Certification Plan Document

Revised May 8, 2019

Table of Contents

- Introduction
    - What is EdgeX certification?
    - Why is Certification needed?
    - Benefits to Stakeholders
- Certification Levels
    - Self-Assessment
    - Formal Certification
- Technical Requirements
    - Assumptions
    - Submission Requirements
        - Core Service
        - Device Service
        - Database Service
        - Application Service
    - Testing Requirements
- Certification Process
- Road Map
    - Crawl - "Assess and Learn"
    - Walk - "Certification"
    - Jog - Automate and Formalize
    - Run - Deeper and Wider
- Commercial Issues
    - Eligibility
    - Fees
    - Version Control
        - EdgeX Changes
        - Component Changes
    - Revoking Certification
- Open Issues
- Glossary
- Appendix 1. Device Service Certification Notes
- Appendix 2. Needed Tools
    - Product and Development
    - Process
    - Marketing and Promotion

Introduction

This document describes a certification program for EdgeX Foundry. This document was developed by the EdgeX Certification Working Group to

- Identify the value of certification for the community
- Describe the certification process
- Define technical details for inspection of components seeking certification and success criteria
- Provide a road map for a phased implementation that delivers increasing value starting from day 1

Note: The term “certification authority” is used in this document to describe the entity that will be conducting the certification testing/evaluation. It is to be determined if this will be done within the EdgeX community or by a third party.

What is EdgeX certification?

EdgeX certification represents a process where software components that interact with EdgeX are evaluated and confirmed to meet a set of technical requirements and standards that will help insure compatibility and interoperability. This document describes the framework that will be used by the EdgeX Foundation to verify that software seeking compliance conforms to specified characteristics.

In general, the certification process will seek to confirm that components receiving certification will:

- Be compliant with the EdgeX APIs for its type of service
- Be interoperable with other EdgeX services
- Support the EdgeX system management agent
- Be robust; reliable, able to recover from faults, start in any order, wait for dependent services, etc.
- Provide documentation on functionality and deployment
- Not negatively impact the operation of other EdgeX services

Components achieving certification will be able to promote this achievement through

- Presentation of a letter of certification

- Use of certification logo
- Joint press announcement with LF Edge
- Being listed on EdgeX web site as being a certified component

EdgeX Certification is initially intended for components that replace core services or interact with core services. As the certification program matures, there may be reason to certify accessories or applications that are outside of that scope. For example, third party applications that interface with application services.

For the purposes of this document, we are focused on core services and those services that interact with core services (e.g., core. device, and application services).

Why is Certification needed?

EdgeX Foundry is based on loosely-coupled micro-services bound by common APIs. Entire subsections can be replaced, combined, etc., with proprietary, differentiated “EdgeX-compliant” offerings. The community and users want assurance that EdgeX will continue to function as designed even when components from other sources are added or replaced.

Certification is critical for multi-vendor ecosystems. By ensuring the services can be easily and reliably integrated to drive an outcome, users have reduced risk and greater confidence in adopting EdgeX and third party components.

Benefits to Stakeholders

There are three primary stakeholders for EdgeX certification. These include the EdgeX developer community, third party partners that use and build products around EdgeX, and the end users that are employing EdgeX to implement projects.

The benefits of certification for each group are listed below:

EdgeX Developer Community

- Defined test and quality standard
- Accelerated market adoption
- Demonstrated commitment to APIs

EdgeX Partners

- Commercial opportunities to leverage EdgeX community
- Proof of compliance for greater market acceptance
- Reduced support issues

EdgeX Users

- Confirmed interoperability as of a given EdgeX release
- Faster implementation
- Reduced integration risks
- Higher reliability

Certification Levels

The EdgeX certification process will consist of two levels: self-assessment and formal certification.

Self-Assessment

The self-assessment process enables an organization to conduct their own evaluation using EdgeX provided tools to confirm meeting basic technical requirements. This assessment will be a prerequisite for submitting a component to EdgeX for formal certification. Self-assessment alone may be useful for organizations to confirm their code even if they have no interest in pursuing formal certification.

Benefits

- Promotes a baseline public standard for validation of EdgeX APIs
- Confirms submitter has done their homework and cleared basic issues before entering the certification process
- Demonstrates EdgeX community commitment to standards around certification
- Can be used by developers to privately check compliance of their code

EdgeX Foundry will make available a set of software tests and procedures that developers can download and use to conduct their own readiness assessment.

