Red Hat

Sherine Khoury <skhoury@redhat.com>

OLM needs for external-dns-operator GA in 4.11

11 messages

Sherine Khoury <skhoury@redhat.com> Mon, Jan 24, 2022 at 4:56 PM
To: Kevin Rizza <krizza@redhat.com>
Cc: Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>

Hi Kevin,
and thanks again for giving us some of your time today.

As discussed, the external-dns-operator is going from tech preview to GA in 4.11.
It has an architecture separating operator and operand namespaces, as a best practice to limit risks on customer workloads that depend on the operated services.
At least 2 other operators, following the same design, are on our 2022 backlog.

ExternalDNS operator therefore needs to :

- create another (operand) namespace, at bundle install
- create rbac resources (role, rolebinding) in the operand namespace
- create clusterrolebinding that would later bind the operand serviceaccount to a clusterrole to allow the operand to watch some resources

You mentioned a possibility to put static resources within the bundle and have OLM create it at installation:

- Is there a way we could add namespace to the list of resources covered?
- Can someone from your team provide us some support to get this working for the rbac resources? A debugging session or something?
  Our previous tentatives to add this failed at bundle validation level

Thanks again for your help
Best regards
Sherine

Kevin Rizza <krizza@redhat.com> Mon, Jan 24, 2022 at 4:57 PM
To: Sherine Khoury <skhoury@redhat.com>, Joe Lanford <jlanford@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>
Cc: Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>

cc @Joe Lanford @Per Goncalves da Silva
[Quoted text hidden]

Joe Lanford <jlanford@redhat.com> Mon, Jan 24, 2022 at 11:34 PM
To: Kevin Rizza <krizza@redhat.com>
Cc: Sherine Khoury <skhoury@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>

Hi Sherine,

The OLM team has been discussing what it would take to add namespaces to the list of supported resources that could be included in a bundle. There was pretty much a team-wide consensus that that seemed like a reasonable thing to do, and it progresses us towards a related goal we have to support arbitrary kinds (see

https://issues.redhat.com/browse/OLM-1781).

However, namespaces in particular are not a trivial addition to the list. The team raised questions like "what should OLM do if the namespace already exists?", "should the namespace be deleted (along with everything inside it) if the CSV is deleted?", and "if the operator supports singlenamespace install mode and there are multiple installations of that operator in a cluster, both of those installations will include a manifest for the same namespace in their bundle, causing a potential conflict; how would OLM handle that?"

Ultimately, I think we should put this particular feature request on the OLM backlog, get it prioritized, write an EP, and get an implementation done, but I'm not sure you could count on that for a 4.11 GA. It's certainly not something OLM would make any strong commitments on for 4.11.

In the meantime, I think your best option is to:
1. Give your main operator permission to get/create namespaces with a specific resourceName (i.e. the namespace you know you need to create).
2. Give your main operator the necessary RBAC `escalate` and `bind` permissions to allow it to create RBAC for the operand. This allows your operator to create/update/bind RBAC without your operator serviceaccount's RBAC being required to actually itself have the permissions it is trying to manage. Obviously, this means the main serviceaccount could escalate its own permissions, but with this mechanism, there's an extra auditable step. You can read more about `escalate` and `bind` here if you aren't familiar: https://kubernetes.io/docs/reference/access-authn-authz/rbac/#privilege-escalation-prevention-and-bootstrapping
3. Rather than dropping manifests into the bundle, ensure the namespace and RBAC exist via code during operator startup.

Thanks!
Joe

[Quoted text hidden]

Sherine Khoury <skhoury@redhat.com> Tue, Jan 25, 2022 at 3:19 PM
To: Joe Lanford <jlanford@redhat.com>
Cc: Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>, Dave Baker <dbaker@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

Hi Joe,

Thanks for your quick reply, it helps with the planning that is ongoing for us right now.

We had discussed using escalate and bind as a solution once already, but we didn't dig into it much because we found that certain issues were already open on it, for example https://github.com/operator-framework/operator-lifecycle-manager/issues/2018.
I'm going to retry though, so reopening an MR and I'll tag whomever is interested so that we can interact on this more easily.

