Combining WasmCloud with Kubernetes

Kubernetes is the hottest project and infrastructure platform in the cloud world, and it's almost safe to say that *moving to the cloud* = *using Kubernetes*.

Currently **WasmCloud** has a completely Kubernetes-agnostic model, and while this is a lighter approach, it may be more limited in terms of spreading it and adopting it on the ground. I think it would be easier to use and land on the ground if it was better integrated with Kubernetes!

The problem with Kubernetes, and one that the **WasmCloud** team should have recognized early on, is that it's getting bloated. However, when combining WasmCloud & Kubernetes, we can try to use only some of the features and components of Kubernetes to ensure that using WasmCloud maintains the simplicity of the resources and architecture, but also takes into account the experience of using it in line with Kubernetes and reduces the user's cost of learning and using it.
Of course, **WasmCloud** can also be well integrated into the existing huge Kubernetes.

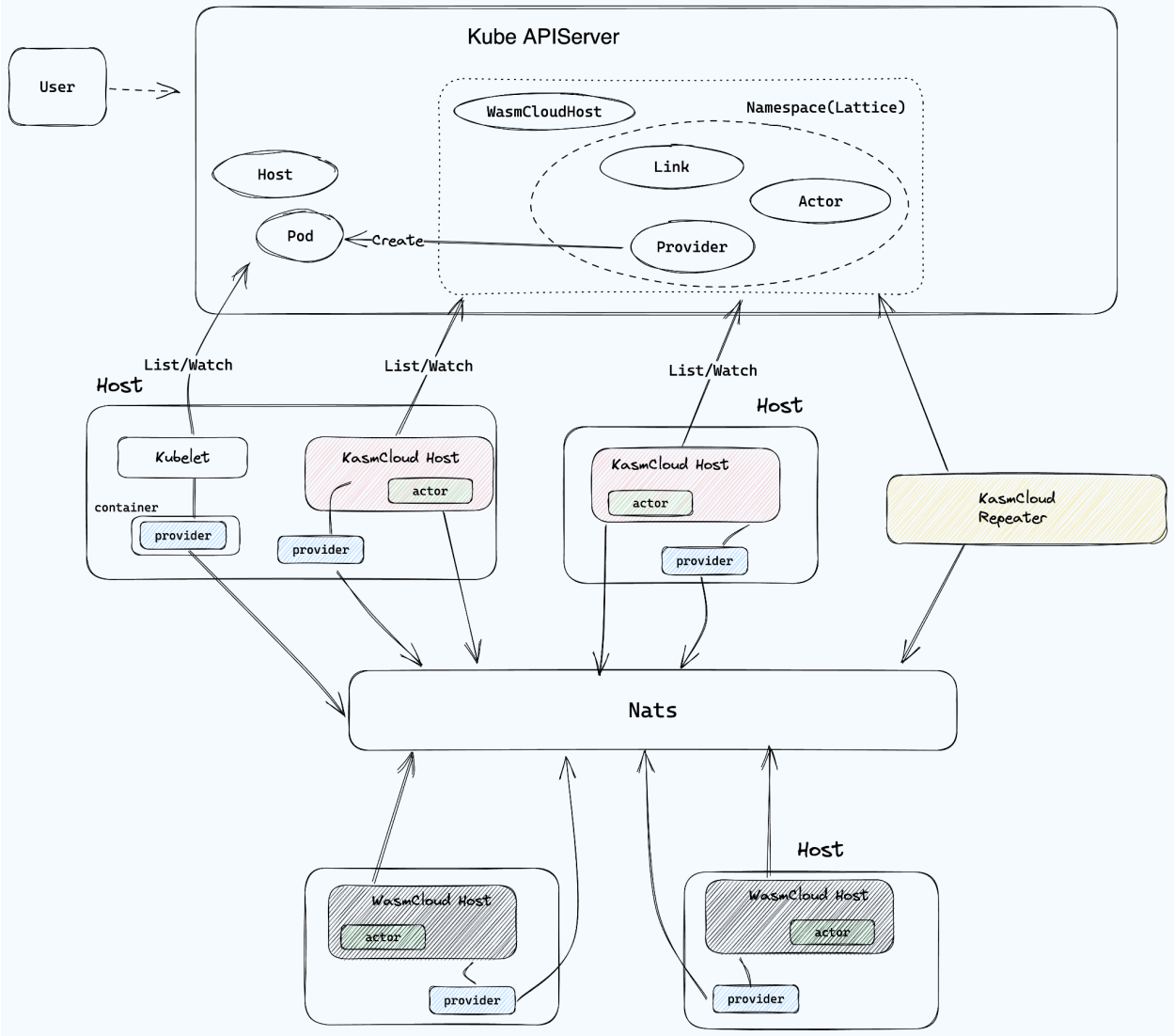
So next I describe the simple design of the new **KasmCloud** (Kubernetes + WasmCloud, the traditional Kube ecosystem nomenclature 😄