There is no registration process required for self-assessment. The tools and instructions for assessment will be publicly available.

Self-assessment is not formally recognized by the EdgeX Foundry. I.e., it does not convey the use of “EdgeX certified” in marketing materials or listing in the EdgeX certified products list. However, it may be used as a qualifier for inclusion on lists of "EdgeX Ready" components.

Formal Certification

Formal EdgeX certification testing will be conducted by a certification authority designated by the EdgeX TSC. Components that successfully complete the certification process will be announced on the EdgeX web site and those organizations will be able to promote their component using the certification logo.

Formal certification will involve a submission, evaluation, and results reporting process.

Developer follows process to request certification

- Conducts self-assessment
- Prepares documentation, sample code, and certification request

Certification authority processes certification request

- Acknowledges receipt of request
- Review submission for completeness and process payment (if necessary)
- Assigns to schedule

Certification authority performs certification testing

- Obtain service code (source, executable, container) and any dependency code to be tested
- Configure according to documentation
- Conducts evaluation testing
- Records test results and produces report

Completion

- Pass – issue certification
- Fail – correspond with submitter with details of failure, wait for resubmission

Submitted code will not be made public or stored beyond the period needed for testing. All test results will be retained.

Technical Requirements

This section describes the technical requirements for self-assessment and certification.

Assumptions

This section states the assumptions and beliefs that were used during development of the technical requirements.

- All certification submissions and work will be conducted in the English language.
- The certification evaluation process does not require submitters to provide source code.
- Services submitted for certification are expected to be packaged in Docker/Docker Compose.
- EdgeX is agnostic with regard to platform hardware and OS, but that platform independence is not tested. Submitters are expected to state exceptions to platform independence.
- Certification is not dependent on the programming language or technology used to develop the component.
- EdgeX is orchestration agnostic. Services should be designed that way and should not make any assumption about orchestration/ QoS
- EdgeX has versions. The testing tools have versions. Certification will be to an EdgeX release. We assume the testing tools are appropriate for the EdgeX release.

Submission Requirements

Prior to submitting a service for certification, submitters ***should*** …

- Perform a self-assessment using the software and instructions provided by EdgeX Foundry.
- Confirm that their service passes all the tests.

When submitting a service for certification, submitters ***must*** provide…

- The service made available in a Docker container that is accessible to the certification authority. The containers must be able to run on a Linux Ubuntu Server or Desktop 18.04 environment.
- An example Docker Compose file using the service under review with all the Docker configuration necessary to replicate the submitters expected runtime network.
- Statement on platform independence and platform requirements (such as minimum memory or other resources). EdgeX assumes platform independence, but does not require independence if stated up front on the limitations and platform requirements.
- Statement on deployment/orchestration dependencies (such as an inability to run outside of Docker). EdgeX is orchestration agnostic and assumes deployment/orchestration independence, but does not require it. Services should be designed in a way that should not make any assumption about orchestration/deployment capability, but where they are not, this should be documented.
- License and usage agreements associated with the service. Certification is available for open source as well as proprietary solutions, but requires the submitter to state the intended license.
- Any licenses necessary to conduct the testing.
- Documentation on self-assessment test results including the version string or fingerprints of the test tools used. If any test failures, provide an explanation on why the failure should be allowed under certification (request an exception).
- Documentation on any API extensions/changes - this document may not be made public but will be needed by the certification authority to understand how to test and provide certification. Services should provide samples of use with documentation where they differ from the reference implementation.

- An application form and legal agreement as required by the EdgeX Certification process.

In order to help in certification, submitters ***may*** provide…

- Guidance on service throughput, performance or scale limitations
- Source code, makefiles, or other development artifacts
- Additional runtime artifacts (e.g., containers that run in alternate platform or environments)
- Alternate configurations

Core Service

Submitters, when providing a core service for certification, ***must*** also provide…

- Information on any operations, configuration, additional functionality, etc. that differ from the reference implementation (example – providing configuration defaults and options that differ from the reference implementation service).
- No documentation is required if the EdgeX reference implementation documentation applies to the service under review.

