

AWX Server Setup

Contents

1. Prerequisites:	2
2. AWX Installation Procedures	3
2.1 Obtain Local IP Address	3
2.2 Install AWX Dependencies	4
2.3 Download AWX Source	5
2.4 AWX Pre-Install Configuration	6
2.5 Install AWX	7
2.6 Verify AWX Installation	8
2.7 Configure and Install Web Server	9
2.8 Additional Configurations	11
2.8.1 Ansible SSH Configuration.....	11
2.8.2 Docker Service Configuration.....	11
3. Ansible CLI Tutorial	12
3.1 Test Switch Connectivity	12
3.2 Run Tasks with <i>ansible</i> Command	13
3.3 Run Sample Playbooks with <i>ansible-playbook</i> Command	14
4. AWX Basic Setup	16
4.1 Initial GUI Login.....	16
4.2 Create Projects.....	17
4.3 Add Credentials.....	18
4.4 Create Inventories.....	20
4.5 Creating Job Templates.....	22
5. Vim Basics	31
6. References	32
7. Additional Resources	32

1. Prerequisites:

- Fresh Ubuntu 18.04 Server install
- RAM: 3+ GB
- Disk Space: 40+ GB
- Bridged network adapter (if using a VM)

The following tutorial will walk through a setup of Ubuntu Server 18.04:

<https://www.fosslinux.com/6406/how-to-install-ubuntu-server-18-04-lts.htm>

2. AWX Installation Procedures

2.1 Obtain Local IP Address

Use the following command to obtain the IP address:

```
ip addr
```

Make a note of the server IP address shown in the output. For example:

```
awxuser@awxserver:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen
1000
    link/ether 00:0c:29:67:b2:b7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.41.66/23 brd 192.168.41.255 scope global dynamic ens33
        valid_lft 85833sec preferred_lft 85833sec
    inet6 fe80::20c:29ff:fe67:b2b7/64 scope link
        valid_lft forever preferred_lft forever
awxuser@awxserver:~$
```

2.2 Install AWX Dependencies

SSH into the server using PuTTY.

When logged into the server using PuTTY, copy and paste the following commands into the PuTTY window:

```
sudo apt update -y &&
sudo apt install docker.io -y &&
sudo apt install python3-pip -y &&
sudo pip3 install --upgrade pip &&
sudo pip3 install docker-compose==1.26.0 &&
sudo pip3 install ansible==2.9.10 &&
ansible --version &&
sudo docker version
```

This will install the initial dependencies for the AWX server and check the version of Ansible and Docker. The following results should be displayed:

```
ansible 2.9.10
  config file = None
  configured module search path = ['/home/awxuser/.ansible/plugins/modules',
'/usr/share/ansi
  ansible python module location = /usr/local/lib/python3.6/dist-packages/ansible
  executable location = /usr/local/bin/ansible
  python version = 3.6.9 (default, Jan 26 2021, 15:33:00) [GCC 8.4.0]
Client:
  Version:      20.10.2
  API version:  1.41
  Go version:   gol.13.8
  Git commit:   20.10.2-0ubuntu1~18.04.2
  Built:        Tue Mar 30 21:24:16 2021
  OS/Arch:      linux/amd64
  Context:      default
  Experimental: true

Server:
  Engine:
    Version:      20.10.2
    API version:  1.41 (minimum version 1.12)
    Go version:   gol.13.8
    Git commit:   20.10.2-0ubuntu1~18.04.2
    Built:        Mon Mar 29 19:27:41 2021
    OS/Arch:      linux/amd64
    Experimental: false
  containerd:
    Version:      1.3.3-0ubuntu1~18.04.4
    GitCommit:
  runc:
    Version:      spec: 1.0.2-dev
    GitCommit:
  docker-init:
    Version:      0.19.0
    GitCommit:
```

Note: The first time the **ansible** command is run, the '~/.ansible' directory is created. It is important that this directory is created with the local user as owner as a root owner can break certain functions. The **ansible** command above accomplishes this task as well as showing the Ansible version.

2.3 Download AWX Source

Download the AWX v13.0.0 server source code and go to the installation directory with the following commands:

```
git clone https://github.com/ansible/awx -b 13.0.0
cd awx/installer
```

2.4 AWX Pre-Install Configuration

Generate a new secret key for AWX using the following command:

```
openssl rand -hex 32
```

Copy the generated secret key to the clipboard.

Edit the 'inventory' configuration file:

```
vim inventory
```

Change the value of 'secret_key' from to the output of the 'openssl' command above.

Change the 'postgres_data_dir' to the `"/var/lib/pgdocker"` directory:

```
postgres_data_dir="/var/lib/pgdocker"
```

Change the 'host_port' to `8080`:

```
host_port=8080
```

Change the 'docker_compose_dir' to the `"/var/lib/awx"` directory:

```
docker_compose_dir="/var/lib/awx"
```

Uncomment the 'project_data_dir' and leave the default value:

```
project_data_dir=/var/lib/awx/projects
```

Change the credentials for the 'pg_password', 'admin_user' and 'admin_password' with your own password credentials. Do not include special characters in the password as this can break the install.

To see the active config entries, issue the following command:

```
grep -v "^#" inventory | grep -v "^$"
```

Results similar to the following should be displayed:

```
localhost ansible_connection=local ansible_python_interpreter="/usr/bin/env python3"
[all:vars]
dockerhub_base=ansible
awx_task_hostname=awx
awx_web_hostname=awxweb
postgres_data_dir="/var/lib/pgdocker"
host_port=8080
host_port_ssl=443
docker_compose_dir="/var/lib/awx"
pg_username=awx
pg_password=CallM3Snak3
pg_database=awx
pg_port=5432
admin_user=awxuser
admin_password=awxpassword
create_preload_data=True
secret_key=aa27d3da1b2bb4192265939e082f8e7ae1e30cd8dce1cfe943f564ca7ca3fbe8
project_data_dir=/var/lib/awx/projects
```

2.5 Install AWX

Install the AWX server with the following command:

```
sudo ansible-playbook -i inventory install.yml
```

This is a snippet of what the result should look like if the install completed successfully:

```
TASK [local_docker : Copy task image to docker execution] *****
skipping: [localhost]

TASK [local_docker : Load web image] *****
skipping: [localhost]

TASK [local_docker : Load task image] *****
skipping: [localhost]

TASK [local_docker : Set full image path for local install] *****
skipping: [localhost]

TASK [local_docker : Set DockerHub Image Paths] *****
ok: [localhost]

TASK [local_docker : Create /var/lib/awx directory] *****
changed: [localhost]

TASK [local_docker : Create Docker Compose Configuration] *****
changed: [localhost] => (item=environment.sh)
changed: [localhost] => (item=credentials.py)
changed: [localhost] => (item=docker-compose.yml)
changed: [localhost] => (item=nginx.conf)

TASK [local_docker : Render SECRET_KEY file] *****
changed: [localhost]

TASK [local_docker : Start the containers] *****
changed: [localhost]

TASK [local_docker : Update CA trust in awx_web container] *****
changed: [localhost]

TASK [local_docker : Update CA trust in awx_task container] *****
changed: [localhost]

PLAY RECAP *****
localhost                : ok=14   changed=6   unreachable=0   failed=0   skipped=100   r
escued=0   ignored=0

awxuser@awxserver:~/awx/installer$
```

2.6 Verify AWX Installation

With the AWX server installed, this is a good time to check for problems before proceeding. Issue the following command to display the AWX log:

```
sudo docker logs -f awx_task
```

The logs will continuously display until Ctrl-c is pressed. Something similar to the following would normally be displayed:

```
2020-02-11 14:00:41,345 DEBUG    awx.main.dispatch publish
awx.main.tasks.awx_periodic_scheduler(46570e74-f9c5-4181-a6cd-45cbf317ca78,
queue=awx_private_queue)
[2020-02-11 14:00:41,345: DEBUG/Process-1] publish
awx.main.tasks.awx_periodic_scheduler(46570e74-f9c5-4181-a6cd-45cbf317ca78,
queue=awx_private_queue)
2020-02-11 14:00:41,379 DEBUG    awx.main.dispatch delivered 46570e74-f9c5-4181-a6cd-
45cbf317ca78 to worker[190] qsize 0
2020-02-11 14:00:41,384 DEBUG    awx.main.dispatch task 46570e74-f9c5-4181-a6cd-45cbf317ca78
starting awx.main.tasks.awx_periodic_scheduler(*[])
2020-02-11 14:00:41,399 DEBUG    awx.main.tasks Starting periodic scheduler
2020-02-11 14:00:41,402 DEBUG    awx.main.tasks Last scheduler run was: 2020-02-11
14:00:11.530984+00:00
2020-02-11 14:00:41,419 DEBUG    awx.main.dispatch task 46570e74-f9c5-4181-a6cd-45cbf317ca78 is
finished
2020-02-11 14:00:51,640 DEBUG    awx.main.dispatch publish
awx.main.scheduler.tasks.run_task_manager(5544bef6-bdfe-4d59-b084-cbea735c4104,
queue=awx_private_queue)
[2020-02-11 14:00:51,640: DEBUG/Process-1] publish
awx.main.scheduler.tasks.run_task_manager(5544bef6-bdfe-4d59-b084-cbea735c4104,
queue=awx_private_queue)
2020-02-11 14:00:51,677 DEBUG    awx.main.dispatch delivered 5544bef6-bdfe-4d59-b084-
cbea735c4104 to worker[193] qsize 0
2020-02-11 14:00:51,680 DEBUG    awx.main.dispatch task 5544bef6-bdfe-4d59-b084-cbea735c4104
starting awx.main.scheduler.tasks.run_task_manager(*[])
2020-02-11 14:00:51,684 DEBUG    awx.main.scheduler Running Tower task manager.
2020-02-11 14:00:51,698 DEBUG    awx.main.scheduler Starting Scheduler
2020-02-11 14:00:51,725 DEBUG    awx.main.dispatch task 5544bef6-bdfe-4d59-b084-cbea735c4104 is
finished
RESULT 2
OKREADY
2020-02-11 14:01:10,907 DEBUG    awx.main.dispatch publish
awx.main.tasks.cluster_node_heartbeat(bae84654-267c-47d5-840c-8cb42fc1f48e, queue=awx)
[2020-02-11 14:01:10,907: DEBUG/Process-1] publish
awx.main.tasks.cluster_node_heartbeat(bae84654-267c-47d5-840c-8cb42fc1f48e, queue=awx)
2020-02-11 14:01:10,951 DEBUG    awx.main.dispatch delivered bae84654-267c-47d5-840c-
8cb42fc1f48e to worker[191] qsize 0
2020-02-11 14:01:10,957 DEBUG    awx.main.dispatch task bae84654-267c-47d5-840c-8cb42fc1f48e
starting awx.main.tasks.cluster_node_heartbeat(*[])
2020-02-11 14:01:10,961 DEBUG    awx.main.tasks Cluster node heartbeat task.
2020-02-11 14:01:10,981 DEBUG    awx.main.dispatch task bae84654-267c-47d5-840c-8cb42fc1f48e is
finished
```

