Ansible Tower on the AWS Cloud

Quick Start Reference Deployment

Tony Vattathil

April 2016

This guide is also available in HTML format at http://aws.amazon.com/quickstart/



Contents

[Overview 3](#_Toc448074451)

[Ansible Tower on AWS 3](#_Toc448074452)

[Quick Links 4](#_Toc448074453)

[Cost and Licenses 4](#_Toc448074454)

[Architecture 5](#_Toc448074455)

[Implementation Details 6](#_Toc448074456)

[AWS Services 6](#_Toc448074457)

[Ansible Tower Installation 7](#_Toc448074458)

[Automated Deployment 8](#_Toc448074459)

[What we’ll Cover 8](#_Toc448074460)

[Step 1. Prepare an AWS Account 9](#_Toc448074461)

[Step 2. Launch the Ansible Tower Stack 12](#_Toc448074462)

[Step 3. AWS Credential for use with Ansible Tower 13](#_Toc448074463)

[Step 4. Licensing Ansible Tower 16](#_Toc448074464)

[Step 5. Configuring Ansible Tower with EC2 integration 20](#_Toc448074465)

[Step 4. Discover and Manage EC2 instance in Ansible Tower 21](#_Toc448074466)

[Step 6. Adding more managed instances 26](#_Toc448074467)

[Troubleshooting 27](#_Toc448074468)

[Security 28](#_Toc448074469)

[Additional Resources 28](#_Toc448074470)

[Send Us Feedback 28](#_Toc448074471)

[Additional Resources 29](#_Toc448074472)

[Document Revisions 29](#_Toc448074473)

About This Guide ``

This Quick Start reference deployment guide discusses the steps for deploying and testing Ansible Tower. It also provides links for viewing and launching [AWS CloudFormation](http://aws.amazon.com/cloudformation/) templates that automates the Ansible Tower deployment and an create Ansible aware Ec2 instances. The walkthrough details how you can configure Ansible Tower to pull Ec2 instances and their metadata into the Tower inventory with a click!

The guide is for IT infrastructure architects, administrators, and DevOps professionals who are planning to implement Ansible Tower to manage on the AWS compute resources.

[Quick Starts](http://aws.amazon.com/quickstart/) are automated reference deployments for key workloads on the AWS cloud. Each Quick Start launches, configures, and runs the AWS compute, network, storage, and other services required to deploy a specific workload on AWS, using AWS best practices for and security.

# Overview

## Ansible Tower on AWS

Ansible is designed to a simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs.

Ansible is designed for multi-tier deployments from its inception, Ansible models your IT infrastructure by describing how all of your systems inter-relate, rather than just managing one system at a time.

A key advantage to Ansible over other automation engines is that it uses no agents and no additional custom security infrastructure, so it's easy to deploy - and most importantly, it uses a very simple language (YAML, in the form of Ansible Playbooks) that allow you to describe your automation jobs in a way that approaches plain English. Ansible works by connecting to your nodes and pushing out small programs, called "Ansible Modules" to them. These programs are written to be resource models of the desired state of the system. Ansible then executes these modules (over SSH by default), and removes them when finished.

## Quick Links

**Launch   
Quick Start**

If you have an AWS account and you’re already familiar with AWS and Ansible, you can use the **Launch Quick Start** button to build the architecture shown in [Figure 1](#_Architecture_Overview). The deployment takes approximately 20 minutes. If you’re new to AWS or Ansible Tower, please review the implementation details and   
follow the [step-by-step instructions](#_Automated_Deployment_and) provided later in this guide   
to launch the Quick Start.

**View template**

If you want to take a look under the covers, you can choose   
**View template** to see the AWS CloudFormation template that automates this deployment. The default configuration deploys three servers that use the t2.medium instance type by default,   
but you can customize the template if you’d like.

