Architecture Topics – Discussion/Decisions

4/28/20

Secrets for message bus connection

We don't want separate docker image for creating database, message bus, etc. secrets. We would prefer to have all secrets generation as part of security secret store setup (adding trigger for new secret generation there).

How do we generate/inject the random passwords into vault?

- We need an abstraction and pluggable mechanism to provide generation of secrets (current Go Key mod is not fit for production) – security WG to take this as first high priority task – what is the ref impl going to be.

How do we consume them in the service?

- Depends on the 3rd party software – needs a shim piece of software to provide (and kicked off from security secret store setup)
- How the "shim" works in order of preference
    - Talk to Vault directly (or piece of code that talks to Vault and then inject's to 3rd party)
    - Injected env var (as a way to pass the secret to service)
    - Secrets file on RAM disk (or other protected location other than Docker volumes; this could be Swarm or Kubernetes secrets – temporary file system that doesn't disk; have to consider reboot circumstances and how to handle)
    - Command line argument

As a side project, we need to explore Consul Templates. We might be able to take better advantage of Vault and Consul Templates for this, but not sure how that would work with 3rd party systems (like Redis). Colin will lead a sub-project in Security WG to explore Consul Templates.

How do we consume them from a client perspective? – here we are good; we get them from Vault

- Approach is encoded in go-mod-secrets
- Other languages (C for DS) needs an equivalent of go-mod-secrets

Distribution of DS (security between DS and Core)

Explore how-to options

SSH

- Pro - Very simple
    - Lower overhead approach of the 3 options
    - No code changes
- Con – SSH and SSHD not installed by default on Windows
- Work needed
    - Document what to do (how to guide)

    - Config changes; docker compose file changes in document (don't need to provide the whole compose)
    - May need some special containers (for ssh/sshd) or instructions to do that

Overlay network option

- Use Swarm, Kubernetes or manually construct the network
- Pro - No extra components needed
    - Added orchestration support
- Cons – Consul chokes on multiple networks (must be resolved)
    - Solution bound to orchestrator (Swarm, Kubernetes, etc or have very complicated networking)
    - Overhead of Swarm or other orchestrator
    - Unfriendly to Snaps
- Work
    - Document what to do (how to guide)
    - Have multiple docker compose files (one for the device service and one for the rest of EdgeX)

Service Mesh

- Add service mesh (Kuma) to all containers
- Pro
    - No code changes required
    - Developers make no changes in what they are doing it
- Cons
    - Document what to do (how to guide)
    - Config changes; docker compose file
    - very Kuma specific but with implications that it service mesh generically could solve
    - Size /complexity of the compose file grows
- Work
    - Document what to do (how to guide)
    - Possible some script to automate

Priority of work is to:

1. SSH
2. Overlay network
3. Service mesh

Colin and Tingu to explore use of Kong and restoration of the Kong routing option.

EdgeX Metrics Collection & Control Plane Event Handling

Points

- Separate metrics collection from events like new device onboarded

- Notification service is something separate. Notifications is about sending an alert to a 3rd party off box (using email, SMS or other means). It is not about handling or reacting to something that is happening.
- We have to take care not to saturate any part of the system with too much data
- Configuration of the service's collection of metrics or handling of events could be done by standard config/Consul mechanisms. We don't necessarily need a new means to provide this or to change it.

Decisions

- Metrics collection should be collected by service (as it sees fit but in a way that is generally generic and configured in a similar way for all services).
- Metrics collection structure is to be determined but would be different than event/reading structure used for sensor readings.
- Services will use go-mod-messaging to post their metrics data messages to a message bus. Collection schedule, which metrics are collected, and where the metrics are sent (i.e. which topic) are defined by the service (and configurable).
- Application services are optional receivers of the metrics collected data. Application services must be refactored to be able to handle this new structure/type of data and be able to prepare it and send it to message topic, REST endpoint, etc.
- Control plane events also require a separate and distinct structure.
- Control plane events will also go through go-mod-messaging, but to a separate topic.
- Application services will also optionally be used to receive and handle control plane events, but implementers may also choose to hook something directly off the event topic to trigger other work/actions.
- System management WG will refine ADR0006 based on these decisions.

More ties to Akraino Blueprints/more liaison with LF Edge projects

- Consensus from community is that much of the work inside of LF Edge seems to be bearing little fruit at this time. Doesn't seem very applicable to what we are doing or to be creating integrated solutions.
- Jim and Ramya will continue explore options to expand use of Akraino test lab with other blueprints and to see if the Akraino lab environment can be used to set up on demand EdgeX instances on ELIOT blueprint for exploration, education, prototyping, etc.
- Jim will try to follow up on additional opportunities for project-to-project collaboration.

Automatic migration of devices between device services

- Keep Trevor's options for how to handle this in the icebox for now. It does a good job of laying out options.
- We don't have enough use cases or device types/protocols that need this right now to hurry to an implementation. BLE DS, to be provided soon, is an example device service that might provide more insight / applicability.
- Table this topic until next release planning meeting to see if we should move on it then.