**Analysis Report for patch a38f899**

**neutron/agent/dhcp/agent.py**

@@ -535,6 +535,8 @@ class DhcpAgent(manager.Manager):

 LOG.debug("Agent IPs on network %s changed from %s to %s",

 network.id, old\_ips, new\_ips)

 driver\_action = 'restart'

+ elif not old\_ips and not new\_ips:

+ driver\_action = 'disable'

 self.cache.put\_port(updated\_port)

 # NOTE(alegacy): the port may no longer have any IP addresses so if

 # that's the case we need to use the old ones in order to determine

**neutron/agent/linux/dhcp.py**

@@ -274,7 +274,7 @@ class DhcpLocalProcess(DhcpBase):

 self.\_get\_process\_manager().disable()

 if not retain\_port:

 self.\_destroy\_namespace\_and\_port()

- self.\_remove\_config\_files()

+ self.\_remove\_config\_files()

 def \_destroy\_namespace\_and\_port(self):

 try:

@@ -287,13 +287,11 @@ class DhcpLocalProcess(DhcpBase):

 if not ns\_ip.netns.exists(self.network.namespace):

 LOG.debug("Namespace already deleted: %s", self.network.namespace)

 return

- if not ns\_ip.namespace\_is\_empty():

- # Since the namespace is shared amongst all of the VLAN

- # tagged subnets on the network do not delete it unless no

- # other subnets are using it

- return

 try:

- ns\_ip.netns.delete(self.network.namespace)

+ # Since the namespace is shared amongst all of the VLAN tagged

+ # subnets on the network do not delete it unless no other subnets

+ # are using it

+ ns\_ip.garbage\_collect\_namespace()

 except RuntimeError:

 LOG.warning('Failed trying to delete namespace: %s',

 self.network.namespace)

@@ -550,6 +548,10 @@ class Dnsmasq(DhcpLocalProcess):

 self.\_output\_addn\_hosts\_file()

 self.\_output\_opts\_file()

+ def \_namespace\_created(self):

+ ip\_wrapper = ip\_lib.IPWrapper()

+ return ip\_wrapper.netns.exists(self.network.namespace)

+

 def reload\_allocations(self):

 """Rebuild the dnsmasq config and signal the dnsmasq to reload."""

@@ -565,6 +567,10 @@ class Dnsmasq(DhcpLocalProcess):

 LOG.debug('Agent does not have an interface on this network '

 'anymore, skipping reload: %s', self.network.id)

 return

+ elif not self.\_namespace\_created():

+ LOG.debug('Cannot reload allocations prior to namespace '

+ 'creation for network %s', self.network.id)

+ raise exceptions.Conflict()

 self.\_release\_unused\_leases()

 self.\_spawn\_or\_reload\_process(reload\_with\_HUP=True)

@@ -1480,6 +1486,8 @@ class DeviceManager(object):

 # we don't care about any ips on subnets irrelevant

 # to us (e.g. auto ipv6 addresses)

 if fixed\_ip.subnet\_id in dhcp\_subnets]

+ if not fixed\_ips and not self.driver.use\_gateway\_ips:

+ raise exceptions.Conflict()

 ips = [DictModel(item) if isinstance(item, dict) else item

 for item in fixed\_ips]