EBook Gratis

APRENDIZAJE amazon-web-services

Free unaffiliated eBook created from **Stack Overflow contributors.**

#amazon-web-services

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: amazon-web-services

It is an unofficial and free amazon-web-services ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official amazon-web-services.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Primeros pasos con amazon-web-services

Observaciones

Esta sección proporciona una descripción general de qué es amazon-web-services y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema importante dentro de amazon-web-services y vincular a los temas relacionados. Dado que la Documentación para los servicios web de amazon es nueva, es posible que deba crear versiones iniciales de esos temas relacionados.

Versiones

Versión	Fecha de lanzamiento
1.0.0	2017-01-10

Examples

Antes de que sea demasiado tarde

Consejos y trucos para evitar situaciones desagradables

Instancias EC2 y EBS

- Establecer roles IAM.

A diferencia de las etiquetas, el rol IAM se **establece de una vez por todas en la instanciación de EC2** (incluso después de 4 años). Intente identificar y categorizar de antemano sus instancias para que pueda asignarles los roles de IAM adecuados. Los roles IAM son una buena forma de identificar sus máquinas, permitirá que Amazon almacene automáticamente las credenciales del perfil de instancia de forma segura en sus máquinas y podrá otorgar privilegios adicionales fácilmente.

Considere la siguiente situación en la que tiene servidores de bases de datos y se da cuenta de que desea monitorear el uso de la memoria / disco. Amazon CloudWatch no proporciona esta métrica fuera de la caja, y tendrá que configurar privilegios adicionales para enviar datos personalizados a CloudWatch. Si tiene un rol de "Base de datos" de IAM, puede adjuntar fácilmente nuevas políticas a sus instancias de base de datos existentes para permitirles enviar informes de memoria a CloudWatch. No hay roles IAM? Tienes que volver a crear tus instancias de base de datos, o darles permiso individualmente.

- Cuidado con la integridad de la instantánea

Amazon le permite captar volúmenes de EBS, sin embargo, en caso de que use varios volúmenes en la misma máquina (en la configuración RAID, varios volúmenes de EBS para su base de datos), es imposible garantizar la integridad de esas instantáneas, lo que puede ocurrir en diferentes momentos en el Diferentes volúmenes de EBS.

Siempre asegúrese de que no se escriban datos (detenga la máquina virtual o use un código específico de la aplicación (por ejemplo, `db.fsyncLock()`) para asegurarse de que no se escriban datos durante la instantánea).

CloudWatch

- Utilice las alertas de Amazon Cloudwatch + SNS sobre los notificadores de error de su aplicación

Cree alertas para un comportamiento normal de sus máquinas y configúrelas para enviar notificaciones a través de Amazon SNS (por ejemplo, direcciones de correo electrónico) en caso de problemas. Tener notificadores de excepción en su aplicación no ayudará si su servidor ni siquiera puede hacer ping. Por otro lado, aparte de los 500 códigos de error, Amazon tiene poca información sobre su aplicación y sobre cómo se supone que funciona, debe considerar agregar monitoreo de estado específico de la aplicación.

Lea Primeros pasos con amazon-web-services en línea: https://riptutorial.com/es/amazon-web-services/topic/798/primeros-pasos-con-amazon-web-services

Capítulo 2: Amazon Cognito

Examples

Gestión de la identidad del usuario utilizando Amazon Cognito.

```
var app = {};

app.signUp = function(){

app.userName        =           $('#userName').val();
app.password        =           $('#password').val();
app.email           =           $('#form-email').val();
app.phoneNumber     =           $('#form-phone').val();
app.emailRegex      =
/^(([^<>()\[\]\.,;:\s@\"]+(\.[^<>()\[\]\.,;:\s@\"]+)*)|(\".+\"))@(([^<>()[\]\.,;:\s@\"]+\.)+[^<>()[\]\


/*
    Put the User input validation logic here.
*/
if (!app.userName) {
    alert("Please provide a user name");
    return;
}

if (!app.password) {
    alert("Please provide a password");
    return;
}

if (!app.email) {
    alert("Please provide an Email address");
    return;
}

if(!app.emailRegex.test(app.email)){
   alert("Please provide a valid Email address");
   return;
}

if (!app.phoneNumber) {
    alert("Please provide a Phone Number");
    return;
}

AWS.config.region = 'us-east-1'; // Region
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
IdentityPoolId: '...' // your identity pool id here
});

AWSCognito.config.region = 'us-east-1';
AWSCognito.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: '...' // your identity pool id here
});

// Need to provide placeholder keys unless unauthorised user access is enabled for user pool
```

```
AWSCognito.config.update({accessKeyId: 'anything', secretAccessKey: 'anything'})

var poolData = {
    UserPoolId :     APP_CONSTANT.USER_POOL_ID,
    ClientId :     APP_CONSTANT.CLIENT_ID
};
userPool = new   AWSCognito.CognitoIdentityServiceProvider.CognitoUserPool(poolData);

var attributeList = [];

var dataEmail = {
    Name : 'email',
    Value : app.email //Email Id where the confirmation code would be sent.
};
var dataPhoneNumber = {
    Name : 'phone_number',
    Value : app.phoneNumber
};
var attributeEmail = new
AWSCognito.CognitoIdentityServiceProvider.CognitoUserAttribute(dataEmail);
//Uncomment below once the phone number format is confirmed
/*var attributePhoneNumber = new
AWSCognito.CognitoIdentityServiceProvider.CognitoUserAttribute(dataPhoneNumber);*/

attributeList.push(attributeEmail);
 /* attributeList.push(attributePhoneNumber);*/
// Put the user id and password collected from user below for signup
userPool.signUp(app.userName, app.password, attributeList, null, function(err, result){
    if (err) {
        alert(err);
        return;
    }
    cognitoUser = result.user;
// Return the user name once the user signed up, still need to confirm with confirmation code
send to mail.
$("#form-confirmCode").css("display", "block");
    alert('user name is ' + cognitoUser.getUsername());
// Now since the user is signed up and pending for confirmaion, disable all the pervious input
but confirmation code.
$("#userName").prop("readonly", true);
$("#password").prop("readonly", true);
$("#form-email").prop("readonly", true);
$("#form-phone").prop("readonly", true);
$("#signUpBtn").hide();
$("#confirm-block").show();

var confirmationCode = prompt("Hello "+cognitoUser.getUsername+" Enter the confirmation code
sent to your email address.","Confirmation code here");
cognitoUser.confirmRegistration(confirmationCode,true,function(err,result){
            if(err){
                    alert(err);
                    return;
            }
            console.log('Call Result: '+result);
    });
return;

});
};
```

Lea Amazon Cognito en línea: https://riptutorial.com/es/amazon-web-services/topic/3425/amazon-

cognito

Capítulo 3: Amazon DynamoDB

Examples

Operación Crud básica de DynamoDB usando NodeJS

```
let doc = require('dynamodb-doc');
let dynamo = new doc.DynamoDB();
var tblName = "MyTable";

exports.handler = (event, context, callback) => {
    readOperation(context);
}

function readOperation(cnxt) {
    var params = {
        TableName: tblName,
        Key: {
            "id": "2013",
            "topic": "Turn It Down, Or Else!"
        },
        AttributesToGet: [
            "id", "client_name", "info"
        ],
        ConsistentRead: false
    };
    dynamo.getItem(params, function(err, data) {
        if (err) console.log("Error: "+err); // an error occurred
        else {
            var jsonDoc = JSON.parse(data.Item.info);   // successful response
            cnxt.succeed(jsonDoc);
        }
    });
}
```

Lea Amazon DynamoDB en línea: https://riptutorial.com/es/amazon-web-services/topic/6429/amazon-dynamodb

Capítulo 4: AWS CloudFormation

Examples

CloudFormation script de ejemplo para crear una instancia de EC2 junto con un grupo de seguridad para asociarse.