Device Service

Submitters, when providing a device service for certification, ***must*** also provide…

- The device (hardware, drivers, etc.) or device simulator required to test the device service.
- Indications if the device service was built with (and therefore compliant with) an EdgeX SDK. If not, the submitter must provide an indication of which features of the SDK are provided and not provided by the device service under consideration.
- Documentation explaining how devices are registered and provisioned with an example.
- Example applicable device profiles
- Guidance/expectations on the amounts and rates of data collection typically seen by the device service with regard to its typical devices.

Database Service

Submitters, when submitting a database using service for certification (Core Data, Metadata, Notifications, Logging, Export Client), ***must*** also provide…

- Database initialization code and how is it executed
- Persistence store details (if different from the reference implementation) including installation, cleanup, startup/shutdown procedures, etc.
- Guidance/expectation on data throughput, and scale

Application Service

Submitters, when submitting a user interface or any client user of EdgeX APIs ***must*** also provide…

- Information on the ability to interact with a single or multiple EdgeX instances
- Information on the ability to operate with or without the API Gateway running

Testing Requirements

The certification authority ***will*** …

- Run the submitter's Dockerized service in the blackbox Ubuntu 18.04 Linux/Intel platform test environment using Docker Compose to start the services (both the EdgeX and submitted service) and altering the Docker Compose file for the test environment per the submitter's Compose file.
- Exercise the submitted service using the automated black box tests for that service for the version of certification requested on an Ubuntu 18.04 Linux/Intel platform. Certification requires the submitted service to pass all the black box tests; making a best effort to account for and accommodate any configuration changes or other aspects that would alter the test runs.
- Run the service for a minimum of 24 hours and measure that resource use and system performance are not negatively impacted over time.
- Manually inspect that the service API is a super set of the EdgeX service API. Not every aspect of the service API may be tested in the EdgeX black box tests.
- Manually inspect the service complies with EdgeX requirements of
    - Providing appropriate logging and offering log statements that comply with EdgeX conventions
    - Use and pass the correlation ID when calling on other services to aid in troubleshooting
    - Getting configuration from the registry service or local file system when not available (on bootstrapping)
    - Retry to connect to dependent services when they are not available
- Confirm compliance with system management requirements. For example, checking that the service's metrics requests are responding as expected and not just returning hardcoded values. [do we need a more complete list here]
- Exercise the security black box API tests to ensure that the service complies with the security service needs.
- Manually inspect the service's compliance with security requirements – for example that the service can be configured to operate behind the API gateway and that no application secrets are stored in the clear. [do we need a more complete list here]
- Collect and review test results – allowing for black box test failures when the black box exceptions are requested and deemed satisfactory.
- Manually inspect and test that the services properly address exception-path use cases (for example, the service continues to try to reconnect with a dependent service when it is not available, or reconnects to a database when one is required and it drops). [do we need a more complete list here]
- Manually review documentation provided by the submitter for completeness, accuracy, language issues, etc.
- Manually check that configuration of the service is in compliance with EdgeX configuration seeding.
- Manually check that database population and use with regard to schemas, seeding, and cleanup is handled in compliance with EdgeX standards.
- Check the performance metrics (CPU, memory, response time, etc.) of the other services for any negative impact. Check that the service does not block or reduce performance of other services. Services causing negative impacts will not be certified.

- Produce a report on the performance of the service. This report may contain information on resource (CPU, memory, storage, etc.) utilization, start time, response and latency times, or any other characteristic that may be of interest to a user. The certification authority will not decline certification on the basis of performance results but will publish the results for user awareness.

The certification authority ***may***…

- Explore and assess any code, configuration, or other artifacts for
  - programming best practices, standardization, language idioms, etc.
  - quality and depth of comments.
  - compliance with copyright and license agreement indications.

Certification Process

When an organization is ready to submit their service or product for certification, they will follow a published process available from the EdgeX web site. The beginning of this process is an application.

The following items are required as part of a request for certification. Submission checklist:

- Legal agreement confirming acceptance of the terms of certification
- Results of self-assessment tests
- The component(s) to be certified
- Supporting configuration files and infrastructure that may be needed to setup and conduct the evaluation
- Documentation for the component
- A technical contact that can assist in the setup and answer questions during the certification testing.

The certification authority will receive the application and code, confirm it is complete, and schedule the evaluation testing.

