Bug List: (24 of 63)

Bug 1947477 - 4.7->4.6 rollback fails due to MCO requiring new ignition spec "Failed to render configuration for pool master: parsing Ignition config failed: unknown version. Supported spec versions: 2.2, 3.0, 3.1"' (edit)

Save Changes
This is a minor update (do not send email)

Keywords: Reopened × Upgrades ×
Status: NEW (edit)
Alias: None (edit)
Product: OpenShift Container Platform
Component: Machine Config Operator
Sub Component: Machine Config Operator
Priority: urgent
Severity: urgent
Assignee: mkrejci@redhat.com
Reset Assignee to default
Pool: None
Personal Tags:
Duplicates (1): ~~1975975~~ (view as bug list)
TreeView+ depends on / blocked

Reported: 2021-04-08 14:40 UTC by **Federico Paolinelli**
Modified: 2022-11-06 04:25 UTC (History)
CC List: 20 users including you (edit)
Ignore Bug Mail: (never email me about this bug)
Fixed In Version:
PM Score: 110
Clone Of: ~~1940207~~
Last Closed: 2021-05-05 20:50:59 UTC
Flags: mkrejci: needinfo -
mkrejci: blocker -
(more flags)

Show advanced fields

Orig. Est.:	Current Est.:	Points Worked:	Points Left:	%Complete:	Gain:	Deadline:
0.0	0.0	0.0 + 0	0.0	0	0.0	

Summarize time (including time for bugs blocking this bug)

Attachments (Terms of Use)
Add an attachment (proposed patch, testcase, etc.)

Links
Add Link: System Bug ID URL Or just paste full URL here +

Groups:
Only users in at least one of the selected groups can view this bug:
Unselecting all groups makes this a more public bug.
Users in the roles selected below can always view this bug:
Reporter
CC List
The assignee , QA contact and Docs contact can always see a bug, and this section does not take effect unless the bug is restricted to at least one group.

Federico Paolinelli 2021-04-08 14:40:58 UTC Description Extra private groups Private

Collapse All Comments
Expand All Comments
Add Comment

RED HAT CONFIDENTIAL

+++ This bug was initially created as a clone of ~~Bug #1940207~~ +++

4.6->4.7->4.6 update jobs are permafailing [1]. Picking a recent job [2], the build-log presentation is the step timing out:

{"component":"entrypoint","file":"prow/entrypoint/run.go:165","func":"k8s.io/test-infra/prow/entrypoint.Options.ExecuteProcess","level":"error","msg":"Process did not finish before 3h0m0s timeout","severity":"error","time":"2021-03-17T00:37:49Z"}

ClusterVersion is stuck early [3]:

```
                    {
                        "lastTransitionTime": "2021-03-16T21:41:26Z",
                        "message": "Working towards 4.6.21: 1% complete",
                        "status": "True",
                        "type": "Progressing"
                    },
```

Checking the ClusterOperators:

```
  $ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-
test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/clusteroperators.json | jq -r '.items[] | (.status.versions[] | select(.name ==
"operator").version) + " " + .metadata.name' | sort
  ...
  4.6.21 storage
  4.7.0-0.ci-2021-03-15-164837 baremetal
  4.7.0-0.ci-2021-03-15-164837 machine-config
```

baremetal is new in 4.7, so that's why it hasn't moved. machine-config is in the process of rolling us back to 4.6, and rolling nodes gives us a new cluster-version operator pod, and we don't preserve completion percentage across CVO restarts, so that's why the CVO is only claiming 1%. Here's where the CVO is stuck:

```
  $ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-
test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/pods/openshift-cluster-version_cluster-version-operator-798b6d49d9-
rg5lp_cluster-version-operator.log | grep 'Running sync.*state\|Result of work' | tail -n5
  I0317 00:31:42.787489       1 sync_worker.go:513] Running sync
registry.build02.ci.openshift.org/ci-op-
t7wzziyv/release@sha256:6ae80e777c206b7314732aff542be105db892bf0e114a6757cb9e34662b8f891
(force=true) on generation 3 in state Updating at attempt 13
  I0317 00:31:42.922011       1 task_graph.go:555] Result of work: []
  I0317 00:37:24.701243       1 task_graph.go:555] Result of work: [Could not update
prometheusrule "openshift-cluster-version/cluster-version-operator" (9 of 619): the server is
reporting an internal error]
  I0317 00:40:18.223604       1 sync_worker.go:513] Running sync
registry.build02.ci.openshift.org/ci-op-
t7wzziyv/release@sha256:6ae80e777c206b7314732aff542be105db892bf0e114a6757cb9e34662b8f891
(force=true) on generation 3 in state Updating at attempt 14
  I0317 00:40:18.365749       1 task_graph.go:555] Result of work: []
  $ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-
test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/pods/openshift-cluster-version_cluster-version-operator-798b6d49d9-
rg5lp_cluster-version-operator.log | grep 'error running apply for prometheusrule.*cluster-
version-operator' | tail -n1
  E0317 00:43:58.615624       1 task.go:81] error running apply for prometheusrule "openshift-
cluster-version/cluster-version-operator" (9 of 619): Internal error occurred: failed calling
webhook "prometheusrules.openshift.io": Post "https://prometheus-operator.openshift-
monitoring.svc:8080/admission-prometheusrules/validate?timeout=5s": dial tcp 10.129.0.38:8080:
connect: no route to host
```

I'm not clear on the "no route to host" bit, but possibly something about the webhook was new in 4.7, and is not getting removed by the 4.6 monitoring components?

[1]: https://testgrid.k8s.io/redhat-openshift-ocp-release-4.7-informing#periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback
[2]: https://prow.ci.openshift.org/view/gcs/origin-ci-test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback/1371929264202452992#1:build-log.txt%3A168
[3]: https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-extra/artifacts/clusterversion.json

--- Additional comment from Simon Pasquier on 2021-03-18 14:22:55 UTC ---

Looking at the endpoint and pod resources, nothing seems wrong:

```
$ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-test/logs/periodic-
ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/endpoints.json |jq '.items | map(. | select(.metadata.name == "prometheus-
operator") | .subsets)'

[
  [
    {
      "addresses": [
        {
          "ip": "10.129.0.38",
          "nodeName": "ip-10-0-159-237.ec2.internal",
```

```
            "targetRef": {
              "kind": "Pod",
              "name": "prometheus-operator-5d47c59c7c-mw8cq",
              "namespace": "openshift-monitoring",
              "resourceVersion": "80128",
              "uid": "a0bf1d06-8008-4337-9201-4ed7db053432"
            }
          }
        ],
        "ports": [
          {
            "name": "web",
            "port": 8080,
            "protocol": "TCP"
          },
          {
            "name": "https",
            "port": 8443,
            "protocol": "TCP"
          }
        ]
      }
    ]
]

$ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-test/logs/periodic-
ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/pods.json | jq '.items | map(. | select(.metadata.name == "prometheus-operator-
5d47c59c7c-mw8cq") | .metadata.name + " " + .status.podIP + " " + .spec.nodeName)'
[
  "prometheus-operator-5d47c59c7c-mw8cq 10.129.0.38 ip-10-0-159-237.ec2.internal"
]
```


