Guida introduttiva a Terraform: Guida per esperti AWS CDK AWS CloudFormation

AWS Guida prescrittiva

AWS Guida prescrittiva: Guida introduttiva a Terraform: Guida per esperti AWS CDK AWS CloudFormation

Table of Contents

Guida introduttiva a Terraform: linee guida per AWS CDK e esperti AWS CloudFormation

Steven Guggenheimer, Amazon Web Services ()AWS

Marzo 2024 (cronologia del documento)

Se la tua esperienza con il provisioning di risorse cloud rientra esclusivamente nell'ambito di AWS, potresti avere un'esperienza limitata con gli strumenti Infrastructure as Code (IaC) oltre la Rand. AWS Cloud Development Kit (AWS CDK)AWS CloudFormation In effetti, strumenti simili, come Hashicorp Terraform, potrebbero esserti del tutto sconosciuti. Tuttavia, più approfondisci il tuo percorso verso il cloud, più inevitabile diventa che incontrerai Terraform. Sarà decisamente a tuo vantaggio conoscerne i concetti fondamentali.

Sebbene Terraform, the AWS CDK, CloudFormation raggiungano obiettivi simili e condividano molti concetti fondamentali, ci sono alcune differenze. Potresti non essere preparato a queste differenze se ti avvicini a Terraform per la prima volta. Dopotutto, AWS CDK gli CloudFormation stack sono tutti basati all'interno Account AWS, quindi in questo modo hanno una relazione diretta con la maggior parte delle risorse che gestiscono. Terraform non si basa sull'ambiente di nessun singolo provider di cloud. Ciò gli offre la flessibilità necessaria per supportare diversi provider, ma deve mantenere le risorse da una posizione remota.

Questa guida aiuta a demistificare i concetti fondamentali alla base di Terraform per aiutarti a gestire qualsiasi sfida IaC che ti si presenta. Si concentra su come Terraform utilizza concetti, come provider, moduli e file di stato, per fornire risorse. Inoltre, contrappone i concetti di Terraform al modo in cui AWS CDK ed CloudFormation esegue operazioni simili.

> **Note**
>
> AWS CDK Aiuta gli sviluppatori a distribuire gli CloudFormation stack utilizzando linguaggi di codifica programmatici. Dopo l'esecuzione`cdk synth`, il codice viene convertito in modelli. CloudFormation Da quel momento in poi, il processo è identico tra AWS CDK e CloudFormation. Per motivi di brevità, questa guida di solito si riferisce al processo AWS IaC in CloudFormation termini, ma i confronti sono altrettanto adatti per. AWS CDK

CloudFormation e terminologia Terraform

Quando si confronta Terraform con AWS CDK and CloudFormation, riconciliare i concetti fondamentali di IaC può essere difficile a causa della terminologia incoerente utilizzata per descriverli. Di seguito sono riportati questi termini e il modo in cui questa guida si riferirà ad essi:

- Stack : uno stack è un IAC distribuito in una CI/CD pipeline e tracciabile come singola unità.Sebbene questo termine sia comune in CloudFormation, Terraform non lo usa realmente. Uno stack Terraform è un modulo root distribuito con tutti i suoi moduli figlio. Tuttavia, per evitare confusione con il termine modulo, questa guida utilizza il termine stack per descrivere una singola implementazione per entrambi gli strumenti.
- Stato: lo stato indica tutte le risorse attualmente monitorate e le relative configurazioni correnti all'interno di uno stack di distribuzione IaC. Come descritto nella Comprensione degli stati e dei backend di Terraform sezione, Terraform usa il termine stato più di. CloudFormation Questo perché il mantenimento dello stato è più visibile in Terraform, ma il monitoraggio e l'aggiornamento dello stato sono altrettanto importanti per. CloudFormation
- File IaC: un file IaC è un singolo file che contiene il linguaggio Infrastructure as Code (IaC). CloudFormation si riferisce a un singolo CloudFormation file come modello. Tuttavia, i modelli e i file modello in Terraform sono qualcosa di completamente diverso. L'equivalente di un CloudFormation modello in Terraform è chiamato file di configurazione. Per ridurre al minimo la confusione in questa guida, il termine file o file IAc viene utilizzato per riferirsi sia ai CloudFormation modelli che ai file di configurazione Terraform.

La tabella seguente confronta la terminologia utilizzata per Terraform e Terraform. CloudFormation L'intento di questa tabella è mostrare le somiglianze. Non si tratta di confronti one-to-one. Ogni concetto differisce almeno leggermente tra CloudFormation e Terraform. I concetti sono spiegati in modo approfondito nelle sezioni pertinenti di questa guida.

CloudFormation termine	Termine Terraform	Sezione di questa guida
Interfacce CDK (ad esempio) IBucket	Origine dati	Comprensione delle fonti di dati Terraform
Cambia set	Pianificazione	Comprensione dei moduli Terraform

CloudFormation termine	Termine Terraform	Sezione di questa guida
Funzioni di condizione	Espressioni condizionali	Comprensione delle funzioni, delle espressioni e dei meta-argomenti di Terraform
Attributo `DependsOn`	`depends_on` meta-argomento	Comprensione delle funzioni, delle espressioni e dei meta-argomenti di Terraform
Funzioni intrinseche	Funzioni	Comprensione delle funzioni, delle espressioni e dei meta-argomenti di Terraform
Moduli	Moduli	Comprensione dei moduli Terraform
Output	Valori di uscita	Comprensione delle variabili Terraform, dei valori locali e degli output
Parameters	Variabili	Comprensione delle variabili Terraform, dei valori locali e degli output
Registro	Provider	Comprendere i fornitori Terraform
Modello	File di configurazione	Tutti

Comprendere le risorse Terraform

Il motivo principale dell'esistenza di entrambi AWS CloudFormation e di Terraform è la creazione e la manutenzione di risorse cloud. Ma cos'è esattamente una risorsa cloud? E le CloudFormation risorse e le risorse Terraform sono la stessa cosa? La risposta è... sì e no. Per illustrare ciò, questa guida fornisce un esempio di utilizzo CloudFormation e quindi di Terraform per creare un bucket Amazon Simple Storage Service (Amazon S3).

Il seguente esempio di CloudFormation codice crea un bucket Amazon S3 di esempio.

```
{
    "myS3Bucket": {
        "Type": "AWS::S3::Bucket",
        "Properties": {
            "BucketName": "my-s3-bucket",
            "BucketEncryption": {
                "ServerSideEncryptionConfiguration": [
                    {
                        "ServerSideEncryptionByDefault": {
                            "SSEAlgorithm": "AES256"
                        }
                    }
                ]
            },
            "PublicAccessBlockConfiguration": {
                "BlockPublicAcls": true,
                "BlockPublicPolicy": true,
                "IgnorePublicAcls": true,
                "RestrictPublicBuckets": true
            },
            "VersioningConfiguration": {
                "Status": "Enabled"
            }
        }
    }
}
```

Il seguente esempio di codice Terraform crea un bucket Amazon S3 identico.

```
resource "aws_s3_bucket" "myS3Bucket" {
```

```
  bucket = "my-s3-bucket"
}

resource "aws_s3_bucket_server_side_encryption_configuration" "bucketencryption" {
  bucket = aws_s3_bucket.myS3Bucket.id
  rule {
    apply_server_side_encryption_by_default {
      sse_algorithm = "AES256"
    }
  }
}

resource "aws_s3_bucket_public_access_block" "publicaccess" {
  bucket                  = aws_s3_bucket.myS3Bucket.id
  block_public_acls       = true
  block_public_policy     = true
  ignore_public_acls      = true
  restrict_public_buckets = true
}

resource "aws_s3_bucket_versioning" "versioning" {
  bucket = aws_s3_bucket.myS3Bucket.id
  versioning_configuration {
    status = "Enabled"
  }
}
```

Per Terraform, un provider definisce la risorsa, quindi gli sviluppatori dichiarano e configurano tali risorse. I fornitori sono un concetto discusso in questa guida nella sezione successiva. L'esempio Terraform crea risorse completamente separate per diverse impostazioni del bucket S3. La creazione di risorse separate per le impostazioni non è necessariamente tipica del modo in cui Terraform AWS Provider tratta le risorse. AWS Tuttavia, questo esempio mostra una distinzione importante. Sebbene una CloudFormation risorsa sia strettamente definita dalle specifiche della CloudFormation risorsa, Terraform non ha tale requisito. In Terraform, il concetto di risorsa è un po' più nebuloso.

Sebbene gli strumenti possano differire per quanto riguarda gli esatti limiti che definiscono cos'è una singola risorsa, in generale, una risorsa cloud è qualsiasi entità particolare che esiste nel cloud e che può essere creata, aggiornata o eliminata. Quindi, indipendentemente dal numero di risorse coinvolte, i due esempi precedenti creano entrambi esattamente la stessa cosa con le stesse impostazioni all'interno di un. Account AWS

Comprendere i fornitori Terraform

In Terraform, un provider è un plug-in che interagisce con provider di cloud, strumenti di terze parti e altre API. Per utilizzare Terraform con AWS, si utilizza il AWS Provider, che interagisce con le risorse. AWS

Se non hai mai utilizzato il AWS CloudFormation registro per incorporare estensioni di terze parti nei tuoi stack di distribuzione, i provider Terraform potrebbero impiegare un po' di tempo per abituarsi. Poiché CloudFormation è nativo di AWS, il provider di AWS risorse è già presente per impostazione predefinita. Terraform, d'altra parte, non ha un unico provider predefinito, quindi non si può presumere nulla sull'origine di una determinata risorsa. Ciò significa che la prima cosa che deve essere dichiarata in un file di configurazione Terraform è esattamente dove stanno andando le risorse e come ci arriveranno.

Questa distinzione aggiunge un ulteriore livello di complessità a Terraform che non esiste. CloudFormation Tuttavia, tale complessità offre una maggiore flessibilità. È possibile dichiarare più provider all'interno di un singolo modulo Terraform e quindi le risorse sottostanti create possono interagire tra loro come parte dello stesso livello di implementazione.

Questo può essere utile in molti modi. I provider non devono necessariamente rivolgersi a provider cloud separati. I provider possono rappresentare qualsiasi fonte di risorse cloud. Prendiamo ad esempio Amazon Elastic Kubernetes Service (Amazon EKS). Quando esegui il provisioning di un cluster Amazon EKS, potresti voler utilizzare i grafici Helm per gestire le estensioni di terze parti e utilizzare Kubernetes stesso per gestire le risorse dei pod. Poiché AWSHelm e Kubernetes hanno tutti i propri provider Terraform, puoi fornire e integrare queste risorse tutte contemporaneamente e quindi passare valori tra di esse.

Nel seguente esempio di codice per Terraform, il AWS provider crea un cluster Amazon EKS, quindi le informazioni di configurazione Kubernetes risultanti vengono passate ai provider Helm e Kubernetes.

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = ">= 4.33.0"
    }

    helm = {
```

```
      source  = "hashicorp/helm"
      version = "2.12.1"
    }

    kubernetes = {
      source  = "hashicorp/kubernetes"
      version = "2.26.0"
    }
  }
  required_version = ">= 1.2.0"
}

provider "aws" {
  region = "us-west-2"
}

resource "aws_eks_cluster" "example_0" {
  name     = "example_0"
  role_arn = aws_iam_role.cluster_role.arn
  vpc_config {
    endpoint_private_access = true
    endpoint_public_access  = true
    subnet_ids              = var.subnet_ids
  }
}

locals {
  host        = aws_eks_cluster.example_0.endpoint
  certificate = base64decode(aws_eks_cluster.example_0.certificate_authority.data)
}

provider "helm" {
  kubernetes {
    host                   = local.host
    cluster_ca_certificate = local.certificate
    # exec allows for an authentication command to be run to obtain user
    # credentials rather than having them stored directly in the file
    exec {
      api_version = "client.authentication.k8s.io/v1beta1"
      args        = ["eks", "get-token", "--cluster-name",
 aws_eks_cluster.example_0.name]
      command     = "aws"
    }
```

```
  }
}

provider "kubernetes" {
  host                   = local.host
  cluster_ca_certificate = local.certificate
  exec {
    api_version = "client.authentication.k8s.io/v1beta1"
    args        = ["eks", "get-token", "--cluster-name",
 aws_eks_cluster.example_0.name]
    command     = "aws"
  }
}
```

C'è un compromesso per quanto riguarda i provider quando si tratta dei due strumenti IaC. Terraform si affida interamente a pacchetti di provider posizionati esternamente, che sono il motore che guida le sue implementazioni. CloudFormation supporta AWS internamente tutti i principali processi. Con CloudFormation, devi preoccuparti dei provider di terze parti solo se desideri incorporare un'estensione di terze parti. Ogni approccio presenta vantaggi e svantaggi. Qual è quello giusto per te non rientra nell'ambito di questa guida, ma è importante ricordare la differenza quando si valutano entrambi gli strumenti.

Utilizzo degli alias Terraform

In Terraform, puoi passare configurazioni personalizzate a ciascun provider. Quindi cosa succede se si desidera utilizzare più configurazioni di provider all'interno dello stesso modulo? In tal caso dovresti usare un alias. Gli alias consentono di selezionare il provider da utilizzare a livello di risorsa o modulo. Quando hai più di un'istanza dello stesso provider, usi un alias per definire le istanze non predefinite. Ad esempio, l'istanza del provider predefinita potrebbe essere specifica Regione AWS, ma si utilizzano alias per definire regioni alternative.

Il seguente esempio di Terraform mostra come utilizzare un alias per effettuare il provisioning di bucket in diversi. Regioni AWS La regione predefinita per il provider è`us-west-2`, ma è possibile utilizzare l'alias east per il provisioning delle risorse. `us-east-2`

```
provider "aws" {
  region = "us-west-2"
}

provider "aws" {
```

```
  alias  = "east"
  region = "us-east-2"
}

resource "aws_s3_bucket" "myWestS3Bucket" {
  bucket = "my-west-s3-bucket"
}

resource "aws_s3_bucket" "myEastS3Bucket" {
  provider = aws.east
  bucket   = "my-east-s3-bucket"
}
```

Quando utilizzate un `alias` insieme al `provider` meta-argomento, come mostrato nell'esempio precedente, potete specificare una configurazione del provider diversa per risorse specifiche. Il provisioning di più risorse Regioni AWS in un unico stack è solo l'inizio. I provider di aliasing sono incredibilmente convenienti in molti modi.

Ad esempio, è molto comune effettuare il provisioning di più cluster Kubernetes contemporaneamente. Gli alias possono aiutarti a configurare provider Helm e Kubernetes aggiuntivi in modo da poter utilizzare questi strumenti di terze parti in modo diverso per diverse risorse Amazon EKS. Il seguente esempio di codice Terraform illustra come utilizzare gli alias per eseguire questa attività.

```
resource "aws_eks_cluster" "example_0" {
  name     = "example_0"
  role_arn = aws_iam_role.cluster_role.arn
  vpc_config {
    endpoint_private_access = true
    endpoint_public_access  = true
    subnet_ids              = var.subnet_ids[0]
  }
}

resource "aws_eks_cluster" "example_1" {
  name     = "example_1"
  role_arn = aws_iam_role.cluster_role.arn
  vpc_config {
    endpoint_private_access = true
    endpoint_public_access  = true
    subnet_ids              = var.subnet_ids[1]
  }
```

```
}

locals {
  host         = aws_eks_cluster.example_0.endpoint
  certificate  = base64decode(aws_eks_cluster.example_0.certificate_authority.data)
  host1        = aws_eks_cluster.example_1.endpoint
  certificate1 = base64decode(aws_eks_cluster.example_1.certificate_authority.data)
}

provider "helm" {
  kubernetes {
    host                   = local.host
    cluster_ca_certificate = local.certificate
    exec {
      api_version = "client.authentication.k8s.io/v1beta1"
      args        = ["eks", "get-token", "--cluster-name",
 aws_eks_cluster.example_0.name]
      command     = "aws"
    }
  }
}

provider "helm" {
  alias = "helm1"
  kubernetes {
    host                   = local.host1
    cluster_ca_certificate = local.certificate1
    exec {
      api_version = "client.authentication.k8s.io/v1beta1"
      args        = ["eks", "get-token", "--cluster-name",
 aws_eks_cluster.example_1.name]
      command     = "aws"
    }
  }
}

provider "kubernetes" {
  host                   = local.host
  cluster_ca_certificate = local.certificate
  exec {
    api_version = "client.authentication.k8s.io/v1beta1"
    args        = ["eks", "get-token", "--cluster-name",
 aws_eks_cluster.example_0.name]
```

```
    command     = "aws"
  }
}

provider "kubernetes" {
  alias                  = "kubernetes1"
  host                   = local.host1
  cluster_ca_certificate = local.certificate1
  exec {
    api_version = "client.authentication.k8s.io/v1beta1"
    args        = ["eks", "get-token", "--cluster-name",
 aws_eks_cluster.example_1.name]
    command     = "aws"
  }
}
```

Comprensione dei moduli Terraform

Nell'ambito dell'infrastruttura come codice (IaC), un modulo è un blocco di codice autonomo che viene isolato e impacchettato per il riutilizzo. Il concetto di moduli è un aspetto inevitabile dello sviluppo di Terraform. Per ulteriori informazioni, consulta Moduli nella documentazione di Terraform. AWS CloudFormation supporta anche i moduli. Per ulteriori informazioni, consulta Introduzione ai AWS CloudFormation moduli nel blog AWS Cloud Operations and Migrations.

La differenza principale tra i moduli di Terraform e CloudFormation è che CloudFormation i moduli vengono importati utilizzando un tipo di risorsa speciale ()`AWS::CloudFormation::ModuleVersion`. In Terraform, ogni configurazione ha almeno un modulo, noto come modulo root. Le risorse Terraform che si trovano nel file.tf principale o i file in un file di configurazione Terraform sono considerate nel modulo root. Il modulo root può quindi chiamare altri moduli per l'inclusione nello stack. L'esempio seguente mostra un modulo root che fornisce un cluster Amazon Elastic Kubernetes Service (Amazon EKS) utilizzando il modulo eks open source.

