에서 제품형 인프라(IaP) 구현 AWS

AWS 권장 가이드

AWS 권장 가이드: 에서 제품형 인프라(IaP) 구현 AWS

Table of Contents

AWS에서 IaP 구현

Kirsten Kissmeyer, Amazon Web Services(AWS)

2023년 1월(문서 기록)

이 가이드에서는 AWS 제품형 인프라(IaP)를 관리하기 위한 접근 방식을 살펴봅니다. IaP는 코드형 인프라(IaC)보다 높은 수준의 추상화 및 제어를 제공하지만 IaC 메서드를 사용하여 목표를 달성합니다. 또한 IaP 관리를 위한 AWS 서비스 및 도구를 살펴보고 각 도구가 인프라 관리 목표를 지원하는 방법을 중점적으로 다룹니다. 이 가이드의 정보는 대규모 금융 부문 회사를 위한 AWS Service Catalog 지원 이니셔티브에서 학습한 내용을 기반으로 합니다.

이 가이드는 여러 조직 사용자, 사업부 및 서드 파티에서 필요에 따라 쉽게 할당하고 권한을 부여할 수 있는 기능별 AWS 클라우드 인프라 서비스를 개발하려는 사용자를 대상으로 합니다.

인프라를 제품으로 관리하는 이유는 무엇인가요?

인프라 리소스를 제품으로 관리하면 소비자 기능을 표준화된 정의 및 구성을 보유한 리소스 세트로 패키징할 수 있다는 이점이 있습니다. 제품은 조직이 AWS 기능을 할당하고 사용하는 방법을 관리하고 제어할 수 있는 편리한 방법을 제공합니다. 제품은 지정된 조직 단위(OU) 또는 이러한 기능 역량이 필요한 개인으로만 제한될 수 있습니다. 제품은 특정 AWS 리전으로 제한될 수도 있습니다.

또한 제품 프로비저닝 모델을 사용하면 중앙 위치에서 제품의 정의를 캡슐화하고 업데이트할 수 있습니다. 그런 다음 시간이 지남에 따라 구현이 변경되면 일회성 또는 예약된 방식으로 제품 업데이트를 배포할 수 있습니다.

목표 비즈니스 성과

조직은 항상 AWS 인프라를 관리하고 프로비저닝하는 더 나은 방법을 모색합니다. 목표에는 다음이 포함될 수 있습니다.

- 높은 수준의 민첩성, 신뢰성, 내결함성 및 중앙 집중식 제어를 달성하여 단일 구성 지점이 진화하는 내부 및 외부 표준 준수 충족.
- 특정 팀 또는 개인에게 필요한 경우 셀프 서비스 액세스를 허용하면서 인프라를 중앙 집중식으로 분산하는 로우 터치 또는 푸시 버튼 메커니즘.
- 내부 직원, 클라이언트 계정 및 파트너 OU 계정에 AWS 인프라 및 서비스를 프로비저닝하는 기능. 특정 리전의 특정 인프라 구성 요소에 액세스할 수 있는 OU 또는 조직을 제어할 수도 있습니다.

- 서드 파티 도구(예: ServiceNow) 또는 사용자 지정 도구를 사용하여 엔터프라이즈 자산 및 인프라에 액세스하고 프로비저닝하기 위한 요청을 관리하는 경우 AWS 인프라와 이러한 도구 사이를 쉽게 통합할 수 있습니다.
- 수십 또는 수백 개의 대상 계정에 AWS 인프라를 동시에 프로비저닝하는 기능.
- 단일 기능을 제공하기 위해 여러 AWS 리소스를 프로비저닝하기 위한 지원.
- 빠듯한 일정 내에 필요한 인프라를 사용하여 새 계정을 생성하는 기능.
- 프로비저닝한 인프라의 인벤토리에 대한 액세스 및 인프라 구성 요소를 업데이트하거나 제거하는 기능.
- 프로비저닝 및 유지 관리 프로세스를 더 쉽고 빠르며 안전하고 신뢰할 수 있도록 지원하는 접근 방식과 기술.

AWS Service Catalog 를 사용하여 IaP 관리

AWS 는 AWS 인프라를 제품으로 관리하고 프로비저닝AWS Service Catalog하는 것을 지원하는 라는 서비스를 제공합니다. Service Catalog를 사용하여 제품 세트로 프로비저닝해야 하는 인프라를 빠르게 정의하고, 원하는 당사자에게 해당 제품에 대한 권한을 부여하며, 개별 제품에 필요한 프로비저닝 및 업데이트 패턴을 구현할 수 있습니다.

Service Catalog는 AWS CloudFormation에 의해 지원됩니다. Service Catalog 포트폴리오, 제품 및 프로비저닝 템플릿은 CloudFormation 스택으로 관리됩니다. 다음과 같은 네 가지 방법으로 이러한 스택을 정의할 수 있습니다.

