AWS Federated Login Advanced Details

Table of contents

Status	READY
Author(s)	Enterprise Information Security Team
Reviewer(s)	☑ Gene Wood ☐ April King
Classification	MOZILLA CONFIDENTIAL - STAFF AND NDA'D MOZILLIANS ONLY

Revisions

Version	Published	Changed By	Comment
CURRENT (v. 15)	**Jun 01, 2020 12:47**	 Gene Wood	
v. 14	Jun 01, 2020 12:46	 Gene Wood	
v. 13	Jun 01, 2020 12:17	 Gene Wood	
v. 12	Jan 16, 2020 12:08	 Gene Wood	
v. 11	Jan 16, 2020 12:06	 Gene Wood	
v. 10	Dec 30, 2019 17:47	 Gene Wood	
v. 9	Dec 30, 2019 17:20	 Gene Wood	
v. 8	Dec 30, 2019 17:12	 Gene Wood	
v. 7	Dec 23, 2019 10:24	 Gene Wood	
v. 6	Dec 10, 2019 13:28	 Gene Wood	
v. 5	Nov 26, 2019 12:35	 Gene Wood	
v. 4	Nov 22, 2019 09:18	 Gene Wood	
v. 3	Nov 22, 2019 09:16	 Gene Wood	
v. 2	Nov 22, 2019 09:11	 Gene Wood	
v. 1	Nov 22, 2019 08:59	 Gene Wood	

Summary & Scope

This document contains advanced details and concepts about the AWS Federated login system and the `maws` tool

For instructions on how an AWS user can access AWS using their federated single sign on login visit the How to login to AWS with Single Sign On page

For instructions on how an AWS account owner can enable federated AWS login with Single Sign On (SSO) in their AWS account visit the AWS Federated Login Account Setup page

AWS Federated Login Advanced Details

Concepts

AWS IAM Identity Provider

The AWS IAM Identity Provider is an object in AWS that is used to bind the AWS IAM Policies to the Auth0 Client (AKA Auth0 Application). This object is referenced by an AWS ARN. The ARN of our production AWS IAM Identity Provider is

```
arn:aws:iam::415589142697:oidc-provider/auth.mozilla.auth0.com/
```

The trust relationship between policies, the identity provider and the Auth0 client are

- AWS IAM Role policies across all Mozilla AWS accounts trust the AWS IAM Identity Provider.
- The AWS IAM Identity Provider is configured with 3 values to bind it to the Auth0 Application (AKA client)
    - The URL of the Mozilla Auth0 OIDC identity provider
        - https://auth.mozilla.auth0.com/
        - This ensures that only our Auth0 tenant can assert that a user is authenticated and a member of a given group
    - The Auth0 OIDC Client ID
        - `N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj`
        - This client ID is passed from Auth0 to AWS in the `aud` claim and is called and "Audience" on the AWS side
        - This client is registered in the production Auth0 tenant and can be seen here : https://manage.mozilla.auth0.com/dashboard/pi/auth/applications/N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj/settings
        - This ensures that only users who have authenticated to the `Mozilla AWS CLI` Auth0 Application (AKA Client) can log into AWS
    - The thumbprint of the TLS certificate of the certificate authority that issued the certificate for our Auth0 tenant, auth.mozilla.auth0.com
        - Prior to December 23 2019, Auth0 used DigiCert is the certificate authority that issued the certificate for our tenant
            - The thumbprint of the DigiCert CA certificate is `7ccc2a87e3949f20572b18482980505fa90cac3b`
        - After December 23 2019, Auth0 uses Sectigo Limited which is a rebranding of Comodo and relies on the `AddTrust External CA Root` CA (which expires in March of 2020)
            - The thumbprint of the AddTrust CA certificate is `02faf3e291435468607857694df5e45b68851868`
        - At some point in the near future Auth0 will update to a new cert chain that is cross signed by a new CA called `USERTrust RSA Certification Authority`
            - The thumbprint of USERTrust CA is `2B8F1B57330DBBA2D07A6C51F70EE90DDAB9AD8E`
        - These thumbprints were obtained by following the Obtaining the Root CA Thumbprint for an OpenID Connect Identity Provider instructions
        - These thumbprints prevents an attacker from getting a TLS certificate for `auth.mozilla.auth0.com` from some other certificate authority and impersonating our Auth0 tenant to AWS
- The Auth0 Application (AKA Client) is the root of the trust in this chain
    - The user groups that Auth0 asserts for a user are used by the AWS IAM Role policies to make authorization decisions about whether the user should or shouldn't be permitted to assume that AWS IAM Role

Deploy the AWS IAM OIDC Identity Provider on the command line

If you'd prefer to use the command line to deploy the identity provider follow these steps.

