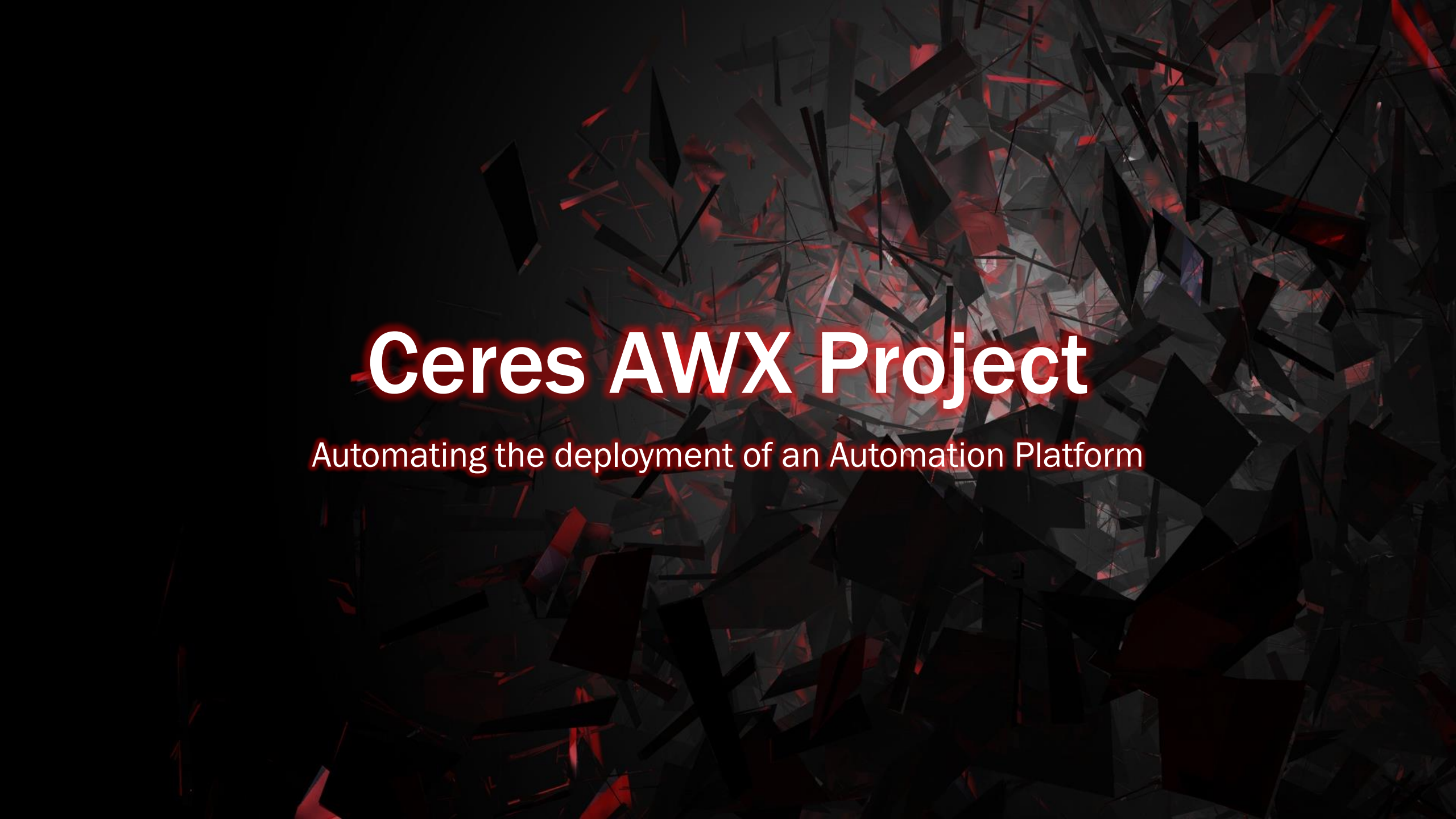

Ceres AWX Project
Automating the deployment of an Automation Platform

Who am I and what is this project?

My name is Richard Gowen and among other things, I architect and implement security automation solutions for cloud infrastructure.

Twitter: `@alt_bier` Mastodon: `@alt_bier@defcon.social`

Projects: `altbier.us` Resume: `gowen.net/resume/`

GitHub: `github.com/gowenrw`

The Ceres AWX Project is my attempt to bring the power of the enterprise (and thus expensive) Ansible Automation Platform (AAP) to the individual user **easily** via AAP's upstream project AWX.

