Tutorial: Automated WebAuthnKit deployment at AWS

Introduction

This tutorial describes the complete and automated deployment of the WebAuthn Starter Kit at the AWS backend and frontend. The automated deployment is based on shell scripts: a PowerShell script for Windows and a Bash script for MacOS/Linux.

The source code used to build the Yubico WebAuthn Starter Kit solution is available in this GitHub repository.

The complete deployment of the WebAuthn Starter Kit at AWS includes build, upload, deployment and configuration of the following AWS services:

- AWS backend:
    - AWS S3:
        - One S3 bucket for hosting the WebAuthn Starter Kit binaries and configuration files
    - AWS RDS database:
        - Aurora Serverless MySQL compatible database with tables used to store the user credential attributes
    - AWS Cognito:
        - One Amazon Cognito User Pool
    - AWS Lambda:
        - Four AWS Lambda Functions used as custom triggers with Cognito User Pool
        - One AWS Lambda Function (Java) as the WebAuthn Relying Party library
    - AWS API Gateway
        - One RESTful API regional endpoint for the WebAuthn Starter Kit
- AWS frontend:
    - AWS Amplify:
        - Configured with the React web app for the browser interaction with the WebAuthn functions
- AWS deployment and configuration:
    - AWS CloudFormation
        - Configured with the stack for the WebAuthn Starter Kit

The WebAuthn Starter Kit backend utilizes the AWS Serverless Application Model (AWS SAM) open-source framework for building serverless applications on AWS. Hence, the WebAuthn Starter Kit will be using the SAM template specification that defines this application's APIs, database, Lambda functions, and event source mappings. This single SAM template file operates as a one-click deployment, repeatable extension of AWS CloudFormation.

Once you have deployed the WebAuthn Starter Kit backend at AWS via SAM, the provided React web client can be used to demonstrate the WebAuthn flows.

Pre-requisites

Client operating systems

The client workstation can be one of the following:

- Microsoft Windows 10
- Apple MacOS Catalina 10.15 or higher
- Linux Ubuntu v18.04 or higher

Software and tools

Make sure that the following software tools are configured at your local machine:

1. Install the AWS CLI v2
2. Install the AWS SAM CLI
3. Install Node.js
4. Install Npm
5. Install Git
6. Install Docker
7. Install the PowerShell AWS tools (if Microsoft PowerShell is used)

AWS accounts and credentials

Make sure to have the following AWS accounts and credentials configured:

1. Setup the AWS Credentials at your local machine
2. Configure the Amplify Command Line Interface (CLI) and the IAM credentials at your local machine by following the steps in this tutorial

Clone or download the GitHub repo

First, at the workstation, create a folder for the WebAuthnKit project.

It is possible to either clone the WebAuthnKit GitHub repo, or download the contents as a zip-file.

Clone the WebAuthn GitHub repo

Clone the WebAuthnKit GitHub repo to the local workstation as follows.

Step 1.1. At the workstation, start a PowerShell or Terminal prompt and navigate to the WebAuthnKit project folder.

Step 1.2. Clone the Yubico WebAuthn Starter Kit Repository to your local project folder by running the following command:

```
$ git clone https://github.com/Yubico/WebAuthnKit.git
```

Download the WebAuthn GitHub repo

It is also possible to download the WebAuthn Starter Kit as a zip-file from the Yubico WebAuthn Starter Kit Repository by pressing the “Code” button.

