Hanoi Pre-wire

March 31 – April 1

Attendance

Geneva Issues

- Race condition testing.
    - Actions
        - Mike & Iain – own task of looking at SDKs and associated services and make sure the makefiles include the race detector (-race command line flag for the Go tool set) in Go builds and tests
        - James indicate that there is a well known issue that race flag does now work with Alpine based images (as in CI/CD). Solving this issue is not in scope for Geneva. The work of instituting use of race flag in CI/CD must be postponed to Hanoi. See https://github.com/golang/go/issues/14481.
- Config seed PR – do we need mini-freeze for Lenny's PR?
    - Lenny keeping up with rebasing
    - Some bugs still being worked but nothing that requires mini-freeze at this time.

Architecture Tenet Discussion

Should EdgeX consider new tenets or architectural approaches given suggested needs of HA, more scalability and distribution support?

> Statement about future project direction
>
> The shape and size of "the edge" is changing. When initially conceived, EdgeX was designed and built to operate on resource constrained edge devices – such as edge gateways and small compute platforms like a Raspberry Pi (representative of the edge compute resource constraints).
>
> Today, edge computing is expanding. Edge computing still addresses computing where the physical sensed world and operational technology (OT) meets the digital world and information technology (IT). While this was traditionally thought to be at the furthest extremes of computing resources, this is no longer the case. Now, OT environments collide with IT in environments

where compute resources are much more plentiful. In fact, the trend is to see edge and IoT computing expanding into use cases (exemplified by retail department store or smart factory environments) where compute resources are nearly equivalent to those in enterprise environments. The compute, network and storage availability in these environments support edge applications that can behave more like enterprise applications – that is be more distributed, more highly available, and scale more gracefully and would otherwise be indistinguishable from cloud-based applications.

To be sure, there are still use cases and edge application needs where compute, network and storage resource constraints require edge applications to remain relatively small and/or flexible to be distributed across the limited resources at the edge.

EdgeX remains committed, for now, to be able to be run on limited resource devices (Raspberry Pi in dimension) and environments (intermittent connectivity, shallow storage devices, etc.). It remains committed to its founding architectural tenets, optimal flexibility principals, and micro service architecture. Without endorsing new architectural tenets or principals at this time, EdgeX will however keep an eye on edge expansion and defining use cases. Depending on user requirements and the evolution of edge compute, EdgeX may embrace the needs of higher order compute environments in the future and lean toward

- Removing services (in favor of technology provided infrastructure – for example removing a logging service in favor of a central logging facility provided by the deployment/orchestration technology)
- Collapsing services (combining services that would not ordinarily be separate micro services in a more resource rich environment – collapsing services to save on complexity and service communication overhead)
- Instituting high availability design into services (such as 12 factor application design principles into services to support HA infrastructure and tooling)

The degree to which the project moves in this direction will be reviewed with each release planning session and subject to user requirements dictating any move in this direction.

If we can align on these basic questions, there is a feeling that this should start to guide our opinion on scope for Hanoi and even beyond.

Hanoi planning

Items listed here are considered potential scope for Hanoi and will be reviewed at the Hanoi Planning Conference.

General / Cross Cutting Concern

- Probably going to be a 1.3 vs 2.0 release
- Ability to institute all of the V2 API and some of the desired non-backward compatible features during one 6-month cycle is dubious.

New feature – tentatively included in Hanoi release

- Move to Go 1.15
  - Anthony Bonafide to do research and provide impact statement (also consider 1.14)

- Implement a message bus between device services and application services. As a stretch goal, core data would be an optional subscriber on the bus (persisting Events/Reading found on the bus); otherwise this feature would assume persistence is off
    - Lenny to present high-level design at the planning meeting (including recommendation on broker vs brokerless vs option for both)
- Allow device services to be distributed to alternate hosts.
    - Must address questions of security and orchestration
    - Must address device service to core as well as command to device service communications
    - All TSC members are requested to be familiar with Bryon's Option B regarding security
    - Bryon/Tony to send out slide deck or document to TSC for review that refines the Option B proposal and looks specifically at DS distribution question.
- As Hanoi is looking less like a version 2.0, should we reconsider LTS? Should the next 1.x release be LTS?