```
terraform {
  required_providers {
    helm = {
      source  = "hashicorp/helm"
      version = "2.12.1"
    }
  }
  required_version = ">= 1.2.0"
}

module "eks" {
  source  = "terraform-aws-modules/eks/aws"
  version = "20.2.1"
  vpc_id  = var.vpc_id
}

provider "helm" {
  kubernetes {
    host                   = module.eks.cluster_endpoint
    cluster_ca_certificate =
 base64decode(module.eks.cluster_certificate_authority_data)
  }
}
```

Avrai notato che il file di configurazione sopra riportato non include il Provider. AWS Questo perché i moduli sono autonomi e possono includere i propri provider. Poiché i provider Terraform sono globali, i provider di un modulo figlio possono essere utilizzati nel modulo root. Tuttavia, questo non è vero per tutti i valori dei moduli. Gli altri valori interni all'interno di un modulo rientrano di default solo nell'ambito di quel modulo e devono essere dichiarati come output per essere accessibili nel modulo root. Puoi sfruttare i moduli open source per semplificare la creazione di risorse all'interno del tuo stack. Ad esempio, il modulo eks non si limita a fornire un cluster EKS: fornisce un ambiente Kubernetes perfettamente funzionante. Usarlo può evitarti di scrivere dozzine di righe di codice aggiuntive, a condizione che la configurazione del modulo eks soddisfi le tue esigenze.

Moduli di chiamata

Due dei comandi CLI Terraform principali che esegui durante l'implementazione di Terraform sono terraform init e terraform apply. Uno dei passaggi predefiniti eseguiti dal `terraform init` comando consiste nell'individuare tutti i moduli figlio e importarli come dipendenze nella directory. `.terraform/modules` Durante lo sviluppo, ogni volta che aggiungete un nuovo modulo di origine esterna, dovete reinizializzarlo prima di utilizzare il comando. `apply` Quando senti un riferimento a un modulo Terraform, si riferisce ai pacchetti in questa directory. A rigor di termini, il modulo che dichiari nel codice è il modulo chiamante, quindi in pratica, la parola chiave module chiama il modulo effettivo, che viene memorizzato come dipendenza.

In questo modo, il modulo chiamante funge da rappresentante più succinto del modulo completo da sostituire al momento della distribuzione. Puoi sfruttare questa idea creando i tuoi moduli all'interno degli stack per applicare la separazione logica delle risorse utilizzando qualsiasi criterio desideri. Ricorda solo che l'obiettivo finale di questa operazione dovrebbe essere quello di ridurre la complessità dello stack. Poiché la condivisione dei dati tra i moduli richiede l'output dei dati dall'interno del modulo, a volte affidarsi troppo ai moduli può complicare eccessivamente le cose.

Il modulo root

Poiché ogni configurazione di Terraform ha almeno un modulo, può essere utile esaminare le proprietà del modulo con cui avrai a che fare di più: il modulo root. Ogni volta che lavori su un progetto Terraform, il modulo root è composto da tutti i `.tf` (`o.tf.json`) file nella tua directory di primo livello. Quando esegui `terraform apply` in quella directory di primo livello, Terraform tenta di eseguire ogni `.tf` file che trova lì. Tutti i file nelle sottodirectory vengono ignorati a meno che non vengano chiamati in uno di questi file di configurazione di primo livello.

Ciò offre una certa flessibilità nel modo in cui strutturate il codice. È anche il motivo per cui è più preciso fare riferimento alla distribuzione di Terraform come modulo piuttosto che come file, perché diversi file potrebbero essere coinvolti in un unico processo. Esiste una struttura di moduli standard che Terraform consiglia per le migliori pratiche. Tuttavia, se dovessi inserire un `.tf` file nella directory di primo livello, verrà eseguito insieme al resto dei file. In effetti, tutti i `.tf` file di primo livello in un modulo vengono distribuiti durante l'esecuzione. `terraform apply` Quindi quale file viene eseguito per primo Terraform? La risposta a questa domanda è molto importante.

Terraform esegue una serie di passaggi dopo l'inizializzazione e prima della distribuzione dello stack. Innanzitutto, vengono analizzate le configurazioni esistenti e quindi viene creato un grafico delle dipendenze. Il grafico delle dipendenze determina quali risorse sono necessarie e in quale ordine devono essere indirizzate. Le risorse che contengono proprietà a cui si fa riferimento in altre risorse, ad esempio, verrebbero gestite prima delle risorse dipendenti. Allo stesso modo, le risorse che dichiarano esplicitamente la dipendenza utilizzando il `depends_on` parametro verrebbero gestite dopo le risorse da esse specificate. Quando possibile, Terraform può implementare il parallelismo e gestire contemporaneamente risorse non dipendenti. È possibile visualizzare il grafico delle dipendenze prima della distribuzione utilizzando il comando terraform graph.

Dopo la creazione del grafico delle dipendenze, Terraform determina cosa deve essere fatto durante l'implementazione. Confronta il grafico delle dipendenze con il file di stato più recente. Il risultato di questo processo si chiama piano ed è molto simile a un set di CloudFormation modifiche. Puoi vedere il piano corrente usando il comando terraform plan.

Come buona pratica, si consiglia di rimanere il più vicino possibile alla struttura del modulo standard. Nei casi in cui i file di configurazione diventano troppo lunghi per essere gestiti in modo efficiente e le separazioni logiche potrebbero semplificare la gestione, è possibile distribuire il codice su più file. Tieni presente come funzionano il grafico delle dipendenze e il processo di pianificazione per far funzionare gli stack nel modo più efficiente possibile.

Comprensione degli stati e dei backend di Terraform

Uno dei concetti più importanti nell'infrastruttura come codice (IaC) è il concetto di stato. I servizi IaC mantengono lo stato, che consente di dichiarare una risorsa in un file IAc senza doverla ricreare ogni volta che si distribuisce. I file IaC documentano lo stato di tutte le risorse al termine di una distribuzione in modo che possa quindi confrontare tale stato con lo stato di destinazione, come dichiarato nella distribuzione successiva. Quindi, se lo stato corrente contiene un bucket Amazon Simple Storage Service (Amazon S3) `my-s3-bucket` denominato e le modifiche in arrivo contengono anche lo stesso bucket, il nuovo processo applicherà tutte le modifiche rilevate al bucket esistente anziché provare a creare un bucket completamente nuovo.

La tabella seguente fornisce esempi del processo generale di stato IAc.

Stato attuale	Stato di destinazione	Azione
Nessun bucket S3 denominato `my-s3-bucket`	Bucket S3 denominato `my-s3-bucket`	Crea un bucket S3 denominato `my-s3-bucket`
`my-s3-bucket` senza che sia configurato il controllo delle versioni del bucket	`my-s3-bucket` senza che sia configurato il controllo delle versioni del bucket	Nessuna operazione
`my-s3-bucket` senza che sia configurato il controllo delle versioni del bucket	`my-s3-bucket` con il controllo delle versioni del bucket configurato	Configura `my-s3-bucket` per avere il controllo delle versioni del bucket
`my-s3-bucket` con il controllo delle versioni del bucket configurato	Nessun bucket S3 denominato `my-s3-bucket`	Tentativo di eliminazione `my-s3-bucket`

Per comprendere i diversi modi in cui AWS CloudFormation Terraform traccia lo stato, è importante ricordare la prima differenza fondamentale tra i due strumenti: CloudFormation è ospitato all'interno di e Terraform è essenzialmente remoto. Cloud AWS Questo fatto consente di CloudFormation mantenere lo stato internamente. Puoi accedere alla CloudFormation console e visualizzare la cronologia degli eventi di un determinato stack, ma il CloudFormation servizio stesso applica le regole statali per te.

Le tre modalità che CloudFormation operano in base a una determinata risorsa sono `Create`, `Update` e. `Delete` La modalità corrente viene determinata in base a ciò che è accaduto nell'ultima distribuzione e non può essere influenzata in altro modo. Forse è possibile aggiornare CloudFormation le risorse manualmente per influenzare la modalità determinata, ma non è possibile passare un comando CloudFormation che dica «Per questa risorsa, operate in `Create` modalità».

Poiché Terraform non è ospitato in Cloud AWS, il processo di mantenimento dello stato deve essere più configurabile. Per questo motivo, lo stato Terraform viene mantenuto all'interno di un file di stato generato automaticamente. Uno sviluppatore Terraform ha a che fare con lo stato in modo molto più diretto di quanto farebbe. CloudFormation La cosa importante da ricordare è che lo stato di tracciamento è altrettanto importante per entrambi gli strumenti.

Per impostazione predefinita, il file di stato Terraform viene archiviato localmente al livello superiore della directory principale che esegue lo stack Terraform. Se esegui il `terraform apply` comando dal tuo ambiente di sviluppo locale, puoi vedere Terraform generare il file terraform.tfstate che utilizza per mantenere lo stato in tempo reale. Nel bene e nel male, questo ti dà molto più controllo sullo stato in Terraform rispetto a quello che hai. CloudFormation Sebbene non dovresti mai aggiornare direttamente il file di stato, puoi eseguire diversi comandi CLI Terraform che aggiorneranno lo stato tra le distribuzioni. Ad esempio, terraform import ti consente di aggiungere risorse create al di fuori di Terraform allo stack di implementazione. Al contrario, puoi rimuovere una risorsa dallo stato eseguendo terraform state rm.

Il fatto che Terraform debba memorizzare il suo stato da qualche parte porta a un altro concetto che non si applica a CloudFormation: il backend. Un backend Terraform è il luogo in cui uno stack Terraform memorizza il proprio file di stato dopo la distribuzione. Questo è anche il luogo in cui si aspetta di trovare il file di stato all'inizio di una nuova distribuzione. Quando esegui lo stack localmente, come descritto sopra, puoi conservare una copia dello stato di Terraform nella directory locale di primo livello. Questo è noto come backend locale.

Quando si sviluppa per un ambiente di integrazione e distribuzione continue (CI/CD), il file di stato locale viene generalmente incluso nel file.gitignore per non consentirne il controllo della versione. Quindi non è presente alcun file di stato locale all'interno della pipeline. Per funzionare correttamente, quella fase della pipeline deve trovare da qualche parte il file di stato corretto.Questo è il motivo per cui i file di configurazione Terraform contengono spesso un blocco di backend. Il blocco di backend indica allo stack Terraform che deve cercare da qualche parte oltre alla propria directory di primo livello per trovare il file di stato.

Un backend Terraform può essere posizionato quasi ovunque: un bucket Amazon S3, un endpoint API o persino uno spazio di lavoro Terraform remoto. Di seguito è riportato un esempio di backend Terraform archiviato in un bucket Amazon S3.

```
terraform {
  backend "s3" {
    bucket = "my-s3-bucket"
    key    = "state-file-folder"
    region = "us-east-1"
  }
}
```

Per evitare di archiviare informazioni sensibili all'interno dei file di configurazione Terraform, i backend supportano anche configurazioni parziali. Nell'esempio precedente, le credenziali necessarie per accedere al bucket non sono presenti nella configurazione. Le credenziali possono essere ottenute da variabili di ambiente o utilizzando altri mezzi, ad esempio. Gestione dei segreti AWS Per ulteriori informazioni, consulta Proteggere i dati sensibili utilizzando Gestione dei segreti AWS e HashiCorp Terraform.

Uno scenario di backend comune è un backend locale utilizzato nell'ambiente locale a scopo di test. Il file terraform.tfstate è incluso nel file.gitignore in modo che non venga inviato al repository remoto. Quindi, ogni ambiente all'interno della pipeline CI/CD manterrebbe il proprio backend. In questo scenario, più sviluppatori potrebbero avere accesso a questo stato remoto, quindi è consigliabile proteggere l'integrità del file di stato. Se più distribuzioni sono in esecuzione e aggiornano lo stato contemporaneamente, il file di stato potrebbe danneggiarsi. Per questo motivo, in situazioni con backend non locali, il file di stato viene in genere bloccato durante la distribuzione.

Comprensione delle fonti di dati Terraform

È molto comune che gli stack di implementazione si basino su dati provenienti da risorse precedentemente esistenti. La maggior parte degli strumenti IaC consente di importare risorse create con altri processi. Queste risorse importate sono generalmente di sola lettura (sebbene i ruoli IAM rappresentino un'eccezione notevole) e vengono utilizzate per accedere ai dati necessari alle risorse all'interno dello stack. AWS CloudFormation consente di importare risorse, ma questa idea può essere spiegata meglio esaminando. AWS Cloud Development Kit (AWS CDK)

AWS CDK Aiuta gli sviluppatori a utilizzare i linguaggi di programmazione esistenti per generare CloudFormation modelli. Il risultato finale di un' AWS CDK operazione è una risorsa importata in CloudFormation. Tuttavia, la sintassi utilizzata con AWS CDK rende più semplice il confronto con Terraform. Ecco un esempio di importazione di una risorsa utilizzando. AWS CDK

```
const importedBucket: IBucket = Bucket.fromBucketAttributes(
    scope,
    "imported-bucket",
    {
        bucketName: "My_S3_Bucket"
    }
);
```

Una risorsa importata viene in genere creata chiamando un metodo statico sulla stessa classe utilizzata per creare una nuova risorsa dello stesso tipo. La chiamata `new Bucket(...` creerebbe una nuova risorsa e la chiamata ne `Bucket.fromBucketAttributes(...` importerebbe una esistente. Passi un sottoinsieme delle proprietà del bucket alla funzione in modo che AWS CDK possa trovare il bucket giusto. Un'altra differenza, tuttavia, è che la creazione di un nuovo bucket restituisce un'istanza completa della `Bucket` classe, con tutte le proprietà e i metodi disponibili all'interno. L'importazione della risorsa restituisce un`IBucket`, che è un tipo che contiene solo le proprietà che `Bucket` deve avere. Sebbene sia possibile importare una risorsa da uno stack esterno, le opzioni su come utilizzarla sono limitate.

In Terraform, un obiettivo simile viene raggiunto utilizzando fonti di dati. La maggior parte delle risorse Terraform definite dispone di una fonte di dati di accompagnamento. Di seguito è riportato un esempio di risorsa bucket Terraform S3 seguita dalla fonte di dati corrispondente.

```
# S3 Bucket resource:
resource "aws_s3_bucket" "My_S3_Bucket" {
```

```
  bucket = "My_S3_Bucket"
}

# S3 Bucket data source:
data "aws_s3_bucket" "My_S3_Bucket" {
  bucket = "My_S3_Bucket"
}
```

L'unica differenza tra questi due elementi è il prefisso del nome. Come illustrato nella documentazione relativa a un'origine dati, è possibile passare a un'origine dati un numero inferiore di parametri rispetto a una risorsa. Questo perché la risorsa utilizza questi parametri per dichiarare tutte le proprietà di un nuovo bucket S3, mentre la fonte di dati richiede solo le informazioni sufficienti per identificare e importare in modo univoco i dati di una risorsa esistente.

La somiglianza tra la sintassi di una risorsa Terraform e una fonte di dati può essere comoda, ma può anche essere problematica. È normale che gli sviluppatori Terraform alle prime armi utilizzino accidentalmente una fonte di dati anziché una risorsa nella loro configurazione. Le fonti di dati Terraform sono sempre di sola lettura. Puoi usarli al posto della risorsa corrispondente per azioni di lettura (come fornire un nome ID a un'altra risorsa). Tuttavia, non è possibile utilizzarli per azioni di scrittura, che modificano radicalmente alcuni aspetti della risorsa sottostante. Per questo motivo, puoi pensare a un'origine dati Terraform come a una versione clonata della risorsa sottostante.

Analogamente al precedente esempio di AWS CDK iBucket, le fonti di dati sono utili per scenari di sola lettura. Se hai bisogno di ottenere dati da una risorsa esistente ma non hai bisogno di mantenerla all'interno del tuo stack, utilizza una fonte di dati. Un buon esempio di ciò è quando crei un'istanza Amazon EC2 che utilizza il VPC predefinito dell'account. Poiché quel VPC esiste già, tutto ciò che devi fare è inserirne i dati. Il seguente esempio di codice mostra come utilizzare i dati per identificare il VPC di destinazione.

