Ansible Workshop - Exercises

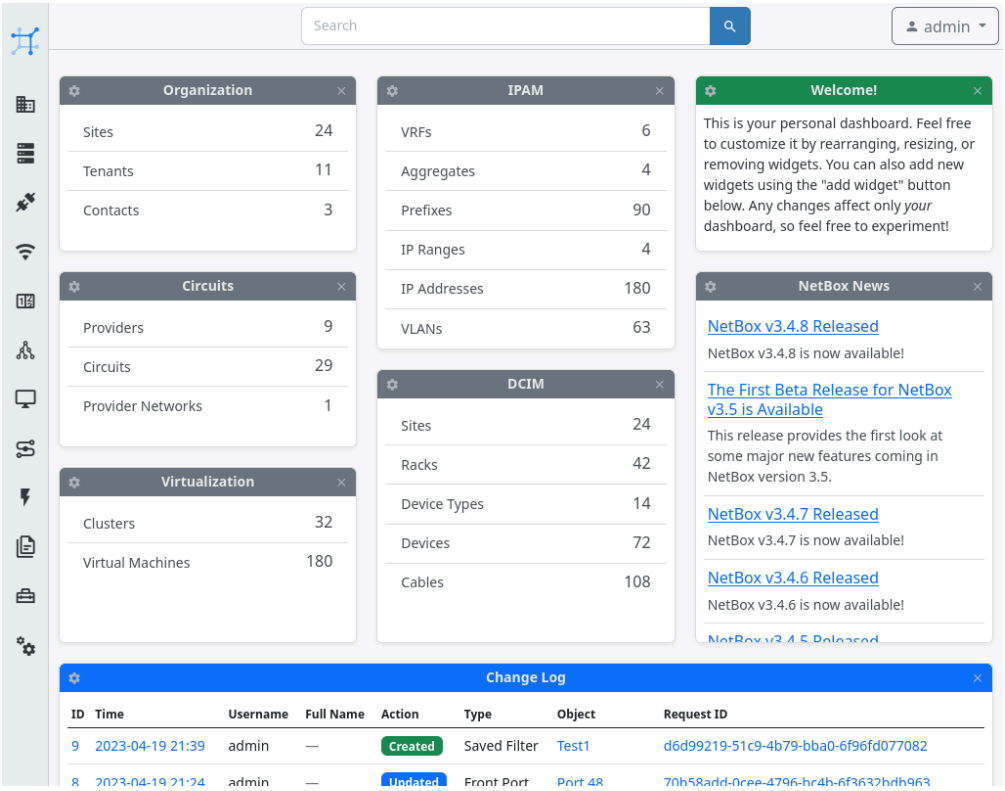
Projects

Use your Ansible skills to complete a couple of small projects.

Project - DCIM Automation

Data**c**enter **I**nventory **M**anagement and IP Address Management are indispensable in today's large data centers. NetBox offers both functions including a versatile API.

NetBox has curated a data model which caters specifically to the needs of network engineers and operators. It delivers a wide assortment of object types to best serve the needs of infrastructure design and documentation. These cover all facets of network technology, from IP address managements to cabling to overlays and more.



Objective

Create an Ansible project *from scratch*, automate against an API and make use of an Ansible *dynamic inventory*.

Guide

Step 1 - Prepare project

Create a new project folder in your home directory:

```
[student@ansible-1 ~]$ mkdir netbox-automation
```

We will be using a Netbox Demo available online.