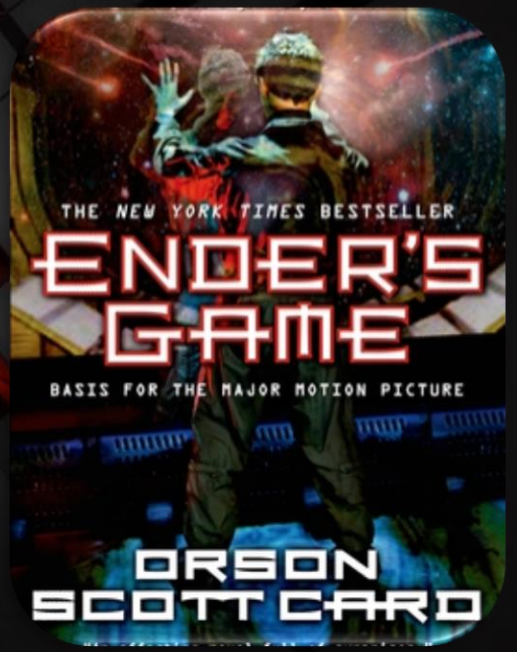
While AWX is free to use, it can be difficult install and configure.

What is Ansible?

1. Fictional Instantaneous Hyperspace Communication System featured in Orson Scott Card's Ender's Game
2. Simple To Use IT Automation Software For Orchestrating Advanced IT Tasks

← THIS ONE

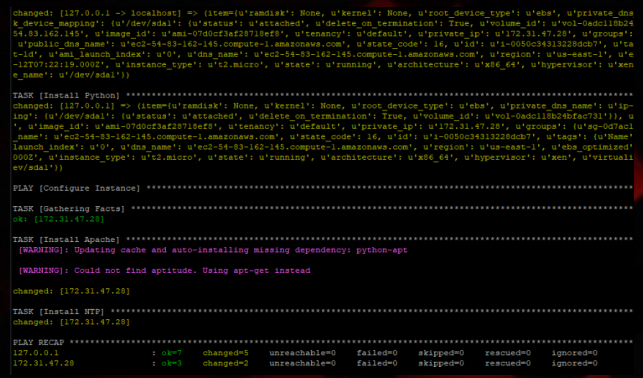
Not this one →



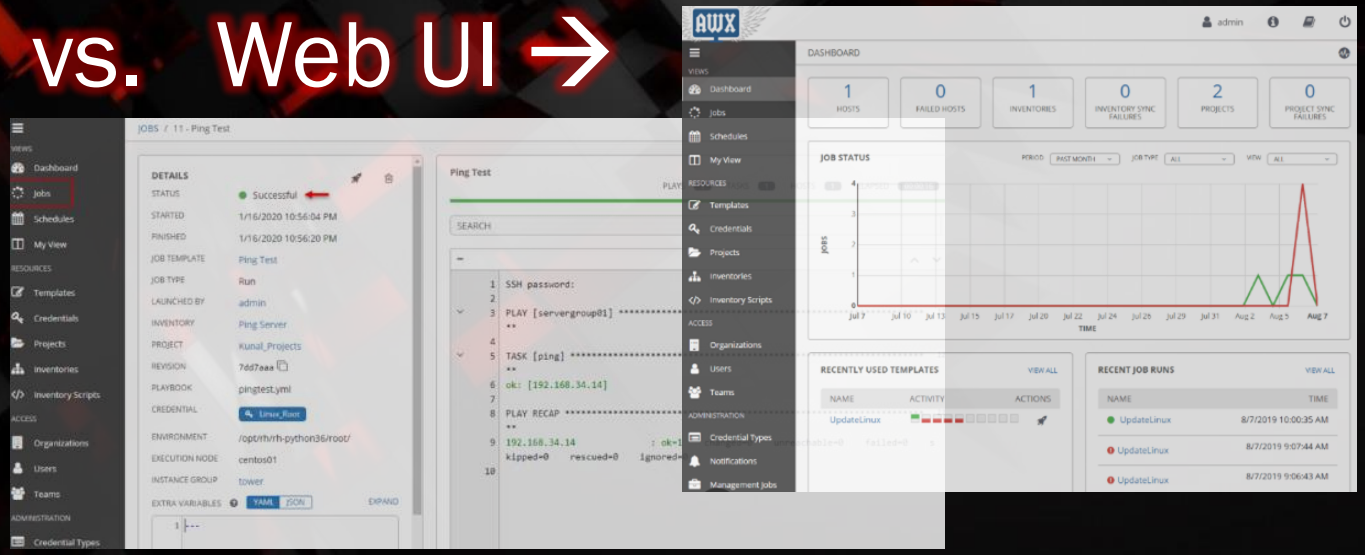
What are Ansible Automation Platform & AWX?

