# RCA for 2019-01-07 Read-Only Mount Outage

## Summary

On January 7th, 2019 around 14:18PM UTC, 33 production servers had their in-memory data synced to disks and the disks remounted in RO (read-only) mode and this caused a short duration of production outage across the services. The sync-and-remount operation was being validated as part of a maintenance activity for staging environment. Unfortunately, an erroneous command issued ended up touching the unintended production servers as well. Rebooting the servers mounted the disks back in RW (read-write) mode and it remedied the immediate outage. This was a manual, user error that shouldn't have happened, we deeply apologize to all of our customers and as part of the RCA we will prevent similar issues from happening again in the future.

1. Service(s) affected :

- pages
- patroni, postgres
- prometheus
- pubsub
- package

2. Team attribution : Infrastructure

3. Minutes down : 20~ minutes

## Impact & Metrics

Start with the following:

- What was the impact of the incident? Ultimately, during the downtime of 20 mins, all services that use postgres to write data were down.
- Who was impacted by this incident? All users.
- How did the incident impact customers? Majority of the reported impact to users was that they were getting 503s on .com. However, any operation that would have triggered a write operation to postgres would have been impacted.
- How many attempts were made to access the impacted service/feature? Since one of the affected services was Prometheus, we don't have datapoints that we can look back and provide as a datapoint. However, a 15-min snapshot prior to the incident shows that our Frontend fleet were responding 2XXs at a rate of 1.8K per 30 seconds [1] (which is 60 tps). If we apply the 60 tps to the downtime of 20 mins, it adds up to around 72000 attempts that potentially were made to access the impacted service.
- How many customers were affected? Ultimately, any customer who would have tried to use the services would have been impacted.
- How many customers tried to access the impacted service/feature? What we know so far is that about 80~ tweets reported the issue @gitlabstatus handle [2] and 60~ tweets reported the issue at @gitlab handle [3].

Include any additional metrics that are of relevance.

We should also consider the impact of the incident to GitLab team members

- During the incident, 8 alerts were fired and paged on-call
- About 10 team members, at most, were in the Incident Zoom Call at once
- Post the incident recovery, 6 alerts fired off for postgres-dr-delayed, TooManyDeadTuples, Replication Lag, Prometheus, StatementTimeOut and High CPU which required on-call and team to work on as well.
- In addition, we sent out about 50+ tweets during and majority of them post the incident corresponding back to tweets sent by our customers. This is an added labor/resource impact.

## Detection & Response

Start with the following:

- How was the incident detected? At 14:18 UTC, our monitors caught the issue and PagerDuty alerts 5435, 5436 and 5437 fired off and paged on-call.
- Did alarming work as expected? Yes.
- How long did it take from the start of the incident to its detection? 3-4 mins
- How long did it take from detection to remediation? 8-9 mins to initial recovery and another 5-6 mins for the rest of the patroni/postgres and pubsub nodes. However, it is important to note that 3 other services: Pages, Packages and Prometheus were addressed later on as soon as the immediate high impacting services were restored
- Were there any issues with the response to the incident? (i.e. bastion host used to access the service was not available, relevant team memeber wasn't page-able, ...) No. On-Call jumped on the issue right away, isolated the issue and another SRE member pointed out to the RO issue right away as well.

## Timeline

- 1 Closed **RCA for 2019-01-07 Read-Only Mount Outage**
- 14:18 UTC - Alerts started coming in and paged on-call
- 14:20 UTC - Tweeted and let users know we are investigating the issue
- 14:21 UTC - Incident channel opened and zoom started
- 14:22 UTC - Detected no postgres process was running on patroni nodes (and filesystems being read-only was observed)
- 14:31 UTC - Recovery Postgres instances have been restarted, postgres successfully performs recovery
- 14:33 UTC - Tweeted and let users know we identified an issue and working on repairing
- 14:38 UTC - Rebooted the remaining affected hosts
- 14:42 UTC - Tweeted and let users know we repaired the DB backend and are monitoring the health of the system
- 14:44 UTC - Tweeted and let users know Pages would be in degraded performance
- 14:47 UTC - Package server was also having the issue
- 14:55 UTC - Tweeted .com and Pages are operational and we were monitoring
- 15:01 UTC - Rebooted package server
- 15:07 UTC - Tweeted saying all services are operational
- 15:31 UTC - Dead tuple alerts because of missing statistics, @abrandl: kicked off `ANALYZE` => done 16:18 UTC, stats are up to date

## Root Cause Analysis

### The problem: GitLab.com outage occurred on January 7th, 2019 between 14:18UTC - 14:38UTC

1. **Why?** - A set of services couldn't do any write operations.

