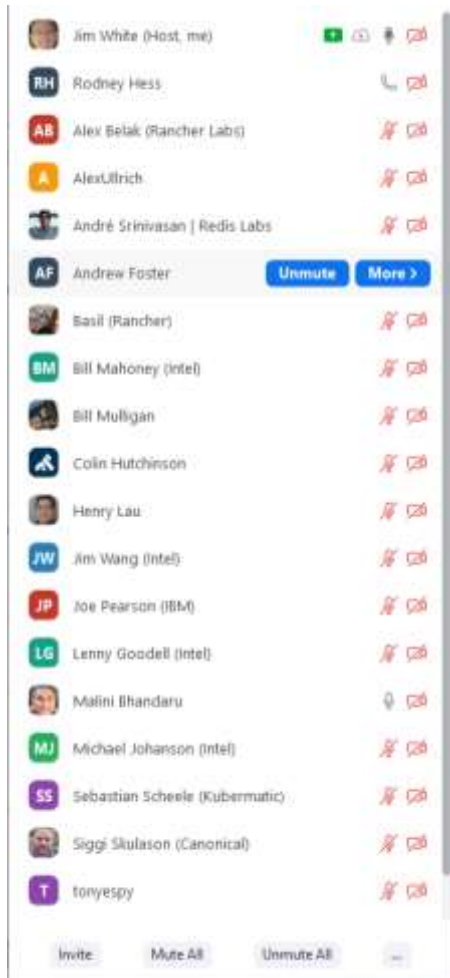


Kubernetes Sub-project Meeting

9/28/20

Attendance



Others may have been in attendance after roll was captured. For this inaugural meeting, participants did introduce themselves. Listen to the recording ([here](#)) for details on each of the attendees.

Agenda

Discuss the crawl, walk, run, etc. phases of EdgeX facilitation of Kubernetes (K8s), why we are doing this as a project, which distributions to support (full K8s, K3s, MicroK8s, etc.), and the schedule we take to achieve the goals.

Discussion/Decisions/Action Items

Why do we want to do this? That is, what is the purpose in facilitating Kubernetes within EdgeX? Answers:

- To offer alternate deployment/orchestration capability (beyond plain Docker containers, Docker-Compose and Snaps) to our adopters. K8s being one of the most highly used tools in the enterprise/cloud space and the one many companies are adopting.
- At some point, build EdgeX to be able to take advantage of the capabilities and services of K8s and high availability (HA) environments such as failover, reliability checks, and even automatic creation of multiple instances of services to address scale needs.

What parts of EdgeX would be included in an K8s EdgeX deployment for the “crawl” phase?

- Deploying a single instance of EdgeX to a single [pod](#).
- Inclusive of Redis.
- Inclusive of Virtual, REST and MQTT Device Services (those that do not require communications with a physical device). The virtual device, in particular, provides the means to run the entire system, affirm it all works and “smoke test” the deployment.
 - With regard to MQTT, a broker would also have to be included.
 - To be researched as part of this first support example.
- There was some discussion as to whether to include Consul. Could we use a cloud provider’s service registry as an alternative? Could we use local configuration?
 - The decision was to include Consul as-is for now.
 - **This is a research spike for a later phase (or an acceleration project for someone in the community to help with).** – that is, could EdgeX run without Consul and use a cloud provided service registry and configuration as alternate?
 - Could we use ConfigMap or native capability?
 - For both configuration and registry, with our abstractions in place for both, what implementations could be provided to fill-in behind these abstractions.
- There was some discussion about whether or not to include security services in this first phase (Vault and Kong).
 - The decision was made to include security as solving security in the crawl phase should help to reduce risk going forward – or highlight additional problems. Security will be more important (and potentially difficult) when each service runs in a different pod.
 - For this crawl phase, we are really talking about support for/inclusion of Kong as the major task as Vault is used under the covers.
 - K8s might lend security tools/facilities that can be embraced going forward (to replace or augment Vault/Kong). **This is a research spike for a later phase (or an acceleration project for someone in the community to help with).**
- In this first phase, we should be explicit about ingress to the cluster – that is, we need to make a statement about what types of EdgeX access is going to be allowed and supported from outside of the cluster.
 - For the foreseeable future, we will support access of EdgeX APIs through Kong.
 - We will not yet support use of CLI or the UI (these are not written to work through Kong yet)
 - We will not support direct access to Redis from the outside.

How will we facilitate K8s as part of the crawl phase?

- We want to initially deploy EdgeX in a single pod (considered the simplest thing first). Again, device services, except those listed above, could be deployed outside of EdgeX pod.
- The desire is to adhere to CNCF distribution APIs and standards so that EdgeX could be deployed to any K8s distribution (K8s, K3s, KubeEdge, MircoK8s, etc.).
- Stretch goal would be to walkthrough or show example deployments to various distributions. [Action item for acceleration and opportunity for members of the community to help with.]

What will we deliver for the crawl phase?

- #1 - YAML file (deployment.yaml) that allows EdgeX as described above to be deployed in a K8s distro environment. Some documentation to provide a how-to-use, explain what is in the YAML, outline limitations, etc. Jim to work with Gavin Lu of VMWare to see how much of this has already been done and if his team can provide this solution by Hanoi.
- #2 - A document outlining how to take a EdgeX Docker Compose file (found <https://github.com/edgexfoundry/developer-scripts/tree/master/releases>) and deploy to a K8s pod using [Kompose](#). Malini Bandaru of VMWare is taking the lead on providing this document/how-to-guide.
- Both the YAML deployment code / example and the Kompose example will be put in [EdgeX examples](#) under a folder called “deployment”
- In the future, this work or related K8s work may be placed in a separate repository.

Schedule and Next Steps

- The goal is to complete the crawl phase by Hanoi release (early November).
- The next step is to explore the next phase (for Ireland delivery)
- Jim will convene a meeting for late October or early November. At this meeting, we'll get a read out from the teams delivering Hanoi tasks.
- We will try to outline the use case/requirements for high availability better.
 - What services should be replicated to address scale?
 - What other K8s services would EdgeX look to leverage in a full K8s/HA environment?
 - What services or infrastructure need to be researched as to HA concerns?
- We will try to outline what is in the Walk phase.

Additional notes

- Helm or Operators could be more customer driven.
- Helm or Operators (or other technology selections) may be given to certain camps within the broader CNCF community and EdgeX will want to remain unbiased or supporting all major camps equally.