```
data "aws_vpc" "default" {
  default = true
}

resource "aws_instance" "instance1" {
  ami           = "ami-123456"
  instance_type = "t2.micro"
  subnet_id     = data.aws_vpc.default.main_route_table_id
}
```

Comprensione delle variabili Terraform, dei valori locali e degli output

Le variabili migliorano la flessibilità del codice consentendo l'inserimento di segnaposti all'interno di blocchi di codice. Le variabili possono rappresentare valori diversi ogni volta che il codice viene riutilizzato. Terraform distingue tra i suoi tipi di variabili in base al loro ambito modulare. Le variabili di input sono valori esterni che possono essere inseriti in un modulo, i valori di output sono valori interni che possono essere condivisi esternamente e i valori locali rimangono sempre all'interno del loro ambito originale.

Variabili

AWS CloudFormation utilizza i parametri per rappresentare valori personalizzati che possono essere impostati e reimpostati da una distribuzione dello stack all'altra. Allo stesso modo, Terraform utilizza variabili di input o variabili. Le variabili possono essere dichiarate ovunque in un file di configurazione Terraform e di solito sono dichiarate con il tipo di dati richiesto o il valore predefinito. Tutte e tre le seguenti espressioni sono dichiarazioni di variabili Terraform valide.

```
variable "thing_i_made_up" {
  type = string
}

variable "random_number" {
  default = 5
}

variable "dogs" {
  type = list(object({
    name  = string
    breed = string
  }))

  default = [
    {
      name  = "Sparky",
      breed = "poodle"
    }
  ]
```

```
}
```

Per accedere alla razza di Sparky all'interno della configurazione, dovresti usare la variabile. `var.dogs[0].breed` Se una variabile non ha valori predefiniti e non è classificata come annullabile, il valore della variabile deve essere impostato per ogni distribuzione. Altrimenti, è facoltativo impostare un nuovo valore per la variabile. In un modulo root, puoi impostare i valori delle variabili correnti sulla riga di comando, come variabili di ambiente o nel file terraform.tfvars. L'esempio seguente mostra come inserire i valori delle variabili nel file terraform.tfvars, che è archiviato nella directory di primo livello del modulo.

```
# terraform.tfvars
dogs = [
   {
      name  = "Sparky",
      breed = "poodle"
   },
   {
      name  = "Fluffy",
      breed = "chihuahua"
   }
]

random_number = 7

thing_i_made_up = "Kabibble"
```

Il valore del file `dogs` terraform.tfvars di esempio sovrascriverebbe il valore predefinito nella dichiarazione della variabile. Se stai dichiarando variabili all'interno di un modulo figlio, puoi impostare i valori delle variabili direttamente all'interno del blocco di dichiarazione del modulo, come mostrato nell'esempio seguente.

```
module "my_custom_module" {
  source        = "modulesource/custom"
  version       = "0.0.1"
  random_number = 8
}
```

Alcuni degli altri argomenti che è possibile utilizzare per dichiarare una variabile includono:

- `sensitive`- Impostandola per `true` evitare che il valore della variabile venga esposto negli output del processo Terraform.
- `nullable`— L'impostazione di questa opzione `true` consente alla variabile di non avere alcun valore. Ciò è utile per le variabili in cui non è impostato un valore predefinito.
- `description`— Aggiungere una descrizione della variabile ai metadati dello stack.
- `validation`— Imposta le regole di convalida per la variabile.

Uno degli aspetti più convenienti delle variabili Terraform è la possibilità di aggiungere uno o più oggetti di convalida all'interno della dichiarazione della variabile. È possibile utilizzare oggetti di convalida per aggiungere una condizione che la variabile deve superare, altrimenti la distribuzione fallisce. Puoi anche impostare un messaggio di errore personalizzato da mostrare ogni volta che la condizione viene violata.

Ad esempio, stai configurando un file di configurazione Terraform che verrà eseguito dai membri del tuo team. Prima di distribuire gli stack, un membro del team deve creare un file terraform.tfvars per impostare un valore di configurazione importante. Per ricordarglielo, puoi fare qualcosa di simile a quanto segue.

```
variable "important_config_setting" {
  type = string

  validation {
    condition     = length(var.important_config_setting) > 0
    error_message = "Don't forget to create the terraform.tfvars file!"
  }

  validation {
    condition     = substr(var.important_config_setting, 0, 7) == "prefix-"
    error_message = "Remember that the value always needs to start with 'prefix-'"
  }
}
```

Come mostrato in questo esempio, puoi impostare più condizioni all'interno di una singola variabile. Terraform mostra solo messaggi di errore per condizioni non riuscite. In questo modo, puoi applicare tutti i tipi di regole sui valori delle variabili. Se un valore variabile causa un errore nella pipeline, sapresti esattamente perché.

Valori locali

Se ci sono valori all'interno di un modulo a cui desideri assegnare un alias, usa la `locals` parola chiave anziché dichiarare una variabile predefinita che non verrà mai aggiornata. Come suggerisce il nome, un `locals` blocco contiene termini che rientrano nell'ambito interno di quel modulo specifico. Se desideri trasformare un valore di stringa, ad esempio aggiungendo un prefisso a un valore variabile da utilizzare nel nome di una risorsa, l'utilizzo di un valore locale potrebbe essere una buona soluzione. Un singolo `locals` blocco può dichiarare tutti i valori locali per il modulo, come illustrato nell'esempio seguente.

```
locals {
  moduleName    = "My Module"
  localConfigId = concat("prefix-", var.important_config_setting)
}
```

Ricorda solo che quando accedi al valore, la `locals` parola chiave diventa singolare, ad esempio. `local.LocalConfigId`

Valori di output

Se le variabili di input Terraform sono come CloudFormation i parametri, allora potresti dire che i valori di output di Terraform sono come CloudFormation gli output. Entrambi vengono utilizzati per esporre valori dall'interno di uno stack di distribuzione. Tuttavia, poiché il modulo Terraform è più radicato nella struttura dello strumento, i valori di output di Terraform vengono utilizzati anche per esporre i valori all'interno di un modulo a un modulo principale o ad altri moduli figlio, anche se tali moduli si trovano tutti all'interno dello stesso stack di distribuzione. Se stai creando due moduli personalizzati e il primo modulo deve accedere al valore ID del secondo modulo, dovrai aggiungere il blocco seguente `output` al secondo modulo.

```
output "module_id" {
  value = local.module_id
}
Then in the first module you could use it like this:
module "first_module" {
  source = "path/to/first/module"
}

resource "example_resource" "example_resource_name" {
```

```
  module_id = module.first_module.module_id
}
```

Poiché i valori di output di Terraform possono essere utilizzati all'interno dello stesso stack, puoi anche utilizzare l'`sensitive`attributo in un `output` blocco per impedire che il valore venga visualizzato nell'output dello stack. Inoltre, un `output` blocco può utilizzare i `precondition` blocchi nello stesso modo in cui le variabili utilizzano i `validation` blocchi: per garantire che le variabili seguano un determinato insieme di regole. Questo aiuta a garantire che tutti i valori all'interno di un modulo esistano come previsto prima di procedere con la distribuzione.

```
output "important_config_setting" {
  value = var.important_config_setting

  precondition {
    condition     = length(var.important_config_setting) > 0
    error_message = "You forgot to create the terraform.tfvars file again."
  }
}
```

Comprensione delle funzioni, delle espressioni e dei meta-argomenti di Terraform

Una critica agli strumenti IaC che utilizzano file di configurazione dichiarativi anziché linguaggi di programmazione comuni è che rendono più difficile l'implementazione di una logica programmatica personalizzata. Nelle configurazioni Terraform, questo problema viene risolto utilizzando funzioni, espressioni e meta-argomenti.

Funzioni

Uno dei grandi vantaggi dell'utilizzo del codice per il provisioning dell'infrastruttura è la possibilità di archiviare flussi di lavoro comuni e riutilizzarli più e più volte, spesso passando argomenti diversi ogni volta. Le funzioni Terraform sono simili alle funzioni AWS CloudFormation intrinseche, sebbene la loro sintassi sia più simile a come le funzioni vengono chiamate nei linguaggi programmatici. Potresti aver già notato alcune funzioni Terraform, come substr, concat, length e base64decode, negli esempi di questa guida. Come per CloudFormation le funzioni intrinseche, Terraform ha una serie di funzioni integrate che possono essere utilizzate nelle configurazioni. Ad esempio, se un particolare attributo di risorsa richiede un oggetto JSON molto grande che sarebbe inefficiente incollarlo direttamente nel file, è possibile inserire l'oggetto in un file.json e utilizzare le funzioni Terraform per accedervi. Nell'esempio seguente, la `file` funzione restituisce il contenuto del file sotto forma di stringa, quindi lo converte in un tipo di oggetto. `jsondecode`

```
resource "example_resource" "example_resource_name" {
  json_object = jsondecode(file("/path/to/file.json"))
}
```

Espressioni

Terraform consente anche espressioni condizionali, che sono simili alle CloudFormation `condition` funzioni tranne per il fatto che utilizzano la più tradizionale sintassi dell'operatore ternario. Nell'esempio seguente, le due espressioni restituiscono esattamente lo stesso risultato. Il secondo esempio è ciò che Terraform chiama espressione splat. L'asterisco fa sì che Terraform scorra l'elenco e crei un nuovo elenco utilizzando solo la `id` proprietà di ciascun elemento.

```
resource "example_resource" "example_resource_name" {
  boolean_value  = var.value ? true : false
```

```
  numeric_value  = var.value > 0 ? 1 : 0
  string_value   = var.value == "change_me" ? "New value" : var.value
  string_value_2 = var.value != "change_me" ? var.value : "New value"
}
There are two ways to express for loops in a Terraform configuration:
resource "example_resource" "example_resource_name" {
  list_value   = [for object in var.ids : object.id]
  list_value_2 = var.ids[*].id
}
```

Meta-argomenti

Nell'esempio di codice precedente, *`list_value`* e *`list_value_2`* sono indicati come argomenti. Alcuni di questi meta-argomenti potrebbero già essere familiari. Terraform ha anche alcuni meta-argomenti, che agiscono proprio come argomenti ma con alcune funzionalità extra:

- Il meta-argomento depends_on è molto simile all'attributo. CloudFormation DependsOn
- Il meta-argomento del provider consente di utilizzare più configurazioni di provider contemporaneamente.
- Il meta-argomento del ciclo di vita consente di personalizzare le impostazioni delle risorse, in modo simile alle politiche di rimozione ed eliminazione di. CloudFormation

Altri meta-argomenti consentono di aggiungere funzionalità di funzioni ed espressioni direttamente a una risorsa. Ad esempio, il meta-argomento count è un meccanismo utile per creare più risorse simili contemporaneamente. L'esempio seguente dimostra come creare due cluster Amazon Elastic Container Service (Amazon EKS) senza utilizzare il `count` meta-argomento.

```
resource "aws_eks_cluster" "example_0" {
  name     = "example_0"
  role_arn = aws_iam_role.cluster_role.arn
  vpc_config {
    endpoint_private_access = true
    endpoint_public_access  = true
    subnet_ids              = var.subnet_ids[0]
  }
}

resource "aws_eks_cluster" "example_1" {
  name     = "example_1"
```

```
  role_arn = aws_iam_role.cluster_role.arn
  vpc_config {
    endpoint_private_access = true
    endpoint_public_access  = true
    subnet_ids              = var.subnet_ids[1]
  }
}
```

L'esempio seguente mostra come utilizzare il `count` meta-argomento per creare due cluster Amazon EKS.

```
resource "aws_eks_cluster" "clusters" {
  count    = 2
  name     = "cluster_${count.index}"
  role_arn = aws_iam_role.cluster_role.arn
  vpc_config {
    endpoint_private_access = true
    endpoint_public_access  = true
    subnet_ids              = var.subnet_ids[count.index]
  }
}
```

Per assegnare a ciascuna unità un nome, puoi accedere all'indice dell'elenco all'interno del blocco di risorse all'indirizzo. `count.index` Ma cosa succede se si desidera creare più risorse simili ma un po' più complesse? È qui che entra in gioco il meta-argomento for_each. Il `for_each` meta-argomento è molto simile `acount`, tranne per il fatto che si passa una lista o un oggetto anziché un numero. Terraform crea una nuova risorsa per ogni membro dell'elenco o dell'oggetto. È simile a se si impostacount `= length(list)`, tranne per il fatto che è possibile accedere al contenuto dell'elenco anziché all'indice del ciclo.

Funziona sia per un elenco di elementi che per un singolo oggetto. L'esempio seguente creerebbe due risorse con `id-0` e `id-1` come loro IDs.

```
variable "ids" {
  default = [
    { id = "id-0" },
    { id = "id-1" },
  ]
}

resource "example_resource" "example_resource_name" {
```

```
  # If your list fails, you might have to call "toset" on it to convert it to a set
  for_each = toset(var.ids)
  id       = each.value
}
```

L'esempio seguente creerebbe anche due risorse, una per Sparky, il barboncino, e una per Fluffy, il chihuahua.

```
variable "dogs" {
  default = {
    poodle    = "Sparky"
    chihuahua = "Fluffy"
  }
}

resource "example_resource" "example_resource_name" {
  for_each = var.dogs
  breed    = each.key
  name     = each.value
}
```

Proprio come puoi accedere all'indice del ciclo in count usando count.index, puoi accedere alla chiave e al valore di ogni elemento in un ciclo for_each utilizzando l'oggetto each. Poiché for_each esegue iterazioni sia sugli elenchi che sugli oggetti, tenere traccia di ogni chiave e valore può creare un po' di confusione. La tabella seguente mostra i diversi modi in cui è possibile utilizzare il meta-argomento for_each e come fare riferimento ai valori a ogni iterazione.

Esempio	**`for_each`** tipo	Prima iterazione	Seconda iterazione

A	`["poodle", "chihuahua"]`	`each.key = "poodle"`
		
		`each.value = null`
			`each.key = "chihuahua"`
			
			`each.value = null`

B	`[`
	
	`{`
		`each.key = {`
		
		`type = "poodle",`
			`each.key = {`
			
			`type = "chihuahua",`

Esempio	**for_each** tipo	Prima iterazione	Seconda iterazione
	`type = "poodle", name = "Sparky" }, { type = "chihuahua", name = "Fluffy" } ]`	`name = "Sparky" } each.value = null`	`name = "Fluffy" } each.value = null`
C	`{ poodle = "Sparky", chihuahua = "Fluffy" }`	`each.key = "poodle" each.value = "Sparky"`	`each.key = "chihuahua" each.value = "Fluffy"`

Esempio	**`for_each`** tipo	Prima iterazione	Seconda iterazione

D	`{`
	
	`dogs = {`
	
	`poodle =`
	` "Sparky",`
	
	`chihuahua =`
	` "Fluffy"`
	
	`},`
	
	`cats = {`
	
	`persian =`
	` "Felix",`
	
	`burmese =`
	` "Morris"`
	
	`}`
	
	`}`
		`each.key = "dogs"`
		
		`each.value = {`
		
		`poodle =`
		` "Sparky",`
		
		`chihuahua =`
		` "Fluffy"`
		
		`}`
			`each.key = "cats"`
			
			`each.value = {`
			
			`persian =`
			` "Felix",`
			
			`burmese =`
			` "Morris"`
			
			`}`

Esempio	**`for_each`** tipo	Prima iterazione	Seconda iterazione

E	`{`
	
	`dogs = [`
	
	`{`
	
	`type = "poodle",`
	
	`name = "Sparky"`
	
	`},`
	
	`{`
	
	`type = "chihuahua",`
	
	`name = "Fluffy"`
	
	`}`
	
	`],`
	
	`cats = [`
	
	`{`
	
	`type = "persian",`
	
	`name = "Felix"`
	
	`},`
	
	`{`
	
	`type = "burmese",`
		`each.key = "dogs"`
		
		`each.value = [`
		
		`{`
		
		`type = "poodle",`
		
		`name = "Sparky"`
		
		`},`
		
		`{`
		
		`type = "chihuahua",`
		
		`name = "Fluffy"`
		
		`}`
		
		`]`
			`each.key = "cats"`
			
			`each.value = [`
			
			`{`
			
			`type = "persian",`
			
			`name = "Felix"`
			
			`},`
			
			`{`
			
			`type = "burmese",`
			
			`name = "Morris"`
			
			`}`
			
			`]`

Esempio	**for_each** tipo	Prima iterazione	Seconda iterazione

	`name = "Morris"`
	
	`}`
	
	`]`
	
	`}`		

Quindi, se `var.animals` fosse uguale alla riga E, allora potresti creare una risorsa per animale usando il seguente codice.

