Smart Proxy - Bug #14911

Racing for free IPs resulting in DHCP reservation conflicts

05/03/2016 02:49 AM - Guido Günther

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:	DHCP		
Target version:			
Difficulty:		**Fixed in Releases:**	
Triaged:		**Found in Releases:**	
Bugzilla link:		**Red Hat JIRA:**	
Pull request:			

Description

Hi,

when creating several hosts at a time via the API I'm seeing of DHCP reservation conflicts (and therefore failed deployments). This is using VMWare image based installs and it happens both with internal IPAM and DHCP IPAM. I'm seeing this on the smart proxy:

```
D, [2016-05-02T16:40:39.381767 #20131] DEBUG -- : Searching for free IP- pinging 192.168.0.179
D, [2016-05-02T16:40:40.384515 #20131] DEBUG -- : Found free IP 192.168.0.179 out of a total of 41
4 free IPs
…
D, [2016-05-02T16:40:11.429082 #20131] DEBUG -- : trying to find an ipaddress, we got {:from=>"192
.168.0.64", :to=>"192.168.1.254"
}
D, [2016-05-02T16:40:11.433183 #20131] DEBUG -- : Searching for free IP- pinging 192.168.0.179
D, [2016-05-02T16:40:12.435926 #20131] DEBUG -- : Found free IP 192.168.0.179 out of a total of 41
3 free IPs
…
W, [2016-05-02T16:42:05.408961 #20131]  WARN -- : Request to create a conflicting record
D, [2016-05-02T16:42:05.409021 #20131] DEBUG -- : request:{"filename"=>"pxelinux.0", :hostname=>"f
oo.example.com",:subnet=>192.168.0.0/255.255.254.0, :ip=>"192.168.0.179",:mac=>"00:50:56:98:1e:7d"
}
D, [2016-05-02T16:42:05.409085 #20131] DEBUG -- : local:{:hostname=>"bar.example.com", :mac=>"00:5
0:56:98:0b:2c",  :ip=>"192.168.0.179", :filename=>"pxelinux.0",:subnet=>192.168.0.0/255.255.254.0}
E, [2016-05-02T16:42:05.409253 #20131] ERROR -- : Record 192.168.0.0/192.168.0.179 already exists
D, [2016-05-02T16:42:05.409362 #20131] DEBUG -- :/usr/share/foreman-proxy/modules/dhcp/server.rb:1
22:in `addRecord'
/usr/share/foreman-proxy/modules/dhcp/providers/server/isc.rb:39:in`addRecord'
/usr/share/foreman-proxy/modules/dhcp/dhcp_api.rb:113:in `block in<class:DhcpApi>'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1603:in `call'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1603:in `block in compile!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:966:in `[]'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:966:in `block (3 levels) inroute!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:985:in `route_eval'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:966:in `block (2 levels) inroute!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1006:in `block in process_route'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1004:in `catch'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1004:in `process_route'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:964:in `block in route!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:963:in `each'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:963:in `route!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1076:in `block in dispatch!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1058:in `block in invoke'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1058:in `catch'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1058:in `invoke'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1073:in `dispatch!'
```

```
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:898:in `block in call!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1058:in `block in invoke'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1058:in `catch'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1058:in `invoke'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:898:in `call!'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:886:in `call'
/usr/lib/ruby/vendor_ruby/rack/methodoverride.rb:21:in `call'
/usr/lib/ruby/vendor_ruby/rack/commonlogger.rb:33:in `call'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:217:in `call'
/usr/share/foreman-proxy/lib/proxy/log.rb:58:in `call'
/usr/lib/ruby/vendor_ruby/rack/protection/xss_header.rb:18:in `call'
/usr/lib/ruby/vendor_ruby/rack/protection/path_traversal.rb:16:in `call'
/usr/lib/ruby/vendor_ruby/rack/protection/json_csrf.rb:18:in `call'
/usr/lib/ruby/vendor_ruby/rack/protection/base.rb:50:in `call'
/usr/lib/ruby/vendor_ruby/rack/protection/base.rb:50:in `call'
/usr/lib/ruby/vendor_ruby/rack/protection/frame_options.rb:31:in `call'
/usr/lib/ruby/vendor_ruby/rack/nulllogger.rb:9:in `call'
/usr/lib/ruby/vendor_ruby/rack/head.rb:11:in `call'
/usr/lib/ruby/vendor_ruby/sinatra/show_exceptions.rb:21:in `call'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:180:in `call'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:2014:in `call'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1478:in `block in call'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1788:in `synchronize'
/usr/lib/ruby/vendor_ruby/sinatra/base.rb:1478:in `call'
/usr/lib/ruby/vendor_ruby/rack/builder.rb:138:in `call'
/usr/lib/ruby/vendor_ruby/rack/urlmap.rb:65:in `block in call'
/usr/lib/ruby/vendor_ruby/rack/urlmap.rb:50:in `each'
/usr/lib/ruby/vendor_ruby/rack/urlmap.rb:50:in `call'
/usr/lib/ruby/vendor_ruby/rack/builder.rb:138:in `call'
/usr/lib/ruby/vendor_ruby/rack/handler/webrick.rb:60:in `service'
/usr/lib/ruby/2.1.0/webrick/httpserver.rb:138:in `service'
/usr/lib/ruby/2.1.0/webrick/httpserver.rb:94:in `run'
/usr/lib/ruby/2.1.0/webrick/server.rb:295:in `block in start_thread'
```