En este ejemplo se creará una instancia EC2 de **t2.micro** tipo en la región **amazónica N.Virginia** corriendo **Linux.** Durante la ejecución, le pedirá que seleccione el KeyPair para usar y un IP CIDR desde donde puede SSH a la instancia, use el valor predeterminado para hacer que SSH se abra a Internet.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template EC2InstanceWithSecurityGroupSample:
Create an Amazon EC2 instance running the Amazon Linux AMI. This example creates an EC2
security group for the instance to give you SSH access. ",

  "Parameters" : {
    "KeyName": {
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the instance",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription" : "must be the name of an existing EC2 KeyPair."
    },
    "SSHLocation" : {
      "Description" : "The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
   }
  },

  "Resources" : {
    "EC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "InstanceType" : "t2.micro",
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : "ami-6869aa05"
      }
    },

    "InstanceSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable SSH access via port 22",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "22",
```

```
          "ToPort" : "22",
          "CidrIp" : { "Ref" : "SSHLocation"}
        } ]
      }
    }
  },

  "Outputs" : {
    "InstanceId" : {
      "Description" : "InstanceId of the newly created EC2 instance",
      "Value" : { "Ref" : "EC2Instance" }
    },
    "AZ" : {
      "Description" : "Availability Zone of the newly created EC2 instance",
      "Value" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone" ] }
    },
    "PublicDNS" : {
      "Description" : "Public DNSName of the newly created EC2 instance",
      "Value" : { "Fn::GetAtt" : [ "EC2Instance", "PublicDnsName" ] }
    },
    "PublicIP" : {
      "Description" : "Public IP address of the newly created EC2 instance",
      "Value" : { "Fn::GetAtt" : [ "EC2Instance", "PublicIp" ] }
    }
  }
}
```

AWS CloudFormer en VPC

La plantilla de CloudFormer convierte las entidades de AWS existentes en una nueva plantilla de CloudFormation para acelerar el tiempo para recrear el entorno. El CloudFormer se inicia en una VPC predeterminada que puede no ser adecuada en los casos en que se elimina la VPC predeterminada. Esta base de código es una bifurcación del CloudFormer original que se lanzaría dentro de una nueva VPC.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "A custom CloudFormer template forked from AWS provided to extend the
capability to provide the ability to specify the VPC. Creates a Separate Stand-Alone VPC,
Subnet - 10.0.0.0/16 to launch the CloudFormer Instance. AWS CloudFormer Beta - template
creation prototype application. This tool allows you to create an AWS CloudFormation template
from the AWS resources in your AWS account. **Warning** This template creates a single EC2
instance in your account to run the application - you will be billed for the instance at
normal AWS EC2 rates.",

  "Parameters" : {

    "Username" : {
      "Description" : "Username to log in to CloudFormer",
      "Type" : "String"
    },
    "Password" : {
      "Description" : "Password to log in to CloudFormer",
      "Type" : "String",
      "NoEcho" : "true"
    },
    "CommonNameTag" : {
```

```
        "Description" : "Common Identifier / Friendly Name for the Stack",
        "Type" : "String",
        "Default" : "CloudFormer - Non Default VPC"
    }
  },

  "Mappings" : {
    "NetworkValues" : {
        "VPC" : {"CIDR" : "10.0.0.0/16"},
        "Subnet" : {"CIDR" : "10.0.10.0/16"}
    },

    "Region2Examples" : {
      "us-east-1"      : { "Examples" : "https://s3.amazonaws.com/cloudformation-examples-us-
east-1" },
      "us-west-2"      : { "Examples" : "https://s3-us-west-2.amazonaws.com/cloudformation-
examples-us-west-2" },
      "us-west-1"      : { "Examples" : "https://s3-us-west-1.amazonaws.com/cloudformation-
examples-us-west-1" },
      "eu-west-1"      : { "Examples" : "https://s3-eu-west-1.amazonaws.com/cloudformation-
examples-eu-west-1" },
      "eu-central-1"   : { "Examples" : "https://s3-eu-central-1.amazonaws.com/cloudformation-
examples-eu-central-1" },
      "ap-southeast-1" : { "Examples" : "https://s3-ap-southeast-
1.amazonaws.com/cloudformation-examples-ap-southeast-1" },
      "ap-northeast-1" : { "Examples" : "https://s3-ap-northeast-
1.amazonaws.com/cloudformation-examples-ap-northeast-1" },
      "ap-southeast-2" : { "Examples" : "https://s3-ap-southeast-
2.amazonaws.com/cloudformation-examples-ap-southeast-2" },
      "ap-northeast-2" : { "Examples" : "https://s3-ap-northeast-
2.amazonaws.com/cloudformation-examples-ap-northeast-2" },
      "sa-east-1"      : { "Examples" : "https://s3-sa-east-1.amazonaws.com/cloudformation-
examples-sa-east-1" },
      "cn-north-1"     : { "Examples" : "https://s3.cn-north-
1.amazonaws.com.cn/cloudformation-examples-cn-north-1" }
    },

    "Region2Principal" : {
      "us-east-1"      : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "us-west-2"      : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "us-west-1"      : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "eu-west-1"      : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "ap-southeast-1" : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "ap-northeast-1" : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "ap-southeast-2" : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "ap-northeast-2" : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "sa-east-1"      : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" },
      "cn-north-1"     : { "EC2Principal" : "ec2.amazonaws.com.cn", "OpsWorksPrincipal" :
"opsworks.amazonaws.com.cn" },
      "eu-central-1"   : { "EC2Principal" : "ec2.amazonaws.com", "OpsWorksPrincipal" :
"opsworks.amazonaws.com" }
    },
```

```
  "AWSInstanceType2Arch" : {
    "t1.micro"    : { "Arch" : "PV64"   },
    "t2.nano"     : { "Arch" : "HVM64"  },
    "t2.micro"    : { "Arch" : "HVM64"  },
    "t2.small"    : { "Arch" : "HVM64"  },
    "t2.medium"   : { "Arch" : "HVM64"  },
    "t2.large"    : { "Arch" : "HVM64"  },
    "m1.small"    : { "Arch" : "PV64"   },
    "m1.medium"   : { "Arch" : "PV64"   },
    "m1.large"    : { "Arch" : "PV64"   },
    "m1.xlarge"   : { "Arch" : "PV64"   },
    "m2.xlarge"   : { "Arch" : "PV64"   },
    "m2.2xlarge"  : { "Arch" : "PV64"   },
    "m2.4xlarge"  : { "Arch" : "PV64"   },
    "m3.medium"   : { "Arch" : "HVM64"  },
    "m3.large"    : { "Arch" : "HVM64"  },
    "m3.xlarge"   : { "Arch" : "HVM64"  },
    "m3.2xlarge"  : { "Arch" : "HVM64"  },
    "m4.large"    : { "Arch" : "HVM64"  },
    "m4.xlarge"   : { "Arch" : "HVM64"  },
    "m4.2xlarge"  : { "Arch" : "HVM64"  },
    "m4.4xlarge"  : { "Arch" : "HVM64"  },
    "m4.10xlarge" : { "Arch" : "HVM64"  },
    "c1.medium"   : { "Arch" : "PV64"   },
    "c1.xlarge"   : { "Arch" : "PV64"   },
    "c3.large"    : { "Arch" : "HVM64"  },
    "c3.xlarge"   : { "Arch" : "HVM64"  },
    "c3.2xlarge"  : { "Arch" : "HVM64"  },
    "c3.4xlarge"  : { "Arch" : "HVM64"  },
    "c3.8xlarge"  : { "Arch" : "HVM64"  },
    "c4.large"    : { "Arch" : "HVM64"  },
    "c4.xlarge"   : { "Arch" : "HVM64"  },
    "c4.2xlarge"  : { "Arch" : "HVM64"  },
    "c4.4xlarge"  : { "Arch" : "HVM64"  },
    "c4.8xlarge"  : { "Arch" : "HVM64"  },
    "g2.2xlarge"  : { "Arch" : "HVMG2"  },
    "g2.8xlarge"  : { "Arch" : "HVMG2"  },
    "r3.large"    : { "Arch" : "HVM64"  },
    "r3.xlarge"   : { "Arch" : "HVM64"  },
    "r3.2xlarge"  : { "Arch" : "HVM64"  },
    "r3.4xlarge"  : { "Arch" : "HVM64"  },
    "r3.8xlarge"  : { "Arch" : "HVM64"  },
    "i2.xlarge"   : { "Arch" : "HVM64"  },
    "i2.2xlarge"  : { "Arch" : "HVM64"  },
    "i2.4xlarge"  : { "Arch" : "HVM64"  },
    "i2.8xlarge"  : { "Arch" : "HVM64"  },
    "d2.xlarge"   : { "Arch" : "HVM64"  },
    "d2.2xlarge"  : { "Arch" : "HVM64"  },
    "d2.4xlarge"  : { "Arch" : "HVM64"  },
    "d2.8xlarge"  : { "Arch" : "HVM64"  },
    "hi1.4xlarge" : { "Arch" : "HVM64"  },
    "hs1.8xlarge" : { "Arch" : "HVM64"  },
    "cr1.8xlarge" : { "Arch" : "HVM64"  },
    "cc2.8xlarge" : { "Arch" : "HVM64"  }
  },

  "AWSInstanceType2NATArch" : {
    "t1.micro"    : { "Arch" : "NATPV64"   },
    "t2.nano"     : { "Arch" : "NATHVM64"  },
    "t2.micro"    : { "Arch" : "NATHVM64"  },
```

```
      "t2.small"    : { "Arch" : "NATHVM64"  },
      "t2.medium"   : { "Arch" : "NATHVM64"  },
      "t2.large"    : { "Arch" : "NATHVM64"  },
      "m1.small"    : { "Arch" : "NATPV64"   },
      "m1.medium"   : { "Arch" : "NATPV64"   },
      "m1.large"    : { "Arch" : "NATPV64"   },
      "m1.xlarge"   : { "Arch" : "NATPV64"   },
      "m2.xlarge"   : { "Arch" : "NATPV64"   },
      "m2.2xlarge"  : { "Arch" : "NATPV64"   },
      "m2.4xlarge"  : { "Arch" : "NATPV64"   },
      "m3.medium"   : { "Arch" : "NATHVM64"  },
      "m3.large"    : { "Arch" : "NATHVM64"  },
      "m3.xlarge"   : { "Arch" : "NATHVM64"  },
      "m3.2xlarge"  : { "Arch" : "NATHVM64"  },
      "m4.large"    : { "Arch" : "NATHVM64"  },
      "m4.xlarge"   : { "Arch" : "NATHVM64"  },
      "m4.2xlarge"  : { "Arch" : "NATHVM64"  },
      "m4.4xlarge"  : { "Arch" : "NATHVM64"  },
      "m4.10xlarge" : { "Arch" : "NATHVM64"  },
      "c1.medium"   : { "Arch" : "NATPV64"   },
      "c1.xlarge"   : { "Arch" : "NATPV64"   },
      "c3.large"    : { "Arch" : "NATHVM64"  },
      "c3.xlarge"   : { "Arch" : "NATHVM64"  },
      "c3.2xlarge"  : { "Arch" : "NATHVM64"  },
      "c3.4xlarge"  : { "Arch" : "NATHVM64"  },
      "c3.8xlarge"  : { "Arch" : "NATHVM64"  },
      "c4.large"    : { "Arch" : "NATHVM64"  },
      "c4.xlarge"   : { "Arch" : "NATHVM64"  },
      "c4.2xlarge"  : { "Arch" : "NATHVM64"  },
      "c4.4xlarge"  : { "Arch" : "NATHVM64"  },
      "c4.8xlarge"  : { "Arch" : "NATHVM64"  },
      "g2.2xlarge"  : { "Arch" : "NATHVMG2"  },
      "g2.8xlarge"  : { "Arch" : "NATHVMG2"  },
      "r3.large"    : { "Arch" : "NATHVM64"  },
      "r3.xlarge"   : { "Arch" : "NATHVM64"  },
      "r3.2xlarge"  : { "Arch" : "NATHVM64"  },
      "r3.4xlarge"  : { "Arch" : "NATHVM64"  },
      "r3.8xlarge"  : { "Arch" : "NATHVM64"  },
      "i2.xlarge"   : { "Arch" : "NATHVM64"  },
      "i2.2xlarge"  : { "Arch" : "NATHVM64"  },
      "i2.4xlarge"  : { "Arch" : "NATHVM64"  },
      "i2.8xlarge"  : { "Arch" : "NATHVM64"  },
      "d2.xlarge"   : { "Arch" : "NATHVM64"  },
      "d2.2xlarge"  : { "Arch" : "NATHVM64"  },
      "d2.4xlarge"  : { "Arch" : "NATHVM64"  },
      "d2.8xlarge"  : { "Arch" : "NATHVM64"  },
      "hi1.4xlarge" : { "Arch" : "NATHVM64"  },
      "hs1.8xlarge" : { "Arch" : "NATHVM64"  },
      "cr1.8xlarge" : { "Arch" : "NATHVM64"  },
      "cc2.8xlarge" : { "Arch" : "NATHVM64"  }
    },

    "AWSRegionArch2AMI" : {
      "us-east-1"        : {"PV64" : "ami-d4f7ddbe", "HVM64" : "ami-2df5df47", "HVMG2" : "ami-
95f7c0ff"},
      "us-west-2"        : {"PV64" : "ami-a9ae4ec9", "HVM64" : "ami-42b15122", "HVMG2" : "ami-