Note : You don't need to deploy the stack twice, once with the web UI and once on the command line. Only one stack need be deployed

1. Run this command

```
aws cloudformation create-stack \
    --stack-name OIDCIdentityProvider \
    --template-url https://s3-us-west-2.amazonaws.com/public.us-west-2.infosec.mozilla.org/oidc-identity-provider/5f48b78c87d98c18b55c98a6e5c285a80d53424c/oidc_identity_provider.5f48b78c87d98c18b55c98a6e5c285a80d53424c.yml \
    --capabilities CAPABILITY_IAM \
    --parameters \
        ParameterKey=Url,ParameterValue=https://auth.mozilla.auth0.com/ \
        ParameterKey=ClientIDList,ParameterValue=N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj \
        ParameterKey=ThumbprintList,ParameterValue=7ccc2a87e3949f20572b18482980505fa90cac3b,02faf3e291435468607857694df5e45b68851868,2B8F1B57330DBBA2D07A6C51F70EE90DDAB9AD8E
```

AWS IAM OIDC Identity Provider Notes

- The values that are being used to configure the production identity provider are
    - URL : https://auth.mozilla.auth0.com/
    - Client ID : `N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj`
    - Certificate Authority Thumbprint : `7ccc2a87e3949f20572b18482980505fa90cac3b` and `02faf3e291435468607857694df5e45b68851868` and `2B8F1B57330DBBA2D07A6C51F70EE90DDAB9AD8E`
    - The source code for this template can be found in https://github.com/mozilla/security/tree/master/operations/cloudformation-templates/oidc_identity_provider

Create the IAM role with the command line tool

You can either author the role by hand or use our `mozfederatedpolicybuilder` tool. To install the tool ensure you have Python installed in your OS as well as pip then run `pip install mozfederatedpolicybuilder`

Once you've installed the tool you can run it to walk you through the wizard

```
$ mozfederatedpolicybuilder
Policy format options :
* c/cloudformation : A YAML CloudFormation template which provisions a
    federated IAM role
* j/json-cloudformation : A JSON CloudFormation template which provisions a
    federated IAM role
* a/awscli : An AWS CLI command line command which creates a federated IAM role
* p/policy : The JSON trust relationship portion of the IAM policy (this can be
    copy pasted into the web console)

What format would you like the policy returned in? (c/cloudformation / j/json-cloudformation / a/awscli / p
/policy)
```

CloudFormation is a good choice as you can commit the resulting template into a code repository for safe keeping

```
c
```

```
User groups can be granted access to the federated IAM role.
* Supported : Allow users in the group foo to assume the IAM role : "foo"
* Supported : Allow users in the group foo as well as users in the group bar to
    assume the IAM role : "foo,bar"
* Supported : Allow users in any group that begins with "foo_" : "foo_*"
What groups would you like to grant access to this role?
```

Here we'll use the `team_hr` group name we learned about above. You would set this to the group name of your team that you want to allow to assume this role

```
team_hr
```

```
What name would you like for the AWS IAM Role?
```

Give your new IAM role a name

```
HRReaders
```

```
Would you like to attach a specific managed policy to the role or just output an example inline policy? If you
want to attach a manage policy enter it's policy ARN otherwise just hit enter:
```

In this example we'll pass a managed AWS policy ARN. You can find a policy which grants the permissions you're looking for in the IAM section of the AWS web console

```
arn:aws:iam::aws:policy/ReadOnlyAccess
```

