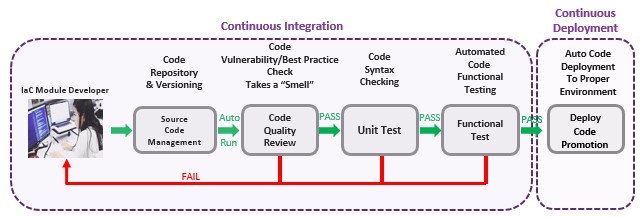
Background

**Traditional scripted automation has lead us to a silo-influence IT automation model with little consistency and near zero collaboration.**

These problems mentioned above are largely solved with an Infrastructure as Code (IaC) implementation. Let’s switch gears and talk about why we believe this is the case.

IaC leverages many of the practices that an application developer uses to keep their work life in order. Development of Infrastructure code is governed by a construct called a CI/CD pipeline. This pipeline is used to develop the code that in turn delivers the IT service it is intended to. This software-based pipeline includes Source Code Management (SCM), Security, Best Practice and Syntax Checking, Unit Testing, Functional Testing, and Automated Code Development. Below is an example of this concept:



In the IaC model, a code (module) developer creates reusable code with the intent to deliver a basic and specific IT service. This code is the authoritative source for that IT delivery service. When they complete the code, they check it into a public SCM solution repository which is the entry door to the CI/CD pipeline. The following steps happen automatically and in order after the code check in.

1. **Code Quality Review** – The pipeline automation engine recognizes a new version of the code; instructing the Code Quality Review technology to run. This “code smell” step examines the code for vulnerabilities and checks for coding best practices. This is typically referred to as a “Smell”. If this step is a PASS, the pipeline continues to the next step. If it is a “FAIL”, the developer must fix the issue(s) and resubmit a new version to the repository.
2. **Unit Test** – This step in the pipeline examines code for proper syntax. If this step is a PASS, the pipeline continues to the next step. If it is a “FAIL”, the developer must fix the issue(s) and resubmit a new version to the repository.
3. **Functional Test** – This step allows the developer to create automated testing so that the code function is vetted out prior to deployment. If this step is a PASS, the pipeline continues to the next step. If it is a “FAIL”, the developer must fix the issue(s) and resubmit a new version to the repository.
4. **Auto-Code Deployment** – Based upon the technology the code will be deployed to, this step will follow a strict pre-defined deployment chain. Each step in the deployment process is purposefully approved after testing is completed. The Code Asset Management area is the only department that can promote code to Production.
5. Once code is in Production, it can be leveraged by any engineering area that needs to use it for IT Service Delivery.

In conclusion, we identified some IT automation scripting traps that organizations have fallen into over the years:

* Hidden/lost scripts
* Lack of script versioning
* Single Author
* IT Silo automation/scripting
* No code governance
* No coding guidelines
* Lack of deployment governance
* Lack of formal code promotion process
* Private scripts
* Lost Intellectual property
* Lack of code reuse

By adopting an Infrastructure as Code culture and framework, we largely solve the problems above. We create an empowered organization by allowing authorized IT service delivery folks the ability to improve upon the code someone else has written. Responsibility for automated IT service delivery is right placed, but it is not hidden from other’s view. Code by nature is public, therefore IT service delivery is shared amongst all of IT.

Pre-requisites

1. A user needs to obtain a GitHub license.
   * In AETH we lock down Jenkins jobs to people who have GitHub requested through the Aetna Appstore
2. A user needs to have an IDE configured
   * IDE’s include VSCode, Geppetto, etc.

# Getting Access

Access to Ansible Tower is controlled by Active Directory. Anyone with an active account in Active Directory is able to access Ansible Tower. The URLs for Tower are:

| **Environment** | **URL** |
| --- | --- |
| AETH | [https://tower.aetna.com](https://tower.aetna.com/) |
| HEH | [https://tower.healthehost.com](https://tower.healthehost.com/) |

Note: Please use firefox or Google Chrome for access to Tower. The Aetna build of Internet Explorer does not work well for rendering screens in Tower.