- 표준 CloudFormation 템플릿 사용.
- 선호하는 지원되는 프로그래밍 언어와 함께 AWS Cloud Development Kit (AWS CDK) 및 Service Catalog Construct Library 사용.
- 스택을 설명하는 선언적 메타데이터에서 CloudFormation 스택 정의를 생성하기 위해 서드 파티 도구에서 제공하는 프레임워크 사용.
- Service Catalog API 사용. 이 API는 제품 빌드를 제외한 모든 기능에 대한 메서드를 제공합니다. 포트폴리오에 제품을 추가하고, 포트폴리오에서 제품을 제거하며, 제품 및 포트폴리오에 태그를 지정하고, 관리 및 운영 제품 서비스 작업을 정의하며, 포트폴리오 및 제품 정의를 찾아 검색할 수 있습니다.

핵심에서 Service Catalog 제품은 집합적이고 사용자 지정 가능한(파라미터화를 통해) 기능을 제공하도록 구성된 하나 이상의 AWS 리소스 집합입니다. 예를 들어 Service Catalog 제품을 정의하여 대상 계정에서 프라이빗 Amazon Simple Storage Service(Amazon S3) 버킷을 프로비저닝할 수 있습니다. S3 버킷은 버킷 이름, 액세스를 허용할 인터넷 주소 범위, 버킷에 액세스할 수 있는 사용자 세트, 수명 주기 계층화 정책 또는 버킷 버전 관리 사양과 같은 입력 파라미터를 포마할 수 있는 제품입니다. AWS Identity and Access Management (IAM) 역할을 정의하여 제품의 일부로 버킷에 대한 액세스를 제공할 수도 있습니다.

Service Catalog 제품을 하나 이상의 포트폴리오에 추가할 수 있습니다. Service Catalog 포트폴리오는 일반적으로 비슷한 목적(예: 분석, 개발, 클라이언트 액세스 서비스, 파트너 액세스 서비스 등)을 지원하기 때문에 함께 그룹화된 제품 모음입니다.

포트폴리오 수준에서 제품을 프로비저닝할 수 있는 액세스 권한을 사용자, 그룹 또는 역할에 제공합니다. 프로비저닝을 위해 제품은 시작 IAM 역할(역할을 수임할 수 있는 모든 사용자에게 셀프 서비스 방식으로 제품을 시작하는 경우) 또는 제품을 프로비저닝할 수 있는 하나 이상의 계정을 정의하는 스택

세트와 연결됩니다. 스택 세트를 사용하려면 Service Catalog 허브 계정에서 Service Catalog 관리자 역할을 정의하고 스택 세트의 각 대상 계정에서 Service Catalog 제품 프로비저닝 실행 역할을 정의해야 합니다.

다음 섹션에서는 Service Catalog IaP 기능을 더 자세히 설명합니다.

주제

- 모듈성 및 코드 재사용 지원
- Service Catalog에서 제품을 정의하기 위한 프로그래밍 옵션
- 외부 프로비저닝 프로세스 및 워크플로와 통합
- 제품 프로비저닝 사양
- DevSecOps 수명 주기 지원
- 사용자 지정된 재사용 및 계정별 프로비저닝
- Service Catalog 제품 리소스를 애플리케이션으로 정의 및 관리
- 재고 관리

모듈성 및 코드 재사용 지원

다양한 AWS 리소스 또는 다른 제품에서 제품을 조합할 수 있습니다. 여러 제품에 재사용할 수 있도록 리소스를 모듈식으로 정의하는 것이 가장 이상적입니다. 리소스 수준 재사용을 사용하면 해당 리소스 유형을 사용하는 모든 제품에서가 아니라 한 곳에서 향후 변경을 수행할 수 있습니다.

Service Catalog는 제품 수준에서 재사용성을 지원하기 위해 체이닝이라는 기능을 제공합니다. 한 제품을 하나 이상의 다른 제품에 체이닝할 수 있습니다. 예를 들어 S3 로깅 버킷 제품을 상위 수준 모니터링 제품에 연결할 수 있습니다. 체이닝은 모듈성을 지원하지만 종속성을 관리해야 하기 때문에 몇 가지 운영상 복잡성이 부과됩니다. Service Catalog는 체인닝된 제품 간 버전 관리를 자동으로 유지하지 않으므로 한 제품에 대한 변경 사항이 해당 제품에 종속된 다른 제품을 변경하지 않도록 보장할 수 없습니다. 체이닝은 신중하게 사용하고, 버전 관리를 보장하고 종속성을 유지하기 위한 자체 메커니즘을 개발합니다.

Service Catalog는 기본적으로 CloudFormation을 사용하여 제품 프로비저닝 템플릿을 CloudFormation 스택으로 배포합니다. 그러나 Service Catalog는 제품 스택의 CloudFormation 배포에 몇 가지 제한 사항을 부과합니다. 특히 Service Catalog 프로비저닝은 재사용 가능한 스크립트 세그먼트를 삽입하거나 중첩된 CloudFormation 스크립트(또는 스택)를 둘 이상의 수준으로 참조하기 위해 CloudFormation `include` 매크로를 지원하지 않습니다. 이러한 서비스 카탈로그 제한 사항은 재

사용 가능한 CloudFormation 템플릿 또는 구성 요소에서 제품을 정의하는 기능을 제한합니다. 이는 CloudFormation에서 스택을 기본적으로 정의할 때 표준 모범 사례입니다.

> **Note**
>
> Service Catalog를 사용하면 이러한 CloudFormation 구문을 사용하는 프로비저닝 템플릿을 사용해 제품을 성공적으로 정의할 수 있습니다. 그러나 `include` 매크로를 사용하거나 Service Catalog CloudFormation 템플릿에 여러 수준의 스크립트를 중첩하는 경우 프로비전 시간 오류가 발생합니다.