It seems Foreman is asking for an IP from the smart-proxy and the server hands out the IP twice in a short time frame while it (or even better foreman itself) should lock the IP since it's already about to create a machine with it. Just retriggering the deployment after the failure works as expected.

Is this a known race condition on parallel vm creation? I searched the tracker and couldn't find anything related.

This is Foreman 10.2 but I didn't spot any changes in this area in more recent versions but may have missed them.

History

#1 - 05/03/2016 02:51 AM - Ivan Necas

- Project changed from Foreman Remote Execution to Foreman

- Category set to DHCP

#2 - 05/03/2016 02:53 AM - Marek Hulán

- Description updated

#3 - 08/04/2016 02:06 PM - Guido Günther

I have poked at this a bit more and it's not DHCP only. If I retry DHCP I can also get

```
fatal: [localhost]: FAILED! => {"changed": false, "failed": true, "msg": "Failed to create host somehost: [u'C
onflict DNS PTR Records 10.0.0.10/anotherhost.example.com already exists', u'Conflict DNS PTR Records 10.0.0.1
0/anotherhost.example.com already exists']"}
```

The problem is that when using IP autosuggest several hosts get the same autosuggested IP which then fails. I think this can only be solved by:

- taking a lock
- call unused_ip()
- make unused_ip() store the IP in a InFlightIPs table
- releasing the lock

unused_ip() would also consult InFlightIPs and request a new one if the returned on is already in the table.

InFlightIPs would be cleared once the host is created, creation failed or after a fixed time interval to get rid of stale entries.

This way creating hosts in parallel would become race free with only a short window that has to take a lock. Does this make any sense?

#4 - 08/08/2016 03:05 AM - Dominic Cleal

- Project changed from Foreman to Smart Proxy

- Category changed from DHCP to DHCP

The smart proxy is meant to retain a lock on the IP for a period to prevent it being reallocated.

#5 - 08/16/2016 12:50 AM - Guido Günther

Dominic Cleal wrote:

> The smart proxy is meant to retain a lock on the IP for a period to prevent it being reallocated.

I've seen this with both DHCP and Internal IPAM. In the later case the SP has no way to reserve the IP I guess?

#6 - 08/16/2016 03:08 AM - Dominic Cleal

Guido Günther wrote:

> Dominic Cleal wrote:
>
> > The smart proxy is meant to retain a lock on the IP for a period to prevent it being reallocated.
>
> I've seen this with both DHCP and Internal IPAM. In the later case the SP has no way to reserve the IP I guess?

No, internal IPAM in Foreman would probably reassign the same IP as it doesn't use the smart proxy.

#7 - 11/20/2017 10:33 PM - Anonymous

- Status changed from New to Closed

This has been resolved in http://projects.theforeman.org/issues/20173, closing the issue.