83a744e3"},
      "us-west-1"        : {"PV64" : "ami-14f68074", "HVM64" : "ami-f7f58397", "HVMG2" : "ami-
ee62138e"},
      "eu-west-1"        : {"PV64" : "ami-a93484da", "HVM64" : "ami-3c38884f", "HVMG2" : "ami-
25d76556"},
```

```
      "eu-central-1"     : {"PV64" : "ami-e8233884", "HVM64" : "ami-d8203bb4", "HVMG2" : "ami-
8fadb6e3"},
      "ap-northeast-1"   : {"PV64" : "ami-c8aca8a6", "HVM64" : "ami-eeabaf80", "HVMG2" : "ami-
71e6e01f"},
      "ap-northeast-2"   : {"PV64" : "NOT_SUPPORTED", "HVM64" : "ami-431fd12d", "HVMG2" :
"NOT_SUPPORTED"},
      "ap-southeast-1"   : {"PV64" : "ami-6702cc04", "HVM64" : "ami-8504cae6", "HVMG2" : "ami-
1e7ab47d"},
      "ap-southeast-2"   : {"PV64" : "ami-4f04232c", "HVM64" : "ami-a30126c0", "HVMG2" : "ami-
68a1860b"},
      "sa-east-1"        : {"PV64" : "ami-daf477b6", "HVM64" : "ami-e2f4778e", "HVMG2" :
"NOT_SUPPORTED"},
      "cn-north-1"       : {"PV64" : "ami-0534fc68", "HVM64" : "ami-3f36fe52", "HVMG2" :
"NOT_SUPPORTED"}
    }

  },

  "Resources" : {
    "VPC" : {
      "Type" : "AWS::EC2::VPC",
      "Properties" : {
          "CidrBlock" : { "Fn::FindInMap" : ["NetworkValues", "VPC", "CIDR"] },
          "EnableDnsHostnames" : "true",
          "Tags" : [
              {"Key": "Name", "Value": {"Ref":"CommonNameTag"} }
          ]
      }
    },
    "Subnet" : {
       "Type" : "AWS::EC2::Subnet",
       "Properties" : {
           "VpcId" : { "Ref" : "VPC" },
           "CidrBlock" : { "Fn::FindInMap" : ["NetworkValues", "Subnet", "CIDR"] },
            "MapPublicIpOnLaunch" : "true",
           "Tags" : [
              {"Key": "Name", "Value": {"Ref":"CommonNameTag"} }
          ]
       }
    },

    "RouteTable" : {
       "Type" : "AWS::EC2::RouteTable",
       "Properties" : {
           "VpcId" : { "Ref" : "VPC" }
       }
    },

    "InternetGateway" : {
        "Type" : "AWS::EC2::InternetGateway",
        "Properties" : {
            "Tags" : [
                {"Key": "Name", "Value": {"Ref":"CommonNameTag"} }
            ]
        }
    },

    "InternetGatewayAttachment" : {
       "Type" : "AWS::EC2::VPCGatewayAttachment",
       "Properties" : {
           "InternetGatewayId" : { "Ref" : "InternetGateway"},
```

```
            "VpcId" : { "Ref" : "VPC"}
        }
    },

    "Route" : {
        "Type" : "AWS::EC2::Route",
        "Properties" : {
            "DestinationCidrBlock" : "0.0.0.0/0",
            "GatewayId" : { "Ref" : "InternetGateway" },
            "RouteTableId" : { "Ref" : "RouteTable" }
        }
    },

    "SubnetRouteTableAttachment" : {
         "Type" : "AWS::EC2::SubnetRouteTableAssociation",
         "Properties" : {
            "RouteTableId" : { "Ref" : "RouteTable" },
            "SubnetId" : { "Ref" : "Subnet" }
         }
    },

    "CFNRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [{
            "Effect": "Allow",
            "Principal": { "Service": { "Fn::FindInMap" : [ "Region2Principal", {"Ref" :
"AWS::Region"}, "EC2Principal"]}},
            "Action": [ "sts:AssumeRole" ]
          }]
        },
        "Path": "/"
      }
    },

    "CFNRolePolicy": {
      "Type": "AWS::IAM::Policy",
      "Properties": {
        "PolicyName": "CloudFormerPolicy",
        "PolicyDocument": {
          "Statement": [ {
            "Effect": "Allow",
            "Action"   : [
              "autoscaling:Describe*",
              "cloudformation:Describe*",
              "cloudformation:List*",
              "cloudfront:List*",
              "cloudFront:Get*",
              "cloudtrail:Describe*",
              "cloudtrail:Get*",
              "cloudwatch:Describe*",
              "dynamodb:List*",
              "dynamodb:Describe*",
              "elasticbeanstalk:Describe*",
              "ec2:Describe*",
              "elasticloadbalancing:Describe*",
              "elasticache:Describe*",
              "rds:Describe*",
              "rds:List*",
              "route53:List*",
```

```
            "route53:Get*",
            "s3:List*",
            "s3:Get*",
            "s3:PutObject",
            "sdb:Get*",
            "sdb:List*",
            "sns:Get*",
            "sns:List*",
            "sqs:Get*",
            "sqs:List*",
            "opsworks:Describe*",
            "redshift:Describe*",
            "kinesis:Describe*",
            "kinesis:List*"
          ],
          "Resource": "*"
        } ]
      },
      "Roles": [ { "Ref": "CFNRole" } ]
    }
  },

  "CFNInstanceProfile": {
    "Type": "AWS::IAM::InstanceProfile",
    "Properties": {
      "Path": "/",
      "Roles": [ { "Ref": "CFNRole" } ]
    }
  },

  "WebServer": {
    "Type": "AWS::EC2::Instance",
    "Metadata" : {
      "AWS::CloudFormation::Init" : {
        "configSets" : {
          "full_install" : ["base", "cloudformer"]
        },
        "base" : {
          "packages" : {
            "yum" : {
              "gcc"              : [],
              "gcc-c++"          : [],
              "make"             : [],
              "libxml2-devel"    : [],
              "libxslt-devel"    : [],
              "sqlite-devel"     : [],
              "patch"            : [],
              "readline"         : [],
              "readline-devel"   : [],
              "zlib"             : [],
              "zlib-devel"       : [],
              "libyaml-devel"    : [],
              "libffi-devel"     : [],
              "openssl-devel"    : [],
              "bzip2"            : [],
              "autoconf"         : [],
              "automake"         : [],
              "libtool"          : [],
              "bison"            : [],
              "ruby-devel"       : []
            }
```

```
            }
          },
          "cloudformer" : {
            "sources" : {
              "/home/ec2-user/cloudformer" : {"Fn::Join" : ["/", [
                                                {"Fn::FindInMap" : ["Region2Examples", {"Ref" :
"AWS::Region"}, "Examples"]},
                                                "AWSCloudFormer041.zip" ]]}
            },
            "files" : {
              "/home/ec2-user/setup_cloudformer" : {
                "content" : { "Fn::Join" : ["", [
                  "#!/usr/bin/env bash\n",
                  "cd /home/ec2-user/cloudformer\n",
                  "# Setup the CloudFormer service\n",
                  "mkdir -p vendor/bundle\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/rake-
10.4.2.gem\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/bundler-
1.7.11.gem\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/bundle-
0.0.1.gem\n",
                  "/usr/local/bin/bundle install --local --path /home/ec2-
user/cloudformer/vendor/bundle\n",
                  "/usr/local/bin/rake RAILS_ENV=production db:migrate\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/rack-
1.6.0.gem\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/eventmachine-
1.0.4.gem\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/daemons-
1.1.9.gem\n",
                  "gem install --local /home/ec2-user/cloudformer/vendor/cache/thin-
1.6.3.gem\n",
                  "# Create certificate and private key for SSL\n",
                  "mkdir -p /home/ec2-user/cloudformer/.ssl\n",
                  "cd /home/ec2-user/cloudformer/.ssl\n",
                  "openssl genrsa -des3 -passout pass:\"" , { "Ref" : "Password" }, "\" -out
server.pass.key 1024\n",
                  "openssl rsa -passin pass:\"", { "Ref" : "Password" }, "\" -in
server.pass.key -out server.key\n",
                  "openssl req -new -key server.key -out server.csr -subj
\"/C=US/ST=Washington/L=Seattle/O=Amazon Web Services/OU=CloudFormer\"\n",
                  "openssl x509 -req -days 365 -in server.csr -signkey server.key -out
server.crt\n",
                  "rm server.pass.key server.csr\n"
                ]]},
                "mode"  : "000755",
                "owner" : "root",
                "group" : "root"
              },
              "/home/ec2-user/cloudformer/config/initializers/user.rb" : {
                "content" : { "Fn::Join" : ["", [
                  "USER_NAME = \"", { "Ref" : "Username" }, "\"\n",
                  "PASSWORD = \"", { "Ref" : "Password" }, "\"\n"
                ]]},
                "mode"  : "000400",
                "owner" : "root",
                "group" : "root"
              },
              "/usr/bin/cloudformer" : {
                "content" : { "Fn::Join" : ["", [
```

```
                  "#!/usr/bin/env bash\n",
                  "cd /home/ec2-user/cloudformer\n",
                  "/usr/local/bin/thin start -p 443 -e production -d --ssl --ssl-key-file
/home/ec2-user/cloudformer/.ssl/server.key --ssl-cert-file /home/ec2-
user/cloudformer/.ssl/server.crt\n"
                ]]},
                "mode"  : "000755",
                "owner" : "root",
                "group" : "root"
              }
            },
            "commands" : {
              "01_install_cloudformer" : {
                "command" : "/home/ec2-user/setup_cloudformer &>
/var/log/setup_cloudformer.log",
                "cwd" : "/home/ec2-user/cloudformer"
              },
              "02_setup_boot" : {
                "command" : "echo '/usr/bin/cloudformer' >> /etc/rc.local",
                "cwd" : "/"
              }
            }
          }
        }
      },
      "Properties": {
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                          { "Fn::FindInMap" : [ "AWSInstanceType2Arch", "t2.medium", "Arch" ]
} ] },
        "InstanceType"       : "t2.medium",
        "SecurityGroupIds"     : [ {"Ref" : "WebServerSecurityGroup"} ],
        "SubnetId" : { "Ref" : "Subnet" },
        "IamInstanceProfile" : { "Ref" : "CFNInstanceProfile" },
        "UserData"           : { "Fn::Base64" : { "Fn::Join" : ["", [
          "#!/bin/bash -xe\n",
          "yum update -y aws-cfn-bootstrap\n",

          "/opt/aws/bin/cfn-init -v ",
          "         --stack ", { "Ref" : "AWS::StackId" },
          "         --resource WebServer ",
          "         --configsets full_install ",
          "         --region ", { "Ref" : "AWS::Region" }, "\n",

          "/opt/aws/bin/cfn-signal -e $? ",
          "         --stack ", { "Ref" : "AWS::StackId" },
          "         --resource WebServer ",
          "         --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
      },
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT30M"
        }
      }
    },

    "WebServerSecurityGroup" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable HTTPS access via port 443",
        "VpcId" : { "Ref" : "VPC" },
```

```
        "SecurityGroupIngress" : [
          {"IpProtocol" : "tcp", "FromPort" : "443", "ToPort" : "443", "CidrIp" : "0.0.0.0/0"}
        ]
      }
    }
  },

  "Outputs" : {
    "WebsiteURL" : {
      "Value" : { "Fn::Join" : ["", ["https://", { "Fn::GetAtt" : [ "WebServer",
"PublicDnsName" ]} ]] },
      "Description" : "URL for CloudFormer"
    }
  }
}
```