```
resource "example_resource" "example_resource_name" {
  for_each = var.animals
  type     = each.key
  breeds   = each.value[*].type
  names    = each.value[*].name
}
```

In alternativa, è possibile creare due risorse per animale utilizzando il codice seguente.

```
resource "example_resource" "example_resource_name" {
  for_each = var.animals.dogs
  type     = "dogs"
  breeds   = each.value.type
  names    = each.value.name
}

resource "example_resource" "example_resource_name" {
  for_each = var.animals.cats
  type     = "cats"
  breeds   = each.value.type
  names    = each.value.name
}
```

Domande frequenti

Quando dovrei usare Terraform invece di? CloudFormation

In generale, se i tuoi carichi di lavoro sono basati principalmente su AWS, AWS CloudFormation fornisce un livello di supporto nativo che Terraform non può eguagliare. Tuttavia, se i tuoi carichi di lavoro includono diversi processi di terze parti o sono distribuiti tra più provider di cloud, Terraform è uno strumento che potresti prendere in considerazione.

Quando devo usare AWS CDK invece di? CloudFormation

Quando usi il AWS Cloud Development Kit (AWS CDK), stai usando anche CloudFormation. Ti AWS CDK consente di utilizzare un linguaggio di programmazione comune per generare CloudFormation modelli. Se hai esperienza in uno qualsiasi dei linguaggi di programmazione AWS CDK supportati, AWS CDK puoi ridurre il tempo necessario per generare i CloudFormation modelli.

Esiste uno strumento come AWS CDK questo che genera configurazioni Terraform?

Rispetto a AWS CDK, il CDK for Terraform (CDKTF) utilizza la stessa libreria di costrutti per fornire risorse e lo stesso motore jsii per supportare più linguaggi di programmazione. Puoi usarlo per generare configurazioni Terraform nello stesso modo in cui genera modelli. AWS CDK CloudFormation

Come posso saperne di più su Terraform?

Per ulteriori informazioni sui concetti avanzati di Terraform, consulta la documentazione di Terraform. Descrive inoltre i componenti di tutti i principali provider e moduli open source.

Risorse correlate

AWS documentazione

- Documentazione di AWS CDK
- Documentazione di AWS CloudFormation
- Terraform: Beyond the Basics con AWS (AWS post sul blog)

Altre risorse

- Documentazione CDK per Terraform
- Documentazione Terraform

Appendice: esempi di accesso agli attributi Terraform

Risorsa

```
resource "aws_s3_bucket" "myS3Bucket" {
     bucket = "my-s3-bucket"
}

bucketName = aws_s3_bucket.myS3Bucket.bucket
```

Origine dati

```
data "aws_s3_bucket" "myS3Bucket" {
     bucket = "my-s3-bucket"
}

bucketName = data.aws_s3_bucket.myS3Bucket.bucket
```

Modulo

```
module "eks" {
    source = "terraform-aws-modules/eks/aws"
    version = "20.2.1"
}

