System Management – Organization and Assignment of Responsibilities

Some recent questions have arisen with regard to system management and SMA that require us to (re)visit and (re)define EdgeX responsibilities.

Assumptions:

- Registry/configuration service (currently provided by Consul) may or may not be up and running or be used. In development, for example, a service can use local file configuration and not register with any other service
- The System Management Agent (SMA) may or may not be up and running or be used. The SMA is in place to facilitate other outside-EdgeX systems. It provides a single-point contact element for health and status information about EdgeX services to 3rd parties (via REST calls today) and also facilitates 3rd parties to make requests to start, stop or restart EdgeX services (one, many, or as a whole).

Discussion

- Now that we have an abstraction for registry – if we have a file base system as a last resort, then could you have a requirement for a registry? So Registry ~~service~~ provider always has to be up.
- It's not as simple as just make it file based, because that is just config. What about "registry"?
- You get a poor-man's registry by going to config file and get location of services through config.

Responsibilities

1. Responsibility of knowing what services are running (for example when a request for all service metrics, how does it know "all services"?)
   a. Should the SMA innately know what services are up and running?
      i. It would require the SMA to be given a manifest.
      ii. If a new service is started, the SMA manifest would need to be updated and restarted, or the new service would have to register with the SMA.
   b. Should the registration service (currently Consul but with a wrapper API soon) know what services are up and running?
      i. If the registration service, what happens if EdgeX is running without the registration service (development or lighter weight deployment)?
      ii. Would the SMA ask the registration service what services are running? If this is the case, the registration service would have to be up before the SMA (or at least be up before the SMA can do anything). And if the registry service is not being used, what would the SMA consult with to know what services are running?

Discussion

- Under a registration interface, Consul would be first to understand what services up and available
- If Consul is not there, then SMA would be under the interface and know what services are up and running
- The Registry would have a "type" provider that could be Consul or SMA.

- If Consul and Registry are not running, then services are own their own (registry flag would be off)
- We want the SMA to be as light as possible, so the registration part would be light weight
- Would we ever have a use case where SMA is up but not a real registration/config service (like Consul)? - possibly in high-availability future concerns where there are many services
- Deregistration of service is now needed, but how do we handle the crashes when a service can't deregister?
    - Could have logic in SMA to try to restart after certain time and serving the role of the registry.
    - We need to have some configurable setting for when to restart/retries
- New device service registers itself with the registry interface (and whatever the implementation is under the covers).
    - SMA can't cash the registry info when it is not the registry provider; it has to get that from the registry service for each request

2. Responsibility of registration
    a. Is a new service expected to register itself as up and running (dynamically register) with the registry service and/or SMA when it comes up? If so, does it have to unregister as well?
    b. What if the SMA or registry service is not being used? How are the services expected to know who to register with?
3. Responsibility of start (what is first up?)

SMA/executors is not the bootstrapper – something else is

After bootstrap, then the SMA/executors can start or stop or restart services

a. Is some component of system management going to be used to start EdgeX for the first time on some deployment? If so, does it not have to be first up? The responsible element has to be brought up first and called on in order to perform the very first start of EdgeX.

b. Whether the SMA or executor is requested to start other services, it has to be brought up first (when it is used) and then it has to know the other services to start. How does it know what services to start (back to the manifest issue and knowing what services are available)?

4. Responsibility of stop and restart

Why would someone bounce the SMA? Use case – what about config change and a need to bounce everything?

SMA can tell the excutor to restart the SMA and it (the executor) can be in charge of status/results.

a. If the SMA is requested to restart or stop other services, can it call to restart or stop itself?

b. If it does stop and restart itself, how can it keep track of what is restarted or stopped? Does it need to?
c. In the case of using docker-compose, a call to restart would cause the container with the executor to be restarted – causing an issue in that the executor would have to know to stop or restart the SMA container last. Could this even work?

Recommendations

- We've already said that the registry service is the home of all config knowledge when it is running. If not running, services use their own local config
- We already have services trying to contact and register with the registry service by default. The registry service also does a regular ping to check if a service is still up an available. Services can even come up long after the initial start of EdgeX. The same registration is required/would apply.
- We need to have services also have the responsibility (in the future) to deregister with the registry service if they need to be taken down. If the registry service is not there, then no deregistration can occur.

