Foreman - Bug #33585

Puma memory usage after update to 2.5

09/29/2021 12:10 PM - Adam Winberg

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

Description

After update from Foreman 2.4 to 2.5 (2.5.3) we have seen a dramatic increase in memory usage. From using about 5/8G of RAM before the update the machine more or less immediately after started swapping and getting OOM'd . Increasing RAM from 8 to 12G did not help, neither did changing puma threads from 0/16 to 5/5. After decreasing puma workers from 6 to 4 the memory situation looked a lot better, but usage is steadily increasing over time and after a couple of days the server started swapping again.

So, with 6 puma workers memory usage was a steady ~5G with Foreman 2.4.
With Foreman 2.5 12G RAM is not sufficient even for 4 puma workers.

Attaching graph showing memory and swap usage over time.
Update to Foreman 2.5 was at Aug 31.
RAM increase from 8 to 12G was at Sep 8.
Puma worker decrease from 6 to 4 was at Sep 24.

Related issues:

Related to Foreman - Bug #33640: Improve caching of SettingRegistry#load_values	**Closed**
Related to Foreman - Bug #33639: Reduce allocations in FactImporter#update_facts	**Closed**
Related to Foreman - Bug #34093: Authorizer cache loads all permitted resourc...	**Closed**

History

#1 - 10/01/2021 07:17 AM - Lukas Zapletal

Adam, can you share more about your deployment? Be as detailed as possible. We are looking for: OS system, OS tuning, Ruby tuning, Foreman plugins installed, hardware or VM specs.

#2 - 10/01/2021 07:32 AM - Adam Winberg

We are running on RHEL8, vSphere VM with 10cpu/12G RAM.
No special OS or Ruby tuning. Puma threads set to 5/5 (min/max) and workers set to 4.

Foreman installed via rpms, configured via Puppet module (latest version).

Foreman plugins:
foreman_templates
foreman_statistics
foreman_snapshot_management
foreman_remote_execution
foreman_hooks
foreman-tasks

#3 - 10/05/2021 07:10 AM - Lukas Zapletal

And just to confirm, was this upgrade 2.4 (puma) to 2.5 (puma) or some other version (with passenger)?

#4 - 10/05/2021 07:23 AM - Adam Winberg

Yes, 2.4 -> 2.5 (puma to puma).

Since my last restart of Foreman on Aug 29 the memory situation looks a bit more stable. After a day or so the usage reached about 9G but has been steady since then. In conjunction with the restart I also removed one plugin which we no longer use - rubygem-smart_proxy_dhcp_infoblox. Don't know if that has anything to do with it.

Still, 9G memory usage with 4 workers is a lot more than 5G with 6 workers. The memory usage in itself is however not a problem for us, as long as its not increasing and causing OOM's. I will continue to monitor the usage.

#5 - 10/07/2021 08:10 AM - Lukas Zapletal

- Status changed from New to Need more information

- Triaged changed from No to Yes

If you spot a controller/action that reports too many allocations in production.log (e.g. "Completed 200 OK in 97ms (Views: 52.1ms | ActiveRecord: 13.4ms | Allocations: 55457)" this could help us to determine which code has an increase of number of allocations which lead to memory fragmentation in Ruby. Digging in old logs pulling out the old allocation numbers would be a huge help.

Red Hat Performance team just finished conducting large-scale testing of Satellite 6.10-beta (Foreman 2.5 with Katello and other plugins) and the recommendation for small-scale deployment up to 5000 hosts is: 4 cores, 20GB, 4-6 workers with 16 threads. While the subject of testing was host registration via subscription-manager, it can give you some context.

#6 - 10/07/2021 09:00 AM - Lukas Zapletal

You might be interested in #33639 as well. Try it out it's a small patch.

#7 - 10/07/2021 10:25 AM - Adam Winberg

Unfortunately I don't have logs from before the version update. But I will try the patch and see if that has any effect in my environment, will report back.

#8 - 10/07/2021 10:36 AM - Adam Winberg

Regarding the recommendations from the Red Hat study I am a bit confused - according to
https://projects.theforeman.org/issues/33277

threads should be max 5 and the old default of 16 was due to a misunderstanding of the puma documentation.

#9 - 10/11/2021 06:01 AM - Adam Winberg

> You might be interested in #33639 as well. Try it out it's a small patch.

I have applied the patch but there is no difference in memory usage. After about one day the usage is up to ~9G again

#10 - 10/11/2021 06:42 AM - Lukas Zapletal

Adam, we have figured another regression and fixed it over the weekend: #33640.

Try to apply that one and get back to us. Both are heading into the 3.0 release as well.

#11 - 10/11/2021 06:42 AM - Lukas Zapletal

- Related to Bug #33640: Improve caching of SettingRegistry#load_values added

#12 - 10/11/2021 06:43 AM - Lukas Zapletal

- Related to Bug #33639: Reduce allocations in FactImporter#update_facts added

#13 - 10/11/2021 11:29 AM - Adam Winberg

ok, I have applied the patch. Will report back with results.

#14 - 10/12/2021 09:30 AM - Adam Winberg

No effect in my environment unfortunately

#15 - 10/13/2021 06:20 AM - Lukas Zapletal

Can you isolate which controller/actions have the highest Allocation counts in production.log? Use grep to pull out "Allocations: DDDDDDDDD" with highest amount of digits.

