AWS IDM Configuration

Overview

These are the configuration made to support a variety of needs including but not limited to:

- Federated access to the AWS console (Authentication and Authorization)
- Elevated VPN access for members of the RHEDcloudAdministratorRole
- Provisioned to a distribution list use to communicate with all users who have AWS account access
- Control access to AWS section of Confluence

Microsoft LDS (LDAP) Configuration

Overview

The LDS directory contains all Emory University and Emory Healthcare user accounts. These accounts reside in separate OUs. Emory University accounts are in ou=People and Emory Healthcare accounts are in ou=EHC_People.

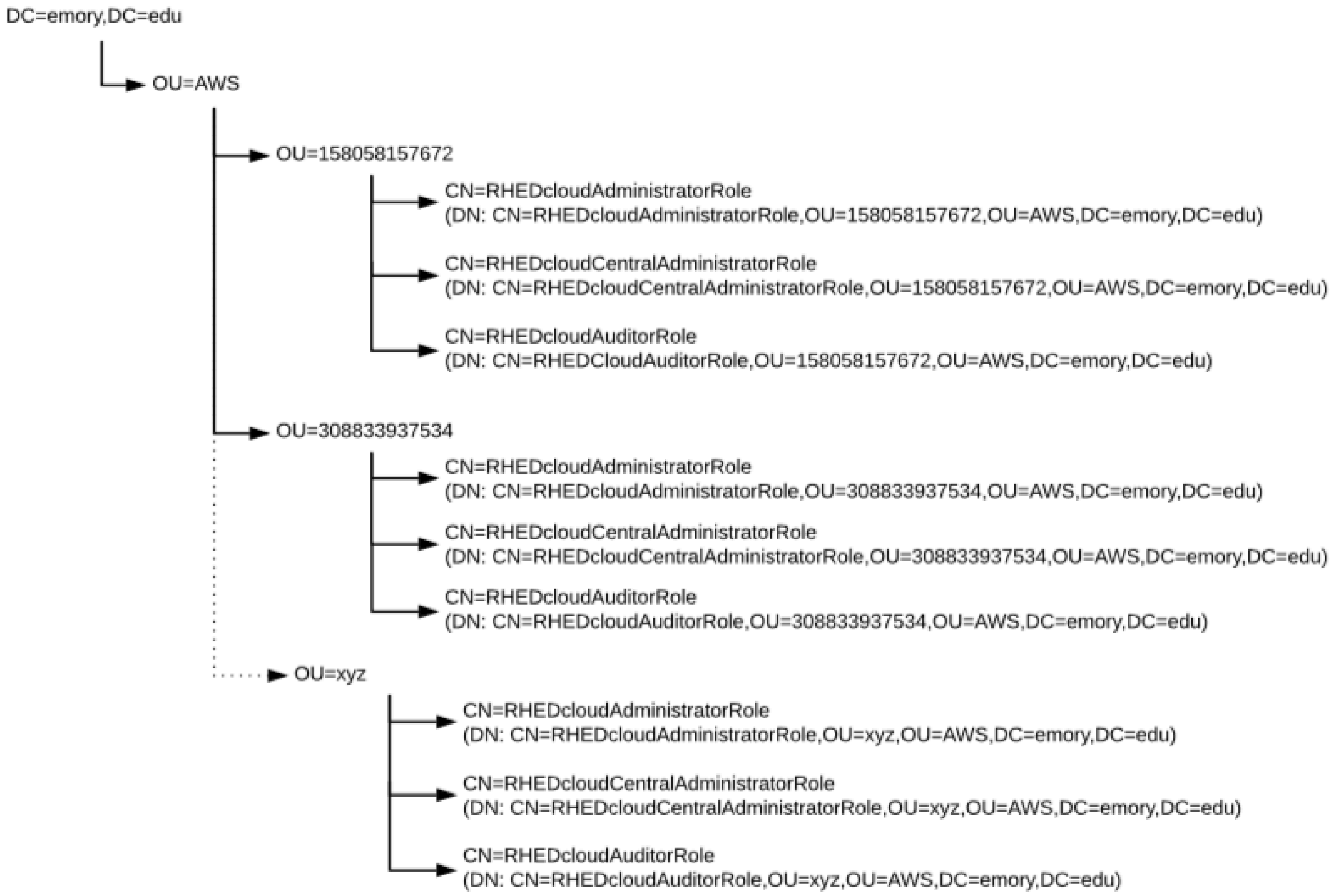
Passwords are not stored in LDS. Passwords for Emory University accounts are proxied to University AD. Passwords for Emory Healthcare accounts are proxied to Healthcare AD.