An indication of a bad install would be a large amount of repeating ERROR logs and stack traces.

2.7 Configure and Install Web Server

Install nginx and navigate to the nginx config directory with the following commands:

```
sudo apt install nginx -y
cd /etc/nginx/sites-available/
```

Create a new virtual host configuration called "awx" using vim:

```
sudo vim awx
```

Substitute the highlighted IP addresses before pasting the following configuration:

```
server {
    listen 80;
    server_name 192.168.41.66;
    add_header Strict-Transport-Security max-age=2592000;
    rewrite ^ https://$server_name$request_uri? permanent;
}

server {
    listen 443 ssl http2;
    server_name 192.168.41.66;

    access_log /var/log/nginx/awx.access.log;
    error_log /var/log/nginx/awx.error.log;

    ssl on;
    ssl_certificate /etc/nginx/ssl/fullchain.pem;
    ssl_certificate_key /etc/nginx/ssl/privkey.pem;
    ssl_session_timeout 5m;
    ssl_ciphers
EECDH+CHACHA20:EECDH+AES128:RSA+AES128:EECDH+AES256:RSA+AES256:EECDH+3DES:RS
A+3DES:!MD5;
    ssl_protocols TLSv1.2;
    ssl_prefer_server_ciphers on;

    location / {
        proxy_http_version 1.1;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_pass http://192.168.41.66:8080/;
    }
}
```

Activate the "awx" virtual host with the following command:

```
sudo ln -s /etc/nginx/sites-available/awx /etc/nginx/sites-enabled/
```

Create the `/etc/nginx/ssl` directory and generate the self-signed SSL certificate and key with the following commands:

```
sudo mkdir /etc/nginx/ssl
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
    -keyout /etc/nginx/ssl/privkey.pem \
    -out /etc/nginx/ssl/fullchain.pem
```

A series of questions will be asked during the creation of the certificate and key:

```
awxuser@awxserver:/etc/nginx/sites-available$ sudo mkdir /etc/nginx/ssl
awxuser@awxserver:/etc/nginx/sites-available$ sudo openssl req -x509 -nodes -days 365 -newkey
rsa:2048 \
> -keyout /etc/nginx/ssl/privkey.pem \
> -out /etc/nginx/ssl/fullchain.pem
Can't load /home/awxuser/.rnd into RNG
140217327120832:error:2406F079:random number generator:RAND_load_file:Cannot open
file:../crypto/rand/randfile.c:88:Filename=/home/awxuser/.rnd
Generating a RSA private key
.....+++++
.....+++++
+++
writing new private key to '/etc/nginx/ssl/privkey.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:Virginia
Locality Name (eg, city) []:Norfolk
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:192.168.41.66
Email Address []:
awxuser@awxserver:/etc/nginx/sites-available$
```

The nginx setup can be tested with the following command:

```
sudo nginx -t
```

If the test was successful, the results will look like this:

```
awxuser@awxserver:/etc/nginx/sites-available$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
awxuser@awxserver:/etc/nginx/sites-available$
```

If there are no errors, restart the nginx service:

```
sudo systemctl restart nginx
```

2.8 Additional Configurations

2.8.1 Ansible SSH Configuration

When running playbooks from the CLI against a host, it is desirable to disable the interactive prompt that will cause the process to fail when the SSH client connects to a new host.

First create a new Ansible config file:

```
vim ~/.ansible.cfg
```

Then paste the following into the file:

```
[defaults]
host_key_checking = False
```

To ensure Ansible is using this config, issue the following command:

```
sudo ansible --version
```

The following results should be displayed:

```
ansible 2.9.10
  config file = /home/awxuser/.ansible.cfg
  configured module search path = ['/home/awxuser/.ansible/plugins/modules',
  '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.6/dist-packages/ansible
  executable location = /usr/local/bin/ansible
  python version = 3.6.9 (default, Apr 18 2020, 01:56:04) [GCC 8.4.0]
```

2.8.2 Docker Service Configuration

By default, Docker must be started manually. To make Docker start on boot, issue the following command:

```
sudo systemctl enable docker
```

3. Ansible CLI Tutorial

3.1 Test Switch Connectivity

To test credentials and connectivity to the switch, use the **ssh** command with this syntax:

```
ssh <user>@<hostip> -o KexAlgorithms=+diffie-hellman-group1-sha1
```

The following output should be displayed:

```
awxuser@awxserver:~$ ssh awxuser@192.168.20.72 -o KexAlgorithms=+diffie-hellman-group1-sha1
The authenticity of host '192.168.20.72 (192.168.20.72)' can't be established.
RSA key fingerprint is SHA256:4Vytdyb7L3DiRc+3CpcvqUGbeqlvCSoyFkg9ln7htNM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.20.72' (RSA) to the list of known hosts.
Password:
SSH@NLAB-U01-AS-01>enable
Password:
SSH@NLAB-U01-AS-01#
```

While in the switch display the config, then exit with the following commands:

```
skip
show running-config
exit
exit
```

3.2 Run Tasks with *ansible* Command

A single **ansible** command can be run from the command line as an alternative to gathering information from switches manually:

```
ansible all \
  -i 192.168.20.72, \
  -c network_cli \
  -k -K -b \
  --become-method enable \
  -e ansible_network_os=icx \
  -m icx_facts -a gather_subset=all \
  -u awxuser
```

Command breakdown:

- Host group we are connecting to (all)
- -i specifies an inventory followed by either target device with a trailing comma.
- -k ask you for the ssh password
- -K ask you for the become (privileged user) password
- -b run operations with become
- --become-method privilege escalation method
- -e sets an extra variable used by ansible, in this case setting the ansible_network_os to 'icx'
- -m name of the python module to execute
- -a sets an argument to pass to the python module
- -u login username for the switch

3.3 Run Sample Playbooks with *ansible-playbook* Command

Tasks run with the **ansible** command can be written into a YAML playbook that can be run with the **ansible-playbook** command and the AWX GUI.

First create a new playbook:

```
vim ~/pbfacts.yml
```

Then copy and paste the following:

```
---
- hosts: all
  connection: network_cli
  gather_facts: no
  vars:
    ansible_network_os: icx
    ansible_become: true
    ansible_become_method: enable

  tasks:
    - name: Gather Switch Info
      icx_facts:
        gather_subset: all
        register: output

    - name: Show Info
      debug:
        msg: "{{ output }}"
```

In this playbook you can see the command 'register' defines a variable named 'output' as the output of the task above it. Variables can be dereferenced using '{{ }}'.

To run this playbook, enter the following:

```
ansible-playbook -i 192.168.20.72, -k -K -u awxuser pbfacts.yml
```

Let's say VLAN 999 is to be configured on a switch. To create the VLAN manually, the following commands might be entered directly on the switch:

```
vlan 999 name U_USER_999 by port
  tagged ethe 1/2/1 to 1/2/2
  spanning-tree
!
write memory
```

As an alternative to manually running the previous commands on a switch, a playbook can be created that can accomplish the same task.

