AWS Security Blog

How to Use AWS Organizations to Automate End-to-End Account Creation

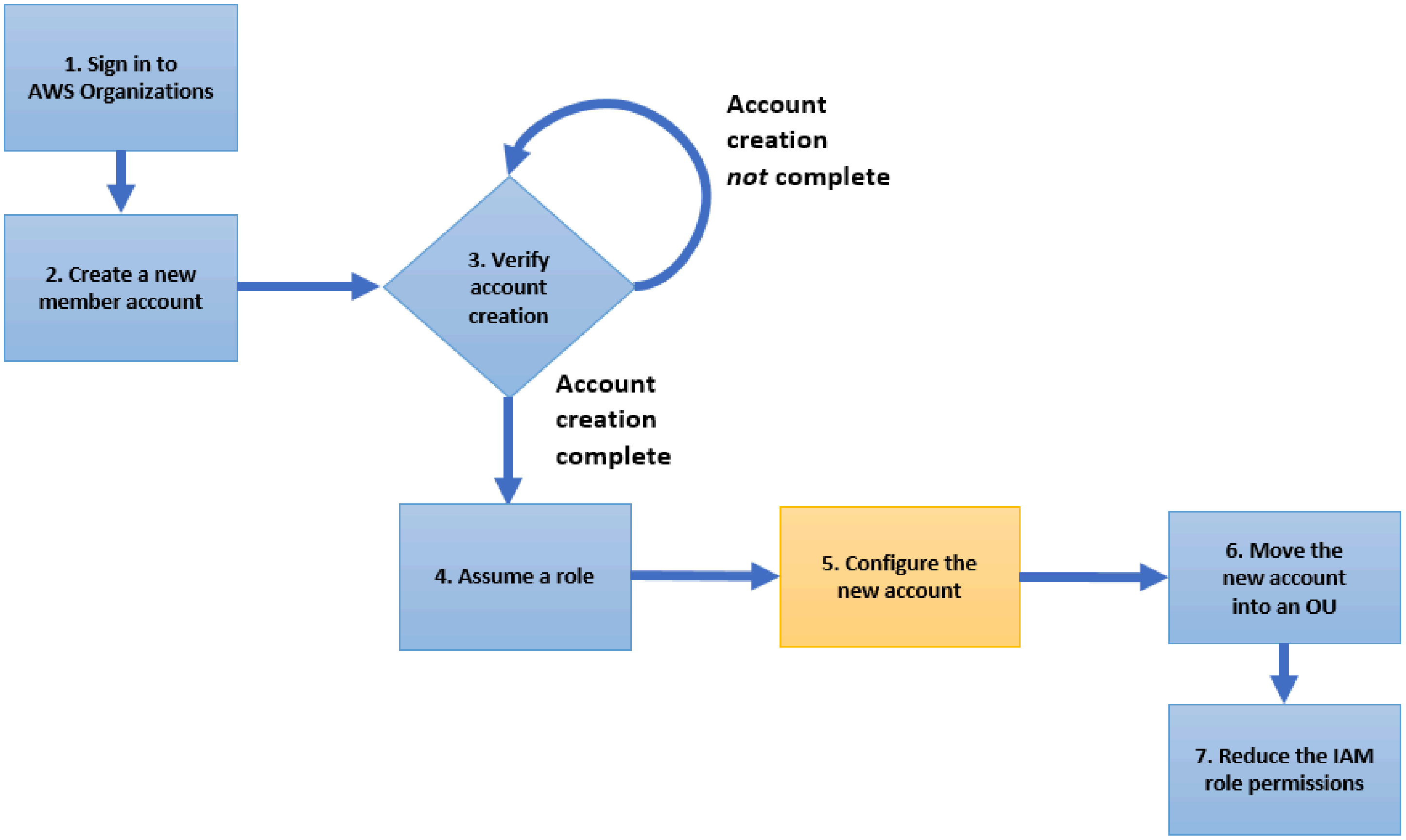
by David Schonbrun | on 24 JUL 2017 | in AWS Organizations, How-To | Permalink | Comments | Share

AWS Organizations offers new capabilities for managing AWS accounts, including automated account creation via the Organizations API. For example, you can bring new development teams onboard by using the Organizations API to create an account, AWS CloudFormation templates to configure the account (such as for AWS Identity and Access Management [IAM] and networking), and service control policies (SCPs) to help enforce corporate policies.

In this blog post, I demonstrate the step-by-step process for end-to-end account creation in Organizations as well as how to automate the entire process. I also show how to move a new account into an organizational unit (OU).