#16 - 10/13/2021 02:32 PM - Adam Winberg

The ones that really stand out are these API requests:

```
Started GET "/api/organizations/1/hosts?per_page=999999&search=os~ELIN" for 192.168.8.23 at 2021-10-13 04:30:2
4 +000
...
Completed 200 OK in 25884ms (Views: 22477.5ms | ActiveRecord: 2679.3ms | Allocations: 8619853)
```

The request results in a host list containing ca. 1000 hosts. This request is however only run once/day and our memory graphs show no increased usage during this.

Second 'worst' are host overview requests:

```
Started GET "/hosts/lxserv1946/overview" for 10.66.1.95 at 2021-10-13 14:01:14 +0000
...
Completed 200 OK in 10167ms (Views: 9142.5ms | ActiveRecord: 986.9ms | Allocations: 3341257)
```

These requests does not show any increased memory usage either in the graphs.

Looking at the graphs, there are several 'jumps' in memory usage after restart of the foreman service. Trying to correlate these jumps with events in the log they often seem to coincide with a specific host parameter edit. This parameter is a long multiline string.

Following is a log excerpt from such an instance, where memory usage suddenly increased with 500M:

```
2021-10-13T13:30:44 [I|app|db7fb46a] Started PATCH "/hosts/3109" for 10.66.1.95 at 2021-10-13 13:30:44 +0000
2021-10-13T13:30:44 [I|app|db7fb46a] Processing by HostsController#update as */*
2021-10-13T13:30:45 [I|aud|db7fb46a] Nic::Managed (9398) update event on ip6 , 
2021-10-13T13:30:46 [I|aud|db7fb46a] LookupValue (3385) update event on value # Add package names, one per row
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
 db7fb46a | <retracted>
2021-10-13T13:30:47 [I|app|db7fb46a] Processed 1 tasks from queue 'Host::Managed Post', completed 1/1
2021-10-13T13:30:47 [I|app|db7fb46a] Redirected to https://foreman.example.com/hosts/lxserv1946
2021-10-13T13:30:47 [I|app|db7fb46a] Completed 302 Found in 3394ms (ActiveRecord: 204.5ms | Allocations: 433406)
```

#17 - 10/14/2021 05:57 AM - Lukas Zapletal

Oh my, why on Earth are you loading all your hosts onto one page? I mean, obviously there is a bug we need to fix, 22 seconds is not acceptable, unless you have BMC turned on for hosts, in that case Foreman needs to do BMC operation per host to show power state icons on the index page. Can you confirm?

If you need to list all hosts daily, can't you use hammer? It also has an export feature to CSV. Alternatively, there is a reporting mechanism that allows you to run daily reports and create customized outputs (JSON, XML, CSV...)

Can you confirm this is the slowest action/controller so far? Then I will rename this issue accordingly.

#18 - 10/14/2021 06:05 AM - Lukas Zapletal

What is so specific about lxserv1946 that it loads so slow? Is that amount of NICs? Can this be for example a docker container or hypervisor with hundreds of NICs reported via facts?

#19 - 10/14/2021 06:16 AM - Adam Winberg

As I said there is no indication that this API call is in fact causing the increased memory usage. So regardless of how 'bad' this practice might be it has worked fine with previous version (memory-wise, can't say if it was faster though but that is not the issue here) and I do not think it has anything to do with the the increased memory usage in 2.5.

And no, we do not use BMC feature.

Regarding lxserv1946 there is nothing special about it, it has one NIC and is a 'normal' VM running on vSphere. What do you mean that it 'loads so slow'? The log excerpt is a PATCH which takes ~3s which does not strike me as particularly slow.

Looking at all the memory jumps in the graphs almost all coincide with the same smart class parameter being edited, but on different hosts. So it is not the host that is the common denominator but rather the parameter. However I can't be certain that it is this particular parameter that is causing the memory usage increase since users rarely edit any other parameters - this is the one that is mostly used so I don't have much to compare with. I will try some testing though.

This particular parameter is as I said very widely used and has ~900 overrides. It is a multiline string which in some cases can be pretty long (1-200 lines).

#20 - 10/19/2021 08:36 AM - Lukas Zapletal

Okay this is out of my knowledge zone, I am trying to poke other community devs:
https://community.theforeman.org/t/increased-memory-on-2-5-with-smart-class-params/25757

Feel free to drop details there, you should have better luck on our forums than here. But you provided good data and there is a good chance folks will take it from here.

#21 - 12/03/2021 12:21 PM - Lukas Zapletal

Not much in the thread I've created. Any news? Were you able to isolate something?

#22 - 12/17/2021 07:25 AM - Adam Winberg

We are running with 12G RAM and 4 puma workers and the server has been stable for some while now. So I don't know if there was some special circumstance causing our outages earlier or if it has just stabilized over time. I am loath to tinker any more with this since it is a production environment, so as long as it works I am happy. So for me it is ok to close this issue, if we encounter any further problems we can re-open it.

#23 - 12/20/2021 08:19 AM - Lukas Zapletal

- Related to Bug #34093: Authorizer cache loads all permitted resources to memory added

#24 - 12/20/2021 08:21 AM - Lukas Zapletal

- Status changed from Need more information to Closed

Thanks for getting back to us.

We have identified one memory problem recently, it will only happen when you are signed in as a regular user (not admin) and on some pages it can generate big allocation numbers due to the way we check permissions. For more info: #34093.

I am closing for now, thanks.

Files

Screenshot from 2021-09-29 14-07-16.png	86.2 KB	09/29/2021	Adam Winberg