Guidelines for VMware module development

The Ansible VMware collection (on Galaxy, source code repository) is maintained by the VMware Working Group. For more information see the team community page.

Testing with govcsim

Most of the existing modules are covered by functional tests. The tests are located here.

By default, the tests run against a vCenter API simulator called govcsim. `ansible-test` will automatically pull a govcsim container and use it to set-up the test environment.

You can trigger the test of a module manually with the `ansible-test` command. For example, to trigger `vcenter_folder` tests:

```
source hacking/env-setup
ansible-test integration --python 3.7 vcenter_folder
```

`govcsim` is handy because it is much faster than a regular test environment. However, `govcsim` does not support all the ESXi or vCenter features.

> **Note**
>
> Do not confuse `govcsim` with `vcsim`. `vcsim` is an older and outdated version of vCenter simulator, whereas `govcsim` is new and written in Go language.

Testing with your own infrastructure

You can also target a regular VMware environment. This paragraph explains step by step how you can run the test-suite yourself.

Requirements

- 2 ESXi hosts (6.5 or 6.7)
  - with 2 NIC, the second ones should be available for the test
- a VCSA host
- a NFS server
- Python dependencies:
  - pyvmomi
  - requests

If you want to deploy your test environment in a hypervisor, both VMware or Libvirt work well.

NFS server configuration

Your NFS server must expose the following directory structure:

```
$ tree /srv/share/
/srv/share/
```

```
â”œâ”€â”€ isos
â”‚Â Â  â”œâ”€â”€ base.iso
â”‚Â Â  â”œâ”€â”€ centos.iso
â”‚Â Â  â””â”€â”€ fedora.iso
â””â”€â”€ vms
2 directories, 3 files
```

On a Linux system, you can expose the directory over NFS with the following export file:

```
$ cat /etc/exports
/srv/share  192.168.122.0/255.255.255.0(rw,anonuid=1000,anongid=1000)
```

> **Note**
>
> With this configuration all the new files will be owned by the user with the UID and GID 1000/1000. Adjust the configuration to match your user's UID/GID.

The service can be enabled with:

```
$ sudo systemctl enable --now nfs-server
```

Configure your installation

Prepare a configuration file that describes your set-up. The file should be called :file:`test/integration/cloud-config-vcenter.ini` and based on :file:`test/lib/ansible_test/config/cloud-config-vcenter.ini.template`. For instance, if you have deployed your lab with vmware-on-libvirt:

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\platforms\[ansible-devel][docs][docsite][rst][dev_guide][platforms]vmware_guidelines.rst, line 89); *backlink***
>
> Unknown interpreted text role "file".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\platforms\[ansible-devel][docs][docsite][rst][dev_guide][platforms]vmware_guidelines.rst, line 89); *backlink***
>
> Unknown interpreted text role "file".

```
[DEFAULT]
vcenter_username: administrator@vsphere.local
vcenter_password: !234AaAa56
vcenter_hostname: vcenter.test
vmware_validate_certs: false
esxi1_hostname: esxi1.test
esxi1_username: root
esxi1_password: root
esxi2_hostname: test2.test
esxi2_username: root
esxi2_password: root
```

Using an HTTP proxy

Hosting test infrastructure behind an HTTP proxy is supported. You can specify the location of the proxy server with the two extra keys:

```
vmware_proxy_host: esxi1-gw.ws.testing.ansible.com
vmware_proxy_port: 11153
```

In addition, you may need to adjust the variables of the following var files to match the configuration of your lab. If you use vmware-on-libvirt to prepare your lab, you do not have anything to change.

Run the test-suite

Once your configuration is ready, you can trigger a run with the following command:

```
source hacking/env-setup
VMWARE_TEST_PLATFORM=static ansible-test integration --python 3.7 vmware_host_firewall_manager
```

`vmware_host_firewall_manager` is the name of the module to test.

`vmware_guest` is much larger than any other test role and is rather slow. You can enable or disable some of its test playbooks in main.yml.

Unit-test

The VMware modules have limited unit-test coverage. You can run the test suite with the following commands:

```
source hacking/env-setup
ansible-test units --venv --python 3.7 '.*vmware.*'
```

Code style and best practice

datacenter argument with ESXi

The `datacenter` parameter should not use `ha-datacenter` by default. This is because the user may not realize that Ansible silently targets the wrong data center.

esxi_hostname should not be mandatory

Depending upon the functionality provided by ESXi or vCenter, some modules can seamlessly work with both. In this case, `esxi_hostname` parameter should be optional.

```
if self.is_vcenter():
    esxi_hostname = module.params.get('esxi_hostname')
    if not esxi_hostname:
        self.module.fail_json("esxi_hostname parameter is mandatory")
    self.host = self.get_all_host_objs(cluster_name=cluster_name, esxi_host_name=esxi_hostname)[0]
else:
    self.host = find_obj(self.content, [vim.HostSystem], None)
if self.host is None:
    self.module.fail_json(msg="Failed to find host system.")
```

Example should use the fully qualified collection name (FQCN)

Use FQCN for examples within module documentation. For instance, you should use `community.vmware.vmware_guest` instead of just `vmware_guest`.

This way, the examples do not depend on the `collections` directive of the playbook.

Functional tests

Writing new tests

If you are writing a new collection of integration tests, there are a few VMware-specific things to note beyond the standard Ansible :ref:`integration testing<testing_integration>` process.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\platforms\[ansible-devel][docs][docsite][rst][dev_guide][platforms]vmware_guidelines.rst, line 187); *backlink***
>
> Unknown interpreted text role "ref".

The test-suite uses a set of common, pre-defined vars located in prepare_vmware_tests role. The resources defined there are automatically created by importing that role at the start of your test:

```
- import_role:
    name: prepare_vmware_tests
  vars:
    setup_datacenter: true
```

This will give you a ready to use cluster, datacenter, datastores, folder, switch, dvswitch, ESXi hosts, and VMs.

No need to create too much resources

Most of the time, it's not necessary to use `with_items` to create multiple resources. By avoiding it, you speed up the test execution and you simplify the clean up afterwards.

VM names should be predictable

If you need to create a new VM during your test, you can use `test_vm1`, `test_vm2` or `test_vm3`. This way it will be automatically clean up for you.

Avoid the common boiler plate code in your test playbook

From Ansible 2.10, the test suite uses *modules_defaults*. This module allow us to preinitialize the following default keys of the

VMware modules:

- hostname
- username
- password
- validate_certs

For example, the following block:

```
- name: Add a VMware vSwitch
  community.vmware.vmware_vswitch:
    hostname: '{{ vcenter_hostname }}'
    username: '{{ vcenter_username }}'
    password: '{{ vcenter_password }}'
    validate_certs: 'no'
    esxi_hostname: 'esxi1'
    switch_name: "boby"
    state: present
```

should be simplified to just:

```
- name: Add a VMware vSwitch
  community.vmware.vmware_vswitch:
    esxi_hostname: 'esxi1'
    switch_name: "boby"
    state: present
```

Typographic convention

Nomenclature

We try to enforce the following rules in our documentation:

- VMware, not VMWare or vmware
- ESXi, not esxi or ESXI
- vCenter, not vcenter or VCenter

We also refer to vcsim's Go implementation with `govcsim`. This to avoid any confusion with the outdated implementation.