Lea AWS CloudFormation en línea: https://riptutorial.com/es/amazon-web-services/topic/6988/aws-cloudformation

Capítulo 5: AWS Lambda

Introducción

AWS Lambda es un servicio que le permite ejecutar código de back-end sin la necesidad de aprovisionar o administrar servidores. AWS Lambda se encarga de la ampliación y alta disponibilidad. El costo depende directamente de la frecuencia y la duración del código.

Encontrará ejemplos de cómo crear e implementar funciones de AWS Lambda en diferentes idiomas.

Observaciones

- El código de AWS Lambda se debe escribir de una manera sin estado. Si bien la instancia de un lambda se puede conservar y reutilizar, nunca debe esperar esto.

Examples

Proyecto Basic Gradle Java

Para implementar el código Java en AWS Lambda, debe crear un archivo zip de distribución que contenga todas las dependencias que se necesitan durante el tiempo de ejecución. Proyecto de ejemplo en Gradle:

```
apply plugin: 'java'

repositories {
    mavenCentral()
}

dependencies {
    compile 'com.amazonaws:aws-lambda-java-core:1.1.0'
}

task buildZip(type: Zip) {
    from compileJava
    from processResources
    into('lib') {
        from configurations.runtime
    }
}

build.dependsOn buildZip
```

La ejecución de `gradle build` creará un archivo zip con todas las dependencias incluidas con su código, listas para su implementación.

