AWS Intermediate

Lab

Azat Mardan @azat_co

Node University

Overview

1. Cloud Automation
2. Connecting Resources and IAM
3. Extra Services and Best Practices

Pre-Reqs

- Python
- pip
- Node and npm
- AWS Account (requires email + credit/debit card)
- Docker deamon/engine

Cloud Automation

- AWS CLI
- SDKs
- CloudFormation
- Others: Ansible, Terraform

AWS CLI

```
pip install awscli
```

http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html

```
aws <command> <subcommand> [options and parameters]
```

Copy your key and secret and put into:

```
aws configure
```

```
aws help

aws ec2 help

aws ec2 describe-regions help
```

```
aws ec2 describe-instances help

aws ec2 run-instances help

awc ec2 create-images help
```

Launch Instance

```
$ aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-type t1.micro --key-name MyKeyPair --security-groups my-sg
```

With subnet:

```
$ aws ec2 run-instances --image-id ami-{xxxxxxxx} --count 1 --instance-type t1.micro --key-name {MyKeyPair} --security-group-ids sg-{xxxxxxxx} --subnet-id subnet-{xxxxxxxx}
```

```
aws ec2 create-tags --resources i-{xxxxxxxx} --tags Key={Name},Value={MyInstance}
```

Replace {xxx}, {Name} and {MyInstance}

```
aws ec2 start-instances --instance-ids i-5203422c
aws ec2 terminate-instances --instance-ids i-5203422c
```

```
$ aws ec2 create-key-pair --key-name {MyKeyPair} --query 'KeyMaterial' --output text > {MyKeyPair}.pem
$ aws ec2 describe-key-pairs --key-name {MyKeyPair}
aws ec2 delete-key-pair --key-name {MyKeyPair}
```

CodeDeploy

» https://aws.amazon.com/codedeploy/

Source Code

- Git
- Rsync
- S3, e.g., `aws s3 cp s3://{mybucket}/latest/install . --region us-east-1`

Auto Startup

- `init.d` or CloudInit for Ubuntu+Debian and other like CentOS with additional installation
- User Data
- Command

Shell Script and User Data Example

```
#!/bin/bash
yum update -y
yum install -y httpd24 php56 mysql55-server php56-mysqlnd
service httpd start
chkconfig httpd on
groupadd www
usermod -a -G www ec2-user
chown -R root:www /var/www
chmod 2775 /var/www
find /var/www -type d -exec chmod 2775 {} +
find /var/www -type f -exec chmod 0664 {} +
echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
```

More info on User Data:

http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/user-data.html

SDKs

Supported

- Amazon S3
- Amazon EC2
- DynamoDB
- Many more

Node SDK

```
mkdir aws-test

cd aws-test

npm init -y

npm i aws-sdk
```

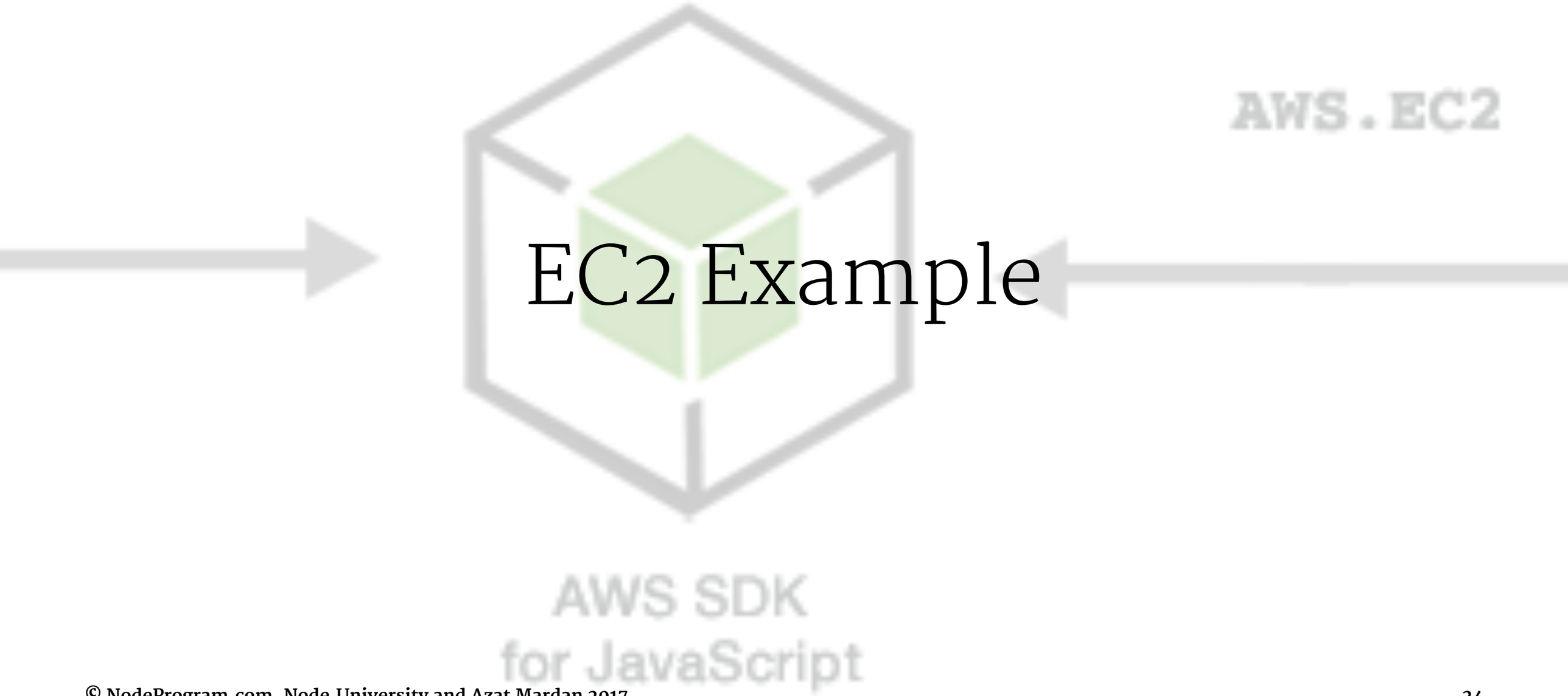
Credentials

- Home directory
- Environment variables
- JavaScript/Node or JSON file

Credentials in Home Directory

`~/.aws/credentials` or `C:\Users\USER_NAME\.aws\credentials` for Windows users

```
[default]
aws_access_key_id = YOUR_ACCESS_KEY_ID
aws_secret_access_key = YOUR_SECRET_ACCESS_KEY
```



EC2 Example

Create and open `create-ec2.js`:

```
// Load the SDK for JavaScript
var AWS = require('aws-sdk');

// Load credentials and set region from JSON file
AWS.config.loadFromPath('./config.json');
```

```
// Load the AWS SDK for Node.js
var AWS = require('aws-sdk');
// Load credentials and set region from JSON file
AWS.config.loadFromPath('./config.json');

// Create EC2 service object
var ec2 = new AWS.EC2({apiVersion: '2016-11-15'});

var params = {
   ImageId: 'ami-10fd7020', // amzn-ami-2011.09.1.x86_64-ebs
   InstanceType: 't1.micro',
   MinCount: 1,
   MaxCount: 1
};

// Create the instance
ec2.runInstances(params, function(err, data) {
   if (err) {
      console.log("Could not create instance", err);
      return;
   }
   var instanceId = data.Instances[0].InstanceId;
   console.log("Created instance", instanceId);
   // Add tags to the instance
   params = {Resources: [instanceId], Tags: [
      {
         Key: 'Name',
         Value: 'SDK Sample'
      }
   ]};
   ec2.createTags(params, function(err) {
      console.log("Tagging instance", err ? "failure" : "success");
   });
});
```

CloudFormation

Samples

OpsWork vs CloudFormation vs Elastic Beanstalk

OpsWork: configuration management (stacks and layers) - narrower app-oriented resources than CloudFormation

CloudFormation: building block service for almost everything

Elastic Beanstalk: only app management service

Lab

Goal: Use CloudFormation to create Autoscaling and load-balancing website in an Amazon VPC

Connecting Resources and IAM

Best IAM Practices

- Lock away your AWS account (root) access keys
- Create individual IAM users
- Use AWS-defined policies to assign permissions whenever possible
- Use groups to assign permissions to IAM users
- Grant least privilege

Best IAM Practices (Cont)

- Configure a strong password policy for your users
- Enable MFA for privileged users
- Use roles for applications that run on Amazon EC2 instances
- Delegate by using roles instead of by sharing credentials

Best IAM Practices (Cont)

- Rotate credentials regularly
- Remove unnecessary credentials
- Use policy conditions for extra security
- Monitor activity in your AWS account

Lab

Goal: Build automation with User Data and AWS CLI

- Create a shell script (script A) to install and run Node.js on Amazon Linux:
  - Install EPEL and Node (npm included) link
  - Install pm2: `npm i pm2 -g`
  - Download server code from GitHub using wget or curl (link
  - Start server with `sudo pm2 start app.js -i 0`
- Use script A in User Data for `run-instances`
- Create a shell script (B) to create a 2 new instance `run-instances` from Amazon Linux in region `us-west-2` with script

Lab 1 Cont.

1. Test by going to the browser
2. Create autoscaling group: CPU>10% +1 (aws cli)
3. Install loadtest: `npm i -g loadtest`
4. Use loadtest to stress test your instance to see if autoscaling kicks in.
5. Terminate the instance(s) with aws cli

Lab 2

Use Elastic Beanstalk to deploy a web app which uses RDS:

http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create_deploy_nodejs.html

Lab 3

Set up CloudFormation for the stack:

- ELB
- 2 EC2 with a web app which uses DynamoDB
- Security Group
- VPC
- DynamoDB
- SNS

Resources for Lab 3:

- http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/CHAP_TemplateQuickRef.html
- http://docs.aws.amazon.com/amazondynamodb/latest/gettingstartedguide/GettingStarted.NodeJs.html
- http://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/dynamodb-examples.html