openQA Project - action #91542

coordination # 91646 (Blocked): [saga][epic] SUSE Maintenance QA workflows with fully automated testing, approval and release

coordination # 89062 (Blocked): [epic] Simplify review for SUSE QAM

openQA API what jobs were/are testing X incident/package

2021-04-21 18:23 - okurz

Status:	Resolved	**Start date:**	2021-04-21
Priority:	High	**Due date:**	
Assignee:	ilausuch	**% Done:**	0%
Category:	Feature requests	**Estimated time:**	0.00 hour
Target version:	Ready		
Difficulty:			

Description

Motivation

Based on chat between okurz and bzoltan1 , linked to #91338 there is a desire to find out from openQA "what jobs were/are testing X incident/package". The use case is not fully clear to okurz though. However okurz considers this not too hard to do, e.g. like the reverse of comments on maintenance.suse.de, e.g. https://maintenance.suse.de/incident/19067/#comments . For "incident" that would be different than package but looking in the openQA database for "what jobs are linked to incident $nr" is trivial. And for package the same is possible, openQA can give a list of "what jobs have package $package in the name"

Acceptance criteria

- **AC1:** An example openQA API call exists yielding all jobs linked to a specified incident
- **AC2:** Same as AC1 but for package

Suggestions

- Take a look on https://openqa.opensuse.org/group_overview/70 how incident tests are identified. E.g. https://openqa.opensuse.org/tests/1708946#settings shows build ":16134:gnuhealth.1619074869" where there is one additional interesting package "INSTALL_PACKAGES" with value "gnuhealth gnuhealth-orthanc"
- Compare to OSD as well where we have multiple products in maintenance at the same time
- Come up with an example API call with optional processing or filtering so that anyone that can implement that e.g. into maintenance.suse.de can use such example.

Further details

I (okurz) thinks that the design is not the best because incidents are triggered by "openQABot" and the encoding of incident and package is only within the build by convention with custom selected separator characters ":" and ".". A likely alternative seems to be to implement such functionality into openQABot which also triggers the tests accordingly but maybe we would like to avoid that to keep maintenance effort and complexity of the bot lower.

History

#2 - 2021-04-22 19:10 - okurz

- Description updated

- Status changed from New to Workable

- Priority changed from Normal to High

- Target version changed from future to Ready

- Parent task set to #89062

#3 - 2021-04-22 20:40 - okurz

- Description updated

#4 - 2021-04-23 09:28 - geor

I would like to add here that it would be very useful to be able to query, out of those jobs that are testing X incident, only their last run.

Also to be able to get only the failed last runs.

#5 - 2021-04-23 11:09 - okurz

Yes, I suggest that we only yield the latest job for each

#6 - 2021-04-27 10:12 - ilausuch

- Assignee set to ilausuch

#7 - 2021-04-27 10:46 - ilausuch

- Status changed from Workable to In Progress

#8 - 2021-04-28 04:15 - openqa_review

- Due date set to 2021-05-12

Setting due date based on mean cycle time of SUSE QE Tools

#9 - 2021-04-28 13:06 - ilausuch

Following the suggestions I found this way to get the latest jobs (ids) that have failed and have the issue 14071

```
openqa-cli api --host http://localhost:9526 /jobs limit=1000 latest=1 |  jq '.[][] |  select(.settings |to_ent
ries[]| .value | contains("14071")) | select(.result == "failed")  | .id'
```

#10 - 2021-04-29 08:55 - ilausuch

Now this PR (https://github.com/os-autoinst/openQA/pull/3870) is only a draft and there are missing things but is the API approach to this feature

#11 - 2021-04-29 09:26 - okurz

As the definition of "ISSUE" is something that SUSE QAM invented for their own internal use cases but there are other objects called "issue" within openQA itself the proposed API change would be a bad idea. I like the idea to introduce new API routes for good use cases but for now I suggest for this ticket to focus on work outside openQA source code. Do you consider the code snippet in #91542#note-9 enough to cover AC1? Feel welcome to crosscheck with users.

#12 - 2021-04-29 12:11 - okurz

ilausuch wrote:

> Following the suggestions I found this way to get the latest jobs (ids) that have failed and have the issue 14071
>
> ```
> openqa-cli api --host http://localhost:9526 /jobs limit=1000 latest=1 |  jq '.[][] |  select(.settings |to
> _entries[]| .value | contains("14071")) | select(.result == "failed")  | .id'
> ```

this will easily fail on OSD which runs many jobs so the limit of 1000 won't be enough. A much more limited initial query result can be achieved by looking for the right build, e.g. to look for tests for incident ":19364:shim". one can do:

```
openqa-cli api --osd jobs build=":19364:shim" limit=1000 latest=1 | jq '.[][] | "\(.name): https://openqa.suse
.de/t\(.id) (\(.result))"'
```

which gives output in format like:

```
"sle-15-SP2-Desktop-DVD-Incidents-x86_64-Build:19364:shim-qam-allpatterns@64bit: https://openqa.suse.de/t59164
80 (softfailed)"
"sle-15-SP2-Desktop-DVD-Incidents-x86_64-Build:19364:shim-qam-gnome@64bit: https://openqa.suse.de/t5916481 (so
ftfailed)"
…
"sle-15-SP1-Server-DVD-Incidents-Minimal-x86_64-Build:19364:shim-qam-minimal@uefi: https://openqa.suse.de/t591
6833 (failed)"
```

this of course can be easily customized or filtered down further.