vpc_id = module.eks.vpc_id
```

Variabile

```
variable "my_variable" = {
   default = "dog"
}

animalType = var.my_variable
```

Locale

```
locals {
  type = "dog"
}

animalType = local.type
```

Cronologia dei documenti

La tabella seguente descrive le modifiche significative apportate a questa guida. Per ricevere notifiche sugli aggiornamenti futuri, puoi abbonarti a un feed RSS.

Modifica	Descrizione	Data
Pubblicazione iniziale	—	29 marzo 2024

AWS Glossario delle linee guida prescrittive

I seguenti sono termini di uso comune nelle strategie, nelle guide e nei modelli forniti da AWS Prescriptive Guidance. Per suggerire voci, utilizza il link Fornisci feedback alla fine del glossario.

Numeri

7 R

Sette strategie di migrazione comuni per trasferire le applicazioni sul cloud. Queste strategie si basano sulle 5 R identificate da Gartner nel 2011 e sono le seguenti:

- Rifattorizzare/riprogettare: trasferisci un'applicazione e modifica la sua architettura sfruttando appieno le funzionalità native del cloud per migliorare l'agilità, le prestazioni e la scalabilità. Ciò comporta in genere la portabilità del sistema operativo e del database. Esempio: migra il tuo database Oracle locale all'edizione compatibile con Amazon Aurora PostgreSQL.
- Ridefinire la piattaforma (lift and reshape): trasferisci un'applicazione nel cloud e introduci un certo livello di ottimizzazione per sfruttare le funzionalità del cloud. Esempio: migra il tuo database Oracle locale ad Amazon Relational Database Service (Amazon RDS) per Oracle in. Cloud AWS
- Riacquistare (drop and shop): passa a un prodotto diverso, in genere effettuando la transizione da una licenza tradizionale a un modello SaaS. Esempio: migra il tuo sistema di gestione delle relazioni con i clienti (CRM) su Salesforce.com.
- Eseguire il rehosting (lift and shift): trasferisci un'applicazione sul cloud senza apportare modifiche per sfruttare le funzionalità del cloud. Esempio: migra il tuo database Oracle locale a Oracle su un'istanza EC2 in. Cloud AWS
- Trasferire (eseguire il rehosting a livello hypervisor): trasferisci l'infrastruttura sul cloud senza acquistare nuovo hardware, riscrivere le applicazioni o modificare le operazioni esistenti. Esegui la migrazione dei server da una piattaforma locale a un servizio cloud per la stessa piattaforma. Esempio: migra un'applicazione suMicrosoft Hyper-V. AWS
- Riesaminare (mantenere): mantieni le applicazioni nell'ambiente di origine. Queste potrebbero includere applicazioni che richiedono una rifattorizzazione significativa che desideri rimandare a un momento successivo e applicazioni legacy che desideri mantenere, perché non vi è alcuna giustificazione aziendale per effettuarne la migrazione.
- Ritirare: disattiva o rimuovi le applicazioni che non sono più necessarie nell'ambiente di origine.

A

ABAC

Vedi controllo degli accessi basato sugli attributi.

servizi astratti

Vedi servizi gestiti.

ACIDO

Vedi atomicità, consistenza, isolamento, durata.

migrazione attiva-attiva

Un metodo di migrazione del database in cui i database di origine e di destinazione vengono mantenuti sincronizzati (utilizzando uno strumento di replica bidirezionale o operazioni di doppia scrittura) ed entrambi i database gestiscono le transazioni provenienti dalle applicazioni di connessione durante la migrazione. Questo metodo supporta la migrazione in piccoli batch controllati anziché richiedere una conversione una tantum. È più flessibile ma richiede più lavoro rispetto alla migrazione attiva-passiva.

migrazione attiva-passiva

Un metodo di migrazione del database in cui i database di origine e di destinazione vengono mantenuti sincronizzati, ma solo il database di origine gestisce le transazioni provenienti dalle applicazioni di connessione mentre i dati vengono replicati nel database di destinazione. Il database di destinazione non accetta alcuna transazione durante la migrazione.

funzione di aggregazione

Una funzione SQL che opera su un gruppo di righe e calcola un singolo valore restituito per il gruppo. Esempi di funzioni aggregate includono `SUM` e. `MAX`

Intelligenza artificiale

Vedi intelligenza artificiale.

AIOps

Guarda le operazioni di intelligenza artificiale.

anonimizzazione

Il processo di eliminazione permanente delle informazioni personali in un set di dati. L'anonimizzazione può aiutare a proteggere la privacy personale. I dati anonimi non sono più considerati dati personali.

anti-modello

Una soluzione utilizzata frequentemente per un problema ricorrente in cui la soluzione è controproducente, inefficace o meno efficace di un'alternativa.

controllo delle applicazioni

Un approccio alla sicurezza che consente l'uso solo di applicazioni approvate per proteggere un sistema dal malware.

portfolio di applicazioni

Una raccolta di informazioni dettagliate su ogni applicazione utilizzata da un'organizzazione, compresi i costi di creazione e manutenzione dell'applicazione e il relativo valore aziendale. Queste informazioni sono fondamentali per il processo di scoperta e analisi del portfolio e aiutano a identificare e ad assegnare la priorità alle applicazioni da migrare, modernizzare e ottimizzare.

intelligenza artificiale (IA)

Il campo dell'informatica dedicato all'uso delle tecnologie informatiche per svolgere funzioni cognitive tipicamente associate agli esseri umani, come l'apprendimento, la risoluzione di problemi e il riconoscimento di schemi. Per ulteriori informazioni, consulta la sezione Che cos'è l'intelligenza artificiale?

operazioni di intelligenza artificiale (AIOps)

Il processo di utilizzo delle tecniche di machine learning per risolvere problemi operativi, ridurre gli incidenti operativi e l'intervento umano e aumentare la qualità del servizio. Per ulteriori informazioni su come AIOps viene utilizzata nella strategia di AWS migrazione, consulta la guida all'integrazione delle operazioni.

crittografia asimmetrica

Un algoritmo di crittografia che utilizza una coppia di chiavi, una chiave pubblica per la crittografia e una chiave privata per la decrittografia. Puoi condividere la chiave pubblica perché non viene utilizzata per la decrittografia, ma l'accesso alla chiave privata deve essere altamente limitato.

atomicità, consistenza, isolamento, durabilità (ACID)

Un insieme di proprietà del software che garantiscono la validità dei dati e l'affidabilità operativa di un database, anche in caso di errori, interruzioni di corrente o altri problemi.

Controllo degli accessi basato su attributi (ABAC)

La pratica di creare autorizzazioni dettagliate basate su attributi utente, come reparto, ruolo professionale e nome del team. Per ulteriori informazioni, consulta ABAC AWS nella documentazione AWS Identity and Access Management (IAM).

fonte di dati autorevole

Una posizione in cui è archiviata la versione principale dei dati, considerata la fonte di informazioni più affidabile. È possibile copiare i dati dalla fonte di dati autorevole in altre posizioni allo scopo di elaborarli o modificarli, ad esempio anonimizzandoli, oscurandoli o pseudonimizzandoli.

Zona di disponibilità

Una posizione distinta all'interno di un edificio Regione AWS che è isolata dai guasti in altre zone di disponibilità e offre una connettività di rete economica e a bassa latenza verso altre zone di disponibilità nella stessa regione.

AWS Cloud Adoption Framework (CAF)AWS

Un framework di linee guida e best practice AWS per aiutare le organizzazioni a sviluppare un piano efficiente ed efficace per passare con successo al cloud. AWS CAF organizza le linee guida in sei aree di interesse chiamate prospettive: business, persone, governance, piattaforma, sicurezza e operazioni. Le prospettive relative ad azienda, persone e governance si concentrano sulle competenze e sui processi aziendali; le prospettive relative alla piattaforma, alla sicurezza e alle operazioni si concentrano sulle competenze e sui processi tecnici. Ad esempio, la prospettiva relativa alle persone si rivolge alle parti interessate che gestiscono le risorse umane (HR), le funzioni del personale e la gestione del personale. In questa prospettiva, AWS CAF fornisce linee guida per lo sviluppo delle persone, la formazione e le comunicazioni per aiutare a preparare l'organizzazione all'adozione del cloud di successo. Per ulteriori informazioni, consulta il sito web di AWS CAF e il white paper AWS CAF.

AWS Workload Qualification Framework (WQF)AWS

Uno strumento che valuta i carichi di lavoro di migrazione dei database, consiglia strategie di migrazione e fornisce stime del lavoro. AWS WQF è incluso in (). AWS Schema Conversion Tool AWS SCT Analizza gli schemi di database e gli oggetti di codice, il codice dell'applicazione, le dipendenze e le caratteristiche delle prestazioni e fornisce report di valutazione.

B

bot difettoso

Un bot che ha lo scopo di interrompere o causare danni a individui o organizzazioni.

BCP

Vedi la pianificazione della continuità operativa.

grafico comportamentale

Una vista unificata, interattiva dei comportamenti delle risorse e delle interazioni nel tempo. Puoi utilizzare un grafico comportamentale con Amazon Detective per esaminare tentativi di accesso non riusciti, chiamate API sospette e azioni simili. Per ulteriori informazioni, consulta Dati in un grafico comportamentale nella documentazione di Detective.

sistema big-endian

Un sistema che memorizza per primo il byte più importante. Vedi anche endianness.

Classificazione binaria

Un processo che prevede un risultato binario (una delle due classi possibili). Ad esempio, il modello di machine learning potrebbe dover prevedere problemi come "Questa e-mail è spam o non è spam?" o "Questo prodotto è un libro o un'auto?"

filtro Bloom

Una struttura di dati probabilistica ed efficiente in termini di memoria che viene utilizzata per verificare se un elemento fa parte di un set.

implementazione blu/verde

Una strategia di implementazione in cui si creano due ambienti separati ma identici. La versione corrente dell'applicazione viene eseguita in un ambiente (blu) e la nuova versione dell'applicazione nell'altro ambiente (verde). Questa strategia consente di ripristinare rapidamente il sistema con un impatto minimo.

bot

Un'applicazione software che esegue attività automatizzate su Internet e simula l'attività o l'interazione umana. Alcuni bot sono utili o utili, come i web crawler che indicizzano le informazioni su Internet. Alcuni altri bot, noti come bot dannosi, hanno lo scopo di disturbare o causare danni a individui o organizzazioni.

botnet

Reti di bot infettate da malware e controllate da un'unica parte, nota come bot herder o bot operator. Le botnet sono il meccanismo più noto per scalare i bot e il loro impatto.

ramo

Un'area contenuta di un repository di codice. Il primo ramo creato in un repository è il ramo principale. È possibile creare un nuovo ramo a partire da un ramo esistente e quindi sviluppare funzionalità o correggere bug al suo interno. Un ramo creato per sviluppare una funzionalità viene comunemente detto ramo di funzionalità. Quando la funzionalità è pronta per il rilascio, il ramo di funzionalità viene ricongiunto al ramo principale. Per ulteriori informazioni, consulta Informazioni sulle filiali (documentazione). GitHub

accesso break-glass

In circostanze eccezionali e tramite una procedura approvata, un mezzo rapido per consentire a un utente di accedere a un sito a Account AWS cui in genere non dispone delle autorizzazioni necessarie. Per ulteriori informazioni, vedere l'indicatore Implementate break-glass procedures nella guida Well-Architected AWS .

strategia brownfield

L'infrastruttura esistente nell'ambiente. Quando si adotta una strategia brownfield per un'architettura di sistema, si progetta l'architettura in base ai vincoli dei sistemi e dell'infrastruttura attuali. Per l'espansione dell'infrastruttura esistente, è possibile combinare strategie brownfield e greenfield.

cache del buffer

L'area di memoria in cui sono archiviati i dati a cui si accede con maggiore frequenza.

capacità di business

Azioni intraprese da un'azienda per generare valore (ad esempio vendite, assistenza clienti o marketing). Le architetture dei microservizi e le decisioni di sviluppo possono essere guidate dalle capacità aziendali. Per ulteriori informazioni, consulta la sezione Organizzazione in base alle funzionalità aziendali del whitepaper Esecuzione di microservizi containerizzati su AWS.

pianificazione della continuità operativa (BCP)

Un piano che affronta il potenziale impatto di un evento che comporta l'interruzione dell'attività, come una migrazione su larga scala, sulle operazioni e consente a un'azienda di riprendere rapidamente le operazioni.

C

CAF

Vedi Cloud Adoption AWS Framework.

implementazione canaria

Il rilascio lento e incrementale di una versione agli utenti finali. Quando sei sicuro, distribuisci la nuova versione e sostituisci la versione corrente nella sua interezza.

CCoE

Vedi Cloud Center of Excellence.

CDC

Vedi Change Data Capture.

Change Data Capture (CDC)

Il processo di tracciamento delle modifiche a un'origine dati, ad esempio una tabella di database, e di registrazione dei metadati relativi alla modifica. È possibile utilizzare CDC per vari scopi, ad esempio il controllo o la replica delle modifiche in un sistema di destinazione per mantenere la sincronizzazione.

ingegneria del caos

Introduzione intenzionale di guasti o eventi dirompenti per testare la resilienza di un sistema. Puoi usare AWS Fault Injection Service (AWS FIS) per eseguire esperimenti che stressano i tuoi AWS carichi di lavoro e valutarne la risposta.

CI/CD

Vedi integrazione continua e distribuzione continua.

classificazione

Un processo di categorizzazione che aiuta a generare previsioni. I modelli di ML per problemi di classificazione prevedono un valore discreto. I valori discreti sono sempre distinti l'uno dall'altro. Ad esempio, un modello potrebbe dover valutare se in un'immagine è presente o meno un'auto.

crittografia lato client

Crittografia dei dati a livello locale, prima che il destinatario li Servizio AWS riceva.

Centro di eccellenza cloud (CCoE)

Un team multidisciplinare che guida le iniziative di adozione del cloud in tutta l'organizzazione, tra cui lo sviluppo di best practice per il cloud, la mobilitazione delle risorse, la definizione delle tempistiche di migrazione e la guida dell'organizzazione attraverso trasformazioni su larga scala. Per ulteriori informazioni, consulta gli CCoE post sull' Cloud AWS Enterprise Strategy Blog.

cloud computing

La tecnologia cloud generalmente utilizzata per l'archiviazione remota di dati e la gestione dei dispositivi IoT. Il cloud computing è generalmente collegato alla tecnologia di edge computing.

modello operativo cloud

In un'organizzazione IT, il modello operativo utilizzato per creare, maturare e ottimizzare uno o più ambienti cloud. Per ulteriori informazioni, consulta Building your Cloud Operating Model.

fasi di adozione del cloud

Le quattro fasi che le organizzazioni in genere attraversano quando migrano verso Cloud AWS:

- Progetto: esecuzione di alcuni progetti relativi al cloud per scopi di dimostrazione e apprendimento
- Fondamento: effettuare investimenti fondamentali per scalare l'adozione del cloud (ad esempio, creazione di una landing zone, definizione di una CCo E, definizione di un modello operativo)
- Migrazione: migrazione di singole applicazioni
- Reinvenzione: ottimizzazione di prodotti e servizi e innovazione nel cloud

Queste fasi sono state definite da Stephen Orban nel post sul blog The Journey Toward Cloud-First & the Stages of Adoption on the Enterprise Strategy. Cloud AWS Per informazioni su come si relazionano alla strategia di AWS migrazione, consulta la guida alla preparazione alla migrazione.

CMDB

Vedi database di gestione della configurazione.

repository di codice

Una posizione in cui il codice di origine e altri asset, come documentazione, esempi e script, vengono archiviati e aggiornati attraverso processi di controllo delle versioni. Gli archivi cloud più comuni includono GitHub oBitbucket Cloud. Ogni versione del codice è denominata ramo. In una struttura a microservizi, ogni repository è dedicato a una singola funzionalità. Una singola pipeline CI/CD può utilizzare più repository.

cache fredda

Una cache del buffer vuota, non ben popolata o contenente dati obsoleti o irrilevanti. Ciò influisce sulle prestazioni perché l'istanza di database deve leggere dalla memoria o dal disco principale, il che richiede più tempo rispetto alla lettura dalla cache del buffer.

dati freddi

Dati a cui si accede raramente e che in genere sono storici. Quando si eseguono interrogazioni di questo tipo di dati, le interrogazioni lente sono in genere accettabili. Lo spostamento di questi dati su livelli o classi di storage meno costosi e con prestazioni inferiori può ridurre i costi.

visione artificiale (CV)

Un campo dell'intelligenza artificiale che utilizza l'apprendimento automatico per analizzare ed estrarre informazioni da formati visivi come immagini e video digitali. Ad esempio, Amazon SageMaker AI fornisce algoritmi di elaborazione delle immagini per CV.

deriva della configurazione

Per un carico di lavoro, una modifica della configurazione rispetto allo stato previsto. Potrebbe causare la non conformità del carico di lavoro e in genere è graduale e involontaria.

database di gestione della configurazione (CMDB)

Un repository che archivia e gestisce le informazioni su un database e il relativo ambiente IT, inclusi i componenti hardware e software e le relative configurazioni. In genere si utilizzano i dati di un CMDB nella fase di individuazione e analisi del portafoglio della migrazione.

Pacchetto di conformità

Una raccolta di AWS Config regole e azioni correttive che puoi assemblare per personalizzare i controlli di conformità e sicurezza. È possibile distribuire un pacchetto di conformità come singola entità in una regione Account AWS and o all'interno di un'organizzazione utilizzando un modello YAML. Per ulteriori informazioni, consulta i Conformance Pack nella documentazione. AWS Config

integrazione e distribuzione continua (continuous integration and continuous delivery, CI/CD)

Il processo di automazione delle fasi di origine, compilazione, test, gestione temporanea e produzione del processo di rilascio del software. CI/CD viene comunemente descritto come una pipeline. CI/CD può aiutarvi ad automatizzare i processi, migliorare la produttività, migliorare la qualità del codice e velocizzare le consegne. Per ulteriori informazioni, consulta Vantaggi

della distribuzione continua. CD può anche significare continuous deployment (implementazione continua). Per ulteriori informazioni, consulta Distribuzione continua e implementazione continua a confronto.

CV

Vedi visione artificiale.

D

dati a riposo

Dati stazionari nella rete, ad esempio i dati archiviati.

classificazione dei dati

Un processo per identificare e classificare i dati nella rete in base alla loro criticità e sensibilità. È un componente fondamentale di qualsiasi strategia di gestione dei rischi di sicurezza informatica perché consente di determinare i controlli di protezione e conservazione appropriati per i dati. La classificazione dei dati è un componente del pilastro della sicurezza nel AWS Well-Architected Framework. Per ulteriori informazioni, consulta Classificazione dei dati.

deriva dei dati

Una variazione significativa tra i dati di produzione e i dati utilizzati per addestrare un modello di machine learning o una modifica significativa dei dati di input nel tempo. La deriva dei dati può ridurre la qualità, l'accuratezza e l'equità complessive nelle previsioni dei modelli ML.

dati in transito

Dati che si spostano attivamente attraverso la rete, ad esempio tra le risorse di rete.

rete di dati

Un framework architettonico che fornisce la proprietà distribuita e decentralizzata dei dati con gestione e governance centralizzate.

riduzione al minimo dei dati

Il principio della raccolta e del trattamento dei soli dati strettamente necessari. Praticare la riduzione al minimo dei dati in the Cloud AWS può ridurre i rischi per la privacy, i costi e l'impronta di carbonio delle analisi.

perimetro dei dati

Una serie di barriere preventive nell' AWS ambiente che aiutano a garantire che solo le identità attendibili accedano alle risorse attendibili delle reti previste. Per ulteriori informazioni, consulta Building a data perimeter on. AWS

pre-elaborazione dei dati

Trasformare i dati grezzi in un formato che possa essere facilmente analizzato dal modello di ML. La pre-elaborazione dei dati può comportare la rimozione di determinate colonne o righe e l'eliminazione di valori mancanti, incoerenti o duplicati.

provenienza dei dati