## Cost and Licenses

You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using the Quick Start. As of the date of publication, the cost for using the Quick Start with default settings is approximately $0.18 an hour. Prices are subject to change. See the pricing pages for each AWS service you will be using or the [AWS Simple Monthly Calculator](http://calculator.s3.amazonaws.com/index.html) for full details.

This QuickStart deploy Ansible Tower on AWS cloud and along with Ansible Linux Ansible Client. Ansible and Ansible Tower have separate licensing models.

Ansible Tower Licenses

Ansible Tower is subject to the Ansible Software Subscription and Services Agreement located at http://www.ansible.com/subscription-agreement. Ansible Tower is a proprietary product offered by Ansible, Inc. and its use is not intended to prohibit the rights under any open source license.

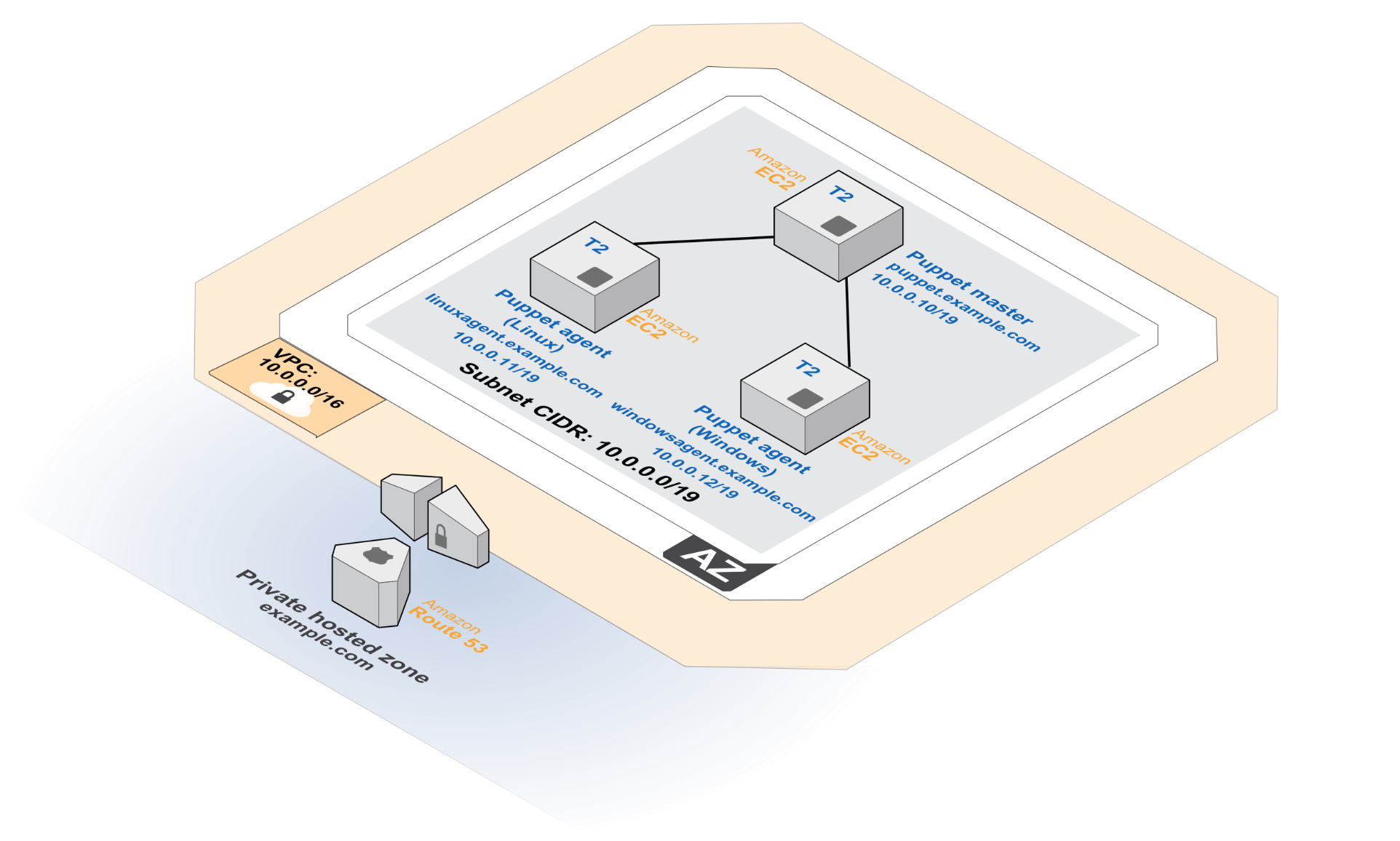
While a license is required for Tower to run, there is no fee for managing up to 10 hosts This document provides a walk though of how to acquire a 10 host license