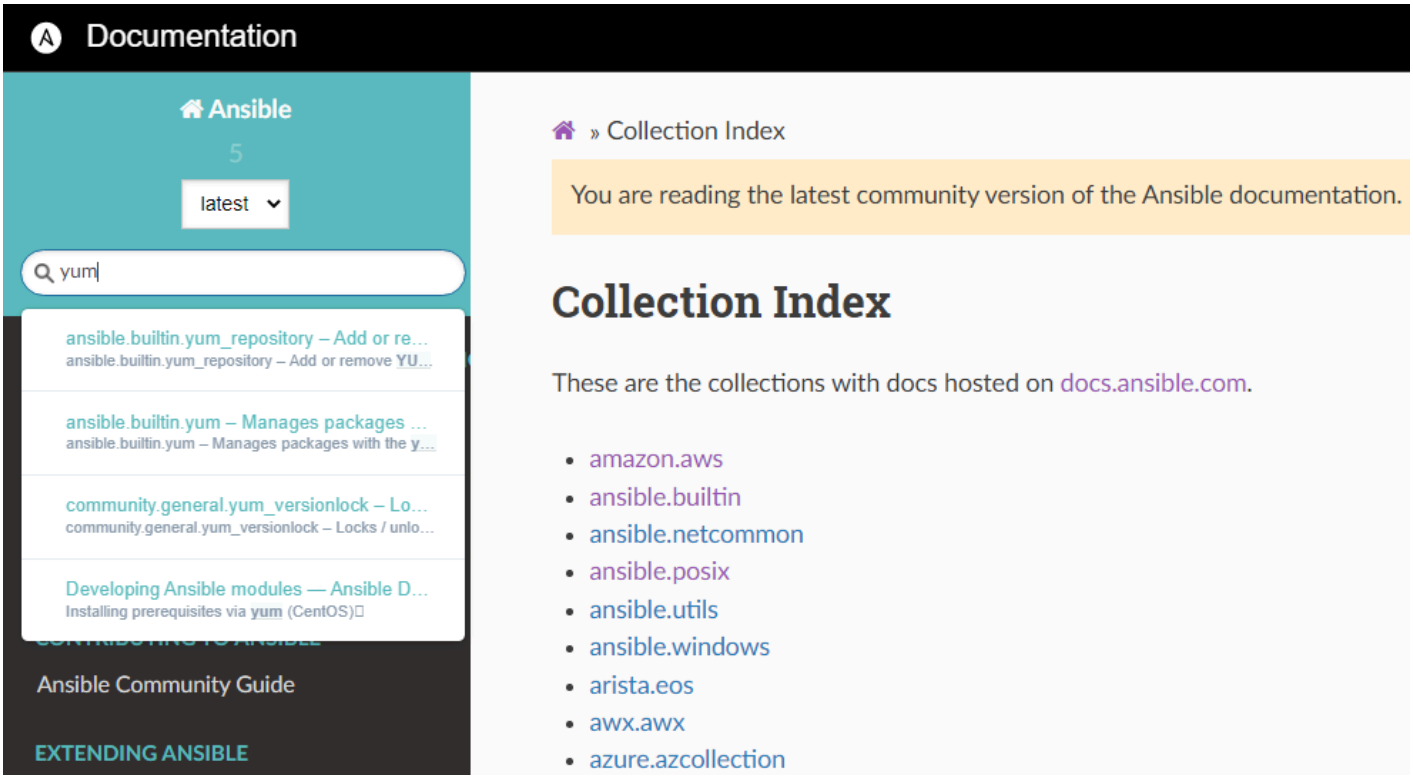
Open a new browser tab and go to https://demo.netbox.dev/.

> **Tip**
>
> You can create personal login credentials yourself. Once logged in, you can create an API token which you will need for your automation tasks. Either use the link or click on your username in the upper right corner of the Netbox UI and select *API Tokens* from the dropdown menu.

Today, you'll need additional Ansible modules. In the first part of the workshop, we only used a handful of modules which are all included in the `ansible-core` binary. With *ansible-core* only 69 of the most used modules are included:

```
[student@ansible-1 ~]$ ansible-doc -l
add_host               Add a host (and alternatively a group) to the ansible-playbook in-memory
inventory
apt                    Manages apt-packages
apt_key                Add or remove an apt key
apt_repository         Add and remove APT repositories
assemble               Assemble configuration files from fragments
assert                 Asserts given expressions are true
async_status           Obtain status of asynchronous task
blockinfile            Insert/update/remove a text block surrounded by marker lines
command                Execute commands on targets
copy                   Copy files to remote locations
...
```

Additional modules are installed through *collections*, search the Collection Index in the Ansible documentation for a module or use the search field.



If, for example, you want to create an EC2 instance in AWS, you will need the module `amazon.aws.ec2_instance`. To get the module, you'll need the collection `aws` of the provider `amazon`. Download the collection with the `ansible-galaxy` utility:

```
[student@ansible-1 ~]$ ansible-galaxy collection install amazon.aws
Starting galaxy collection install process
Process install dependency map
Starting collection install process
Downloading https://galaxy.ansible.com/download/amazon-aws-3.2.0.tar.gz to
/home/student/.ansible/tmp/ansible-local-55382m3kkt4we/tmp7b2kxag4/amazon-aws-3.2.0-3itpmahr
Installing 'amazon.aws:3.2.0' to
'/home/student/.ansible/collections/ansible_collections/amazon/aws'
amazon.aws:3.2.0 was installed successfully
```

Tip

Well, you won't need the AWS collection, but automating the Netbox with Ansible also requires additional modules, these are not included in the `ansible-core` binary and need to be installed with Ansible Galaxy.

