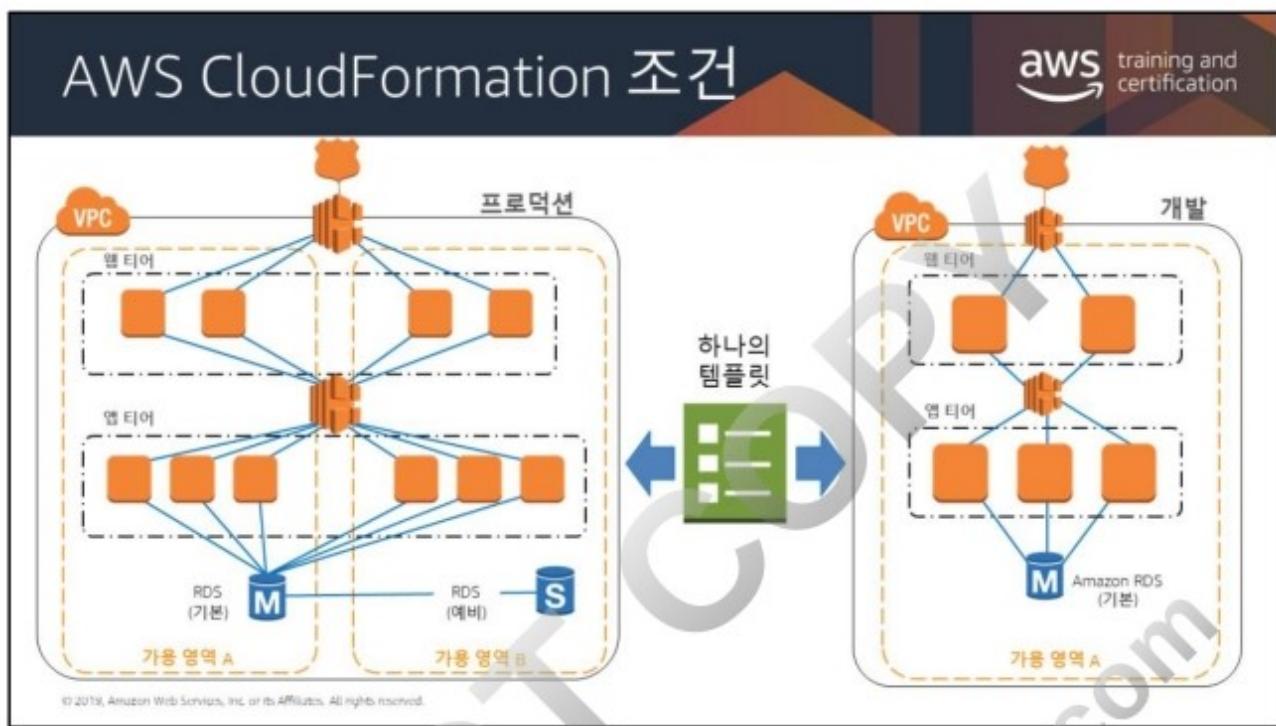
AWS에서 이를 사용하는 경우, 조건에 따라 다른 환경을 생성할 수도 있습니다. 즉 생성했던 환경의 컨텍스트에 맞게 환경이 구축되도록 할 수 있습니다. 예를 들어 템플릿이 개발 환경 또는 프로덕션 환경에서 시작되었는지에 따라 서로 다른 AMI가 사용되도록 템플릿을 설계할 수 있습니다.



이 시나리오에서는 인스턴스 스택에 새로운 보안 그룹을 추가하도록 템플릿이 업데이트되었습니다.

이러한 환경을 시작하는 데 사용된 템플릿을 하나만 변경하면, 모든 네 개의 환경에 새로운 보안 그룹 리소스가 추가됩니다.

이 기능은 리소스의 간편한 유지 관리성뿐만 아니라 뛰어난 일관성과 병렬화를 통한 필요한 작업 감소라는 이점을 제공합니다.



프로덕션 환경과 개발 환경을 동일한 스택에서 구축해야 합니다. 이렇게 해야 애플리케이션이 프로덕션에서도 설계 및 개발된 방식대로 작동합니다.

또한 개발 환경과 테스트 환경에서 동일한 스택을 사용해야 합니다. 모든 환경이 동일한 애플리케이션 및 구성을 갖게 됩니다.

기능 테스트, 사용자 승인 테스트 및 로드 테스트를 위해 여러 테스트 환경이 필요할 수 있습니다. 이러한 환경을 수동으로 생성하면 큰 위험이 수반됩니다.

AWS CloudFormation 템플릿에서 조건 문을 사용해 개발, 테스트 및 프로덕션이 크기 및 범위는 다르지만 나머지는 동일하게 구성되도록 할 수 있습니다.



스택을 업데이트하는 한 가지 방법은 기존 템플릿을 편집한 후 다시 실행하는 것입니다.

그러나 사용자가 AWS CloudFormation이 스택을 업데이트할 때 수행할 변경에 대한 추가 통찰이 필요할 경우 변경 세트를 사용할 수 있습니다.

변경 세트를 사용하면 업데이트가 진행되기 전에 변경 사항을 미리 보고 변경 사항이 예상과 일치하는지 확인한 후 업데이트를 승인할 수 있습니다.

다음은 변경 세트를 사용하는 기본 워크플로우입니다.

1. 업데이트하려는 스택의 변경 사항을 제출하여 변경 세트를 생성합니다.
2. 변경 집합을 보고 어떤 스택 설정과 리소스가 변경될지 확인합니다.
3. 어떻게 변경할지 결정하기 전에 다른 변경 사항을 고려하려면 추가 변경 집합을 만듭니다.
4. 변경 집합을 실행합니다. AWS CloudFormation이 이러한 변경 사항을 사용하여 스택을 업데이트합니다.

자세한 내용은

<http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-changesets.html>를 참조하십시오.