Microsoft LDS is the identity store used but by Shibboleth to perform authentication and attribute release. For AWS, Shibboleth will provide authentication and release attributes used for authorization.

1. User needs an account in LDS in order to authenticate to AWS. Account provisioning/de-provisioning process is handled by NetIQ IDM.
2. User needs to be in the correct LDS groups in order to "assume the role" in an AWS Account ID. The LDS OUs and groups are created by generate requests made by the AWS account service to the LDSService.

All groups associated with an AWS Account ID should be placed in the the OU that is named the AWS Account ID. For example, to add a new group called RHEDCloudAdministratorRole to the AWS Account ID (158058156672), a new group object needs to be added under OU=158058156672 called CN=RHEDCloudAdministratorRole. The full DN would be CN=RHEDCloudAdministratorRole,OU=158058156672, OU=AWS,DC=emory,DC=edu

LDS AWS Group Structure



Example User in LDS assigned to AWS Groups

This is a sample of a user that has been assigned to a couple of AWS groups. These groups are assigned just like any other LDS group. When a user becomes a member of a group, the user's memberOf attribute is populated with the DN of the group.

Sample LDS Attributes of cn=userid1

```
DN: CN=rsmith59,OU=People,DC=emory,DC=edu
displayName: Robert Smith
mail: bob.smith@emory.edu
memberOf: CN=RHEDcloudAdministratorRole,OU=631868038169,OU=AWS,DC=emory,
DC=edu
memberOf: CN=RHEDcloudAuditorRole,OU=158058157672,OU=AWS,DC=emory,DC=edu
memberOf: CN=EU_STAFF,OU=Groups,DC=emory,DC=edu
memberOf: CN=Idm Team,OU=Groups,DC=emory,DC=edu
```

MyNetID Roles and Resources

Overview

All AWS account will be created with 3 roles:

- RHEDcloudAdministratorRole
- RHEDcloudCentralAdministratorRole (LITS Administrators)
- RHEDcloudAuditorRole (Read Only)

These roles are used to facilitate federated login to the AWS console using Shibboleth and LDS groups. These roles are also used to provision users to a variety of resources that provide various functionality. Some of the function are:

- Elevated VPN access for members of the RHEDcloudAdministratorRole
- Provisioned to a distribution list use to communicate with all users who have AWS account access
- Control access to AWS section of Confluence

Standard Roles Assigned to each AWS Account

The following roles and resources are created for every Emory supported AWS account.

NOTE: xxx represents the AWS Account ID. xxxDLxxx is the name of the distribution list in Enterprise AD.

Role

RGR_AWS-xxx-RHEDcloudAdministratorRole

Resource(s)

MDSG_AWS-xxx-RHEDcloudAdministratorRole

Provisions University LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

HDSG_AWS-xxx-RHEDcloudAdministratorRole

Provisions Healthcare LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

EADG_xxxDLxxx

Enterprise AD distribution list that is used as the primary email of the AWS account.

RGR_AwsUsers

An eDirectory Group that contains all AWS users.

RGR_AwsVpnAllow

An eDirectory Group that contains all users who are automatically granted **RGR_VPNAllow** and **RGR_AWSAllow**

Role

RGR_AWS-xxx-RHEDcloudCentralAdministratorRole

This role is provisioned the with LITS Central Administrators eDirectory group (**RGR_AwsCentralAdministrators**).

Resource(s)

MDSG_AWS-xxx- RHEDcloudCentralAdministratorRole

Provisions University LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

HDSG_AWS-xxx- RHEDcloudCentralAdministratorRole

Provisions Healthcare LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

Role

RGR_AWS-xxx-RHEDcloudAuditorRole

Resource(s)

MDSG_AWS-xxx-RHEDcloudAuditorRole

Provisions University LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

HDSG_AWS-xxx-RHEDcloudAuditorRole

Provisions Healthcare LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

RGR_AwsUsers

An eDirectory Group that contains all AWS users.

Configuration Roles

Role

RGR_AwsCentralAdministrators

LITS Central Administrators Role

Resource(s)

RGR_AwsUsers

An eDirectory Group that contains all AWS users.