First create the playbook file:

```
vim ~/pbvlan.yml
```

Then copy and pasted the following:

```
---
- hosts: all
  connection: network_cli
  gather_facts: no
  vars:
    ansible_network_os: icx
    ansible_become: true
    ansible_become_method: enable

  tasks:
    - name: create vlan
      icx_config:
        lines: |
          vlan {{ vlan_number }} name {{ vlan_name }}
          tagged {{ trunk_ports }}
          spanning-tree
          write memory
```

This playbook dereferences three variables that are not defined in the playbook. These variables can be defined with '-e' on the command line when running a playbook:

```
ansible-playbook -i 192.168.20.72, -k -K \
    -u awxuser \
    -e trunk_ports="'e1/2/1 e1/2/2'" \
    -e vlan_name=U_USER_999 \
    -e vlan_number=999 \
    pbvlan.yml
```

In order to run the playbooks from the AWX GUI, they will need to be in a projects folder. First create a new project directory:

```
sudo mkdir /var/lib/awx/projects/ruckus
```

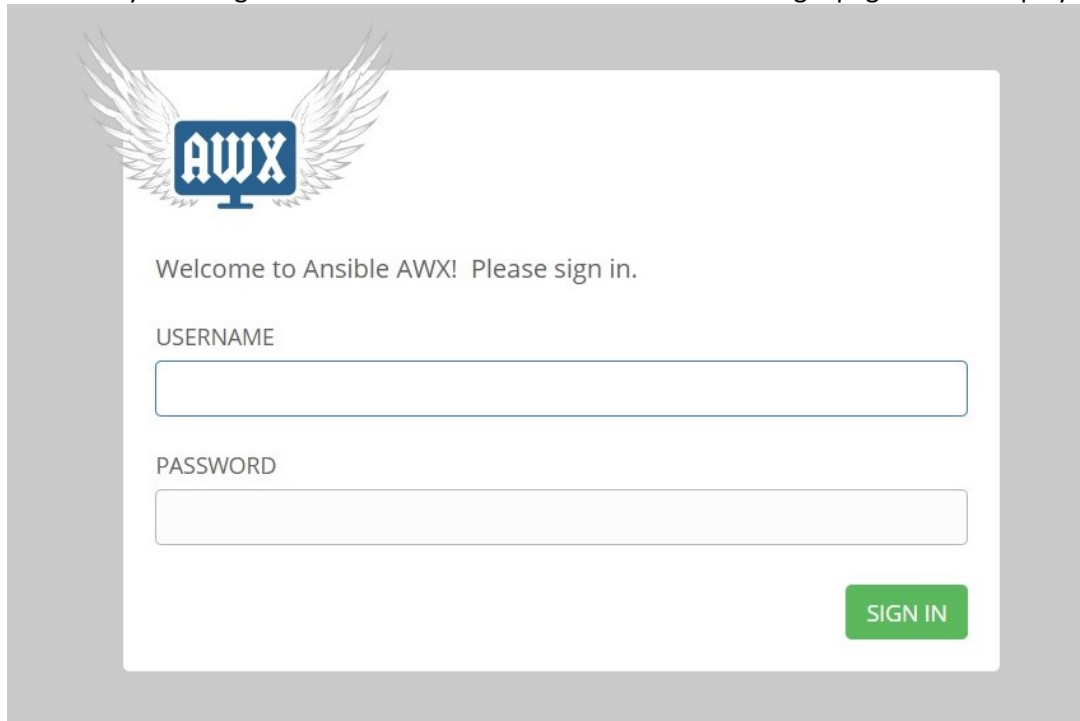
Then copy the playbooks into the project directory:

```
sudo cp ~/pbfacts.yml ~/pbvlan.yml /var/lib/awx/projects/ruckus
```

4. AWX Basic Setup

4.1 Initial GUI Login

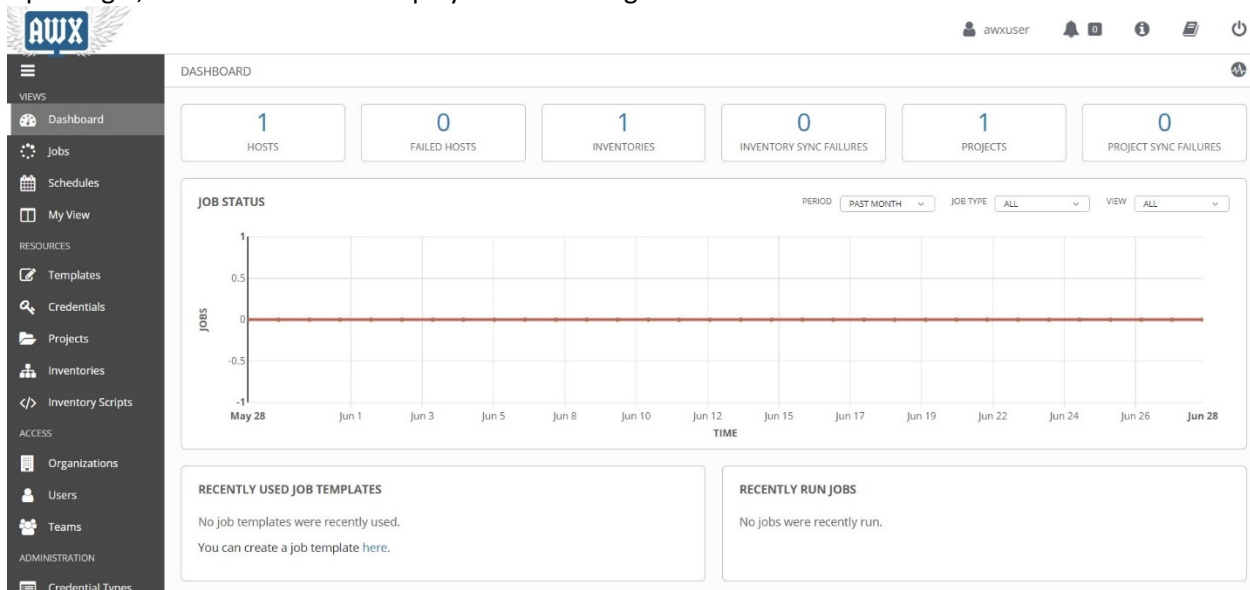
At this point the AWX server setup should be finished and the AWX web GUI can be accessed from a browser by entering the AWX IP into the address bar. The AWX login page should display:



The login page features the AWX logo at the top, which consists of the letters 'AWX' in a blue box with white wings. Below the logo, the text 'Welcome to Ansible AWX! Please sign in.' is displayed. There are two input fields: 'USERNAME' and 'PASSWORD'. A green 'SIGN IN' button is located at the bottom right of the form.

Login with the username and password that was set in the Inventory file (admin_user & admin_password).

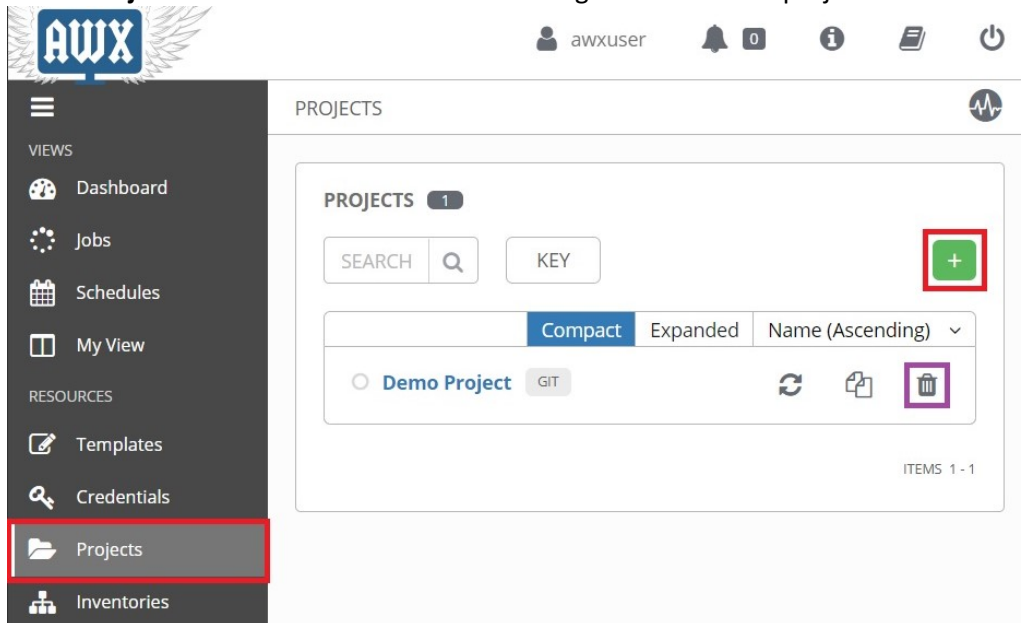
Upon login, the dashboard will display with the navigation bar on the left:



4.2 Create Projects

A project is a logical collection of playbooks. The following will guide you through the creation of a project whose playbooks have been manually placed in the project base path of the AWX server.

Click **Projects** from the left navigation menu. Click the delete (🗑️) button to optionally remove the **Demo Project**. Then click the + button on the right to add a new project:



The **New Project** page will display. Complete the following fields:

NAME: Create a name for the project. Here it is named "*ICX Test*".

SCM TYPE: Select **Manual**.

PLAYBOOK DIRECTORY: Select the name of the folder you copied the playbooks into.