I think the best part of **WasmCloud**'s design is the use of NATS to decouple Actor and Provider calls, and **KasmCloud** is designed for this pattern, and Providers will be able to run as containers as well!

For kasmcloud, it has some core differences:

1. the use of Kube's declarative management of resources, in addition to the declarative features, **it allows users to use and manage KasmCloud's Provider, Actor, Host, and more advanced application resources in the Kubernetes-based management platform.**
2. Providers can be made to run in containers, Wasm is a secure sandbox, but the provider is previously free to run on the host. Although we can run wasmcloud host in containers/pods, **it's easier to isolate and control resources by having the provider run in a single container.**
3. we could probably make any program a provider, by which I mean they could all call actor through a library.
4. like the *Cosmonic Kubernetes Applier*, kasmcloud will communicate well with other services within kubernetes, and this capability may require the use of the *Cosmonic Kubernetes Applier*.
5. etc... (I haven't finished thinking about it yet😂)

Architecture:



The KasmCloud Host can run in Kubernetes as a container, either as a Deployment or a DaemonSet.

CustomResource:

Based CustomResource:

- WasmCloudHost
- WasmCloudProvider:
- WasmCloudActor

- WasmCloudLink

The prefixes are used to differentiate from other resources, so that when Kind has the same name, *kubectl get* won't work the way it's supposed to. Of course, we can also omit the prefixes for some resources, such as Provider, Actor, Link, but this is just a DEMO design

Advance CustomResource:

- TBD: Define more advanced Actor, Provider node distribution related definitions, somewhat similar to WADM

Components:

- **Kube APIServer**: Manage **KasmCloud**'s CR resources and provide List-Watch
- **Kubelet**(optional): Can run other users' pods, or **WasmCloud** Provider Containers.
- **NATS**: **By default, it is used as a relay for Actor and Provider calls, and perhaps for other purposes as well.**
- **KasmCloud Host**: This is mainly used for running the Actor, but you can also choose to run the Provider binary in the traditional way using **KasmCloud**.

There should be some other functionally related components included as well

Of course, we also need to ensure that the legacy **WasmCloud Host** can access the **KasmCloud** cluster, and we need to deploy the following components

- **WasmCloud Host**: In some nodes, connect to NATS.
- **KasmCloud Repeater**: Compatibility between KasmCloud's CR resource management and traditional WasmCloud's resource management and command control.

The purpose of supporting the running of a legacy WasmCloud is to ensure that WasmCloud can be run and managed based on NATS alone, and can also be connected to a KasmCloud cluster as an option.

A traditional WasmCloud Host does not connect to the Kube APIServer, but instead uses NATS as a way to invoke commands, event delivery, and the KV Store, which connects to the same NATS as the KasmCloud cluster.

KasmCloud Repeater

The **KasmCloud Repeater** listens to the KasmCloud NATS and coordinates with resources within the Kube APIServer:

- Watch for WasmCloud Host access in NATS and create WasmCloudHost in APIServer.

- Observe the creation of Actors and Providers in the WasmCloud Host and synchronize them to the APIServer.
- Simulate the behavior of WasmCloud Host for KasmCloud Host so that WasmCloud Host can discover and be compatible with KasmCloud Host.
- etc.

Questions

Minimal deployment of KasmCloud

* K3S APIServer
* NATS
* KasmCloud Host

No Kubelet required, just an APIServer compared to traditional WasmCloud.

Lattice's relationship to Namespace

Tentatively, a Lattice corresponds to a Namespace, Link, Provider, Actor resources are NamespaceScope resources, and WasmCloudHost resources are ClusterScope resources.