RGR_AwsCentralAdministrators

An eDirectory Group that containing all LITS central administrators.

Role

RGR_AWS-120396672714-RHEDcloudMaintenanceOperatorRole

RGR_AWS-473457766643-RHEDcloudMaintenanceOperatorRole

RGR_AWS-522074860894-RHEDcloudMaintenanceOperatorRole

The purpose of this role is to allow the user to assume the role on the master aws accounts (Test, Stage, and Prod). This means administrators in these roles can assume the RHEDcloudMaintenanceOperatorRole in the master accounts via the Emory SSO Web Console or the TKI Service. The RHEDcloudMaintenanceOperatorRole in the master account has been given assumeRole permissions to the RHEDcloudMaintenanceOperatorRole in all of the customer accounts. So once the administrator has logged into the master account with RHEDcloudMaintenanceOperatorRole, he or she will be able to perform maintenance on the customer accounts.

Resource(s)

MDSG_AWS-xxx-RHEDcloudMaintenanceOperatorRole

Provisions University LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

HDSG_AWS-xxx-RHEDcloudMaintenanceOperatorRole

Provisions Healthcare LDS user accounts to LDS Group. LDS Group that is used by Shibboleth to generate the AWS Role ARN used for federated login.

Role

RGR_AWSAllow

Role that allows users to gain elevated VPN access for AWS.

Resource(s)

MDSG_AWSAllow

Provisions University LDS user accounts to an LDS Group.

HDSG_AWSAllow

Provisions Healthcare LDS user accounts to an LDS Group.

UADG_AWSAllow

Provisions University AD user account to a AD group.

Role

RGR_AWS-Support

Role used to provision users to the support distribution list

Resource(s)

EADG_AWS-Support

Provisions Enterprise AD user accounts to an Enterprise AD DL.

Role

RGR_AWS-Security

Role used to provision users to the security distribution list

Resource(s)

EADG_AWS-Security

Provisions Enterprise AD user accounts to an Enterprise AD DL.

Role

RGR_Aws-Project-Env-Comm

Distribution List for communicating the status of MyNetID. MyNetID was being upgraded during a portion of the AWS project. It was important to communicate downtime to team members.

Resource(s)

EADG_Aws-Project-Env-Comm

Provisions Enterprise AD user accounts to an Enterprise AD DL.

Role

RGR_AWS-Central-Billing

Role used to provision users to the central-billing distribution list

Resource(s)

EADG_AWS-Central-Billing

Provisions Enterprise AD user accounts to an Enterprise AD DL.

Role

RGR_AwsAccountWebService

Used to provision users to a OpenEAIGroup used by ESB.

Resource(s)

MDSG_AwsAccountWebService

OpenEAIGroup

Role

RGR_AwsWhiteList

Role used to determine who is eligible to create an account, log in to the VPCP, access the Service-now form, etc. However, there is the rare chance that we might have a student who we want in our environment or perhaps even a retiree or maybe even a pre-start (but that’s really highly unlikely). There couple of other scenarios where someone who is not eligible per our business rules might actually be eligible through an exception process.

Resource(s)

None

This role is used directly by the VPCP application.

Role

RGR_NetworkAdministrators

Contains all network administrators. This roles is used by the VPCP application. It wasn't given an AWS specific name because it might get used elsewhere.

Resource(s)

None

This role is used directly by the VPCP application.

Role

RGR_AwsUsers

This role contains all AWS users. This is anybody in an AWS central administrator, administrator, or auditor role.

Resource(s)

EADG_aws-users

Distribution List

MDSG_aws-users

This group is used by Confluence for authorization to the AWS section of the wiki.

eDirectory resources and entitlements

Resource

RGR_AwsUsers

An eDirectory Group that contains all AWS users.

Entitlement

\EMORYPROD\EmoryProd\DataGroups\AwsUsers

All User Distribution List

aws-users@emory.edu

AWS Account Shibboleth Configuration

Emory IDP configuration

The Emory IDP must releases two SAML2 attributes for AWS:

- **Role** - The Amazon Resource Name (ARN) of the role to assume. This attribute is dynamically populated within the Emory IDP at the time of authentication. It is built using values of the user's memberOf attribute in LDS that are specific to AWS.
- **RoleSessionName** - An identifier for the assumed role session. This is the public person identifier (serialNumber in LDS). It is a non-sensitive immutable identifier.