이러한 제한 사항으로 인해 Service Catalog에서 재사용 가능한 모듈식 제품을 구현하기 어려울 수 있습니다. 모듈성이 필요한 경우 AWS CDK를 사용하여 제품 및 프로비저닝 템플릿을 구현하거나 AWS Labs Service Catalog Tools 프로젝트에서 프로비저닝 워크플로 및 엔진을 사용할 수 있습니다. 두 가지 대안 모두 이 가이드의 뒷부분에서 설명합니다.

Service Catalog에서 제품을 정의하기 위한 프로그래밍 옵션

Service Catalog를 사용하여 AWS 인프라를 프로비저닝하기 위한 두 가지 프로그래밍 옵션은 CloudFormation 템플릿 또는 입니다 AWS CDK. 현재 Service Catalog 제품을 정의하기 위한 선언적 또는 노코드 메커니즘은 없습니다.

CloudFormation 스크립팅

CloudFormation 는 인프라를 프로비저닝하기 AWS 위해 시도된 진정한 IaC 네이티브 스크립팅 언어입니다. 에서 AWS Management Console 또는 Visual Studio Code(또는 간단한 텍스트 편집기) 및 () 와 같은 개발 도구를 사용하여 CloudFormation 스크립트를 개발할 수 있습니다 AWS Command Line Interface AWS CLI.

자세한 내용은 CloudFormation 설명서를 참조하세요. CloudFormation 템플릿을 사용하여 Service Catalog 제품을 지정하는 방법에 대한 자세한 내용은 CloudFormation 설명서의 AWS::ServiceCatalog::CloudFormationProduct resource를 참조하세요.

를 사용한 프로그래밍 방식 AWS CDK

는 다양한 프로그래밍 언어를 사용하여 AWS 인프라를 정의하고 유지하기 위한 고급스럽고 강력한 객체 지향 프로그래밍 프레임워크를 AWS CDK 제공합니다. AWS CDK 를 사용하여 객체 지향적이고 세분화된 사용자 지정 및 AWS 클래스 프레임워크 확장을 개발할 수 있습니다. AWS CDK 는 보다 정교

한 인프라 요구 사항에 AWS 서비스 맞게 사용자 지정하고 필요한 프로그래밍 기술과 경험을 갖춘 사용자를 위한 것입니다.

를 사용하여 Service Catalog 솔루션을 구현하려면 기본 제공 Service Catalog 클래스를 AWS CDK사용하여 제품 및 포트폴리오를 정의합니다. 이러한 클래스는 AWS CDK theaws-cdk-lib.aws_servicecatalog 모듈에서 제공합니다.

를 사용하여 여러 가지 방법으로 제품을 구현 AWS CDK 할 수 있습니다. CloudFormation에서 제품에 대한 프로비저닝 템플릿을 작성하고 재사용성을 유지하려면 AWS CDK ProductStack 클래스를 사용하여 프로비저닝 템플릿을 나타내는 것이 좋습니다. `ProductStack` 인스턴스는 프로그래밍 방식으로 리소스를 추가하는 AWS CDK 스택입니다. 예를 들어 S3 버킷, IAM 역할 또는 Amazon CloudWatch 로그를 추가할 수 있습니다. 를 호출하여 정의된 `servicecatalog.CloudFormationProduct` 인스턴스에 `ProductStack` 인스턴스를 프로비저닝 템플릿으로 추가하면가 CloudFormation 템플릿을 `servicecatalog.CloudFormationTemplate.fromProductStack (<ProductStack instance>)` AWS CDK 자동으로 생성합니다.

다음은 Amazon S3 제품에 대한 Java `ProductStack` 구현 예제입니다.

```
import * as s3 from 'aws-cdk-lib/aws-s3';
import * as cdk from 'aws-cdk-lib';

class S3BucketProduct extends servicecatalog.ProductStack {
  constructor(scope: Construct, id: string) {
    super(scope, id);

    new s3.Bucket(this, 'BucketProduct');
  }
}

const product = new servicecatalog.CloudFormationProduct(this, 'Product', {
  productName: "My Product",
  owner: "Product Owner",
  productVersions: [
    {
      productVersionName: "v1",
      cloudFormationTemplate:
 servicecatalog.CloudFormationTemplate.fromProductStack(new S3BucketProduct(this,
 'S3BucketProduct')),
    },
  ],
});
```

는 내장 지속적 통합 및 지속적 배포(CI/CD) 파이프라인을 AWS CDK 제공합니다. 자체 프로세스 표준 및 목표에 맞게 이러한 기본 제공 파이프라인 및 소프트웨어 개발 수명 주기(SDLC) 프로세스를 사용자 지정할 수 있습니다.

사용자 지정 AWS CDK 클래스는 다른 클래스에서 상속하여 특수 함수를 제공할 수 있으며, 클래스는 다른 클래스의 인스턴스로 구성될 수 있습니다. 공유 AWS CDK 클래스 프레임워크를 사용하여 여러 Service Catalog 제품을 구현하는 경우 특히 여러 개발 팀에서 버전 관리 또는 호환성에 미치는 영향을 고려하세요. 변경 사항이 이전 버전과 호환되는지 또는 한 제품의 클래스 변경으로 인해 다른 제품과의 호환을 차단하지 않도록 버전 관리 체계를 따르고 있는지 확인해야 합니다.