Tech Debt - tentatively included in Hanoi release

- V2 API implemented as "experimental" or alternate API while V1 is deprecated but still available
    - IOTech to provide readout on timing and considerations
- Deprecate Mongo
    - Requires Redis with secure access to be on first (Jim checking with Andre)
    - If so, mark Mongo as deprecated for Geneva and explore ridding project of Mongo for Hanoi.
- Remove value descriptor
    - For backward compatibility, can only be removed if in a 2.0 situation
- Devices with embedded device profile and removing addressable from EdgeX
    - For backward compatibility, can only be removed if in a 2.0 situation
- Develop process for security and license vetting of 3rd party components
    - Security WG (Tingyu and Diana with WG inputs) to provide a high level idea of the process and what it would require of developers and work groups
    - License compatibility with project also to be considered part of this review
- Getting consistent and legal with license and attribution files/marks
    - This topic is schedule for upcoming architect's meeting and will be reviewed at planning meeting if not sufficiently answered
- Restructuring our compose files to take advantage of compose file overrides, which removed the duplication in all our compose files. See https://devilbox.readthedocs.io/en/latest/configuration-files/docker-compose-override-yml.html
    - Lenny to provide readout/demo for consideration at the planning meeting

Core / Supporting

New feature – tentatively included in Hanoi release

- Drop the logging service
    - Turn off persistence of the logging in Geneva. Change the logging configuration in each service TOML should have the following settings.
        - [Logging]

        - EnableRemote = false
        - File = ‘’
    - This will cause all services to log to stdout. Overrides can be done by the user for their deployment.
    - Use bootstrap to write to standard out and optionally file
    - This is a non-backward comp change, therefore can not be removed in dot release
    - Mark deprecated in Geneva (and remove it from compose) and remove whenever we graduate to v2.
    - UI Concerns – does the UI look at logging? Jim to check with UI group

Tech Debt - tentatively included in Hanoi release

- Tests should run cleanly when passed the "-race" argument
    - Will require DevOps assistance
- Restructure the Redis DBClient implementation for Event/Reading
    - Normal refactoring exercise – does not require consideration at the planning meeting but will be looked into
    - https://github.com/edgexfoundry/edgex-go/issues/2166

Application Services / App Functions SDK / Analytics Integration

New feature – tentatively included in Hanoi release

- Kuiper – explore with EMQ team and what parts are EdgeX responsibility
    - Decouple Kuiper from EdgeX – go-mod-contracts and messaging
    - Walk stage
        - Handles multiple devices and filtering in the rules engine itself (discuss with Kuiper team)
        - Enhance REST to match CLI??
- Deprecate Drools rules engine
    - Drop Kuiper in compose now (Geneva)
    - Label Drools as deprecated
    - Remove Drools in Hanoi
- Provide an example of how to accomplish enrichment of data in an application service – building a custom app service via the SDK.
- Define metadata about the “gateway” or host platform (identity, location, …)
    - Potentially make that information/metadata extensible
    - How can we then make metadata of gateway available with app service export
    - Goes beyond app service
    - Malini – action item to layout what this feature includes/high level design for planning
- Fork the pipeline & have a pipeline per topic
    - Support multiple topics in SDK - go mod messaging
    - Allow various paths for different sensors for example
    - Can be done with code internal to a function - but a bit kludgy; could be useful and nice
    - Lenny/Mike – to provide read out on design/scope level of priority

Tech Debt - tentatively included in Hanoi release

None at this time

Device Services / Device Service SDKs

New feature – tentatively included in Hanoi release

- Protect the device from harmful commands, there should be the possibility to set a Min and Max limit (or other profile checks to protect the device).
    - There is a preexisting issue for this
- Data filter design between DS and Core Data
    - Provide a design about how to implement this before implementing.
    - If possible, can the filter functions be shared across App Services and D.S. (w/ App WG)
        - Iain, Tony, Steve & Mike, Lenny to chat before planning meeting about possible alignment, use of app funct pipeline architecture?
- Bound checking
    - Number of operations that can be done
    - Max request size (that lends to DoS, etc.)
    - Could be more globally applied – a REST QoS
    - Each WG should explore implications and design
    - Articulate the problem better; limit scope a bit; design target for Hanoi