#13 - 2021-04-29 21:07 - ilausuch

Thanks for the ideas

This returns the last failed jobs than has any issue 17 (this is an example, should be a complete issue value) and are done, and the result are failed

```
openqa-cli api --host http://localhost:9526 /jobs limit=1000 latest=1 |  jq  '.[][] |  select(.settings |to_en
tries[]| .value | contains("17")) | select(.result == "failed") | select(.state == "done" ) | {date: (.t_finis
hed + "Z" | fromdate), id: .id, t_finished: .t_finished}' | jq --slurp 'group_by(.id) | .[] | last'
```

```
{
  "date": 1583845259,
  "id": 3976352,
  "t_finished": "2020-03-10T13:00:59"
}
{
  "date": 1583851807,
  "id": 3976455,
  "t_finished": "2020-03-10T14:50:07"
}
{
  "date": 1583852617,
  "id": 3976457,
  "t_finished": "2020-03-10T15:03:37"
}
{
  "date": 1583860604,
  "id": 3976572,
  "t_finished": "2020-03-10T17:16:44"
}
{
  "date": 1583861145,
  "id": 3976730,
  "t_finished": "2020-03-10T17:25:45"
}
```


#14 - 2021-04-30 10:31 - tinita

I'm curious. if you do select(.state == "done"), why not use that as a parameter for the API call?
/jobs limit=1000 latest=1 state=done

#15 - 2021-04-30 10:36 - ilausuch

Good point. Will do. The reason is because I was using a output.json file to no interact with the API all the time

#16 - 2021-04-30 11:07 - ilausuch

I had a chat with geor and some things came up

- He need to choose more than one options for result 'failed' but also 'parallel_failed' for now
- About what tinita mentioned, could in the openqa_cli command line could we specify multiple result values (OR)?
- Remove the date to epoch thing is not necessary

#17 - 2021-04-30 12:37 - okurz

ilausuch wrote:

> Good point. Will do. The reason is because I was using a output.json file to no interact with the API all the time

And that makes sense if multiple requests are done based on the same data set.

#18 - 2021-04-30 12:38 - okurz

ilausuch wrote:

> - About what tinita mentioned, could in the openqa_cli command line could we specify multiple result values (OR)?

I don't think this is necessary but I think it's not necessary. As you already did you can gather the data from openQA in a more generic query and filter that down with e.g. jq locally

#19 - 2021-05-03 08:43 - ilausuch

Well,
Based on the comments this could be a good approach

```
openqa-cli api --host http://localhost:9526 /jobs limit=1000 latest=1 > output.json
```

```
cat output.json |  jq  '.[][] |  select(.settings |to_entries[]| .value | contains("17")) | select(.result == "failed" or .result == "parallel_failed") | select(.state == "done" ) | {id: .id, t_finished: .t_finished}' | jq --slurp 'group_by(.id) | .[] | last'
```

#20 - 2021-05-03 10:59 - ilausuch

Based on the solution for issues this solves the same problem for packages

```
cat output.openseuse.json | jq  '.[][] |  select(.settings |to_entries[]| .value | contains("yast2-network"))
| select(.result == "failed" or .result == "parallel_failed") | select(.state == "done" ) | {id: .id, t_finish
ed: .t_finished, group: .group, name: .name}
```

Where results are:

```
...
{
  "id": 1725317,
  "t_finished": "2021-05-03T10:05:10",
  "group": "openSUSE Tumbleweed",
  "name": "microos-Tumbleweed-MicroOS-Image-ContainerHost-x86_64-Build20210502-container-host-old2microosnext@
64bit"
}
{
  "id": 1725471,
  "t_finished": "2021-05-03T10:29:46",
  "group": null,
  "name": "opensuse-Tumbleweed-DVD-x86_64-ltp_kernel_misc:investigate:retry@64bit"
}
```

#21 - 2021-05-03 11:03 - ilausuch

Additionally the group of packages can be included, such as "REPO_OSS_SOURCE_PACKAGES"

```
cat output.openseuse.json | jq  '.[][] | select(.settings | to_entries[] | .key | contains("REPO_OSS_SOURCE_PA
CKAGES")) | select(.settings |to_entries[]| .value | contains("yast2-network")) | select(.result == "failed" o
r .result == "parallel_failed") | select(.state == "done" ) | {id: .id, t_finished: .t_finished, group: .group
, name: .name}'
```

#22 - 2021-05-04 08:41 - ilausuch

- Status changed from In Progress to Feedback

Waiting for George to confirm results

#23 - 2021-05-06 14:05 - ilausuch

With George we identified two different scenarios for the (issues) AC

1. Single incident jobs: Where each build is related only with one incident
2. Aggregated jobs: Where each build can contains more than one incident.