   Details: Immediate symptoms were that pgbouncer was throwing errors and no postgres process was running on our patroni nodes.

2. **Why?** - A set of filesystems were remounted in RO (read-only) mode.

   Details: This occurred on 33 servers and these servers covered services: patroni/postgres, pages, pubsub, prometheus and package and thus any transaction that would have needed a write operation would have failed.

3. **Why?** - An erroneous command was ran with intention to touch a single staging host, but ended up touching the production servers as well as staging and ops servers.

   Details: As part of an ongoing kernel maintenance activity, we needed a way to gracefully reboot servers that have dedicated data disks to prevent potential data-loss. Therefore, we were considering to use `/proc/sysrq-trigger` command to issue `s` (for sync), `u` (for remounting in RO mode) and `b` (for rebooting) after updating a server's kernel. Once the sequence was manually validated on a single, staging server, the next step was to incorporate the script to a bigger automation script which handles other operations such as talking to GCE, handling GCE custom metadata, handling LB state of nodes (i.e draining traffic) and running in parallelization mode.

4. **Why?** - When the single staging host name was provided to the script, instead of treating the name as an element of a list the host name itself was iterated over its characters.

   Details: The automation script is in Python. The targeted staging host was: `patroni-01-db-gstg`. This string should have been put in a list [] because once the actual automation runs, it would work with a list of servers and would loop through them. Unfortunately, while manually providing a single host name it was not put in a list. Thus, the iteration logic started with a `p`. However, the real issue was that inside the iteration a `knife ssh` command is ran to run the bash commands to do the `sync`, `remount in RO` and `reboot` and the search criteria for the `knife ssh` was `fqdn:HOST_NAME*`. And in this case, the HOST_NAME was `p` instead of `patroni-01-db-gstg`. Therefore, the command was ran against all hosts that 1) are managed by Chef 2) starts with a 'p'. As soon as this was realized via a printed debug lines, the script was killed.

5. **Why?** - GCE VM name and Chef FQDN for a server differ (i.e: `service-01-sv-gstg vs service-01-sv-gstg.c.gitlab-staging-1.internal`). And the recipe for the storm was:

- Server name captured from GCE was provided to the automation script
- The server name not getting contained in a list [] in the script
- The wild-card search criteria ended up with this unintended result

  Deatails: Our provisioning is done via terraform. We set kernel version in tf. This then gets set as a custom metadata in GCE VM properties. As part of starting up a VM, a startup script runs and validates this metadata against currently running kernel and make sure they are identical. Therefore, given that we control our kernel version on GCE it makes sense to pull a list of servers, that need kernel updated, from GCE. Pulling the list of servers can be done through 1) GCP Console 2) gcloud 3) some script that uses GCE SDK. However, in any of these means, a VM node name returns as, for example, `service-01-sv-gstg`. There is no `.c.gitlab-staging-1.internal`, though this can be retrieved from the custom metadata section. Once a GCE node name is retrieved, doing a wild-card match on FQDN via knife yielded the correct results until this incident. But this could have been prevented if we hadn't used the wild-card match and relied on the actual FQDN. This will be covered under corrective action items further.

## What went well

- Our alerts worked well. First user reported the issue was at 14:19 UTC, while our alarms triggered at 14:18 UTC
- On-Call SRE jumped on the issue immediately and isolated the incident
- A

- Team members convened fast, jumped on zoom, communicated well, tracked progress and remediated the immediate issue without any challenges

## What can be improved

### Technical improvements

- Avoid using wild-card search criteria as part of `knife ssh` if an exact-match option is present or use it only if the search output is first validated
- GCE VM names could be same as Chef FQDNs
- Run remote ssh commands through something other than `knife ssh` (Ansible?)

### Incident Management improvements

- One of the feedback we received was that Status page wasn't updated even though we were tweeting. [4], [5]. If we had tweeted via status.io, we would have tweeted as well as updated our status page.