Ansible Licenses

Ansible is licensed under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License)

# Architecture

Deploying this Quick Start with the **default parameters** builds the following environment in the AWS cloud.

Figure 1: Quick Start Ansible Tower on AWS

This Quick Start deploys the resources shown in Figure 1 and uses them as follows:

* An Amazon VPC is created in the region you choose when you launch the stack. A single, public VPC subnet is created in a random Availability Zone.
* One Centos client is deployed as a client into the VPC subnet. RHEL Clients can also be attached using the same procedure detailed in this document.

# Implementation Details

This section discusses the implementation of this Quick Start and explains the considerations for installing and configuring Ansible Tower on AWS. Note that some steps are manual and others are automated for you by this Quick Start.

## AWS Services

The core AWS components used by this Quick Start include the following AWS services. (If you are new to AWS, see the [Getting Started section](http://docs.aws.amazon.com/gettingstarted/latest/awsgsg-intro/intro.html) of the AWS documentation.)

* [Amazon VPC](http://aws.amazon.com/documentation/vpc/) – The Amazon Virtual Private Cloud (Amazon VPC) service lets you provision a private, isolated section of the AWS cloud where you can launch AWS services and other resources in a virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways.
* [Amazon EC2](http://aws.amazon.com/documentation/ec2/) – The Amazon Elastic Compute Cloud (Amazon EC2) service enables you to launch virtual machine instances with a variety of operating systems. You can choose from existing Amazon Machine Images (AMIs) or import your own virtual machine images.

## Ansible Tower Installation

This Quick Start deploys Ansible Tower on an EC2 instance that is running Centos7. The installation is automated with a user data script that executes when the instance is launched via AWS CloudFormation. Ansible tower installation files are installed directly from Ansible’s release server

In addition to installing the Ansible Tower a Linux Client is also deployed into the VPC. The Client is tagged with the key *Tower* later in this document we will walk you through how you can pull in Ec2 resources so they can be managed by Ansible Tower.

# Automated Deployment

The AWS CloudFormation template provided with this Quick Start bootstraps the AWS infrastructure and automates the deployment of Ansible Tower on the AWS cloud from scratch. Follow the step-by-step instructions in this section to set up your AWS account, customize the template, and deploy the software into your account.

## What we’ll Cover

The procedure for deploying Ansible Tower on AWS consists of the following steps. For detailed instructions, follow the links for each step.

[Prerequisites](#_Prerequisites)

* Set up and enable name resolution via DNS on your workstation
* Make sure you can use Secure Shell (SSH) for remote connections.

Step 1. Prepare an AWS account

* Sign up for an AWS account, if you don’t already have one.
* Choose the region where you want to deploy the stack on AWS.
* Create a key pair in the region.
* Review account limits for Amazon EC2 instances, and request a limit increase, if needed.

Step 3. Launch the stack

* Launch the AWS CloudFormation template into your AWS account.
* Enter a value for **All parameters** the require input
* Review the other template parameters, and customize their values if necessary.