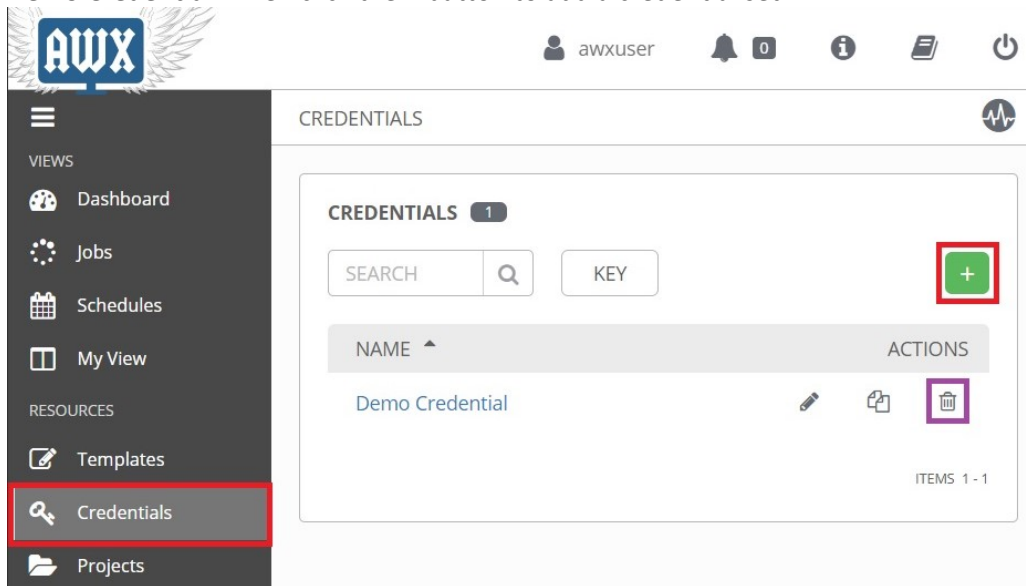
The screenshot shows the 'NEW PROJECT' form in the AWX interface. The form has four tabs: 'DETAILS', 'PERMISSIONS', 'JOB TEMPLATES', and 'SCHEDULES'. The 'DETAILS' tab is active. It contains several fields: 'NAME' (required, value 'ICX Test'), 'DESCRIPTION', 'ORGANIZATION' (required, value 'Default'), 'SCM TYPE' (required, dropdown menu with 'Manual' selected), 'PROJECT BASE PATH' (value '/var/lib/awx/projects'), and 'PLAYBOOK DIRECTORY' (required, dropdown menu with 'ruckus' selected). At the bottom right, there are 'CANCEL' and 'SAVE' buttons. The 'SAVE' button is highlighted with a green rectangular box. The 'NAME' field and the 'SCM TYPE' dropdown are also highlighted with red boxes.

When finished, click the **Save** button.

4.3 Add Credentials

For AWX to log into a device, you will need to create a set of credentials that can be used in a playbook.

Click **Credentials** from the left navigation menu. Click the delete (🗑️) button to optionally delete the **Demo Credential**. Then click the + button to add a credential set:



The **New Credential** page will display. Complete the following fields:

NAME: Create a name for the credential set. Here it is named "ICX Login"

CREDENTIAL TYPE: Select **Machine**.

USERNAME & PASSWORD: Enter the username and password used to login to the switch.

The screenshot shows the 'NEW CREDENTIAL' form in AWX. The form has tabs for 'DETAILS' and 'PERMISSIONS'. Fields for NAME, CREDENTIAL TYPE, USERNAME, and PASSWORD are highlighted with red boxes. The NAME field contains 'ICX Login', CREDENTIAL TYPE is set to 'Machine', USERNAME is 'awxuser', and PASSWORD is masked with dots.

Scroll to the bottom of the **New Credential** page. Here you can set the enable username and/or password:

Q Machine


TYPE DETAILS

USERNAME

Q awxuser

PASSWORD

☐ Prompt on launch

Q 

SSH PRIVATE KEY

HINT: Drag and drop private file on the field below.

Q


SIGNED SSH CERTIFICATE

HINT: Drag and drop private file on the field below.


Q

PRIVATE KEY PASSPHRASE

☐ Prompt on launch

Q 

PRIVILEGE ESCALATION METHOD


 enable ▼

PRIVILEGE ESCALATION USERNAME

Q

PRIVILEGE ESCALATION PASSWORD

☐ Prompt on launch

Q 

CANCEL

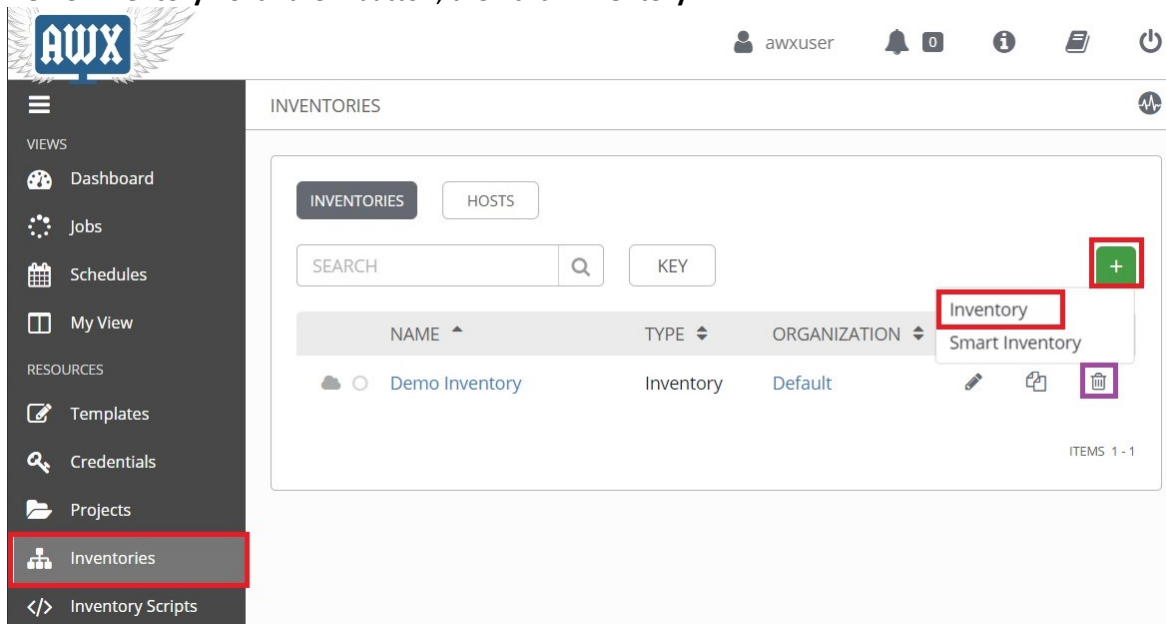
SAVE

When finished, click **Save**.

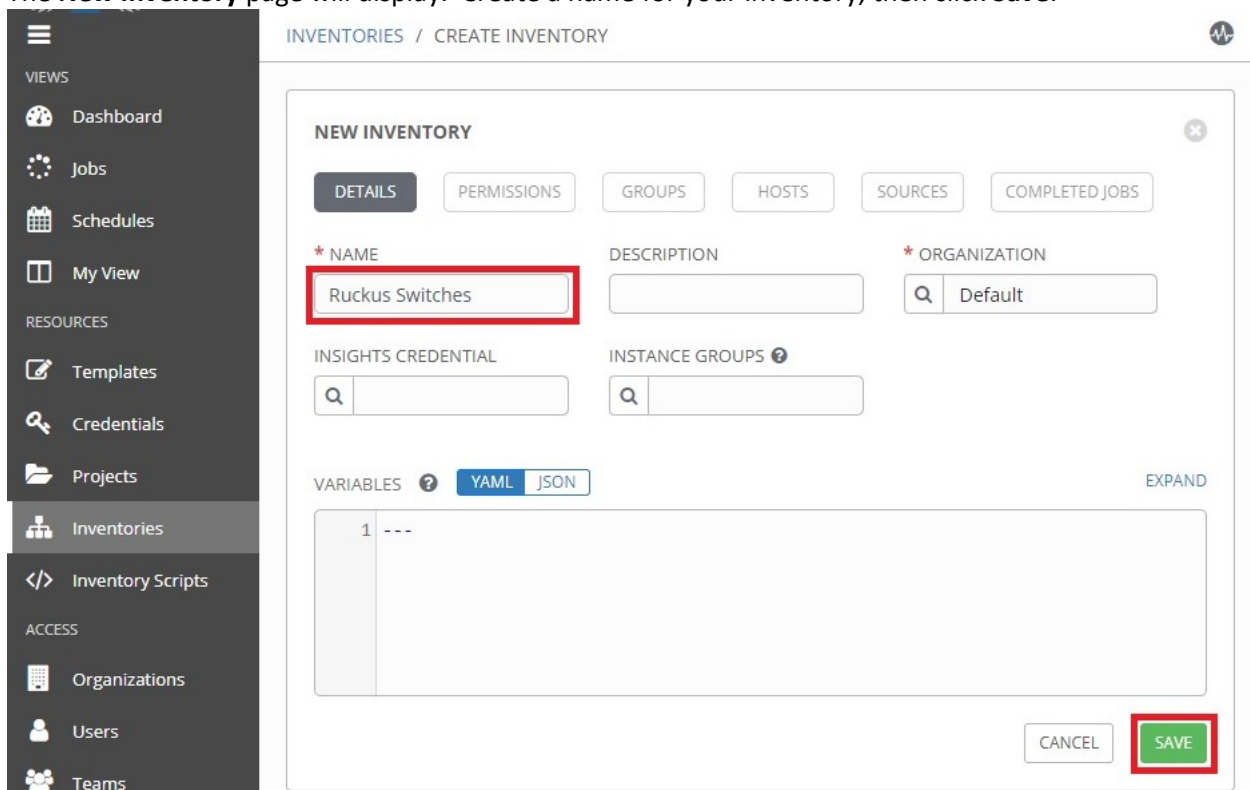
4.4 Create Inventories

To add a switch for AWX to manage, you will first need to create an inventory.

