Ipv6 Test Cases Status

1. related projects

<table>
<thead>
<tr>
<th>requirement/use case</th>
<th>project</th>
<th>link</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ipv6-enabled OPNFV</td>
<td><a href="https://git.opnfv.org/cgit/ipv6/">https://git.opnfv.org/cgit/ipv6/</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="https://git.opnfv.org/cgit/ipv6/tree/docs/userguide/featureusage.rst">https://git.opnfv.org/cgit/ipv6/tree/docs/userguide/featureusage.rst</a></td>
</tr>
<tr>
<td></td>
<td>Yardstick</td>
<td><a href="https://git.opnfv.org/cgit/yardstick/tree/tests/opnfv/test_cases/opnfv_yardstick_tc027.yaml">https://git.opnfv.org/cgit/yardstick/tree/tests/opnfv/test_cases/opnfv_yardstick_tc027.yaml</a></td>
</tr>
</tbody>
</table>

platform            | installers, SDN controllers, etc |

2. Ipv6 Overlay test cases & Underlay test cases

2.1 overlay test cases status

Tempest/Functest

- 2.1.1 The basic Overlay IPv6 Testing (i.e. tenant IPv6 networks using Neutron) are loaded by Tempest by default in the Smoke Scenario (Tempest Scenario Testing).

  the process is

  1. Create network with subnets:
     1.1. one IPv4 and
     1.2. one or more IPv6 in a given address mode(DHCPv6 (stateless), SLAAC)
  2. Boot 2 VMs on this network
  3. Allocate and assign 2 FIP4
  4. Check that vNICs of all VMs gets all addresses actually assigned
  5. Each VM will ping the other’s v4 private address
  6. If ping6 available in VM, each VM will ping all of the other’s v6 addresses as well as the router’s

- 2.1.2 DHCPv6 specific features api test case for openstack

  Test DHCPv6 specific features using SLAAC, stateless and stateful settings for subnets. Also it shall check dual-stack functionality (IPv4 + IPv6 together).

  The tests include:
  1. generating of SLAAC EUI-64 address in subnets with various settings
  2. receiving SLAAC addresses in combinations of various subnets
  3. receiving stateful IPv6 addresses
  4. addressing in subnets with router
- 2.1.3 networks ipv6 api test for openstack

  such as
  test_create_delete_subnet_with_gw
  test_create_delete_subnet_with_default_gw
  test_create_list_subnet_with_no_gw64_one_network

These can be reused as the basis.

Action: how to show the result on the dashboard, one example in https://build.opnfv.org/ci/job/functest-fuel-baremetal-daily-master/261/console, you can see the ipv6 test result. work with/in functest.

Yardstick

- 2.14 already have Overlay IPv6 Testing for specific use case
  “Service VM as IPv6 vRouter”. https://git.opnfv.org/cgit/yardstick/tree/tests/opnfv/test_cases/opnfv_yardstick_tc027.yaml

  step 1 To setup IPv6 testing environment:

  1. disable security group
2. create (ipv6, ipv4) router, network and subnet
3. create VRouter, VM1, VM2

step 2  To run ping6 to verify IPV6 connectivity:
   1. ssh to VM1
   2. Ping6 to ipv6 router from VM1
   3. Get the result(RTT) and logs are stored

step 3  To teardown IPV6 testing environment
   1. delete VRouter, VM1, VM2
   2. delete (ipv6, ipv4) router, network and subnet
   3. enable security group

• Some hardcoded names, IP addresses etc. need change, of course. test case description will also be added in the test case.

(Note: there should be a JIRA task here for yardstick, https://github.com/openstack/tempest/blob/master/tempest/config.py#L632 can be good examples, such as packetsize, ping_count, etc. During the release time, some problems with the developing, MatthewLi will work on this, parameter options will be added, etc)

2.2 underlay test cases status, Not completed ready, will develop in later releases

3. installers & SDN controller support status

3.1 since all installers are inherited from upstream openstack, for ipv6 with openstack only, see detail in the Gap analysis in ipv6-enabled OPNFV project https://git.opnfv.org/cgit/ipv6/tree/docs/userguide/featureusage.rst

3.2 ODL support Gap analysis is also shown in https://git.opnfv.org/cgit/ipv6/tree/docs/userguide/featureusage.rst, lack of ONOS/opencontrail, etc, waiting for contribution.