Código Lambda básico en Java

Un lambda necesita un controlador, que servirá como punto de entrada a su aplicación. Cada manejador necesita implementar la interfaz `RequestHandler<I, O>` donde `I` es el tipo de entrada y `O` es el tipo de salida. El tipo de entrada se pasa al método `handleRequest()` y el método devuelve el tipo de salida.

Un ejemplo simple podría verse así:

```
package com.example.lambda;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;

public class HelloNumber implements RequestHandler<Integer, String> {

    @Override
    public String handleRequest(Integer input, Context context) {
        return "You sent " + input;
    }
}
```

Código Lambda básico en JavaScript (NodeJs)

Un lambda necesita un controlador, que servirá como punto de entrada a su aplicación. En el caso más simple, obtienes tu entrada del `context` y pasas el resultado usando la función de `callback()` :

exports.handler = (evento, contexto, devolución de llamada) => {callback (nulo, 'Enviaste' + event.number); };

Creación de AWS Lambda mediante la GUI web (Java)

Para crear una Java Lambda usando la GUI, primero asegúrese de tener listo su zip de distribución. También necesitará una cuenta de AWS que pueda crear lambdas.

- Cambie a AWS Lambda, descarte la pantalla de introducción (o lea la documentación de Inicio) y presione el botón `Create a Lambda Function` .
- Seleccione un plano apropiado. Un modelo es una plantilla de ejemplo que puede ayudarlo a crear su lambda. Por ahora, comience con una `Blank Function` .
- Por ahora, no te preocupes por los disparadores y salta esa pantalla.
- Nombra tu función y sube el archivo zip.
- En la misma pantalla de configuración, escriba el nombre de su Controlador como clase Java completa con paquete, opcionalmente con el método que manejará las solicitudes. Si ha usado la interfaz Java del ejemplo del Código Lambda Básico, el nombre de la clase es suficiente.
- En la misma pantalla, seleccione `Create a new role from template(s)` y nombre su rol. La función se utiliza para otorgarle a su lambda acceso a otros recursos de AWS. En este caso, déjelo vacío ya que no necesitamos ningún otro recurso.
- Seleccione la memoria más baja disponible (128MB). El costo de las afluencias de memoria también. Guarda tu función.

Probando el código Lambda básico

Si ha implementado con éxito Lambda, también puede probarlo directamente en la GUI. Cuando hace clic en el botón de `Test` azul por primera vez, le presenta una creación de un nuevo evento de prueba.

Si está probando el código Java del código *Lambda básico en el ejemplo de Java* , elimine todo el cuerpo y simplemente agregue un número `1` .

Si está probando el código Javascript del ejemplo del *código Lambda básico en Javascript (Nodo)* , cree un json que tenga este aspecto:

```
{
    "number": "1"
}
```

Ejecutar la prueba. Deberías ver el resultado:

"Enviaste 1"

Agregar el activador de la puerta de enlace de AWS API

Probablemente la forma más común de invocar un Lambda es usar la puerta de enlace API, que asigna una llamada REST al código lambda. Puede agregar múltiples disparadores a su Lambda en cualquier momento, incluso al crear un nuevo lambda.

Si está familiarizado con la API Gateway, puede asociar cualquier método con un lambda implementado. Simplemente cree un nuevo método con el tipo de integración establecido en la función Lambda y apunte a su lambda. Puede asignar las entradas REST a las entradas lambda y las salidas a las salidas REST.

TODO: Decida si desea describir la puerta de enlace de la API aquí o hacer referencia a otra parte de la documentación.

Lea AWS Lambda en línea: https://riptutorial.com/es/amazon-web-services/topic/8918/aws-lambda

Capítulo 6: Clase raíz

Examples

Amazon clase de raíz de API es la siguiente.

```
public class AmazonRootobject
{
    public Itemsearchresponse ItemSearchResponse { get; set; }
}

public class Itemsearchresponse
{
    public string xmlns { get; set; }
    public Operationrequest OperationRequest { get; set; }
    public Items Items { get; set; }
}

public class Operationrequest
{
    public Httpheaders HTTPHeaders { get; set; }
    public string RequestId { get; set; }
    public Arguments Arguments { get; set; }
    public string RequestProcessingTime { get; set; }
}

public class Httpheaders
{
    public Header Header { get; set; }
}

public class Header
{
    public string Name { get; set; }
    public string Value { get; set; }
}

public class Arguments
{
    public Argument[] Argument { get; set; }
}

public class Argument
{
    public string Name { get; set; }
    public object Value { get; set; }
}

public class Items
{
    public Request Request { get; set; }
    public string TotalResults { get; set; }
    public string TotalPages { get; set; }
    public string MoreSearchResultsUrl { get; set; }
    public Item[] Item { get; set; }
}
```

```
public class Request
{
    public string IsValid { get; set; }
    public Itemsearchrequest ItemSearchRequest { get; set; }
}

public class Itemsearchrequest
{
    public string Keywords { get; set; }
    public string[] ResponseGroup { get; set; }
    public string SearchIndex { get; set; }
    public string Sort { get; set; }
}

public class Item
{
    public string ASIN { get; set; }
    public string ParentASIN { get; set; }
    public string DetailPageURL { get; set; }
    public Itemlinks ItemLinks { get; set; }
    public Smallimage SmallImage { get; set; }
    public Mediumimage MediumImage { get; set; }
    public Largeimage LargeImage { get; set; }
    public Imagesets ImageSets { get; set; }
    public Itemattributes ItemAttributes { get; set; }
    public OfferSummary OfferSummary { get; set; }
    public Offers Offers { get; set; }

    public Variationsummary VariationSummary { get; set; }
}

public class Variationsummary
{
    public Highestprice HighestPrice { get; set; }
    public Lowestprice LowestPrice { get; set; }
    public Highestsaleprice HighestSalePrice { get; set; }
    public Lowestsaleprice LowestSalePrice { get; set; }
}

public class Highestprice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Lowestprice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Highestsaleprice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Lowestsaleprice
```

```
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Itemlinks
{
    public Itemlink[] ItemLink { get; set; }
}

public class Itemlink
{
    public string Description { get; set; }
    public string URL { get; set; }
}

public class Smallimage
{
    public string URL { get; set; }
    public Height Height { get; set; }
    public Width Width { get; set; }
}

public class Height
{
    public string Units { get; set; }
    public string text { get; set; }
}

public class Width
{
    public string Units { get; set; }
    public string text { get; set; }
}

public class Mediumimage
{
    public string URL { get; set; }
    public Height1 Height { get; set; }
    public Width1 Width { get; set; }
}

public class Height1
{
    public string Units { get; set; }
    public string text { get; set; }
}

public class Width1
{
    public string Units { get; set; }
    public string text { get; set; }
}

public class Largeimage
{
    public string URL { get; set; }
    public Height2 Height { get; set; }
    public Width2 Width { get; set; }
}
```

```
public class Height2
{
    public string Units { get; set; }
    public string text { get; set; }
}

public class Width2
{
    public string Units { get; set; }
    public string text { get; set; }
}

public class Imagesets
{
    public object ImageSet { get; set; }
}

public class Itemattributes
{
    public string Binding { get; set; }
    public string Brand { get; set; }
    public string Color { get; set; }
    public string Model { get; set; }
    public string Manufacturer { get; set; }
    public string ProductGroup { get; set; }
    public string Title { get; set; }
    public ListPrice ListPrice { get; set; }
}

public class ListPrice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class OfferSummary
{
    public Lowestnewprice LowestNewPrice { get; set; }
    public Lowestusedprice LowestUsedPrice { get; set; }
    public string TotalNew { get; set; }
    public string TotalUsed { get; set; }
    public string TotalCollectible { get; set; }
    public string TotalRefurbished { get; set; }
    public Lowestrefurbishedprice LowestRefurbishedPrice { get; set; }
}

public class Lowestnewprice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Lowestusedprice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}
```

```
public class Lowestrefurbishedprice
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Offers
{
    public string TotalOffers { get; set; }
    public string TotalOfferPages { get; set; }
    public string MoreOffersUrl { get; set; }
    public Offer Offer { get; set; }
}

public class Offer
{
    public Merchant Merchant { get; set; }
    public Offerattributes OfferAttributes { get; set; }
    public Offerlisting OfferListing { get; set; }
}

public class Merchant
{
    public string Name { get; set; }
}

public class Offerattributes
{
    public string Condition { get; set; }
}

public class Offerlisting
{
    public string OfferListingId { get; set; }
    public string PricePerUnit { get; set; }
    public Price Price { get; set; }
    public string Availability { get; set; }
    public Availabilityattributes AvailabilityAttributes { get; set; }
    public string IsEligibleForSuperSaverShipping { get; set; }
    public string IsEligibleForPrime { get; set; }
    public Saleprice SalePrice { get; set; }
    public Amountsaved AmountSaved { get; set; }
    public string PercentageSaved { get; set; }
}

public class Price
{
    public string Amount { get; set; }
    public string CurrencyCode { get; set; }
    public string FormattedPrice { get; set; }
}

public class Availabilityattributes
{
    public string AvailabilityType { get; set; }
    public string MinimumHours { get; set; }
    public string MaximumHours { get; set; }
}
```

```
    public class Saleprice
    {
        public string Amount { get; set; }
        public string CurrencyCode { get; set; }
        public string FormattedPrice { get; set; }
    }

    public class Amountsaved
    {
        public string Amount { get; set; }
        public string CurrencyCode { get; set; }
        public string FormattedPrice { get; set; }
    }
```