DO NOT COPY  
zlagusdbs@gmail.com

## 설계 예

### 계층화된 아키텍처

프런트 엔드	CRM 웹 인터페이스, 관리자 인터페이스, 분석 대시보드
백엔드	고객, 캠페인, 제품, 마케팅 자료, 분석
공유	CRM DB, 일반 모니터링/경보, 서브넷, 보안 그룹
기본 네트워크	VPC, 인터넷 게이트웨이, VPN, NAT
자격 증명	IAM 사용자, 그룹, 역할

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

# AWS Quick Start

aws training and certification

## 표준화된 템플릿

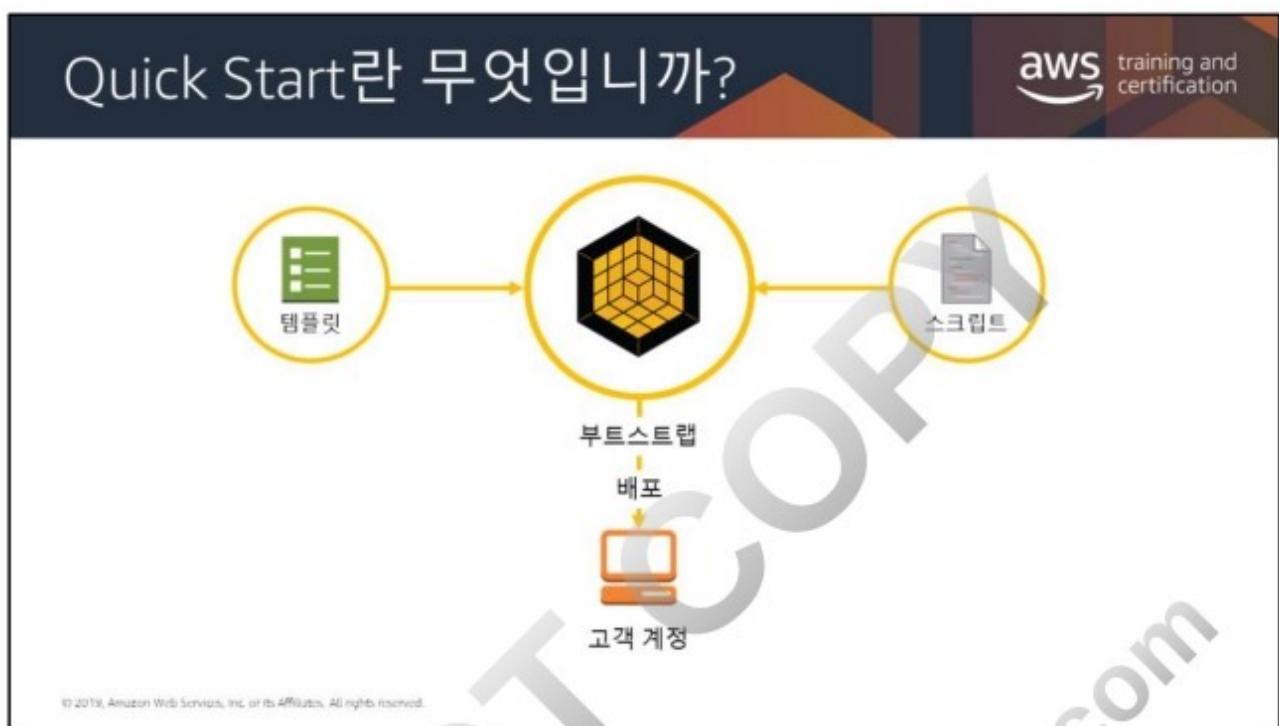


- AWS 클라우드에서의 모범 표준 배포
- AWS 보안 및 고가용성 모범 사례에 기초
- 클릭 한 번으로 1시간 내에 전체 아키텍처 생성
- 실험 및 간편한 구축에 적합

AWS 솔루션스 아키텍트가 구축한  
AWS CloudFormation 템플릿

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Quick Start는 고객이 AWS에서 인기 있는 솔루션을 배포하는 데 활용할 수 있도록 AWS 솔루션스 아키텍트 및 파트너가 보안 및 고가용성 관련 AWS 모범 사례를 기반으로 구축합니다. 이러한 참조 배포는 AWS 클라우드에서 자동으로 주요 기술을 구현하는데, 흔히 한 번의 클릭으로 한 시간이 채 걸리지 않습니다. 몇 단계를 통해 테스트 또는 프로덕션 환경을 구축하여 즉시 사용을 시작할 수 있습니다.

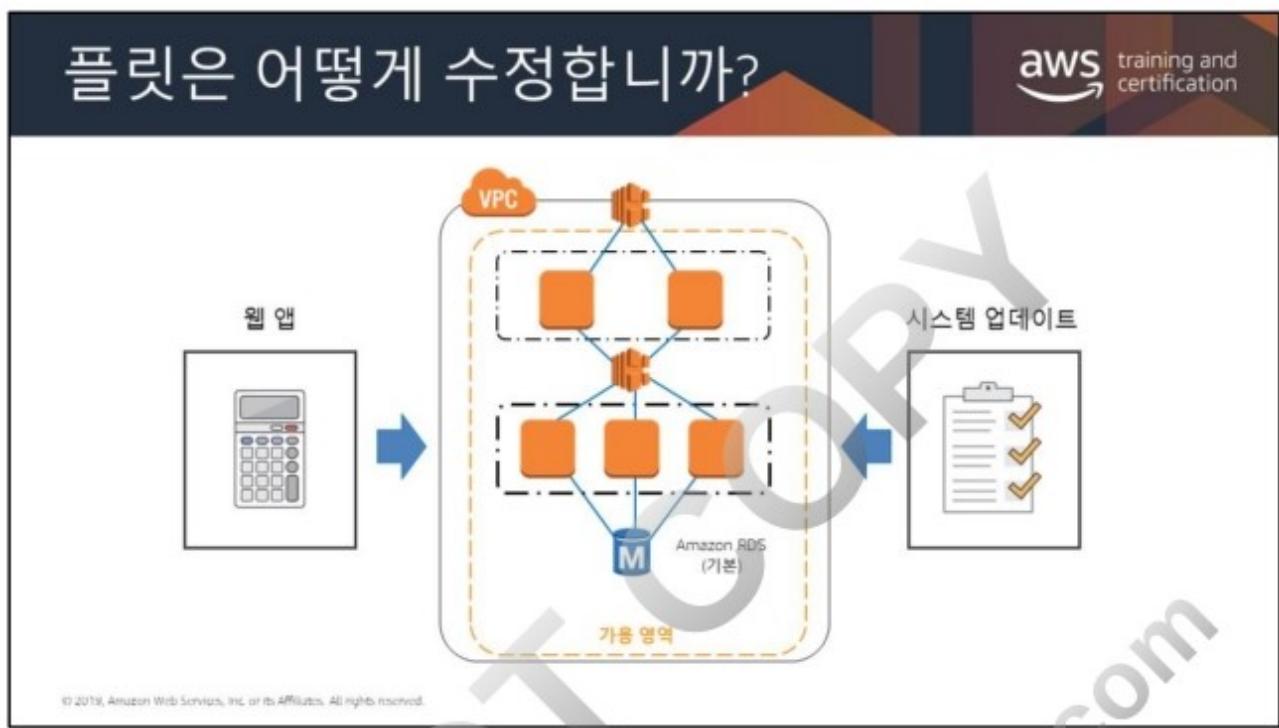


이 Quick Start는 AWS 계정에서 환경을 생성하기 위한 AWS CloudFormation 템플릿 및 관련 스크립트로 구성됩니다. 모든 부트스트래핑 및 배포를 사용자 대신 처리합니다. 또한 모든 구성 요소가 어떻게 생성되었는지 보여주는 배포 안내서가 제공됩니다.