자세한 내용은 AWS CDK 설명서를 참조하세요.

외부 프로비저닝 프로세스 및 워크플로와 통합

AWS SDK APIs 또는를 사용하여 Service Catalog 구성 요소와 상호 작용할 수 있습니다 AWS CLI. AWS SDK Service Catalog API를 사용하여 Service Catalog API 직접 호출을 통합할 수 있는 모든 도구에서 Service Catalog 제품을 관리할 수 있습니다. API는 Service Catalog 생성 및 관리의 모든 측면을 포괄합니다. 예를 들어 Terraform은 Launch Wizard에서 AWS SDK Service Catalog API를 호출하여 Service Catalog 제품의 시작(프로비저닝)을 지원합니다. 자세한 내용은 AWS 설명서의 Terraform으로 AWS Service Catalog 제품 시작을 참조하세요.

AWS CLI Service Catalog 명령을 호출하여 Service Catalog에서 작업을 수행할 수도 있습니다. 지원되는 명령에 대한 자세한 내용은 AWS CLI 명령 참조의 servicecatalog를 참조하세요.

제품 프로비저닝 사양

Service Catalog는 CloudFormation 프로비저닝 템플릿에 지정된 리소스의 CloudFormation 스택 세트 배포로 프로비저닝 프로세스를 시작합니다. (템플릿은에서 직접 생성 CloudFormation 하거나 구문에서 생성할 수 있습니다 AWS CDK `ProductStack`.) Service Catalog 제품 프로비저닝은 닫힌 프로세스입니다. 즉, 예비 또는 사후 프로세스 단계를 추가하거나 조정하도록 사용자 지정할 수 없습니다. 그러나 프로비저닝 템플릿을 수정하여 CloudFormation 리소스 사양의 형태로 단계를 추가할 수 있습니다. 예비 단계(예: 프로비저닝 중에 사용되는 접속 호스트를 설정하기 위한 사용자 지정 부트스트래핑) 및 사후 단계(예: 접속 호스트 해제)를 수행하는 AWS Lambda AWS Step Functions또는 Lambda 지원 사용자 지정 리소스일 수 있습니다. 사전 프로비저닝 및 사후 프로비저닝 단계를 구현하는 이 방법에는 프로비저닝 템플릿과 동일한 `include` 및 중첩 스택 제한 사항이 적용됩니다.

대상 계정을 조직 단위(OU)가 아닌 개별 계정으로 지정할 수 있습니다. 사용자 지정 리소스 또는 함수를 작성하여 이 제한 사항을 해결할 수 있습니다. 대부분의 조직은 계정 생성을 자동화하고 계정 목록

을 수동으로 유지 관리하지 않기 때문에 개별 계정이 아닌 OU에 제품 포트폴리오를 프로비저닝합니다.

DevSecOps 수명 주기 지원

현재 Service Catalog CloudFormation 스크립트로 프로비저닝된 제품은 CI/CD 프로세스를 기본적으로 지원하지 않습니다. AWS CodePipeline 또는 기타 DevOps 도구에서 CI/CD 프로세스를 생성하여 개발, 테스트, 단계 및 프로덕션과 같은 수명 주기 환경을 통해 제품을 개발, 테스트 및 릴리스하는 것이 좋습니다.

AWS CDK 는이 가이드의 앞부분에서 설명한 대로 제품에 대한 기본 제공 CI/CD 지원을 제공합니다.

사용자 지정된 재사용 및 계정별 프로비저닝

제품은 최대한 여러 다수의 사용자 지정된 용도로 재사용할 수 있어야 합니다. Service Catalog는 제품 파라미터를 통한 재사용성을 지원합니다. 프로비저닝할 때 이러한 파라미터를 제품에 대한 입력으로 제공할 수 있습니다.

CloudFormation 템플릿 수준에서 이러한 파라미터를 AWS Systems Manager 파라미터 스토어 값으로 지정하여 계정별 및 OU별 값을 적용할 수도 있습니다. 이는 CloudFormation 프로비저닝 템플릿 설계 모범 사례입니다. 대상 계정 내에서 명명된 파라미터 값은 제품이 프로비저닝될 때 적용됩니다. 예를 들어 서브넷 파라미터를 Parameter Store 값으로 지정하고 특정 OU 계정에 대한 제품 프로비저닝 시간에 해당 서브넷을 적용할 수 있습니다. 파라미터 스토어 값에 대한 자세한 내용은 CloudFormation 설명서의 동적 참조를 사용하여 템플릿 값 지정을 CloudFormation 참조하세요.

Service Catalog 제품 리소스를 애플리케이션으로 정의 및 관리

AWS Service Catalog AppRegistry 는 중앙 집중식 애플리케이션 검색, 보고 및 관리 기능을 제공합니다. AppRegistry 애플리케이션에는 하나 이상의 프로비저닝된 제품 스택 및 Service Catalog와 독립된 CloudFormation 스택이 포함될 수 있습니다. 배포 대상으로 정의 AWS 계정 한에서 모든 애플리케이션 리소스 컬렉션을 그룹화하고 볼 수 있습니다. 이러한 계정은 개발, 테스트 및 프로덕션 수명 주기 계정일 수 있습니다.