Il processo di tracciamento dell'origine e della cronologia dei dati durante il loro ciclo di vita, ad esempio il modo in cui i dati sono stati generati, trasmessi e archiviati.

soggetto dei dati

Un individuo i cui dati vengono raccolti ed elaborati.

data warehouse

Un sistema di gestione dei dati che supporta la business intelligence, come l'analisi. I data warehouse contengono in genere grandi quantità di dati storici e vengono generalmente utilizzati per interrogazioni e analisi.

linguaggio di definizione del database (DDL)

Istruzioni o comandi per creare o modificare la struttura di tabelle e oggetti in un database.

linguaggio di manipolazione del database (DML)

Istruzioni o comandi per modificare (inserire, aggiornare ed eliminare) informazioni in un database.

DDL

Vedi linguaggio di definizione del database.

deep ensemble

Combinare più modelli di deep learning per la previsione. È possibile utilizzare i deep ensemble per ottenere una previsione più accurata o per stimare l'incertezza nelle previsioni.

deep learning

Un sottocampo del ML che utilizza più livelli di reti neurali artificiali per identificare la mappatura tra i dati di input e le variabili target di interesse.

defense-in-depth

Un approccio alla sicurezza delle informazioni in cui una serie di meccanismi e controlli di sicurezza sono accuratamente stratificati su una rete di computer per proteggere la riservatezza, l'integrità e la disponibilità della rete e dei dati al suo interno. Quando si adotta questa strategia AWS, si aggiungono più controlli a diversi livelli della AWS Organizations struttura per proteggere le risorse. Ad esempio, un defense-in-depth approccio potrebbe combinare l'autenticazione a più fattori, la segmentazione della rete e la crittografia.

amministratore delegato

In AWS Organizations, un servizio compatibile può registrare un account AWS membro per amministrare gli account dell'organizzazione e gestire le autorizzazioni per quel servizio. Questo account è denominato amministratore delegato per quel servizio specifico. Per ulteriori informazioni e un elenco di servizi compatibili, consulta Servizi che funzionano con AWS Organizations nella documentazione di AWS Organizations .

implementazione

Il processo di creazione di un'applicazione, di nuove funzionalità o di correzioni di codice disponibili nell'ambiente di destinazione. L'implementazione prevede l'applicazione di modifiche in una base di codice, seguita dalla creazione e dall'esecuzione di tale base di codice negli ambienti applicativi.

Ambiente di sviluppo

Vedi ambiente.

controllo di rilevamento

Un controllo di sicurezza progettato per rilevare, registrare e avvisare dopo che si è verificato un evento. Questi controlli rappresentano una seconda linea di difesa e avvisano l'utente in caso di eventi di sicurezza che aggirano i controlli preventivi in vigore. Per ulteriori informazioni, consulta Controlli di rilevamento in Implementazione dei controlli di sicurezza in AWS.

mappatura del flusso di valore dello sviluppo (DVSM)

Un processo utilizzato per identificare e dare priorità ai vincoli che influiscono negativamente sulla velocità e sulla qualità nel ciclo di vita dello sviluppo del software. DVSM estende il processo di

mappatura del flusso di valore originariamente progettato per pratiche di produzione snella. Si concentra sulle fasi e sui team necessari per creare e trasferire valore attraverso il processo di sviluppo del software.

gemello digitale

Una rappresentazione virtuale di un sistema reale, ad esempio un edificio, una fabbrica, un'attrezzatura industriale o una linea di produzione. I gemelli digitali supportano la manutenzione predittiva, il monitoraggio remoto e l'ottimizzazione della produzione.

tabella delle dimensioni

In uno schema a stella, una tabella più piccola che contiene gli attributi dei dati quantitativi in una tabella dei fatti. Gli attributi della tabella delle dimensioni sono in genere campi di testo o numeri discreti che si comportano come testo. Questi attributi vengono comunemente utilizzati per il vincolo delle query, il filtraggio e l'etichettatura dei set di risultati.

disastro

Un evento che impedisce a un carico di lavoro o a un sistema di raggiungere gli obiettivi aziendali nella sua sede principale di implementazione. Questi eventi possono essere disastri naturali, guasti tecnici o il risultato di azioni umane, come errori di configurazione involontari o attacchi di malware.

disaster recovery (DR)

La strategia e il processo utilizzati per ridurre al minimo i tempi di inattività e la perdita di dati causati da un disastro. Per ulteriori informazioni, consulta Disaster Recovery of Workloads su AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Vedi linguaggio di manipolazione del database.

progettazione basata sul dominio

Un approccio allo sviluppo di un sistema software complesso collegandone i componenti a domini in evoluzione, o obiettivi aziendali principali, perseguiti da ciascun componente. Questo concetto è stato introdotto da Eric Evans nel suo libro,Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003). Per informazioni su come utilizzare la progettazione basata sul dominio con il modello del fico strangolatore (Strangler Fig), consulta la sezione Modernizzazione incrementale dei servizi Web Microsoft ASP.NET (ASMX) legacy utilizzando container e il Gateway Amazon API.

DOTT.

Vedi disaster recovery.

rilevamento della deriva

Tracciamento delle deviazioni da una configurazione di base. Ad esempio, è possibile AWS CloudFormation utilizzarlo per rilevare deviazioni nelle risorse di sistema oppure AWS Control Tower per rilevare cambiamenti nella landing zone che potrebbero influire sulla conformità ai requisiti di governance.

DVSM

Vedi la mappatura del flusso di valore dello sviluppo.

E

EDA

Vedi analisi esplorativa dei dati.

MODIFICA

Vedi scambio elettronico di dati.

edge computing

La tecnologia che aumenta la potenza di calcolo per i dispositivi intelligenti all'edge di una rete IoT. Rispetto al cloud computing, l'edge computing può ridurre la latenza di comunicazione e migliorare i tempi di risposta.

scambio elettronico di dati (EDI)

Lo scambio automatizzato di documenti aziendali tra organizzazioni. Per ulteriori informazioni, vedere Cos'è lo scambio elettronico di dati.

crittografia

Un processo di elaborazione che trasforma i dati in chiaro, leggibili dall'uomo, in testo cifrato.

chiave crittografica

Una stringa crittografica di bit randomizzati generata da un algoritmo di crittografia. Le chiavi possono variare di lunghezza e ogni chiave è progettata per essere imprevedibile e univoca.

endianità

L'ordine in cui i byte vengono archiviati nella memoria del computer. I sistemi big-endian memorizzano per primo il byte più importante. I sistemi little-endian memorizzano per primo il byte meno importante.

endpoint

Vedi service endpoint.

servizio endpoint

Un servizio che puoi ospitare in un cloud privato virtuale (VPC) da condividere con altri utenti. Puoi creare un servizio endpoint con AWS PrivateLink e concedere autorizzazioni ad altri Account AWS o a AWS Identity and Access Management (IAM) principali. Questi account o principali possono connettersi al servizio endpoint in privato creando endpoint VPC di interfaccia. Per ulteriori informazioni, consulta Creazione di un servizio endpoint nella documentazione di Amazon Virtual Private Cloud (Amazon VPC).

pianificazione delle risorse aziendali (ERP)

Un sistema che automatizza e gestisce i processi aziendali chiave (come contabilità, MES e gestione dei progetti) per un'azienda.

crittografia envelope

Il processo di crittografia di una chiave di crittografia con un'altra chiave di crittografia. Per ulteriori informazioni, vedete Envelope encryption nella documentazione AWS Key Management Service (AWS KMS).

ambiente

Un'istanza di un'applicazione in esecuzione. Di seguito sono riportati i tipi di ambiente più comuni nel cloud computing:

- ambiente di sviluppo: un'istanza di un'applicazione in esecuzione disponibile solo per il team principale responsabile della manutenzione dell'applicazione. Gli ambienti di sviluppo vengono utilizzati per testare le modifiche prima di promuoverle negli ambienti superiori. Questo tipo di ambiente viene talvolta definito ambiente di test.
- ambienti inferiori: tutti gli ambienti di sviluppo di un'applicazione, ad esempio quelli utilizzati per le build e i test iniziali.

- ambiente di produzione: un'istanza di un'applicazione in esecuzione a cui gli utenti finali possono accedere. In una CI/CD pipeline, l'ambiente di produzione è l'ultimo ambiente di distribuzione.
- ambienti superiori: tutti gli ambienti a cui possono accedere utenti diversi dal team di sviluppo principale. Si può trattare di un ambiente di produzione, ambienti di preproduzione e ambienti per i test di accettazione da parte degli utenti.

epica

Nelle metodologie agili, categorie funzionali che aiutano a organizzare e dare priorità al lavoro. Le epiche forniscono una descrizione di alto livello dei requisiti e delle attività di implementazione. Ad esempio, le epopee della sicurezza AWS CAF includono la gestione delle identità e degli accessi, i controlli investigativi, la sicurezza dell'infrastruttura, la protezione dei dati e la risposta agli incidenti. Per ulteriori informazioni sulle epiche, consulta la strategia di migrazione AWS , consulta la guida all'implementazione del programma.

ERP

Vedi pianificazione delle risorse aziendali.

analisi esplorativa dei dati (EDA)

Il processo di analisi di un set di dati per comprenderne le caratteristiche principali. Si raccolgono o si aggregano dati e quindi si eseguono indagini iniziali per trovare modelli, rilevare anomalie e verificare ipotesi. L'EDA viene eseguita calcolando statistiche di riepilogo e creando visualizzazioni di dati.

F

tabella dei fatti

Il tavolo centrale in uno schema a stella. Memorizza dati quantitativi sulle operazioni aziendali. In genere, una tabella dei fatti contiene due tipi di colonne: quelle che contengono misure e quelle che contengono una chiave esterna per una tabella di dimensioni.

fallire velocemente

Una filosofia che utilizza test frequenti e incrementali per ridurre il ciclo di vita dello sviluppo. È una parte fondamentale di un approccio agile.

limite di isolamento dei guasti

Nel Cloud AWS, un limite come una zona di disponibilità Regione AWS, un piano di controllo o un piano dati che limita l'effetto di un errore e aiuta a migliorare la resilienza dei carichi di lavoro. Per ulteriori informazioni, consulta AWS Fault Isolation Boundaries.

ramo di funzionalità

Vedi filiale.

caratteristiche

I dati di input che usi per fare una previsione. Ad esempio, in un contesto di produzione, le caratteristiche potrebbero essere immagini acquisite periodicamente dalla linea di produzione.

importanza delle caratteristiche

Quanto è importante una caratteristica per le previsioni di un modello. Di solito viene espresso come punteggio numerico che può essere calcolato con varie tecniche, come Shapley Additive Explanations (SHAP) e gradienti integrati. Per ulteriori informazioni, consulta Interpretabilità del modello di machine learning con AWS.

trasformazione delle funzionalità

Per ottimizzare i dati per il processo di machine learning, incluso l'arricchimento dei dati con fonti aggiuntive, il dimensionamento dei valori o l'estrazione di più set di informazioni da un singolo campo di dati. Ciò consente al modello di ML di trarre vantaggio dai dati. Ad esempio, se suddividi la data "2021-05-27 00:15:37" in "2021", "maggio", "giovedì" e "15", puoi aiutare l'algoritmo di apprendimento ad apprendere modelli sfumati associati a diversi componenti dei dati.

prompt con pochi scatti

Fornire a un LLM un numero limitato di esempi che dimostrino l'attività e il risultato desiderato prima di chiedergli di eseguire un'attività simile. Questa tecnica è un'applicazione dell'apprendimento contestuale, in cui i modelli imparano da esempi (immagini) incorporati nei prompt. I prompt con pochi passaggi possono essere efficaci per attività che richiedono una formattazione, un ragionamento o una conoscenza del dominio specifici. Vedi anche zero-shot prompting.

FGAC

Vedi il controllo granulare degli accessi.

controllo granulare degli accessi (FGAC)

L'uso di più condizioni per consentire o rifiutare una richiesta di accesso.

migrazione flash-cut

Un metodo di migrazione del database che utilizza la replica continua dei dati tramite l'acquisizione dei dati delle modifiche per migrare i dati nel più breve tempo possibile, anziché utilizzare un approccio graduale. L'obiettivo è ridurre al minimo i tempi di inattività.

FM

Vedi modello di base.

modello di fondazione (FM)

Una grande rete neurale di deep learning che si è addestrata su enormi set di dati generalizzati e non etichettati. FMs sono in grado di svolgere un'ampia varietà di attività generali, come comprendere il linguaggio, generare testo e immagini e conversare in linguaggio naturale. Per ulteriori informazioni, consulta Cosa sono i modelli Foundation.

G

IA generativa

Un sottoinsieme di modelli di intelligenza artificiale che sono stati addestrati su grandi quantità di dati e che possono utilizzare un semplice messaggio di testo per creare nuovi contenuti e artefatti, come immagini, video, testo e audio. Per ulteriori informazioni, consulta Cos'è l'IA generativa.

blocco geografico

Vedi restrizioni geografiche.

limitazioni geografiche (blocco geografico)

In Amazon CloudFront, un'opzione per impedire agli utenti di determinati paesi di accedere alle distribuzioni di contenuti. Puoi utilizzare un elenco consentito o un elenco di blocco per specificare i paesi approvati e vietati. Per ulteriori informazioni, consulta Limitare la distribuzione geografica dei contenuti nella CloudFront documentazione.

Flusso di lavoro di GitFlow

Un approccio in cui gli ambienti inferiori e superiori utilizzano rami diversi in un repository di codice di origine. Il flusso di lavoro Gitflow è considerato obsoleto e il flusso di lavoro basato su trunk è l'approccio moderno e preferito.

immagine dorata

Un'istantanea di un sistema o di un software utilizzata come modello per distribuire nuove istanze di quel sistema o software. Ad esempio, nella produzione, un'immagine dorata può essere utilizzata per fornire software su più dispositivi e contribuire a migliorare la velocità, la scalabilità e la produttività nelle operazioni di produzione dei dispositivi.

strategia greenfield

L'assenza di infrastrutture esistenti in un nuovo ambiente. Quando si adotta una strategia greenfield per un'architettura di sistema, è possibile selezionare tutte le nuove tecnologie senza il vincolo della compatibilità con l'infrastruttura esistente, nota anche come brownfield. Per l'espansione dell'infrastruttura esistente, è possibile combinare strategie brownfield e greenfield.

guardrail

Una regola di alto livello che aiuta a governare le risorse, le politiche e la conformità tra le unità organizzative (). OUs I guardrail preventivi applicano le policy per garantire l'allineamento agli standard di conformità. Vengono implementati utilizzando le policy di controllo dei servizi e i limiti delle autorizzazioni IAM. I guardrail di rilevamento rilevano le violazioni delle policy e i problemi di conformità e generano avvisi per porvi rimedio. Sono implementati utilizzando Amazon AWS Config AWS Security Hub CSPM GuardDuty AWS Trusted Advisor, Amazon Inspector e controlli personalizzati AWS Lambda .

H

AH

Vedi disponibilità elevata.

migrazione di database eterogenea

Migrazione del database di origine in un database di destinazione che utilizza un motore di database diverso (ad esempio, da Oracle ad Amazon Aurora). La migrazione eterogenea fa in

genere parte di uno sforzo di riprogettazione e la conversione dello schema può essere un'attività complessa. AWS offre AWS SCT che aiuta con le conversioni dello schema.

alta disponibilità (HA)

La capacità di un carico di lavoro di funzionare in modo continuo, senza intervento, in caso di sfide o disastri. I sistemi HA sono progettati per il failover automatico, fornire costantemente prestazioni di alta qualità e gestire carichi e guasti diversi con un impatto minimo sulle prestazioni.

modernizzazione storica

Un approccio utilizzato per modernizzare e aggiornare i sistemi di tecnologia operativa (OT) per soddisfare meglio le esigenze dell'industria manifatturiera. Uno storico è un tipo di database utilizzato per raccogliere e archiviare dati da varie fonti in una fabbrica.

dati di blocco

Una parte di dati storici etichettati che viene trattenuta da un set di dati utilizzata per addestrare un modello di apprendimento automatico. È possibile utilizzare i dati di holdout per valutare le prestazioni del modello confrontando le previsioni del modello con i dati di holdout.

migrazione di database omogenea

Migrazione del database di origine in un database di destinazione che condivide lo stesso motore di database (ad esempio, da Microsoft SQL Server ad Amazon RDS per SQL Server). La migrazione omogenea fa in genere parte di un'operazione di rehosting o ridefinizione della piattaforma. Per migrare lo schema è possibile utilizzare le utilità native del database.

dati caldi

Dati a cui si accede frequentemente, come dati in tempo reale o dati di traduzione recenti. Questi dati richiedono in genere un livello o una classe di storage ad alte prestazioni per fornire risposte rapide alle query.

hotfix

Una soluzione urgente per un problema critico in un ambiente di produzione. A causa della sua urgenza, un hotfix viene in genere creato al di fuori del tipico DevOps flusso di lavoro di rilascio.

periodo di hypercare

Subito dopo la conversione, il periodo di tempo in cui un team di migrazione gestisce e monitora le applicazioni migrate nel cloud per risolvere eventuali problemi. In genere, questo periodo dura

da 1 a 4 giorni. Al termine del periodo di hypercare, il team addetto alla migrazione in genere trasferisce la responsabilità delle applicazioni al team addetto alle operazioni cloud.

I

IaC

Vedi l'infrastruttura come codice.

Policy basata su identità

Una policy associata a uno o più principi IAM che definisce le relative autorizzazioni all'interno dell' Cloud AWS ambiente.

applicazione inattiva

Un'applicazione che prevede un uso di CPU e memoria medio compreso tra il 5% e il 20% in un periodo di 90 giorni. In un progetto di migrazione, è normale ritirare queste applicazioni o mantenerle on-premise.

IIoT

Vedi Industrial Internet of Things.

infrastruttura immutabile

Un modello che implementa una nuova infrastruttura per i carichi di lavoro di produzione anziché aggiornare, applicare patch o modificare l'infrastruttura esistente. Le infrastrutture immutabili sono intrinsecamente più coerenti, affidabili e prevedibili delle infrastrutture mutabili. Per ulteriori informazioni, consulta la best practice Deploy using immutable infrastructure in Well-Architected AWS Framework.

VPC in ingresso (ingress)

In un'architettura AWS multi-account, un VPC che accetta, ispeziona e indirizza le connessioni di rete dall'esterno di un'applicazione. La AWS Security Reference Architecture consiglia di configurare l'account di rete con funzionalità in entrata, in uscita e di ispezione VPCs per proteggere l'interfaccia bidirezionale tra l'applicazione e Internet in generale.

migrazione incrementale

Una strategia di conversione in cui si esegue la migrazione dell'applicazione in piccole parti anziché eseguire una conversione singola e completa. Ad esempio, inizialmente potresti spostare

solo alcuni microservizi o utenti nel nuovo sistema. Dopo aver verificato che tutto funzioni correttamente, puoi spostare in modo incrementale microservizi o utenti aggiuntivi fino alla disattivazione del sistema legacy. Questa strategia riduce i rischi associati alle migrazioni di grandi dimensioni.

Industria 4.0

Un termine introdotto da Klaus Schwab nel 2016 per riferirsi alla modernizzazione dei processi di produzione attraverso progressi in termini di connettività, dati in tempo reale, automazione, analisi e AI/ML.

infrastruttura

Tutte le risorse e gli asset contenuti nell'ambiente di un'applicazione.

infrastruttura come codice (IaC)

Il processo di provisioning e gestione dell'infrastruttura di un'applicazione tramite un insieme di file di configurazione. Il processo IaC è progettato per aiutarti a centralizzare la gestione dell'infrastruttura, a standardizzare le risorse e a dimensionare rapidamente, in modo che i nuovi ambienti siano ripetibili, affidabili e coerenti.

IIoInternet delle cose industriale (T)

L'uso di sensori e dispositivi connessi a Internet nei settori industriali, come quello manifatturiero, energetico, automobilistico, sanitario, delle scienze della vita e dell'agricoltura. Per ulteriori informazioni, vedere Creazione di una strategia di trasformazione digitale per l'Internet of Things (IIoT) industriale.

