云资源自动化 for Crossplane

操作指南

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2024 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【商标声明】

腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

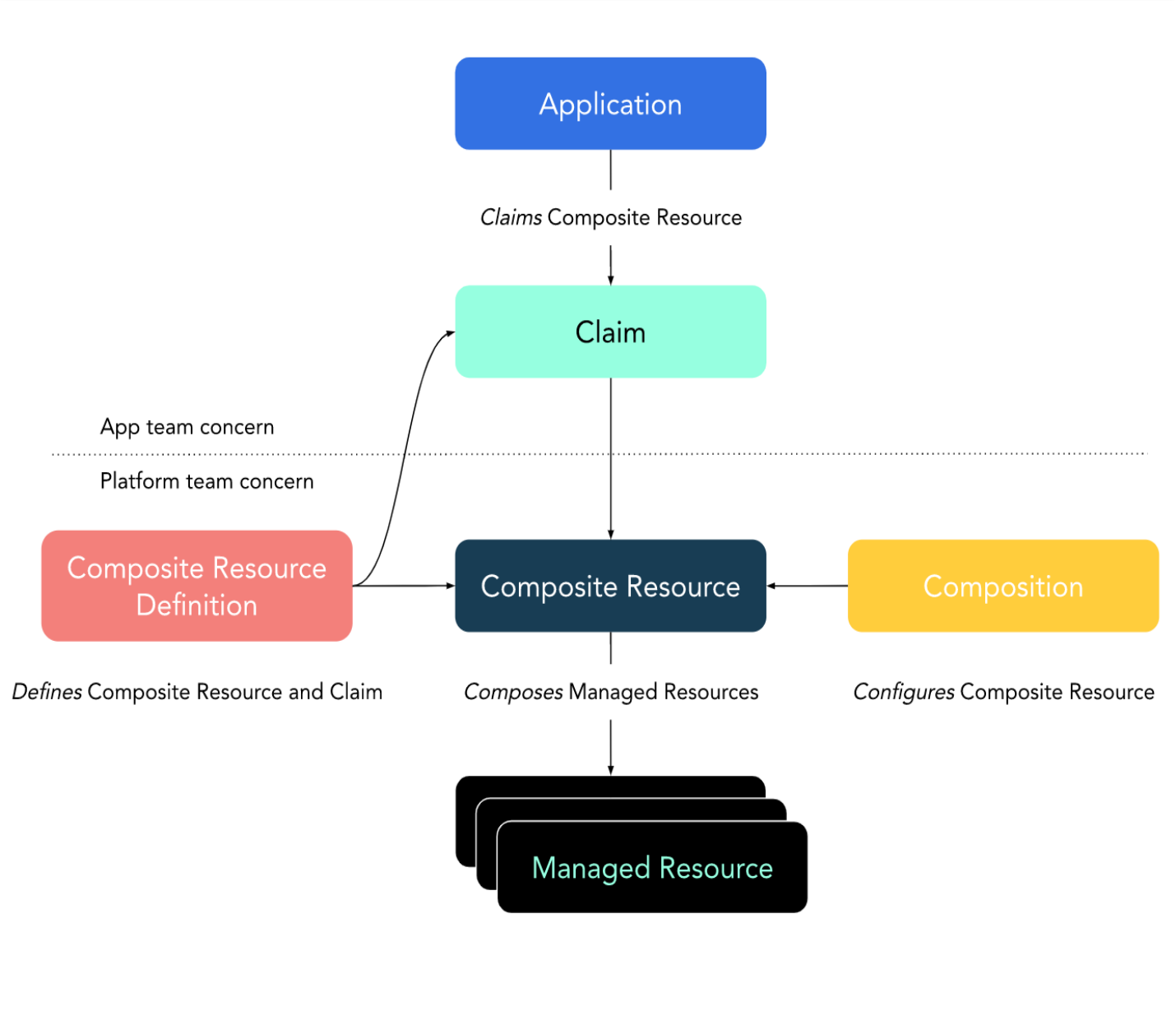
文档目录

操作指南
Crossplane 核心概念
Crossplane CLI 简明指南

操作指南
Crossplane 核心概念

最近更新时间：2024-12-18 10:53:42

Crossplane 扩展了 Kubernetes，使其能够创建和管理 Kubernetes 集群外部的资源。Crossplane 使用了多个核心组件来管理通过 Kubernetes 构建和管理外部资源的各个元素。它们之间的关系如下：



本文向您介绍以上核心组件的概念和使用。您也可参考 示例教程 获取以上概念的示例。

云资源供应商（Provider）

云资源供应商（Provider）使 Crossplane 能够在外部服务上配置基础设施。Provider 创建新的 Kubernetes API 并将它们映射到外部 API。云资源供应商负责连接非 Kubernetes 资源的各个方面。腾讯云作为 Crossplane 的供应商 一员，提供对应腾讯云资源的支持和服务。