The prometheus-operator logs [1] show lots of connect errors with the same "no route to host" issue:


```
E0317 00:44:22.308929       1 reflector.go:178] github.com/coreos/prometheus-
operator/pkg/informers/informers.go:75: Failed to list *v1.PrometheusRule: Get
"https://172.30.0.1:443/apis/monitoring.coreos.com/v1/namespaces/openshift-cluster-samples-
operator/prometheusrules?resourceVersion=83959": dial tcp 172.30.0.1:443: connect: no route to
host
E0317 00:44:22.308944       1 reflector.go:178] github.com/coreos/prometheus-
operator/pkg/informers/informers.go:75: Failed to list *v1.ConfigMap: Get
"https://172.30.0.1:443/api/v1/namespaces/openshift-controller-manager-operator/configmaps?
labelSelector=prometheus-name&resourceVersion=84656": dial tcp 172.30.0.1:443: connect: no
route to host
E0317 00:44:22.308965       1 reflector.go:178] github.com/coreos/prometheus-
operator/pkg/informers/informers.go:75: Failed to list *v1.PrometheusRule: Get
"https://172.30.0.1:443/apis/monitoring.coreos.com/v1/namespaces/openshift-authentication-
operator/prometheusrules?resourceVersion=83959": dial tcp 172.30.0.1:443: connect: no route to
host
E0317 00:44:22.308987       1 reflector.go:178] github.com/coreos/prometheus-
operator/pkg/informers/informers.go:75: Failed to list *v1.ConfigMap: Get
"https://172.30.0.1:443/api/v1/namespaces/openshift-ingress/configmaps?
labelSelector=prometheus-name&resourceVersion=84656": dial tcp 172.30.0.1:443: connect: no
route to host
E0317 00:44:22.309002       1 reflector.go:178] github.com/coreos/prometheus-
operator/pkg/informers/informers.go:75: Failed to list *v1.ServiceMonitor: Get
"https://172.30.0.1:443/apis/monitoring.coreos.com/v1/namespaces/openshift-
multus/servicemonitors?resourceVersion=84164": dial tcp 172.30.0.1:443: connect: no route to
host
```


Same goes for cluster-monitoring-operator [2]:


```
E0317 00:43:30.787516       1 reflector.go:127] github.com/openshift/cluster-monitoring-
operator/pkg/operator/operator.go:197: Failed to watch *v1.ConfigMap: failed to list
*v1.ConfigMap: Get "https://172.30.0.1:443/api/v1/namespaces/kube-system/configmaps?
resourceVersion=84838": dial tcp 172.30.0.1:443: connect: no route to host
E0317 00:43:48.707537       1 reflector.go:127] github.com/openshift/cluster-monitoring-
operator/pkg/operator/operator.go:194: Failed to watch *v1.ConfigMap: failed to list
*v1.ConfigMap: Get "https://172.30.0.1:443/api/v1/namespaces/openshift-monitoring/configmaps?
resourceVersion=84501": dial tcp 172.30.0.1:443: connect: no route to host
E0317 00:43:51.779555       1 reflector.go:127] github.com/openshift/cluster-monitoring-
operator/pkg/operator/operator.go:197: Failed to watch *v1.ConfigMap: failed to list
*v1.ConfigMap: Get "https://172.30.0.1:443/api/v1/namespaces/openshift-config-
managed/configmaps?resourceVersion=84501": dial tcp 172.30.0.1:443: connect: no route to host
E0317 00:44:06.371587       1 reflector.go:127] github.com/openshift/cluster-monitoring-
operator/pkg/operator/operator.go:197: Failed to watch *v1.ConfigMap: failed to list
*v1.ConfigMap: Get "https://172.30.0.1:443/api/v1/namespaces/kube-system/configmaps?
resourceVersion=84838": dial tcp 172.30.0.1:443: connect: no route to host
E0317 00:44:09.444569       1 reflector.go:127] github.com/openshift/cluster-monitoring-
operator/pkg/operator/operator.go:197: Failed to watch *v1.ConfigMap: failed to list
*v1.ConfigMap: Get "https://172.30.0.1:443/api/v1/namespaces/openshift-user-workload-
monitoring/configmaps?resourceVersion=84501": dial tcp 172.30.0.1:443: connect: no route to
```

host
E0317 00:44:12.579576 1 reflector.go:127] github.com/openshift/cluster-monitoring-operator/pkg/operator/operator.go:197: Failed to watch *v1.ConfigMap: failed to list *v1.ConfigMap: Get "https://172.30.0.1:443/api/v1/namespaces/openshift-config/configmaps?resourceVersion=84501": dial tcp 172.30.0.1:443: connect: no route to host

Based on the current information, I'm reassigning to the Networking component.

[1] https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-extra/artifacts/pods/openshift-monitoring_prometheus-operator-5d47c59c7c-mw8cq_prometheus-operator.log
[2] https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback/1371929264202452992/artifacts/e2e-aws-upgrade-rollback/gather-extra/artifacts/pods/openshift-monitoring_cluster-monitoring-operator-655bc474db-97dgn_cluster-monitoring-operator.log

--- Additional comment from Eric Paris on 2021-03-23 16:00:39 UTC ---

This bug has set a target release without specifying a severity. As part of triage when determining the importance of bugs a severity should be specified. Since these bugs have not been properly triaged we are removing the target release. Teams will need to add a severity before setting the target release again.

--- Additional comment from Federico Paolinelli on 2021-03-31 16:26:34 UTC ---

I think I found the reason.

The 4.6 ovs pod relies [1] on a file dropped by MCO [2] to understand if ovs is running on the host or not.
That file is not created anymore in 4.7.

When we rollback, CNO is rolled back before MCO, so it starts with the 4.7 version of that systemd unit that does not
create the sentinel file anymore [3].
The result is two instances of ovs running on the node at the same time, which are likely to cause the errors I am seeing in the ovs logs. I am testing the rollback manually right now.

[1] https://github.com/openshift/cluster-network-operator/blob/bb19869f526665792d4e42effee98afc4688e766/bindata/network/openshift-sdn/sdn-ovs.yaml#L55
[2] https://github.com/openshift/machine-config-operator/blob/0d140929e3758f5bac3e50c561b467fada11a1ed/templates/common/_base/files/configure-ovs-network.yaml#L17
[3] https://github.com/openshift/machine-config-operator/blob/6c42eaa4d333d2c575540eec7dc866e7cce527d7/templates/common/_base/files/configure-ovs-network.yaml#L7