The intention is that testing will be conducted without involvement of the submitter. I.e., the documentation and information provided with the submission will be sufficient. The submitter may be contacted if there are questions or issues detected during the setup or testing.

Certification testing (will be service dependent)

- APIs
  - Dependency checking
  - Correlation Id
  - Logging standards
  - RESTful APIs
  - Security APIs (or ability to run under Security services)
  - System Management APIs
  - Error handling??
- Security
- Quality checks (what exactly are we looking for in terms of qualities… like documentation, etc.)
- Measure of specific performance criteria

If issues are found, the submitter will be given the opportunity to resolve them within 15 business days.

Upon successful completion of the tests...

- The Certification WG and TSC will be notified
- Once approved, the submitter will be notified
- Announcement of the certification will be coordinated with the submitter
  - Press announcement
  - Appearance on EdgeX website

Road Map

As can be seen from the list of support tools required to support certification presented above, there is a lot of work required to get to the point of issuing meaningful certification of EdgeX components. We want to balance market value, time to market, and effort.

Decision 1 - Where to start?

Device services offer the most commercial value. Core services have the most tests available that could be used to more quickly establish and prove out the process. Application services are currently in development and may have the most resources available to provide necessary test artifacts. The Certification WG recommended Device Services as having the greatest short term commercial value.

It was decided that the EdgeX community focus on certification of Device Servers as its first priority.

Decision 2 - Self-Assessment or Certification?

There was some debate over the best way to start. Advocates of certification liked the formality and gravitas it would provide to commercial offerings. Advocates of self-assessment pointed out the resource challenges and the extra burden of establishing a certification authority, handling submitter code, setting up and executing tests, etc. There was also a concern that only offering certification may be a burden that would reduce participation by developers in creating EdgeX components.

The decision was made to start with self-assessment. This will enable the EdgeX community to achieve its goals for interoperability of components, while avoiding the burden of creating formal processes until the community and market demand has been established.

Given the complexity and effort to create a meaningful certification program, we are suggesting a phased approach in the crawl, walk, run model. Release targets for the phases will be defined in collaboration with the TSC and other WG.

Crawl - "Assess and Learn"

The objective of the Crawl phase is to establish a self-assessment package for Device Services and make it available by Q4 2019. See Appendix 1 for examples of the types of things that should be evaluated.

This will require the following deliverables from the EdgeX community:

Technical

- General test plan for Device Services that covers APIs, SMA support, operation, and basic performance metrics.
- Black box tests that can be run against any device service and produce results based on the test plan
- EdgeX base configuration to use as a “test harness”
- Instructions for how a user can deploy and execute the tests, and what metrics to record to validate the tests were performed.

Commercial

- Program promotion (“EdgeX Ready”?)
- Process for organizations to submit their test results for review and, if approved, inclusion on the EdgeX web site and other promotion.

Success will be having a self-assessment package available and being used by developers to confirm readiness of their components. The learning from this phase will be used to inform the development of the formal certification process.

Walk - "Certification"

The objective of the Walk phase is to be able to evaluate and declare an EdgeX Device Service component as being certified. The focus is on making sure the process and tests exist and are sufficient, even if highly manual for execution.

MVP for certification will include:

- Submission process

- Evaluation process (manual)
    - Infrastructure
    - Black box tests
    - Exercise the service against appropriate APIs
- Results reporting process
- Promotion of components that have achieved certification

It is likely that the initial certification candidates will be done using a very manual process. There will be gaps discovered, lessons learned, and need for more documentation. Once the process and tools are proven, we can plan future improvements.

Success criteria:

- A minimum of 5 EdgeX device services are certified

Jog - Automate and Formalize

With a working process in hand, the priority of the Walk phase will be to make setting up tests and recording results more automated. A second goal of this phase is to make a decision on the what resources will fulfill the certification authority role.

Success criteria:

- Ability to execute certification tests of device services and gather results in a repeatable manner by someone who may not be an EdgeX expert or developer
- Identify the requirements and selection process for a third party to act as certification authority

Run - Deeper and Wider

With a working and automated process, the Run phase will focus on expanding tests to cover the remaining service types.

Success criteria:

- Certification of all EdgeX service types (Device, Application, Core)
- Establishment of a certification authority

Commercial Issues

This section covers various commercial issues that will be involved in the operation of the certification program.