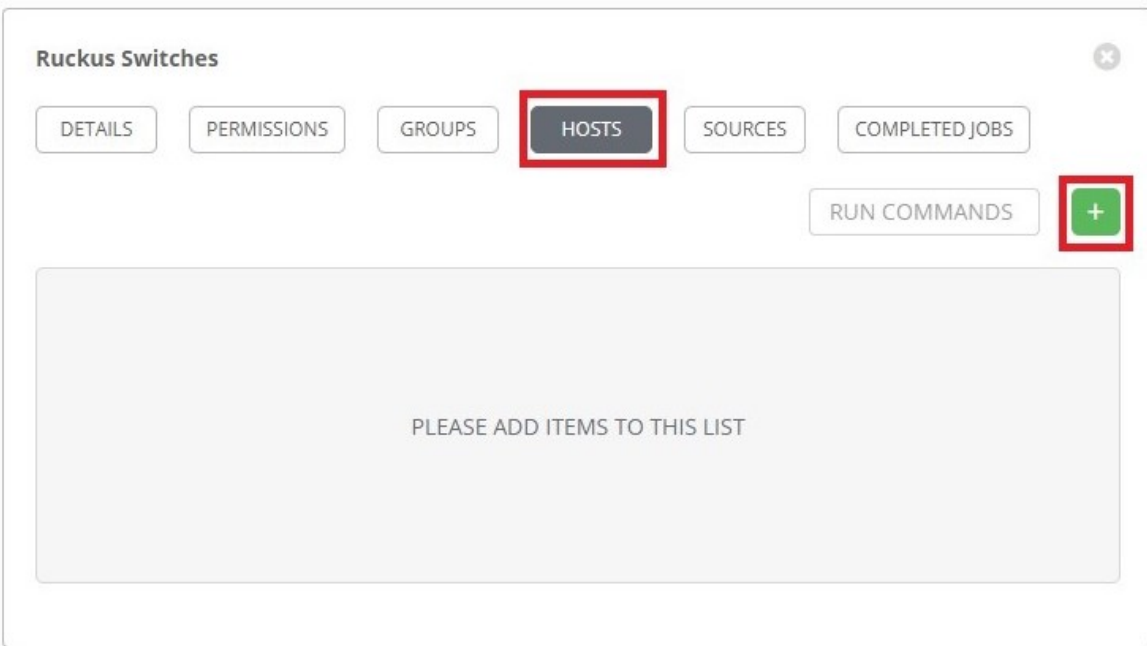
Click **Inventories** from the left navigation menu. Click the delete (🗑️) button to optionally delete the **Demo Inventory**. Click the + button, then click **Inventory**:



The **New Inventory** page will display. Create a name for your inventory, then click **Save**:



Switches can now be added to the newly created inventory.
Click on the **Hosts** tab, then click the **+** button:

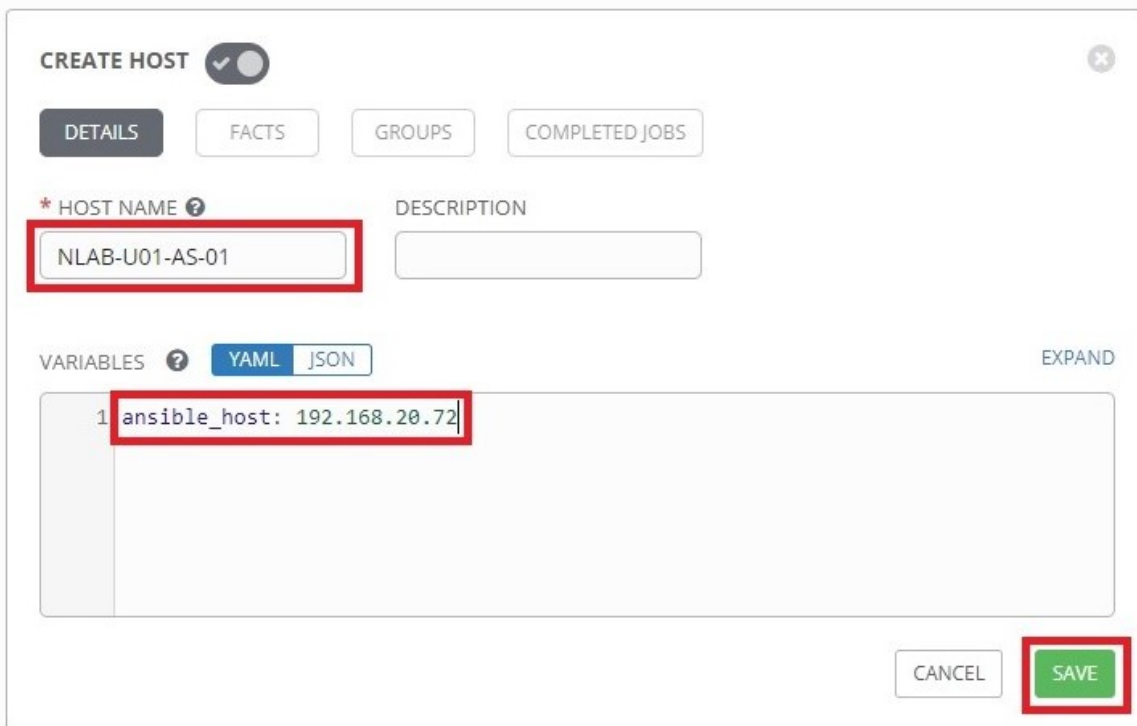


The screenshot shows a window titled "Ruckus Switches" with a close button in the top right. Below the title bar are five tabs: "DETAILS", "PERMISSIONS", "GROUPS", "HOSTS", and "SOURCES". The "HOSTS" tab is selected and highlighted with a red box. To the right of these tabs is a "COMPLETED JOBS" button. Below the tabs is a "RUN COMMANDS" button and a green square button with a white plus sign, which is also highlighted with a red box. The main area of the window is a large light gray rectangle with the text "PLEASE ADD ITEMS TO THIS LIST" centered inside.

The **Create Host** page will display. Complete the following fields:

HOST NAME: Enter the hostname of the switch.

VARIABLES: Type "**ansible_host:** " followed by the IP address of the switch.



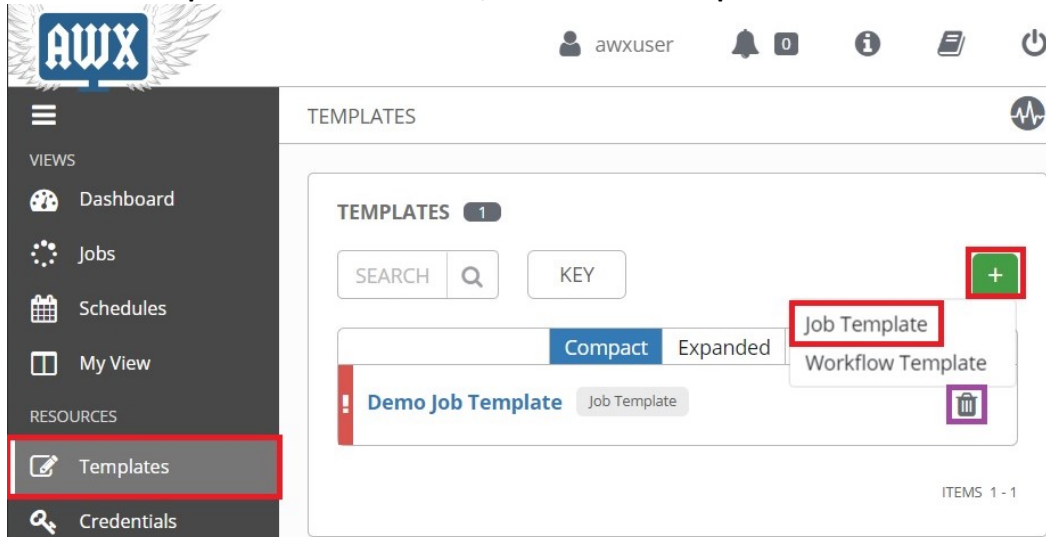
The screenshot shows a "CREATE HOST" window with a toggle switch that is turned on. Below the title bar are four tabs: "DETAILS", "FACTS", "GROUPS", and "COMPLETED JOBS". The "DETAILS" tab is selected. There are two input fields: "HOST NAME" and "DESCRIPTION". The "HOST NAME" field contains the text "NLAB-U01-AS-01" and is highlighted with a red box. Below these fields are two tabs: "YAML" and "JSON". The "YAML" tab is selected. Below the tabs is a large text area for variables. The first line of the text area contains the text "1 ansible_host: 192.168.20.72" and is highlighted with a red box. At the bottom right of the window are two buttons: "CANCEL" and "SAVE". The "SAVE" button is highlighted with a red box.

When finished, click **Save**.

4.5 Creating Job Templates

Now that the other pieces are in place, job templates can be created to run against host devices. The following guide will walk through the creation of a template that uses the '*pbfacts.yml*' playbook created earlier.

Click **Templates** from the left navigation menu. Click the delete (🗑️) button to optionally remove the **Demo Job Template**. Click the + button, then click **Job Template**:



The **New Job Template** page will display. Complete the following fields:

NAME: Create a name for the template. Here it is named "*ICX Facts*".

JOB TYPE: Select **Run**.

INVENTORY: Select the inventory created for the switch.

PROJECT: Select the project that contains the '*pbfacts.yml*' playbook.

PLAYBOOK: Select **pbfacts.yml**.

CREDENTIALS: Select the credential set that holds the switch login credentials.

The screenshot shows the 'NEW JOB TEMPLATE' form. It has tabs for 'DETAILS', 'PERMISSIONS', 'COMPLETED JOBS', 'SCHEDULES', and 'ADD SURVEY'. The 'DETAILS' tab is active. The form contains several fields: 'NAME' (text input with 'ICX Facts'), 'DESCRIPTION' (text input), 'JOB TYPE' (dropdown menu with 'Run'), 'INVENTORY' (searchable dropdown with 'Ruckus Switches'), 'PROJECT' (searchable dropdown with 'ICX Test'), 'PLAYBOOK' (dropdown menu with 'pbfacts.yml'), 'CREDENTIALS' (searchable dropdown with 'ICX Login'), 'FORKS' (text input with '0'), 'LIMIT' (text input), 'VERBOSITY' (checkbox), 'JOB TAGS' (checkbox), and 'SKIP TAGS' (checkbox). The fields for 'NAME', 'INVENTORY', 'CREDENTIALS', and 'JOB TYPE' are highlighted with red rectangular boxes.