With the services responsible for registering and deregistering with the registry service, the registry service is the source of information about what services are registered as well as configuration.

The SMA, should therefore,

- Call on the registry service first and foremost for the current list of registered services whenever it needs to know what services are currently up/running. As the registry service allows for the dynamic coming and going of services, the registry service becomes the source of truth about the services the SMA should know about at any given instance.
- In the event that the registry service is not up, the SMA will use a local manifest of services and expect these services to be the ones that are up. There is no dynamic registration or deregistration in this case. In this case, if new services are brought up, the SMA's local manifest must be altered and the SMA restarted.

Start, stop and restart

- The SMA can be brought up first and be used to start everything. If it does this, it must rely on the local manifest for the list of services to start. The registry, as one of the services the SMA brings up, would not be up yet.
- The SMA will use the registry first and then the local manifest (per above) for a list of current services whenever it needs to stop or restart services.
- The SMA could stop or restart itself as just another EdgeX service, but it is system/executor implementation specific. Some executor situation may not allow for this to happen or would require stop/restart to happen in a particular order. Therefore, the ability to restart or stop the SMA is left to executor implementation.
  - Research: Can an SMA->executor->Docker Compose call to Stop or Restart itself: could a call from Docker Compose from inside the Docker-in-Docker container holding both the SMA and the Docker Compose executor work? Wouldn't it have to be the last of the services it stops or restarts if a collection of services is sent in?
  - How would the SNAP implementation work?

Feedback from Trevor

- It could be that the SMA should serve as a registry when Consul isn't present. This would require that we add to the [Registry] section in the config a "Type" key with a value of Consul or SMA, similar to how we do with databases. We have a Consul provider in the go-mod-registry module now, so maybe we should add one for the SMA.
  - I think we should also support a Type of "None" where there is no registry at all – just straight service to service communication based on static configuration
- The more I think about it, I don't know that I'm in favor of a static manifest as a document that is deployed with some EdgeX installation. It's too confining. If we view the SMA as a registry when Consul isn't present, then services would register themselves through the same abstraction they use today.
  - This would allow us to remove the individual client configuration sections from the SMA configuration.toml since services would be pushing their host/port/key
- The state of what services are registered should be held in a database in case the SMA goes down.
- The SMA should have the responsibility of performing health checks in deployments where it is serving the role of registry.
- I do think we need to implement the service Unregister capability.
- With regard to the need for the SMA to come up first, we can use the same service resiliency pattern we currently use for Consul. If the SMA is simply a provider behind the go-mod-registry abstraction, as described above, this won't even require any work.
  - If the SMA were to not come up successfully, an option would be falling back to local config.

I think we should also be specific that this role is only recommended for the SMA in a smaller, localized deployment. If someone's doing HA/Scaling, they should use Consul.

From Akram

- While it's true that the SMA may or may not be up and running, or even be used in the first place, we still need to make good on the promise that the SMA is in place to facilitate other extra-EdgeX systems when the SMA is being used.
  - For the scenario where Registry is down (or not being used), I agree that we should have the SMA itself serve as a Registry mechanism (with persistence capability).
  - And tying in with providing an SMA-centric mechanism for maintaining the state of what services are registered, a database mechanism sounds more flexible than a static-on-creation (yet updateable) manifest, referring here to the point in the doc (regarding how "the SMA's local manifest must be *altered* and the SMA restarted")
- It makes sense—for the long haul—to regard the SMA as more than a pass-through to the Registry (specifically thinking here to the one recently contributed to EdgeX by Lenny (Intel)) since the Registry (as in the *interface* to the underlying registry implementation, such as Consul) may be unavailable.
  - Exactly, if Consul is up, we won't even require any work on the SMA end since we'll be able to, in that scenario, simply pass-through incoming "config"-utilizing requests to the Registry.

- Yes, we should provide a deregistration.
- We will need a mechanism to keep the state of services, as registered with the SMA, and in turn persisted to the manifest (either static-on-creation or database-based) to remain in synch with the corresponding state of services, if users are expecting either of SMA or a Registry-mediated registry (aka Consul) as being available.