安装 Provider

在开始使用 Crossplane 前，我们需要先安装 Provider 程序。

```
apiVersion: pkg.crossplane.io/v1
kind: Provider
metadata:
  name: provider-tencentcloud
spec:
  package: xpkg.upbound.io/crossplane-contrib/provider-
tencentcloud:v0.7.1
```

托管资源（Managed Resources，MR）

托管资源（Managed Resources，MR）是表示 Provider 在 Kubernetes 外部创建的事物的 Kubernetes 对象。在 Kubernetes 中创建托管资源需要 Provider 来创建相应的外部资源。删除托管资源需要提供者删除关联的外部资源。例如，腾讯云提供了 CVM、MySQL、Ckafka 等资源，这些资源在 Kubernetes 中以托管资源的形式存在。
一个可用的托管资源如下：

```
apiVersion: vpc.tencentcloud.crossplane.io/v1alpha1
kind: VPC
metadata:
  name: resource-vpc
  namespace: crossplane-system
spec:
  forProvider:
    cidrBlock: 10.1.0.0/16
    name: crossplane-test-vpc
```

group，kind 和 version

每个托管资源都是一个独特的 API 端点，具有自己的**组**、**种类**和**版本信息**。
例如，以下定义了一个**组**、**种类**和**版本**分别为 vpc.tencentcloud.crossplane.io、VPC、v1alpha1的资源。

```
apiVersion: vpc.tencentcloud.crossplane.io/v1alpha1
kind: VPC
```

forProvider

`spec.forProvider` 用于定义映射到外部资源的参数。

```
spec:
  forProvider: # 定义外部资源的参数
    cidrBlock: 10.1.0.0/16 # cidr
    name: crossplane-test-vpc # 名称
```

name

`name` 标识该资源在 Kubernetes 中的名称，`EXTERNAL-NAME` 为该资源的真实 Id。

```
❯ k get managed
NAME                    READY   SYNCED   EXTERNAL-NAME   AGE
resource-vpc-m8zgg      True    True     vpc-hbgs1o5r    2d16h
```

组合（Composition）

组合是用于将多个托管资源创建为单个对象的模板。组合描述了更复杂的部署，组合了多个托管资源和任何资源自定义，例如数据库或云提供商区域的大小。
以下是一个名为 "composition-cvm" 的组合示例：

```
apiVersion: apiextensions.crossplane.io/v1
kind: Composition
metadata:
  name: composition-cvm
spec:
  compositeTypeRef:
    apiVersion: crd.tencentcloud.crossplane.io/v1alpha1
    kind: XCvm
  resources:
    - name: vpc
    （略）
    - name: subnet
    （略）
    - name: cvm
    （略）
```

在这个示例中，我们将多个不同类型的资源（vpc、subnet、cvm）定义在一起，以便一并进行部署和管理。

compositeTypeRef

定义了该组合实际应用于哪个 Composite Resource Definition (XRD) 上，通过 `apiVersion` 和 `kind` 匹配对应的 XRD。

base

`base` 用于定义该资源实际应用在哪个资源上，通过 `apiVersion` 和 `kind` 匹配实际资源。

```
resources:
  - name: cvm
    base:
      apiVersion: cvm.tencentcloud.crossplane.io/v1alpha1
      kind: Instance
      spec:
        forProvider:
          # imageId: "img-9qrfy1xt"
    patches:
    （略）
```

patches

`pathches` 用户定义外部资源和配置中参数的映射，您可以通过该字段，将不同名称或来源的参数在此进行映射匹配。
例如，在上述示例中，为 cvm 资源的 `vpcId` 和 `subnetId` 参数定义来源。

```
resources:
  - name: cvm
    base:
    （略）
    patches:
      - fromFieldPath: status.share.vpcId
        toFieldPath: spec.forProvider.vpcId
      - fromFieldPath: status.share.subnetId
        toFieldPath: spec.forProvider.subnetId
```

复合资源定义（Composition Resource Definition，XRD）

XRD 代表自定义 API，由平台工程师创建并由开发人员或最终用户使用。最终用户参考 XRD，即可知道如何使用该复合资源。
以下为一个 XRD 的示例：

```
apiVersion: apiextensions.crossplane.io/v1
kind: CompositeResourceDefinition
```

```
metadata:
  name: xcvms.crd.tencentcloud.crossplane.io
spec:
  group: crd.tencentcloud.crossplane.io
  names:
    kind: XCvm
    plural: xcvms
  versions:
    - name: v1alpha1
      served: true
      referenceable: true
      schema:
        openAPIV3Schema:
          type: object
          properties:
            spec:
              type: object
              properties:
                vpcName:
                  type: string
                  （略）
              required:
                - vpcName
                （略）
            status:
              description: A Status represents the observed state
              properties:
                share:
                  （略）
              type: object
```

group、kind

`group` 和 `kind` 字段用于匹配 组合（Composition）里的 `compositeTypeRef` 字段。

```
# composition.yaml
apiVersion: apiextensions.crossplane.io/v1
kind: Composition
metadata:
  name: composition-cvm
spec:
  compositeTypeRef:
    apiVersion: crd.tencentcloud.crossplane.io/v1alpha1
```

```
    kind: XCvm
```

versions

可为 XRD 定义多套参数，提供给多个 Composition 复用，用版本进行管理。

properties

用于定义 `spec` 和 `status` 。

- spec

  定义组合可以接收的参数。required 为必选参数。

```
        properties:
          spec:
            type: object
            properties:
              vpcName:
                type: string
              （略）
            required:
              - vpcName
              （略）
```