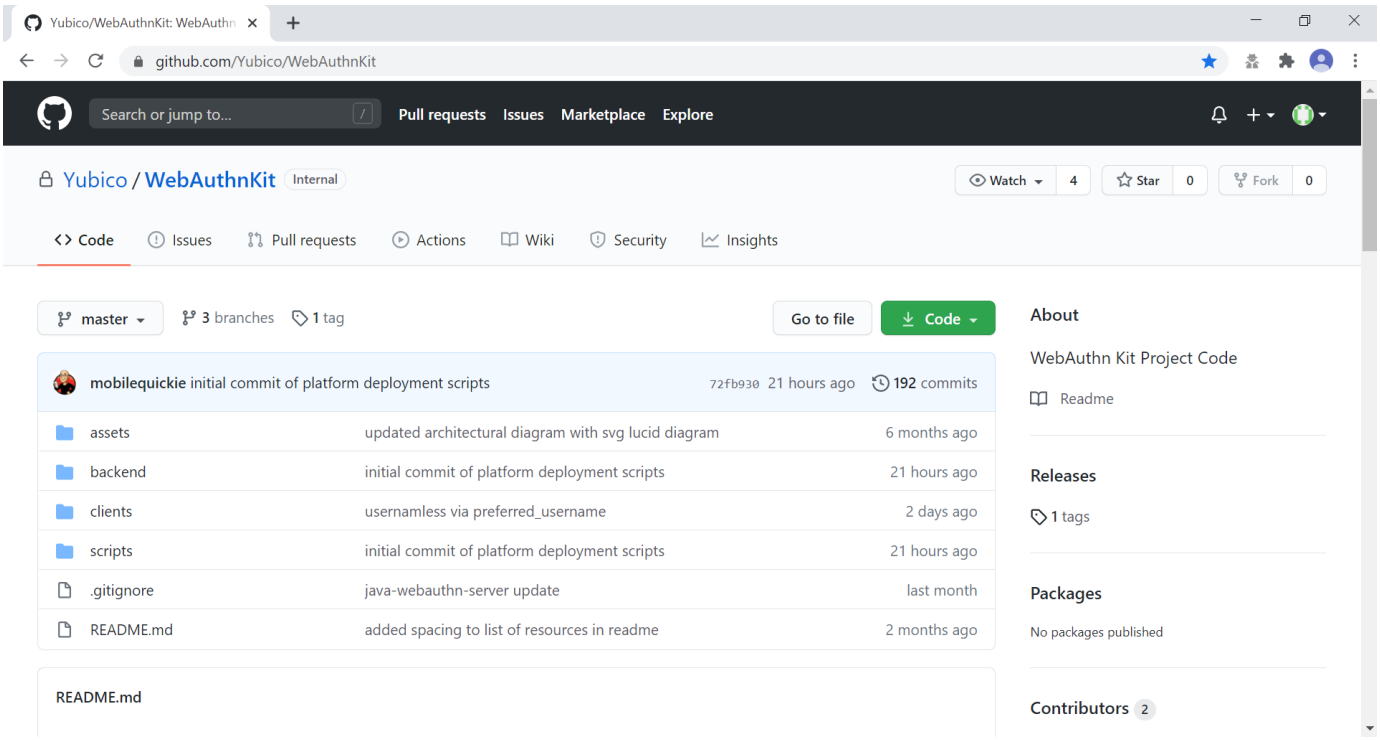

Figure 1 - *Downloading the WebAuthnKit zip-file from the GitHub repo*

Download the WebAuthnKit to the local project folder and extract the zip-file. The root of the local WebAuthnKit folder is denoted as ~\WebAuthnKit\.

Windows PowerShell script

This section explains how to use the Windows PowerShell script, which manages the complete build and deployment of WebAuthnKit at the AWS backend, the AWS frontend, and the additional steps.

Configuring Docker for Windows

Docker is used for building and deploying several WebAuthnKit components at AWS.

Configure Docker to use Linux containers

When Docker is used for building the WebAuthnKit, also on a Windows machine, it needs to be configured to use Linux containers. This is the default setting, but it is recommended to verify this. Right-click on the Docker Desktop icon in the tracemenu to ensure that Linux containers are used. (This can be checked by Docker displaying “Switch to Windows containers”, which means that Linux containers are active.)

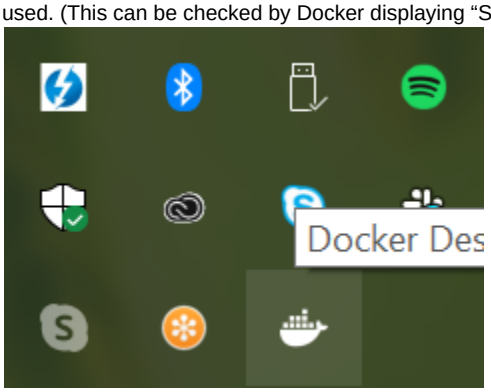

Figure 2 - *Docker Desktop tray icon*

Configure Docker’s file sharing permissions

In order to give the Docker container permissions to access the WebAuthnKit folders at the Windows machine, it is necessary to configure Docker as follows.