### Others

- Is there anything that could have been done to improve the detection or time to detection? No. I believe our detection was good timing wise and alerting wise.
- Is there anything that could have been done to improve the response or time to response? No. OnCall jumped on the issue immediately.
- Is there an existing issue that would have either prevented this incident or reduced the impact? No. And this was a user error. But we have corrective action items for prevention.
- Did we have any indication or beforehand knowledge that this incident might take place? This was an unintended result through running an erroneous command. Therefore, there was no indication or beforehand that this incident might take place. However, we had beforehand knowledge of how to recover from the issue and it was to do a reboot.

## Corrective actions

The following are the immediate corrective actions. Once we collect feedback from team members, will add more to the list.

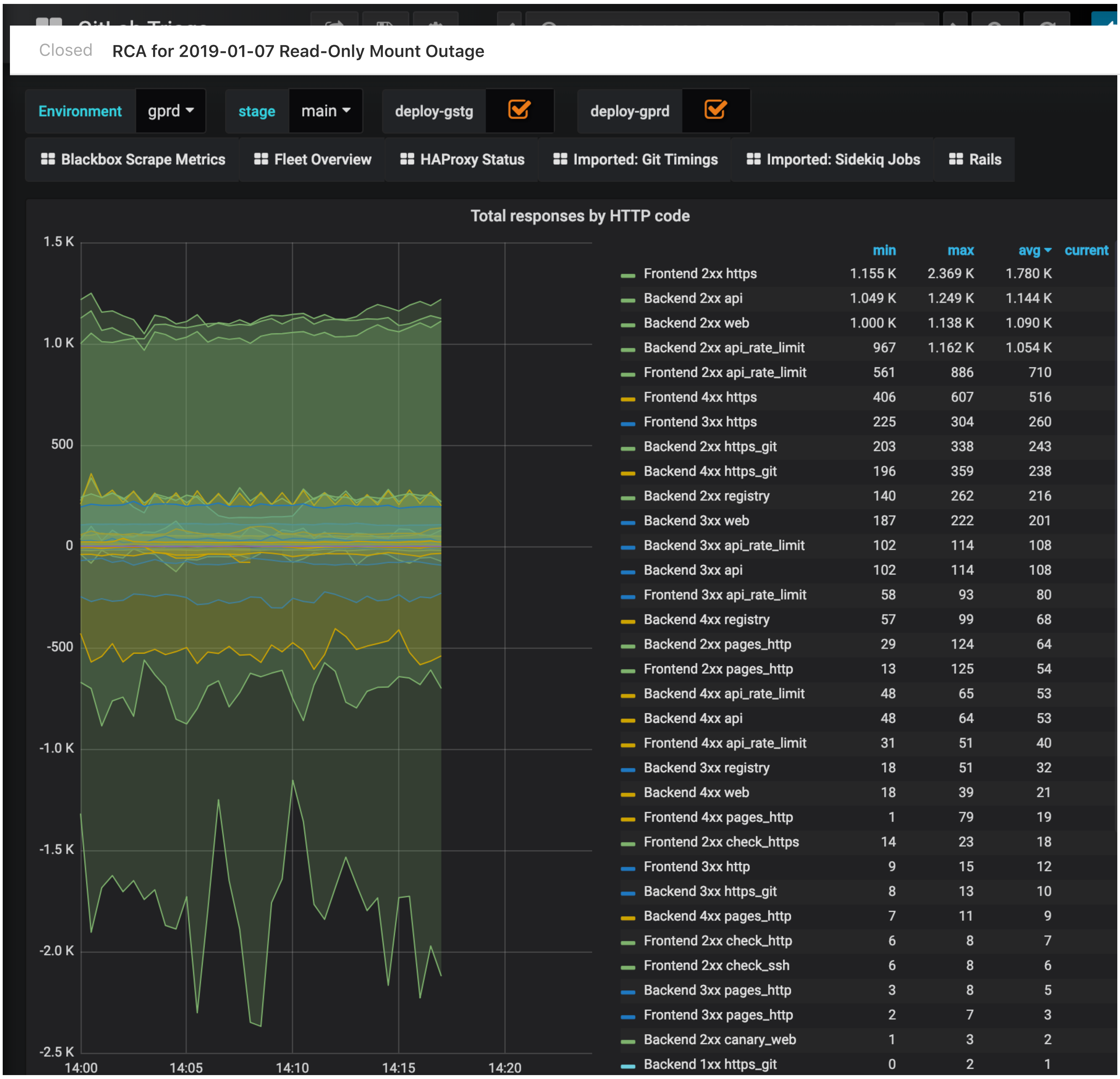
- #5881 (closed)
- #5882 (closed) (closed in favor of existing issue: #5121 (closed))
- #5883 (closed)
- #5915 (closed)
- #5886 (closed) Suggested by @Finotto to evaluate if we can have separate Chef for prod, staging and ops.
- #5887 (closed) change process change to recommend dry run for all scripts without output that can be reviewed.
- #5911 (closed)
- #5912 (closed)
- #5913 (closed)
- #5914 (closed)

