EDGEXFOUNDRY™

EdgeX Foundry Performance Report

Jakarta Release

November 2021

Jakarta (version 2.1) marks the 9th community release of EdgeX and was formally released in November 2021. The key new features added in Ireland are described on the EdgeX Community Wiki here

This report aims to provide EdgeX users with important performance information that can guide solution development and deployment strategies. The information also helps the EdgeX development community to ensure the platform remains suitable for lightweight edge deployments and can help to identify future performance targets.

The performance metrics described in this report relate to data obtained on the following hardware platform:

HP MP9 G4 Desktop Mini PC

- Intel Core i7-8700T processor @2.4GHz
- 16GB RAM
- Ubuntu 20.04 LTS

www.edgexfoundry.org

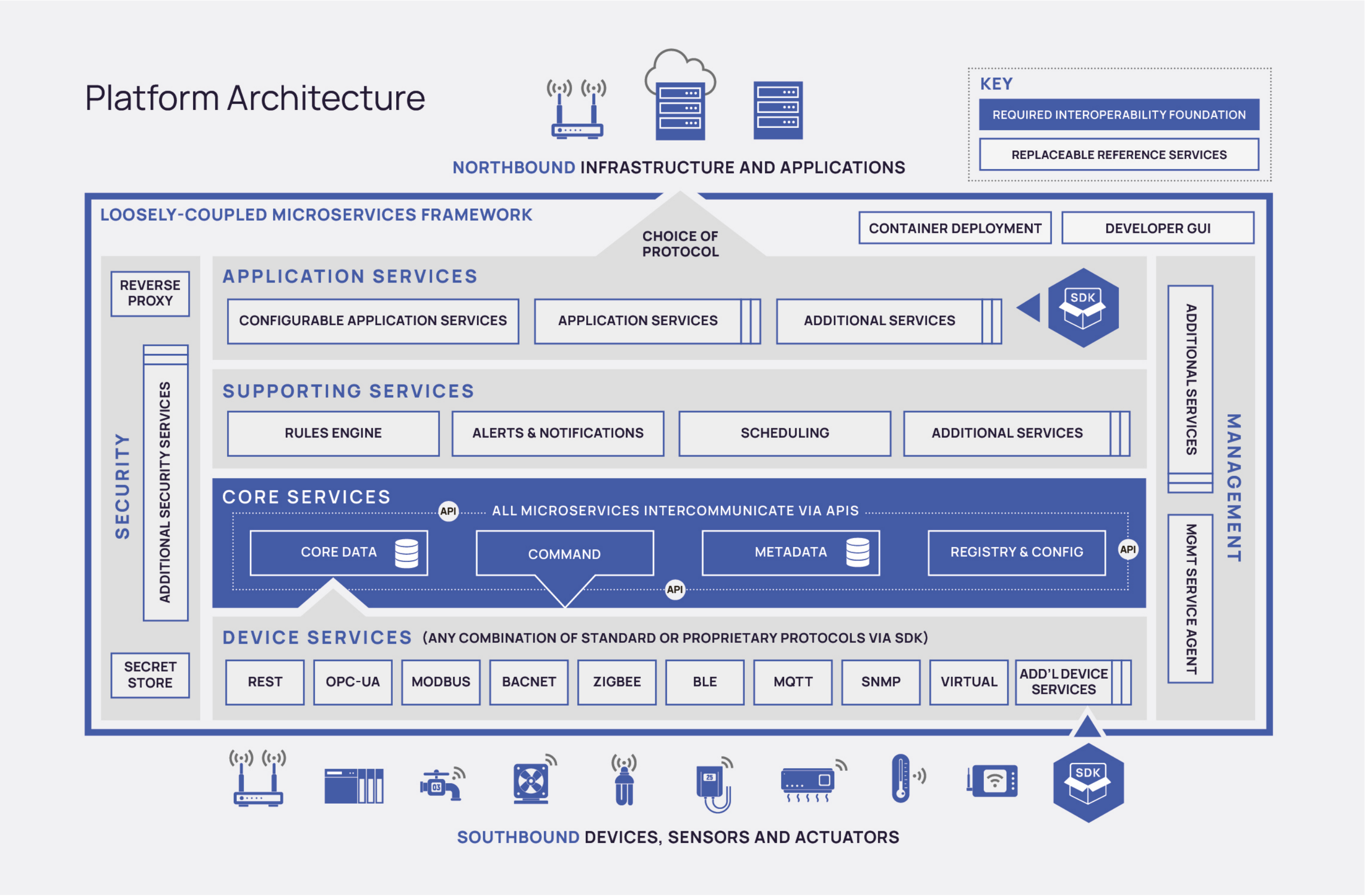
LF EDGE
THE LINUX FOUNDATION

EdgeX Foundry Deployment Options

The EdgeX Foundry platform is a collection of modular microservices that each perform a specific role at the IoT edge. While some EdgeX microservices are used in almost all deployments of the platform (e.g. the Core Services), the specific set of microservices needed will depend on the exact requirements of each use case.

Since EdgeX consists of a set of required and optional microservices, this report provides performance data and resource usage relating to two EdgeX deployment classifications:

- A **typical** EdgeX deployment provides all of the microservices as envisioned by the original architects of the system. This includes edge microservices for ingesting data from different edge protocols, decision making, notifications and alerts, device actuation and streaming to a cloud or IT endpoint. A typical EdgeX deployment also includes the full EdgeX security services
- A **minimal** EdgeX deployment provides only the microservices needed to ingest data from a single edge protocol and stream that data northbound to a cloud or IT endpoint. A minimal deployment may implement alternative security approaches to the EdgeX security services that are provided as standard.



The EdgeX Microservices

Microservice	EdgeX Deployment Name	Description
CORE SERVICES		
Core Data	edgex-core-data	Optional data store for readings collected by devices and sensors
Core Metadata	edgex-core-metadata	Used by other services for knowledge about the devices and how to communicate with them
Core Command	edgex-core-command	Optional invocation of devices on behalf of other services, applications or external systems
Database	edgex-redis	The default database, implemented via Redis
Configuration & Registry	edgex-core-consul	Optional centralized service configuration, implemented via Consul
DEVICE SERVICES		
REST Device Service	edgex-device-rest	For interacting with edge devices that provide a REST-based API