I just added Dave Baker to the thread too, in order to give us again his opinion about using the escalate + bind from the auditing perspective. If I remember correctly, it was a little bit better than giving cluster wide permissions, but not sure how more secure, as the operator could escalate its permissions... This is especially true in our context, with the permissions we would be escalating to... (creates and deletes on secrets, namespaces for ex)

Since yesterday, Andrey also tested pushing manifests for the operand namespace into the bundle.
In order to do that, we had to bypass the bundle validation, which refuses resources having another namespace than the operator's.
At bundle install, the namespace is completely ignored, and the resources are created in the operator namespace instead.
Is that why you propose that we ensure operand namespace and rbac through code?

Thanks for your help
Best regards
Sherine

[Quoted text hidden]

Dave Baker <dbaker@redhat.com> Tue, Jan 25, 2022 at 3:43 PM
To: Sherine Khoury <skhoury@redhat.com>
Cc: Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

In my opinion, escalate and bind should only be a short term, emergency measure to get through something like validation. Even with the auditable steps it's not (imho) viable for long term use and is definitely not something we would wish for anyone else out in the community to see us doing and consider it fair game for their own use.

I maintain the belief that there's something askew in the fundamental design of the system that would permit the total avoidance of our difficulties. The dual namespace in particular is a key candidate to try to eliminate since it's this that generates the most noise.

Without looking at those fundamental design decision we are left chasing the symptoms of the problem, and not the problem itself.

Dave

[Quoted text hidden]
--

Dave Baker | Red Hat Product Security | irc: dbaker | TZ:UTC | dbaker@redhat.com

---

Sherine Khoury <skhoury@redhat.com> Tue, Jan 25, 2022 at 4:08 PM
To: Dave Baker <dbaker@redhat.com>
Cc: Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

Hi Dave,

To clarify the reasons why the design of this operator (like a lot of other core, especially Network operators) has this separation, here is a link that was passed on to me:
https://mailman-int.corp.redhat.com/archives/aos-devel/2019-January/msg00088.html

We also had discussed merging everything into a single namespace, and during one of the brainstorming sessions we did, David Eads also was totally against a change of the design. The same reasons were mentioned: avoiding problems at cleanup, especially when finalizers are involved.

Maybe there's a way to avoid these as well that other teams are using and that we simply don't know about... I don't know.
For a more detailed view or the design that is used for external dns operator, you can see https://docs.google.com/document/d/1PmOfU61_xvY6xREDYycUNPmrkq6SzM7pxbH7sBySsJI/edit?usp=sharing

Thanks
Sherine
[Quoted text hidden]

Dave Baker <dbaker@redhat.com> Tue, Jan 25, 2022 at 6:16 PM
To: Sherine Khoury <skhoury@redhat.com>
Cc: Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Andrey Lebedev <alebedev@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

I hear what you're saying, but my opinion still stands. We are dealing with the symptoms, not the root cause.

Half of the problems of cleanup/finalizers came from the usage of validating admission webhooks. If that webhook is present solely for the purpose of validating CRD contents (which is what was mentioned at one point, briefly) then it should be questioned as to whether it serves any real purpose being there, and whether removing it would simplify sufficiently to allow a single namespace (CRD validation then being done either only via the CRD schema, or before use rather than on admission).

The other candidate solution from the discussions at the end of last year was to avoid the multi-namespace and consider an operator that installed multiple times, once for each instance of what is now the CRD. The multi-namespace for that resolves itself.

I believe the operand-namespace / operator-namespace distinction comes from copying a design idea from core components that are not beholden to the limitations of OLM. So, not that it's a bad design, but it's a bad reuse of a good design.

If you're asking for my opinion, it still stands -- granting external-dns-operator these cluster-wide permissions is something that should be avoided at all costs. If we need a short- or middle-term workaround prior to finding the time needed to avoid them, then we have some wiggle room to be more or less creative with how we can limit its impact; or make it more auditable; or whatever. As a long term solution it simply doesn't hold water. A component of this nature should be granted only those permissions most closely matching the ones it needs; not such permissions as to enable it to compromise the entire cluster.