When finished, click **Save**.

Once the template is saved, the **Launch** button will no longer be grayed out. Click the **Launch** button to execute the template:



EXTRA VARIABLES ?

YAML JSON

PROMPT ON LAUNCH

1 ---

LAUNCH CANCEL SAVE

TEMPLATES

1

SEARCH

Q

KEY

+

Compact

Expanded

Name (Ascending) v

ICX Facts

Job Template



This will bring up the **Details** page with a status window:

Dashboard

- [Jobs](#)
- [Schedules](#)
- [My View](#)
- [Resources](#)
- [Templates](#)
- [Credentials](#)
- [Projects](#)
- [Inventories](#)
- [Inventory Scripts](#)
- [Access](#)
- [Organizations](#)
- [Users](#)
- [Teams](#)
- [Administration](#)
- [Credential Types](#)
- [Notifications](#)
- [Management Jobs](#)
- [Instance Groups](#)
- [Applications](#)

DETAILS

STATUS ● Successful

STARTED 3/24/2020 1:05:34 PM

FINISHED 3/24/2020 1:05:45 PM

JOB TEMPLATE ICX Facts

JOB TYPE Run

LAUNCHED BY awxuser

INVENTORY Ruckus Switches

PROJECT ICX Test

PLAYBOOK pbfacts.yml

CREDENTIAL [ICX Login](#)

ENVIRONMENT /var/lib/awx/venv/ansible

EXECUTION NODE awx

INSTANCE GROUP tower

EXTRA VARIABLES [YAML](#) [JSON](#) [EXPAND](#)

1 ...

ICX Facts

PLAYS 1 TASKS 2 HOSTS 1 ELAPSED 00:00:10

```

3
4 PLAY [all] ***** 13:05:30
5
6 TASK [Gather Switch Info] ***** 13:05:30
7 ok: [NLAB-U01-AS-01]
8
9 TASK [Show Info] ***** 13:05:44
10 ok: [NLAB-U01-AS-01] => {
11   "msg": {
12     "ansible_facts": {
13       "ansible_net_all_ipv4_addresses": [],
14       "ansible_net_all_ipv6_addresses": [],
15       "ansible_net_config": "Current configuration:\n\nver 08.0.30gT203\n\nstack unit 1\n module 1 icx77
50-48-xgc-port-management-module\n module 2 icx7750-qsf-6port-qsf-240g-module\n\n\n\n\n\nvlan 1 name DE
FAULT-VLAN by port\n\nvlan 20 name TEMP_LAB_MGMT by port\n tagged ethe 1/1/41 \n router-interface ve 20\n\n\n\n\n\n\nnaaa authentication login default local\nenable super-user-password ....\nhostname NLAB-U01-AS-01\nip rou
te 0.0.0.0/0 192.168.20.1\n\nusername axuser password ..... \n\n\n\n\n\n\n\n\n\ninterface ve 20\n port-
name TEMP_LAB_MGMT\n ip address 192.168.20.72 255.255.255.0\n\n\n\n\n\n\n\n\n\nend",
16     "ansible_net_filesystems": "flash",
17     "ansible_net_filesystems_info": {
18       ..
19     }
20   }
21
22 PLAY RECAP ***** 13:05:44
23 NLAB-U01-AS-01 : ok=2 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ig
nored=0
24

```

The status window will display an abbreviated output when the template has finished executing. You can click on the output to show it in full. Also note the color of the status text. Green means the template executed successfully whereas red would indicate a failure.

The next guide will walk through the creation of a template that uses variables. Go back to the **Templates** page and add a new job template. Configure the new template to use the '*pbvlan.yml*' playbook:



NEW JOB TEMPLATE

DETAILS

PERMISSIONS

COMPLETED JOBS

SCHEDULES

ADD SURVEY

* NAME

ICX VLAN

DESCRIPTION

* JOB TYPE ?

Run

* INVENTORY ?

PROMPT ON LAUNCH

Ruckus Switches

* PROJECT ?

ICX Test

* PLAYBOOK ?

pbvlan.yml

CREDENTIALS ?

PROMPT ON LAUNCH

ICX Login

FORKS ?

0

LIMIT ?

PROMPT ON LAUNCH

Scroll to the bottom. Select **Enable Privilege Escalation**. Fill out the variable section as seen in the picture below. When you are finished, click **Save**:

TIMEOUT ? 0

SHOW CHANGES ? ☐ PROMPT ON LAUNCH

OPTIONS

- ☒ ENABLE PRIVILEGE ESCALATION
- ☐ ENABLE PROVISIONING CALLBACKS
- ☐ ENABLE WEBHOOK ?
- ☐ ENABLE CONCURRENT JOBS
- ☐ ENABLE FACT CACHE ?

EXTRA VARIABLES ? **YAML** JSON ☐ PROMPT ON LAUNCH

```
1 vlan_number: 998
2 vlan_name: U_VLAN_998
3 trunk_ports: e1/2/1
```

LAUNCH **CANCEL** **SAVE**

When the template has saved, click **Launch**. When the template has finished executing, you should see yellow status text if the job was successful. This indicates a change has occurred:

Dashboard

- Jobs
- Schedules
- My View
- RESOURCES
- Templates
- Credentials
- Projects
- Inventories
- Inventory Scripts
- ACCESS
- Organizations
- Users
- Teams
- ADMINISTRATION
- Credential Types
- Notifications
- Management Jobs
- Instance Groups
- Applications
- Settings

DETAILS

STATUS: ● Successful

STARTED: 3/24/2020 2:04:10 PM

FINISHED: 3/24/2020 2:04:17 PM

JOB TEMPLATE: ICX VLAN

JOB TYPE: Run

LAUNCHED BY: awxuser

INVENTORY: Ruckus Switches

PROJECT: ICX Test

PLAYBOOK: pbvlan.yml

CREDENTIAL: [ICX Login](#)

ENVIRONMENT: /var/lib/awx/venv/ansible

EXECUTION NODE: awx

INSTANCE GROUP: tower

EXTRA VARIABLES **YAML** JSON **EXPAND**

```
1 trunk_ports: e1/2/1
2 vlan_name: U_VLAN_998
3 vlan_number: 998
4
```

ICX VLAN

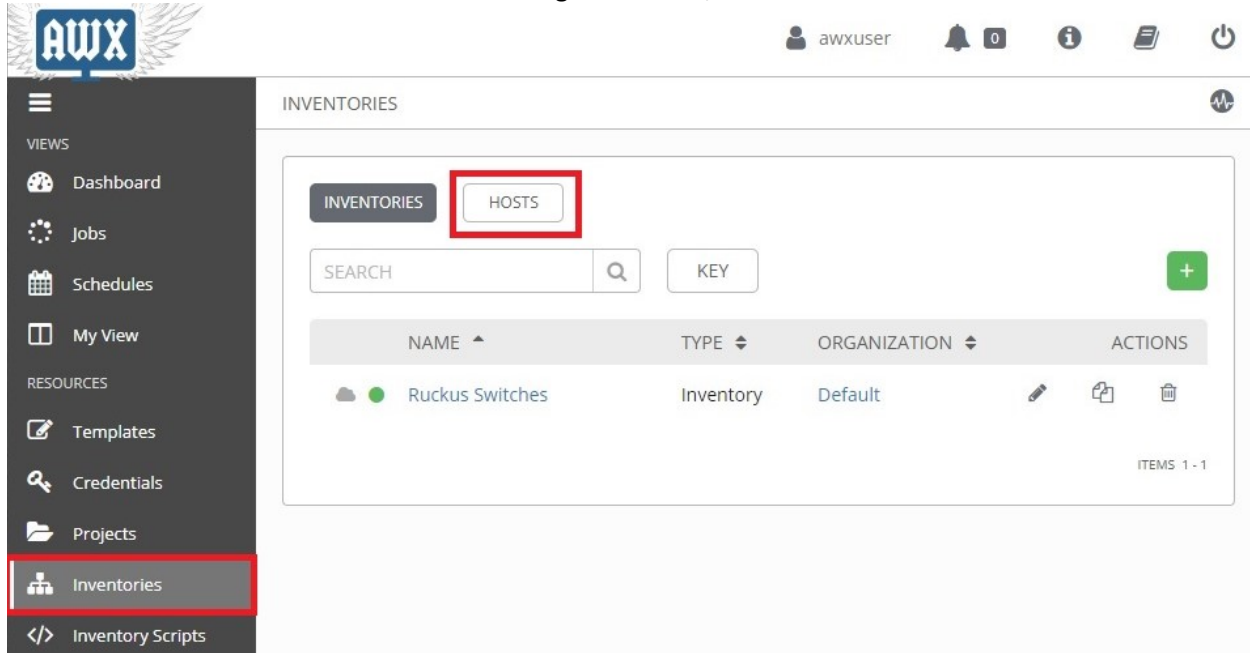
PLAYS: 1 TASKS: 1 HOSTS: 1 ELAPSED: 00:00:06

SEARCH

```
1 SSH password:
2 BECOME password[defaults to SSH password]:
3
4 PLAY [all] ***** 14:04:13
5
6 TASK [create vlan] ***** 14:04:13
7 changed: [NLAB-001-AS-01]
8
9 PLAY RECAP ***** 14:04:17
10 NLAB-001-AS-01 : ok=1 changed=1 unreachable=0 failed=0 skipped=0 rescued=0
11 nored=0
```

For the 'ICX VLAN' template that was created, the 'trunk_ports' variable was defined within the template itself. Suppose there was a group of switches that had a different trunk than the one that was defined in the template. For those switches the template would be tagging the wrong port for the VLAN you are creating. One solution to this would be to define the trunk ports on the **Hosts** page for the switch.