- status

  定义组合可以提供访问的参数。用户通过 `share.stauts` 访问。

```
        properties:
          status:
            description: A Status represents the observed state
            properties:
              share:
                description: Freeform field containing status
information
                type: object
                x-kubernetes-preserve-unknown-fields: true
            type: object
```

复合资源（Composition Resource，XR）

复合资源将一组托管资源表示为单个 Kubernetes 对象。当用户访问 XRD 中定义的自定义 API 时，Crossplane 会创建复合资源。

用户参考 XRD 中的参数定义，在 XR 中指定实际参数值。

以下为一个复合资源的示例：

```
apiVersion: crd.tencentcloud.crossplane.io/v1alpha1
kind: XCvm
metadata:
  name: xcvm-example
spec:
  vpcName: xvpcName # 指定具体vpcName
  subnetName: xsubnetName # 指定具体subnetName
  availabilityZone: ap-guangzhou-3 # 指定可用区
  # imageId: img-9qrfy1xt # 指定镜像（非必选参数）
```

声明（Claim）

类似于复合资源，但存在于 Kubernetes 命名空间中。每个声明都链接到单个集群范围的复合资源。平台用户在其独特的命名空间中创建声明，将其资源与其他命名空间中的其他团队隔离。详情请参见官网文档，获取 Claims 更多信息。

Crossplane CLI 简明指南

最近更新时间：2024-12-18 10:53:42

安装 CLI

请参考 安装 Crossplane 部分，详细内容请查阅官网 Crossplane 安装指南 。

CLI 命令指南

Crossplane CLI 提供了一些实用程序，以下为常用命令：

xpkg

`crossplane xpkg` 命令用于创建、安装和更新 Crossplane package，以及启用身份验证和将 crossplane 软件包发布到 crossplane 软件包注册表。

xpkg build

`crossplane xpkg build` 提供了构建 crossplane 软件包的自动化和简化。crossplane CLI 将 YAML 文件目录组合在一起，并将其打包为 OCI 容器镜像 。支持构建 配置 、函数 和 Provider 包 等类型。

xpkg install

`crossplane xpkg install` 用于将指定的软件包下载并安装到 Crossplane 中。默认情况下，该命令会使用在 `~/.kube/config` 中定义的 Kubernetes 配置。您可以使用环境变量 `KUBECONFIG` 来定义自定义的 Kubernetes 配置文件位置。该命令需要指定软件包类型、软件包文件，并可选择为软件包在 Crossplane 中指定一个名称。命令定义如下：

```
crossplane xpkg install<package-kind> <registry URL package name and tag>
[<optional-name>]
```

例如，要安装 0.8.1 版的腾讯云 Provider crossplane-contrib/provider-tencentcloud@v0.8.1 。

```
crossplane xpkg install provider xpkg.upbound.io/crossplane-
contrib/provider-tencentcloud:v0.8.1
```

xpkg login

`crossplane xpkg login` 用于对 Upbound Marketplace 容器注册表 xpkg.upbound.io 进行身份验证，以便您在 Upbound Register 中推送软件包和创建私有软件仓库。

xpkg logout

`crossplane xpkg logout` 用于使当前的 crossplane xpkg login 会话失效。该命令会删除 ~/.crossplane/config.json 文件中的 `session` ，但不会删除配置文件。

xpkg push

`crossplane xpkg push` 用于将 Crossplane 软件包文件推送到软件包注册表。Crossplane CLI 默认会将镜像推送到位于 xpkg.upbound.io 的 Upbound Marketplace 。您可以使用 `crossplane xpkg push <package>` 命令指定组织、软件包名称和标签。默认情况下，该命令会在当前目录下查找要推送的单个 .xpkg 文件。若要推送多个文件或指定特定的 .xpkg 文件，请使用 -f flag。
例如，要将名为 “my-package” 的本地软件包推送到 “crossplane-docs/my-package:v0.14.0”，示例如下：

```
crossplane xpkg push -f my-package.xpkg crossplane-docs/my-package:v0.14.0
```

要推送到其他软件包注册表，如 DockerHub ，需要提供完整的 URL 和软件包名称。
例如，要将名为 “my-package” 的本地包推送到 DockerHub 组织 “crossplane-docs/my-package:v0.14.0”，示例如下：

```
crossplane xpkg push -f my-package.xpkg index.docker.io/crossplane-docs/my-package:v0.14.0
```