Eligibility

Formal certification is available to any organization or developer that is willing and able to submit the required information and successfully complete the testing. There is no requirement for a submitter to be a member of EdgeX Foundry or Linux Foundation. This is to encourage maximum participation in the certification program.

Fees

Certification is a process that will take some effort by both the submitter and the certification authority. Organizations that will most benefit from certification will be those that intend to make their components available in the market. These organizations should be expected to help cover the cost of conducting the certification program.

Certification is not intended as a “profit” generating revenue stream. Any fees are intended solely to cover the cost of administration.

During early phases of the certification program, while testing is being conducted by the community, there should be a period where the fees are waived. This will encourage developers to submit their components for certification and allow EdgeX Foundry to learn and improve the process. Regardless of fees or certification authority, any component that is certified will have met the appropriate technical requirements. Once self-assessment is available, and/or if a third party certification authority is involved, the fee waiver will end.

The fee schedule is to be determined once the typical level of effort has been identified for each type of service.

If the effort needed to retest a component is found to be low, there may the opportunity to have a reduced fee for recertifying a component on a new EdgeX release. This is to be determined once we have more experience with the testing process and potential costs of the certification authority.

Q: Should there be a reduced fee for developers or organizations that are members of EdgeX Foundry?

Version Control

The certification process will be executed against the current release of EdgeX. The certification results and certificate will specify the version of EdgeX that was used for testing. There is no expiration of certification provided: 1) the certified component has not been modified, and 2) the component will be used with that release of EdgeX.

EdgeX Changes

EdgeX will continue to evolve with regular releases. Any list of EdgeX certified services will include the version information of the EdgeX release that was used for the certification. It will be up to users to evaluate if the changes to associated APIs are significant enough to invalidate the certification for the EdgeX release they plan to use. It will be up to the developer of a component to decide if they want to certify their component on a new EdgeX release.

Component Changes

It is expected that certified components will change as developers fix bugs and add new functionality. We want to encourage a process of continuous improvement. However, it is easy to imagine scenarios where the changes could cause a component to no longer meet the certification tests.

In these cases, the developer should continue to make the version of the certified component available. That assures users they have access to what was evaluated and certified. At the same time, the developer may make the revised version of their component available. This new component will no longer be considered as certified.

A process for recertifying component changes will be needed. If the changes are for bug fixes, it could be that self-assessment and request for review is sufficient to recertify the component. For enhancements, it may be that a full certification process will be required. This will be determined as we have a better understanding of the certification process as well as the volume and extent of changes to certified components.

Revoking Certification

There may be situations where EdgeX Foundry determines it is necessary to revoke certification of a component. This could be due to

- Errors discovered in certification testing
- Component changed after certification without informing users
- Complaints about component functionality or quality from the EdgeX user community
- Improper use or claims of EdgeX certification

Any decision to revoke certification will be made by the EdgeX TSC. The decision will be made following a report of the reasons prepared by the certification WG and a response from the component developer/organization.

We do not expect this to happen often, but it is necessary to protect the integrity of the certification program.

Open Issues

This section contains questions that are still open and under discussion by the Certification WG.

Q: What does the self-assessment package look like (software + docs)? Where does someone get it?

Q: Should evaluation of documentation evaluation be a requirement for certification. For example, would the documentation have to be to a certain standard, who would determine this or simply just a tick box on a list of certification requirements to indicate that there is some documentation regardless of quality/content.

Q: Should certification be conducted by the EdgeX Community or by a third party?

- Fee based? Fees require a certain level of quality of service that isn't going to come from a volunteer organization. If we collect a fee, it'd almost have to drive a 3rd party to run it.
- Would the 3rd party be paid by the community or volunteer? Without funding approved by the LF Edge community, this is a challenge.
- Selection of third party?
    - DIY to start and then find an org to do it?

Q: Do we need some sort of repo for the requester to submit their code?

Q: What is our liability for protecting submitted code?

Q: Do we charge to certify clients/users of EdgeX. Could be good revenue, but does that make sense and could we practically do it?

Q: How should we define the fee schedule?

Q: Any types of organizations that should have reduced or no fees?

Glossary

Validation: The assurance that a product, service, or system meets the needs of the customer and other identified stakeholders. *Is the project designed to meet the desired objectives/success criteria?*

Verification: Evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. *Is the software in compliance with a defined set of requirements?*

Certification: Verification based on a pre-defined scope by a certification body.