--- Additional comment from Federico Paolinelli on 2021-03-31 16:27:26 UTC ---

And here the list of operators (from the last failed job):

omg get clusteroperators
NAME VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication 4.6.23 True False True 45m
baremetal 4.7.0-0.ci-2021-03-30-013032 True False False 2h40m
cloud-credential 4.6.23 True False False 1h43m
cluster-autoscaler 4.6.23 True False False 1h36m
config-operator 4.6.23 True False False 3h38m
console 4.6.23 True False False 1h24m
csi-snapshot-controller 4.6.23 True False False 1h43m
dns 4.7.0-0.ci-2021-03-30-013032 True False True 1h34m
etcd 4.6.23 True False False 2h4m
image-registry 4.6.23 True True False 1h23m
ingress 4.6.23 True False False 1h39m
insights 4.6.23 True False False 3h32m
kube-apiserver 4.6.23 True False False 1h50m

```
kube-controller-manager                    4.6.23                         True       False
False     1h49m
kube-scheduler                             4.6.23                         True       False
False     1h48m
kube-storage-version-migrator              4.6.23                         True       False
False     1h43m
machine-api                                4.6.23                         True       False
True      1h18m
machine-approver                           4.6.23                         True       False
False     3h37m
machine-config                             4.7.0-0.ci-2021-03-30-013032   True       False
False     2h9m
marketplace                                4.6.23                         True       False
False     1h42m
monitoring                                 4.6.23                         True       False
False     1h37m
network                                    4.6.23                         True       False
False     1h24m
node-tuning                                4.6.23                         True       False
False     1h42m
openshift-apiserver                        4.6.23                         True       False
False     1h44m
openshift-controller-manager               4.6.23                         True       False
False     1h42m
openshift-samples                          4.6.23                         True       False
False     1h39m
operator-lifecycle-manager                 4.6.23                         True       False
False     1h34m
operator-lifecycle-manager-catalog         4.6.23                         True       True
False     1h39m
operator-lifecycle-manager-packageserver   4.6.23                         False      True
False     10m
service-ca                                 4.6.23                         True       False
False     1h38m
storage                                    4.6.23                         True       False
False     1h23m
```

--- Additional comment from Federico Paolinelli on 2021-04-06 14:37:04 UTC ---

I tested the fix, sdn works with that but the rollback is stopped by MCO with:

```
    lastSyncError: 'pool master has not progressed to latest configuration: controller version
mismatch for rendered-master-cb2db7df54e993c796b76a2242b3e08a expected
d5dc2b519aed5b3ed6a6ab9e7f70f33740f9f8af has b5723620cfe40e2e4e8cbdcb105d6ae534be1753: pool is
degraded because rendering fails with "": "Failed to render configuration for pool master:
parsing Ignition config failed: unknown version. Supported spec versions: 2.2, 3.0, 3.1",
retrying'
    master: 'pool is degraded because rendering fails with "": "Failed to render configuration
for pool master: parsing Ignition config failed: unknown version. Supported spec versions: 2.2,
3.0, 3.1"'
    worker: 'pool is degraded because rendering fails with "": "Failed to render configuration
for pool worker: parsing Ignition config failed: unknown version. Supported spec versions: 2.2,
3.0, 3.1"'
```

I think that's another bug on MCO that will block rollbacks.

Not sure how to handle that from a bug tracking perspective, it won't show up until the network error won't be fixed.

--- Additional comment from W. Trevor King on 2021-04-06 18:12:51 UTC ---

> Not sure how to handle that from a bug tracking perspective, it won't show up until the network error won't be fixed.

Separate bug filed after this one gets to MODIFIED or later makes sense to me.

--- Additional comment from Federico Paolinelli on 2021-04-07 07:34:32 UTC ---

(In reply to W. Trevor King from comment #6)
> > Not sure how to handle that from a bug tracking perspective, it won't show up until the network error won't be fixed.
>
> Separate bug filed after this one gets to MODIFIED or later makes sense to
> me.

Sound good to me

--- Additional comment from Eric Paris on 2021-04-07 10:16:01 UTC ---

This bug has set a target release without specifying a severity. As part of triage when determining the importance of bugs a severity should be specified. Since these bugs have not been properly triaged we are removing the target release. Teams will need to add a severity before setting the target release again.

--- Additional comment from Yu Qi Zhang on 2021-04-07 15:56:51 UTC ---

MCO today does not support downgrades so unfortunately the 4.7->4.6 downgrade would not succeed either way. A previous bug similar to this: ~~https://bugzilla.redhat.com/show_bug.cgi?id=1911841~~

If we wish to support downgrades fully that would require MCO to become backwards compatible

--- Additional comment from Clayton Coleman on 2021-04-08 13:09:59 UTC ---

All components in the product need to be forward and backward compatible. Exceptions are only granted in specific circumstances. While we do not tell customers to rollback, we specifically support it for ourselves because it is a key point.

So MCO not being able to rollback from 4.7 is indeed a serious bug, and it needs a justification and explanation (the impact both to normal upgrades, to hung upgrades, and to emergency rollback).

All components are required to talk to their N-1 versions correctly, so if you're telling me during an upgrade a paused 4.6 pool would fail to rollout a new machine (because MCO can't speak to that difference) that is a product release blocking bug.

Federico Paolinelli 2021-04-08 14:41:53 UTC Comment 1 Private

Cloned this to track the MCO side described here ~~https://bugzilla.redhat.com/show_bug.cgi?id=1940207#c5~~ and to leave 1940207 on the network bug

Federico Paolinelli 2021-04-08 14:43:19 UTC

Summary: 4.7->4.6 rollbacks stuck on prometheusrules admission webhook "no route to host" → 4.7->4.6 rollbacks stuck on master: 'pool is degraded because rendering fails with "": "Failed to re...

Federico Paolinelli 2021-04-08 14:46:06 UTC Comment 2 Private

Adding more context: the original bz was related to a downgrade issue on CNO. While fixing it, I saw the rollback did not finish due to:

```
    lastSyncError: 'pool master has not progressed to latest configuration: controller version
mismatch for rendered-master-cb2db7df54e993c796b76a2242b3e08a expected
d5dc2b519aed5b3ed6a6ab9e7f70f33740f9f8af has b5723620cfe40e2e4e8cbdcb105d6ae534be1753: pool is
degraded because rendering fails with "": "Failed to render configuration for pool master:
parsing Ignition config failed: unknown version. Supported spec versions: 2.2, 3.0, 3.1",
retrying'
    master: 'pool is degraded because rendering fails with "": "Failed to render configuration
for pool master: parsing Ignition config failed: unknown version. Supported spec versions: 2.2,
3.0, 3.1"'
    worker: 'pool is degraded because rendering fails with "": "Failed to render configuration
for pool worker: parsing Ignition config failed: unknown version. Supported spec versions: 2.2,
3.0, 3.1"'
```

There is a discussion on the original bz whether MCO should support or not rollbacks, but I'd like to keep the MCO and CNO issues separated so we don't loose track.

Eric Paris 2021-04-08 15:00:15 UTC

Flags: blocker?

Yang Yang 2021-04-09 01:50:33 UTC

CC: yanyang@redhat.com

~~CEE Openshift PM Score Bot~~ 2021-04-09 07:01:54 UTC

PM Score: 0 → 110

Michelle Krejci 2021-04-13 22:16:23 UTC Comment 3 Private

The MCO team, in discussion with @mrussell@ceon.com and @acrawfor@redhat.com, does not regard rollbacks as a bug. It may be a feature that we want to support. Given the level of discussion on the original bug, we can discuss further at the Platform and Lifecycle Architecture meeting. Pending the outcome of that meeting, I will close this bug.

CC: mkrejci@redhat.com, mrussell@ceon.com
Flags: needinfo?(mrussell@ceon.com)

OpenShift BugZilla Robot 2021-04-16 08:41:36 UTC

Depends On: ~~1950261~~

Michelle Krejci 2021-04-19 17:06:56 UTC

Flags: needinfo?(mrussell@ceon.com) blocker? → needinfo- blocker-

Michelle Krejci 2021-04-26 18:28:04 UTC Comment 4 Private

Currently, we state (to customers) that we don't support rollbacks: "Reverting your cluster to a previous version, or a rollback, is not supported. Only upgrading to a newer version is supported." https://docs.openshift.com/container-platform/4.7/updating/understanding-the-update-service.html#update-service-overview_understanding-the-update-service

This was also discussed in greater detail at the Lifecycle and Pillar meeting April 22, 2021 https://docs.google.com/document/d/1QrvzW0QEftUMjNBCSkzG1m4yO9tNnTn3SHuoU5hiUMw/edit#heading=h.qcyk8b39hg2h. Closing for now since this is not currently supported.