The login screen for Tower will look like this: 

On this screen enter your Active Directory username and password for the appropriate environment.

Once logged in you may or may not be able to access assets within Tower. This is due to a feature in Ansible Tower called [Roles Based Access Control](https://docs.ansible.com/ansible-tower/3.2.4/html/userguide/security.html#rbac-ug) (aka RBAC). Each asset (a credential, a job template, a workflow, etc) has roles associated with it which control who can see, use or edit the asset. Access to the various roles are control by Ansible Tower [teams](https://docs.ansible.com/ansible-tower/latest/html/userguide/teams.html). Teams are collections of Ansible Tower [users](https://docs.ansible.com/ansible-tower/latest/html/userguide/users.html).

The first time you authenticate to Tower with your Active Directory credentials a user will automatically be created for you in Ansible Tower. Your user ID will be the same as your Active Directory ID and your email address will also be imported and associated with your user. In addition, if you are a member of a group in Active Directory and that group has team membership in Tower, your Tower user will automatically be associated to the team in Tower.

Each time you authenticate to Tower your team association will be adjusted to match your Active Directory groups so your access to assets in Tower is dynamic.

In addition, there is a team called everyone which anyone who logs into Tower will be granted access to. There are a limited number of items which are exposed to everyone.

../../../images/ansible/logout.png To log out of Ansible Tower, click on the logout (power) button in the upper left corner of the Ansible Tower screen

# Setting up a Tower Project

The setup of a [project](https://docs.ansible.com/ansible-tower/latest/html/userguide/projects.html) in Tower is an automated process and is completely controlled by the [Pipeline build process](https://github.aetna.com/pages/IaC/Docs/iacservices/gettingstarted/index.html#step-3-run-pipeline-creation-scripts).

During the pipeline creation process a Tower project will be setup with a name matching your GitHub project in the format <GitHub Organization>/<GitHub Project>. For example, if your Organization was WebEngineering and your project was ansible-websphere, the pipeline would create a Tower project called WebEngineering/ansible-websphere.

Note: This project will only be setup in the lowest environment of Ansible Tower. As you migrate your code through the pipeline that migration will add the project to further Tower environments.

The pipeline will also make sure that the Active Directory groups you specified for access to the GitHub project are also created in Ansible Tower as [teams](https://docs.ansible.com/ansible-tower/latest/html/userguide/teams.html) and that those teams have read access to your project.

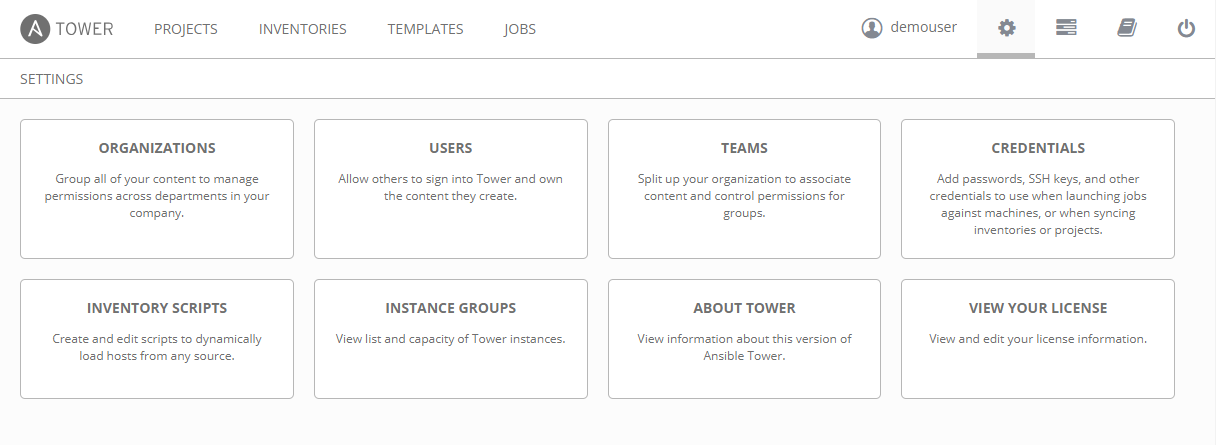
Once your project is setup by the pipeline process it does not need to be altered in any way. No teams will need permissions to your project other than the teams that will be creating job templates from the files in your project. This is why you only have the view [role](https://docs.ansible.com/ansible-tower/latest/html/userguide/security.html#rbac-ug) on your project. If, for any reason, your project needs to be altered please contact [Ansible Support](mailto://AnsibleSupport@AETNA.com)