Emory IDP configuration sample

Simple Configuration example using regex

attribute-resolver.xml

```
<resolver:AttributeDefinition id="awsRoles" xsi:type="ad:Mapped"
sourceAttributeID="memberOf">
        <resolver:Dependency ref="vlad"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="
https://aws.amazon.com/SAML/Attributes/Role" friendlyName="Role" />
        <ad:ValueMap>
                <ad:ReturnValue>arn:aws:iam::$2:saml-provider
/Emory_Prod_IDP,arn:aws:iam::$2:role/$1</ad:ReturnValue>
                <ad:SourceValue>CN=([^,]*),OU=([^,]*),OU=AWS,DC=emory,
DC=edu</ad:SourceValue>
        </ad:ValueMap>
</resolver:AttributeDefinition>


<resolver:AttributeDefinition id="awsRoleSessionName" xsi:type="ad:Simple"
sourceAttributeID="serialNumber">
        <resolver:Dependency ref="vlad"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="
https://aws.amazon.com/SAML/Attributes/RoleSessionName" friendlyName="
RoleSessionName" />
</resolver:AttributeDefinition>
```

The following is a more complicated example using javascript. This is the code currently deployed in production. AWS@Emory has multiple CFTs currently deployed. There are three different role naming conventions used based upon the CFT. The latest CFT adds a path tot the role ARN. This code determines when to add the path when building the ARN.

attribute-resolver.xml

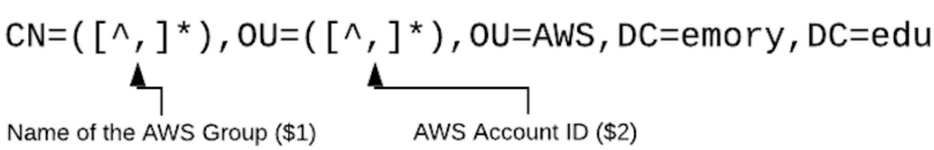
```
<!--  START for Amazon Web Services (AWS) 20190116 -->
<resolver:AttributeDefinition id="awsRoles" xsi:type="Script" xmlns="urn:
mace:shibboleth:2.0:resolver:ad" sourceAttributeID="memberOf">
        <resolver:Dependency ref="vlad_shib_attr" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="
https://aws.amazon.com/SAML/Attributes/Role" friendlyName="Role" />
    <Script>
            <![CDATA[
                if(memberOf != null)
            {
```

```
                    // IMPORTANT: This value needs to change depending
upon the IDP. Remove the comment for the
                //            variable declaration that matches the IDP
environment.
                // Dev
                //var awsIdp = "Emory_Dev_IDP";
                // QA
                //var awsIdp = "Emory_QA_IDP";
                // Prod
                var awsIdp = "Emory_Prod_IDP";

                // Setup arrays containing the AWS Account IDs that use
particular paths
                var noPath = ["308833937534","158058157672","
073387691482","315249003173","698677649721",
                "134285882893","709172338785","468626563127","
926150410375","931785389836","919369240275","728238754923",
                "791630828116","332334003451","872172917288","
944522672192", "181123324664", "934920654945"];

                var path = "";
                // Spin through all the values of the memberOf attribute
                for( i=0; i < memberOf.getValues().size(); i++)
                {
                        // Setup a work string. Just used for easier to
read code
                    var workString = memberOf.getValues().get(i);
                    // Process the AWS groups in OU=AWS
                    if(workString.indexOf(",OU=AWS,DC=emory,DC=edu") != -1)
                    {
                            var workArray = workString.split(",");
                        // Extract role name and aws account id
                        var roleName = (workArray[0].split("="))[1];
                        var awsAccountId = (workArray[1].split("="))[1];
                        // Check to see if the account needs a path
                        if(noPath.indexOf(awsAccountId) != -1)
                        {
                                path = "";
                        }else{
                                path = "rhedcloud/";
                        }
                        // Build the AWS ARN
                        awsRoleArn = "arn:aws:iam::" + awsAccountId + ":
saml-provider/" + awsIdp + ",arn:aws:iam::" + awsAccountId + ":role/" +
path + roleName;
                        // Add it to the awsRole attribute
                        awsRoles.getValues().add(awsRoleArn);
                    }
                    // Process the AWS groups in OU=AWS
                    else if(workString.indexOf(",OU=AWS-HOSTING,DC=emory,
DC=edu") != -1)
                    {
                            var workArray = workString.split(",");
```

