Implementation Guide

cloudtamer.io Cloud Governance: Financial Management for AWS Control Tower

cloudtamer.io

November 2020
# Table of Contents

Foreword ............................................................................................................................................................. 3

Solution overview and features .......................................................................................................................... 4

Architecture diagram........................................................................................................................................... 5

Account creation from cloudtamer.io workflow ............................................................................................. 5

Account creation from AWS Control Tower workflow .................................................................................... 6

Pre-requisites ...................................................................................................................................................... 6

Deployment and configuration steps .................................................................................................................. 7

Installation of cloudtamer.io ........................................................................................................................... 8

Configuration: integration setup ................................................................................................................... 10

What to expect .............................................................................................................................................. 13

Best practices .................................................................................................................................................... 13

Solution estimated pricing ................................................................................................................................ 13

FAQs ................................................................................................................................................................... 14

Additional resources .......................................................................................................................................... 14

Partner contact information .............................................................................................................................. 14
Foreword

cloudtamer.io is a cloud governance solution built to simplify cloud complexity. We make cloud account management, budget enforcement, and continuous compliance easy. Your organization gets the value of innovating in the cloud without the complexity of cloud management. With cloudtamer.io, you get visibility, assurance, and scalability in one solution and across your entire cloud lifecycle.

The purpose of this AWS Implementation Guide is to enable AWS Marketplace customers to seamlessly activate, deploy, and configure cloudtamer.io Cloud Governance in an AWS Control Tower environment. Customers can take full advantage of the resources pre-configured by AWS Control Tower as part of the initialization.
Solution overview and features

When you migrate to a public cloud provider, you realize some scalability and cost-saving benefits. But as your cloud adoption grows, without an enforceable governance model, you’re left with a virtual “rat’s nest” of accounts and resources that are nearly impossible to manage.

With cloudtamer.io, it’s easy for your technical staff to get access to the resources they need. Your staff gets native access to the console and application programming interfaces (APIs); cloudtamer.io is not a cloud broker. Additionally, cloudtamer.io gives leaders confidence that users can’t exceed established budgets or violate regulatory standards. cloudtamer.io not only reports, it enforces.

With cloudtamer.io, you get:

- Centralized management of all cloud accounts and resources aligned to the organizational hierarchy
- Federated single sign-on (SSO) and multi-factor authentication (MFA) for secure access to the cloud console
- Automated, self-service account creation with native console, command-line interface (CLI), and API access
- Hierarchical budget alignment to projects and organizational units
- Configurable enforcement actions to alert, freeze spending, and terminate cloud resources when project spending thresholds are exceeded
- Spending policies to proactively enforce responsible use of cloud services
- Robust inheritable access policies to restrict use of non-compliant cloud services
- Hierarchical inheritance of compliance policies across organizations
- Reports and remediation to visualize the compliance gap and take automatic or on-demand action

Customers using AWS Control Tower to rapidly onboard internal team members may need more oversight and control over the cloud spend associated with each of their AWS accounts. Customers often want to set limits on each of their projects to ensure they are not overspending their departmental budgets. AWS Control Tower and cloudtamer.io are separate solutions. When an AWS account is created from cloudtamer.io, the account must be enrolled in AWS Control Tower so it can be baseline and set up with proper guardrails. Customers can now purchase cloudtamer.io through the AWS Marketplace, install the software in their AWS account, and then deploy a few CloudFormation templates to auto-enroll newly created accounts in AWS Control Tower. This integration ensures all AWS accounts created through cloudtamer.io are under the governance of AWS Control Tower. Customers can use the granular budget controls within cloudtamer.io to ensure each of the AWS accounts remains within budget. The
integration also ensures that accounts created through AWS Control Tower are automatically added to the cloudtamer.io Account Cache so they are ready to be assigned to a project.

Architecture diagram

When AWS Control Tower provisions an account, it will create an Identity and Access Management (IAM) role called **AWSControlTowerExecution** with a cross-account trust policy that replaces the **OrganizationAccountAccessRole**. The IAM role is typically generated when creating AWS accounts through AWS Organizations. cloudtamer.io will leverage the **AWSControlTowerExecution** role by assuming into it to apply its own IAM role with a cross-account trust policy that is deployed via a CloudFormation template.

The solution will deploy CloudWatch event rules, Lambda functions, and Lambda roles. The solution automation will receive events from AWS Control Tower and AWS Organizations.