Step 3 Get Ansible Tower License

* Connect to your Ansible Tower via web browser and follow the steps to license Ansible Tower
* Follow the walkthrough to integrate AWS compute resources into the Tower inventory

## Step 1. Prepare an AWS Account

1. If you don’t already have an AWS account, create one at <http://aws.amazon.com> by following the on-screen instructions. Part of the sign-up process involves receiving a phone call and entering a PIN using the phone keypad.
2. Use the region selector in the navigation bar to choose the Amazon EC2 region where you want to deploy Ansible Tower on AWS.

Amazon EC2 locations are composed of [*Regions* and *Availability Zones*](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html). Regions are dispersed and located in separate geographic areas.



Figure 2: Choosing an Amazon EC2 Region

**Tip** Consider choosing a region closest to your data center or corporate network to reduce network latency between systems running on AWS and the systems and users on your corporate network.

1. Create a [key pair](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html) in your preferred region. To do this, in the navigation pane of the Amazon EC2 console, choose **Key Pairs**, **Create Key Pair**, type a name, and then choose **Create**.

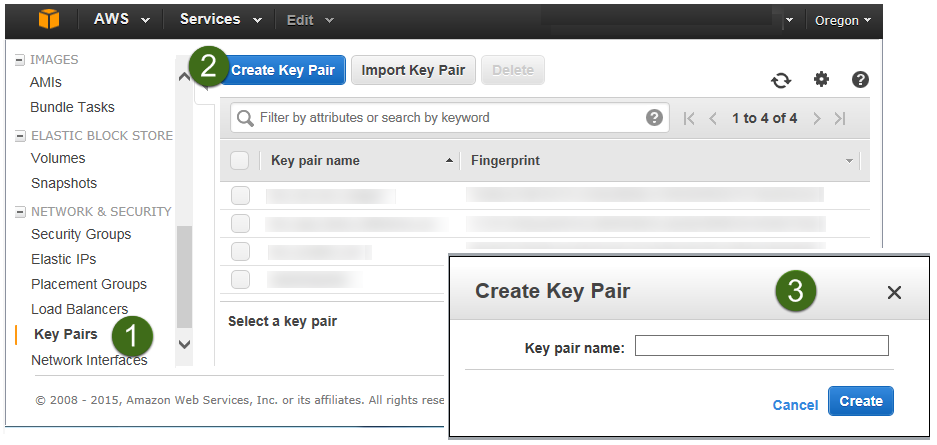


Figure 3: Creating a Key Pair

Amazon EC2 uses public-key cryptography to encrypt and decrypt login information. To be able to log in to your instances, you must create a key pair. On Linux, we use the key pair to authenticate SSH login.

1. If you would like to use a different instance type, [request a service limit increase](https://console.aws.amazon.com/support/home#/case/create?issueType=service-limit-increase&limitType=service-code-) for the Amazon EC2 **t2.medium** instance type. To do this, in the AWS Support Center, choose **Create Case**, **Service Limit Increase**, **EC2 instances**, and then complete the fields in the limit increase form. The current default limit for this instance type is 20 instances.

You might need to request an increase if you already have an existing deployment that uses this instance type, and you think you might exceed the default limit with this reference deployment. It might take a few days for the new service limit to become effective. To learn more, see [Amazon EC2 Service Limits](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-resource-limits.html) in the AWS documentation.