## Guidelines

- Blameless RCA Guideline
- 5 whys

## References

- [1] Screenshot - Grafana 15 min snapshow of HTTP response prior to the incident

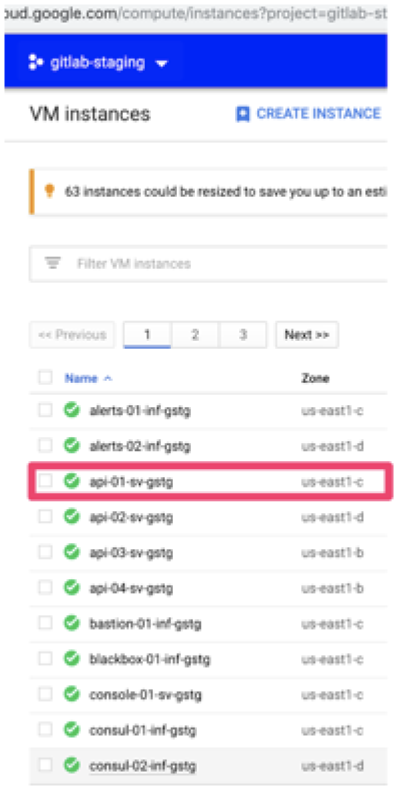


| | min | max | avg | current |
|---|---|---|---|---|
| Frontend 2xx https | 1.155 K | 2.369 K | 1.780 K | |
| Backend 2xx api | 1.049 K | 1.249 K | 1.144 K | |
| Backend 2xx web | 1.000 K | 1.138 K | 1.090 K | |
| Backend 2xx api_rate_limit | 967 | 1.162 K | 1.054 K | |
| Frontend 2xx api_rate_limit | 561 | 886 | 710 | |
| Frontend 4xx https | 406 | 607 | 516 | |
| Frontend 3xx https | 225 | 304 | 260 | |
| Backend 2xx https_git | 203 | 338 | 243 | |
| Backend 4xx https_git | 196 | 359 | 238 | |
| Backend 2xx registry | 140 | 262 | 216 | |
| Backend 3xx web | 187 | 222 | 201 | |
| Backend 3xx api_rate_limit | 102 | 114 | 108 | |
| Backend 3xx api | 102 | 114 | 108 | |
| Frontend 3xx api_rate_limit | 58 | 93 | 80 | |
| Backend 4xx registry | 57 | 99 | 68 | |
| Backend 2xx pages_http | 29 | 124 | 64 | |
| Frontend 2xx pages_http | 13 | 125 | 54 | |
| Backend 4xx api_rate_limit | 48 | 65 | 53 | |
| Backend 4xx api | 48 | 64 | 53 | |
| Frontend 4xx api_rate_limit | 31 | 51 | 40 | |
| Backend 3xx registry | 18 | 51 | 32 | |
| Backend 4xx web | 18 | 39 | 21 | |
| Frontend 4xx pages_http | 1 | 79 | 19 | |
| Frontend 2xx check_https | 14 | 23 | 18 | |
| Frontend 3xx http | 9 | 15 | 12 | |
| Backend 3xx https_git | 8 | 13 | 10 | |
| Backend 4xx pages_http | 7 | 11 | 9 | |
| Frontend 2xx check_http | 6 | 8 | 7 | |
| Frontend 2xx check_ssh | 6 | 8 | 6 | |
| Backend 3xx pages_http | 3 | 8 | 5 | |
| Frontend 3xx pages_http | 2 | 7 | 3 | |
| Backend 2xx canary_web | 1 | 3 | 2 | |
| Backend 1xx https_git | 0 | 2 | 1 | |

- [2] https://twitter.com/search?f=tweets&vertical=default&q=gitlabstatus&src=typd
- [3] https://twitter.com/search?f=tweets&vertical=news&q=gitlab&src=typd
- [4] https://gitlab.slack.com/archives/CB7P5CJS1/p1546876277019400
- [5] https://gitlab.slack.com/archives/C0259241C/p1546871812299200