While Ansible in it's CLI form has been around for more than a decade, scaling it up for enterprise use has always been a challenge.

Ansible Automation Platform (AAP) and its upstream project AWX tackle this challenge by bundling Ansible and several other tools together in a scalable enterprise grade platform with an easy-to-use web interface.



← CLI vs. Web UI →



Why use AWX versus the CLI?

The power of Ansible is that there are many community collections of code already written to do the things you need to do. But they all have different python module requirements which may cause conflicts. This can lead to complex python virtual environment configurations on the Ansible host.

Newer versions of Ansible tackle this with ‘Execution Environment’ containers. While its possible to use EE’s in the CLI this adds complexity and is not scalable

AWX and AAP make use of EE containers in a portable, scalable, user-friendly interface. In addition, these platforms have other features not available in the CLI, such as job scheduling, workflows (tying jobs together), target inventory mgt, credential mgt, external authentication hooks (e.g., key vaults), and more.

What does the Ceres AWX project do?

The main goal of this project is to automate the provisioning of Ansible AWX on a local VM or a VM in the cloud.

Another goal is to automate the creation of custom Ansible execution environments for use with ansible-navigator and AWX.

Finally, I wanted to meet these goals in a way that is accessible to those with limited Ansible knowledge (i.e., lots of documentation).

Where Can I Get The Ceres AWX Project?

Ceres AWX is a public project on GitHub available here:

`https://github.com/gowenrw/ceres_awx`

The project is an ongoing work as I look to expand its use to other operating systems and other cloud environments.

Feel free to let me know if you see something that can be improved.

How to use the Ceres AWX project

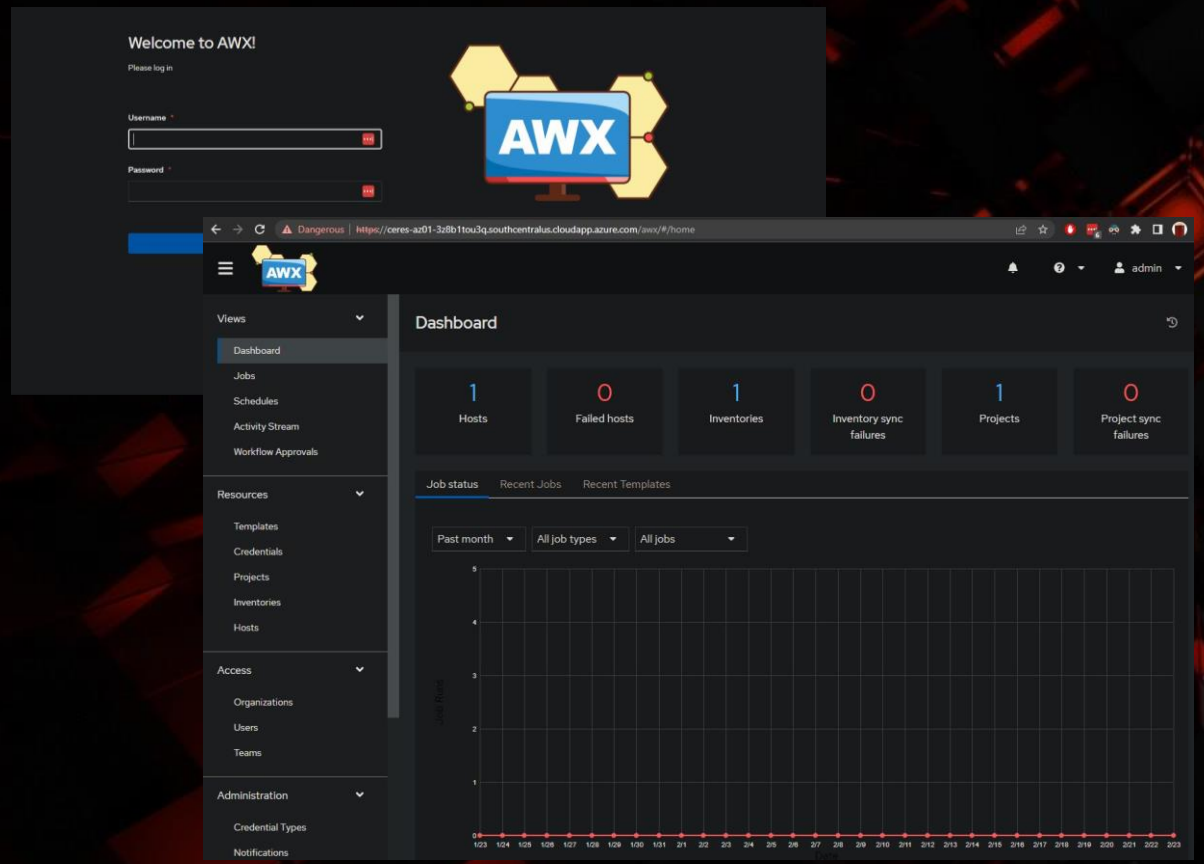
- Ready the local environment by installing requirements
  - ansible-navigator and podman or docker (see project README for details)
- Get a Target VM ready for AWX
  - Minimum Hardware: 2 CPU cores & 8GB RAM & 10GB Free Disk
  - This can be a local VM or a VM in a cloud environment
  - Current OS requirements:
    - Local: CentOS Stream 9 or Rocky Linux 9 - Cloud: RHEL 9
- For convenience, this project contains a Vagrant/VirtualBox configuration as well as an Azure setup automation job to provision the target VM
  - For a local VM simply invoke vagrant with **`vagrant up ceres-c9`**
  - The Azure setup job creates the VNET, Subnet, NSG, ASG, NICs, and VM
  - You need to supply as variables the Azure Tenant-ID, Subscription, Resource-Group, and Service-Principal details then run this playbook: **`ansible-navigator run ceres.playbook.azsetup.yml`**