Status: NEW → CLOSED
Resolution: --- → NOTABUG
Last Closed: 2021-04-26 18:28:04

Ben Parees 2021-04-29 20:05:22 UTC Comment 5 Extra private groups Private

RED HAT CONFIDENTIAL

Based on Clayton's latest feedback on the aos-devel thread "What's the deal with rollbacks and downgrades? Do we support them?", i am reopening this bug.

We either need to:

1) fix whatever is breaking the job
2) define how a customer does rollback an upgrade(and when it is allowed vs when they are past the point of no return and we must progress them forward) and develop a job that tests that procedure instead, if the existing job is not testing a "valid/possible" rollback scenario.

Status: CLOSED → NEW
CC: bparees@redhat.com
Resolution: NOTABUG → ---
Keywords: Reopened

Ben Parees 2021-04-29 20:06:39 UTC

Environment: job=periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback=all

~~Clayton Coleman~~ 2021-04-30 02:40:28 UTC Comment 6 Private

Can I get more human readable description of what the actual bug in MCO is:

We upgrade to 99 in CVO to 4.7 from 4.6 (so new MCO, new nodes). We then start a rollback, which means we go from beginning of payload to end on 4.6. So we should be applying apiserver, network, etc then mco. MCO is deployed to 4.6 (nodes are still 4.7).

At this point, MCO/MCS is serving some older spec version???

^ fill in the explanation here in something a non-MCO-guru can understand (i.e. component X at 4.6 can't work with component Y at 4.7, etc).

W. Trevor King 2021-04-30 05:23:10 UTC Comment 7 Private

Machine-config server should only come in for new nodes, which isn't the issue here. I dunno if anyone actually linked to gathered assets from the run that spawned this bug [2] (maybe wasn't even in the CI account). But Jerry points out [3]:

> The error is expected. The MCO doesn't support downgrades, so the 4.6 MCO doesn't understand how to parse ignition 3.2 configs (4.7). This in turn means unfortunately all downgrades from 4.7->4.6 will fail

So if I'm understanding, the fix would be "teach the 4.6 MCO to understand 3.2 Ignition configs and down-convert when it sees them, mumble mumble anything which cannot be down-converted". However, looking at [4,5]:

```
  $ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-
test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1387886270079832064/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/clusterversion.json | jq -r '.items[].status.history[] | .startedTime + " " +
(.completionTime // "-") + " " + .state + " " + .version'
  2021-04-29T23:34:37Z - Partial 4.6.27
  2021-04-29T22:32:41Z 2021-04-29T23:34:37Z Partial 4.7.0-0.ci-2021-04-29-142719
  2021-04-29T22:01:14Z 2021-04-29T22:29:21Z Completed 4.6.27
  $ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-
test/logs/periodic-ci-openshift-release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-
rollback/1387886270079832064/artifacts/e2e-aws-upgrade-rollback/gather-
extra/artifacts/clusterversion.json | jq -r '.items[].status.conditions[] | .lastTransitionTime
+ " " + .type + "=" + .status + " " + .reason + ": " + .message'
  2021-04-29T22:29:21Z Available=True : Done applying 4.6.27
  2021-04-30T01:12:02Z Failing=True ClusterOperatorDegraded: Cluster operator ingress is
reporting a failure: Some ingresscontrollers are degraded: ingresscontroller "default" is
degraded: DegradedConditions: One or more other status conditions indicate a degraded state:
DeploymentReplicasAllAvailable=False (DeploymentReplicasNotAvailable: 1/2 of replicas are
available)
  2021-04-29T22:32:41Z Progressing=True ClusterOperatorDegraded: Unable to apply 4.6.27: the
cluster operator ingress is degraded
  2021-04-29T22:01:14Z RetrievedUpdates=False NoChannel: The update channel has not been
configured.

So probably not worth sinking time into the MCO vs. Ignition spec issues until that earlier
hang-up gets sorted.

[1]: https://prow.ci.openshift.org/view/gs/origin-ci-test/logs/periodic-ci-openshift-release-
master-ci-4.8-upgrade-from-stable-4.7-e2e-aws-upgrade-rollback/1387886269920448512
[2]: https://github.com/openshift/machine-config-operator/pull/2506#issuecomment-814168869
[3]: https://github.com/openshift/machine-config-operator/pull/2506#issuecomment-815024614
[4]: https://testgrid.k8s.io/redhat-openshift-ocp-release-4.7-informing#periodic-ci-openshift-
release-master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback
[5]: https://prow.ci.openshift.org/view/gs/origin-ci-test/logs/periodic-ci-openshift-release-
master-ci-4.7-upgrade-from-stable-4.6-e2e-aws-upgrade-rollback/1387886270079832064
```

Yu Qi Zhang 2021-05-03 23:49:13 UTC Comment 8 Private

```
Like Trevor says, the fundamental issue is basically that different versions of MCO support
different versions of ignition spec'ed configs. A breakdown

4.5: Ignition spec v2.2, 3.0 (sort of)
4.6: Ignition spec v2.2, 3.0, 3.1
4.7: Ignition spec v2.2, 3.0, 3.1, 3.2
4.8: same as 4.7

Correspondingly the version of MCO that gets deployed generates a rendered config in the newest
version, so

4.5: rendered config on 2.2
4.6: rendered config on 3.1
4.7: rendered config on 3.2
4.8: same as 4.7

So this means a rendered config created by the 4.7 MCO cannot be read by the 4.6 MCO, because
newer ignition versions can contain spec fields that the older one does not have. This is
especially the case for the 4.5->4.6 update where we went from spec 2 to spec 3, and that is
basically a one-way trip, since the major version bump is a complete overhaul.

This is fundamentally where the error is coming from: when rolling back the 4.6 MCO rolls out,
sees that the nodes have updated to some "3.2" config, tries to understand it, and cannot. So
it cannot validate if the nodes are in a good state or not to perform the downgrade, and thus
it fails.

So then, the request here is to: in any future version of Openshift where we update the
ignition spec version, we must also backport it as a feature to the previous y release. In this
case, that would mean we need to backport 3.2 support (which was added in 4.7) to 4.6.

I personally feel that this is a feature request for 4.8+ (framing: MCO backports ignition spec
bumps to 4.y-1 for 2-way compatibility always). I think that in itself should be 99.9% good for
downgrades. For example, based on the above, 4.8 today should not fail the rollback to 4.7. As
for 4.7->4.6 and 4.6->4.5, I am leaning towards marking those as not bugs. Feel free to tell me
if the above assessment is wrong in any way
```

W. Trevor King 2021-05-03 23:57:06 UTC Comment 9 Private

```
> So then, the request here is to: in any future version of Openshift where we update the
ignition spec version, we must also backport it as a feature to the previous y release.
```

Alternative, although forward-looking-only, approach would be to teach the MCO to write a version that the previous minor understands. If 4.7 had written 3.1, 4.7->4.6 rollbacks would have been fine. And if 4.6 wrote 2.2, 4.6->4.5 rollbacks would have been fine. I dunno how much effort it is to fix existing releases via backports, but if, say, 4.9 learns how to read 3.3 (or whatever) but keeps the default at 3.2 for a minor, we'll avoid getting into this situation with 4.9 -> 4.8 rollbacks.