Virtual Device Service	edgex-device-virtual	For simulating device data
Note that many other Device Services are available both in EdgeX and provided by commercial vendors		
SUPPORTING SERVICES		
Rules Engine	edgex-kuiper	Reference rules engine implemented by EMQ X Kuiper
Support Notifications	edgex-support-notifications	Delivers notifications to inform of important system events
Support Scheduler	edgex-support-scheduler	Executes operations on a configured interval or schedule
APPLICATION SERVICES		
App Service Configurable	edgex-app-rules-engine	Application Service that can be configured to execute built-in transform and export functions
SECURITY SERVICES		
API Gateway	edgex-kong	Provides a single point of authorized entry for all EdgeX REST traffic
API Gateway Database	edgex-kong-db	Database required to work with the API Gateway
API Gateway Setup	edgex-security-proxy-setup	Service required to configure the API Gateway
Secret Store	edgex-vault	Central repository to securely store EdgeX tokens, passwords and certificates etc
Secret Store Setup	edgex-security-secretstore-setup	Service required to configure the Secret Store
Security Bootstrapper	edgex-security-bootstrapper	Provides secure activation of the EdgeX security services

EdgeX Footprint and Container Sizes

Each EdgeX microservice is typically implemented in either Go or C and then compiled into an executable which has a size or footprint as it sits on disk. For convenient deployment and orchestration, each microservice executable can also be built into a container image so it can be run as a standalone service with less dependencies on the host. This greatly improves portability and platform independence. EdgeX provides container images for both Docker and Ubuntu Snaps.

Both containerized Docker image footprint data and non-containerized executable footprint data is measured and shown below.

EdgeX Microservice	Image Footprint (MB)	Executable Footprint (MB)
edgex-core-data	20.67	12.31
edgex-core-metadata*	16.87	11.18
edgex-core-command	16.13	10.43
edgex-redis*	32.4	N/A
edgex-core-consul	116.43	N/A
edgex-device-virtual	24.76	16.37
edgex-device-rest*	21.45	13.07
edgex-support-notifications	16.96	10.78
edgex-support-scheduler	16.21	10.52
edgex-kuiper	24.54	N/A
edgex-app-rules-engine*	25.23	16.34
edgex-kong	141.61	N/A
edgex-kong-db	191.51	N/A
edgex-security-proxy-setup	26.86	N/A
edgex-vault	186.3	N/A
edgex-security-secretstore-setup	28.36	N/A
edgex-security-bootsrapper	18.87	N/A
TOTAL	**925.16**	**N/A**

Typical Deployment

The footprint for a typical EdgeX deployment using Docker containers is approximately 925 MB. This includes the configuration service, the command service, two device services, all supporting services and all security services.

