NAME
ovs-l3ping – check network deployment for L3 tunneling problems

SYNOPSIS
ovs-l3ping -s TunnelRemoteIP,InnerIP[/mask] -t tunnelmode
ovs-l3ping -s TunnelRemoteIP,InnerIP[/mask][:ControlPort] -t tunnelmode
ovs-l3ping -c TunnelRemoteIP,InnerIP[/mask],RemoteInnerIP -t tunnelmode
ovs-l3ping -c TunnelRemoteIP,InnerIP[/mask][:ControlPort][:DataPort],RemoteInnerIP[:ControlPort][:DataPort] [-b targetbandwidth] [-i testinterval] -t tunnelmode

Common options:
[-h | --help] [-V | --version]

DESCRIPTION
The ovsl3ping program may be used to check for problems that could be caused by invalid routing policy, misconfigured firewall in the tunnel path or a bad NIC driver. On one of the nodes, run ovsl3ping in server mode and on the other node run it in client mode. The client and server will establish L3 tunnel, over which client will give further testing instructions. The ovsl3ping client will perform UDP and TCP tests. This tool is different from ovsltest that it encapsulates XML/RPC control connection over the tunnel, so there is no need to open special holes in firewall.

UDP tests can report packet loss and achieved bandwidth for various datagram sizes. By default target bandwidth for UDP tests is 1Mbit/s.

TCP tests report only achieved bandwidth, because kernel TCP stack takes care of flow control and packet loss.

Client Mode
An ovsl3ping client will create a L3 tunnel and connect over it to the ovsl3ping server to schedule the tests. TunnelRemoteIP is the peer’s IP address, where tunnel will be terminated. InnerIP is the address that will be temporarily assigned during testing. All test traffic originating from this IP address to the RemoteInnerIP will be tunneled. It is possible to override default ControlPort and DataPort, if there is any other application that already listens on those two ports.

Server Mode
To conduct tests, ovsl3ping server must be running. It is required that both client and server InnerIP addresses are in the same subnet. It is possible to specify InnerIP with netmask in CIDR format.

OPTIONS
One of -s or -c is required. The -t option is also required.

-s TunnelRemoteIP,InnerIP[/mask][:ControlPort]
--server TunnelRemoteIP,InnerIP[/mask][:ControlPort]
Run in server mode and create L3 tunnel with the client that will be accepting tunnel at TunnelRemoteIP address. The socket on InnerIP[:ControlPort] will be used to receive further instructions from the client.

-c TunnelRemoteIP,InnerIP[/mask][:ControlPort][:DataPort],RemoteInnerIP[:ControlPort][:DataPort]
--client TunnelRemoteIP,InnerIP[/mask][:ControlPort][:DataPort],RemoteInnerIP[:ControlPort][:DataPort]
Run in client mode and create L3 tunnel with the server on TunnelRemoteIP. The client will use InnerIP to generate test traffic with the server’s RemoteInnerIP.

-b targetbandwidth
--bandwidth targetbandwidth
Target bandwidth for UDP tests. The targetbandwidth must be given in bits per second. It is possible to use postfix M or K to alter the target bandwidth magnitude.
### ovs-l3ping(1)

#### Synopsis

`ovs-l3ping [options]`

#### Options

- `-i testinterval`
  ```
  --interval testinterval
  
  How long each test should run. By default 5 seconds.
  ```

- `-t tunnelmode`
  ```
  --tunnel-mode tunnelmode
  
  Specify the tunnel type. This option must match on server and client.
  ```

- `-h`
  ```
  --help
  
  Prints a brief help message to the console.
  ```

- `-V`
  ```
  --version
  
  Prints version information to the console.
  ```

#### Examples

On host 192.168.122.220 start `ovs-l3ping` in server mode. This command will create a temporary GRE tunnel with the host 192.168.122.236 and assign 10.1.1.1/28 as the inner IP address, where client will have to connect:

```
  ovs-l3ping -s 192.168.122.236,10.1.1.1/28 -t gre
```

On host 192.168.122.236 start `ovs-l3ping` in client mode. This command will use 10.1.1.2/28 as the local inner IP address and will connect over the L3 tunnel to the server’s inner IP address at 10.1.1.1.

```
  ovs-l3ping -c 192.168.122.220,10.1.1.2/28,10.1.1.1 -t gre
```

#### See Also

ovs-vswitchd(8), ovs-ofctl(8), ovs-vsctl(8), ovs-vlan-test(8), ovs-test(8), ethtool(8), uname(1)