# Create IAM User in AWS.

IAM User creation catalogue is used to provision IAM user in AWS Environment and also it will do the below functionalities during the provision.

* IAM User is created using AWS Management Console Access (User Name and Password).
* Default Password is **‘/iam/usr/pwd’.**
* Password can also be changed on first login by enabling the option Password Reset on First Login.
* When this option is enabled , it auto-attaches the policy called “IAMUserChangePassword”.
* Additionally , we can also attach the other policies through policy table.

## Service Now Workflow:

## Sample Payload

### IAM User Creation

{

"ScriptFiles": [

{

"Name": "IAM\_User.py",

"Type": "PY",

"Uri": "CE\_Core\_Templates/AWS/IaaS\_Automation/IAM Management/IAM\_User.py"

},

{

"Name": "IAM\_User.yml",

"Type": "CFT",

"Uri": "CE\_Core\_Templates/AWS/IaaS\_Automation/IAM Management/IAM\_User.yml"

}

],

"ScriptPayload": {

"Region": "us-west-2",

"UserName": "Test",

"PasswordResetRequired": "true",

"ManagedPolicyArns": "arn:aws:iam::aws:policy/SecurityAudit,arn:aws:iam::aws:policy/AmazonFSxConsoleFullAccess,arn:aws:iam::aws:policy/service-role/ServerMigrationServiceLaunchRole,arn:aws:iam::aws:policy/AdministratorAccess,arn:aws:iam::aws:policy/AmazonESFullAccess,arn:aws:iam::aws:policy/AmazonEKS\_CNI\_Policy,arn:aws:iam::aws:policy/IAMUserChangePassword",

"Path": "/",

"StackName": "SCTASK0013229",

"Tags": [

{

"Key": "Owner",

"Value": "Cloud\_Exponence"

},

{

"Key": "Project",

"Value": "IT"

},

{

"Key": "BusinessUnit",

"Value": "IT IS"

},

{

"Key": "Application",

"Value": "Ansible"

},

{

"Key": "Environment",

"Value": "Non Production"

},

{

"Key": "CreatedBy",

"Value": "CloudExponence"

},

{

"Key": "RequestID",

"Value": "RITM0012430"

}

]

}

}

### Developer Notes:

IAM User Creation – Form Design

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Validation if any** |
| User Name | Single Line text | Name of the user | Validate UserName |
| Path | Single Line text | Path for user name | Validate Path |
| Password Reset Required on First Login | Check box | If to enable password reset on first login | None |
| Policy Name | List Collector | Attaches the policies to the user created | ValidatePolicyCountTo10 |
| User Check | Check box | To check if all validation are satisfied | OnSubmit |

Default Values

|  |  |  |
| --- | --- | --- |
| **Key Name** | **Default Values** |  |
| Path | / |  |

Internal Mapping in Form

|  |  |  |  |
| --- | --- | --- | --- |
| **Key Name** | **Type** | **Table** |  |
| PolicyName | List Collector | CE IAM Policy[u\_cmdb\_ci\_ce\_iam\_policy] |  |

\*Validate UserName :

This Script is to validate if given user name is unique and matches the regex pattern provided. The Regex pattern consists of alphanumeric characters and \_+=,.@-' . And this will invoke the backend script “ce\_validate\_user\_aws” which in turn return a list of users from aws console via REST API “AWSOAUTHENDPOINT”. This is to avoid the duplicate value of user name.

\*Validate Path :

This script is to validate the path provided using regex pattern. The path should begin and end with ‘/’ and contain alphanumeric characters and \_.

\*Validate PolicyCountTo10 :

This script is to validate the number of policies per user must be 10.

\*OnSubmit:

This script is to validate the variable usercheck. The request will be processed only if the variable returns true.

Automation Workflow:

The “ScriptFiles” section contains the scripts location in the github repo and ScriptPayload” contains the list of inputs to be passed.

The ResourceDeploymentStepapigateway invokes ResourceDeploymentSingleStepFunction.

The following operations happen in the “ResourceDeploymentSingle”stepfunction:

1. The “ResourceDeploymentStep” lambda is invoked .
2. Lambda will check the input format of payload and verifies it. It also get the credentials through the role associated to it for the below executions.
3. It will get the git repo credentials from the SSM parameter store and then download the IAM\_User.py and IAM\_User.yml files in s3.
4. Then the inputs will be passed to the python file where it will trigger the CloudFormation Template file from S3 bucket if the stack has to be created. If there is no stack creation, python file itself will do the required activity.
5. The final activity of the step function is to notify success message to servicenow if the stack is successfully created.
6. If there is no stack creation involved, it will directly notify the servicenow after the python scripts successfully executes

Error Handling:

1. Step function will check if the stack is successfully created. It will notify service now of the status of stack creation if error
2. If there is no stack creation, it will notify servicenow if there are any error from python scripts.
3. Any other runtime error from step function /lambda is also notified to servicenow.