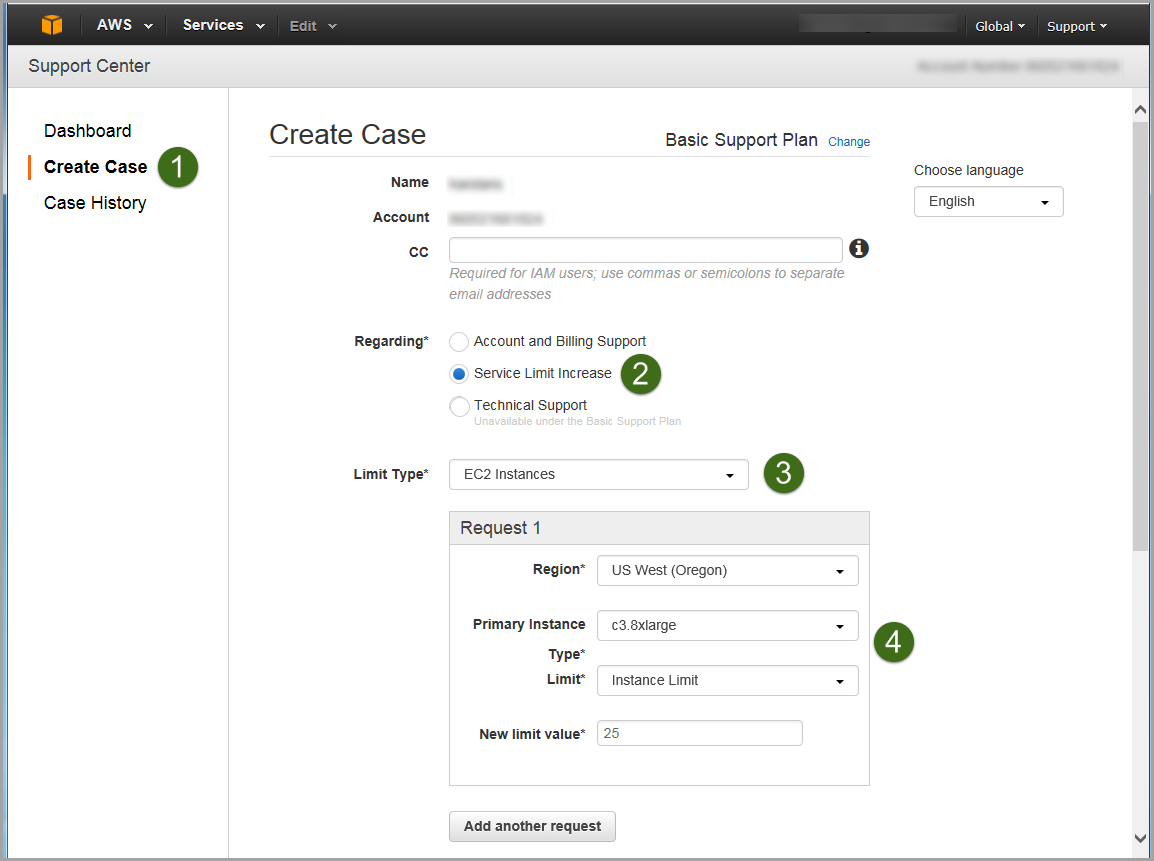


Figure 4: Requesting a Service Limit Increase

## Step 2. Launch the Ansible Tower Stack

The automated AWS CloudFormation template provided with this Quick Start deploys Ansible Tower into an Amazon VPC. Please make sure that you’ve completed the previous steps before launching the stack.

**Launch   
Quick Start**

1. Launch the AWS CloudFormation template into your AWS account.

The template is launched in the US West (Oregon) Region by default. You can change the region by using the region selector in the navigation bar.

This stack takes approximately 20 minutes to create.

**Note** You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no **additional cost** for using this Quick Start.

You can also [download the template](https://s3.amazonaws.com/quickstart-reference/puppet/latest/templates/Puppet-QuickStart.template) to use it as a starting point for your own implementation.

1. On the **Select Template** page, keep the default URL for the AWS CloudFormation template, and then choose **Next**.
2. On the **Specify Details** page, review the parameters for the template. These are described in the following table.