![Figure 1 cloudtamer.io Architecture Diagram](image)

**Account creation from cloudtamer.io workflow**

When you create a new AWS account from cloudtamer.io, the software will make an API call to the AWS Organizations service to create a new AWS account. Once the account is created, a **CreateAccount** event will come through CloudTrail and trigger the CloudWatch event rule that forwards the event to a Lambda function. The Lambda function will:

1. Only process the event if the origin of the call is not from AWS Control Tower.
2. Wait for the AWS account creation to finalize with a **SUCCEEDED** status.
3. Assume into the new account using the **OrganizationAccountAccessRole**.
4. Create the **AWSControlTowerExecution** IAM role with a cross-account trust policy for the account where AWS Control Tower is set up.

5. Attach the **AdministratorAccess** IAM policy to the role.

6. In the master payer account, provision a new Service Catalog product with the new account number.

Once the product is provisioned, the AWS account will be available in both cloutamer.io and AWS Control Tower.

**Account creation from AWS Control Tower workflow**

When you create a new AWS account from AWS Control Tower and the account creation completes, a **CreateManagedAccount** event will come through CloudTrail and trigger the CloudWatch event rule that forwards the event to a Lambda function. The Lambda function will:

1. Read the cloutamer.io API access key from a **SecureString** in the Parameter Store of the AWS Systems Manager service.
2. Make a GET request to cloutamer.io to get the Billing Source ID that matches the master payer AWS account number.
3. Make a POST request to cloutamer.io with the account name, account number, and IAM role to use to assume into the AWS account.
4. Create the cloutamer.io cross-account IAM role via a CloudFormation template.
5. Add the AWS account to the cloutamer.io Account Cache.

The AWS account will then be available in both AWS Control Tower and cloutamer.io.

**Pre-requisites**

The solution assumes that you have AWS Control Tower deployed. To get started with AWS Control Tower, check out the [Getting Started guide](#).

Currently, AWS Control Tower only allows one account to be enrolled at a time. If you create an AWS account through cloutamer.io, wait until the account is fully enrolled in AWS Control Tower before creating another account.

There are a few features inside cloutamer.io that may require increases in AWS Service Limits. Those service limits are:

- Number of AWS accounts that can be created via AWS Organizations
- Number of IAM policies per IAM role
The integration with AWS Control Tower requires the paid cloudtamer.io SaaS listing. The cloudtamer.io BYOL listing is not supported for the integration with AWS Control Tower.

Before you implement this solution, we recommend becoming familiar with the following services:

- Amazon Elastic Compute Cloud (EC2)
- Elastic Block Storage (EBS)
- Elastic Load Balancing - Classic (ELB)
- AWS Data Transfer
- AWS Key Management Service (KMS)
- Amazon Aurora Storage
- Amazon Aurora IO
- Amazon Relational Database Service for Aurora MySQL (RDS)
- Simple Storage Service (S3)
- CloudWatch Logs
- AWS Certificate Manager (ACM)

If you are new to AWS, see Getting Started with AWS.

Learn additional information on AWS Marketplace here.

Deployment and configuration steps

Step 1.1: Subscribe to cloudtamer.io – Cloud Governance (SaaS) on AWS Marketplace.

Locate the cloudtamer.io – Cloud Governance (SaaS) product in the AWS Marketplace

Click on the Continue to Subscribe button.
Step 1.2: Guidance on Contract Duration and Renewal

In the new screen, you can configure your contract. Select the **Contract Duration** and set the **Renewal Settings**.

Step 1.3: Select Contract Options

Select the Contract Options to be activated with your contract.

Step 1.4: Create the Contract and Pay

Once you have configured your contract, click on the **Create Contract** button.

You will be prompted to confirm the contract. If you agree to the pricing, select the **Pay Now** button.

Step 1.5: Set up Account

You will be redirected to our registration page. Complete the details to gain access to our Support Center and proceed with the deployment.

Installation of cl oudtamer.io

Step 2.1: Log into the Support Center

You can access our Support Center via: [https://support.cloudtamer.io](https://support.cloudtamer.io). Your email has been whitelisted so you can now create an account and login.
Step 2.2: Access the AWS Deployment Guide

Access the AWS Deployment Guide here.

Step 2.3: Log in to your AWS Account Designated for cloudtamer.io

Log in to your AWS account to deploy the CloudFormation templates into your AWS account. We recommend putting cloudtamer.io in a separate AWS account from the AWS Control Tower account to separate workloads, but they can be in the same AWS account if desired.