Clase de negocios

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Reflection;
using System.Text;
using System.Threading.Tasks;
using System.Xml;
using System.Xml.Linq;
using ApplicationDataServices.SBEntityBox;

namespace ApplicationManagementLayer.Affiliate
{
    public class Amazon
    {
        private int ItemPage { get; set; }
        public int TotalNumberOfItem { get; set; }

public Amazon()
{
    ItemPage = 1;
    TotalNumberOfItem = 0;
}


string XMLURL = string.Empty;



 public async Task<AmazonRootobject> getProductsByKeywords(string q, Dictionary<string,
string> CategoryNames, int ItemPage, string categorynamebyid)
{
    try
    {
        AWSSignedRequestHelper helper = new AWSSignedRequestHelper("para1", "para2",
"webservices.amazon.in", "para3");
        IDictionary<string, string> r1 = new Dictionary<string, String>();
        r1["Service"] = "AWSECommerceService";

        r1["Operation"] = "ItemSearch";


        if (CategoryNames != null && CategoryNames.Any() && CategoryNames.Where(o =>
o.Key.Equals("AmazonCategoryName")).Any())
```

```
            r1["SearchIndex"] = CategoryNames.Where(o =>
o.Key.Equals("AmazonCategoryName")).First().Value;
        else
            r1["SearchIndex"] = "All";

        if (!r1["SearchIndex"].Equals("All") && CategoryNames != null && CategoryNames.Any()
&& CategoryNames.Where(o => o.Key.Equals("AmazonReferenceCategoryId")).Any())
            r1["BrowseNode"] = CategoryNames.Where(o =>
o.Key.Equals("AmazonReferenceCategoryId")).First().Value;

        if (!string.IsNullOrEmpty(q))
            r1["Keywords"] = q;
        else if (!string.IsNullOrEmpty(categorynamebyid))
            r1["Keywords"] = categorynamebyid;
        else if (CategoryNames != null && CategoryNames.Any() && CategoryNames.Where(o =>
o.Key.Equals("AmazonCategoryName")).Any())
            r1["Keywords"] = CategoryNames.Where(o =>
o.Key.Equals("AmazonCategoryName")).First().Value;
        else
            return null;

        r1["ResponseGroup"] = "Images,ItemAttributes,OfferFull,Offers,Variations";
        r1["Version"] = "2013-08-01";
        r1["ItemPage"] = ItemPage.ToString();
        //r1["Sort"] = "salesrank";

        string strRequestUrl = helper.Sign(r1);

        string output = null;
        using (System.Net.Http.HttpClient wc = new System.Net.Http.HttpClient())
        {
            var request = new System.Net.Http.HttpRequestMessage()
            {
                RequestUri = new Uri(strRequestUrl),
                Method = System.Net.Http.HttpMethod.Get,
            };

            /*var task =*/
            await wc.SendAsync(request)
                .ContinueWith((taskwithmsg) =>
                {
                    var response = taskwithmsg.Result;

                    var jsonTask = response.Content.ReadAsStringAsync();
                    jsonTask.Wait();
                    output = jsonTask.Result;
                });
            //task.Wait();

        }

        XmlDocument doc = new XmlDocument();
        doc.LoadXml(output);
        string outputJson = XmlToJSON(doc);

        var pro = new
System.Web.Script.Serialization.JavaScriptSerializer().Deserialize<AmazonRootobject>(outputJson);

        TotalNumberOfItem = !string.IsNullOrEmpty(pro.ItemSearchResponse.Items.TotalResults) ?
Convert.ToInt32(pro.ItemSearchResponse.Items.TotalResults) : 0;
        return pro;
```