IMPORTANT: Take a note of the password you provide for the **AnsibleAdminPassword** parameter you will need in the steps to follow

Provide a value for the **KeyPairName** parameter. This parameter require your input. For all other parameters, the template provides default settings that you can customize.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter label | Parameter name | Default | Description |
| Select a key pair | **KeyPairName** | *Requires input* | Public/private key pair, which enables you to connect securely to your instance after it launches. When you created an AWS account, this is the key pair you created in your preferred region. |
| Set you Tower Admin Password | **AnsibleAdminPassword** | *Requires input* | Must be at least 8 characters containing letters and (minimum 1 capital letter)  (e.g., An$ibl3ChgMe). |
| Specify a DataBaseAdminPassword | **DatabaseAdminPassword** | *Requires input* | All dB resources will use this password (redis/postgress) |
| Source IP for remote access | **RemoteAdminCIDR** | 0.0.0.0/0 | CIDR block or IP address for SSH (e.g., 1.1.1.1/32). |
| CIDR range for your VPC | **VPCCIDR** | 10.0.0.0/16 | CIDR block for the VPC. |
| CIDR range for the subnet in your VPC | **SubnetCIDR** | 10.0.0.0/19 | CIDR block for the subnet. |
| IP address for the Ansible Tower | **AnsibleTowerIP** | 10.0.0.10 | This is the private IP of your Ansible tower |
| IP address for the Linux Client. (bare Linux instance) | **AnsibleClientLinuxIP** | 10.0.0.11 | IP address of a Linux Instance that can be managed by Ansible |

**Note** You can also [download the template](https://s3.amazonaws.com/quickstart-reference/ansible/latest/templates/Ansible-Tower-QuickStart.template) and edit it to create your own parameters based on your specific deployment scenario.

1. On the **Options** page, you can [specify tags](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-resource-tags.html) (key-value pairs) for resources in your stack and [set additional options](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-console-add-tags.html). When you’re done, choose **Next**.
2. On the **Review** page, review and confirm the settings.
3. Choose **Create** to deploy the stack.
4. Monitor the status of the stack. When the status is **CREATE\_COMPLETE**, your Ansible Tower is ready.

## Step 3. AWS Credential for use with Ansible Tower

**As part of this setup I created a new user called *ansibleadm*. This is an optional step you can use an existing user if you wish - given that user has the proper privileges.**

**To create a user, navigate to this https://console.aws.amazon.com/iam/home?region=us-west-1#users > Log in and Click Create User then Click Create on the bottom right corner**

****

**Click Show User Security Credentials copy or download them to your workstation.**

****

Click on ansibleadm to assign IAM roles to this user



I gave this user PowerUser IAM privileges. If you wish you can lock it down further.

For More info on IAM roles please see this link

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html>

## Step 4. Licensing Ansible Tower

1. Navigate to the IP or hostname of your Ansible tower.

Note: The ipaddress can be found in the output section of the Ec2Stack or in the Ec2 console

1. You will need to trust the self signed cert. (we recommend that you add your cert to Ansible Tower for higher security). See the user guide in the appendix for more info.
2. Click on Advanced then click proceed to reach the Ansible Web UI 
3. Provide the **AnsibleAdminPassword** you entered in the **Specify Detail** page

The default **username** is: **admin**

The **password** is :<the password you set during launch>



If you forgot your password it will be in the *tower\_setup\_conf.yml* which can be found here in your root home dir.

The following command will display the admin password

**[centos@ip-10-0-0-10 ~]$ sudo grep admin\_password /root/ansible-tower-setup-bundle-2.4.4-1.el7/tower\_setup\_conf.yml**

**WARNING: *tower\_setup\_conf.yml* file contains admin user credential we recommend that you delete this folder after you retrieve your credentials**

If you need instructions on how to ssh into your ec2-instance, **please see the section on accessing ec2 instances**

Note: On CentOS the username to connect vi ssh is **centos** not ec2-user

example command(CentOS): ssh –i your.pem centos@ip-or-hostname

example command(RHEL): ssh –i your.pem centos@ip-or-hostname

1. Upon successful login you will reach the screen show below. Click the red box to Get a Free Tower Trial License. This action will open a new window navigate to the new window and select **10-NODE BASIC TOWER - FREE TRIAL.** Enter the information requested and Click Submit. Check your email for trial license info