Yu Qi Zhang 2021-05-04 00:28:32 UTC Comment 10 Private

> Alternative, although forward-looking-only, approach would be to teach the MCO to write a version that the previous minor understands.

That does get into another issue. For example, the 4.7 spec 3.2 bump was to support LUKS encrypted storage in 4.7, which as I understand was not in 3.1, so I don't think that's feasible

W. Trevor King 2021-05-04 00:57:13 UTC Comment 11 Private

Looks like 3.2 was stabilized 2020-10-13 [1]. 4.6.0 was 2020-10-21 [2]. The bump to 3.2 landed in the MCO on 2020-12-04 [3]. I agree that if we need a feature that's so young, backporting an ability to read the new spec version to the older minor's MCO is our only option. But hopefully Ignition will get out in front of us a bit further, and we can auto bump an understanding of the new spec versions before we need to consume the features they add. Any ideas when Ignition 3.3 is due to be stabilized?

[1]: https://github.com/coreos/ignition/pull/1103#event-3874509356
[2]: https://amd64.ocp.releases.ci.openshift.org/releasestream/4-stable/release/4.6.0
[3]: https://github.com/openshift/machine-config-operator/pull/2248#event-4072173071

Benjamin Gilbert 2021-05-04 02:36:05 UTC Comment 12 Private

Ignition spec 3.3 is expected for OCP 4.9, and we usually stabilize the spec fairly late in the OCP development cycle in case last-minute issues pop up.

So far, spec stabilizations have been driven by OCP needs, and it wouldn't have been practical to stabilize a spec one OCP release before it was needed. Also, we prefer to develop all the cooperating pieces of code together (Ignition, RHCOS support glue, Butane transpilation, client code), since that helps flush out any design flaws before we commit to a stable config spec with particular semantics. As a result, for Ignition to lead OCP by one release, we'd essentially need to develop an entire feature set, land it, and then tell users not to use it for one cycle.

As an alternative, ign-converter <https://github.com/coreos/ign-converter/> may be able to help here. It already has the ability to downconvert an Ignition config to an earlier spec version, or fail if the input config can't be expressed in that spec. If we backported a newer converter to the previous MCO release we should be able to get the correct semantics: automatically downgrade a config if the cluster hasn't started using newer features yet, and fail with a clear explanation if it has.

CC: bgilbert@redhat.com

W. Trevor King 2021-05-04 03:52:52 UTC Comment 13 Private

> If we backported a newer converter to the previous MCO release we should be able to get the correct semantics: automatically downgrade a config if the cluster hasn't started using newer features yet, and fail with a clear explanation if it has.

But if you defer the bump until you need the new feature, you'll break rollbacks once the new MCO comes around and starts upconverting Ignition config spec versions, right? Unless the downconverter can say "I can't express $NEW_STUFF in $OLD_SPEC_VERSION, but I can certainly express which files will need to be ripped up off the disk to unroll this MachineConfig". Because all we need is enough information for the incoming, older-version MCO to be able to roll back to an older-version MachineConfig.

> Also, we prefer to develop all the cooperating pieces of code together (Ignition, RHCOS support glue, Butane transpilation, client code), since that helps flush out any design flaws before we commit to a stable config spec with particular semantics.

I didn't notice any 3.2-alpha PRs landing in the MCO repo, but I could certainly have missed

them. Or does the combined development mostly happen in an in-flight MCO PR? I went back through [1], and I didn't notice any references off to parallel Ignition PRs, RHCOS merge requests, etc. I understand that you want to know that a plan will work before you commit to it in a branch that will eventually end up in production, but it's not clear to me yet why Ignition config spec bumps have to be forward incompatible.

[1]: https://github.com/openshift/machine-config-operator/pull/2248

Benjamin Gilbert 2021-05-04 04:41:54 UTC Comment 14 Private

>> If we backported a newer converter to the previous MCO release we should be able to get the correct semantics: automatically downgrade a config if the cluster hasn't started using newer features yet, and fail with a clear explanation if it has.
> But if you defer the bump until you need the new feature, you'll break rollbacks once the new MCO comes around and starts upconverting Ignition config spec versions, right?

Not generally. New Ignition features, so far, have been conditional to particular use cases. For example, if you don't need partition resizing, LUKS, gs:// URLs, or boot disk RAID, it's perfectly fine to continue using 3.1.0 configs.

If we don't want to deal with downconverting configs in the previous release (and thus backporting code), we may be able to be more selective about using newer config versions in the current release. I believe the MCO currently always uses the current stable spec, since that lets it avoid introspecting the features needed by a particular config. But it could probably just e.g. generate 3.2.0 configs, try downconverting them to 3.1.0, and if that succeeds, serve them as 3.1.0 instead. I may be missing something though.

> Unless the downconverter can say "I can't express $NEW_STUFF in $OLD_SPEC_VERSION, but I can certainly express which files will need to be ripped up off the disk to unroll this MachineConfig".

The functionality of the files stage (files, systemd units, passwd) has been fairly stable, actually. The new features have been more on the disk partitioning/formatting side, which the MCO doesn't reconcile at runtime anyway. Also, upgrades/downgrades don't currently matter for day-1-only functionality, since upgraded clusters stay with their original bootimage and thus their original Ignition version. Net result, "which files will need to be ripped up" isn't really a thing.

> I didn't notice any 3.2-alpha PRs landing in the MCO repo, but I could certainly have missed them. Or does the combined development mostly happen in an in-flight MCO PR?

The MCO changes land pretty late, currently. That part isn't great and we'd like to improve it. We do a better job at the OS integration level.

Every Ignition release includes a WIP copy of the next stable spec, with no stability guarantees. The recent 2.10.1 release includes probably more than half of the new functionality that will end up stabilizing as spec 3.3.0. It can be invoked by writing a config with version "3.3.0-experimental". Crucially, 3.3.0-experimental configs will no longer be accepted by newer Ignition releases after 3.3.0 is stabilized.

Butane uses a similar versioning trick, and the Dracut glue in fedora-coreos-config and openshift/os generally keeps pace with the latest Ignition release. Those are the packages where co-development is most visibly helping us right now. It's actually been pretty common to have to rethink Ignition functionality as we get experience with how it integrates into the OS.

W. Trevor King 2021-05-04 18:52:41 UTC Comment 15 Private

> Not generally. New Ignition features, so far, have been conditional to particular use cases... The new features have been more on the disk partitioning/formatting side, which the MCO doesn't reconcile at runtime anyway.

Oh, nice :). Seems like either downconverting or introspecting features and picking the minimal required spec version would work for most rollbacks, and I have no opinions on which of those the MCO folks feel would be less work. But can we pick one, or some other alternative, and start doing that the next time we bump the Ignition spec? I'll leave it to other folks to decide if it's worth it to try and green up 4.7 -> 4.6, but I'd like 4.9 -> 4.8, etc., to not get stuck on something similar.