Certification Authority: The organization that manages and executes the certification process on behalf of EdgeX Foundry.

self-assessment: A process that enables users to do their own informal testing prior to submission for certification.

Conformance Properties

- *Consistency:* do the different (parts of) software artifacts conform to each other?
- *Functional:* does input to the system produce the expected output?
- *Behavioral:* does the system meet general safety and progress properties like absence of deadlocks or are constraints on the specific states of the system met?
- *Quality:* do the artifacts fulfill nonfunctional requirements in the areas of for example performance, security, and usability?
- *Compliance:* do the artifacts conform to standards, guidelines, or legislation?

May, Should, Must

RFC 2119 "Key words for use in RFCs to Indicate Requirement Levels" goes into specifics of what different words on requirements mean.

- MUST is equivalent to REQUIRED and SHALL indicating that the definition is an absolute requirement.
- MUST NOT is equivalent to SHALL NOT and indicates that it is an absolute prohibition of the specs.
- SHOULD is equivalent to RECOMMENDED means that there are valid reasons to ignore a particular requirement, but the implications need to be weighed.
- SHOULD NOT and NOT RECOMMENDED means that a particular behavior may be acceptable or useful, but again, the implications need to be weighed.
- MAY means OPTIONAL and that the requirement is *truly* optional. Interoperability with different systems that may or may not implement an optional requirement must be done.

Appendix 1. Device Service Certification Notes

This appendix is provided to give some insight into the level of detail that may be required for conducting certification of each service type.

Questions:

- Is this for new or existing (replacement) device service? Is it commercial vs open source?
    - We would assume the process remains the same but there may be some points of departure. See below.
- Are the tests only testing what is common to the DS SDK? Are there other things that are to be tested – like how data is deposited to Core Data?
    - We assume that it does include how it impacts and behaves with other services, but this may come in phases (crawl, walk, run).
- Do we need to modify the current DS SDK black box tests to be more parameter driven – especially with regard to Command services?
    - Yes, again, over time.
- How do we test for and explore “unusual behavior” – like how a DS might send data to Export or something like Security that we did not expect (or maybe want)?
    - We won’t be able to do much at first (other than manual inspection) but over time this is something we’ll want to add more tests for.

Assumptions:

- It is assumed that the device service tests would be the same for all device services. Tests would not be unique to a device service.
- We are certifying only to a specific platform at first. (Ubuntu 18.04)
- Guidelines on documentation review that is subjective need to be written to make this more comfortable to submitters; making it less subjective.

Process:

1. Self-assessment is done by the submitter.
2. Device Service submitter sends the Certification WG the following:
    a. Self-assessment results/report
    b. Device service (containerized),
    c. Device (or simulator) used to test the device service
    d. Device profile(s) – and any documentation necessary to explain the profile(s)
    e. Documentation as per below.
        i. Specifically, the protocol details should be included in the documentation – for example in a Modbus device service, the submitter would indicate whether using Modbus RTU vs TCP/IP
    f. Guidance on how much data is going to be captured by default
3. Perform black box tests that we already have (API check)
    a. DS SDK black box/API tests
    b. Device Service specific tests (for device services that we already have – if any are written, which to date they are not)
4. Inspect or test that the service behaves well with regard to rest of EdgeX (some of this could be done via running existing other black box tests; most will have to be done by manual inspection today)
    a. Appropriately populates Core Data with readings/events
    b. Appropriately populates Value Descriptors
    c. Appropriately populates metadata (Devices, Device Services, Profiles,...)
    d. Appropriately receives and interacts with Command service (responding as expected)
    e. Appropriately gets its configuration through the config/registry service
    f. Uses the Logging micro service to log errors, warnings, information and debug statements.
5. Perform "stability" exercise and report results. Leave the device service running for a certain extended period of time (time period to be determine) and examine:
    a. How much event/reading data did it report (did it seem to match expectations provided by the submitter?)
    b. How long does the service stay up – did it run out of resources or otherwise crash?
    c. Did it impact other services negatively (like causing any of them to fail due to data volume or size issues)?
6. Performance evaluation and metric collection
    a. Report on CPU, Memory and other resource usage
    b. Report on time to come up
    c. Report on storage usage
7. Documentation review (subjective review)
    a. Information on any operations, configuration, additional functionality, etc. that differ from the reference implementation (example – providing configuration defaults and options that differ from the reference implementation service).
8. After each phase of above, any failure, test observation or concern can be addressed with the submitter – allowing the submitter to address issues and re-iterate the entire process or a section as required
    a. Are the failures in line with submitters self-assessment documentation and exceptions noted? Are these reasonable and does EdgeX need to revamp tests to accommodate or at least grant an exception on the basis of what is discussed.
    b. Cert WG Privately reviews and makes a determination of the results and failures – allowing the submitter reiterate through the process