1. Copy and Paste the license file into the UI and click Submit



1. Click on the checkbox on the bottom of the screen if you agree with the End User License Agreement. Then Submit.

## Step 5. Configuring Ansible Tower with EC2 integration

In this step we are going to setup the basics needed to configure EC2 integration. For the full Tower configuration procedures and information on how you can setup Organizations, Teams and Projects please see the Ansible user guide at this link <http://docs.ansible.com/ansible-tower/2.2.0/html/userguide/>

1. Click on the wrench and screwdriver icon to navigate to the setup UI



1. Click on Credentials

Then click on the blue plus 

* Provide a **Name** and description
* Select admin from **User that owns this credential**
* Click Type and **Choose Amazon Web Services**
* Enter you **Access Key** and **Secret Key**
* Click **Save**

****

## Step 4. Discover and Manage EC2 instance in Ansible Tower

1. Create an Inventory

* Click on Inventories in the main menu
* Then click the blue plus sign to bring up the Create Inventory dialog 
* Provide a Name for the Default Organization. Leave the Variables section as it and click Save to create an inventory



1. Next Create a *Group*

* Click the Blue plus sign in the Group Section



* Provide a Name and Description
* You can leave the Variables section as is.



Click on the second tab labeled **Source**

* From the dropdown list Click **Amazon EC2** as the **Source**
* Click on the magnifying glass under **Cloud Credentials** and choose **AWS Credentials**
* Under **Region** Choose **All**
* Under **Instance Filter** type tag-key=Tower
* Leave **Only Group By tag-key=Tower** blank
* Leave **Source Variables** and **Update Options** as is
* Click **Save**



Click on bi directional arrows to start sync process. Once the Sync successfully complete you will see a *green cloud* and a *green circle* indication the sync with Amazon is complete



In the **Hosts** section you will see two hosts.

* Your Ansible Tower and a bare Linux instance both managed by Ansible



You can click on the host to see the discovery info



## Step 6. Adding more managed instances



Since the Cloudformation template initially created the Ec2 instance and tagged it with the key *Tower* **Using the Launch More Like This** feature will allow Ansible Tower to discover the new instance when a sync is requested.

To scan for new instances click on the bidirectional arrows 

Here you can also see the second host (I edited the hostname and gave the host friendly names)



# Troubleshooting

If you encounter a CREATE\_FAILED error, we recommend that you relaunch the template with **Rollback on failure** set to **No**. (This setting is under **Advanced** in the AWS CloudFormation console, **Options** page.) With this setting, the stack’s state will be retained and the instance will be left running, so you can troubleshoot the issue.

**Important** When you set **Rollback on failure** to **No**, you’ll continue to incur AWS charges for this stack. Please make sure to delete the stack when you’ve finished troubleshooting.

For additional information, see [Troubleshooting AWS CloudFormation](http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/troubleshooting.html) on the AWS website. If the problem you encounter isn’t covered on that page above, please visit the [AWS Support Center](https://console.aws.amazon.com/support/)

# Security

A security group acts as a firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time. The new rules are automatically applied to all instances that are associated with the security group.

The security groups created and assigned to the individual instances as part of this solution are restricted as much as possible while allowing access to the various functions needed by Ansible Tower. We recommend that you review security groups and further restrict access as needed once Ansible is up and running.

# Additional Resources

**AWS services**

* Amazon EC2  
  <http://aws.amazon.com/documentation/ec2/>
* AWS CloudFormation  
  <http://aws.amazon.com/documentation/cloudformation/>
* Amazon VPC  
  <http://aws.amazon.com/documentation/vpc/>

**Ansible resources**

* Ansible Tower User Guide
* <http://docs.ansible.com/ansible-tower/2.2.0/html/userguide/>

# Send Us Feedback

We welcome your questions and comments. Please post your feedback on the [AWS Quick Start Discussion Forum](https://forums.aws.amazon.com/forum.jspa?forumID=178).

# Document Revisions

|  |  |  |
| --- | --- | --- |
| Date | Change | In sections |
| April 2016 | Initial publication (Draft) | * All |