AppRegistry를 사용하여 메타데이터 속성을 애플리케이션에 연결할 수도 있습니다. 속성 세트를 포함하는 재사용 가능한 속성 그룹을 지정할 수 있습니다. 그런 다음 AppRegistry 또는 통합 서비스를 사용하여 지정된 속성이 있는 애플리케이션 리소스를 검색하고 작업을 수행할 수 있습니다. 이러한 통합 서비스에는 다음이 포함됩니다.

- 애플리케이션 및 클러스터의 컨텍스트에서 리소스 관련 문제를 조사하고 해결하는의 기능인 Application Manager AWS Systems Manager AWS
- AWS Resource Access Manager, 애플리케이션 및 속성 그룹을 AWS 조직 보안 주체와 공유
- AWS 에서 작동하는 서비스 AWS Resource Groups
- AWS Resilience Hub: 제품 구조 검색 및 복원력 평가를 위한 기능
- AWS Service Management Connector: ServiceNow, JIRA 및 기타 널리 사용되는 도구에 대한 연결을 선언하고 설정하는 기능

AppRegistry에 대한 자세한 내용은 다음을 참조하세요.

- AppRegistry 관리자 안내서.
- Increase application visibility and governance using AWS Service Catalog AppRegistry 블로그 게시물. 이 문서에서는 AppRegistry의 애플리케이션으로 인프라를 등록하는 명령줄 예제를 통해 인프라 거버넌스에서 AppRegistry를 사용하는 방법에 대한 개요를 제공합니다.
- Govern your applications centrally using AppRegistry and Application Manager 블로그 게시물. 이 문서에서는 AppRegistry를 적용하여에 LAMP 웹 애플리케이션을 등록 AWS Management Console 하고 Application Manager를 사용하여 관리하는 방법에 대한 개요를 제공합니다.

재고 관리

Service Catalog에는 제품 공유 및 셀프 서비스를 통해 제품을 프로비저닝할 때 제품을 등록하는 자체 내부 인벤토리 관리 기능이 있습니다. 그러나 AWS Config 또는 AppRegistry 및 관련 서비스를 사용하여 제품에서 프로비저닝한 리소스를 관리하는 것이 좋습니다. 이러한 도구는 나머지 AWS 인프라를 사용하여 프로비저닝된 Service Catalog 제품을 관리하는 보다 포괄적이고 통합된 접근 방식을 제공합니다.를 AWS Config 사용하면 콘솔에서 또는 AWS SDK API를 사용하여 프로비저닝된 제품에 대한 작업을 인벤토리에 추가하고 수행할 수 있습니다. Application Manager와 통합된 AppRegistry는 Service Catalog 프로비저닝 제품에 대한 인벤토리 관리 기능도 제공합니다.

AWS Service Catalog 도구 사용

보다 선언 중심의 방식으로 사용자 지정 프로비저닝 워크플로에서 IaC 제품을 프로비저닝하려면 Service Catalog 기능의 일부를 보강해야 할 수 있습니다. AWS에서는 이러한 요구 사항을 지원하는 몇 가지 도구를 제공합니다. AWS Labs 프로젝트에서 널리 사용되는 Service Catalog Puppet 및 Service Catalog Factory라는 두 가지 도구를 제공합니다.

주제

- Service Catalog Puppet
- Service Catalog Factory

Service Catalog Puppet

Service Catalog Puppet은 AWS Boto3 API를 사용하여 Python에서 구현됩니다. 이 도구는 Service Catalog 제품을 구성하고 프로비저닝하기 위한 몇 가지 강력한 기능을 제공합니다. 개발자는 매니페스트 역할을 하는 YAML 템플릿을 사용하여 Service Catalog 제품 및 포트폴리오 프로비저닝 정보를 구성할 수 있습니다. Service Catalog Puppet 프로비저닝 워크플로는 Service Catalog보다 더 복잡한 배포 프로세스가 필요한 제품을 지원합니다. 또한 성능 최적화를 지원하여 공격적인 기간에 대규모로 제품을 프로비저닝할 수 있습니다.

Service Catalog Puppet은 배포 시 제품 프로비저닝을 위해 Service Catalog CloudFormation 템플릿에 액세스합니다. CloudFormation을 직접 호출하여 제품의 프로비저닝 템플릿 스택을 배포하고 Service Catalog의 자체 스택 세트 프로비저닝 프로세스에서 부과하는 제한 사항을 우회합니다. 프로비저닝 템플릿이 매크로를 사용하여 다른 CloudFormation 스크립트를 포함하거나 중첩된 CloudFormation 스크립트를 사용하는 경우 대상 계정에서 프로비저닝 워크플로의 부트스트래핑 부분에 있는 해당 스크립트에 대한 액세스를 제공해야 합니다.

자세한 내용:

- Service Catalog Puppet 설명서 및 GitHub 리포지토리를 참조하세요.
- Service Catalog Puppet SDK를 사용하여 프로그래밍 방식으로 도구와 상호 작용하여 제품 및 포트폴리오 프로비저닝을 시작하려면 SDK 설명서를 참조하세요.
- GitOps는 Service Catalog Puppet 환경을 관리하기 위한 기본 메커니즘입니다.

