Foreman - Feature #17737

Optionally return structured facts nested under the root fact via API, not individually

12/19/2016 09:39 AM - Leonardo Sorrentino

Otation	Nau	
Status:	New .	
Priority:	Normal	
Assignee:		
Category:	API	
Target version:		
Difficulty:		Fixed in Releases:
Triaged:		Found in Releases:
Bugzilla link:		Red Hat JIRA:
Pull request:		
Description		
I have a user with the 'viewer' role, and if I try for example:		
<pre>curl -k -u username:password "https://localhost/api/v2/fact_values?search=name=disks" python -mj son.tool I get the following response:</pre>		
<pre>{ "page": 1, "per_page": 30, "results": { "disks": null }, "hostP": { "disks": null }, "hostC": { "disks": null }, "hostD": { "disks": null }, "hostE": { "disks": null }, "hostF": { "disks": null }, "sort": { "disks": null }, "sort": { "disks": null }, "sotbtotal": 7, "total": 0 } </pre>		
In the "results" fields, it is supposed to show the "disks" fact content, it shows a null value instead. This problem seems to be limited to structured facts only.		

The field "Total" is also always 0 with non-Administrator users. If the query is from an Administrator it always returns a fixed value:

"total": 60232
Which is exactly the record count in my fact_values table:
MariaDB [foremandb]> select count(*) from fact_values;
+-----+
| count(*) |
+----++
| 60232 |
+----++
1 row in set (0.01 sec)
It was working in Foreman 1.12.4, it broke after the upgrade to 1.13.2

History

#1 - 12/19/2016 09:43 AM - Dominic Cleal

- Category set to API

In the "results" fields, it is supposed to show the "disks" fact content, it shows a null value instead. This problem seems to be limited to structured facts only.

Structured top-level facts probably don't have a value, hence the null. Are you expecting to return facts that *start* with the keyword disk? If so you might want something like name ~ disk% to search prefixes.

It was working in Foreman 1.12.4, it broke after the upgrade to 1.13.2

Note that 1.13 stores structured facts from Puppet as structured data now so you may need to change your search query to match.

The totals is probably a different issue to this, not sure that it's related.

#2 - 12/19/2016 11:05 AM - Leonardo Sorrentino

- Subject changed from Wrong Total and Null values when querying structured facts via API to Null values when querying structured facts via API

Dominic Cleal wrote:

In the "results" fields, it is supposed to show the "disks" fact content, it shows a null value instead. This problem seems to be limited to structured facts only.

Structured top-level facts probably don't have a value, hence the null. Are you expecting to return facts that *start* with the keyword disk? If so you might want something like name ~ disk% to search prefixes.

I was expecting the same structured data from the fact named exactly "disks", something like following output from facter:

```
{
  "disks": {
    "sda": {
        "model": "PERC H310",
        "size": "136.13 GiB",
        "size_bytes": 146163105792,
        "vendor": "DELL"
    }
}
```

Now, if I change the query as you suggest, I get the following result:

```
"HostA": {
    "disks": null,
    "disks::sda": null,
    "disks::sda::size": "136.13 GiB",
    "disks::sda::size_bytes": "146163105792"
}
```

Which is a slightly different structure. Also, as far I understand, changing the search operator to ~ will not just find any fact that starts with "disks", but will find all that ones containing the word "disks".

An example with the term "network":

```
curl -s -k -u username:password "https://localhost/api/v2/fact_values?search=name~network" | python -mjson.too
1
{
    "page": 1,
    "per_page": 30,
    "results": {
        "HostX": {
            "ec2_metadata::network": null,
            "ec2_metadata::network::interfaces": null,
            "ec2_metadata::network::interfaces::macs": null,
            "ec2_metadata::network::interfaces::macs::XX:XX:XX:XX:XX:XX": null,
            "ec2_metadata::network::interfaces::macs::XX:XX:XX:XX:XX:XX:ipv4-associations": null
        },
        "HostY": {
            "networking": null,
            "networking::interfaces": null,
            "networking::interfaces::bond0": null,
            "networking::interfaces::eth0": null,
            "networking::interfaces::lo": null
        },
[...]
```

Note that 1.13 stores structured facts from Puppet as structured data now so you may need to change your search query to match.

I read this, I didn't expect a different behaviour on how the data is exposed from the API.