Minimal Deployment

The footprint for a minimal EdgeX deployment using Docker containers is approximately 96 MB. These services are marked in the table opposite with an asterix (*).

***Note**: The executable footprint is not recorded for third-party open source services (Redis, Consul, Kuiper, etc) so N/A is listed for those services.*

EdgeX CPU Consumption

Each EdgeX microservice has its CPU consumption measured as it is started as a Docker container. The CPU usage is reported by the Docker engine and is measured as a percentage of the available CPU on the machine. In general, the measure of usage at startup is a good indication on the upper bound for many of the services. Note that the characteristics of different chip architectures may affect the CPU utilization.

EdgeX Microservice	Maximum (%)	Minimum (%)	Average (%)
edgex-core-data	0.03	0.02	0.02
edgex-core-metadata*	0.04	0.02	0.03
edgex-core-command	0.16	0.02	0.09
edgex-redis*	0.21	0.18	0.19
edgex-core-consul	1.36	0.30	0.60
edgex-device-virtual	0.06	0.00	0.04
edgex-device-rest*	0.14	0.00	0.01
edgex-support-notifications	0.17	0.01	0.08
edgex-support-scheduler	1.40	0.06	0.44
edgex-kuiper	0.03	0.00	0.01
edgex-app-rules-engine *	2.83	0.00	0.39
edgex-kong	0.16	0.01	0.05
edgex-kong-db	1.36	0.0	0.25
edgex-security-proxy-setup	0.00	0.00	0.00
edgex-kong-vault	5.42	0.63	1.20
edgex-security-secretstore-setup	0.00	0.00	0.00
edgex-security-bootstrapper	0.00	0.00	0.00
TOTAL	**13.37**	**1.25**	**3.40**

Typical Deployment

Maximal CPU usage recorded for a typical EdgeX deployment using Docker containers on the test hardware is approximately 13%. This includes the configuration service, the command service, two device services, all supporting services and all security services.

Minimal Deployment

Maximal CPU usage recorded for a minimal EdgeX deployment using Docker containers on the test hardware is approximately 0.81%. These services are marked in the table opposite with an asterix (*).

EdgeX Memory Consumption

Each EdgeX microservice has its memory consumption measured as it is started as a Docker container. The memory consumption is reported by the Docker engine and is measured in Megabytes (MB). In general, the measure of usage at startup is a good indication on the upper bound for many of the services.

EdgeX Microservice	Maximum (MB)	Minimum (MB)	Average (MB)
edgex-core-data	9.27	7.97	8.75
edgex-core-metadata*	9.53	8.62	8.84
edgex-core-command	7.91	7.15	7.65
edgex-redis*	3.10	2.63	2.90
edgex-core-consul	39.45	33.32	36.16
edgex-device-virtual	10.65	9.69	10.14
edgex-device-rest*	8.66	8.20	8.42
edgex-support-notifications	8.14	7.77	7.97
edgex-support-scheduler	8.70	7.64	8.04
edgex-kuiper	37.30	36.68	37.03
edgex-app-rules-engine*	9.14	8.27	8.81
edgex-kong	118.04	107.35	113.98
edgex-kong-db	18.87	18.54	18.68
edgex-security-proxy-setup	0.00	0.00	0.00
edgex-vault	121.77	121.64	121.69
edgex-security-secretstore-setup	4.78	4.78	4.78
edgex-security-bootstrapper	6.42	6.03	6.28
TOTAL	**422.00**	**396.00**	**410.00**

Typical Deployment

Maximal memory consumption recorded for a typical EdgeX deployment using Docker containers on the test hardware is approximately 422 MB. This includes the configuration service, the command service, two device services, all supporting services and all security services.

Minimal Deployment

Maximal memory consumption recorded for a minimal EdgeX deployment using Docker containers on the test hardware is approximately 30 MB. These services are marked in the table opposite with an asterix (*).