Step 2.4: Set up your Virtual Private Cloud (VPC)

Set up your AWS VPC to support the cloudtamer.io installation. You need at least two subnets in different availability zones to support the installation. You can read more about the architecture and the requirements in the deployment guide.

Step 2.5: Launch the Database

Scroll to the bottom of the deployment guide to download the CloudFormation template (control-tower-account.json) to set up the RDS Aurora database. Navigate to the CloudFormation service and launch the stack.

Step 2.6: Launch the Application

Download the CloudFormation template (cloudtamer-app-X.XX.X.json) for the application. Navigate to the CloudFormation service and launch the stack.

Step 2.7: Initialize and Configure the Application
Now that cloudtamer.io is up and running, refer to the following guides to configure the application:

- Tour: Homepage and Important Concepts
- AWS Quickstart Guide
- AWS RDS Backup Guide

Configuration: integration setup

**Step 3.1: Set up the CloudFormation for AWS Control Tower Integration**

To allow accounts created in cloudtamer.io to enroll with AWS Control Tower, download the CloudFormation template (control-tower-account.json). Run the CloudFormation template in the AWS account where AWS Control Tower is set up (the payer account). You will need the following information:

- Service Catalog portfolio ID for AWS Control Tower Account Factory
  o You can get this from Service Catalog in the account where AWS Control Tower is set up.
- Service Catalog product ID for the AWS Control Tower Account Factory
  o You can get this from Service Catalog in the account where AWS Control Tower is set up.
- Service Catalog product version for the AWS Control Tower Account Factory
  o You can get this from Service Catalog in the account where AWS Control Tower is set up.
- Name of the OU in which to move new AWS accounts when they are created
  o You should choose an OU from AWS Organizations.
- First name of the SSO user for AWS Control Tower
  o This is the placeholder name for the SSO user for AWS Control Tower.
- Last name of the SSO user for AWS Control Tower
  o This is the placeholder name for the SSO user for AWS Control Tower.
- Service role name that is accessible from the payer
  o We recommend using the `OrganizationAccountAccessRole`.
- cloudtamer.io AWS account number
  o This is the AWS account number where cloudtamer.io is installed. It can be the same account as the AWS Control Tower account.
- S3 bucket where the Lambda code exists
  o We recommend leaving this as the default unless you want to use your own Lambda code.
- S3 key where the Lambda code exists
  o We recommend leaving this as the default unless you want to use your own Lambda code.

The source code for the Lambda is available [here](#) if you wish to modify it.
Step 3.2: Create a cloudtamer.io Service User and API key

To allow the Lambda to send an API request to cloudtamer.io, you should:

1. Log in to cloudtamer.io.
2. Click Settings in the left navigation menu.
3. Click the App API Access menu item.
4. Turn on App API Key Generation.
5. Click the Update App API Access Settings button.
6. Click Users in the left navigation menu.
7. Click the plus button at the top to create a new user.
   a. Name the user so it is clear it is a service user.
   b. Save the password.
8. Click Settings -> Permissions in the left navigation menu.
9. Click the Roles tab.
10. Create a new role called: Control Tower Service User.
11. Click the Permission Schemes tab.
12. Edit the Global Permissions Scheme by clicking on the ellipsis menu.
13. Next to the Manage AWS Accounts, Manage Own API Keys, and Manage Billing Sources permissions, select the role: Control Tower Service User.
14. Click the Update Permission Scheme button at the bottom of the page.
15. Click the Global Permissions tab.
16. Next to the Control Tower Service User role, select the service user created above from the USERS dropdown.
17. Click the Save button at the bottom of the page.
18. Click on your name in the top navigation bar -> Sign Out.
19. Login with the service user you created above.
20. Click on your name in the top navigation bar -> App API Keys.
21. Click Create App API Key.
22. Name the key so it’s descriptive.
23. Save the API key.
24. Click on your name in the top navigation bar -> Sign Out.

Step 3.3: Add the API key to Parameter Store

1. Log in to your AWS account where cloudtamer.io is installed.
2. Navigate to the AWS Systems Manager service and then to the Parameter Store.
3. Click **Create parameter** and enter in a **Name**, **Description**, and select the type: **SecureString**.

4. Paste in the API key from above into the **Value** field.

5. Click **Create parameter**.

6. Save the name value for the next step.

**Step 3.4: Set up the CloudFormation for cloudtamer.io Integration**

To allow accounts created in AWS Control Tower to be registered in the cloudtamer.io Account Cache, download the CloudFormation template (**cloudtamer-io-account.json**). Run the CloudFormation template in the AWS account where cloudtamer.io is installed.