이 환경을 생성 및 실행하는 데 사용된 리소스의 요금이 부과됩니다.

자세한 내용은 다음을 참조하십시오. <https://aws.amazon.com/quickstart/>





AWS CloudFormation을 사용하여 전체 인프라를 자동으로 생성할 수 있습니다.  
하지만 여전히 몇 가지 중요한 고려 사항이 있습니다.

어떻게 Amazon EC2 인스턴스를 업데이트할 것인가? 각 상자에 로그인하여 직접 명령을 업데이트 해야 하는가? 최신 버전의 웹 앱을 다운로드하는가? 오류가 발생할 경우 어떻게 변경 사항을 되돌리는가? 3가지 앱을 실행하는 서버가 100개라면 어떻게 할 것인가?

이러한 시나리오에 도움이 될 수 있는 기존의 도구들이 있지만, 즉시 사용 가능한 솔루션이 더 편리할 것입니다.

# Systems Manager

aws training and certification



자동화된 구성 및 대규모 시스템의 지속적 관리가 가능한 기능의 집합

- 모든 Windows 및 Linux 워크로드
- Amazon EC2 또는 온프레미스에서 실행

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Systems Manager는 소프트웨어 인벤토리 수집, OS 패치 적용, 시스템 이미지 생성, Windows 및 Linux 운영 체제 구성 등 자동으로 수행할 수 있는 관리형 서비스입니다. 이러한 기능을 활용해 시스템 구성은 정의 및 추적하고, 드리프트를 방지하고, Amazon EC2 및 온프레미스 구성의 소프트웨어 규정 준수를 유지할 수 있습니다. AWS Systems Manager는 클라우드의 규모 및 민첩성을 고려하여 설계되었지만 온프레미스 데이터 센터로 확장되는 관리 접근 방식을 제공하므로, 기존 인프라를 AWS와 더 쉽고 완벽하게 연결할 수 있습니다.

AWS Systems Manager는 Amazon EC2 콘솔에서 열 수 있습니다. 관리할 인스턴스를 선택한 후 수행할 관리 작업을 정의합니다. AWS Systems Manager는 무료로 제공되며, Amazon EC2 리소스와 온프레미스 리소스를 모두 관리할 수 있습니다.

## 무엇을 할 수 있습니까?

The diagram illustrates five key features of AWS Systems Manager:

- 명령 실행 (Command)
- 패치 관리 (Patch Manager)
- 세션 관리자 (Session Manager)
- 유지 관리 기간 (Change Set)
- 인벤토리 (Inventory)

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Systems Manager Run Command를 사용하여 대규모 관리형 인스턴스의 구성을 원격으로 안전하게 관리합니다. Run Command를 사용하여 수십 또는 수백 개의 대상 인스턴스 집합에서 애플리케이션 업데이트 또는 Linux 셸 스크립트 및 Windows PowerShell 명령 실행과 같은 온디맨드 변경을 수행합니다.

자세한 내용은 다음을 참조하십시오. <https://docs.aws.amazon.com/systems-manager/latest/userguide/execute-remote-commands.html>

Patch Manager를 사용하여 관리형 인스턴스의 패치 적용 프로세스를 자동화합니다. 이 기능을 사용하면 인스턴스를 스캔하여 패치 누락 여부를 확인하고 누락된 패치를 개별적으로 적용하거나 Amazon EC2 태그를 사용하여 대규모 인스턴스 그룹에 적용할 수 있습니다. 보안 패치의 경우, Patch Manager가 승인 및 거부된 패치 목록뿐 아니라 릴리스 후 며칠 이내에 패치를 자동 승인하는 규칙을 포함하는 패치 기준선을 사용합니다. 보안 패치는 해당 인스턴스에 대해 구성된 패치 기본 리포지토리로부터 설치됩니다. Systems Manager 유지 관리 기간 작업으로 실행되도록 패치 적용을 예약하여 보안 패치를 정기적으로 설치할 수 있습니다. Linux 운영 체제의 경우 사용자가 패치 기준선의 일부로 패치 적용 작업에 사용할 리포지토리를 정의할 수 있습니다.

따라서 인스턴스에서 어떤 리포지토리가 구성되어 있는지 상관없이 업데이트가 신뢰할 수 있는 리포지토리로부터만 설치되도록 할 수 있습니다. Linux의 경우, 운영 체제 보안 업데이트로 분류된 업데이트뿐 아니라 인스턴스의 모든 패키지를 업데이트할 수도 있습니다.

자세한 내용은 다음을 참조하십시오. <https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-patch.html>

Maintenance Windows를 사용해 관리형 인스턴스가 비즈니스 크리티컬 작업을 중단하지 않고 패치 및 업데이트 설치와 같은 반복적 관리 작업을 실행하는 일정을 설정합니다.

자세한 내용은 다음을 참조하십시오. <https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-maintenance.html>

Systems Manager 상태 관리자를 사용하여 관리형 인스턴스를 정의된 상태로 유지하는 프로세스를 자동화합니다. 상태 관리자를 사용하여 시작 시 인스턴스가 특정 소프트웨어로 부트스트랩되거나 Windows 도메인에 조인되거나(Windows 인스턴스만 해당) 특정 소프트웨어 업데이트로 패치되도록 할 수 있습니다.

자세한 내용은 다음을 참조하십시오. <https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-state.html>

세션 관리자를 사용하여 대화형 원클릭 브라우저 기반 셀 또는 AWS CLI를 통해 Amazon EC2 인스턴스를 관리합니다. 세션 관리자는 인바운드 포트를 열거나 접속 호스트를 유지하거나 SSH 키를 관리할 필요 없이 안전하고 감사 가능한 인스턴스 관리를 제공합니다. 또한 세션 관리자를 사용하면 인스턴스 액세스 제어, 엄격한 보안 관행, 인스턴스 액세스 세부 정보가 포함된 전면 감사가 가능한 로그를 요구하는 기업 정책을 쉽게 준수하면서 최종 사용자에게 Amazon EC2 인스턴스에 대한 원클릭 교차 플랫폼 액세스를 제공할 수 있습니다.