Step 2.1. Right-click on the Docker Desktop icon in the tracemenu and select Settings.

Step 2.2. In the Docker Settings GUI, select Resources and File Sharing.

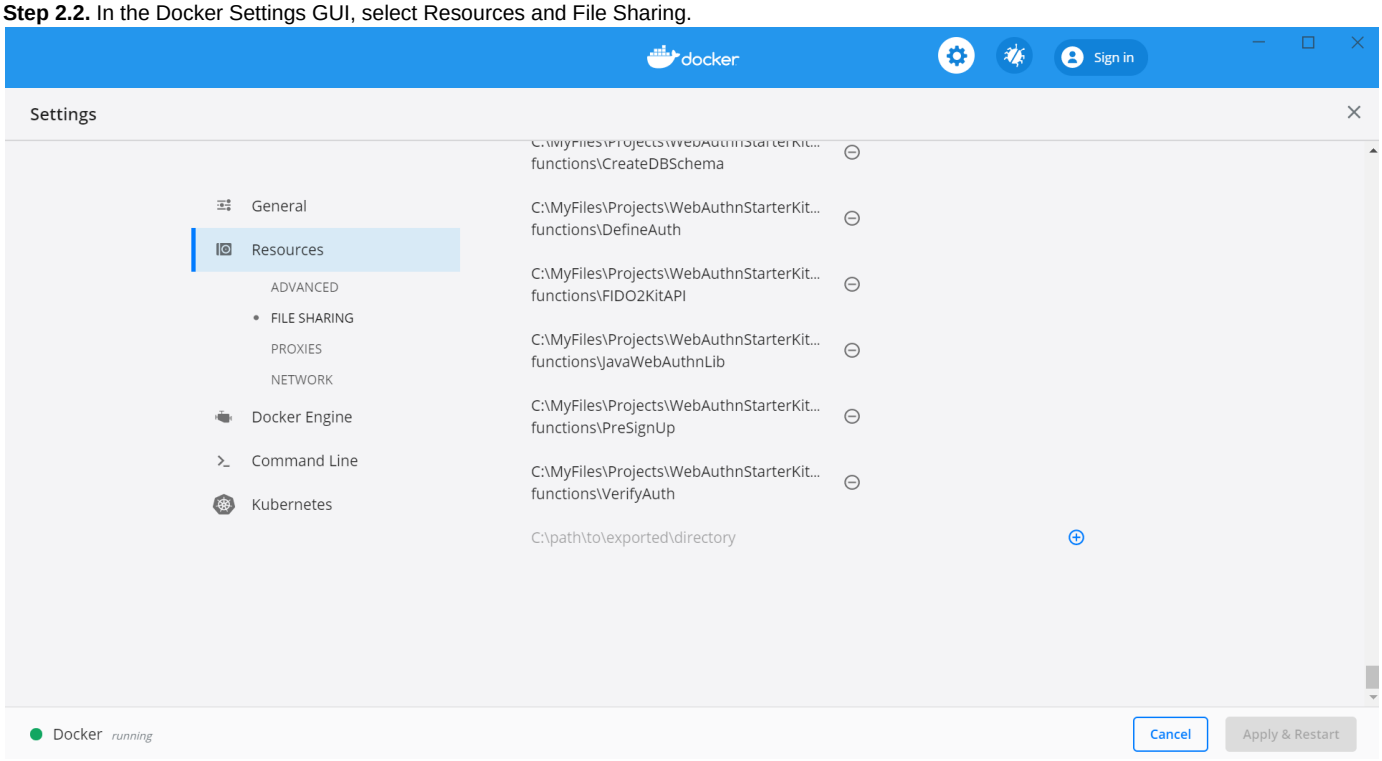

Figure 3 - *Configuring Docker Desktop file sharing*

Step 2.3. Press the “+” button in the Docker Desktop GUI, and the Select Folder GUI appears.