It seems also that some structured facts are not correctly handled, for example:

```
curl -s -k -u user:password "https://localhost/api/v2/fact_values?search=name~processors" | python -mjson.tool
{
    "page": 1,
    "per_page": 30,
    "results": {
        "HostA": {
            "processors": null
        },
        "HostB": {
            "processors": null
        },
        "HostC": {
            "processors": null
        },
    [..]
   },
    "search": "name~processors",
    "sort": {
        "by": null,
       "order": null
   }.
    "subtotal": 30,
    "total": 0
}
```

It is not consistent with a similar call to /hosts/\$host/facts :

```
curl -s -k -u user:password "https://localhost/api/v2/hosts/HostA/facts?search=name~processors" | python -mjso
n.tool
{
    "page": 1,
    "per_page": 30,
    "results": {
        "HostA": {
            "processors": null,
            "processors::count": "24",
            "processors::isa": "x86_64",
            "processors::isa": "x86_64",
            "processors::models": "[\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2
630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon
```

```
eon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-26
30L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeo
on(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L
0L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\",
\"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz\", \"Intel(R) Xeon(R) CPU E5-2630L L 0 @ 2.00GHz\"]",
```

```
"processors::physicalcount": "2",
    "processors::speed": "2.50 GHz"
    },
    "search": "name~processors host = HostA",
    "sort": {
        "by": null,
        "order": null
},
    "subtotal": 1,
    "total": 0
```

Which doesn't look like properly structured (the array processors::models is escaped-flattened into a string)

The totals is probably a different issue to this, not sure that it's related.

Reported as a new issue http://projects.theforeman.org/issues/17741

#3 - 12/20/2016 03:14 AM - Dominic Cleal

- Status changed from New to Feedback

}

```
Now, if I change the query as you suggest, I get the following result:

"HostA": {

"disks": null,

"disks::sda": null,

"disks::sda::size": "136.13 GiB",

"disks::sda::size_bytes": "146163105792"
```

Which is a slightly different structure.

Yes, this looks correct. The facts are stored separately with this notation by Foreman, they're not stored as a single hash, so the API will not return nested hashes. This method also allows individual facts to be specified.

Also, as far I understand, changing the search operator to ~ will not just find any fact that starts with "disks", but will find all that ones containing the word "disks".

Use name ~ disks% (see https://theforeman.org/manuals/1.13/index.html#4.1.5Searching for more details).

It seems also that some structured facts are not correctly handled, for example: "processors": null

At least with a single host I'm unable to reproduce this, I see all processor facts returned:

```
"foreman.example.com": {
    "processors": null,
    "processors::count": "1",
    "processors::physicalcount": "1",
    "processors::models": "[\"Intel Core Processor (Broadwell)\"]",
    "processors::isa": "x86_64"
}
```

Which doesn't look like properly structured (the array processors::models is escaped-flattened into a string)

Foreman doesn't support array facts, issue #15522

Setting to feedback as I think I've addressed your points - if there's a specific bug still present then the description can be updated or file it separately to this discussion. Cheers.

#4 - 01/12/2017 12:05 PM - Leonardo Sorrentino

Dominic Cleal wrote:

Now, if I change the query as you suggest, I get the following result: "HostA": { "disks": null, "disks::sda": null, "disks::sda::size": "136.13 GiB", "disks::sda::size_bytes": "146163105792"

Which is a slightly different structure.

Yes, this looks correct. The facts are stored separately with this notation by Foreman, they're not stored as a single hash, so the API will not return nested hashes. This method also allows individual facts to be specified.

So, as far I understand, top keys are returned as Null, and individual facts are returned with a value.

From a user point of view, IMHO, returning Null is a bit misleading and not being able to retrieve (anymore) a structured fact from its root key is a bit inconvenient. Would this be a feature request?

Also, as far I understand, changing the search operator to ~ will not just find any fact that starts with "disks", but will find all that ones containing the word "disks".

Use name ~ disks% (see https://theforeman.org/manuals/1.13/index.html#4.1.5Searching for more details).

As the doc states:

[...] For example, the search name ~ corp% will match both "corp" and "corporation"

This is something slightly different from what I am trying to achieve.

#5 - 01/13/2017 03:32 AM - Dominic Cleal

- Tracker changed from Bug to Feature

- Subject changed from Null values when querying structured facts via API to Optionally return structured facts nested under the root fact via API, not individually

- Status changed from Feedback to New

From a user point of view, IMHO, returning Null is a bit misleading and not being able to retrieve (anymore) a structured fact from its root key is a bit inconvenient. Would this be a feature request?

Sure, changed to a feature request.