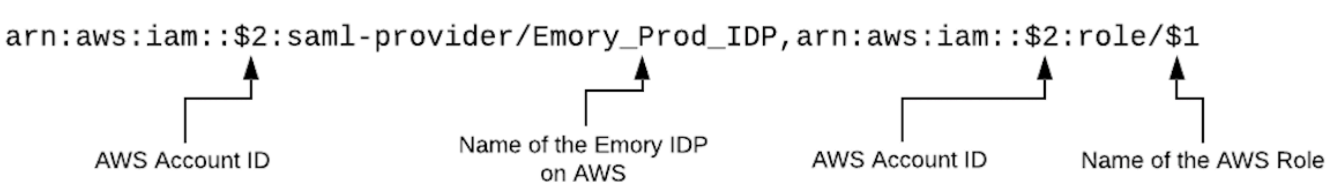
```
                        // Extract role name and aws account id
                        var roleName = (workArray[0].split("="))[1];
                        var awsAccountId = (workArray[1].split("="))[1];
                        // Use this path but we may change it
                        path = "rhedcloud/";
                        // Build the AWS ARN
                        awsRoleArn = "arn:aws:iam::" + awsAccountId + ":
saml-provider/" + awsIdp + ",arn:aws:iam::" + awsAccountId + ":role/" +
path + roleName;
                        // Add it to the awsRole attribute
                        awsRoles.getValues().add(awsRoleArn);
                    }
                    }
            }
            ]]>
        </Script>
</resolver:AttributeDefinition>


<resolver:AttributeDefinition id="awsRoleSessionName" xsi:type="ad:Simple"
sourceAttributeID="serialNumber">
        <resolver:Dependency ref="vlad"/>
    <resolver:AttributeEncoder xsi:type="enc:SAML2String"
            name="https://aws.amazon.com/SAML/Attributes/RoleSessionName"
friendlyName="RoleSessionName" />
</resolver:AttributeDefinition>
<!--  END for Amazon Web Services (AWS)  20180523 -->
```

Explanation of RegEx that generated the ARN from memberOf attribute

The regex expression extracts the group name and the AWS account ID from the memberOf attribute in LDS.



The Role is built using the values obtained from the regex and some hardcoded values.



Example

Sample LDS Attributes of cn=userid1

```
DN: CN=userid1,OU=People,DC=emory,DC=edu
displayName: Robert Smith mail: bob.smtih@emory.edu
memberOf: CN=Administrator,OU=631868038169,OU=AWS,DC=emory,DC=edu
memberOf: CN=CentralAdministrator,OU=158058157672,OU=AWS,DC=emory,DC=edu
memberOf: CN=CentralAdministrator,OU=308833937534,OU=AWS,DC=emory,DC=edu
memberOf: CN=EU_STAFF,OU=Groups,DC=emory,DC=edu
memberOf: CN=Idm Team,OU=Groups,DC=emory,DC=edu
```

Sample extract from the SAML assertion


```
<saml2:AttributeStatement>
        <saml2:Attribute FriendlyName="Role" Name="https://aws.amazon.com
/SAML/Attributes/Role" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:uri" >
                <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xsi:type="xsd:string" >arn:aws:iam::308833937534:saml-
provider/Emory_Prod_IDP,arn:aws:iam::308833937534:role
/CentralAdministrator</saml2:AttributeValue>
                <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xsi:type="xsd:string" >arn:aws:iam::158058157672:saml-
provider/Emory_Prod_IDP,arn:aws:iam::158058157672:role
/CentralAdministrator</saml2:AttributeValue>
                <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xsi:type="xsd:string" >arn:aws:iam::631868038169:saml-
provider/Emory_Prod_IDP,arn:aws:iam::631868038169:role/Administrator<
/saml2:AttributeValue>
        </saml2:Attribute>
        <saml2:Attribute FriendlyName="RoleSessionName" Name="https://aws.
amazon.com/SAML/Attributes/RoleSessionName" NameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri" >
                <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001
/XMLSchema-instance" xsi:type="xsd:string" >P1234567</saml2:
AttributeValue>
        </saml2:Attribute>
</saml2:AttributeStatement>
```