Tech Debt - tentatively included in Hanoi release

None

Security

New feature – tentatively included in Hanoi release

- Secure service to service communications
    - For Hanoi – secure the DS to Core/Appl Service (as DS could be distributed to alternate host)
    - As stretch goal, secure all service to service comms.
    - Bryon to provide the readout/high level scope
- Create a hardware secret storage design
    - Define hardware secure storage abstraction layer
    - Figure out how to protect the Vault Master Key
    - Bryon working this for Hanoi planning meeting
- Signing docker images
    - Malini and Bryon to do some research on what VWMare/Intel do
    - What do most in Docker Hub do?
- Enable user-specified Kong proxy certificate
    - BYO Cert for external only
- Configure TLS encryption for kong+postgres channel (one way TLS)
    - Cert generation for services.
    - Special case of service of service TLS (Postgress vs our service)

Tech Debt - tentatively included in Hanoi release

- Secure Consul – currently there is no communication protection with Consul
    - There is no authentication/authorization when seed into Consul
    - Malini / Tingyu to provide readout on scope, high level how-to on this
- Blackbox tests of APIs through the API gateway

    - Determine what else we need to do. Do we need “negative test” – that is try to use the services when security is on and make sure we get rejected on APIs we don’t have access to
    - Should be part of the TestQA Hanoi planning?
    - Tingyu to find out what we have and what we might need
- Container security (running of the reference images via compose file)
    - Set no-new-privileges option on containers
    - Make docker container images read-only
    - Make docker container images run as non-root users
- Secure Kong admin port with TLS
- Create an ADR for secrets provider (abstraction for go-mod-secrets)
    - Lenny to create the initial ADR

System Management

New feature – tentatively included in Hanoi release

- Better facilitate Kubernetes
    - Provide Support Helm Chart, Pod yaml, Draft, Skaffold
    - Jim to provide readout/scope

Tech Debt - tentatively included in Hanoi release

None

DevOps (per DevOps WG 4/2/20)

New feature – tentatively included in Hanoi release

- Performance Optimizations
    - Jenkins Pipeline optimizations for edgex-go
    - Explore options from LF for supporting Jenkins on K8s
- Performance of the Build Environment
    - Monitoring / Alerting optimizations (Continuous Improvement Opportunity)
- Open Horizons Enablement
    - Shared Infra with Open Horizons
    - Build Automation for OH
- Code Coverage for Jenkins Global Libraries (codecov.io) - stretch
- Snap improvements - stretch
- Support for – race flag - stretch

Tech Debt - tentatively included in Hanoi release

- Caching Dependencies - speed it up (upstream dependencies)

Certification (to be covered in upcoming WG)

??

Vertical Solutions (to be covered in upcoming WG)

??

Test / QA / Documentation (to be covered in upcoming WG

New feature – tentatively included in Hanoi release

- User guidance on platform needs
  - More performance statistics
  - # of devices/per recommendations
  - Providing EdgeX users guidance on platform needs, sensor data throughput and deployment based on performance metrics. Specifically, with the Geneva performance testing apparatus, the EdgeX community will be able to answer these questions for the user:
    - Will EdgeX fit on my system? - size of EdgeX services, infrastructure, etc. and hardware/platform requirements
    - What is the speed of data through the system? - from device service sensor data ingestion to the rules engine and back down through command to another device service to trigger a put command, how long does this take?
    - How many "things" can be processed at a time? – with caveats on the type of thing, type of data, etc.
    - These questions need to be answered on real hardware (both Intel and ARM)

Tech Debt - tentatively included in Hanoi release

- Device Service testing – complete testing for current set of EdgeX Device Service (w/ DS WG)
  - Virtual, Modbus, OPC UA and BACnet done or being worked. Others to do.
- System integration / interoperability tests - Device Service read data -> Core Data -> Rules Engine or Application/Export Service -> Command
- Add unit tests/testing for global libraries. (w/ DevOps help)
- Blackbox tests to edgex-go repo?
- Blackbox tests against snaps
- Configuration testing – testing non-default and dynamic config changes

Open Horizons (to be covered in upcoming WG)

New feature – tentatively included in Hanoi release

- Should be a separate project in spring 2020.
- Roadmapping/direction is self determined
- Any additional next steps for EdgeX integration?