Click the **Inventories** button from the left navigation menu, then click on the **Hosts** tab:

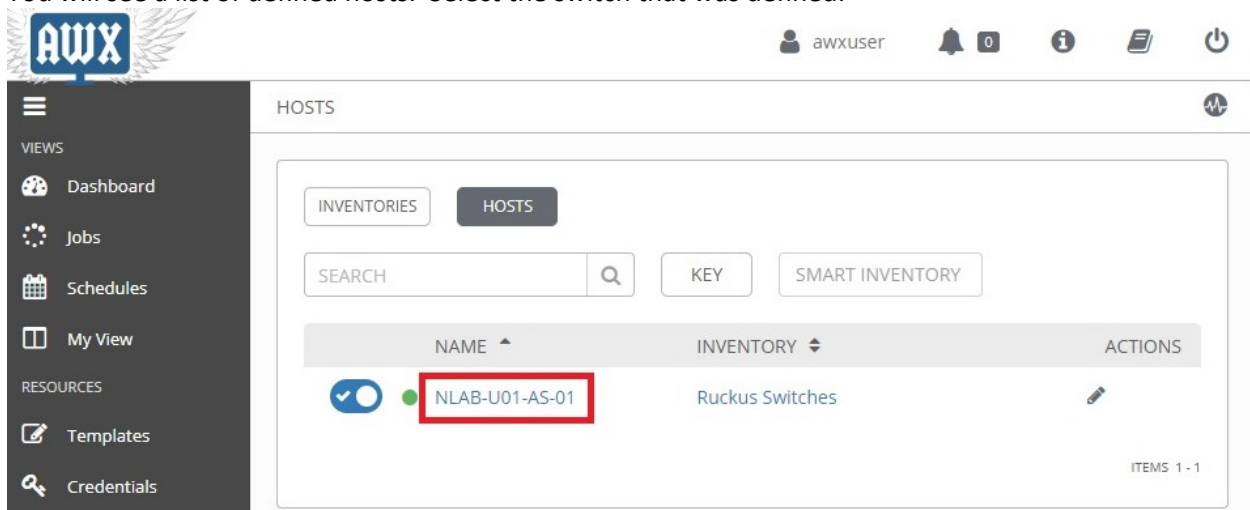


The screenshot shows the AWX web interface. The left navigation menu has 'Inventories' highlighted with a red box. The main content area is titled 'INVENTORIES' and has a 'HOSTS' tab selected, also highlighted with a red box. Below the tabs are search and key input fields, and a table of inventory items.

NAME	TYPE	ORGANIZATION	ACTIONS
Ruckus Switches	Inventory	Default	[Edit] [Copy] [Delete]

ITEMS 1 - 1

You will see a list of defined hosts. Select the switch that was defined:

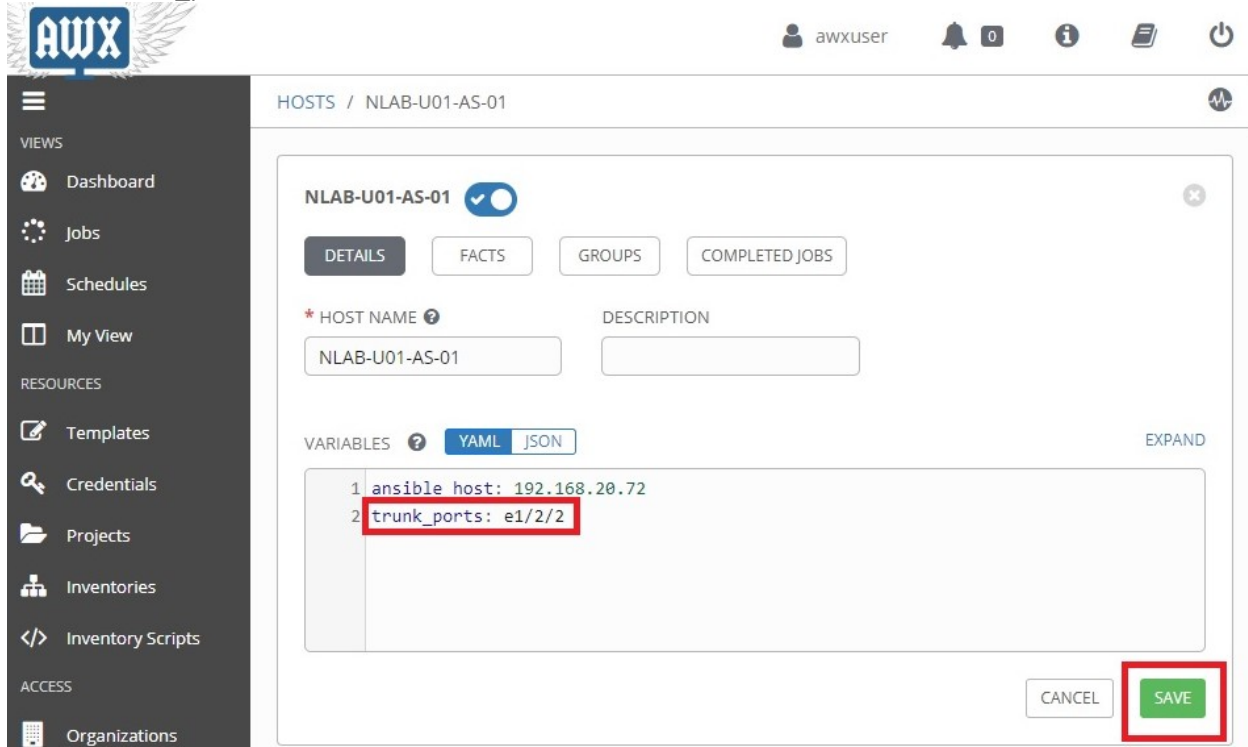


The screenshot shows the AWX web interface with the 'HOSTS' tab selected. The left navigation menu is visible. The main content area is titled 'HOSTS' and has a 'HOSTS' tab selected. Below the tabs are search, key, and smart inventory input fields, and a table of host items.

NAME	INVENTORY	ACTIONS
NLAB-U01-AS-01	Ruckus Switches	[Edit]

ITEMS 1 - 1

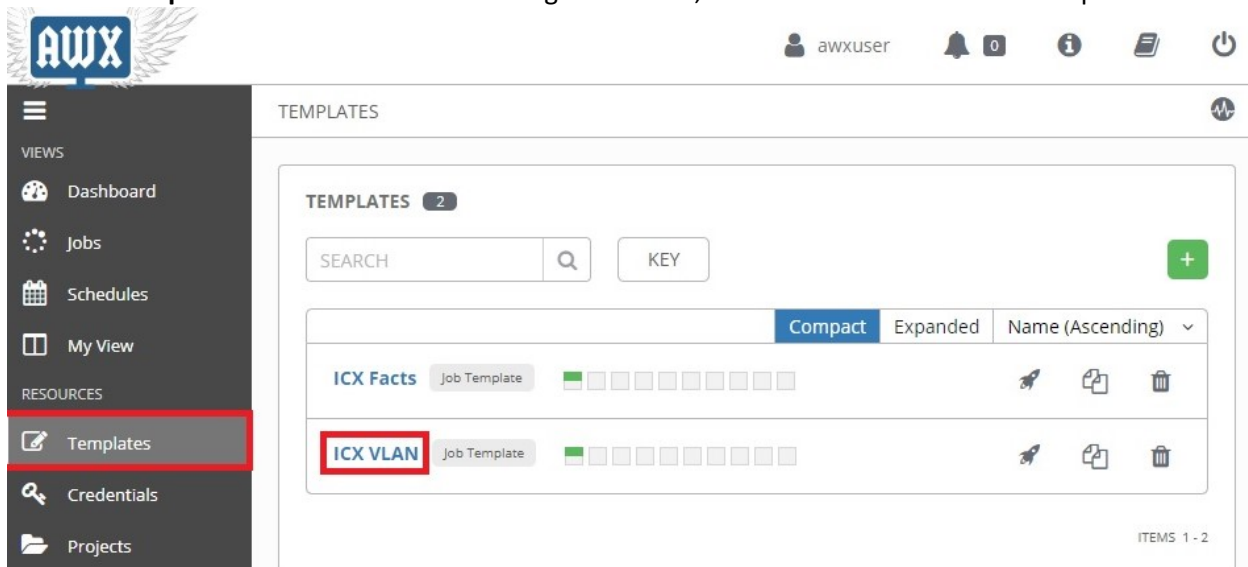
Add the 'trunk_ports' variable as seen below, then click **Save**:



The screenshot shows the AWX interface for configuring a host. The left sidebar contains navigation links for Views (Dashboard, Jobs, Schedules, My View) and Resources (Templates, Credentials, Projects, Inventories, Inventory Scripts, Organizations). The main content area is titled 'HOSTS / NLAB-U01-AS-01'. It features a toggle switch for 'NLAB-U01-AS-01' and tabs for 'DETAILS', 'FACTS', 'GROUPS', and 'COMPLETED JOBS'. The 'DETAILS' tab is active, showing a form for 'HOST NAME' (NLAB-U01-AS-01) and 'DESCRIPTION'. Below this is the 'VARIABLES' section, which is expanded to show a list of variables. The second variable, 'trunk_ports: e1/2/2', is highlighted with a red box. At the bottom right, there are 'CANCEL' and 'SAVE' buttons, with the 'SAVE' button highlighted by a red box.