Setting up Credentials

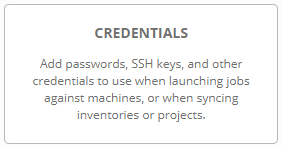
[Credentials](https://docs.ansible.com/ansible-tower/latest/html/userguide/credentials.html) allow Ansible Tower to access resources such as: machines, network devices, vault files, cloud providers, etc. Credentials are stored securely in Ansible Tower an *no one* will be able to access the secrets (passwords, ssh keys, etc) once they are entered into Tower (not even the owner of a credential).

Add a credential

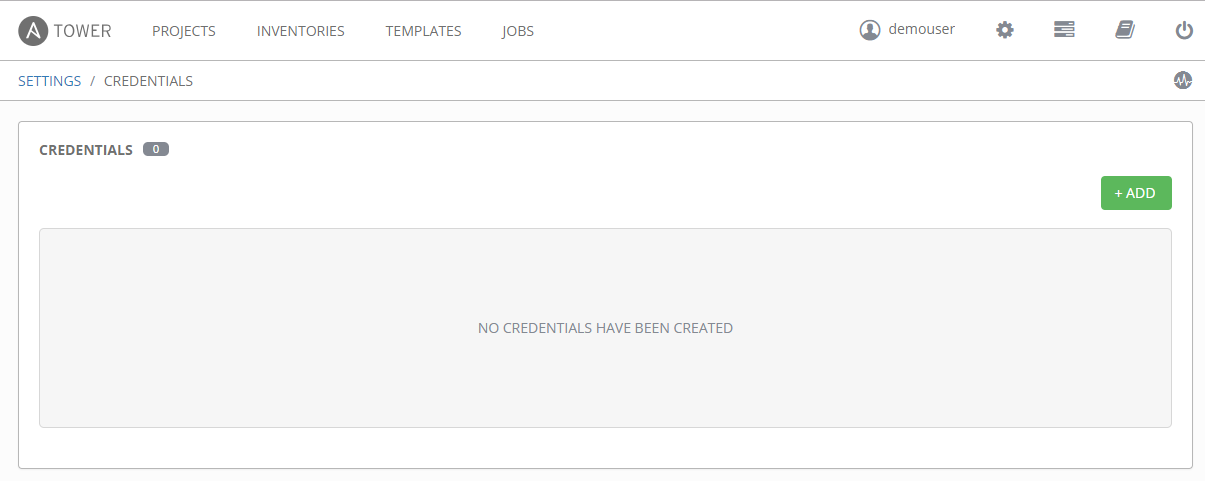
../../../images/ansible/settings.png To build a credential click on the Settings (gear) icon located in the upper right screen of Tower. This will take you to the settings screen:



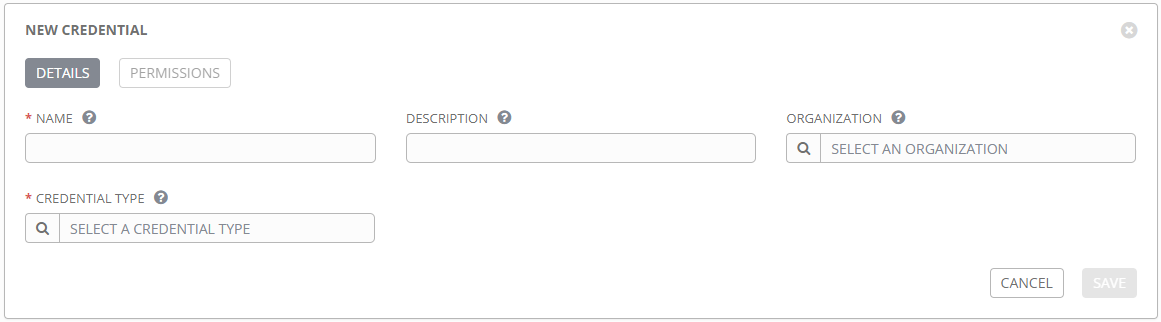
On this screen click on the "Credentials" box:



On the credentials screen you may or may not see some existing credentials:



../../../images/ansible/add.pngClick on the green Add button on the right hand side. This will open a new section above the list of credentials for adding a new credential:



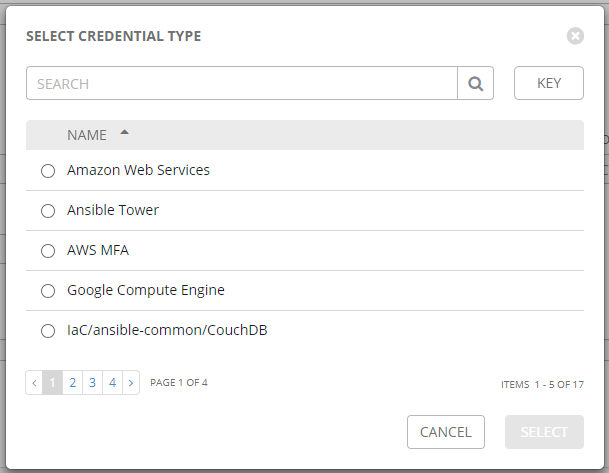
On this screen:

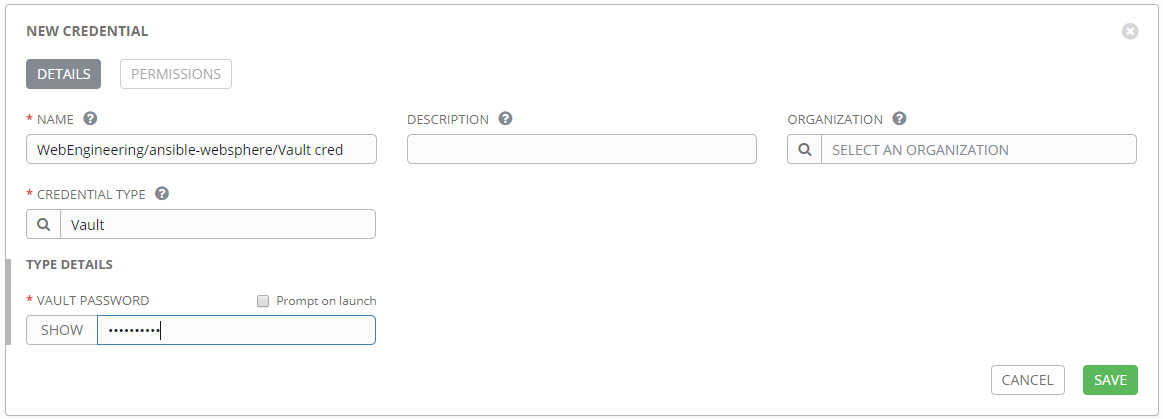
Note: Fields with a red asterisk (\*) are required fields

* Enter the name of the credential. This must be in the format <project name>/<credential name>. For example, if your project is WebEngineering/ansible-websphere and your credential will be called "Vault cred" the name of your credential will be: WebEngineering/ansible-websphere/Vault cred.
* Enter an optional description.

Note: By default, you will not be able to add an Organization to your credential. The credential will initially be for personal use. See the [raven](https://github.aetna.com/pages/IaC/Docs/iacservices/ansible/tower/raven.html#permissions) page for information on how to add permissions to the credential.

