Foreman - Bug #1861

host macaddress gets reset to the facter value for 'macaddress' when using storeconfig

09/13/2012 06:34 PM - Joshua Hoblitt

Status:	Resolved		
Priority:	Normal		
Assignee:	Ohad Levy		
Category:	Puppet integration		
Target version:			
Difficulty:		**Fixed in Releases:**	
Triaged:		**Found in Releases:**	
Bugzilla link:		**Red Hat JIRA:**	
Pull request:			

Description

I have two populations of 10G connected hosts. The first population pxe boots from eth0 (1GB) and then via provisioning scripts sets up eth2 with the hostname's primary IP (eth0 is actually unconfigured at this point but the ethernet cable is needed for the BMC anyways). This setup is required as the 10G Intel x520 NICs in that population do not support PXE booting. The second population has new generation of motherboards with Intel x540 NICs onboard and they are capable of PXE booting. The problem is that these systems also have 2 1GB interfaces on board that are enumerated first, making the 10G interfaces eth2 & eth3. Putting the mac address of eth2 into foreman works fine for the initial provisioning but after the first time puppet runs on the host the mac address gets reset to that of eth0 (which isn't even connected in this case as the BMC has a dedicated port). This is a problem for re-provisioning hosts in this population as the operator has to remember to manually reset the mac address of the host to that of eth2.

This is running foreman-0.4.2-0.1.noarch with only a couple of one liner fixes applied.

There are actually 4 x 10G interfaces in the second population:

```
[root@pollux3 ~]# facter  | grep mac
macaddress => 00:25:90:7C:FB:36
macaddress_eth0 => 00:25:90:7C:FB:36
macaddress_eth1 => 00:25:90:7C:FB:37
macaddress_eth2 => 00:25:90:7C:FB:9C
macaddress_eth3 => 00:25:90:7C:FB:9D
macaddress_eth4 => 00:1B:21:9D:0F:94
macaddress_eth5 => 00:1B:21:9D:0F:95
```

One possible (but very ugly) solution would be make facter return macaddress == macaddress_eth2 on these systems. I could also add udev rules to change the interface naming (I really hate this option) but the mac address would probably ping pong on the first reboot after the udev rules get applied. Another possible solution would be to add a flag to 'pin' the mac address in foreman so facter/storeconfigs can't update it. Yet another possible solution would be to add the option in foreman of selecting the boot interface.

History

#1 - 09/13/2012 06:58 PM - Joshua Hoblitt

Since I'm not sure what the correct solution is (or even if a solution would be accepted into mainline), I've implemented this very ugly solution for the population with the x540 NICs. Perhaps it will be useful to someone else with the same problem that stumbles over this ticket.

```
class facter::macaddress {

  file { "/etc/profile.d/facter_macaddress.sh":
    owner   => 'root',
    group   => 'root',
    mode    => 0644,
    ensure  => present,
    replace => true,
    content => "export FACTER_macaddress=\"$::macaddress_eth2\"\n",
  }
}
```

#2 - 06/19/2013 08:38 AM - Benjamin Papillon

- Status changed from New to Feedback

Hello,
In the newer foreman builds, there is a setting to turn off updating provisionning informations from facts.
In the provision tab setting, set ignore_puppet_facts_for_provisioning to true.

Does it solve your problem? Did you find it yourself when upgrading?

Regards

#3 - 05/22/2014 11:48 AM - Simon Mügge

I've just stumbeld upon this issue as well, in a slightly different situation, actually BEFORE even being able to start deployment:

All our VM hosts already are in foreman (puppet fact upload), now I have just added all compute resources that provide these VMs.

Within our company, eth0 is used for provisioning in 99% of cases - with VMs that'd be 100%.

Foreman though takes the "main identifying" MACaddress from puppets "macaddress" fact, which in turn seems to take the first lexically sorted match from the list of all other macaddress_$name facts - and in our case, most VMs have these macaddress facts:

```
macaddress_dummy0 = "something"       #virtual NOARP interface
macaddress_eth0 = "something_else"    #actual provisioning-NIC
macaddress = "something "             #from macaddress_dummy0, because "d" comes before "e"
[other]
```

So foreman sets the hosts "main identifying" MAC to that of dummy0 (via "macaddress" fact) - an
d now there is a mismatch between what the compute resource exports (a host with the "main identifying" MAC address with the value from macaddress_eth0), and that "same" hosts discovered MAC (via normal facts), which leads to about half of our VMs not being able to be associated to their compute resources, becuase MAC addresses do not match.

Setting ignore_puppet_facts_for_provisioning does not help here - and we actually do not want that, we want to have puppet update foremans data, including IPs and MACs, so we f.e. can replace a broken eth0 NIC, reboot the machine and foreman automatically knows of the changed MAC behind that interface, and provisioning just keeps working.

The solution I would think best is for foreman to accept an optional, custom fact "foreman_provisioning_mac" whcih gets used in stead of the default "macaddress" fact, if it exists, so we do not have to so something nasty like Joshua Hoblitt suggested.

Then each host could just export the correct MAC to provision to, and we could have nice things like "Yes, all of YOU provision through eth0, but I provision through eth4, always" decided by the host.

To clarify: If I just where to go ahead and hack that default macaddress fact so that it always skips over dummy0 NICs or only ever exports macaddress_eth0, that'd be wrong too, since we do have machines with bonded NICs as the primary interface, or machines where, due to reasons, eth1 is used to provision in stead of eth0.

hth,
Simon

#4 - 05/17/2017 09:33 AM - Anonymous

- Description updated

- Status changed from Feedback to Resolved

storeconfig is long gone