자세한 내용은 다음을 참조하십시오. <https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager.html>

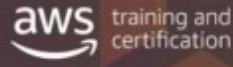
인벤토리를 사용하면 Amazon EC2 및 온프레미스 컴퓨팅 환경에 대한 가시성을 확보할 수 있습니다. 인벤토리를 사용하여 관리형 인스턴스에서 메타데이터를 수집할 수 있습니다.

자세한 내용은 다음을 참조하십시오. <https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-inventory.html>

[www.flaticon.com](http://www.flaticon.com)의 [smalllikeart](#)가 만든 세션 관리자 아이콘  
[www.flaticon.com](http://www.flaticon.com)의 [wanicon](#)이 만든 인벤토리 아이콘

DO NOT COPY  
zlagusdbs@gmail.com

## 인프라 및 배포 자동화를 위한 AWS OpsWorks



구성 관리 서비스

- AWS OpsWorks Stacks
- AWS OpsWorks for Chef Automate
- AWS OpsWorks for Puppet Enterprise



© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS OpsWorks Stacks는 Chef를 사용하여 모든 형태와 규모의 애플리케이션을 구성하고 운영하도록 지원하는 구성 관리 서비스입니다. 애플리케이션의 아키텍처 및 각 구성 요소의 사양을 정의할 수 있습니다. 구성 요소에는 패키지 설치, 소프트웨어 구성 및 리소스(예: 스토리지)가 포함됩니다. 애플리케이션 서버 및 데이터베이스 같은 일반적인 기술을 위한 템플릿에서 시작할 수도 있고, 스크립팅 가능한 작업을 수행하도록 자체적으로 구축할 수도 있습니다. AWS OpsWorks Stacks에는 시간 또는 부하를 기반으로 애플리케이션을 조정하는 자동화 기능과 환경이 조정됨에 따라 변경 사항을 조정하는 동적 구성이 포함되어 있습니다.

AWS OpsWorks for Chef Automate는 완전 관리형 Chef Automate 서버뿐 아니라 지속적 배포를 위한 워크플로 자동화, 규정 준수 및 보안을 위한 자동 테스트, 노드와 노드 상태를 볼 수 있는 사용자 인터페이스를 제공하는 자동화 도구 세트를 제공합니다. Chef Automate 플랫폼은 소프트웨어 및 운영 체제 구성, 지속적 규정 준수, 패키지 설치, 데이터베이스 설정 등의 운영 작업을 처리하여 전체 스택 자동화를 제공합니다. Chef 서버는 중앙에서 구성 작업을 저장하고, 노드 몇 개부터 수천 개까지 규모에 상관없이 컴퓨팅 환경에 있는 각 노드에 이러한 작업을 제공합니다. OpsWorks for Chef Automate는 Chef 커뮤니티의 도구 및 툴과 완벽히 호환되며, Chef 서버에 새로운 노드를 자동으로 등록합니다.

AWS OpsWorks for Puppet Enterprise는 관리형 Puppet Enterprise 서버뿐 아니라 조정을 위한 워크플로 자동화, 자동 프로비저닝, 추적 가능성을 위한 시각화를 제공하는 자동화 도구 세트를 제공합니다. Puppet Enterprise 서버는 소프트웨어 및 운영 체제 구성, 패키지 설치, 데이터베이스 설정 등의 운영 작업을 처리하여 풀 스택 자동화를 제공합니다. Puppet 마스터는 중앙에서 구성 작업을 저장하고, 노드 몇 개부터 수천 개까지 규모에 상관없이 컴퓨팅 환경에 있는 각 노드에 이러한 작업을 제공합니다.

DO NOT COPY  
zlagusdbs@gmail.com

OpsWorks Stacks에는 수명 주기 이벤트가 있습니다.

aws training and certification

다음 트리거에서 스크립트를 실행할 수 있습니다.

Setup은 새 인스턴스가 성공적으로 부팅된 후 새 인스턴스에서 발생합니다.

Configure는 인스턴스가 온라인 상태에 진입하거나 온라인 상태에서 나갈 때 스택의 모든 인스턴스에서 발생합니다.

Deploy는 앱을 배포할 때 발생합니다.

Undeploy는 앱을 삭제할 때 발생합니다.

Shutdown은 인스턴스를 중지할 때 발생합니다.

AWS OpsWorks Stacks

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

DO NOT  
zlagusdbs@gmail.com

## AWS CloudFormation과 함께 AWS OpsWorks Stacks 사용하기

AWS CloudFormation을 사용하여 인프라(VPC, IAM 역할)를 구축하고, AWS OpsWorks Stacks를 사용하여 애플리케이션 계층을 배포합니다.