```
        //return "";
    }
    catch
    {
        return null;
    }
}

http://stackoverflow.com/documentation/amazon-web-services/drafts/87373#



    }
}
```

Lea Clase raíz en línea: https://riptutorial.com/es/amazon-web-services/topic/7357/clase-raiz

Capítulo 7: Habichuela elástica

Observaciones

Limitaciones actuales (a partir del 2016-10-03)

- Las etiquetas de entorno no se pueden cambiar una vez que se crea el *entorno* , así que elija con cuidado.
- La autoescala en Elastic Beanstalk está limitada a *Simple* y *Programada* , por lo tanto, si desea usar *el Escalado por pasos* , reconsidere si Elastic Beanstalk es una buena opción.

Automatización con Jenkins

Hay un excelente complemento de implementación de AWSEB para Jenkins que se conectará y ejecutará para su implementación en Elastic Beanstalk (las implementaciones en azul / verde con terminación automática de inactividad están a solo una casilla de verificación).

Examples

Introducción a Elastic Beanstalk

Elastic Beanstalk (EB) es esencialmente un híbrido entre Golden AMIs y CloudFormation , mientras que simplifica enormemente la curva de aprendizaje de Puppet o Chef .

Un despliegue de Elastic Beanstalk se divide en dos componentes: Aplicación y Ambiente.

Solicitud

Considera esto como tu agrupación de nivel superior, tu propia aplicación. Por ejemplo, una sola aplicación ("MyWebApp") puede tener varios entornos ("Producción" y "Puesta en escena").

Ambiente

Cada entorno constará de una implementación de arquitectura completa (Instancias EC2 , Elastic Load Balancer , Autoscaling Group y Cloudwatch Alarms). Toda la configuración del entorno se configura y mantiene automáticamente.

Despliegue de una aplicación

La implementación de su aplicación es tan simple como cargar un archivo zip que contiene su código. Cada archivo zip (llamado *Versión de la aplicación*) que carga se asocia a una *Aplicación* , por lo que puede cargarlo una vez e implementarlo en múltiples *entornos* .

Personalizando el entorno

De forma predeterminada, Elastic Beanstalk implementará AMI "en stock" mantenidas por Amazon. Para la mayoría de las aplicaciones, esto es suficiente, pero puede haber ajustes ambientales que desee hacer (por ejemplo, cambiar la zona horaria, agregar paquetes / dependencias que no estén presentes en el código, etc.).

Hay dos formas de personalizar el EBI de EB que se utiliza: ebextensions o un AMI personalizado.

ebextensions - Una carpeta, llamada literalmente '.ebextensions', que puede colocarse opcionalmente en la raíz de su *Versión* de la *Aplicación* (el zip que cargó que contiene su código). Dentro de la carpeta ebextensions, puede colocar los archivos YAML que definen cualquier script, dependencia, etc. personalizado que desee que se ejecute en el lado del servidor durante el proceso de implementación. Hay una serie de puntos de conexión disponibles, para obtener la información más reciente, consulte la documentación correspondiente: http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/ebextensions.html

---

Gotchas / Cosas a tener en cuenta

Casilla de verificación de VPC : al crear un entorno, la opción está discretamente disponible en cuanto a si el entorno debe crearse o colocarse dentro de una VPC. Si necesita que su aplicación se comunique con los recursos existentes que ha creado, COMPRUEBE ESTA CAJA. De lo contrario, Elastic Beanstalk creará un nuevo grupo de seguridad que estará aislado del resto de su VPC. *Si bien podrá ajustar manualmente la configuración del grupo de seguridad después de la creación, intentar esencialmente "agregarla" a una VPC causará una variedad de problemas más adelante.*

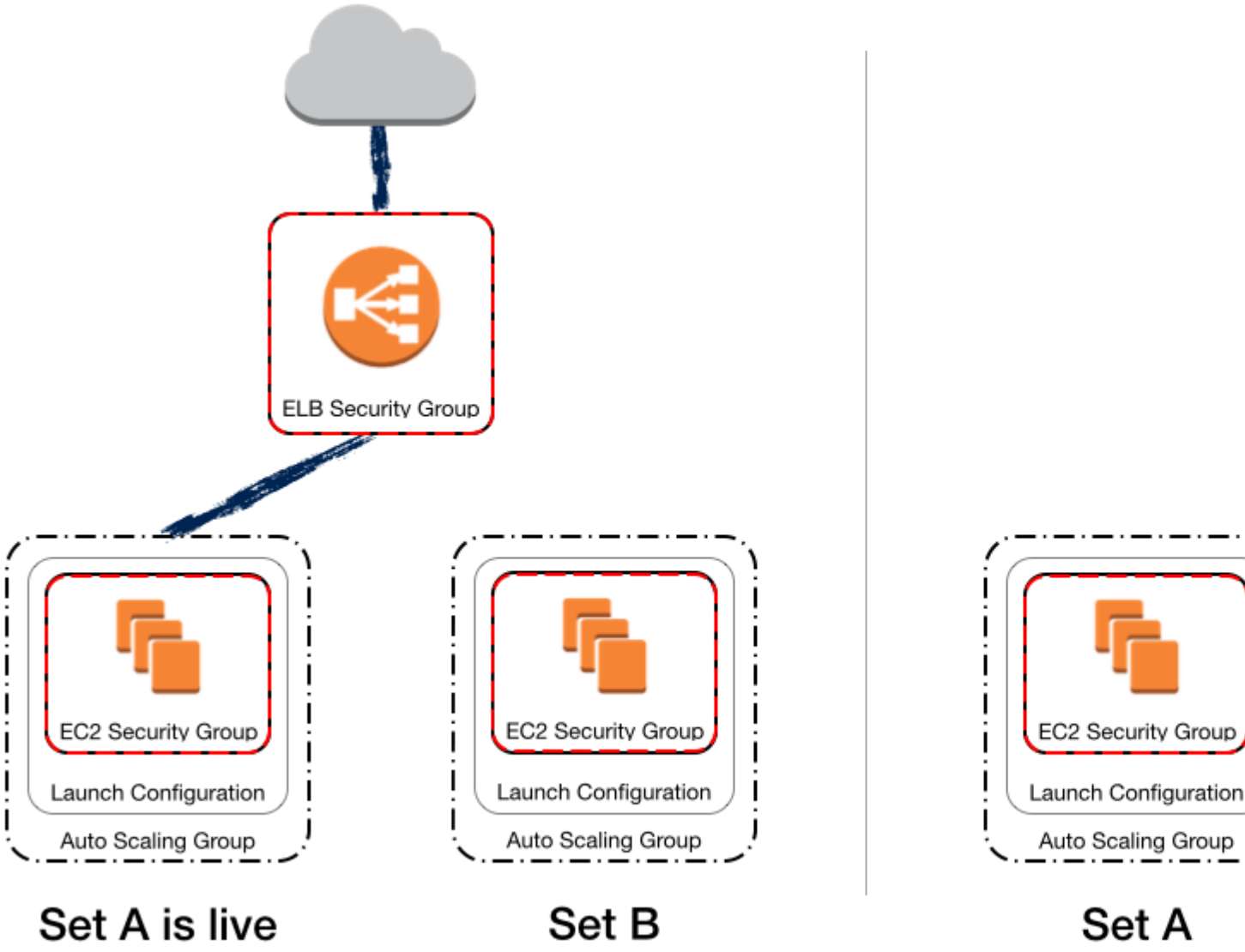
RDS : al crear un entorno, tiene la opción de crear una instancia de RDS como parte del entorno. No se recomienda usar esto, ya que cada vez que necesite "reconstruir" el entorno (por ejemplo, implementaciones en azul / verde, solución de problemas) destruirá y recreará la instancia de RDS (junto con todos los datos).

Despliegues azules / verdes en alubias elásticas

La implementación de Blue / Green es una técnica de lanzamiento que reduce el tiempo de inactividad y el riesgo al ejecutar dos entornos de producción idénticos (uno llamado "Blue", el otro llamado "Green"). En cualquier momento, solo uno de los entornos está sirviendo tráfico en vivo, mientras que el otro está inactivo.

Al implementar una nueva versión, el código se implementa en el entorno inactivo (p. Ej., Verde) y después de confirmar una implementación exitosa, el tráfico en vivo se cambia (p. Ej., Verde tiene un nuevo código, el tráfico de Azul ahora se enruta a Verde). El siguiente despliegue de código ocurrirá en el nuevo entorno inactivo (siguiendo el ejemplo, ahora sería Azul).

Ejemplo / Ayuda visual :



Fuente de la imagen: https://cloudnative.io/statics/blog/aws-blue-green-deployment.png

---

Al usar Elastic Beanstalk (EB), puede crear fácilmente múltiples entornos que son clones exactos entre sí para la implementación del código. Después de confirmar que el nuevo entorno está listo para funcionar, es tan simple como usar la función "Cambiar URL" en EB para intercambiar entornos.

Instrucciones paso a paso: http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.CNAMESwap.html

Lea Habichuela elástica en línea: https://riptutorial.com/es/amazon-web-services/topic/7207/habichuela-elastica

Capítulo 8: Implementar una imagen de contenedor docker utilizando ECS

Observaciones

Antes de que pueda agregar instancias de ECS a un clúster, primero debe ir a la Consola de administración de EC2 y crear instancias `ecs-optimized` con una función de IAM que tenga `AmazonEC2ContainerServiceforEC2Role` política `AmazonEC2ContainerServiceforEC2Role` .

1. Vaya a su Panel de EC2 y haga clic en el botón `Launch Instance` .
2. En `Community AMIs` , busque `ecs-optimized` y seleccione la que mejor se adapte a las necesidades de su proyecto. Cualquiera funcionará. Haga clic en Siguiente.
3. Cuando llegue a `Configure Instance Details` , haga clic en el `create new IAM role link` y cree un nuevo rol llamado `ecsInstanceRole` .
4. Adjunte la política `AmazonEC2ContainerServiceforEC2Role` a ese rol.
5. De forma predeterminada, su instancia de contenedor se inicia en su clúster `default` . Si desea iniciar su propio clúster en lugar del predeterminado, elija la lista de `Advanced Details` y pegue la siguiente secuencia de comandos en el campo de `User data` del `User data` , reemplazando `your_cluster_name` con el nombre de su clúster.

```
#!/bin/bash
echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
```

6. Luego, termina de configurar tu instancia de ECS.

NOTA: Si está creando un servidor web, querrá crear un grupo de `securityGroup` para permitir el acceso al puerto 80.

7. Cree un repositorio: `aws ecr create-repository --repository-name example-repository`
8. Autentique su cliente Docker en su registro: `aws ecr get-login --region us-east-1 | sh`
9. Construye tu imagen Docker: `docker build -t example-image .`
10. Etiquete su imagen para que pueda empujarla a este repositorio: `docker tag example-image:latest example-namespace/example-image:latest`
11. Inserte esta imagen en su repositorio de AWS recién creado: `docker push example-namespace/example-image:latest`
12. Registre una definición de tarea de ECS: `aws ecs register-task-definition --cli-input-json example-task.json`
13. Ejecute la tarea: `aws ecs run-task --task-definition example-task`

Examples

example-task.json

```
{
  "family": "example-task",
```

```
  "containerDefinitions": [
    {
        "environment": [],
        "name": "example-container",
        "image": "example-namespace/example-image:latest",
        "cpu": 10,
        "memory": 500,
        "portMappings": [
            {
                "containerPort": 8080,
                "hostPort": 80
            }
        ],
        "entryPoint": [],
        "essential": true
    }
  ]
}
```

Implemente una aplicación de muestra en el servicio AWS ECS como prueba de concepto

Siga los siguientes pasos para probar una aplicación de muestra en el servicio AWS ECS como prueba de concepto.

1. Inicie sesión en la consola de administración de AWS y vaya al **catálogo de servicios de AWS -> Compute -> Ec2**
2. Cree una VM (instancia de EC2) utilizando el sistema operativo de 64 bits de Amazon, esto lo usaremos para configurar la ventana acoplable, git, la herramienta de agente de AWS ECS y otras herramientas. También utilizaremos la misma máquina virtual como nodo en el clúster de ECS para implementar aplicaciones basadas en contenedores. Siga los pasos a continuación para crear una máquina virtual.
   a) Siga los pasos habituales para crear una instancia de EC2, aplique especial énfasis en los pasos posteriores durante la creación de la instancia de EC2.
   b) Seleccione un rol de IAM con menos permisos siguientes:
   AmazonEC2ContainerServiceforEC2Role
   c) Asegúrate de que java está instalado en la máquina virtual

```
[ec2-user@ip-172-31-4-129 ~]$ java -version
java version "1.7.0_111"
OpenJDK Runtime Environment (amzn-2.6.7.2.68.amzn1-x86_64 u111-b01)
OpenJDK 64-Bit Server VM (build 24.111-b01, mixed mode)
[ec2-user@ip-172-31-4-129 ~]$
```

3. Instalar la ventana acoplable [ejecutar los comandos siguientes]
   primero actualiza el repositorio de paquetes yum

```
sudo yum update -y
```

ahora para instalar la ventana acoplable ejecutar yum instalar

```
sudo yum install -y docker
```

4. Iniciar servicio docker

```
sudo service docker start
```

5. Agregue el usuario ec2 al grupo docker para que pueda ejecutar los comandos de Docker sin usar sudo.