Some templates might be run multiple times with different variables defined for each run. In this case you might not want to have the variable definitions saved in the template. Templates can be configured to prompt for variables on launch.

Click the **Templates** button from the left navigation menu, then click on the **ICX VLAN** template:



The screenshot shows the AWX interface for managing templates. The left sidebar is the same as in the previous screenshot, but the 'Templates' button is highlighted with a red box. The main content area is titled 'TEMPLATES' and shows a list of templates. The 'ICX VLAN' template is highlighted with a red box. The interface includes a search bar, a 'KEY' button, and a '+ ' button. The templates are listed in a table with columns for 'Name (Ascending)', 'Job Template', and 'Status'. The 'ICX VLAN' template is listed as a 'Job Template' with a status of 'Green'.

Scroll down to **Extra Variables**. Click the checkbox next to **Prompt on Launch**. Remove the variable for 'trunk_ports' and make the 'vlan_number' and 'vlan_name' variables undefined like the picture below, then click **Save**:

AWX

awxuser

TIMEOUT ? 0

SHOW CHANGES ?

PROMPT ON LAUNCH

OPTIONS

- ☒ ENABLE PRIVILEGE ESCALATION ?
- ☐ ENABLE PROVISIONING CALLBACKS ?
- ☐ ENABLE WEBHOOK ?
- ☐ ENABLE CONCURRENT JOBS ?
- ☐ ENABLE FACT CACHE ?

EXTRA VARIABLES ?

YAML JSON

☒ PROMPT ON LAUNCH

```
1 vlan_number:
2 vlan_name:
```

LAUNCH CANCEL **SAVE**

Click the **Launch** button. Define the variables as seen below, then click **Next**:

AWX

awxuser

TIMEOUT ?

SHOW CHANGES ?

PROMPT ON LAUNCH

OPTIONS

OTHER PROMPTS PREVIEW

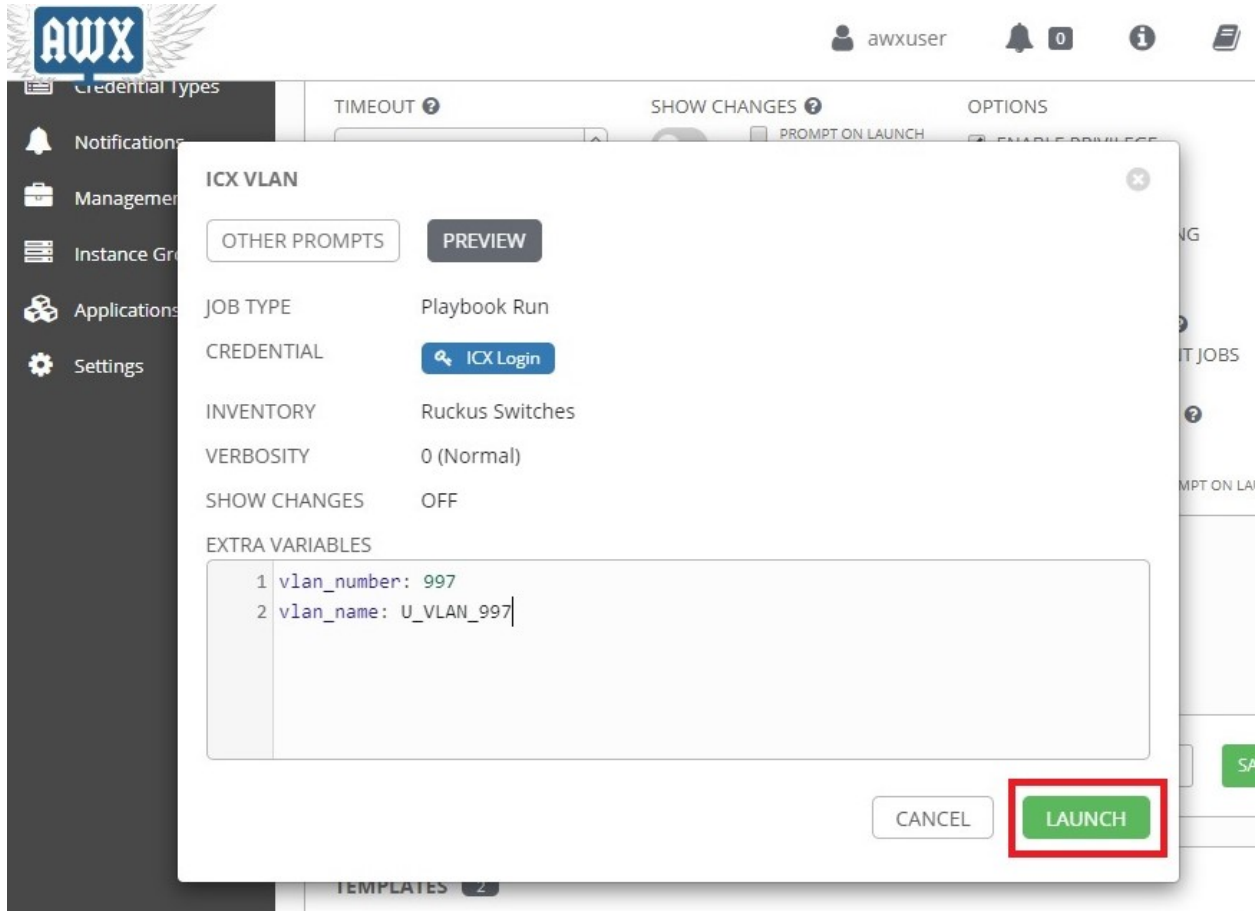
EXTRA VARIABLES ?

YAML JSON

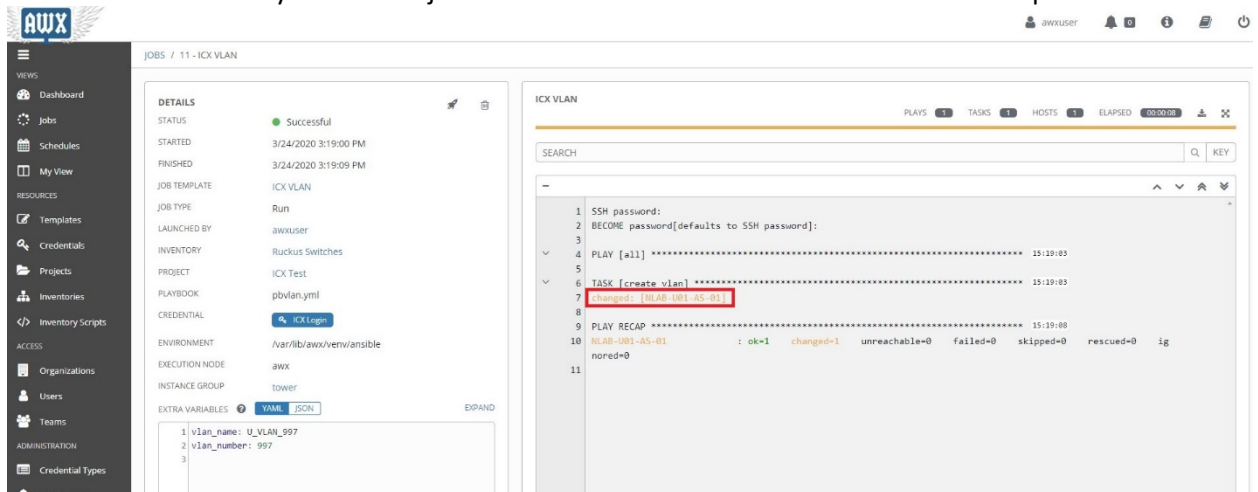
```
1 vlan_number: 997
2 vlan_name: U_VLAN_997
```

CANCEL **NEXT**

You will see the **Preview** window. Click the **Launch** button:



The status text will be yellow if the job execution was successful. Click on the task output:



Here you can see the template was executed with the '*trunk_port*' variable defined under the host:

NLAB-U01-AS-01

CREATED

3/24/2020 3:19:08 PM

ID

98

PLAY

all

TASK

create vlan

MODULE

icx_config

JSON

1 {

2 "changed": true,

3 "commands": [

4 "vlan 997 name U_VLAN_997",

5 "tagged e1/2/2",

6 "spanning-tree",

7 "write memory"

8],

CLOSE

5. Vim Basics

The following can be entered from global mode:

:w	Save
:q	Quit
:q!	Force Quit
a	Enter Insert (edit) Mode
v	Enter Visual Mode

Insert mode allows you to edit a document.

Visual mode allows you to select text.

Hitting the Escape key returns to global mode.

6. References

AWX server install instructions are derived from this site:

<https://www.howtoforge.com/how-to-install-ansible-awx-with-nginx-reverse-proxy-on-ubuntu-1804/>

SSL certificate and key creation tutorial:

<https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-nginx-in-ubuntu-16-04>

Ansible Playbook Tutorial:

https://docs.ansible.com/ansible/latest/network/getting_started/first_playbook.html

7. Additional Resources

AWX Documentation: <https://docs.ansible.com/ansible-tower/latest/html/userguide/index.html>

VIM Cheat Sheet: <https://vim.rtorr.com/>