PHP 애플리케이션 스택

- 로드 밸런싱 계층
- 확장 가능한 PHP 앱 계층
- Amazon RDS 계층

경보

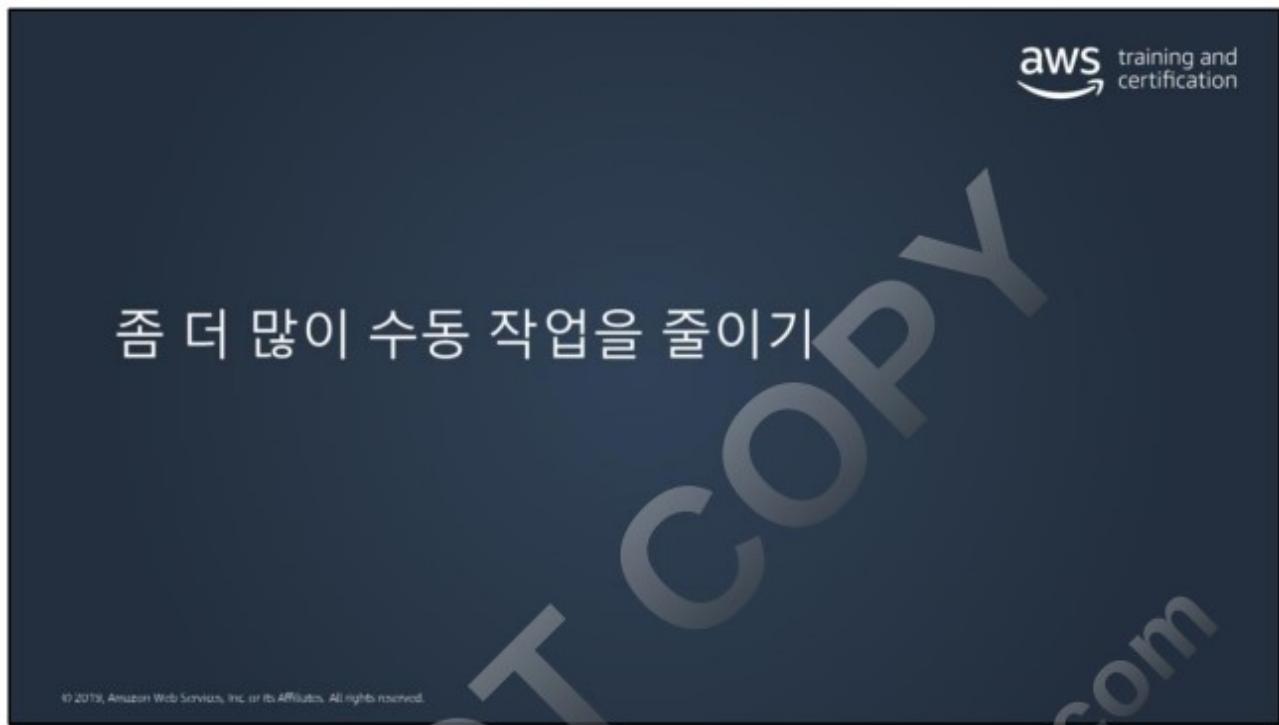
로그

Amazon VPC

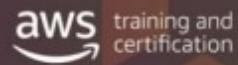
IAM 권한

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS OpsWorks Stacks는 AWS CloudFormation을 통해 생성할 수 있으므로 두 기술을 동시에 사용할 수 있습니다. 계층화된 AWS CloudFormation 템플릿 세트를 사용할 수 있습니다. 즉, 한 템플릿으로 환경의 인프라를 생성하고(예: Amazon VPC, IAM 역할, 외부 애플리케이션과의 통신용 Amazon SQS 대기열), 별도의 AWS CloudFormation 템플릿을 사용하여 해당 인프라 내에 배포되는 AWS OpsWorks Stacks 스택을 생성합니다.



## 일반적 문제



- 앱 배포에 관련된 인프라 관리는 어려울 수 있습니다
- 서버 관리와 구성에 많은 시간이 걸릴 수 있습니다
- 여러 프로젝트나 애플리케이션에서 일관성 부족

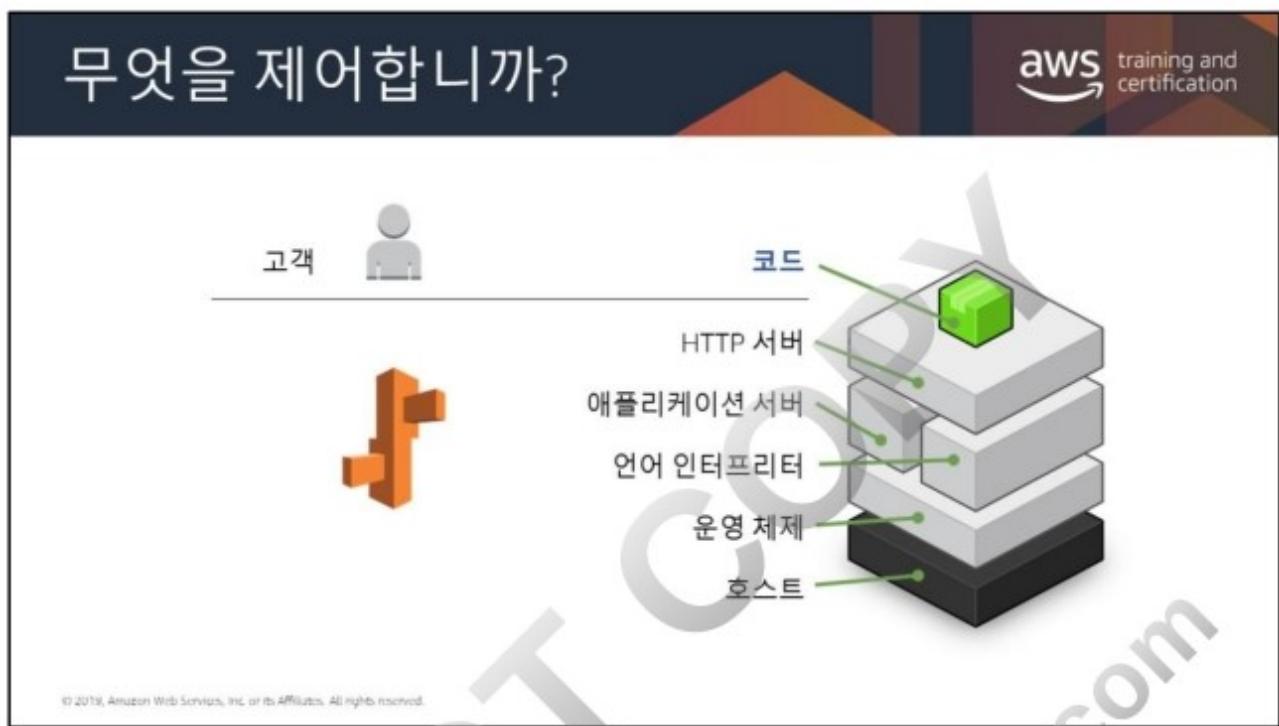
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



The image shows the AWS Elastic Beanstalk landing page. At the top left is the title "AWS Elastic Beanstalk". To the right is the "aws training and certification" logo. On the left side, there is a large orange square containing the AWS logo (a stylized orange 'F'). Below the logo, the text "AWS Elastic Beanstalk" is displayed. To the right of the logo, there are three bullet points in Korean:

- 인프라를 프로비저닝 및 운영하고 사용자를 위해 애플리케이션 스택을 관리
- 완전한 투명성 - 생성되는 모든 것을 확인할 수 있습니다
- 적절한 규모를 유지합니다. 애플리케이션의 크기를 자동으로 늘리거나 줄입니다

At the bottom left, there is a small copyright notice: "© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved."



AWS Elastic Beanstalk의 목표는 개발자가 기본 인프라에 대해 걱정할 필요 없이 클라우드에 확장 가능한 웹 애플리케이션 및 서비스를 배포하고 유지 관리하도록 돋는 것입니다. Elastic Beanstalk은 환경 내 각 EC2 인스턴스를 선택된 플랫폼에서 애플리케이션을 실행하는 데 필요한 구성 요소로 구성합니다. 애플리케이션 스택을 설치 및 구성하기 위해 인스턴스에 로깅하는 것에 대해 걱정할 필요가 없습니다.