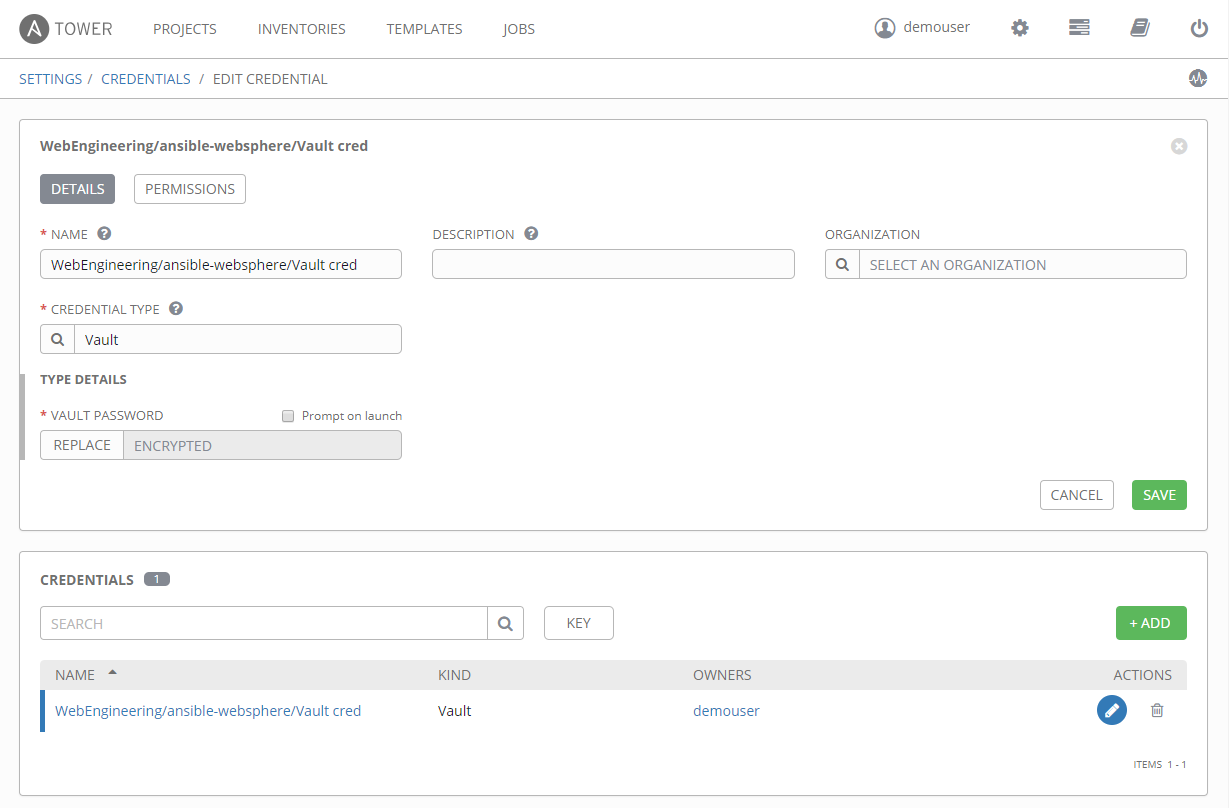
../../../images/ansible/spyglass.png Next select the credential type by clicking on the spyglass icon. This will open the credential type selection screen:

 On this screen you can use the search bar to search for a credential type or use the page selection to scroll through the different types of credentials. Continuing with our example, we will be selecting a credential type of Vault. Once selected, this will open a new section on the screen to ask for additional details based on the credential type. For a vault credential, the only additional information is "Vault Password":



Note: the password I entered is added with asterisks

../../../images/ansible/save.pngOnce all of your credential and credential type information has been properly entered click the green save button. This will bring you to the edit screen of your credential and will add the credential to the list of credentials listed below the edit screen:



Note: The Vault Password field has changed from asterisks to the word "ENCRYPTED". This indicates that the field has been securely stored in the database and can not be retrieved by anyone ever again.