We extract information using tree API calls

```
openqa-cli api --host http://openqa.suse.de /jobs limit=5000 result="passed" latest=1 >output.passed.json
openqa-cli api --host http://openqa.suse.de /jobs limit=5000 result="failed" latest=1 >output.failed.json
openqa-cli api --host http://openqa.suse.de /jobs limit=5000 result="softfailed" latest=1 >output.softfailed.j
son
```

And we created an initial script for each scenario

1. Single jobs

```
#!/bin/bash
#Usage: $0 <issue>

filter_group="select(.group != null) | select(.group | contains(\"Incident\"))"
issue="select(.settings | to_entries[] | .value | contains(\"$1\"))"

failed=$(cat output.failed.json | jq ".[][] | $filter_group | $issue | .name " | sort | uniq)

for name in $failed; do
  echo $name
  failed_ts=$(cat output.failed.json | jq ".[][] | $filter_group | $issue | select(.name==$name)" | jq --slurp
 'group_by(.name) | .[] | last | .t_finished+"Z" | fromdate')
  job_id=$(cat output.failed.json | jq ".[][] | $filter_group | $issue | select(.name==$name)" | jq --slurp 'g
roup_by(.name) | .[] | last | .id')
  passed_ts=$(cat output.passed.json | jq ".[][] | $filter_group | $issue | select(.name==$name)" | jq --slurp
 'group_by(.name) | .[] | last | .t_finished+"Z" | fromdate')
  softfailed_ts=$(cat output.softfailed.json | jq ".[][] | $filter_group | $issue | select(.name==$name)" | jq
 --slurp 'group_by(.name) | .[] | last | .t_finished+"Z" | fromdate')
```

```
  [[ ! $passed_ts ]] && passed_ts=0
  [[ ! $softfailed_ts ]] && softfailed_ts=0

  [[ $failed_ts -gt $passed_ts ]] && [[ $failed_ts -gt $softfailed_ts ]] && echo "Not passed $job_id"
done
```

1. Aggregated jobs

```
#!/bin/bash

#Usage: $0 <issue> <group>

add_hash='.hash=(.settings.DISTRI+"-"+.settings.VERSION+"-"+.settings.FLAVOR+"-"+.settings.TEST+"-"+.settings.ARCH+"-"+.settings.MACHINE)'
filter_group="select(.group != null) | select(.group | contains(\"$2\"))"
issue="select(.settings | to_entries[] | .value | contains(\"$1\"))"

failed=$(cat output.failed.json | jq ".[][] | $filter_group | $issue | $add_hash | .hash")

for hash in $failed; do
  echo $hash
  failed_ts=$(cat output.failed.json | jq ".[][] | $filter_group | $issue | $add_hash | select(.hash==$hash)" | jq --slurp 'group_by(.hash) | .[] | last | .t_finished+"Z" | fromdate')
  job_id=$(cat output.failed.json | jq ".[][] | $filter_group | $issue | $add_hash | select(.hash==$hash)" | jq --slurp 'group_by(.hash) | .[] | last | .id')
  passed_ts=$(cat output.passed.json | jq ".[][] | $filter_group | $issue | $add_hash | select(.hash==$hash)" | jq --slurp 'group_by(.hash) | .[] | last | .t_finished+"Z" | fromdate')

  [[ ! $passed_ts ]] || [[ $failed_ts -gt $passed_ts ]] && echo "Not passed $job_id"
done
```

The aggregated script is already confirm, but the single one is pendent.
Also these are initial scripts than can be optimized.

#24 - 2021-05-06 14:14 - ilausuch

Talking with George, we think that the second AC should be an other ticket because of

- It's a different problem with different solutions
- To solve that is needed to access to the comments in the tests source code

#25 - 2021-05-06 14:38 - cdywan

ilausuch wrote:

> Talking with George, we think that the second AC should be an other ticket because of
>
> - It's a different problem with different solutions
> - To solve that is needed to access to the comments in the tests source code

I feel like you went way overboard here. The AC was "example API call" and you provide two nice looking scripts :-D

For the second one I'd suggest the search API route which can filter by Package: if that's what you want here, for instance:

```
cat https://openqa.opensuse.org/api/v1/experimental/search?q=Package:
```

#26 - 2021-05-06 15:16 - ilausuch

- Status changed from Feedback to Resolved

I confirmed with George that all provided here is enough to change the status of this ticket to resolved

#27 - 2021-05-07 11:59 - cdywan

FYI as a follow-up I suggested a workshop topic about openQA API. And I think @bzoltan georilausuch one of you guys should file some follow-up tickets with concrete user stories (think about where the API would be used e.g. in scripts, curl, smelt, openQA UX)

#28 - 2021-05-17 09:26 - okurz

- Due date deleted (2021-05-12)

#29 - 2021-05-24 22:06 - okurz

Workshop has been conducted, recording https://youtu.be/EfXZKbQS-Kg