## Illustration

1. In GCP console, instance_names should up like below:

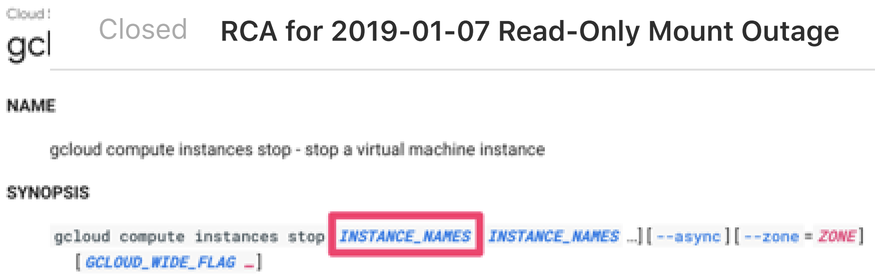


63 instances could be resized to save you up to an esti

| Name | Zone |
|---|---|
| alerts-01-inf-gstg | us-east1-c |
| alerts-02-inf-gstg | us-east1-d |
| api-01-sv-gstg | us-east1-c |
| api-02-sv-gstg | us-east1-d |
| api-03-sv-gstg | us-east1-b |
| api-04-sv-gstg | us-east1-b |
| bastion-01-inf-gstg | us-east1-c |
| blackbox-01-inf-gstg | us-east1-c |
| console-01-sv-gstg | us-east1-c |
| consul-01-inf-gstg | us-east1-c |
| consul-02-inf-gstg | us-east1-d |

2. Whether it is gcloud or an SDK (client library API), operations on an instance takes an instance_name. Example:

Closed RCA for 2019-01-07 Read-Only Mount Outage



Cloud
gcl

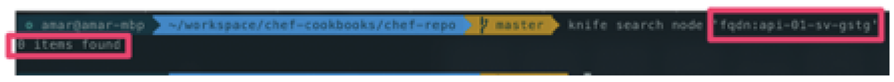
NAME

gcloud compute instances stop - stop a virtual machine instance

SYNOPSIS

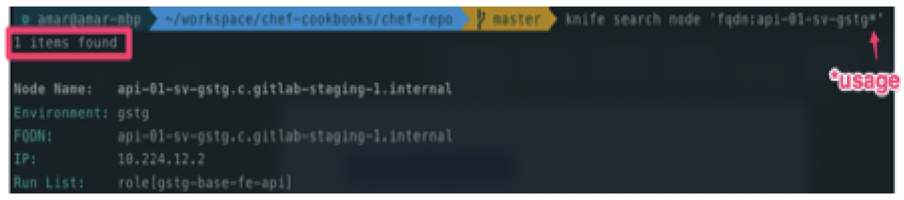
```
gcloud compute instances stop INSTANCE_NAMES INSTANCE_NAMES …][--async][--zone=ZONE]
  [GCLOUD_WIDE_FLAG …]
```

3. Once we have the instance_name in the script, looking up this instance_name in Chef didn't yield any result:



```
o amar@amar-mbp  ~/workspace/chef-cookbooks/chef-repo  master  knife search node 'fqdn:api-01-sv-gstg'
0 items found
```

4. Hence, the culprit * was used (unfortunately)



```
o amar@amar-mbp  ~/workspace/chef-cookbooks/chef-repo  master  knife search node 'fqdn:api-01-sv-gstg*'
1 items found

Node Name:   api-01-sv-gstg.c.gitlab-staging-1.internal
Environment: gstg
FQDN:        api-01-sv-gstg.c.gitlab-staging-1.internal
IP:          10.224.12.2
Run List:    role[gstg-base-fe-api]
```

Edited 1 year ago by Amarbayar Amarsanaa

Linked issues 0

Related merge requests 1

Any change request that involves running a script must have a dry-run...
gitlab-com/www-gitlab-... !18037

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5881 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5882 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5883 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa changed the description 1 year ago

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5886 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa changed the description 1 year ago

John Jarvis @jarv mentioned in issue #5887 (closed) 1 year ago

John Jarvis @jarv changed the description 1 year ago

David Smith 🌴 @dawsmith · 1 year ago Owner

Adding another question for @gitlab-com/gl-infra here. Should we invest more in ephemeral environments for testing ideas like this? In order to address #5886 (closed) - would we have been better off as a team if we had an easy way to make a test env for ideas like what we started with here?