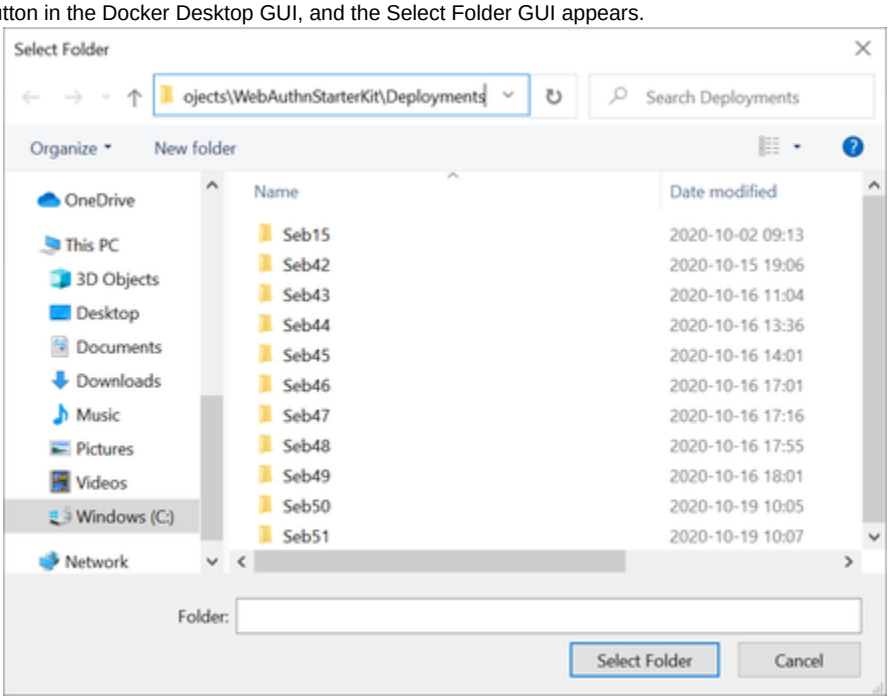

Figure 4 - *Adding folder for Docker Desktop file sharing*

Step 2.4. Select your folder for the WebAuthnKit project, or a folder higher up, to give Docker file sharing permissions to all files and subfolders under the selected folder.

Hint: If you are planning to do multiple WebAuthnKit deployments, it makes sense to give Docker file sharing permissions to the folder with all WebAuthnKit deployments.

Step 2.5. Finally, press the button “Apply & Restart” in the Docker Desktop.