Yu Qi Zhang 2021-05-05 00:31:10 UTC Comment 16 Private

I think we can forward look to enable 4.9->4.8. One possibility to prevent the need of backporting in the future, is if the MCD sees a 3.x+1 config of its current support (e.g. 3.3 in 4.9, but it allows a 3.4 config), it just interprets it as a 3.3 config.

This should be ok since any new functionality added to 3.4 shouldn't be used during upgrades anyways, and new installs shouldn't be able to go to a previous version.

Yu Qi Zhang 2021-05-05 20:50:59 UTC Comment 17 Private

After some further considerations I am going to close this as NOTABUG again, and revert my position on the MCO support of rollbacks on y-stream downgrades (which is what this bug is). There is no point in doing so unless we have underlying RHCOS and RHEL support, and e.g. if RHCOS or RHEL went from RHEL 8.3->8.4 during a y-stream, there is no support as far as I'm aware for downgrading to 8.3 again.

Thus once the nodes have begun the upgrade process, the MCO cannot confidently say you will be able to rollback. Thus there is no reason for us to consider this support. We can reconsider this once the higher level discussions for RHCOS happens.

Status: NEW → CLOSED
Resolution: --- → NOTABUG
Last Closed: 2021-04-26 18:28:04 → 2021-05-05 20:50:59

~~Clayton Coleman~~ 2021-05-06 14:29:41 UTC Comment 18 Private

I have reopened this, please do not close again without my approval.

> 4.5: Ignition spec v2.2, 3.0 (sort of)
4.6: Ignition spec v2.2, 3.0, 3.1
4.7: Ignition spec v2.2, 3.0, 3.1, 3.2
4.8: same as 4.7

> Correspondingly the version of MCO that gets deployed generates a rendered config in the newest version, so

4.5: rendered config on 2.2
4.6: rendered config on 3.1
4.7: rendered config on 3.2
4.8: same as 4.7

> Alternative, although forward-looking-only, approach would be to teach the MCO to write a version that the previous minor understands. If 4.7 had written 3.1, 4.7->4.6 rollbacks would have been fine. And if 4.6 wrote 2.2, 4.6->4.5 rollbacks would have been fine. I dunno how much effort it is to fix existing releases via backports, but if, say, 4.9 learns how to read 3.3 (or whatever) but keeps the default at 3.2 for a minor, we'll avoid getting into this situation with 4.9 -> 4.8 rollbacks.

This is not correct. We should NEVER require the use of a new API version until ALL supported components have been upgraded to a new version. Effectively what MCO *MUST* do (this is not an option) is use a rendered config the previous Y version of openshift understands until the *next* Y. We do this for kube and internal etcd, and we should be doing it for ignition.

To describe what SHOULD have been done in rendering config:

4.5: rendered config on 2.2
4.6: rendered config on 2.2 (because 4.5 didn't support 3.0 really)
4.7: rendered config on 3.1 (because 4.6 didn't support 3.2)
4.8: rendered config on 3.2 (because 4.7 supports 3.2)

When doing N-1 compatibility you need to remain fully compatible throughout the whole upgrade cycle by upgrading control plane first (this has been how kube and OCP have worked).

So the bug here is that (i think?) MCO is too aggressively switching to rendered config, and instead has to do those transitions ONLY on minors when new versions support everything. It's still fine during CI and everything to test the newer verisons (probably as an optional job), but it is NOT safe to start requiring a new config version until it has been live for at least one Y release.

Status: CLOSED → POST
Resolution: NOTABUG → ---

~~Clayton Coleman~~ 2021-05-06 14:32:30 UTC Comment 19 Private

Backporting the required ignition changes is probably not a good plan since that introduces a lot of risk in the previous z (is an API change?) and the safest path is for MCO to follow the same versioning and API rules that control plane, kubelet, etc all do (we do the same thing with CNI/CRI/CSI). I know we have a better doc upstream than

https://kubernetes.io/docs/setup/release/version-skew-policy/ that describes WHY we have this policy, but we have considered this policy authoritative for OCP since 3.6 or so and it applies to all components and all APIs that must support the openshift version skew rules.

Steve Milner 2021-05-06 21:19:45 UTC Comment 20 Private

There are a few things here in this BZ I'd like to unpack:

1. Rollbacks
As noted in previous comments this isn't something that is currently supported, though we do have a test for it. Adding support for rollbacks is a fair enhancement request. If we want to support this direction let's get an RFC and/or enhancement. Adding support for this will probably involve multiple teams and require design.

2. Config update control
Fair point. As Benjamin noted above, and Clayton did later, new features that Ignition adds in a spec could wait a release so we could ensure N-1. While most of the spec bumps have been additive for new features it may make sense for us to slow down and batch a bit more when it comes to Ignition/MachineConfig in releases.

CC: smilner@redhat.com

Aravindh Puthiyaparambil 2021-05-07 14:27:27 UTC
CC: aravindh@redhat.com

Aravindh Puthiyaparambil 2021-05-07 16:04:25 UTC Comment 21 Private

@ccoleman@redhat.com is there a reason for the bug to be in POST? POST implies a fix is in the works which is not the case. I am moving it to NEW to reflect the accurate state.

Status: POST → NEW
Flags: needinfo?(ccoleman@redhat.com)

Aravindh Puthiyaparambil 2021-05-07 17:27:22 UTC Comment 22 Extra private groups Private

RED HAT CONFIDENTIAL

Linking to http://mailman-int.corp.redhat.com/archives/aos-devel/2021-May/msg00044.html which has pertinent information for this bug.

W. Trevor King 2021-05-07 21:33:29 UTC Comment 23 Private

I'm pretty sure POST was supposed to be NEW in comment 18.

Flags: ~~needinfo?(ccoleman@redhat.com)~~

~~Clayton Coleman~~ 2021-05-10 15:28:30 UTC Comment 24 Private

Attempting to talk through the various issues here, I want to start at the key point that sometimes is confused with rollbacks (we have not asked teams to support rollbacks, but we DO require the following):

"Does MCO, during a Y-upgrade, ever put itself in a spot where it cannot function properly?"

It is required (and would be a release blocking bug) that all components of OpenShift remain functional during the upgrade process (y or z). That means to a user there is either no disruption (a goal for our API endpoints) or at most a few seconds of disruption (for controller driven processes that are not directly hit) from a user's perspective. To achieve that, we set up certain patterns and models (what I referred to from kube above, but not well documented in general):

1. We run our API services HA, and make sure that API changes are always forwards compatible during ANY upgrade (so an old client, talking to the new version behind a load balancer, sees

no behavior change if they are suddenly connected back to the old version)
2. We generally do not deprecate or remove APIs, and when we do we gate upgrades that might block them on ensuring no old client can still be talking to the version about to be removed
3. Controllers could be connected to an arbitrary API server during upgrade (old or new) so they must be written and tested as if they could be regressed during an upgrade (upgrades are not one way)
4. If we have to make breaking format changes, we do that *after* the upgrade via a separate process that is initiated by a user (for instance, our migration from etcd2 -> 3 involved a breaking change and was required to be done after upgrading to 3.6 but before upgrading to 3.7, so 3.7 simply required that the format change was already in place)
5. We make minimal / no API / incompatible changes during z streams in order to simplify reasoning about the safety of z streams.