Now the tool will output the resulting CloudFormation template

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Federated IAM Role which the groups team_hr are permitted to assume
Resources:
  HRReadersIAMRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: HRReaders
      Description: Federated IAM Role which the groups team_hr are permitted to assume
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
        - Principal:
            Federated:
              Fn::Join:
              - ''
              - - 'arn:aws:iam::'
                - Ref: AWS::AccountId
                - :oidc-provider/
                - auth.mozilla.auth0.com/
          Action: sts:AssumeRoleWithWebIdentity
          Effect: Allow
          Condition:
            StringEquals:
              auth.mozilla.auth0.com/:aud: N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj
            ForAnyValue:StringEquals:
              auth.mozilla.auth0.com/:amr:
              - team_hr
      ManagedPolicyArns:
      - arn:aws:iam::aws:policy/ReadOnlyAccess
```

Copy and paste this template into a file which you can use to create a new CloudFormation stack in the web console

Things to notice about this `AssumeRolePolicyDocument`

- It has a `Principal` that references the AWS IAM OIDC Identity Provider that you deployed in your account in the earlier steps
- It requires that users attempting to assume this role authenticate with the global Mozilla Auth0 client created for AWS federated login which is indicated by the `N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj` audience rule
- It requires that users be a member of the `team_hr` group as indicated by the `amr` condition

Create the IAM Role by hand

You can also create an IAM Role by hand instead of using the tool. Make sure that the role's Trust Policy or "AssumeRolePolicyDocument" has as statement with these elements

- `Principal` : The principal should be `Federated` with a value of `arn:aws:iam::YOUR-ACCOUNT-ID-GOES-HERE:oidc-provider/auth.mozilla.auth0.com/` where `YOUR-ACCOUNT-ID-GOES-HERE` is the AWS Account ID for your account
- `Action` : `sts:AssumeRoleWithWebIdentity`
- `Condition`
  - Ensure that the claim `auth.mozilla.auth0.com/:aud` is equal to (`StringEquals`) the client ID of `N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj`
  - Ensure that the claim `auth.mozilla.auth0.com/:amr` is equal to (`ForAnyValue:StringEquals`) the group name you determined above (for example `team_hr`)

Here is an example trust policy document that grants the team `team_hr` rights to assume the IAM role

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::YOUR-ACCOUNT-ID-GOES-HERE:oidc-provider/auth.mozilla.auth0.com/"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "auth.mozilla.auth0.com/:aud": "N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj"
        },
        "ForAnyValue:StringEquals": {
          "auth.mozilla.auth0.com/:amr": "team_hr"
        }
      }
    }
  ]
}
```

Quick create IAM Role details

The quick create IAM roles link in the AWS Federated Login Account Setup document launches a CloudFormation stack with two IAM Roles. The template for this stack is hosted in S3 at

https://s3-us-west-2.amazonaws.com/public.us-west-2.infosec.mozilla.org/federated-roles/default_oidc_federated_roles.yaml

The version controlled source of this template can be found at https://github.com/mozilla/security/blob/master/operations/cloudformation-templates/default_oidc_federated_roles.yaml

Session Length

The length of the session that a user gets when they perform a federated login, either in the web console or on the command line, depends on two things :

- Maximum session duration allowed by the IAM Role
- Session duration requested by `maws` or the https://aws.sso.mozilla.com/ website

IAM Role Max Session Duration

When an AWS account owner creates the IAM Roles that users will assume, the maximum session duration can be set to a value between 15 minutes and 12 hours. The quick setup of the two default IAM roles sets the maximum session duration to 12 hours. The mozfederatedpolicybuilder sets a 12 hour default for the CloudFormation template output and the awscli output (but not the JSON policy output as the maximum session duration lives outside the policy as an attribute of the role).

Requesting a session duration

When running `maws` the tool first requests a 12 hour session duration. If the maximum session duration for the IAM role is 12 hours, the resulting session will last for 12 hours. If the maximum session duration of the role is less than 12 hours, then `maws` attempts again with the default of 1 hour. If the maximum session duration of the IAM role is less than 1 hour `maws` will fail.

When visiting https://aws.sso.mozilla.com/ the default session length requested is 12 hours. The user can also pass a `duration_seconds` query parameter to set the set the session duration to a given number of seconds. If the maximum session duration allowed for the role is less than the session length requested, the login will fail.

Configuring maws

If you'd like to manually configure `maws` instead of using the `mozilla-aws-cli-mozilla` python package, you can create the file `~/.maws/config` in your home directory with the following contents:

```
[DEFAULT]
client_id = N7lULzWtfVUDGymwDs0yDEq6ZcwmFazj
idtoken_for_roles_url = https://roles-and-aliases.security.mozilla.org/roles
well_known_url = https://auth.mozilla.auth0.com/.well-known/openid-configuration
```

Using maws with Firefox Multi-Account Containers

If you use Firefox Multi-Account Containers and have a dedicated container for Mozilla activities, such that your default non-container isn't ever logged into SSO, you can use the containerise Firefox add-on to address this.

You'll know that you have this problem if you run maws, it spawns a tab in your browser, but that tab isn't logged into SSO, because you do Mozilla stuff in tabs in a dedicated container.

To solve this

1. Install containerise in Firefox
2. Click the containerise icon in the toolbar 
3. Click the down arrow in the upper right corner of the containerise window to show you the list of your containers
4. Select the container you use for Mozilla. The containerise window should now show you the Mozilla container name at the top
5. Click the plus icon
6. In the text box that is created, replace whatever domain name that's there (which is the domain name of the page you're currently on) with this domain name

   ```
   auth.mozilla.auth0.com
   ```

7. Click the diskette icon to save

Now test that it's working by launching maws from the command line. It should open a login window in the Firefox container you selected.

Notes

- If you wanted to create an identity provider that links to the development Auth0 instance you would click this link but this is only used by the IAM team while developing the IAM system