Appendix 2. Needed Tools

Execution of the self-assessment and certification process will require some tools and infrastructure to be assembled by various EdgeX working groups.

Product and Development

The following product and infrastructure items that will be needed:

Deliverable	Lead
Test plan for each type of service	QA
Black box tests for each type of service based on the test plan	QA + Dev
Parameterize black box tests with the git location and access credentials so the certification authority can use/automate existing tests and point them to location that may not be under our control (submitter provided repository)	Dev
Performance/longevity tests for 24 hour exercise of service	Dev
Test of the certification tests. • Confirm black box tests are valid for certification. I.e., the cert tests may have slightly different needs than dev tests. • Confirm that certified services are truly interoperable (important for credibility of cert program)	QA
Process and instructions for executing tests • For self-assessment users • For certification testers	QA + Certification

Independent infrastructure that can be used by the certification testing team ◦ Infrastructure only accessible by certification testers ◦ Each test would be scripted to initialize from scratch ◦ Store results of testing ◦ Environment to be protected during period of performance/longevity testing (~24-36 hours) ◦ May need multiple instances running at the same time	DevOps
Repository only accessible to certification testing team to store raw test results and results report (for use in the event of later question)	DevOps
Location for public to obtain self-assessment tests and documentation	DevOps
Clean up process following completion of certification process • Delete submitter code/artifacts • Store results	DevOps + Certification

Other notes:

- Consider using existing device services to confirm process and that they can be certified (test our own dog food)
- Assumption that some submitters will want to provide their code from a private repository. Need to investigate how to script pulling the code (private certs?) and running tests.

Process

The following items that will be needed to execute the certification process:

Deliverable	**Responsible**
Application form to gather details about the requester and code being submitted	Certification
Legal agreement to confirm rules for code handling by certification authority and rights and responsibilities for use of Certification by requestor	Certification + LF
Code submission process • Specification of format/content/code • Intake process (controlled access)	Certification
Record and store testing results/observations	Certification + DevOps
Report of failed test with information on failure and process for remediation	Certification
Create formal certification test results document	Certification
Issue formal letter of certification or failure to submitter	Certification
Periodic reports of certification learnings (anonymized) to the community	Certification

Other notes:

- The certification testing process and results are private between submitter and EdgeX Foundry until the process is completed.

Marketing and Promotion

In order to achieve the full benefits of the certification program it must be seen as having value and momentum. The following items will be helpful in that effort:

Deliverable	**Responsible**
Web page(s) about certification • Information about certification program and how to apply • Listing of certified EdgeX components	Marketing
Coordinate with submitter on when they want to make public announcement	Marketing
Joint press release template that can be used to announce new certified component	Marketing

Periodic follow up with users of certified components to confirm value and gather quotes that could be used for additional promotion	Marketing

External Certification Authority

It may be necessary for the EdgeX working groups to be very hands on during the initial certification testing in order to quickly respond to any issues found in the process or tools. However, it may not be the best use of community members' time to execute the certification process once the process has been established and the volume of requests begins to grow. This is especially true if a decision is made to charge fees for certification or there is a need to have independent testing for requesters submitting sensitive code.

We believe it will be necessary for a third party to be contracted to manage the certification process. The third party, operating under the review of the TSC and the Certification WG, will take on administration and execution of the certification processing and testing. In addition to reducing the load on the community, it will also provide a repeatable and independent process that may be more respected by the marketplace.

If a third party is to be used, the Certification WG will create a document to guide the evaluation and selection process. This will include:

- Duties and requirements
- Selection criteria and evaluation process
- Cost and budget (handling payments)
- Service quality expectations