Test Scenarios

Scenarios are distinct environments that describe a Ceph deployment and configuration. Scenarios are isolated as well, and define what machines are needed aside from any ceph-ansible configuration.

Scenario Files

The scenario is described in a `vagrant_variables.yml` file, which is consumed by `Vagrant` when bringing up an environment.

This yaml file is loaded in the `Vagrantfile` so that the settings can be used to bring up the boxes and pass some configuration to ansible when running.

> **Note:**
>
> The basic layout of a scenario is covered in Layout and conventions There are just a handful of required files, this is the most basic layout.

There are just a handful of required files, these sections will cover the required (most basic) ones. Alternatively, other ceph-ansible files can be added to customize the behavior of a scenario deployment.

`vagrant_variables.yml`

There are a few sections in the `vagrant_variables.yml` file which are easy to follow (most of them are 1 line settings).

- **docker**: (bool) Indicates if the scenario will deploy docker daemons
- **VMS**: (int) These integer values are just a count of how many machines will be needed. Each supported type is listed, defaulting to 0:

  ```
  mon_vms: 0
  osd_vms: 0
  mds_vms: 0
  rgw_vms: 0
  nfs_vms: 0
  rbd_mirror_vms: 0
  client_vms: 0
  iscsi_gw_vms: 0
  mgr_vms: 0
  ```

  For a deployment that needs 1 MON and 1 OSD, the list would look like:

  ```
  mon_vms: 1
  osd_vms: 1
  ```

- **RESTAPI**: (bool) Deploy RESTAPI on each of the monitor(s) restapi: true
- **CEPH SOURCE**: (string) indicate whether a `dev` or `stable` release is needed. A `stable` release will use the latest stable release of Ceph, a `dev` will use `shaman` (http://shaman.ceph.com)
- **SUBNETS**: These are used for configuring the network availability of each server that will be booted as well as being used as configuration for ceph-ansible (and eventually ceph). The two values that are **required**:

  ```
  public_subnet: 192.168.13
  cluster_subnet: 192.168.14
  ```

- **MEMORY**: Memory requirements (in megabytes) for each server, e.g. `memory: 512`
- **interfaces**: some vagrant boxes (and linux distros) set specific interfaces. For Ubuntu releases older than Xenial it was common to have `eth1`, for CentOS and some Xenial boxes `enp0s8` is used. **However** the public Vagrant boxes normalize the interface to `eth1` for all boxes, making it easier to configure them with Ansible later.

> **Warning:**
>
> Do *not* change the interface from `eth1` unless absolutely certain that is needed for a box. Some tests that depend on that naming will fail.

- **disks**: The disks that will be created for each machine, for most environments `/dev/sd*` style of disks will work, like: `[ '/dev/sda', '/dev/sdb' ]`
- **vagrant_box**: We have published our own boxes to normalize what we test against. These boxes are published in Atlas (https://atlas.hashicorp.com/ceph/). Currently valid values are: `ceph/ubuntu-xenial`, and `ceph/centos7`

The following aren't usually changed/enabled for tests, since they don't have an impact, however they are documented here for general knowledge in case they are needed:

- **ssh_private_key_path** : The path to the `id_rsa` (or other private SSH key) that should be used to connect to these boxes.
- **vagrant_sync_dir** : what should be "synced" (made available on the new servers) from the host.
- **vagrant_disable_synced_folder** : (bool) when disabled, it will make booting machines faster because no files need to be synced over.
- **os_tuning_params** : These are passed onto ceph-ansible as part of the variables for "system tunning". These shouldn't be changed.

Vagrantfile

The `Vagrantfile` should not need to change, and it is symlinked back to the `Vagrantfile` that exists in the root of the project. It is linked in this way so that a vagrant environment can be isolated to the given scenario.

hosts

The `hosts` file should contain the hosts needed for the scenario. This might seem a bit repetitive since machines are already defined in vagrant_variables.yml but it allows granular changes to hosts (for example defining an interface vs. an IP on a monitor) which can help catch issues in ceph-ansible configuration. For example:

```
[mons]
mon0 monitor_address=192.168.5.10
mon1 monitor_address=192.168.5.11
mon2 monitor_interface=eth1
```

group_vars

This directory holds any configuration change that will affect ceph-ansible deployments in the same way as if ansible was executed from the root of the project.

The file that will need to be defined always is `all` where (again) certain values like `public_network` and `cluster_network` will need to be defined along with any customizations that ceph-ansible supports.

Scenario Wiring

Scenarios are just meant to provide the Ceph environment for testing, but they do need to be defined in the `tox.ini` so that they are available to the test framework. To see a list of available scenarios, the following command (ran from the root of the project) will list them, shortened for brevity:

```
    $ tox -l
    ...
    luminous-ansible2.4-centos7_cluster
    ...

These scenarios are made from different variables, in the above command there
are 3:
```

- jewel: the Ceph version to test
- ansible2.4: the Ansible version to install
- `centos7_cluster`: the name of the scenario

The last one is important in the *wiring up* of the scenario. It is a variable that will define in what path the scenario lives. For example, the `changedir` section for `centos7_cluster` that looks like:

```
centos7_cluster: {toxinidir}/tests/functional/centos/7/cluster
```

The actual tests are written for specific daemon types, for all daemon types, and for specific use cases (e.g. journal collocation), those have their own conventions as well which are explained in detail in Conventions and Test Files.

As long as a test scenario defines OSDs and MONs, the OSD tests and MON tests will run.

Conventions

Environment configuration

Ansible configuration