VPC di ispezione

In un'architettura AWS multi-account, un VPC centralizzato che gestisce le ispezioni del traffico di rete tra VPCs (nello stesso o in modo diverso Regioni AWS), Internet e le reti locali. La AWS Security Reference Architecture consiglia di configurare l'account di rete con informazioni in entrata, in uscita e di ispezione VPCs per proteggere l'interfaccia bidirezionale tra l'applicazione e Internet in generale.

Internet of Things (IoT)

La rete di oggetti fisici connessi con sensori o processori incorporati che comunicano con altri dispositivi e sistemi tramite Internet o una rete di comunicazione locale. Per ulteriori informazioni, consulta Cos'è l'IoT?

interpretabilità

Una caratteristica di un modello di machine learning che descrive il grado in cui un essere umano è in grado di comprendere in che modo le previsioni del modello dipendono dai suoi input. Per ulteriori informazioni, vedere Interpretabilità del modello di machine learning con. AWS

IoT

Vedi Internet of Things.

libreria di informazioni IT (ITIL)

Una serie di best practice per offrire servizi IT e allinearli ai requisiti aziendali. ITIL fornisce le basi per ITSM.

gestione dei servizi IT (ITSM)

Attività associate alla progettazione, implementazione, gestione e supporto dei servizi IT per un'organizzazione. Per informazioni sull'integrazione delle operazioni cloud con gli strumenti ITSM, consulta la guida all'integrazione delle operazioni.

ITIL

Vedi la libreria di informazioni IT.

ITSM

Vedi Gestione dei servizi IT.

L

controllo degli accessi basato su etichette (LBAC)

Un'implementazione del controllo di accesso obbligatorio (MAC) in cui agli utenti e ai dati stessi viene assegnato esplicitamente un valore di etichetta di sicurezza. L'intersezione tra l'etichetta di sicurezza utente e l'etichetta di sicurezza dei dati determina quali righe e colonne possono essere visualizzate dall'utente.

zona di destinazione

Una landing zone è un AWS ambiente multi-account ben progettato, scalabile e sicuro. Questo è un punto di partenza dal quale le organizzazioni possono avviare e distribuire rapidamente carichi di lavoro e applicazioni con fiducia nel loro ambiente di sicurezza e infrastruttura. Per ulteriori

informazioni sulle zone di destinazione, consulta la sezione Configurazione di un ambiente AWS multi-account sicuro e scalabile.

modello linguistico di grandi dimensioni (LLM)

Un modello di intelligenza artificiale di deep learning preaddestrato su una grande quantità di dati. Un LLM può svolgere più attività, come rispondere a domande, riepilogare documenti, tradurre testo in altre lingue e completare frasi. Per ulteriori informazioni, consulta Cosa sono. LLMs

migrazione su larga scala

Una migrazione di 300 o più server.

BIANCO

Vedi controllo degli accessi basato su etichette.

Privilegio minimo

La best practice di sicurezza per la concessione delle autorizzazioni minime richieste per eseguire un'attività. Per ulteriori informazioni, consulta Applicazione delle autorizzazioni del privilegio minimo nella documentazione di IAM.

eseguire il rehosting (lift and shift)

Vedi 7 R.

sistema little-endian

Un sistema che memorizza per primo il byte meno importante. Vedi anche endianità.

LLM

Vedi modello linguistico di grandi dimensioni.

ambienti inferiori

Vedi ambiente.

M

machine learning (ML)

Un tipo di intelligenza artificiale che utilizza algoritmi e tecniche per il riconoscimento e l'apprendimento di schemi. Il machine learning analizza e apprende dai dati registrati, come i dati

dell'Internet delle cose (IoT), per generare un modello statistico basato su modelli. Per ulteriori informazioni, consulta la sezione Machine learning.

ramo principale

Vedi filiale.

malware

Software progettato per compromettere la sicurezza o la privacy del computer. Il malware potrebbe interrompere i sistemi informatici, divulgare informazioni sensibili o ottenere accessi non autorizzati. Esempi di malware includono virus, worm, ransomware, trojan horse, spyware e keylogger.

servizi gestiti

Servizi AWS per cui AWS gestisce il livello di infrastruttura, il sistema operativo e le piattaforme e si accede agli endpoint per archiviare e recuperare i dati. Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) e Amazon DynamoDB sono esempi di servizi gestiti. Questi sono noti anche come servizi astratti.

sistema di esecuzione della produzione (MES)

Un sistema software per tracciare, monitorare, documentare e controllare i processi di produzione che convertono le materie prime in prodotti finiti in officina.

MAP

Vedi Migration Acceleration Program.

meccanismo

Un processo completo in cui si crea uno strumento, si promuove l'adozione dello strumento e quindi si esaminano i risultati per apportare le modifiche. Un meccanismo è un ciclo che si rafforza e si migliora man mano che funziona. Per ulteriori informazioni, consulta Creazione di meccanismi nel AWS Well-Architected Framework.

account membro

Tutti gli account Account AWS diversi dall'account di gestione che fanno parte di un'organizzazione in. AWS Organizations Un account può essere membro di una sola organizzazione alla volta.

MEH

Vedi sistema di esecuzione della produzione.

Message Queuing Telemetry Transport (MQTT)

Un protocollo di comunicazione machine-to-machine (M2M) leggero, basato sul modello di pubblicazione/sottoscrizione, per dispositivi IoT con risorse limitate.

microservizio

Un servizio piccolo e indipendente che comunica tramite canali ben definiti ed è in genere di proprietà di piccoli team autonomi. APIs Ad esempio, un sistema assicurativo potrebbe includere microservizi che si riferiscono a funzionalità aziendali, come vendite o marketing, o sottodomini, come acquisti, reclami o analisi. I vantaggi dei microservizi includono agilità, dimensionamento flessibile, facilità di implementazione, codice riutilizzabile e resilienza. Per ulteriori informazioni, consulta Integrazione dei microservizi utilizzando servizi serverless. AWS

architettura di microservizi

Un approccio alla creazione di un'applicazione con componenti indipendenti che eseguono ogni processo applicativo come microservizio. Questi microservizi comunicano attraverso un'interfaccia ben definita utilizzando sistemi leggeri. APIs Ogni microservizio in questa architettura può essere aggiornato, distribuito e dimensionato per soddisfare la richiesta di funzioni specifiche di un'applicazione. Per ulteriori informazioni, vedere Implementazione dei microservizi su. AWS

Programma di accelerazione della migrazione (MAP)

Un AWS programma che fornisce consulenza, supporto, formazione e servizi per aiutare le organizzazioni a costruire una solida base operativa per il passaggio al cloud e per contribuire a compensare il costo iniziale delle migrazioni. MAP include una metodologia di migrazione per eseguire le migrazioni precedenti in modo metodico e un set di strumenti per automatizzare e accelerare gli scenari di migrazione comuni.

migrazione su larga scala

Il processo di trasferimento della maggior parte del portfolio di applicazioni sul cloud avviene a ondate, con più applicazioni trasferite a una velocità maggiore in ogni ondata. Questa fase utilizza le migliori pratiche e le lezioni apprese nelle fasi precedenti per implementare una fabbrica di migrazione di team, strumenti e processi per semplificare la migrazione dei carichi di lavoro attraverso l'automazione e la distribuzione agile. Questa è la terza fase della strategia di migrazione AWS.

fabbrica di migrazione

Team interfunzionali che semplificano la migrazione dei carichi di lavoro attraverso approcci automatizzati e agili. I team di Migration Factory in genere includono addetti alle operazioni,

analisti e proprietari aziendali, ingegneri addetti alla migrazione, sviluppatori e DevOps professionisti che lavorano nell'ambito degli sprint. Tra il 20% e il 50% di un portfolio di applicazioni aziendali è costituito da schemi ripetuti che possono essere ottimizzati con un approccio di fabbrica. Per ulteriori informazioni, consulta la discussione sulle fabbriche di migrazione e la Guida alla fabbrica di migrazione al cloud in questo set di contenuti.

metadati di migrazione

Le informazioni sull'applicazione e sul server necessarie per completare la migrazione. Ogni modello di migrazione richiede un set diverso di metadati di migrazione. Esempi di metadati di migrazione includono la sottorete, il gruppo di sicurezza e l'account di destinazione. AWS

modello di migrazione

Un'attività di migrazione ripetibile che descrive in dettaglio la strategia di migrazione, la destinazione della migrazione e l'applicazione o il servizio di migrazione utilizzati. Esempio: riorganizza la migrazione su Amazon EC2 AWS con Application Migration Service.

Valutazione del portfolio di migrazione (MPA)

Uno strumento online che fornisce informazioni per la convalida del business case per la migrazione a. Cloud AWS MPA offre una valutazione dettagliata del portfolio (dimensionamento corretto dei server, prezzi, confronto del TCO, analisi dei costi di migrazione) e pianificazione della migrazione (analisi e raccolta dei dati delle applicazioni, raggruppamento delle applicazioni, prioritizzazione delle migrazioni e pianificazione delle ondate). Lo strumento MPA (richiede l'accesso) è disponibile gratuitamente per tutti i AWS consulenti e i consulenti dei partner APN.

valutazione della preparazione alla migrazione (MRA)

Il processo di acquisizione di informazioni sullo stato di preparazione al cloud di un'organizzazione, l'identificazione dei punti di forza e di debolezza e la creazione di un piano d'azione per colmare le lacune identificate, utilizzando il CAF. AWS Per ulteriori informazioni, consulta la guida di preparazione alla migrazione. MRA è la prima fase della strategia di migrazione AWS.

strategia di migrazione

L'approccio utilizzato per migrare un carico di lavoro verso. Cloud AWS Per ulteriori informazioni, consulta la voce 7 R in questo glossario e consulta Mobilita la tua organizzazione per accelerare le migrazioni su larga scala.

ML

Vedi machine learning.

modernizzazione

Trasformazione di un'applicazione obsoleta (legacy o monolitica) e della relativa infrastruttura in un sistema agile, elastico e altamente disponibile nel cloud per ridurre i costi, aumentare l'efficienza e sfruttare le innovazioni. Per ulteriori informazioni, vedere Strategia per la modernizzazione delle applicazioni in. Cloud AWS

valutazione della preparazione alla modernizzazione

Una valutazione che aiuta a determinare la preparazione alla modernizzazione delle applicazioni di un'organizzazione, identifica vantaggi, rischi e dipendenze e determina in che misura l'organizzazione può supportare lo stato futuro di tali applicazioni. Il risultato della valutazione è uno schema dell'architettura di destinazione, una tabella di marcia che descrive in dettaglio le fasi di sviluppo e le tappe fondamentali del processo di modernizzazione e un piano d'azione per colmare le lacune identificate. Per ulteriori informazioni, vedere Valutazione della preparazione alla modernizzazione per le applicazioni in. Cloud AWS

applicazioni monolitiche (monoliti)

Applicazioni eseguite come un unico servizio con processi strettamente collegati. Le applicazioni monolitiche presentano diversi inconvenienti. Se una funzionalità dell'applicazione registra un picco di domanda, l'intera architettura deve essere dimensionata. L'aggiunta o il miglioramento delle funzionalità di un'applicazione monolitica diventa inoltre più complessa man mano che la base di codice cresce. Per risolvere questi problemi, puoi utilizzare un'architettura di microservizi. Per ulteriori informazioni, consulta la sezione Scomposizione dei monoliti in microservizi.

MAPPA

Vedi Migration Portfolio Assessment.

MQTT

Vedi Message Queuing Telemetry Transport.

classificazione multiclasse

Un processo che aiuta a generare previsioni per più classi (prevedendo uno o più di due risultati). Ad esempio, un modello di machine learning potrebbe chiedere "Questo prodotto è un libro, un'auto o un telefono?" oppure "Quale categoria di prodotti è più interessante per questo cliente?"

infrastruttura mutabile

Un modello che aggiorna e modifica l'infrastruttura esistente per i carichi di lavoro di produzione. Per migliorare la coerenza, l'affidabilità e la prevedibilità, il AWS Well-Architected Framework consiglia l'uso di un'infrastruttura immutabile come best practice.

O

OAC

Vedi Origin Access Control.

QUERCIA

Vedi Origin Access Identity.

OCM

Vedi gestione delle modifiche organizzative.

migrazione offline

Un metodo di migrazione in cui il carico di lavoro di origine viene eliminato durante il processo di migrazione. Questo metodo prevede tempi di inattività prolungati e viene in genere utilizzato per carichi di lavoro piccoli e non critici.

OI

Vedi l'integrazione delle operazioni.

OLA

Vedi accordo a livello operativo.

migrazione online

Un metodo di migrazione in cui il carico di lavoro di origine viene copiato sul sistema di destinazione senza essere messo offline. Le applicazioni connesse al carico di lavoro possono continuare a funzionare durante la migrazione. Questo metodo comporta tempi di inattività pari a zero o comunque minimi e viene in genere utilizzato per carichi di lavoro di produzione critici.

OPC-UA

Vedi Open Process Communications - Unified Architecture.

Comunicazioni a processo aperto - Architettura unificata (OPC-UA)

Un protocollo di comunicazione machine-to-machine (M2M) per l'automazione industriale. OPC-UA fornisce uno standard di interoperabilità con schemi di crittografia, autenticazione e autorizzazione dei dati.

accordo a livello operativo (OLA)

Un accordo che chiarisce quali sono gli impegni reciproci tra i gruppi IT funzionali, a supporto di un accordo sul livello di servizio (SLA).

revisione della prontezza operativa (ORR)

Un elenco di domande e best practice associate che aiutano a comprendere, valutare, prevenire o ridurre la portata degli incidenti e dei possibili guasti. Per ulteriori informazioni, vedere Operational Readiness Reviews (ORR) nel Well-Architected AWS Framework.

tecnologia operativa (OT)

Sistemi hardware e software che interagiscono con l'ambiente fisico per controllare le operazioni, le apparecchiature e le infrastrutture industriali. Nella produzione, l'integrazione di sistemi OT e di tecnologia dell'informazione (IT) è un obiettivo chiave per le trasformazioni dell'Industria 4.0.

integrazione delle operazioni (OI)

Il processo di modernizzazione delle operazioni nel cloud, che prevede la pianificazione, l'automazione e l'integrazione della disponibilità. Per ulteriori informazioni, consulta la guida all'integrazione delle operazioni.

trail organizzativo

Un percorso creato da noi AWS CloudTrail che registra tutti gli eventi di un'organizzazione per tutti Account AWS . AWS Organizations Questo percorso viene creato in ogni Account AWS che fa parte dell'organizzazione e tiene traccia dell'attività in ogni account. Per ulteriori informazioni, consulta Creazione di un percorso per un'organizzazione nella CloudTrail documentazione.

gestione del cambiamento organizzativo (OCM)

Un framework per la gestione di trasformazioni aziendali importanti e che comportano l'interruzione delle attività dal punto di vista delle persone, della cultura e della leadership. OCM aiuta le organizzazioni a prepararsi e passare a nuovi sistemi e strategie accelerando l'adozione del cambiamento, affrontando i problemi di transizione e promuovendo cambiamenti culturali e organizzativi. Nella strategia di AWS migrazione, questo framework si chiama accelerazione delle

persone, a causa della velocità di cambiamento richiesta nei progetti di adozione del cloud. Per ulteriori informazioni, consultare la Guida OCM.

controllo dell'accesso all'origine (OAC)

In CloudFront, un'opzione avanzata per limitare l'accesso per proteggere i contenuti di Amazon Simple Storage Service (Amazon S3). OAC supporta tutti i bucket S3 in generale Regioni AWS, la crittografia lato server con AWS KMS (SSE-KMS) e le richieste dinamiche e dirette al bucket S3. `PUT` `DELETE`

identità di accesso origine (OAI)

Nel CloudFront, un'opzione per limitare l'accesso per proteggere i tuoi contenuti Amazon S3. Quando usi OAI, CloudFront crea un principale con cui Amazon S3 può autenticarsi. I principali autenticati possono accedere ai contenuti in un bucket S3 solo tramite una distribuzione specifica. CloudFront Vedi anche OAC, che fornisce un controllo degli accessi più granulare e avanzato.

ORR

Vedi la revisione della prontezza operativa.

NON

Vedi la tecnologia operativa.

VPC in uscita (egress)

In un'architettura AWS multi-account, un VPC che gestisce le connessioni di rete avviate dall'interno di un'applicazione. La AWS Security Reference Architecture consiglia di configurare l'account di rete con funzionalità in entrata, in uscita e di ispezione VPCs per proteggere l'interfaccia bidirezionale tra l'applicazione e Internet in generale.

P

limite delle autorizzazioni

Una policy di gestione IAM collegata ai principali IAM per impostare le autorizzazioni massime che l'utente o il ruolo possono avere. Per ulteriori informazioni, consulta Limiti delle autorizzazioni nella documentazione di IAM.

informazioni di identificazione personale (PII)

Informazioni che, se visualizzate direttamente o abbinate ad altri dati correlati, possono essere utilizzate per dedurre ragionevolmente l'identità di un individuo. Esempi di informazioni personali includono nomi, indirizzi e informazioni di contatto.

Informazioni che consentono l'identificazione personale degli utenti

Visualizza le informazioni di identificazione personale.

playbook

Una serie di passaggi predefiniti che raccolgono il lavoro associato alle migrazioni, come l'erogazione delle funzioni operative principali nel cloud. Un playbook può assumere la forma di script, runbook automatici o un riepilogo dei processi o dei passaggi necessari per gestire un ambiente modernizzato.

PLC

Vedi controllore logico programmabile.

PLM

Vedi la gestione del ciclo di vita del prodotto.

policy

Un oggetto in grado di definire le autorizzazioni (vedi politica basata sull'identità), specificare le condizioni di accesso (vedi politicabasata sulle risorse) o definire le autorizzazioni massime per tutti gli account di un'organizzazione in (vedi politica di controllo dei servizi). AWS Organizations

persistenza poliglotta

Scelta indipendente della tecnologia di archiviazione di dati di un microservizio in base ai modelli di accesso ai dati e ad altri requisiti. Se i microservizi utilizzano la stessa tecnologia di archiviazione di dati, possono incontrare problemi di implementazione o registrare prestazioni scadenti. I microservizi vengono implementati più facilmente e ottengono prestazioni e scalabilità migliori se utilizzano l'archivio dati più adatto alle loro esigenze.

valutazione del portfolio

Un processo di scoperta, analisi e definizione delle priorità del portfolio di applicazioni per pianificare la migrazione. Per ulteriori informazioni, consulta la pagina Valutazione della preparazione alla migrazione.

predicate

Una condizione di interrogazione che restituisce o, in genere, si trova in una clausola`WHERE`. `true` `false`

predicato pushdown

Una tecnica di ottimizzazione delle query del database che filtra i dati della query prima del trasferimento. Ciò riduce la quantità di dati che devono essere recuperati ed elaborati dal database relazionale e migliora le prestazioni delle query.

controllo preventivo

Un controllo di sicurezza progettato per impedire il verificarsi di un evento. Questi controlli sono la prima linea di difesa per impedire accessi non autorizzati o modifiche indesiderate alla rete. Per ulteriori informazioni, consulta Controlli preventivi in Implementazione dei controlli di sicurezza in AWS.

principale

Un'entità in AWS grado di eseguire azioni e accedere alle risorse. Questa entità è in genere un utente root per un Account AWS ruolo IAM o un utente. Per ulteriori informazioni, consulta Principali in Termini e concetti dei ruoli nella documentazione di IAM.

privacy fin dalla progettazione

Un approccio di ingegneria dei sistemi che tiene conto della privacy durante l'intero processo di sviluppo.

zone ospitate private

Un contenitore che contiene informazioni su come desideri che Amazon Route 53 risponda alle query DNS per un dominio e i relativi sottodomini all'interno di uno o più. VPCs Per ulteriori informazioni, consulta Utilizzo delle zone ospitate private nella documentazione di Route 53.

controllo proattivo