You will need the following pieces of information:

- Management / Payer AWS Account Number
  - This is the AWS account number where AWS Control Tower is set up.
- cloudtamer.io URL
  - This is the URL to cloudtamer.io so the Lambda function knows how to access the APIs.
- API Key
  - This is the cloudtamer.io API key for a user that has the Global Manage Accounts permission.
- Service Role
  - This is the IAM service role that cloudtamer.io can use to assume into the accounts from AWS Organizations. We recommend you use the AWS Control Tower role: **AWSControlTowerExecution**.
- S3 bucket where the Lambda code exists
  - We recommend leaving this as the default unless you want to use your own Lambda code.
- S3 key where the Lambda code exists
  - We recommend leaving this as the default unless you want to use your own Lambda code.
- Security Group
  - This should be the Security Group that will be attached to the Lambda.
- Private Subnets
  - These should be subnets where the Lambda will be created. They should be in the same VPC as cloudtamer.io if possible.

The source code for the Lambda function is available [here](#) if you wish to modify it.

**Step 3.5: Ensure the Lambda can communicate with cloudtamer.io**
If the cloudtamer.io user-facing load balancer is **public-facing**, ensure the Lambda can get out to the internet via a network address translation (NAT). Add the NAT IP(s) to the security group attached to the user-facing load balancer.

If the cloudtamer.io user-facing load balancer is **internal**, add the IP(s) of the Lambda function to the security group attached to the user-facing load balancer. It’s recommended to use the VPC IP range so if the IPs of the Lambda change, they will still be able to access the cloudtamer.io API.

**What to Expect**

**Step 4.1: Verify the integration**

To verify the integration, either create an AWS account from cloudtamer.io or create an AWS account from AWS Control Tower – depending on which method you want to use in production. The new AWS account should display as enrolled in AWS Control Tower and in the Account Cache of cloudtamer.io. Since the integrations rely on Lambda, you can view the logs in either the AWS Control Tower account or the cloudtamer.io account in CloudWatch. This is an easy way to verify if there are any problems with the integration. You can also check in Service Catalog in the AWS Control Tower account to ensure the products provisioned for the AWS accounts succeeded. If they did not, they will show an error. The process is documented in the FAQs.

**Step 4.2: How to use**

No additional steps are needed to take advantage of the integration. The CloudFormation templates will automatically trigger based on account creation events by AWS Organizations or AWS Control Tower.

**Best practices**

- Securely rotate your API key. It’s stored securely in Parameter Store, but you should rotate it on a scheduled basis to reduce the chances of it being compromised.
- Ensure you manually backup your cloudtamer.io database so you can recover if necessary. Learn how in our [RDS Backup Guide](#).

**Solution estimated pricing**

For cloudtamer.io, a typical installation ranges from $400 (single EC2 node, single RDS node) to $800 (3 EC2 nodes, 2 RDS nodes) per month. The integration itself should be less than $1 a month because it utilizes serverless Lambda, which is very inexpensive to run. You can view the AWS Control Tower pricing [here](#).
FAQs

How do I enroll existing AWS accounts in AWS Control Tower?

If you already have AWS accounts that were created before the AWS Control Tower service was enabled, review this article to manually enroll each of them in AWS Control Tower: Enrolling an existing AWS account in AWS Control Tower.

How do I troubleshoot an AWS Control Tower account enrollment?

AWS Control Tower requires the Account Factory Service Catalog product to be provisioned for each AWS-managed account. To view the status of the Service Catalog product that was deployed:

1. Log in to the AWS master payer account where AWS Control Tower is enabled.
2. Navigate to the AWS Service Catalog service.
3. Click Provisioned products on the left navigation menu.
4. If needed, change the Filter by drop-down to Accounts.
5. View the product status under the Status column.

In the case of a failure, cancel the failed provisioned product and then relaunch the Service Catalog product for the new account with the same information.

Additional Resources

- cloudtamer.io Resources
- cloudtamer.io FAQs

Partner Contact Information

cloUdtamer.io offers both standard and premium support options. Customers can access our cloudtamer.io Support Center at https://support.cloUdtamer.io to view documentation and create/manage support tickets. To purchase enterprise support, contact us: https://www.cloUdtamer.io/contact-us/.