Process overview

The following process flow diagram illustrates the steps required to create an account, configure the account, and then move it into an OU so that the account can take advantage of the centralized SCP functionality in Organizations. The tasks in the blue nodes occur in the master account in the organization in question, and the task in the orange node occurs in the new member account I create. In this post, I provide a script (in both Bash/CLI and Python) that you can use to automate this account creation process, and I walk through each step shown in the diagram to explain the process in detail. For the purposes of this post, I use the AWS CLI in combination with CloudFormation to create and configure an account.


1. Sign in to AWS Organizations
2. Create a new member account
3. Verify account creation
Account creation *not* complete
Account creation complete
4. Assume a role
5. Configure the new account
6. Move the new account into an OU
7. Reduce the IAM role permissions

The account creation process

Follow the steps in this section to create an account, configure it, and move it into an OU. I am also providing a script and CloudFormation templates that you can use to automate the entire process.

1. Sign in to AWS Organizations

In order to create an account, you must sign in to your organization’s master account with a minimum of the following permissions:

- `organizations:DescribeOrganization`
- `organizations:CreateAccount`

2. Create a new member account

After signing in to your organization’s master account, create a new member account. Before you can create the member account, you need three pieces of information:

- **An account name** – The friendly name of the member account, which you can find on the **Accounts** tab in the master account.
- **An email address** – The email address of the owner of the new member account. This email address is used by AWS when we need to contact the account owner.
- **An IAM role name** – The name of an IAM role that Organizations automatically preconfigures in the new member account. This role trusts the master account, allowing users in the master account to assume the role, as permitted by the master account administrator. The role also has administrator permissions in the new member account. If you do not change the role’s name, the name defaults to `OrganizationAccountAccessRole`.

The following AWS CLI command creates a new member account.

```
aws organizations create-account --email <newAccEmail> --account-name "<newAccName>" --
```

To explain the **placeholder values** in the preceding command that you must update with your own values:

- **newAccEmail** – The email address of the owner of the new member account. This email address must *not* already be associated with another AWS account.
- **newAccName** – The friendly name of the new member account.
- **roleName** – The name of an IAM role that Organizations automatically preconfigures in the new member account. The default name is `OrganizationAccountAccessRole`.

This CLI command returns a `request_id` that uniquely identifies the request, a value that is required for in Step 3.

Important: When you create an account using Organizations, you currently cannot remove this account from your organization. This, in turn, can prevent you from later deleting the organization.

3. Verify account creation

Account creation may take a few seconds to complete, so before doing anything with the newly created account, you must first verify the account creation status. To check the status, you must have at least the following permission:

- `organizations:DescribeCreateAccountStatus`

The following CLI command, with the `request_id` returned in the previous step as an input parameter, verifies that the account was created:

```
aws organizations describe-create-account-status --create-account-request-id <request_id
```

The command returns the state of your account creation request and can have three different values: `IN_PROGRESS`, `SUCCEEDED`, and `FAILED`.

4. Assume a role

After you have verified that the new account has been created, configure the account. In order to configure the newly created account, you must sign in with a user who has permission to assume the role submitted in the `createAccount` API call. In the example in Step 1, I named the role `OrganizationAccountAccessRole`; however, if you revised the name of the role, you must use that revised name when assuming the role. Note that when an account is created from within an organization, cross-account trust between the master and programmatically created accounts is automatically established.

The following CLI command assumes a role.

```
aws sts assume-role --role-arn <role-arn> --role-session-name <"role-session-name">
```

To explain the **placeholder values** in the preceding command that you must update with your own values:

- **role-arn –** The Amazon Resource Name (ARN) of the role to assume.
- **role-session-name –** An identifier for the assumed role session.

5. Configure the new account

After you assume the role, build the new account's networking, IAM, and governance resources as explained in this section. Again, to learn more about and download the account creation script and the templates that can automate this process, see “Automating the entire end-to-end process” later in this post.

A. Networking – Amazon VPC, web access control lists (ACLs), and Internet gateway:

1. Create a new Amazon VPC to enable you to launch AWS resources in a virtual network that you define.
2. Run the script at the end of this post to create a VPC with two subnets (one public subnet and one private subnet) in each of two Availability Zones.
3. Set up web ACLs to control traffic in and out of the subnets. You can set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC.
4. Connect your VPC to remote networks by using a VPN connection.
5. If the resources in the VPC require access to the Internet, create an Internet gateway to allow communication between instances in your VPC and the Internet.

B. IAM – Identity provider (IdP), IAM policies, and IAM roles:

1. Many customers use enterprise federated authentication (such as Active Directory) to manage users and permissions centrally outside AWS. If you use federated authentication, set up an IdP.
2. After you set up the IdP, author the customer managed IAM policies you will use.
3. Use AWS managed policies or your customer managed policies to manage access to your AWS resources.