## Elastic Beanstalk - 환경

Elastic Beanstalk는 필요한 인프라 리소스를 프로비저닝합니다

Elastic Beanstalk는 애플리케이션 환경에 고유한 도메인 이름을 제공합니다(예: [http://\[your app\].elasticbeanstalk.com](http://[your app].elasticbeanstalk.com))

- Route 53을 사용하여 사용자 고유의 도메인 이름을 이 도메인 이름으로 확인할 수 있습니다

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Elastic Beanstalk으로 작업 시 두 가지 유형의 환경을 선택할 수 있습니다. 단일 인스턴스 환경은 단일 EC2 인스턴스를 시작할 수 있지만 로드 밸런싱 또는 Auto Scaling이 포함되지 않습니다. 다른 유형의 환경은 여러 EC2 인스턴스를 시작할 수 있지만 로드 밸런싱 및 Auto Scaling 구성이 포함됩니다.

Elastic Beanstalk은 ELB, Auto Scaling 그룹, 보안 그룹, 데이터베이스(선택 사항) 등과 같이 필요한 인프라 리소스를 프로비저닝합니다.



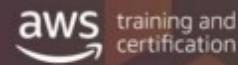
자주 하는 질문 중 하나가 애플리케이션 관리를 제공하는 다양한 서비스와 이러한 서비스를 구분하는 것에 관한 것입니다. 고객이 필요한 간편성 및 제어의 수준에 따라 달라집니다.

AWS Elastic Beanstalk는 널리 사용되는 컨테이너인 Java, PHP, Node.js, Python, Ruby 및 Docker로 웹 애플리케이션을 구축할 수 있는 사용이 간편한 애플리케이션 서비스입니다. 코드를 업로드하길 원하고, 환경을 사용자 정의할 필요가 없다면, Elastic Beanstalk가 적합합니다.

AWS OpsWorks를 사용하면 애플리케이션을 시작하고 애플리케이션의 아키텍처 및 각 구성 요소의 사양을 정의할 수 있습니다. 구성 요소에는 패키지 설치, 소프트웨어 구성 및 리소스(예: 스토리지)가 포함됩니다. 앱 서버, 데이터베이스 등과 같은 일반 기술용 템플릿을 사용하거나 자체 템플릿을 작성할 수 있습니다.



## 실습 5: 인프라 배포 자동화



"일관되고 반복 가능한 방식으로 인프라를 배포하고 싶습니다."

### 사용된 기술:

- AWS CloudFormation

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

## 실습 5: 인프라 배포 자동화



계층에 인프라를 배포합니다.

- 네트워크 계층
- 애플리케이션 계층

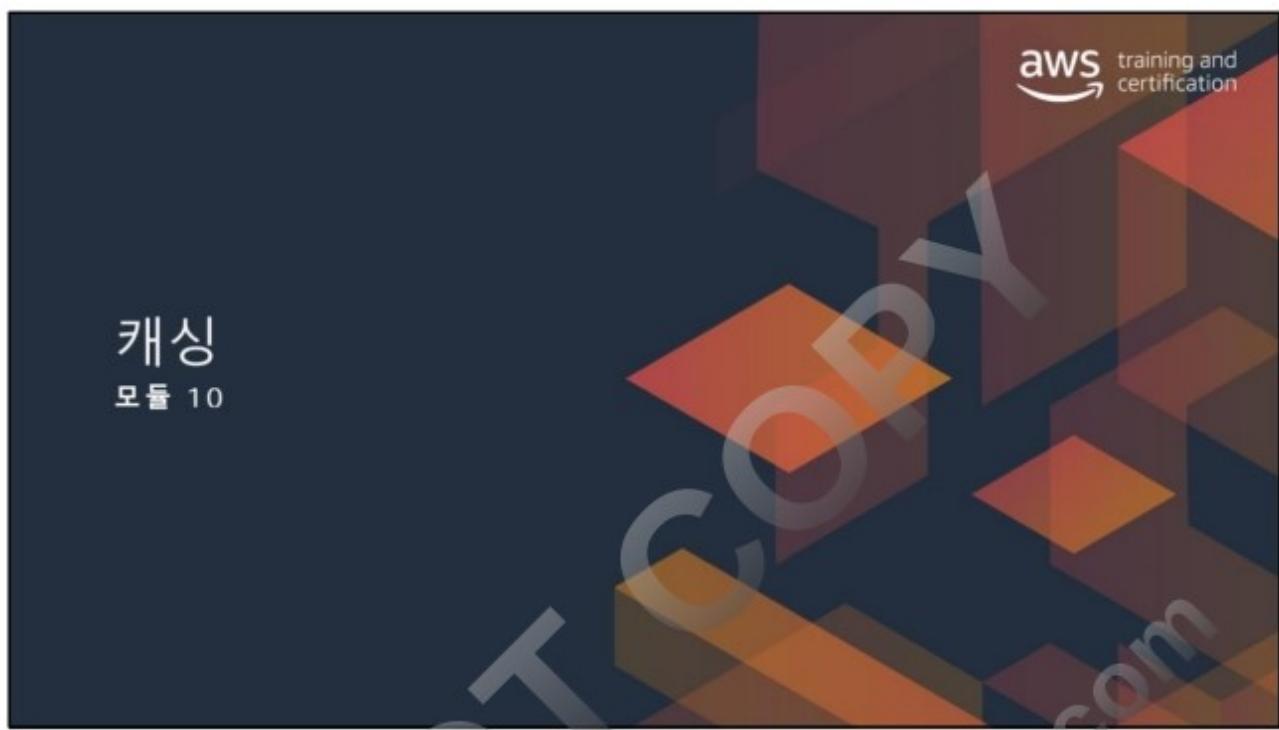
추가 작업:

- 스택 업데이트
- 삭제 정책이 있는 스택 삭제

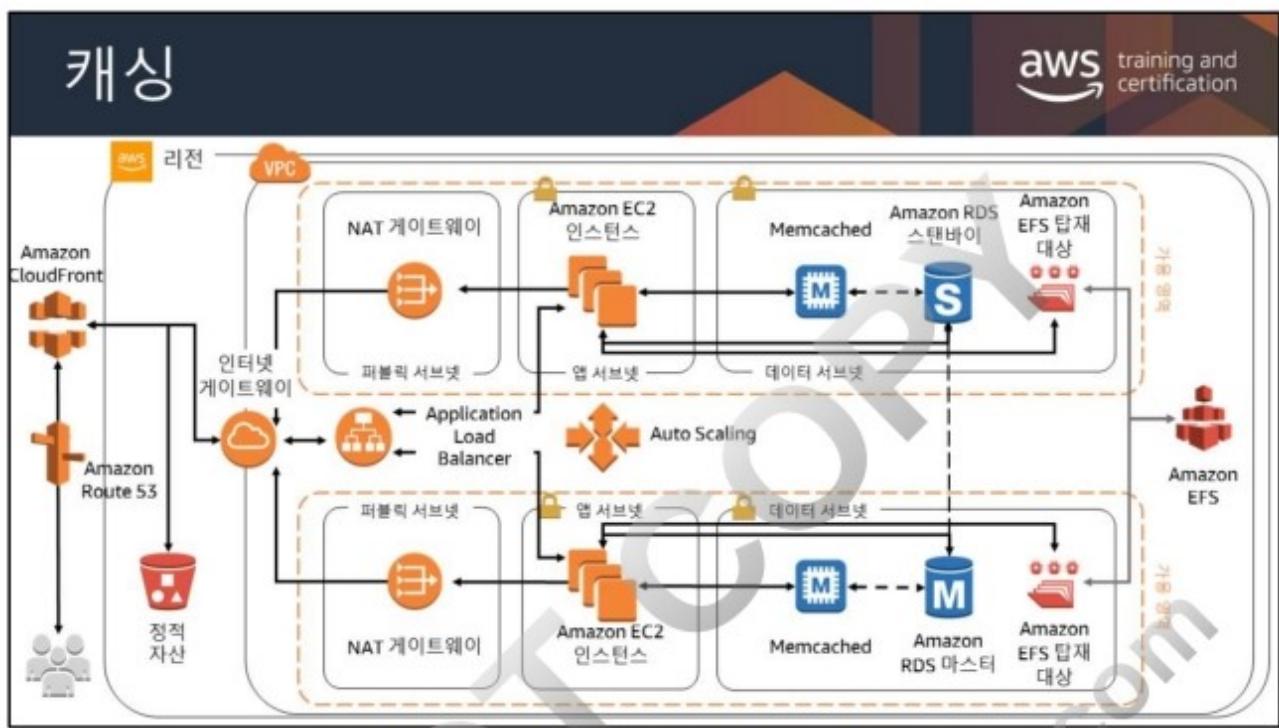
시간: 30분

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.





DO NOT COPY  
zlagusdbs@gmail.com



수업이 끝나면 이 아키텍처 디어그램의 모든 구성 요소를 이해할 수 있습니다.  
또한 규모가 크고 견고한 자체 아키텍처 솔루션을 구성할 수도 있습니다.

## 모듈 10



### 아키텍처 측면에서의 필요성

동일한 요청으로 인프라 용량이 지속적으로 과부하됩니다. 이는 비효율적으로 비용 및 지연 시간을 늘립니다.

#### 모듈 개요

- 캐싱 개요
- 엣지 캐싱
- 데이터베이스 캐싱

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.





캐시가 어떻게 성능을 향상시키는지를 설명하기 위해 철물점으로 이동하는 것을 고려해 보시기 바랍니다.

철물점이 수 마일 떨어져 있다면 뭔가 필요할 때마다 그 곳에 가기 위해 상당한 노력을 기울일 것입니다.



그 대신 공구 창고(캐시)가 가까운 곳에 있다면 여러분이 필요한 소모품으로 공구 창고를 채우면 됩니다. 그렇다면 이제 뭔가 필요할 때 철물점으로 이동하는 대신, 그냥 여러분의 공구 창고로 가면 됩니다.

그러나 저장한 내용물(stockpile)을 새로운 것으로 바꿔야 할 때, 철물점은 항상 옵션에 있습니다.

## 무엇을 캐시해야 합니까?

aws training and certification

-  수집하려면 느리고 비싼 쿼리가 필요한 데이터
-  비교적 정적이고 자주 액세스하는 데이터(예: 소셜 미디어 웹 사이트의 프로필)
-  공개 거래되는 주식 가격처럼 일정 기간 동안 변화가 없을 수 있는 정보

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

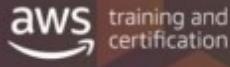
**속도 및 비용** – 데이터베이스에서 데이터를 획득하는 것은 캐시에 비해 언제나 더 많은 시간과 비용이 듭니다. 일부 데이터베이스 쿼리는 본래부터 다른 데이터베이스 쿼리에 비해 더 많은 시간과 비용이 듦니다. 예를 들면, 여러 테이블에서 조인을 수행하는 쿼리는 단순한 단일 테이블 쿼리보다 훨씬 더 많은 시간과 비용이 듦니다. 관심이 있는 데이터를 획득하기 위해 시간과 비용이 많이 드는 쿼리가 필요할 경우, 이러한 쿼리는 캐싱 후보에 속합니다. 데이터를 획득하기 위해 비교적 빠르고 간단한 쿼리가 필요할 경우, 이러한 쿼리는 그 밖의 요인에 따라 여전히 캐싱 후보가 될 수 있습니다.

**데이터 및 액세스 패턴** – 캐싱할 항목을 결정할 때에도 데이터 그 자체와 데이터의 액세스 패턴을 이해해야 합니다. 예를 들면, 변화 속도가 빠르거나 액세스가 거의 없는 데이터는 캐싱할 필요가 없습니다. 캐싱을 통해 유의미한 이점을 얻으려면 소셜 미디어 사이트의 개인 프로필과 같이 비교적 정적이면서도 액세스빈도가 높은 데이터가 있어야 합니다. 이와는 반대로, 캐싱을 해도 속도나 비용 면에서 이득이 없다면 데이터를 캐싱할 필요가 없습니다. 예를 들면, 검색 결과를 반환하는 웹 페이지를 굳이 캐싱할 필요는 없습니다. 그 이유는 그러한 쿼리 및 결과가 거의 항상 고유한 것이기 때문입니다.

**기한 경과** – 기본적으로 캐싱된 데이터는 기한이 지난 데이터입니다. 이러한 데이터는 특정 상황에서 기한이 경과하지 않을 경우에도 항상 기한이 경과한 것으로 간주 및 취급해야 합니다. 사용 중인 데이터가 캐싱 후보인지 여부를 결정할 때 기한이 경과한 데이터에 대한 애플리케이션의 내결함성을 결정해야 합니다. 사용 중인 애플리케이션의 경우, 하나의 컨텍스트에서 기한이 경과한 데이터를 허용할 수 있더라도 다른 컨텍스트에서는 그럴 수 없습니다.

예를 들면, 공개적으로 거래되는 주식의 가격 정보를 웹 사이트 상에서 제공하는 경우, 가격 공개는 최대  $n$  분까지 지연될 수 있다는 면책 요건을 전제로 하여 기한 경과를 허용할 수 있습니다. 그러나 주식을 매매하는 중개인(브로커)에게 동일한 주식에 대한 가격 정보를 제공할 때에는 실시간 데이터가 필요합니다.