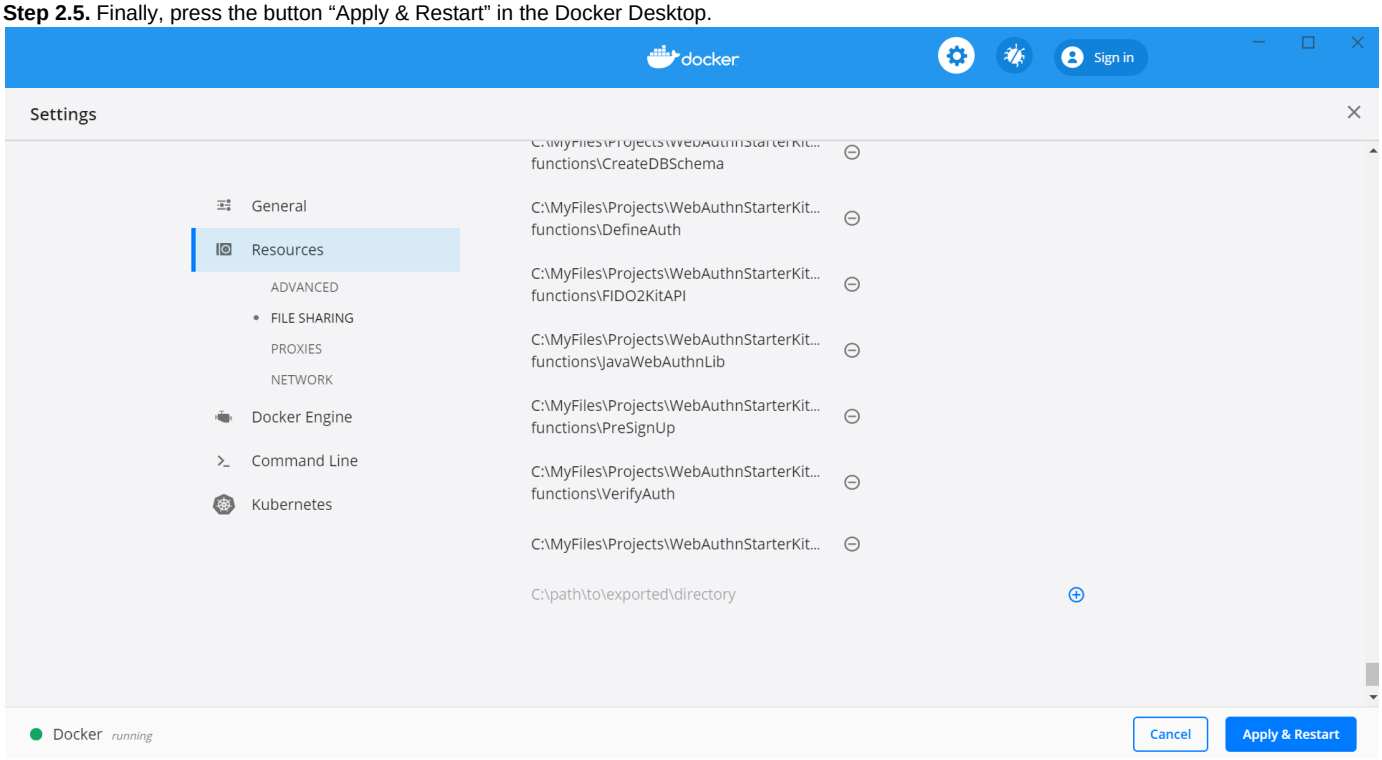

Figure 5 - *Finalizing Docker Desktop file sharing*

Warning: If the Docker Desktop is not configured for file sharing, then Docker will launch several pop-up GUIs with notifications to allow file sharing on demand for each deployment.

Editing the PowerShell script configuration file

The PowerShell script configuration file, ~\WebAuthnKit\scripts\PowerShell\deployStarterKitPs.json, should be reviewed and, if needed, edited.

```
{
    "AwsCliProfile": "",
    "AwsRegion":"eu-west-2",
    "S3BucketName":"",
    "CfStackName":"",
    "Suffix":"",
    "UserPoolName":"",
    "DatabaseName":"",
    "DatabaseMasterUsername":"",
    "DatabaseMasterPassword":"",
    "DefineAuthChallengeFuncName":"",
    "CreateAuthChallengeFuncName":"",
    "VerifyAuthChallengeFuncName":"",
    "WebAuthnKitApiName":"",
    "WebAuthnKitApiFuncName":"",
    "PreSignUpFuncName":"",
    "JavaWebAuthnLibFuncName":"",
    "CreateDatabaseSchemaFuncName":"",
```

```
    "CreateDatabaseSchemaCallerFuncName":"",
    "AmplifyHostingAppName":"",
    "AmplifyBranchName":""
}
```

In particular, the `AwsRegion` parameter must be set to a valid AWS region for your AWS deployment. If the `AwsRegion` parameter is not set, the PowerShell script will prompt the user to enter it.

If the `Suffix` parameter is not set, the PowerShell script will set this to a random six digit numeric value.

If the `DatabaseMasterPassword` parameter is not set, the PowerShell script will set this to a random sixteen character string.

All other parameters that are not specified in the configuration file will be set to default values, and the `Suffix` will be appended to these default values.

If a parameter is declared in the configuration file, it will be used by the PowerShell script exactly as declared, without appending any suffix.