Achieve the following tasks:

- Find appropriate collection for Netbox automation in the documentation
- Collection installed

You can view the installed collections with this command:

```
[student@ansible-1 netbox-automation]$ ansible-galaxy collection list
# /home/student/.ansible/collections/ansible_collections
Collection        Version
----------------- -------
ansible.posix     1.4.0
community.docker  2.7.0
community.general 5.3.0
```

Step 2 - Inventory and playbook

Within your newly created project folder, create an inventory file and a playbook file.

Tip

You have to instruct Ansible to communicate with the Netbox API, by default Ansible would try to communicate via SSH. This will not work.
Use the API token you created in the Netbox UI.

Testing the successful communication with the API could be done by querying all available tenants with the `nb_lookup` plugin. Take a look at the documentation for how to use it, use the *search* to find it.
Create your playbook and add a task with the *debug* module, utilizing the *lookup plugin*.
In the documented example the loop uses the query function, instead of `devices` search for `tenant`, the variable to

output can be `{{ item.value.display }}` for the name of the respective tenant.
Run your playbook, if it returns a green *ok* status, communication is established.

Help wanted?

Use the following task to get a list of all already configured tenants.

```
- name: Obtain list of tenants from NetBox
  debug:
    msg: "{{ item.value.display }}"
  loop: "{{ query('netbox.netbox.nb_lookup', 'tenants', api_endpoint=https://demo.netbox.dev/,
token=YOUR_NETBOX_TOKEN) }}"
  loop_control:
    label: "ID: {{ item.key }}"
```

The *loop_control* is not really necessary, but improves readability.

Tip

You need to input your personal API token.

Achieve the following tasks:

- Inventory and playbook created
- Use variables where possible (and useful)
- Successful communication with API established

Step 3 - Create a new Tenant

Most core objects within NetBox's data model support tenancy. This is the association of an object with a particular tenant to convey ownership or dependency.

The goal is to create a new Netbox tenant with Ansible. The tenant should have the following properties, which can be set with the parameters of the appropriate module:

Parameter	Value
name	`Demo Tenant <Initials>`
slug	`demo_tenant_<initials>`
description	`Workshop tenant`
tenant_group	`cc_workshop`

> **Warning**
>
> Replace `<Initials>` with your personal initials to identify the objects later on.

Achieve the following tasks:

- Tenant created
- Tenant is part of `cc_workshop` tenant group
- Inspect tenant in the UI

Step 4 - Create group for VMs

Let's add your three *managed nodes* to a logical group within Netbox. In the Netbox UI, click on *Virtualization*, here you can find *Clusters*.
Find an appropriate module to create a cluster and set the following module parameters:

Parameter	Value
name	`Demo Tenant <Initials> VMs`
site	`rh_demo_environment`
cluster_type	`Amazon Web Services`
group	`EMEA`

Achieve the following tasks:

- Cluster created

Step 5 - Create VMs

A virtual machine (VM) represents a virtual compute instance hosted within a cluster. Each VM must be assigned to a site and/or cluster.
Let's create multiple virtual machine objects, one for every host in your inventory group `web`.
As we need additional information about our VMs (number of vCPU cores, memory, disk space), add a task which *gathers facts* about your managed nodes. Find the appropriate module to do this, Ansible documentation shows you how to do this, the keyword here is *delegating facts*.

Once you gathered all facts about your managed nodes, add a task to create virtual machine objects in the Netbox with a loop, iterating over the `web` group of your inventory.
Find the correct module, every VM object should use the following parameters:

Parameter	Value	Example (rendered to)
name	`"{{ hostvars[item]['ansible_fqdn'] }}"`	*node2.example.com*
site	`rh_demo_environment`	
cluster	`Demo Tenant <Initials> VMs`	*Demo Tenant TG VMs*
tenant	`demo_tenant_<initials>`	*student2*
platform	`"{{ hostvars[item]['ansible_distribution'] | lower }}_{{ hostvars[item]['ansible_distribution_major_version'] }}"`	*Redhat 8*
vcpus	`"{{ hostvars[item]['ansible_processor_vcpus'] }}"`	*2*
memory	`"{{ hostvars[item]['ansible_memtotal_mb'] }}"`	*1024*
disk	`"{{ hostvars[item]['ansible_devices']['nvme0n1']['size'] | split(' ') | first | int }}"`	*10*
virtual_machine_role	`application-server`	
status	`Active`	

Warning

Again, replace `<initials>` with your own Initials.

Achieve the following tasks:

- VM objects for all managed nodes created

© Tim Grützmacher 2025