# Delhi

The Delhi release is the third formal release of EdgeX Foundry. Key features for the Delhi release will include:

- The initial system management capability
- Device Service SDKs in Go and C (along with some demonstration device services)
- The next wave of security features to include access control to grant access to appropriate services, and improved security service bootstrapping.
- Refactored and improved Go Lang microservices
- Options and implementation plan for adding abstraction for database (and other resources)
- A potential plan to replace the existing underlying database
- Plans for export service replacement with application services
- Plans for supporting binary data through CBOR
- Provide an EdgeX UI suitable for use in exploring several instances of EdgeX
- Provide a performance testing strategy and plan

## Release Information

- Provide a system management agent (SMA) that coordinates control plane information and management of EdgeX micro service metrics, configuration, and control (start, stop, restart) through a central service endpoint.
- Provide system management APIs into each EdgeX micro service in support of system management needs and facilitate the SMA.
- The SMA and micro service level management APIs serve as the first building blocks to manage EdgeX and its associated devices/sensors in the future.
- Device Service SDKs in Go and C will replace the Java device service SDK and allow the last Java micro services to be replaced (allowing to improve footprint and performance)
- Documentation and sample device services to assist developers replace existing large/slow Java device services, along with adding new device /sensor device service connectors.
- In California, some of the initial security services were put in place - namely the reverse proxy to protect the micro service APIs and a secure store for storing EdgeX secrets (like database passwords, certificates, etc.) Additional security capability will be added on top of these initial building blocks in Delhi. New security features to be added include:
    - Use an ACL plugin to provide access control to micro services
    - Adding various EdgeX secrets to the secure storage
    - Improve the secure bootstrapping
- The Scheduling service remains the last of the Java micro services in California (other than the device services and rules engine). This service will be replaced in Delhi with a Go implementation. Device services and the scheduling service will share a common metadata schema.
- Core, supporting and export services are refactored to offer better code organization, abstraction from certain resources (offering better flexibility) and improve unit testing.
- A design and implementation plan will be provided with Delhi (for Edinburgh implementation) to add support for binary data (namely via CBOR) through device services, core data, and export services
- Improve the configuration of EdgeX
    - Refactor config-seed to only load the configuration appropriate to the application setting based on profiles (i.e. development, production, production docker or snap, etc.)
    - Replace the simple key/value pair organization of service configuration with more structured and hierarchical configuration as supported by Consul
    - Upgrade Consul and improve the configuration seed
    - Implement a process to move the latest configuration into the config-seed through the CI process
- While MongoDB has been the foundational EdgeX data persistence store since its beginning, concerns around its license, ARM 32 support, and size/performance have the community looking into possible alternatives. At the very least, the platform should more easily support the swap of the database by 3rd parties. The Delhi release will provide designs and plans for implementation in the Edinburgh release of:
    - Better database abstraction architecture
    - A potential change or addition in database support (potentially replacing MongoDB)
- The existing export services will not scale because additional code must be added to the client and distribution services with each additional transformation, filter, formatting, or endpoint need. Further, it does not allow for eliminating filter, transformation, or endpoint distribution code when it is not needed. Delhi will include a design and implementation plan (for Edinburgh) to eventually replace the export services with a new application services (as initially specified in the document here).
- An user interface will be provided to assist manage EdgeX instances, demonstrate device actuation, and visualize the sensor data collected.

## Release Dates

Freeze Date: Oct 22, 2018

Release Date: Nov 16, 2018

## Release Docker Compose

https://github.com/edgexfoundry/developer-scripts/tree/master/releases/delhi/compose-files

Version0.7.1 Release Docker Compose file: https://github.com/edgexfoundry/developer-scripts/blob/master/releases/delhi/compose-files/docker-compose-delhi-0.7.1.yml