We test these in a variety of ways (and are trying to always improve them), and one of those tests is the rollback test because done properly (for the kube control plane at least) any implications of 1-5 are automatically hit by rollback tests in a way that causes detectable failures. That assumption *does not* always hold for other components, which is why I was focused on how your use of ignition (a client facing API provided by your component) behaves during upgrades and whether it indicates we might not satisfy the "keeps working" invariant. So that's the first thing to check from this bug. For example, only new machines are ignited then your spec bump is ok - but if they aren't you would be "functionally unavailable" and thus this would be an urgent bug.

Secondly, Ben's request here was at my request because the failure of MCO to rollback blocks other components ability (the control plane) to test these critical assumptions as we add z streams to both releases. If you have broken 4.7 to 4.6 rollback tests, we cannot keep verifying that our EUS *forward* upgrade is still safe, which is of massive importance for EUS stream.

Next, the other constraint for upgrades is:

"Is MCO sufficiently tested so that the MCO team is confident that if the upgrade process is stopped or disrupted AT ANY POINT that the MCO remains available and functional for users"

The other goal of rollback tests (as described above) is that they simulate one class of problems that can occur due to stopped upgrades. The CVO could fail or deadlock at any point, or during a deployment rollout the new pod could be killed at any point and the old pod resurrected. A team that isn't aggressively testing for that themselves is implicitly leveraging the rollback tests and our high level detection to find those issues. Since having each team have the expertise to debug these sorts of problems does not necessarily scale, rollback and other tests are our vehicle for simulating problems - when we hit the issues I'm describing in a general way we can identify the assumptions / bugs that led to that and guide teams to fix without each team having to build their own tests (or lots of their own tests). If the MCO team believes that they are 100% available during their entire upgrade process even if ANY part of the upgrade process hangs arbitarily long (i.e. your new CRD is rolled out but not your operator, your operator gets disrupted so the old code starts running AFTER the new code is rolled out) then this is less of an issue for them.

An example about this causing you to fail with respect to ignition spec is: "if you require new ignition for booting machines, and that assumes that you're using new RHCOS, are you aware that the osimageurl is updated AFTER your operator rolls out, so if CVO hangs before it updates osimageurl you can't boot new machines?" That would be another reason this would be an urgent bug, which rollback is intended to help verify (although it's not optimally efficient)

However, like the previous test, the failure of 4.6 to 4.7 rollback now leaves the entire platform (other componetns) unable to exercise those simulations, which means that MCO has regressed our ability to catch issues of this sort in 4.8.

So, the three asks are:

1. confirm that you are functionally available during upgrade
2. confirm that you remaining functionally available if any part of the process fails
3. help us fix 4.7 to 4.6 rollback with a workaround for MCO because otherwise we have regressed ALL test coverage of upgrade safety on our first EUS step which is critical to our ability to deliver EUS.

~~Clayton Coleman~~ 2021-05-10 15:29:04 UTC

Summary: 4.7->4.6 rollbacks stuck on master: 'pool is degraded because rendering fails with "": "Failed to re... → 4.7->4.6 rollback fails due to MCO requiring new ignition spec "Failed to render configuration for p...

Colin Walters 2021-05-10 18:42:41 UTC Comment 25 Private

This is a semi-aside but: We had a chat around this and I think on the OS side what we need to support ideally is "Keep running with latest OS but rollback to specific N-1 kernel". Because 95% of the issues on an upgrade that *could* be fixed by rollback are going to be the kernel. That could be some sort of MCO feature, even something as streamlined as:

```
operatingSystemHotfix:
  kernel: 4.6
```

```

To implement that the MCO would walk the CVO upgrade history, find the last release image that matched 4.6, pull its machine-os-content, pull the kernel out of that and apply it to the nodes.

---

CC: walters@redhat.com

**Yu Qi Zhang** 2021-05-11 03:50:35 UTC Comment 26 Private

A few things to note:

> 2. We generally do not deprecate or remove APIs, and when we do we gate upgrades that might block them on ensuring no old client can still be talking to the version about to be removed

The MCO never fully deprecated any API in the ignition spec. The newest MCO has support all the way back to 4.1 generated configs.

> 3. Controllers could be connected to an arbitrary API server during upgrade (old or new) so they must be written and tested as if they could be regressed during an upgrade (upgrades are not one way)

The "upgrades are not one way" is what concerns me, since I was under the impression that the whole point of upgrades and graphs is that they are "one way" today. The new MCO controller, again, understands old configs always

> 4. If we have to make breaking format changes, we do that *after* the upgrade via a separate process that is initiated by a user

Using this as an easier to highlight example, this wasn't really done in the MCO up to now (and correspondingly RHCOS) for at least the major ignition spec bump we performed during the 4.5->4.6 timeframe. The ignition spec bump from 2.x to 3.x (which is much more complex than the 3.1->3.2 in this bug) was done automatically. Meaning that if you had somehow made older definitions we did not support, the upgrade would break (but only for rare scenarios that I don't think ever manifested, which is good).

This is partially why I wanted to frame this as a forward looking feature: to design upgrades in the MCO such that we are able to perform format breaking changes with guarantees.

> For example, only new machines are ignited then your spec bump is ok - but if they aren't you would be "functionally unavailable" and thus this would be an urgent bug.

and

> "if you require new ignition for booting machines, and that assumes that you're using new RHCOS, are you aware that the osimageurl is updated AFTER your operator rolls out, so if CVO hangs before it updates osimageurl you can't boot new machines?"

Fortunately newly ignited machines do not see this issue. The machine-config-server always serves spec versions based on the incoming ignition binary request version, so the MCS is able to on the fly serve all supported ignition versions up to the newest one in that MCO version. This is required since we do not bump bootimages by default today, so a 4.7 cluster may still be using 4.4 bootimages, thus the compatibility will be there for the foreseeable future.

> "Is MCO sufficiently tested so that the MCO team is confident that if the upgrade process is stopped or disrupted AT ANY POINT that the MCO remains available and functional for users"

Just my personal perspective: the MCO itself is relatively resilient in the upgrade process, in the sense that the new/old MCO components are able to intercommunicate provided that they are in the one-way upgrade path. The issue (e.g. this bug) only occurs if the rollout order is not observed, e.g. an old daemonset attempting to manage an updated node (daemonset should roll out before nodes are updated).

> 1. confirm that you are functionally available during upgrade

This should not have changed in the general sense for the MCO (barring some bugs in the new version perhaps)

> 2. confirm that you remaining functionally available if any part of the process fails

We should return the error as noted, but otherwise the MCO is able to run still during failure scenarios presented.

> 3. help us fix 4.7 to 4.6 rollback with a workaround for MCO because otherwise we have regressed ALL test coverage of upgrade safety on our first EUS step which is critical to our ability to deliver EUS.

This really is the crux of the discussion. As I see it one option moving forward with the MCO (e.g. spec 3.3 in 4.9) would be the MCO (or ignition converter), when generating the rendered config (which is what's causing the issue in this bug), chain parsed it such that it renders it
```

in the oldest supported version of the spec.

This means that, for example, all specs would render to 3.1 unless it requires a config snippet only supported in 3.2, etc., such that new installs can still take advantage of the feature, but upgrades don't immediately jump to the new version to facilitate a rollback test like this.