Running the PowerShell script

The Windows PowerShell script for deploying the WebAuthn Starter Kit is published at the WebAuthnKit GitHub repo. The script should be previously cloned or downloaded to the workstation.

Step 3.1. In order to run the PowerShell script, open a PowerShell prompt and navigate to the folder ~\WebAuthnKit\scripts\PowerShell\.

Step 3.2. Execute the command `.\deployStarterKit.ps1`.

```
cd ~\WebAuthnKit\scripts\PowerShell\
.\deployStarterKit.ps1
```

This will execute the PowerShell script that builds and deploys WebAuthnKit at AWS. The events of the deployment should be displayed through the command line interface. The deployment at AWS takes approximately 10 minutes to create all resources. If a rollback is necessary, it will take 10 minutes to delete all resources.

Linux/MacOS Bash script

This section explains how to use the Linux/MacOS Bash script, which manages the complete build and deployment of WebAuthnKit at the AWS backend, the AWS frontend, and the additional steps.

Editing the Linux/MacOS Bash script configuration file

The Linux/MacOS Bash script configuration file, ~\WebAuthnKit\scripts\Mac-Linux\deployStarterKit.config, should be reviewed and, if needed, edited.

```
{
    "AWS_CLI_PROFILE": "yubico",
    "AWS_REGION":"us-east-1",
    "S3_BUCKET_NAME":"",
    "CF_STACK_NAME":"",
    "SUFFIX":"",
    "USER_POOL_NAME":"",
    "DATABASE_NAME":"",
    "DATABASE_MASTER_USERNAME":"",
    "DATABASE_MASTER_PASSWORD":"",
```

```
    "DEFINE_AUTH_CHALLENGE_FUNC_NAME":"",
    "CREATE_AUTH_CHALLENGE_FUNC_NAME":"",
    "VERIFY_AUTH_CHALLENGE_FUNC_NAME":"",
    "WEBAUTHN_KIT_API_NAME":"",
    "WEBAUTHN_KIT_API_FUNC_NAME":"",
    "PRE_SIGNUP_FUNC_NAME":"",
    "JAVA_WEBAUTHN_LIB_FUNC_NAME":"",
    "CREATE_DATABASE_SCHEMA_FUNC_NAME":"",
    "CREATE_DATABASE_SCHEMA_CALLER_FUNC_NAME":"",
    "AMPLIFY_HOSTING_APP_NAME":""
}
```

In particular, the `AWS_REGION` parameter must be set to a valid AWS region for your AWS deployment. If the `AWS_REGION` parameter is not set, the PowerShell script will prompt the user to enter it.

If the `SUFFIX` parameter is not set, the PowerShell script will set this to a random six digit numeric value.

If the `DATABASE_MASTER_PASSWORD` parameter is not set, the PowerShell script will set this to a random sixteen character string.

All other parameters that are not specified in the configuration file will be set to default values, and the `SUFFIX` will be appended to these default values.

If a parameter is declared in the configuration file, it will be used by the PowerShell script exactly as declared, without appending any suffix.

Running the Bash script

The MacOS/Linux Bash script for deploying the WebAuthn Starter Kit is published at the WebAuthnKit GitHub repo. The script should be previously cloned or downloaded to the workstation.

Step 4.1. In order to run the Bash script, open a Terminal prompt and navigate to the folder ~\WebAuthnKit\scripts\Mac-Linux\.

Step 4.2. Execute the command `.\deployStarterKit.sh`.

```
cd ~\WebAuthnKit\scripts\Mac-Linux\
.\deployStarterKit.sh
```

This will execute the Bash script that builds and deploys WebAuthnKit at AWS. The events of the deployment should be displayed through the command line interface. The deployment at AWS takes approximately 10 minutes to create all resources. If a rollback is necessary, it will take 10 minutes to delete all resources.

Getting started with the clients

Once the deployment is completed, you can use the React client below to connect the AWS backend. The source code is available in this GitHub repository, and the deployment is described in this tutorial. A web browser with the React web client is launched when the WebAuthnKit deployment script is finished.