```
sudo usermod -a -G docker ec2-user
```

6. Cierre sesión en EC2 y vuelva a iniciar sesión para obtener los nuevos permisos de grupo de docker.
7. Verifique que el usuario ec2 pueda ejecutar comandos de Docker sin sudo.

```
docker info
```

8. Instalando Git

```
sudo yum install -y git
```

9. Clone la aplicación PHP de muestra en la instancia de Ec2 desde git. Usaremos esta aplicación para nuestro POC.

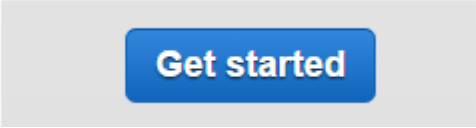
```
git clone https://github.com/awslabs/ecs-demo-php-simple-app

cd ecs-demo-php-simple-app
```

verifique que Dockerfile existe listando los contenidos del directorio

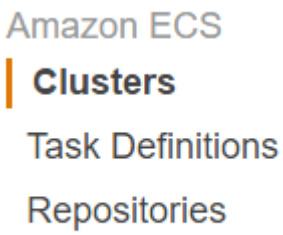
```
ls
```

10. Vaya al catálogo de servicios de AWS -> Compute -> Ec2 Container Service
11. Haga clic en Comenzar



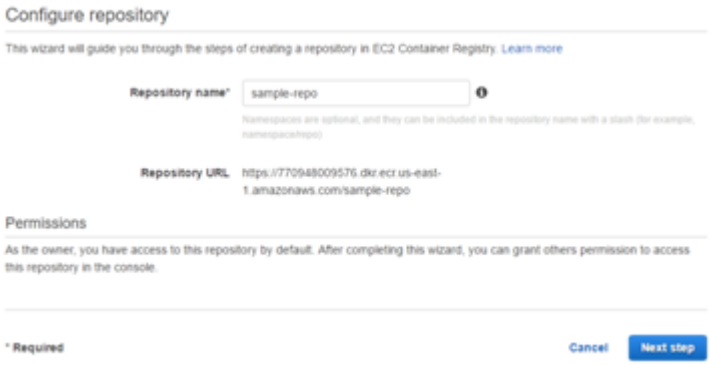
12. Haga clic en cancelar

13. Haga clic en repositorios desde el menú Repositorios a la izquierda



14. Haga clic en Comenzar

15. Ingrese el nombre del repositorio y haga clic en siguiente



16. Configurar herramientas ec2

```
aws configure
```

proporcione la ID de clave de acceso de AWS, la clave de acceso secreta, el nombre de región predeterminado según su cuenta

17. Construye, etiqueta y presiona la imagen de Docker
    a) Recupere el comando de inicio de sesión de la ventana acoplable que puede usar para autenticar su cliente Docker en su registro:

```
aws ecr get-login --region us-east-1
```

b) Ejecutar el comando retorno como salida del paso anterior

18. Construye la imagen de Docker desde tu Dockerfile. (Recuerda el Paso 9, donde descargaste una aplicación de docker de muestra)
    una)

```
docker build -t amazon-ecs-sample .
```

(Tenga en cuenta que el "." Significa directorio actual)
b) Ejecute las imágenes de la ventana acoplable para verificar que la imagen se haya creado

correctamente y que el nombre de la imagen contenga un repositorio donde pueda enviar sus cambios a la imagen de la ventana acoplable.

```
docker images
```

```
[ec2-user@ip-172-31-4-129 ecs-demo-php-simple-app]$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
amazon-ecs-sample   latest              9431571817d8        About a minute ago
 228.6 MB
```

c) Ejecutar la imagen recién construida. La opción -p 80:80 asigna el puerto 80 expuesto en el contenedor al puerto 80 en el sistema host (instancia Ec2 en este caso).

```
docker run -p 80:80 amazon-ecs-sample
```

Ignore la advertencia "apache2: no se pudo determinar de manera confiable el nombre de dominio completo del servidor, utilizando 172.17.0.2 para ServerName"

19. Intente acceder a la página web de la aplicación de muestra en el navegador, asegúrese de que el puerto 80 esté abierto en los grupos de seguridad asociados con la instancia

```
http://<ec2-instance-dns-address>
```

20. Presione la tecla ctrl + c, esto detendrá la imagen de la ventana acoplable. La aplicación de muestra no debe ser accesible.
21. Ahora, después de verificar con éxito nuestra aplicación de docker de muestra, intentaremos configurar un clúster para ejecutar la aplicación de muestra automáticamente. Además, para fines de demostración, intentaremos utilizar la instancia ec2 existente como un nodo en el clúster. Esto se puede lograr instalando un programa de agente en la instancia ec2.
22. Instalación de Amazon ECS Container Agent en la instancia ec2
    una)

```
sudo yum install -y ecs-init
```

b) Reinicie el demonio docker

```
sudo service docker restart
```

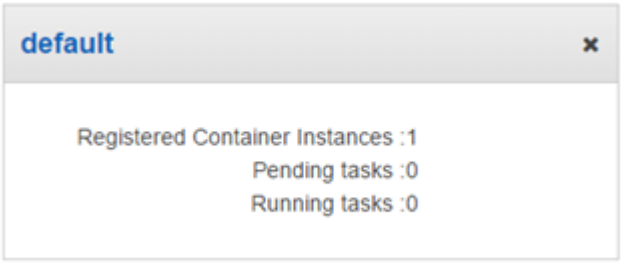
c) Iniciar el trabajo ecs-init upstart

```
sudo start ecs
```

d) (Opcional) Puede verificar que el agente se está ejecutando y ver cierta información sobre su nueva instancia de contenedor con la API de introspección del agente. Asegúrese de que el puerto 51678 esté abierto en el grupo de seguridad.

```
curl http://localhost:51678/v1/metadata
```

23. Vaya al catálogo de servicios de AWS -> Compute -> Ec2 Container Service -> Cluster y verifique que se haya creado un cluster predeterminado



24. Ahora continuamos con la creación de un grupo de tareas y agregando nuestra imagen de ventana acoplable como tarea para ejecutar en el clúster
    a) Examine el archivo simple-app-task-def.json en la carpeta ecs-demo-php-simple-app.
    b) Edite el archivo simple-app-task-def.json y modifique la memoria, para que pueda ejecutarse en una instancia elegible de nivel libre (supongo que una está utilizando la instancia ec2 elegible de nivel libre para este POC, de lo contrario no es necesario reducir la memoria límite)
    c) Actualizar **memoria = 250** en toda la aparición en el archivo simple-app-task-def.json
    d) Registre una definición de tarea con el archivo simple-app-task-def.json.

```
aws ecs register-task-definition --cli-input-json file://simple-app-task-def.json
```

e) Vaya a la definición de tarea en la página de servicio de contenedor ec2, encontrará la definición de tarea registrada

f) Utilice el siguiente comando de AWS CLI para ejecutar una tarea con la definición de tarea console-sample-app.

```
aws ecs run-task --task-definition console-sample-app
```

g) Abra la aplicación web de muestra en el navegador, debe ser accesible (consulte el paso 19)

Gracias por leer, comparta sus comentarios y consultas para una discusión de seguimiento.

Lea Implementar una imagen de contenedor docker utilizando ECS en línea: https://riptutorial.com/es/amazon-web-services/topic/7711/implementar-una-imagen-de-contenedor-docker-utilizando-ecs

Creditos

S. No	Capítulos	Contributors
1	Primeros pasos con amazon-web-services	Community, Cyril Duchon-Doris, Karan Shah, Lynn Langit
2	Amazon Cognito	Jeet
3	Amazon DynamoDB	swapnil kadu
4	AWS CloudFormation	Hardeep Singh, Naveen Vijay
5	AWS Lambda	jarmod, sm4
6	Clase raíz	vicky
7	Habichuela elástica	Lorenzo Aiello
8	Implementar una imagen de contenedor docker utilizando ECS	mrded, Satish