Service Catalog Puppet은 개발자가 매우 쉽게 배울 수 있는 도구입니다. 제품 프로비저닝 템플릿 및 YAML 템플릿을 구현하여 매니페스트를 구현하려면 CloudFormation에 익숙해져야 합니다. 자기 주도형 워크숍과 같이 새로운 개발자의 속도를 높이는 데 사용할 수 있는 좋은 워크숍이 있습니다.

프로비저닝 워크플로에 대한 지원

Service Catalog Puppet은 Python Luigi 작업 오케스트레이션 엔진을 사용하여 부트스트래핑 및 프로비저닝 워크플로를 구현합니다. 이러한 워크플로의 모든 단계는 Luigi 워크플로 태스크로 구현됩니다. Luigi에 대한 개요 및 이를 다른 인기 워크플로 도구와 비교하는 방법은 Data Revenue 블로그의 Airflow vs. Luigi vs. Argo vs. MLFlow vs. KubeFlow를 참조하세요.

Luigi를 사용하면 Service Catalog Puppet이 워크플로 태스크와 관련된 작업자 수를 제어하고 규모 조정과 성능 개선을 위해 워크플로의 다른 측면을 제어할 수 있습니다. 또한 Service Catalog Puppet은 제품 및 단계 종속성을 관리하고 제품 프로비저닝을 오케스트레이션하기 위한 depends_on 메커니즘을 제공합니다. 이 기능을 사용하면 세분화된 제품 정의와 복잡한 종속성을 구현하고 운영적으로 관리할 수 있습니다.

프로비저닝 모드

Service Catalog Puppet은 허브, 스포크, 비동기라는 세 가지 실행 모드를 지원합니다. 세 가지 모드 모두 Service Catalog에 이미 정의된 포트폴리오 내에서 제품을 프로비저닝합니다. 이들은 대상 계정에 대한 Service Catalog 제품 공유에 의존하고 Service Catalog 관리자 및 시작 역할을 사용하여 해당 대상에서 프로비저닝을 실현합니다. Service Catalog Puppet은 YAML 구성 파일에 제공된 역할 구성을 기반으로 동일한 조직 내에서 부트스트래핑 단계를 수행합니다. 또한 이 도구는 단일 허브 계정에서 여러 조직에 대한 프로비저닝을 지원합니다. 이 시나리오에서는 Service Catalog Puppet이 외부 조직의 계정에서 필요한 프로비저닝 작업을 수행할 수 있도록 외부 조직에서 부트스트래핑을 수동으로 수행해야 합니다.

모든 프로비저닝 모드에서 Service Catalog Puppet은 Service Catalog의 프로비저닝 프로세스를 직접 호출하지 않고 제품 프로비저닝을 직접 구현합니다. 기존 Service Catalog 스택 세트 제약 조건의 역할 및 대상 계정 사양을 사용하도록 프로비저닝 매니페스트를 구성할 수 있습니다. Service Catalog Puppet은 이 정보를 사용하여 Luigi 워크플로로 자체 프로비저닝을 수행합니다.

OU 또는 계정을 직접 지정하는 것 외에도 계정 태그 지정 접근 방식을 기반으로 제품 포트폴리오 프로비저닝 대상을 정의할 수 있습니다. 계정 태그 기반 프로비저닝에서 포트폴리오 제품은 지정된 매니페스트 프로비저닝 태그 세트에 모든 태그가 있는 모든 계정으로 프로비저닝됩니다. 예를 들어 미국 동부 리전에서 모든 기관의 프로덕션 계정에 대해 포트폴리오 제품을 실행하려는 경우 태그 `type:prod`, `partition:us-east` 및 `scope:institutional-client`를 지정할 수 있습니다. 계정 및 OU 제

외를 선언하여 사용자가 지정한 태그가 있는 OU나 계정 또는 OU에서 지정한 대상의 멤버인 계정으로의 프로비저닝을 금지할 수도 있습니다. 계정 태그 지정에 대한 자세한 내용은 Service Catalog 도구 설명서를 참조하세요.

허브 모드

허브 프로비저닝 모드에서 스포크 계정에 대한 모든 Luigi 워크플로는 지정된 중앙 허브 계정에서 관리됩니다. 허브 계정에서는 스포크 계정에서 작업을 수행할 수 있는 IAM 역할을 수임하지만, 태스크 관리는 허브 계정 내에서 이루어집니다. 허브 계정은 모든 스포크 계정 프로비저닝 태스크가 성공적으로 완료되거나 실패할 때까지 동기적으로 대기합니다. 그런 다음 최종 상태를 보고합니다. 허브 계정 모드는 가장 오래되고 성숙한 프로비저닝 모드입니다. 그러나 많은 사용자가 프로비저닝 동시성과 속도를 더 높이기 위해 스포크 프로비저닝 모드로 이전했습니다.

스포크 모드

스포크 모드에서 Service Catalog 허브 계정은 지정된 부트스트래핑된 스포크 계정에서 실행할 Luigi 워크플로를 시작합니다. 스포크 워크플로가 완료되면 허브 계정에 알립니다. 스포크 계정에서 발생한 장애는 허브 계정으로 확대됩니다. 허브 계정은 스포크 계정을 폴링하여 완료 여부를 확인하고 상태를 판단합니다.