On Tue, Jan 25, 2022 at 3:08 PM Sherine Khoury <skhoury@redhat.com> wrote:

> Hi Dave,
>
> To clarify the reasons why the design of this operator (like a lot of other core, especially Network operators) has this separation, here is a link that was passed on to me:
> https://mailman-int.corp.redhat.com/archives/aos-devel/2019-January/msg00088.html

For emphasis, from re-reading this mail thread I see it is SPECIFICALLY referring to core components. Not non-core (installed via OLM) components.

> We also had discussed merging everything into a single namespace, and during one of the brainstorming sessions we did, David Eads also was totally against a change of the design. The same reasons were mentioned: avoiding problems at cleanup, especially when finalizers are involved.

As noted above; part of that rationale was because you had validating admission webhooks. It could come up for debate as to whether these are valuable. (I forget what the other concern was, but recall there were two in all).

[Quoted text hidden]

[Quoted text hidden]

Andrey Lebedev <alebedev@redhat.com> Tue, Jan 25, 2022 at 9:32 PM
To: Dave Baker <dbaker@redhat.com>
Cc: Sherine Khoury <skhoury@redhat.com>, Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

Let me disagree on the fact that the root cause is the operator-namespace/operand-namespace design combined with the usage of the validating webhook.

1. I find the concept of the separation of the workload from the workload manager using namespaces quite natural. For any type of the operator: OLM, core, homemade. It separates the concerns, makes the operator lifecycle detached from the operands.
2. The boundaries between what's called a core and not core operator is quite abstract. ExternalDNS Operator, even with the absolute minimal set of permissions is a highly privileged entity. It manages the DNS records of the underlying

cloud provider's DNS. It can change the DNS zone so that the whole OpenShift cluster disappears or worse, it can redirect all incoming traffic to a third party cluster making it act like the right one. By the way, there were plans to use ExternalDNS Operator to create the initial (cluster) DNS records, what's currently done by a core operator (Cluster Ingress Operator). This would make an OLM operator quite close to the core and this tendency will continue as more and more operators go the OLM way.
3. Usage of the validation webhook is the only way to forbid the creation/update of a resource whose fields (some or all) cannot be validated by OpenAPI schema. Also, it too adheres to the separation of concerns principle putting the validation and operational logic in different places. Alternatives cannot prevent the creation/update (can only signal misconfiguration) and mix the reconciliation and the validation logic.

In my opinion the root cause is the limited flexibility of OLM. I don't see anything unnatural in the need to create a namespace of my choice with some local roles/rolebindings in it at the installation time. Any utility dealing with manifests can do this: kubectl/kustomize, Helm. And we have seen that this need has been acknowledged by OLM too: OLM-1781.

Regards,
Andrey
[Quoted text hidden]

Dave Baker <dbaker@redhat.com> Thu, Jan 27, 2022 at 9:43 AM
To: Andrey Lebedev <alebedev@redhat.com>
Cc: Sherine Khoury <skhoury@redhat.com>, Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Daniel Messer <dmesser@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

tldr:
* split namespace design = good
* split namespace design *in conjunction with OLM assigning permissions* = bad
* if you need to move forward with escalate + bind, then consider it to be a security exception, for which an end-date for the workaround must be established.

On Tue, Jan 25, 2022 at 8:34 PM Andrey Lebedev <alebedev@redhat.com> wrote:

> Let me disagree on the fact that the root cause is the operator-namespace/operand-namespace design combined with the usage of the validating webhook.

This isn't what I said.

The problem is the operator-namespace/operand-namespace combined with the requirement on OLM.

We have an "unmoveable object meets unstoppable object" scenario here. The dual namespaces is a fine design idea for core components, but not suited for OLM deployments.

I'll restate my fundamental argument:
- A design, that is perfect in every way EXCEPT THAT it does not work with the tools required is .... {bad, inappropriate, unsuited to this situation}

The dual namespaces, in conjunction with OLM, is ~~a bad design~~ design we should not use.