EdgeX Startup Times

The startup times are measured for each of the microservices developed by the EdgeX Foundry community, both with and without the security services enabled.

Startup times include any overhead associated with creating the Docker containers in which the microservices run. Note that the metrics are obtained while starting all microservices at the same time so any dependencies between the services starting are included in the data. The total time recorded is the time taken for all of the services to be started, rather than a sum of all individual startup times.

EdgeX Microservice	Average with Security (s)	Average without Security (s)
edgex-core-data	17.89	10.11
edgex-core-metadata*	17.79	10.11
edgex-core-command	17.79	10.02
edgex-device-virtual	25.53	15.81
edgex-device-rest*	24.62	15.85
edgex-support-notifications	17.79	10.16
edgex-support-scheduler	17.83	10.35
edgex-app-rules-engine*	18.49	10.39
TOTAL	**24.76**	**15.87**

Typical Deployment

The average startup time recorded for a typical EdgeX deployment using Docker containers on the test hardware is approximately 25 seconds This includes the configuration service, the command service, two device services, all supporting services and all security services.

Minimal Deployment

The average startup time recorded for a minimal EdgeX deployment using Docker containers on the test hardware is approximately 16 seconds. These services are marked in the table opposite with an asterix (*).

Note also that services can of course be started and stopped as EdgeX runs. In this case it is not necessary to create a new Docker container for each service. Restarting an already-created Docker container can reduce the startup overhead.

EdgeX Operational Latency

Ping Response Times

The ping response times are measured for each of the microservices developed by the EdgeX Foundry community and represent the reactivity of each service when it receives an HTTP/REST request.

EdgeX Microservice	Average with Security (ms)	Average without Security (ms)
edgex-core-data	8.82	1.01
edgex-core-metadata	8.54	1.03
edgex-core-command	8.74	1.01
edgex-device-virtual	8.56	1.00
edgex-device-rest	8.46	0.99
edgex-support-notifications	8.55	1.03
edgex-support-scheduler	8.57	1.05
edgex-app-rules-engine	8.42	1.03

Ping response times recorded on the test hardware are consistently around 8-9ms when the API Gateway is included. Without the API Gateway, response times are consistently around 1ms.

Data Export Latency

Data export latency represents the time it takes to collect EdgeX data from the "southside" devices & sensors and deliver it through the platform to the "northside" applications.

The latency reported here relates to the Virtual Device Service, which simulates 3 different data attributes. Each Device Service type (Modbus, BACnet, etc), will obviously have its own data rates associated with collecting the data from the physical edge devices or sensors, but the Virtual Device provides useful indicative performance data. It is planned to provide performance relating to more Device Service types in future versions.

Virtual Device Name	Average with Security (ms)	Average without Security (ms)
Random-Integer-Device	2.3	2.5
Random-UnsignedInteger-Device	2.0	3.1
Random-Boolean-Device	3.3	2.9

The Virtual Device data is delivered through the EdgeX platform on the test hardware at an average latency of approximately 2-3 ms. There is no discernable overhead when security is enabled.

EDGEXFOUNDRY™

Test Methodology

All EdgeX microservices were run with their default configurations. In order to provide accurate and reliable statistics the tests were ran as follows:

- The CPU and memory consumption metrics were gathered by obtaining the data 10 times at an interval of 7 seconds between each measurement
- The startup time data was obtained by starting the EdgeX microservices 5 different times
- The ping response test was performed 100 times for each EdgeX microservice
- The data export latency metric was performed 10 times for each of the Virtual Device attributes

The full raw performance metrics, as well as data for other hardware platforms, is provided on the **EdgeX Community Wiki here**

Have you got any questions?

This document was produced by IOTech Systems. IOTech is the Edge Software Company with products that include Edge Xpert which is a value-add and commercially-supported implementation of EdgeX Foundry. IOTech also provides lightweight and real-time extensions to EdgeX and edge management orchestration software.

If you have questions about this report or if you need more information on IOTech or Edge Xpert, email **info@iotechsys.com** or visit the website **www.iotechsys.com**

IOTech

Contact Us

For general information about EdgeX Foundry, or membership inquiries, please email **info@lfedge.org**

Visit our website at **www.edgexfoundry.org**

LF EDGE
THE LINUX FOUNDATION