거의 모든 기능이 별도의 AWS 서비스 스포크 계정에서 실행되므로 스포크 모드에서 할당량 증가가 필요한 경우는 거의 없습니다. 또한 스포크 모드는 중앙 제어를 유지하면서 허브 모드보다 훨씬 더 큰 동시성을 제공합니다. 허브 모드보다 프로비저닝 속도를 800% 개선할 수 있습니다. 스포크 모드는 제품 간 Depends0n 관계를 통한 제품 체이닝을 지원하며, 이를 통해 이에 종속된 제품은 이미 프로비저닝되어 있습니다. 체이닝된 제품으로 구성된 제품은 구성 요소로 체인닝된 제품을 프로비저닝할 수도 있습니다. 특수 AWS Lambda 함수 직접 호출을 사용하여 필요한 단계를 수행할 수도 있습니다. 한 스포크에서 발생한 결함은 다른 스포크와 격리됩니다.

최대 7개 리전에서 980개 이상의 계정을 보유하는 기업에서 스포크 모드가 사용됩니다. 이러한 기업은 일반적으로 1시간 이내에 인프라의 모든 리전과 계정으로 제품을 프로비저닝할 수 있습니다.

> **Note**
>
> 이때 결과는 네트워킹 인프라, 워크로드, AWS 조직 허브 및 스포크 계정에 적용되는 할당량과 같은 요인에 따라 달라질 수 있습니다. 또한 프로비저닝되는 제품 리소스, 고유한 생성 시간 및 다른 리소스에 대한 종속성에 따라 달라집니다.

비동기 모드

비동기 모드에서는 스포크 계정에서 프로비저닝 워크플로를 시작하지만 스포크에서 완료 응답을 기다리거나 받지 않습니다.

캐싱

Service Catalog Puppet이 워크플로 속도를 최적화하는 데 사용하는 또 다른 메커니즘은 워크플로의 단계를 나타내는 일반적인 태스크를 캐싱하는 기능입니다. 캐싱된 태스크가 완료되면 Amazon Simple Storage Service(Amazon S3)에 출력을 씁니다. 다음에 동일한 파라미터로 동일한 세션에서 태스크가 간접 호출될 때 Service Catalog Puppet은 태스크를 다시 실행하는 대신 캐싱된 값을 사용합니다. 자세한 내용은 Service Catalog Puppet 설명서의 Caching을 참조하세요.

DevSecOps 수명 주기 지원

Service Catalog Puppet에는 DevSecOps 파이프라인 관리에 대한 지원이 포함되어 있습니다. Service Catalog Tools 작업(Service Catalog Puppet 개요에 설명됨)을 사용하여 권장되는 카나리 계정을 포함한 AWS 수명 주기 계정 전체에서 테스트를 자동화하고 제품을 승격할 수 있습니다. 자세한 내용은 Service Catalog Puppet 설명서의 Managing your environments를 참조하세요.

광범위한 프로덕션 사용 전에 제품 변경과 관련된 문제를 감지하려면 Service Catalog Puppet에 초기 배포를 위한 카나리 계정이 하나 이상 필요합니다. 새 릴리스를 테스트하고 확신을 얻은 후 카나리가아닌 프로덕션 계정으로 승격할 수 있습니다. 문제가 식별되는 경우 릴리스를 롤백하고 문제가 해결되면 다시 도입할 수 있습니다. 이 접근 방식을 사용하는 경우 프로덕션 계정에 문제가 있는 카나리 버전을 릴리스하면 프로덕션 문제가 발생할 수 있습니다. 대안으로, 변경 사항을 프로덕션에 릴리스하기 전에 각 제품 변경 사항에 대해 전체 회귀 테스트를 실행할 수 있습니다. 이로 인해 CI/CD 프로세스에 추가 오버헤드가 발생하지만 프로덕션 문제를 방지하는 데 도움이 됩니다. 개발 팀에 가장 적합한 기능 릴리스 시나리오와 접근 방식을 결정하는 것은 DevSecOps 관리자의 책임입니다.

Service Catalog Puppet을 사용하면 여러 팀이 Service Catalog 제품 솔루션의 프로비저닝을 동시에 개발하고 테스트할 수 있습니다. 여러 개발자가 동시에 제품을 변경하지 않는 것이 모범 사례입니다. 대신 별도로 동시에 수정할 수 있도록 제품을 더 세분화된 구성 요소로 나눌 수 있습니다.

Service Catalog Puppet은 정적 및 유닛 테스트 기능을 제공하는 어설션 문을 통해 테스트를 자동화하는 데도 도움이 됩니다. 정책 시뮬레이터를 사용하여 서비스 제어 정책(SCP) 및 IAM 정책을 테스트할 수 있습니다. 기술적으로 포괄적인 테스트이지만 시스템 통합 테스트(SIT) 환경에서 사용할 수 있습니다. 자세한 내용은 Service Catalog Puppet 설명서의 Using policy simulations 및 Applying service control policies를 참조하세요.