C. Governance – AWS Config Rules:

1. Create AWS Config rules to help manage and enforce standards for resources deployed on AWS.
2. Develop a tagging strategy that specifies a minimum set of tags required on every taggable resource. A tagging rule checks that all resources created or edited fulfill this requirement. A noncompliance report is created to document resources that do not meet the AWS Config rule. AWS Lambda scripts can also be launched as a result of AWS Config rules.

6. Move the new account into an OU

Before allowing your development teams to access the new member account that you configured in the previous steps, apply an SCP to the account to limit the API calls that can be made by all users. To do this, you must move the member account into an OU that has an SCP attached to it.

An OU is a container for accounts. It can contain other OUs, allowing you to create a hierarchy that resembles an upside-down tree with a “root” at the top and OU “branches” that reach down, ending with accounts that are the “leaves” of the tree. When you attach a policy to one of the nodes in the hierarchy, it affects all the branches (OUs) and leaves (accounts) under it. An OU can have exactly one parent, and currently, each account can be a member of exactly one OU.

The following CLI command moves an account into an OU.

```
aws organizations move-account --account-id <account_id> --source-parent-id <source_parent
```

To explain the **placeholder values** in the preceding command that you must update with your own values:

- **account_id** – The unique identifier (ID) of the account you want to move.
- **source_parent_id** – The unique ID of the root or OU from which you want to move the account.
- **destination_parent_id** – The unique ID of the root or OU to which you want to move the account.

7. Reduce the IAM role permissions

The `OrganizationAccountAccessRole` is created with full administrative permissions to enable the creation and development of the new member account. After you complete the development process and you have moved the member account into an OU, reduce the permissions of `OrganizationAccountAccessRole` to match your anticipated use of this role going forward.

Automating the entire end-to-end process

To help you fully automate the process of creating new member accounts, setting up those accounts, and moving new member accounts into an OU, I am providing a script in both Bash/CLI and Python. You can modify or call additional CloudFormation templates as needed.

Download the script and CloudFormation templates

Download the script and CloudFormation templates to help you automate this end-to-end process. The global variables in the script are set in the opening lines of code. Update these variables' values, and they will flow as input parameters to the API commands when the script is executed. I have prepopulated the `roleName` by using AWS best practices nomenclature, but you can use a custom name.

I am including the following descriptions of the elements of the script to give you a better idea of how the script works.

Bash/CLI:

- `Organization-new-acc.sh` – An example shell script that includes parameters, account creation, and a call to the JSON sample templates for each of three subtasks in Step 5 earlier in this post.
- `CF-VPC.json` – An example Cloud Formation template that creates and configures a VPC in the new member account. Each AWS account must have at least one VPC as a networking construct where you can deploy customer resources. Though AWS does create a default VPC when an account is created, you must configure that VPC to meet your needs. This includes creating subnets with specific IP Classless Inter-Domain Routing (CIDR) blocks, creating gateways (including an Internet gateway, a customer gateway, a VPN tunnel, AWS Storage

Gateway, Amazon API Gateway, and a NAT gateway), and VPC peering connections. Web ACLs are also part of this process to limit the source IP addresses and ports that can access the VPC. The VPC created by this script includes four subnets across two Availability Zones. Two of the subnets are public and two are private.

- `CF-IAM.json` – An example Cloud Formation template that creates IAM roles in the new member account. As part of a security baseline, you should develop a standard set of IAM roles and related policies. Update this template with the IAM role definitions and policies you want to create in the member account to controls privilege and access.
- `CF-ConfigRules.json` – An example Cloud Formation template that creates an AWS Config rule to enforce tagging standards on resources created in the new account.
- `Organization_Output.docx` – Example output of the results from running `Organization-new-acc.sh`.

Python:

- `Create_account_with_iam.py` – An example Python template that creates an account, moves it into an OU, applies an SCP, and then calls additional templates to deploy resources. `CF-VPC.JSON` can be called by this template if you first customize the `.json` file.
- `Baseline.yml` – An example CloudFormation template for creating a new IAM administrative user, IAM user group, IAM role, and IAM policy in the account.

Summary

In this post, I have demonstrated the step-by-step process for end-to-end account creation in Organizations as well as how to automate the entire process. I also showed how to move a new account into an OU. This solution should save you some time and help you avoid common issues that tend to crop up in the manual account-creation process. To learn more about the features of Organizations, see the AWS Organizations User Guide. For more information about the APIs used in this post, see the Organizations API Reference.

If you have comments about this blog post, submit them in the "Comments" section below. If you have implementation or troubleshooting questions, start a new thread on the Organizations forum.

– David

Want more AWS Security how-to content, news, and feature announcements? Follow us on Twitter.

TAGS: account creation, AWS Accounts, AWS CloudFormation, AWS Organizations, DevOps automation, organizational unit, OUs, Security Blog

Comments