> 1. I find the concept of the separation of the workload from the workload manager using namespaces quite natural. For any type of the operator: OLM, core, homemade. It separates the concerns, makes the operator lifecycle detached from the operands.

I strongly disagree. Separation of workload from workload manager is, in isolation, a great design motif. However, for

an OLM operator it is simply not viable in it's current implementation as OLM currently lacks the ability to directly provision the permissions needed for this cross namespace functionality.

> ExternalDNS Operator, even with the absolute minimal set of permissions is a highly privileged entity. It manages the DNS records of the underlying cloud provider's DNS. It can change the DNS zone so that the whole OpenShift cluster disappears or worse, it can redirect all incoming traffic to a third party cluster making it act like the right one. By the way, there were plans to use ExternalDNS Operator to create the initial (cluster) DNS records, what's currently done by a core operator (Cluster Ingress Operator). This would make an OLM operator quite close to the core and this tendency will continue as more and more operators go the OLM way.

This is a red herring. This is one of many possible use-cases but is not the REQUIRED use-case for the operator. If it was, we would have a strong argument to have it installed by the system, negating our OLM permissions problems, and ending the discussion.

Any one with direct access to DNS is able to perform an availability attack against the resources they point to. There are many mitigations (mostly in the realm of MITM prevention) to help manage this -- certificate checking and the like. The ability for any adversary to tamper with DNS, and the ability for any adversary to tamper with internal cluster objects are quite distinct and separate. Mitigation for DNS tampering (e.g. MITM prevention) is, in short, easier than the other.

> 3. Usage of the validation webhook is the only way to forbid the creation/update of a resource whose fields (some or all) cannot be validated by OpenAPI schema. Also, it too adheres to the separation of concerns principle putting the validation and operational logic in different places. Alternatives cannot prevent the creation/update (can only signal misconfiguration) and mix the reconciliation and the validation logic.
>
> **In my opinion the root cause is the limited flexibility of OLM.** I don't see anything unnatural in the need to create a namespace of my choice with some local roles/rolebindings in it at the installation time. Any utility dealing with manifests can do this: kubectl/kustomize, Helm. And we have seen that this need has been acknowledged by OLM too: OLM-1781.

This is my point exactly (bolded). Perhaps the refinement of my statement from "design = bad" to "design + OLM = bad" helps us re-align on this point. Yes, this is what we are dealing with.

So, with that said .... what type of solution are we looking for? Short- or middle- term, or long- term?

As I see it, if we have the expectation that OLM will EVER support this type of permission (in the meeting we had at the end of last year, I was led to believe that this may simply never happen), then I think it's reasonable to plan for that, establish a date estimation, and find the "least bad" way of limping along with extraneous permissions until this happens and they can be removed.

If we have the expectation that OLM won't support this type of permission, then we're driving down a metaphorical dead end. We need to make a U-turn, and sooner is probably better.

For a short- to middle- term:
* Before granting escalate + bind, I propose that more time is spent examining alternatives.
* OLM supports operator groups, operator dependencies and the like. Two OLM operators - one for each namespace - might be a viable workaround. They're marked as deprecated in the OLM docs (again, we looked at this last year, and unfortunately I don't have the URL to hand) but for this short/middle term solution it might be preferable. (the problem, in more detail, is creation of cross-namespace permissions in OLM - two olm operators, each in charge of it's own namespace, can very probably do this without as much difficulty).
* OLM supports single namespaces for each operator. So, for this use-case, an operator that installs multiple times for each instance of the operand, with each one living entirely within the one namespace.

I don't know if there are other problems down the line with these alternative approaches but I think they warrant investigation.

If you are to move forward with escalate + bind, this should be considered a security exception and have an end-date assigned. If an end-date can't be determined (e.g. there is no ETA for OLM supporting the additional functionality needed), then I return to my opinion that we need to make that U-turn.