## 캐싱의 이점



The slide features three icons illustrating the benefits of caching:

- 애플리케이션 속도 향상**: Represented by a blue cloud icon containing a computer monitor displaying a chart.
- 시간이 많이 걸리는 DB 쿼리의 부담 완화**: Represented by a purple gear icon with a green triangle in the center.
- 응답 지연 시간 감소**: Represented by a hand pointing at a yellow button with a light effect around it.

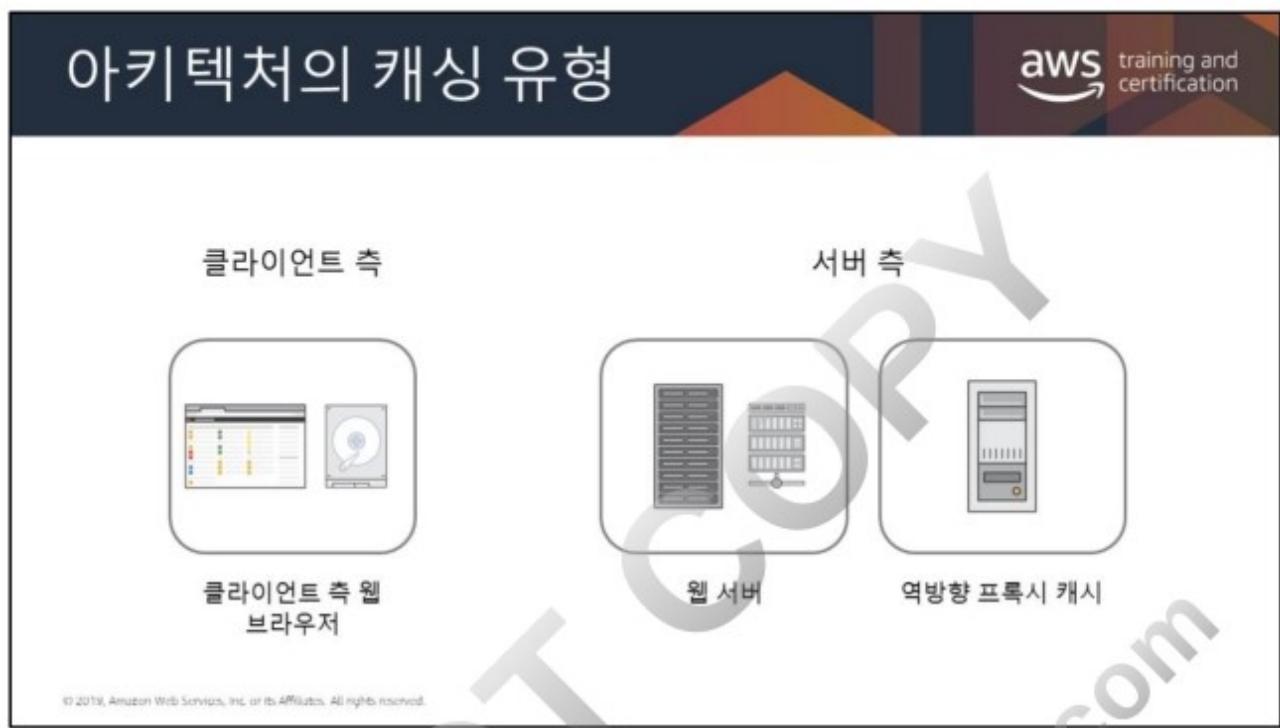
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

캐시는 메모리에 데이터를 저장하여 흔히 액세스하는 애플리케이션 데이터에 대해 높은 처리량, 지연 시간이 짧은 액세스를 제공합니다. 캐싱은 애플리케이션의 속도를 높일 수 있습니다. 캐싱은 애플리케이션 사용자가 경험하는 응답 지연 시간을 줄입니다. 시간 소모적인 데이터베이스 쿼리와 복잡한 쿼리는 애플리케이션에 병목 현상을 일으키는 경우가 많습니다. 읽기 집약적인 애플리케이션에서 캐싱은 애플리케이션 처리 시간과 데이터베이스 액세스 시간을 줄임으로써 유의한 수준의 성능 향상을 제공합니다.

쓰기 집약적인 애플리케이션은 대체로 캐싱에서 큰 효과를 보지 못합니다. 하지만 쓰기 집약적인 애플리케이션도 보통 읽기/쓰기 비율이 1보다 큽니다. 즉, 읽기 캐싱은 여전히 유용하다는 것을 나타냅니다.

데이터 캐싱을 고려해야 할 경우를 열거하면 다음과 같습니다.

- 캐시 검색에 비해 데이터 획득에 더 많은 시간과 비용이 드는 경우
- 충분한 빈도로 데이터에 액세스하는 경우
- 데이터가 비교적 정적인 경우 또는 신속한 변화와 기한 경과가 큰 문제에 속하지 않는 경우



컴퓨팅에서 캐시는 대체로 일시적인 성격의 데이터 하위 집합을 저장하는 고속 데이터 스토리지 계층에 속하기 때문에 향후 그러한 데이터에 대한 요청은 해당 데이터의 기본 스토리지 위치를 액세스할 때보다 더 신속하게 처리됩니다. 캐싱을 이용하면 이전에 검색하거나 계산된 데이터를 효과적으로 재사용할 수 있습니다.

웹 캐싱은 오리진 서버보다는 오히려 캐시로부터 향후 요청을 이행하기 위해 캐시에서 HTTP 응답 및 웹 리소스를 보관함으로써 진행됩니다.

웹 캐시는 다양한 기술에서 효과적으로 활용할 수 있습니다. 가장 기본적인 레벨은 클라이언트 측 웹 캐싱입니다. 데이터는 반복된 쿼리를 웹 서버에 전달하기보다는 오히려 브라우저 내에 저장됩니다. HTTP 캐시 헤더는 저장된 웹 콘텐츠에 대한 캐시로부터 향후 응답을 브라우저가 얼마나 오랫동안 이행할 수 있는지에 관한 세부 정보를 제공합니다.

서버 측에서는 다양한 웹 캐싱 기법을 활용해 웹 사이트 성능을 향상시킬 수 있습니다.

역방향 프록시 캐시 또는 웹 애플리케이션 액셀러레이터는 캐시된 버전의 HTTP 응답을 보관된 상태에서 제공하기 위해 애플리케이션 및 웹 서버의 앞쪽에 배치할 수 있습니다. 이러한 캐시들은 사이트 관리자가 구현하며, 브라우저와 오리진 서버 간의 중개자 역할을 담당합니다. 또한 이러한 캐시들은 흔히 HTTP 캐시 지시문을 기반으로 합니다.

DO NOT COPY  
zlagusdbs@gmail.com