Amarbayar Amarsanaa @aamarsanaa changed the description 1 year ago

Amarbayar Amarsanaa @aamarsanaa changed the description 1 year ago

John Skarbek @skarbek · 1 year ago Owner

I don't believe ephemeral environments would help in this particular situation. Ephemeral environments won't have the full chef stack and such for which we can test scripts against the chef server. Right now that work is fundamentally a completely different setup and configuration of what our production and staging environments look like today.

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5911 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5912 (closed) 1 year ago

Closed RCA for 2019-01-07 Read-Only Mount Outage

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5914 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa · 1 year ago Maintainer

On 1/9/19, we held a meeting and reviewed the RCA. Meeting outcome:

- We came up with 4 more corrective action items (added them to the RCA)
- Reviewed existing corrective action items and closed 2 of them since we already had an issue for one of them and the other one was concluded that the direction of the issue was not what we would want as there is a better way to approach (and an issue was created for the latter: #5913 (closed)).
- Emphasized on the need to prioritize the corrective action items (this is WIP)

Amarbayar Amarsanaa @aamarsanaa mentioned in issue #5915 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa changed the description 1 year ago

Amarbayar Amarsanaa @aamarsanaa · 1 year ago Maintainer

Proposed priority in order:

| **#5915 (closed)** | **Remove,wild-card searches in all paths** | **1** |
|---|---|---|
| #5887 (closed) | Require,that all scripts performing maintenance have a dry-run output that can be,reviewed | 2 |
| #5911 (closed) | Handbook,update - get DBRE review/approval for any change that involves DB / DB,servers | 3 |
| #5912 (closed) | Handbook,update: Get a second pair of eyes to review changes that touch redis | 3 |
| #5913 (closed) | Handbook,update: Avoid running scripts from laptop that might end up touching,production and use bastion instead | 3 |
| #5914 (closed) | Add,Ongres to PagerDuty | 3 |
| #5121 (closed) | Add,PagerDuty webhook to open a GitLab incident issue for all PagerDuty incidents | 4 |
| #5881 (closed) | Evaluate,if we can rename GCE VM node names to match Chef FQDN | 5 |
| #5883 (closed) | Broadcast,the usage of status.io to send tweets and update Status p | 6 |

Amarbayar Amarsanaa @aamarsanaa mentioned in commit gitlab-com/www-gitlab-com@90e844aa 1 year ago

Amarbayar Amarsanaa @aamarsanaa mentioned in merge request gitlab-com/www-gitlab-com!18037 (merged) 1 year ago

Devin Sylva @devin · 1 year ago Maintainer

Although Ephemeral environments currently don't use the full chef stack, I think it's still worthwhile to use them - especially for early development of changes like this.

The answer is different depending on whether we are asking "would it help now" vs. "would it help if we were were we wanted to be". I think it makes sense to move towards a place where we do things in a sane way, even if we are not yet in a place where it will catch everything.

The more we use ephemeral environments, the more we will evolve them towards what we need. This iteration is what will get us there, and without it things will always be how they are now.

David Smith 🌴 @dawsmith changed milestone to %JF team 2019 week 3 1 year ago

Amarbayar Amarsanaa @aamarsanaa · 1 year ago Maintainer

Closed RCA for 2019-01-07 Read-Only Mount Outage

Currently, 4 of 9 are closed. Of the remaining 5:

- 3 of them are handbook MRs
- 1 is PD schedule (to add Ongres)
- 1 is likely not going to be closed anytime soon as it had been an existing issue for a while.

Jose Finotto @Finotto changed milestone to %JF team 2019 week 5 1 year ago

Jose Finotto @Finotto changed milestone to %JF team 2019 week 5 1 year ago

Jose Finotto @Finotto changed milestone to %JF team 2019 week 7 1 year ago

Jose Finotto @Finotto changed milestone to %JF Team 2019 Week 9 1 year ago

Jose Finotto @Finotto changed milestone to %JF team 2019 week 7 1 year ago

Jose Finotto @Finotto closed 1 year ago

ops-gitlab-net @ops-gitlab-net mentioned in issue #6255 (closed) 1 year ago

Amarbayar Amarsanaa @aamarsanaa mentioned in issue production#640 (closed) 1 year ago

Please register or sign in to reply