**Configuring a Custom Domain with Amazon Cognito**

Introduction

Amazon Cognito lets you add user sign-up, sign-in, and access control to applications quickly and easily. With Cognito, you can provision a hosted authentication UI that you can add to your application to handle sign-up and sign-in workflows. Cognito's hosted UI is the foundation for other features, such as the ability to sign in directly to your user pool through Facebook, Amazon, Google, and Apple, as well as through OpenID Connect and SAML identity providers.

You can also use your own custom domain for Cognito's hosted UI. This offers your users a unified login experience, allowing them to stay within your application's domain, and it enforces a consistent branding and security experience.

In this hands-on lab, you will configure a number of AWS services, such as Cognito, ACM, S3, and Route 53, in order to configure a custom domain for use with Cognito's hosted UI.

So let's take a look at the diagram and our sample application to understand our scenario and goals for the lab a little better.

Lab Prerequisites

* Understand how to log into and use the **AWS Management Console**
* Understand how to use the **AWS Command Line Interface (CLI)**
* Understand how to use ssh from the command line
* Understand how DNS, HTTP, and HTTPS work
* Familiarity with Amazon S3 and Route 53

Additional Information and Resources

Make sure you are operating the lab within the N. Virginia (us-east-1) region.

This GitHub repository is used in the lab: <https://github.com/linuxacademy/content-aws-sam>

Logging In

Use the credentials provided on the hands-on lab page, and log into the AWS console as cloud\_user.

Objective 1 - Create a Cognito User Pool

* Navigate to Amazon Cognito in the AWS console.
* Click **Manage User Pools** at the first screen.
* Click the link that says **Click here to create a user pool**.
* Give it a *Pool name* of **CognitoLab**.
* Click **Review defaults**.
* Click **Create pool**:
  + Make note of the Pool Id (copy it and paste it into a text file somewhere for later use).
* Scroll down to the *App clients* section, and click **Add app client**.
* On the next screen, click **Add an app client**:
  + Give it an *App client name* of **CognitoLabApp**.
  + **Uncheck** *Generate client secret*:
    - User pool apps with a client secret are not supported by the JavaScript SDK
  + Click the **Create app client** button at the bottom of the screen:
    - Just like we did with the Pool Id, grab the App client id and paste it into the same text file for later use.

Objective 2 - Create an ACM Certificate

* Navigate to Certificate Manager in the AWS console.
* Click **Get started** on the *Provision certificates* side of the next screen.
* On the following screen, click **Request a certificate**:
  + For this next process, we need the domain name of our lab environment. We can get it in Route 53:
    - Navigate to Route 53 in the AWS console (in another browser tab).
    - Go to *Hosted Zones* to see the domain name.
    - Copy it, then get back to the *Request a certificate* screen we were in.
* In the domain text box (that currently says [*www.example.com*](http://www.example.com/)), type \*. (that's asterisk, then dot) and then paste in the domain name we just copied from Route 53. Make sure to delete any trailing periods.
* Click **Next**.
* On the next screen, make sure **DNS validation** is selected, and click **Next**.
* Leave tags alone on the next screen, and click **Review**.
* On the *Review* page, click **Confirm and request**.
* In the *Validation* screen, expand our domain (by clicking the arrow next to it), and click **Create record in Route 53**:
  + There will be a popup window, and we need to click **Create** in it.
* Click **Continue**.
* Refresh the browser page until the *Status* goes from **Pending validation** to **Issued**. It will take a few minutes.

Objective 3 - Configure a Custom Domain for Cognito

Preliminary Work

Before we can go into Cognito and configure a custom domain, we've got a bit of work to do in Route 53. Navigate there in the AWS console.

* Click **Create Record Set**.
* In the form that shows up on the right:
  + Leave *Name* blank.
  + Make sure *Type* is set to **A - IPv4 address**.
  + Select **Yes** for *Alias*:
    - In the dropdown that appears, select our S3 bucket from the list.
* Click **Create**.

Back in Cognito

* Navigate to our user pool.
* In the left-hand menu, under *App integration*, click **Domain name**.
* In the *Your own domain* section, click **Use your domain**:
  + For a *Domain name*, use auth.<labdomain>, where <labdomain> is what was assigned to the lab (and what we found in Route 53 a few steps ago).
  + Select our ACM certificate from the *AWS managed certificate* dropdown:
    - Note that it may take a few minutes for this to show up in the list. Keep refreshing the browser every so often until it does.
* Click **Save changes**.
* Wait for the *Domain status* to go from **CREATING** to **ACTIVE**, which may take about 15 minutes.
* Note the *Alias target* (copy it and paste it into that text file we've been using for other things we're "noting").