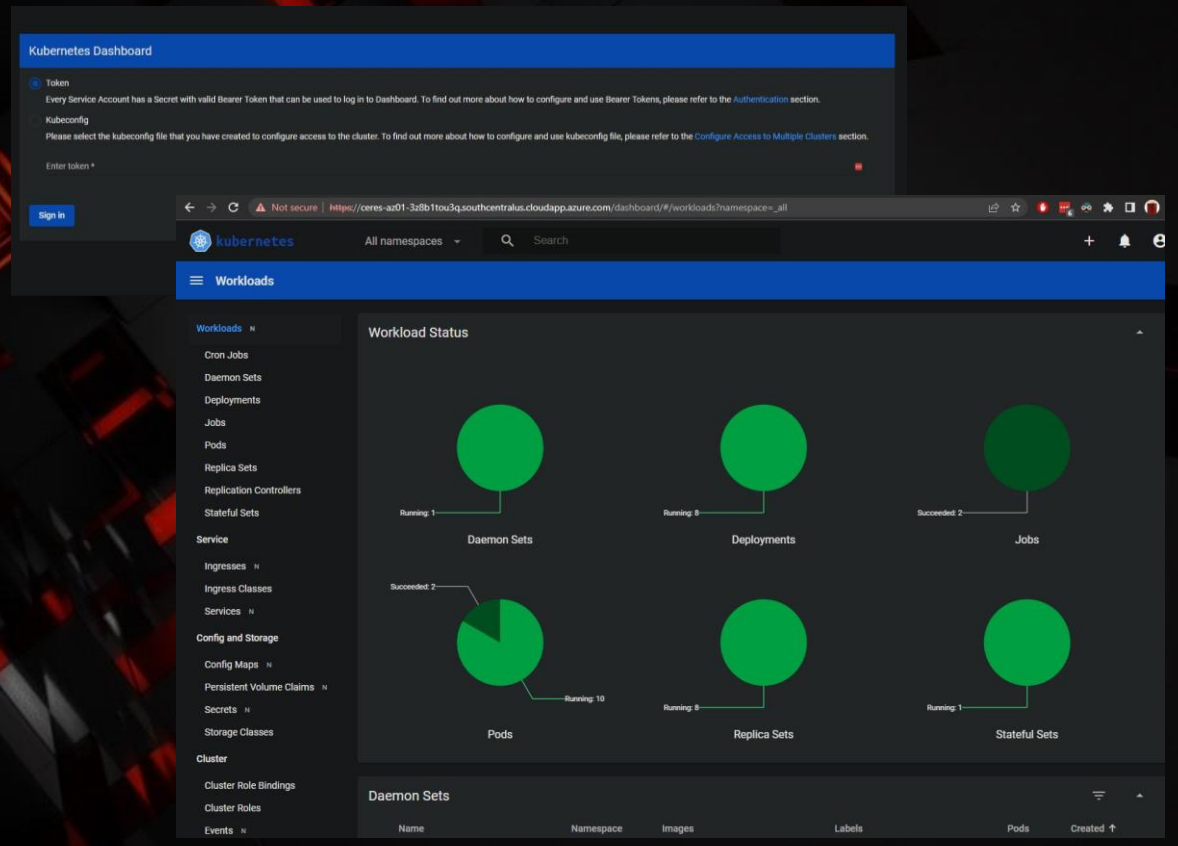
How to use the Ceres AWX project

- Update Inventory File or Copy Vagrant SSH Keys
  - If you provisioned a local or cloud VM without the provided scripts then update the file **`ceres.inventory`** with the details of your target VM.
  - If you provisioned a local VM using the provided Vagrantfile then you need to copy the SSH keys from .vagrant to the root project directory. This can be done by running the included script: **`./bin/wkeycp.sh`**
- Execute the Ansible automation playbook to provision AWX (including all it's requirements such as Kubernetes) onto the target VM
  - For a Local VM (CentOS Stream 9 / Rocky Linux 9) use this playbook: **`ansible-navigator run ceres.playbook.dev.yml`**
  - For a Cloud VM (RHEL 9) use this playbook: **`ansible-navigator run ceres.playbook.qa.yml`**

Enjoy AWX!

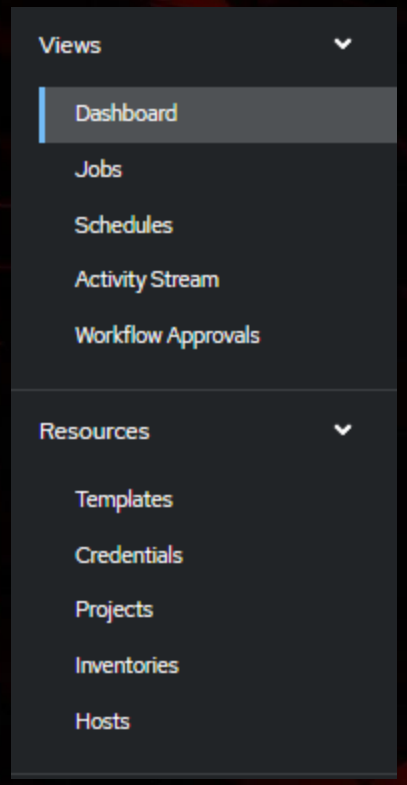
When the automation job completes it will present you with the login information (URL & Auth) for the Kubernetes Dashboard and AWX.



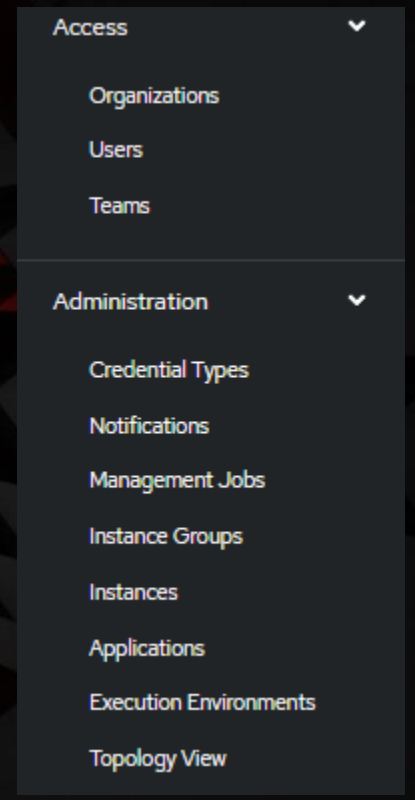


AWX Basics

Here are some AWX basics to get you started using it



- Projects – Code repositories (e.g., GitHub, GitLab, Azure Devops)
  - This is where the Ansible code is to use
  - Can be configured to do a fresh git pull before each job run
- Templates – Job templates and Workflow Templates to run
  - Job templates reference a playbook within Project code repository
  - Workflow templates group together Jobs to run conditionally
  - Schedules – Job or Workflow templates can be scheduled
  - Jobs – The Jobs view shows all past or currently running jobs.
- Inventories – Target hosts to run jobs against
  - These are the same as local inventory files in the CLI
  - Hosts is similar to a local host file and can be used like inventories
- Credentials – Creds to use for Vault, SSH(Machine), Repo, and more
  - Credential Types – Custom Credential Templates
- Execution Environments – The EE's that are available to run jobs in



Thank You!

Ceres AWX project:

https://github.com/gowenrw/ceres_awx

Where you can find me:

Twitter: `@alt_bier` Mastodon: `@alt_bier@defcon.social`

Projects: `altbier.us` Resume: `gowen.net/resume/`

GitHub: `github.com/gowenrw`