Now for this bug specifically, this probably would not be recommended since 4.7 has been released, so we'd have upgraded customers from 3.1 to 3.2 back to 3.1. So our best alternative right now is either to
1. backport full 3.2 support to 4.6
2. make an exception of dummy support of 3.2 in 4.6 via parser hacks

I think generally speaking we'd still like to position ourselves to not backport features. The MCO has not operated under the assumptions above (perhaps mistakenly) up until now, so for the MCO, it would be a feature since we've never designed it that way, although it may be considered more of a bug from the general platform perspective.

Sorry for the text dump, let me know if that doesn't make sense, and thanks for the overall context.

W. Trevor King 2021-05-11 04:11:53 UTC Comment 27 Private

> The "upgrades are not one way" is what concerns me, since I was under the impression that the whole point of upgrades and graphs is that they are "one way" today. The new MCO controller, again, understands old configs always

Here, the incompat is between the outgoing MachineConfig controller and the incoming MachineConfig controller, right? Checking 4.8.0-rc.3 update CI:

```
  $ curl -s https://gcsweb-ci.apps.ci.l2s4.p1.openshiftapps.com/gcs/origin-ci-test/logs/release-openshift-origin-installer-e2e-aws-upgrade/1390739944372178944/artifacts/e2e-aws-upgrade/deployments.json | jq -r '.items[] | select(.metadata.name == "machine-config-controller").spec | {replicas, strategy}'
  {
    "replicas": 1,
    "strategy": {
      "rollingUpdate": {
        "maxSurge": "25%",
        "maxUnavailable": "25%"
      },
      "type": "RollingUpdate"
    }
  }
```

So the update risk would be something like:

1. You surge in a new controller pod.
2. Outgoing pod is terminated and release its leader lease.
3. Incoming pod acquires the leader lease and updates to the new Ignition spec.
4. Incoming pod has some kind of disaster and dies.
5. Outgoing pod re-acquires the leader lease, sees the new, unrecognized Ignition specs, and sticks.
6. Another disaster keeps the Deployment controller from scheduling a replacement controller pod with the new code.

That's two disasters, and (6) in particular seems pretty low risk. If replicas was larger than 1, I'd be more concerned.

> So our best alternative right now is...

Third option would be to ensure the rollback for 4.6->4.7->4.6 tests always kicks in before the CVO asks the MCO to update. That conveniently preserves the rollback test for other components, because the bulk of the content that happens after the MCO is PrometheusRule and similar stuff that shouldn't be all that exposed to rollback issues. That approach wouldn't work for non-MCO components, because if we reversed course before updating them, we'd leave MCO-rollback uncovered.

[1]: https://prow.ci.openshift.org/view/gs/origin-ci-test/logs/release-openshift-origin-installer-e2e-aws-upgrade/1390739944372178944

Colin Walters 2021-05-13 16:32:11 UTC Comment 28 Private

I think the simple way to say this is that the rendered config object in the cluster is entirely contained inside the MCO. The controller renders it and each pod in the daemonset reads it. So this fix:

> As I see it one option moving forward with the MCO (e.g. spec 3.3 in 4.9) would be the MCO (or ignition converter), when generating the rendered config (which is what's causing the issue

in this bug), chain parsed it such that it renders it in the oldest supported version of the spec.

seems to me by far the simplest and most reliable.

The node (RHCOS) version is mostly irrelevant here because as Jerry noted, the MCS already translates when serving to the node.

Benjamin Gilbert 2021-05-29 00:07:19 UTC Comment 29 Private

Upstream ign-converter RFE for an function to downconvert a config as far as possible: https://github.com/coreos/ign-converter/issues/22

Yu Qi Zhang 2021-06-24 21:35:06 UTC
Duplicate of this bug: ~~1975975~~
DPCR Bugzilla Bot 2021-06-24 21:35:31 UTC
CC: sippy@dptools.openshift.org

Red Hat Bugzilla 2021-08-31 22:34:00 UTC Comment 31 Extra private groups Private

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

CC: ~~pkrupa@redhat.com~~

Red Hat Bugzilla 2021-08-31 22:34:44 UTC Comment 32 Extra private groups Private

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

Red Hat Bugzilla 2021-09-15 05:48:10 UTC Comment 33 Extra private groups Private

RED HAT CONFIDENTIAL

Terminated for repeated audit failures

CC: ~~kakkoyun@redhat.com~~

Vikas Laad 2021-11-09 17:08:13 UTC
Sub Component: Machine Config Operator
CC: aos-bugs@redhat.com, vlaad@redhat.com
Assignee: jerzhang@redhat.com → team-mco@redhat.com
QA Contact: mnguyen@redhat.com → rioliu@redhat.com

Red Hat Bugzilla 2021-11-10 23:42:49 UTC Comment 34 Extra private groups Private

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

CC: ~~lcosic@redhat.com~~

Red Hat Bugzilla 2022-01-28 23:23:05 Comment 35 Extra private groups Private

UTC

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

CC: ~~aconstan@redhat.com~~

Apoorva Jagtap 2022-03-18 08:31:05 UTC
CC: apjagtap@redhat.com

Red Hat Bugzilla 2022-05-09 08:29:24 Comment 36 Extra private groups Private

UTC

RED HAT CONFIDENTIAL

Account disabled by LDAP Audit for extended failure

Assignee: team-mco@redhat.com → jerzhang@redhat.com

Sinny Kumari 2022-05-11 13:31:58 UTC
CC: skumari@redhat.com
Assignee: jerzhang@redhat.com → mco-triage@bot.bugzilla.redhat.com

Red Hat Bugzilla 2022-05-21 04:06:13 Comment 37 Extra private groups Private

UTC

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

CC: ~~ecoleman@redhat.com~~

Red Hat Bugzilla 2022-05-21 04:06:15 Comment 38 Extra private groups Private

UTC

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

Red Hat Bugzilla 2022-06-30 23:03:21 Comment 39 Extra private groups Private

UTC

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

CC: ~~erooth@redhat.com~~

Red Hat Bugzilla 2022-06-30 23:03:53 Comment 40 Extra private groups Private

UTC

RED HAT CONFIDENTIAL

remove performed by PnT Account Manager <pnt-expunge@redhat.com>

Colin Walters 2022-10-13 16:56:53 UTC Comment 41 Private

This came up in a chat, I have two points:

- This bug came up in the context of the Ignition spec 2 -> 3 transition, but at this point all supported clusters have made that transition
- I made a comment earlier around kernel/operating-system level rollbacks https://bugzilla.redhat.com/show_bug.cgi?id=1947477#c25 and we now actually have that implemented as part of https://github.com/openshift/enhancements/blob/master/enhancements/ocp-coreos-layering/ocp-coreos-layering.md

Benjamin Gilbert 2022-11-06 04:25:02 UTC Comment 42 Private

The 2 -> 3 spec transition isn't relevant here. The issue AIUI is that rendered configs are currently always rendered to the latest supported Ignition spec version, even if that's not necessary for encoding their contents. If that spec isn't supported by the next older MCO version, downgrade tests will fail.

Additional Comments: Make comment private (visible only to members of the **redhat** group)

Comment | Preview

Type '@' and the first 3 letters of a users login to select a user to needinfo

Need additional information from other

Needinfo users from roles:

Needinfo other users:

Status: NEW

Mark as Duplicate

Save Changes

This is a minor update (do not send email)

Bug List: (24 of 63)