Un controllo di sicurezza progettato per impedire l'implementazione di risorse non conformi. Questi controlli analizzano le risorse prima del loro provisioning. Se la risorsa non è conforme al controllo, non viene fornita. Per ulteriori informazioni, consulta la guida di riferimento sui controlli nella AWS Control Tower documentazione e consulta Controlli proattivi in Implementazione dei controlli di sicurezza su. AWS

gestione del ciclo di vita del prodotto (PLM)

La gestione dei dati e dei processi di un prodotto durante l'intero ciclo di vita, dalla progettazione, sviluppo e lancio, attraverso la crescita e la maturità, fino al declino e alla rimozione.

Ambiente di produzione

Vedi ambiente.

controllore logico programmabile (PLC)

Nella produzione, un computer altamente affidabile e adattabile che monitora le macchine e automatizza i processi di produzione.

concatenamento rapido

Utilizzo dell'output di un prompt LLM come input per il prompt successivo per generare risposte migliori. Questa tecnica viene utilizzata per suddividere un'attività complessa in sottoattività o per perfezionare o espandere iterativamente una risposta preliminare. Aiuta a migliorare l'accuratezza e la pertinenza delle risposte di un modello e consente risultati più granulari e personalizzati.

pseudonimizzazione

Il processo di sostituzione degli identificatori personali in un set di dati con valori segnaposto. La pseudonimizzazione può aiutare a proteggere la privacy personale. I dati pseudonimizzati sono ancora considerati dati personali.

publish/subscribe (pub/sub)

Un modello che consente comunicazioni asincrone tra microservizi per migliorare la scalabilità e la reattività. Ad esempio, in un MES basato su microservizi, un microservizio può pubblicare messaggi di eventi su un canale a cui altri microservizi possono abbonarsi. Il sistema può aggiungere nuovi microservizi senza modificare il servizio di pubblicazione.

Q

Piano di query

Una serie di passaggi, come le istruzioni, utilizzati per accedere ai dati in un sistema di database relazionale SQL.

regressione del piano di query

Quando un ottimizzatore del servizio di database sceglie un piano non ottimale rispetto a prima di una determinata modifica all'ambiente di database. Questo può essere causato da modifiche a statistiche, vincoli, impostazioni dell'ambiente, associazioni dei parametri di query e aggiornamenti al motore di database.

R

Matrice RACI

Vedi responsabile, responsabile, consultato, informato (RACI).

RAG

Vedi Retrieval Augmented Generation.

ransomware

Un software dannoso progettato per bloccare l'accesso a un sistema informatico o ai dati fino a quando non viene effettuato un pagamento.

Matrice RASCI

Vedi responsabile, responsabile, consultato, informato (RACI).

RCAC

Vedi controllo dell'accesso a righe e colonne.

replica di lettura

Una copia di un database utilizzata per scopi di sola lettura. È possibile indirizzare le query alla replica di lettura per ridurre il carico sul database principale.

riprogettare

Vedi 7 Rs.

obiettivo del punto di ripristino (RPO)

Il periodo di tempo massimo accettabile dall'ultimo punto di ripristino dei dati. Questo determina ciò che si considera una perdita di dati accettabile tra l'ultimo punto di ripristino e l'interruzione del servizio.

obiettivo del tempo di ripristino (RTO)

Il ritardo massimo accettabile tra l'interruzione del servizio e il ripristino del servizio.

rifattorizzare

Vedi 7 R.

Region

Una raccolta di AWS risorse in un'area geografica. Ciascuna Regione AWS è isolata e indipendente dalle altre per fornire tolleranza agli errori, stabilità e resilienza. Per ulteriori informazioni, consulta Specificare cosa può usare Regioni AWS il tuo account.

regressione

Una tecnica di ML che prevede un valore numerico. Ad esempio, per risolvere il problema "A che prezzo verrà venduta questa casa?" un modello di ML potrebbe utilizzare un modello di regressione lineare per prevedere il prezzo di vendita di una casa sulla base di dati noti sulla casa (ad esempio, la metratura).

riospitare

Vedi 7 R.

rilascio

In un processo di implementazione, l'atto di promuovere modifiche a un ambiente di produzione.

trasferisco

Vedi 7 Rs.

ripiattaforma

Vedi 7 Rs.

riacquisto

Vedi 7 Rs.

resilienza

La capacità di un'applicazione di resistere alle interruzioni o di ripristinarle. L'elevata disponibilità e il disaster recovery sono considerazioni comuni quando si pianifica la resilienza in. Cloud AWSPer ulteriori informazioni, vedere Cloud AWS Resilience.

policy basata su risorse

Una policy associata a una risorsa, ad esempio un bucket Amazon S3, un endpoint o una chiave di crittografia. Questo tipo di policy specifica a quali principali è consentito l'accesso, le azioni supportate e qualsiasi altra condizione che deve essere soddisfatta.

matrice di assegnazione di responsabilità (RACI)

Una matrice che definisce i ruoli e le responsabilità di tutte le parti coinvolte nelle attività di migrazione e nelle operazioni cloud. Il nome della matrice deriva dai tipi di responsabilità definiti nella matrice: responsabile (R), responsabile (A), consultato (C) e informato (I). Il tipo di supporto (S) è facoltativo. Se includi il supporto, la matrice viene chiamata matrice RASCI e, se la escludi, viene chiamata matrice RACI.

controllo reattivo

Un controllo di sicurezza progettato per favorire la correzione di eventi avversi o deviazioni dalla baseline di sicurezza. Per ulteriori informazioni, consulta Controlli reattivi in Implementazione dei controlli di sicurezza in AWS.

retain

Vedi 7 R.

andare in pensione

Vedi 7 Rs.

Retrieval Augmented Generation (RAG)

Una tecnologia di intelligenza artificiale generativa in cui un LLM fa riferimento a una fonte di dati autorevole esterna alle sue fonti di dati di formazione prima di generare una risposta. Ad esempio, un modello RAG potrebbe eseguire una ricerca semantica nella knowledge base o nei dati personalizzati di un'organizzazione. Per ulteriori informazioni, consulta Cos'è il RAG.

rotazione

Processo di aggiornamento periodico di un segreto per rendere più difficile l'accesso alle credenziali da parte di un utente malintenzionato.

controllo dell'accesso a righe e colonne (RCAC)

L'uso di espressioni SQL di base e flessibili con regole di accesso definite. RCAC è costituito da autorizzazioni di riga e maschere di colonna.

RPO

Vedi obiettivo del punto di ripristino.

VERSO

Vedi obiettivo del tempo di ripristino.

runbook

Un insieme di procedure manuali o automatizzate necessarie per eseguire un'attività specifica. In genere sono progettati per semplificare operazioni o procedure ripetitive con tassi di errore elevati.

S

SAML 2.0

Uno standard aperto utilizzato da molti provider di identità (IdPs). Questa funzionalità abilita il single sign-on (SSO) federato, in modo che gli utenti possano accedere Console di gestione AWS o chiamare le operazioni AWS API senza che tu debba creare un utente in IAM per tutti i membri dell'organizzazione. Per ulteriori informazioni sulla federazione basata su SAML 2.0, consulta Informazioni sulla federazione basata su SAML 2.0 nella documentazione di IAM.

SCADA

Vedi controllo di supervisione e acquisizione dati.

SCP

Vedi la politica di controllo del servizio.

Secret

In Gestione dei segreti AWS, informazioni riservate o riservate, come una password o le credenziali utente, archiviate in forma crittografata. È costituito dal valore segreto e dai relativi metadati. Il valore segreto può essere binario, una stringa singola o più stringhe. Per ulteriori informazioni, consulta Cosa c'è in un segreto di Secrets Manager? nella documentazione di Secrets Manager.

sicurezza fin dalla progettazione

Un approccio di ingegneria dei sistemi che tiene conto della sicurezza durante l'intero processo di sviluppo.

controllo di sicurezza

Un guardrail tecnico o amministrativo che impedisce, rileva o riduce la capacità di un autore di minacce di sfruttare una vulnerabilità di sicurezza. Esistono quattro tipi principali di controlli di sicurezza: preventivi, investigativi, reattivi e proattivi.

rafforzamento della sicurezza

Il processo di riduzione della superficie di attacco per renderla più resistente agli attacchi. Può includere azioni come la rimozione di risorse che non sono più necessarie, l'implementazione di best practice di sicurezza che prevedono la concessione del privilegio minimo o la disattivazione di funzionalità non necessarie nei file di configurazione.

sistema di gestione delle informazioni e degli eventi di sicurezza (SIEM)

Strumenti e servizi che combinano sistemi di gestione delle informazioni di sicurezza (SIM) e sistemi di gestione degli eventi di sicurezza (SEM). Un sistema SIEM raccoglie, monitora e analizza i dati da server, reti, dispositivi e altre fonti per rilevare minacce e violazioni della sicurezza e generare avvisi.

automazione della risposta alla sicurezza

Un'azione predefinita e programmata progettata per rispondere o porre rimedio automaticamente a un evento di sicurezza. Queste automazioni fungono da controlli di sicurezza investigativi o reattivi che aiutano a implementare le migliori pratiche di sicurezza. AWS Esempi di azioni di risposta automatizzate includono la modifica di un gruppo di sicurezza VPC, l'applicazione di patch a un'istanza Amazon EC2 o la rotazione delle credenziali.

Crittografia lato server

Crittografia dei dati a destinazione, da parte di chi li riceve. Servizio AWS

Policy di controllo dei servizi (SCP)

Una politica che fornisce il controllo centralizzato sulle autorizzazioni per tutti gli account di un'organizzazione in. AWS Organizations SCPs definire barriere o fissare limiti alle azioni che un amministratore può delegare a utenti o ruoli. È possibile utilizzarli SCPs come elenchi consentiti o elenchi di rifiuto, per specificare quali servizi o azioni sono consentiti o proibiti. Per ulteriori informazioni, consulta le politiche di controllo del servizio nella AWS Organizations documentazione.

endpoint del servizio

L'URL del punto di ingresso per un Servizio AWS. Puoi utilizzare l'endpoint per connetterti a livello di programmazione al servizio di destinazione. Per ulteriori informazioni, consulta Endpoint del Servizio AWS nei Riferimenti generali di AWS.

accordo sul livello di servizio (SLA)

Un accordo che chiarisce ciò che un team IT promette di offrire ai propri clienti, ad esempio l'operatività e le prestazioni del servizio.

indicatore del livello di servizio (SLI)

Misurazione di un aspetto prestazionale di un servizio, ad esempio il tasso di errore, la disponibilità o la velocità effettiva.

obiettivo a livello di servizio (SLO)

Una metrica target che rappresenta lo stato di un servizio, misurato da un indicatore del livello di servizio.

Modello di responsabilità condivisa

Un modello che descrive la responsabilità condivisa AWS per la sicurezza e la conformità del cloud. AWS è responsabile della sicurezza del cloud, mentre tu sei responsabile della sicurezza nel cloud. Per ulteriori informazioni, consulta Modello di responsabilità condivisa.

SIEM

Vedi il sistema di gestione delle informazioni e degli eventi sulla sicurezza.

punto di errore singolo (SPOF)

Un guasto in un singolo componente critico di un'applicazione che può disturbare il sistema.

SLAM

Vedi il contratto sul livello di servizio.

SLI

Vedi l'indicatore del livello di servizio.

LENTA

Vedi obiettivo del livello di servizio.

split-and-seed modello

Un modello per dimensionare e accelerare i progetti di modernizzazione. Man mano che vengono definite nuove funzionalità e versioni dei prodotti, il team principale si divide per creare nuovi team di prodotto. Questo aiuta a dimensionare le capacità e i servizi dell'organizzazione, migliora la produttività degli sviluppatori e supporta una rapida innovazione. Per ulteriori informazioni, vedere Approccio graduale alla modernizzazione delle applicazioni in. Cloud AWS

SPOF

Vedi punto di errore singolo.

schema a stella

Una struttura organizzativa di database che utilizza un'unica tabella dei fatti di grandi dimensioni per archiviare i dati transazionali o misurati e utilizza una o più tabelle dimensionali più piccole per memorizzare gli attributi dei dati. Questa struttura è progettata per l'uso in un data warehouse o per scopi di business intelligence.

modello del fico strangolatore

Un approccio alla modernizzazione dei sistemi monolitici mediante la riscrittura e la sostituzione incrementali delle funzionalità del sistema fino alla disattivazione del sistema legacy. Questo modello utilizza l'analogia di una pianta di fico che cresce fino a diventare un albero robusto e alla fine annienta e sostituisce il suo ospite. Il modello è stato introdotto da Martin Fowler come metodo per gestire il rischio durante la riscrittura di sistemi monolitici. Per un esempio di come applicare questo modello, consulta Modernizzazione incrementale dei servizi Web legacy di Microsoft ASP.NET (ASMX) mediante container e Gateway Amazon API.

sottorete

Un intervallo di indirizzi IP nel VPC. Una sottorete deve risiedere in una singola zona di disponibilità.

controllo di supervisione e acquisizione dati (SCADA)

Nella produzione, un sistema che utilizza hardware e software per monitorare gli asset fisici e le operazioni di produzione.

crittografia simmetrica

Un algoritmo di crittografia che utilizza la stessa chiave per crittografare e decrittografare i dati.

test sintetici

Test di un sistema in modo da simulare le interazioni degli utenti per rilevare potenziali problemi o monitorare le prestazioni. Puoi usare Amazon CloudWatch Synthetics per creare questi test.

prompt di sistema

Una tecnica per fornire contesto, istruzioni o linee guida a un LLM per indirizzarne il comportamento. I prompt di sistema aiutano a impostare il contesto e stabilire regole per le interazioni con gli utenti.

T

tag

Coppie chiave-valore che fungono da metadati per l'organizzazione delle risorse. AWS Con i tag è possibile a gestire, identificare, organizzare, cercare e filtrare le risorse. Per ulteriori informazioni, consulta Tagging delle risorse AWS.

variabile di destinazione

Il valore che stai cercando di prevedere nel machine learning supervisionato. Questo è indicato anche come variabile di risultato. Ad esempio, in un ambiente di produzione la variabile di destinazione potrebbe essere un difetto del prodotto.

elenco di attività

Uno strumento che viene utilizzato per tenere traccia dei progressi tramite un runbook. Un elenco di attività contiene una panoramica del runbook e un elenco di attività generali da completare. Per ogni attività generale, include la quantità stimata di tempo richiesta, il proprietario e lo stato di avanzamento.

ambiente di test

Vedi ambiente.

training

Fornire dati da cui trarre ispirazione dal modello di machine learning. I dati di training devono contenere la risposta corretta. L'algoritmo di apprendimento trova nei dati di addestramento i pattern che mappano gli attributi dei dati di input al target (la risposta che si desidera prevedere). Produce un modello di ML che acquisisce questi modelli. Puoi quindi utilizzare il modello di ML per creare previsioni su nuovi dati di cui non si conosce il target.

Transit Gateway

Un hub di transito di rete che puoi utilizzare per interconnettere le tue reti VPCs e quelle locali. Per ulteriori informazioni, consulta Cos'è un gateway di transito nella AWS Transit Gateway documentazione.

flusso di lavoro basato su trunk

Un approccio in cui gli sviluppatori creano e testano le funzionalità localmente in un ramo di funzionalità e quindi uniscono tali modifiche al ramo principale. Il ramo principale viene quindi integrato negli ambienti di sviluppo, preproduzione e produzione, in sequenza.

Accesso attendibile

Concessione delle autorizzazioni a un servizio specificato dall'utente per eseguire attività all'interno dell'organizzazione AWS Organizations e nei suoi account per conto dell'utente. Il servizio attendibile crea un ruolo collegato al servizio in ogni account, quando tale ruolo è necessario, per eseguire attività di gestione per conto dell'utente. Per ulteriori informazioni, consulta Utilizzo AWS Organizations con altri AWS servizi nella AWS Organizations documentazione.

regolazione

Modificare alcuni aspetti del processo di training per migliorare la precisione del modello di ML. Ad esempio, puoi addestrare il modello di ML generando un set di etichette, aggiungendo etichette e quindi ripetendo questi passaggi più volte con impostazioni diverse per ottimizzare il modello.

team da due pizze

Una piccola DevOps squadra che puoi sfamare con due pizze. Un team composto da due persone garantisce la migliore opportunità possibile di collaborazione nello sviluppo del software.

U

incertezza

Un concetto che si riferisce a informazioni imprecise, incomplete o sconosciute che possono minare l'affidabilità dei modelli di machine learning predittivi. Esistono due tipi di incertezza: l'incertezza epistemica, che è causata da dati limitati e incompleti, mentre l'incertezza aleatoria è causata dal rumore e dalla casualità insiti nei dati. Per ulteriori informazioni, consulta la guida Quantificazione dell'incertezza nei sistemi di deep learning.

compiti indifferenziati

Conosciuto anche come sollevamento di carichi pesanti, è un lavoro necessario per creare e far funzionare un'applicazione, ma che non apporta valore diretto all'utente finale né offre vantaggi competitivi. Esempi di attività indifferenziate includono l'approvvigionamento, la manutenzione e la pianificazione della capacità.

ambienti superiori

Vedi ambiente.

V

vacuum

Un'operazione di manutenzione del database che prevede la pulizia dopo aggiornamenti incrementali per recuperare lo spazio di archiviazione e migliorare le prestazioni.

controllo delle versioni

Processi e strumenti che tengono traccia delle modifiche, ad esempio le modifiche al codice di origine in un repository.

Peering VPC

Una connessione tra due VPCs che consente di indirizzare il traffico utilizzando indirizzi IP privati. Per ulteriori informazioni, consulta Che cos'è il peering VPC? nella documentazione di Amazon VPC.

vulnerabilità

Un difetto software o hardware che compromette la sicurezza del sistema.

W

cache calda

Una cache del buffer che contiene dati correnti e pertinenti a cui si accede frequentemente. L'istanza di database può leggere dalla cache del buffer, il che richiede meno tempo rispetto alla lettura dalla memoria dal disco principale.

dati caldi

Dati a cui si accede raramente. Quando si eseguono interrogazioni di questo tipo di dati, in genere sono accettabili query moderatamente lente.

funzione finestra

Una funzione SQL che esegue un calcolo su un gruppo di righe che si riferiscono in qualche modo al record corrente. Le funzioni della finestra sono utili per l'elaborazione di attività, come il calcolo di una media mobile o l'accesso al valore delle righe in base alla posizione relativa della riga corrente.

Carico di lavoro

Una raccolta di risorse e codice che fornisce valore aziendale, ad esempio un'applicazione rivolta ai clienti o un processo back-end.

flusso di lavoro

Gruppi funzionali in un progetto di migrazione responsabili di una serie specifica di attività. Ogni flusso di lavoro è indipendente ma supporta gli altri flussi di lavoro del progetto. Ad esempio, il flusso di lavoro del portfolio è responsabile della definizione delle priorità delle applicazioni, della pianificazione delle ondate e della raccolta dei metadati di migrazione. Il flusso di lavoro del portfolio fornisce queste risorse al flusso di lavoro di migrazione, che quindi migra i server e le applicazioni.

VERME

Vedi scrivere una volta, leggere molti.

WQF

Vedi AWS Workload Qualification Framework.

scrivi una volta, leggi molte (WORM)

Un modello di storage che scrive i dati una sola volta e ne impedisce l'eliminazione o la modifica. Gli utenti autorizzati possono leggere i dati tutte le volte che è necessario, ma non possono modificarli. Questa infrastruttura di archiviazione dei dati è considerata immutabile.

Z

exploit zero-day

Un attacco, in genere malware, che sfrutta una vulnerabilità zero-day.

vulnerabilità zero-day

Un difetto o una vulnerabilità assoluta in un sistema di produzione. Gli autori delle minacce possono utilizzare questo tipo di vulnerabilità per attaccare il sistema. Gli sviluppatori vengono spesso a conoscenza della vulnerabilità causata dall'attacco.

prompt zero-shot

Fornire a un LLM le istruzioni per eseguire un'attività ma non esempi (immagini) che possano aiutarla. Il LLM deve utilizzare le sue conoscenze pre-addestrate per gestire l'attività. L'efficacia del prompt zero-shot dipende dalla complessità dell'attività e dalla qualità del prompt. Vedi anche few-shot prompting.

applicazione zombie

Un'applicazione che prevede un utilizzo CPU e memoria inferiore al 5%. In un progetto di migrazione, è normale ritirare queste applicazioni.

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.