The final thing I want to emphasize is this -- these are my opinions. If you end up moving forward with escalate + bind today, we won't block this, but if you're seeking my/our/ProdSec's blessing that it's appropriate or secure then you won't receive that either. If other factors (business needs, priorities, time availability, etc) mean we make a quick fix and move on, then file the security exception (again: end date!) and proceed with what we hope to be the least bad of the available options.

I'm always happy to hop on a video call to talk about this more.

Dave

[Quoted text hidden]

Daniel Messer <dmesser@redhat.com> Thu, Jan 27, 2022 at 10:59 AM
To: Dave Baker <dbaker@redhat.com>
Cc: Andrey Lebedev <alebedev@redhat.com>, Sherine Khoury <skhoury@redhat.com>, Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

OLM is planning to support more complex deployments by virtue of allowing authors of OLM packages to ship arbitrary Kube manifests. This is part of https://issues.redhat.com/browse/OCPPLAN-7737. This would allow you to create additional namespaces and lay down additional objects on the cluster as part of operator installation and updates.

I may have missed this in this thread, but it seems like the use case for this here is somewhere in between shipping two controllers in one package and separate them out into two different namespaces and shipping one controller but two namespaces, one of them as sort of a preparation for a later operand deployment. It seems that this operand is actually also a controller. If that's the case I think it might be worth exploring shipping two packages for now, if you can't / don't want to rely on RBAC escalation.

/D

[Quoted text hidden]

--

Daniel Messer
Product Manager Operator Framework & Quay
Red Hat OpenShift

Sherine Khoury <skhoury@redhat.com> Thu, Jan 27, 2022 at 6:10 PM
To: Daniel Messer <dmesser@redhat.com>, Miciah Masters <mmasters@redhat.com>
Cc: Dave Baker <dbaker@redhat.com>, Andrey Lebedev <alebedev@redhat.com>, Joe Lanford <jlanford@redhat.com>, Kevin Rizza <krizza@redhat.com>, Per Goncalves da Silva <pegoncal@redhat.com>, Luigi Mario Zuccarelli <luzuccar@redhat.com>, Miciah Masters <miciah.masters@redhat.com>

Hi everyone,
First thanks so much for all your valuable inputs.
To sum up a bit our 4.11 options and answer the various questions:

- @Daniel Messer yes and no: the operand is not a controller in the way that it can be deployed by OLM on its own: it doesn't handle any custom resource. it's just a namespace, with a serviceaccount, a role, rolebinding, a clusterrole + clusterrolebinding.
  When the user installs the operator, no operand controller is up... it's only when a first CR is created that a controller deployment in operand namespace will start monitoring services with a certain annotation, and will apply a DNS record for them
  - @Joe Lanford @Daniel Messer @Kevin Rizza Could OLM deploy an empty shell bundle? Do you think this would work?
  - @Miciah Masters What do you think about the dependencies between the 2 operators? update lifecycles? any danger removing the operand operator on customer workloads?
- We mentioned OperatorGroups as a possible alternative, but @Kevin Rizza discouraged us from using it as it's going to be deprecated
  - We would have only 1 bundle to install, but this doesn't solve the need to create the operand namespace manually and an operatorGroup resource as well
  - @Joe Lanford @Daniel Messer @Kevin Rizza ,
    - Do you think this is a viable short/mid-term solution while we wait for OCPPLAN-7737
    - Is there any way we could create the operand namespace? I'm guessing not, because of what @Joe Lanford explained on OLM-1781
    - Does OperatorGroup combined with installing 2 bundles (assuming solution from previous point valid) enhance the solution in any way
- I'm not sure about the 1 externalDNS = 1 operator. One of the problems we saw was with the watching of the custom resources, or with the validation webhooks
- @Dave Baker in the worst case scenario(none of the above works), would it be better to have a security exception with escalate bind? or with the old RBAC (explicit RBAC)?
- @Miciah Masters Should we consider removing the webhooks and going single namespace nonetheless? Temporarily at least?

Thanks everyone for your valuable help
Regards
Sherine

[Quoted text hidden]