Ansible inventory from a csv file.

General

Create an Ansible inventory in YAML format using the following steps.

Assuming that the CSV file has the following structure:

```
Hostname,IP Address,Variable1,Variable2,Variable3
host1,192.168.1.1,value1,value2,value3
host2,192.168.1.2,value4,value5,value6
```

Local Use

1. Convert the CSV file to a YAML file format
2. Use Ansible's `yaml_inventory_plugin` to parse the YAML file and create the inventory

Sample playbook

```
---
- hosts: localhost
  gather_facts: no

  vars:
    csv_file: /path/to/csv/file.csv
    yaml_file: /path/to/yaml/file.yaml

  tasks:
    - name: Convert CSV to YAML
      community.general.csv_to_yaml:
```

```
    path: "{{ csv_file }}"
    output_file: "{{ yaml_file }}"

- name: Create inventory from YAML
  ansible.builtin.add_host:
    name: "{{ item.Hostname }}"
    ansible_host: "{{ item['IP Address'] }}"
    variable1: "{{ item.Variable1 }}"
    variable2: "{{ item.Variable2 }}"
    variable3: "{{ item.Variable3 }}"
  loop: "{{ lookup('yaml', yaml_file) }}"
```

In this example, the csv_to_yaml Ansible Galaxy module is used to convert the CSV file to YAML format. The add_host module is then used to create the inventory based on the YAML file contents.

You can run this playbook with the following command:

```
ansible-playbook -i localhost, inventory.yml
```

AWX/AAP/Tower

Assuming that the CSV file has the following structure:

```
group,host,IP Address,Variable1,Variable2,Variable3
group1,host1,192.168.1.1,value1,value2,value3
group2,host2,192.168.1.2,value4,value5,value6
group2,host3,192.168.1.3,value4,value5,value6
```

Here's an example Ansible playbook that reads a CSV file and creates an inventory in Ansible AWX, or Ansible Automation Platform.

You'll need to fill in the values for `tower_host`, `tower_username`, `tower_password`, `tower_org`, `tower_inventory_name`, and `csv_file`.

The playbook has four tasks:

1. Load CSV file: This task loads the CSV file and stores the content in the csv_content variable.
2. Create groups in Ansible Tower: This task creates groups in Ansible Tower based on the values in the group column of the CSV file. The loop parameter iterates over the unique values of the group column.

3. Create hosts in Ansible Tower: This task creates hosts in Ansible Tower based on the values in the host column of the CSV file. The loop parameter iterates over the unique values of the host column.
4. Add host variables to Ansible Tower hosts: This task adds variables to the hosts in Ansible Tower based on the values in the CSV file. The loop parameter iterates over each row in the CSV file.

```
---
- name: Create Ansible Tower Inventory from CSV
  hosts: localhost
  gather_facts: no

  vars:
    csv_file: /path/to/csv/file.csv
    tower_host: <Ansible Tower Host>
    tower_username: <Ansible Tower Username>
    tower_password: <Ansible Tower Password>
    tower_org: <Ansible Tower Organization>
    tower_inventory_name: <Ansible Tower Inventory Name>

  tasks:
    - name: Load CSV file
      read_csv:
        path: "{{ csv_file }}"
        delimiter: ","
      register: csv_content

    - name: Create groups in Ansible Tower
      tower_group:
        tower_host: "{{ tower_host }}"
        tower_username: "{{ tower_username }}"
        tower_password: "{{ tower_password }}"
        tower_organization: "{{ tower_org }}"
        name: "{{ item.group }}"
        state: present
      loop: "{{ csv_content.list | unique('group') }}"

    - name: Create hosts in Ansible Tower
      tower_host:
        tower_host: "{{ tower_host }}"
        tower_username: "{{ tower_username }}"
```

```
    tower_password: "{{ tower_password }}"
    tower_organization: "{{ tower_org }}"
    inventory_name: "{{ tower_inventory_name }}"
    name: "{{ item.host }}"
    state: present
  loop: "{{ csv_content.list | unique('host') }}"

- name: Add host variables to Ansible Tower hosts
  tower_host:
    tower_host: "{{ tower_host }}"
    tower_username: "{{ tower_username }}"
    tower_password: "{{ tower_password }}"
    tower_organization: "{{ tower_org }}"
    inventory_name: "{{ tower_inventory_name }}"
    name: "{{ item.host }}"
    variables: "{{ item.vars }}"
    state: present
  loop: "{{ csv_content.list }}"
```

A config file can be used in place of credentials being located in the playbook.

The ~/.tower_cli.cfg file is a configuration file used by the Ansible Tower CLI tool, tower-cli. It is located in the home directory of the user running tower-cli.

This file stores configuration settings for tower-cli such as the URL of the Ansible Tower server, the username and password used to authenticate to the server, and other options related to the tool's behavior.

```
[tower]
host = https://my-ansible-tower-server.com
username = my-username
password = my-password
verify_ssl = false
```

In this example, the [tower] section specifies the configuration settings for the Ansible Tower server. The host parameter specifies the URL of the server, while the username and password parameters specify the credentials used to authenticate to the server. The verify_ssl parameter can be set to true or false to indicate whether SSL certificates should be verified when making requests to the server.

By default, tower-cli looks for the ~/.tower_cli.cfg file in the user's home directory. However, you can specify a different location for the configuration file by setting the TOWERCLI_CONFIG

environment variable to the path of the file.

Revision #5
Created 5 April 2023 13:42:32 by Michael Cleary
Updated 6 February 2024 13:25:52 by Michael Cleary