성숙도, 완전성 및 지원

Service Catalog Puppet은 공식적으로 지원되는 AWS 서비스는 아니지만 널리 채택되고 있습니다. 이 도구는 지난 몇 년 동안 대규모 조직에서 원하는 프로비저닝 기간에 중앙에서 수백 개의 OU 계정으로 제품을 성공적으로 프로비저닝하는 데 사용되었습니다. 대규모로 내결함성 제품을 프로비저닝하는 기능은 입증되었습니다. Service Catalog Puppet에 문제가 발생하는 사용자는 이 AWS Labs 솔루션의 기여자가 해결할 수 있도록 GitHub 리포지토리에 문제를 로깅할 수 있습니다.

Service Catalog Factory

Service Catalog Factory는 AWS Labs에서 제공하는 또 다른 도구입니다. 이는 AWS Control Tower와 비슷합니다. 즉, 계정을 생성하고 Puppet을 통해 Service Catalog를 직접 호출하여 해당 계정 내에서 IaP를 프로비저닝합니다. Service Catalog Puppet과 동일한 여러 메커니즘을 사용하여 해당 기능을 구현합니다. Service Catalog Factory는 Service Catalog 또는 Service Catalog Puppet을 직접 호출하여 계정에 있는 제품에 대한 인프라를 프로비저닝할 수 있습니다. 또한 이 도구는 여러 AWS 리전 및 조직에서 계정 생성을 지원합니다. 자세한 내용은 Service Catalog Factory 설명서 및 GitHub 리포지토리를 참조하세요.

요약 및 다음 단계

Service Catalog를 사용하면 인프라를 제품으로 빠르고 신뢰할 수 있는 방식으로 프로비저닝할 수 있습니다. 정의된 제품 카탈로그에서 인프라를 셀프 서비스로 지원하거나 허브 및 스포크 모델의 지정된 대상 계정으로 제품을 푸시할 수 있습니다. CloudFormation 스크립팅을 사용하거나 AWS CDK를 사용하여 Service Catalog 제품 및 해당 프로비저닝 템플릿을 정의할 수 있습니다. 두 접근 방식 모두에서 Service Catalog는 CloudFormation을 직접 호출하여 제품의 프로비저닝 템플릿을 나타내는 스택을 배포해 제품을 프로비저닝합니다. 스택은 CloudFormation 스택 세트 내 지정된 모든 대상 계정에 배포됩니다.

Service Catalog 개발을 위한 AWS CDK 접근 방식은 사전 정의된 Service Catalog 제품 및 포트폴리오 클래스와 사전 정의된 리소스 유형을 사용하여 제품 및 해당 리소스를 정의할 수 있으므로 CloudFormation보다 더 포괄적인 모듈화와 재사용을 지원합니다. AWS CDK 구현에는 고급 프로그래밍 기술이 필요합니다. 이는 조직이 표준화된 리소스 구성 및 동작을 통해 AWS 인프라 개발의 기반으로 재사용 가능한 자체 제품 프레임워크를 설정하려는 경우 정당화될 수 있습니다.

Service Catalog Puppet 및 Service Catalog Factory를 사용하여 주로 프로비저닝을 위해 Service Catalog 기능을 보강할 수 있습니다. Service Catalog Puppet은 선언적 및 태그 기반 제품 프로비저닝 사양, 기본 제공, 사용자 지정 가능한 고성능 및 목적별 프로비저닝 워크플로, 기본 제공, 사용자 지정 가능한 작업 기반 CI/CD 및 SDLC 파이프라인을 제공합니다. 워크플로 종속성 관리 및 기본 제공 테스트 자동화 기능을 사용하면 운영 위험이 적은 Service Catalog 제품을 체이닝할 수 있습니다. Service Catalog Puppet을 사용하면 공격적인 기간에 수백 개의 계정에 제품을 신뢰할 수 있는 방식으로 프로비저닝할 수 있습니다. Service Catalog Factory는 AWS Control Tower와 유사합니다. 계정을 생성하고 Service Catalog를 직접 호출하여 해당 계정 내에서 IaP를 프로비저닝합니다.

Service Catalog 및 Service Catalog 도구는 AWS에서 IaP를 관리하는 데 도움이 되는 광범위한 기능을 제공합니다. Service Catalog와 이러한 도구는 지속적으로 개선되고 있습니다. 최신 기능은 AWS Service Catalog 기능 및 AWS Service Catalog Products repository를 참조하세요.

리소스

참조:

- Service Catalog 설명서
- Service Catalog API
- AppRegistry
- CloudFormation 설명서
- CloudFormation StackSets
- AWS::ServiceCatalog::CloudFormationProduct resource
- Terraform을 사용한 AWS Service Catalog 제품 시작
- AWS Cloud Development Kit (AWS CDK)
- Service Catalog Construct Library
- AWS CDK ProductStack class
- AWS Organizations 설명서

도구

- Service Catalog Puppet 설명서
- Service Catalog Puppet GitHub repository
- Service Catalog Factory 설명서
- Service Catalog Factory GitHub repository

AWS 권장 가이드 패턴

- Manage AWS Service Catalog products in multiple AWS 계정 and AWS 리전
- Copy AWS Service Catalog products across different AWS 계정 and AWS 리전
- Automate AWS Service Catalog portfolio and product deployment by using AWS CDK

문서 기록

아래 표에 이 가이드의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
최초 게시	—	2023년 1월 30일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.