Objective 4 - Complete App Client Configuration and Create CloudFront Distribution

More Preliminary Work

* In Route 53, we're going to create another A record:
  + *Name:* **auth**
  + *Type:* **A - IPv4 address**
  + *Alias:* **Yes**:
    - *Alias Target:* Paste in the *Alias target* we copied in the last step.
  + Click **Create**.

Back in Cognito

* Navigate to **App client settings** (in the left-hand menu, under the *App integration* section):
  + Check the box next to *Enabled Identity Providers*:
    - Check the *Cognito User Pool* box too.
  + Provide a *Callback URL* (https://www.<labdomain>).
  + Provide a *Sign out URL* (https://www.<labdomain>).
  + Check the **Authorization code grant** box.
  + Below that, in the *Allowed OAuth Scopes* section, check **email**, **openid**, and **profile**.
  + Click **Save changes**.

Creating a CloudFront Distribution

* Navigate to CloudFront in the AWS console.
* Click **Create Distribution**.
* We're going to use the *Web* delivery method, so click **Get Started** in that section.
* Fill out the following form with these values:
  + *Origin Domain Name:* This will be www.<labdomain>. followed by s3.amazonaws.com... (the bucket).
  + Farther down the form, set *Viewer Protocol Policy* to **Redirect HTTP to HTTPS**.
  + Scroll even farther down, to the *Distribution Settings* section:
    - In the *Alternate domain names* box, enter:
      * www.<labdomain>
    - Select **Custom SSL Certificate** in the list below that:
      * Choose the wildcard certificate, \*.<labdomain>, from the dropdown.
    - Farther still down the page, set the *Default Root Object* to **index.html** (because S3's index.html doesn't work behind CloudFront).
  + Click **Create Distribution**.
* Wait for the *Status* of **In Progress** to go to **Deployed**.
* Click on the distribution ID:
  + Note distribution domain name (e.g. **d3XXXXXXXXXXX.cloudfront.net**).
  + Copy that into the text file we've been using to hold values, because we'll need it in a bit.
* Go back to Route 53:
  + Click **Create Record Set** and set the following:
    - *Name:* **www**
    - *Type:* **A - IPv4 address**
    - *Alias:* **Yes**:
      * *Alias Target:* Paste in the *Alias target* (the distribution domain name) we copied in the last step.
    - Click **Create**.

Objective 5 - Configure and Deploy the Application

On the lab instructions page, there were login credentials for an EC2 instance. Use them now to log into it, with SSH.

Once we're in, we've got to clone the GitHub repository mentioned way back in the beginning of the lab:

git clone https://github.com/linuxacademy/content-aws-sam

Then get into that directory:

cd content-aws-sam/labs/Configuring-Custom-Domain-Cognito

We need to install a few dependencies:

npm install

Now get into the app directory:

cd app

We have to edit a configuration file and set some values:

vim main.js

We need to specify our Cognito configuration. Down in the Amplify.configure section:

* Set the userPoolId to the user pool ID we have stowed away in our text file.
* Set the userPoolWebClientId to what we saved out when we performed the **Add app client** step.
* A couple of lines down, the domain: needs to be auth.<labdomain>, where <labdomain is the one we've been using in other steps.
* redirectSignIn and redirectSignOut are both going to be 'https://www.<labdomain>'
  + Save and quit the file, then move up a directory:

cd ..

Now we can compile the application:

npm run build

Next we'll get into our dist directory:

cd dist

Finally, let's deploy the application:

aws s3 sync . s3://www.<labdomain>

Testing

In a web browser, navigate to https://www.<labdomain>. Click **Sign in with Cognito**, and we should land at a Cognito login page.

Here, we've got to sign up for a new account. Enter a username, email, and password into each respective field, and click **Sign up**.

Check the email account you entered, grab the code that was sent, and plug it into the *Verification Code* pop up that's waiting in our browser. Click **Confirm Account**, and we should be logged in.

Conclusion

We created a user pool in Cognito, and an app client for our Cognito-hosted UI, at our custom subdomain. Then we created an AWS Certificate Manager certificate for both our www and auth subdomains (which entailed creating and configuring all the correct records in Route 53). We configured our Cognito app client for the OAuth 2.0 Authorization Code Grant Flow, and created a CloudFront distribution pointing to our S3-hosted origin. Finally, we configured and deployed our web application with our Cognito configuration. Congratulations! You made it through. Give yourself a pat on the back for completing this hands-on lab.