

Providing a Hackathon Edge Platform for Application Developers

Intel, Cisco, Equinix, and LINKS Foundation collaborate on hardware and software Multi-access Edge Computing (MEC) platform for MEC hackathons that enables application developers to learn and demonstrate their technologies.

Table of Contents

Executive Summary	1
Introduction to MEC and the Benefits of Hackathons	1
Goals and Design of a MEC Hackathon Platform	2
F2F Hackathon Configuration with On-premises Servers	2
Remote Hackathon Configuration Using Bare-Metal PaaS	4
Intel® Xeon® Scalable Processors .	5
Software Platform for Edge Developers.....	5
OpenNESS	5
Intel® Distribution of OpenVINO™ Toolkit.....	6
MEC Location API.....	6
Consumer Registration	7
Location API Discovering and Registration.....	7
How to Use Location API – Simulator	7
How to Use Location API – Practical Use	7
Detailed Description of the MEC Hackathon Server Solution..	8
Setup for F2F Participation.....	8
Setup for Virtual Participation .	8
Conclusion.....	10
Authors.....	11
Appendix 1: Abbreviations	11

Executive Summary

Multi-access Edge Computing (MEC) offers application developers and content providers cloud-computing capabilities and an IT service environment at the edge of the network. From a market adoption perspective, this new technology needs not only the presence of adequate edge infrastructure but also active engagement of software developers that create suitable applications utilizing this technology. Thus, an ecosystem of new edge applications is needed to offer high-demand services that attract paying customers, thus providing financial returns for communications service providers. One way to build this ecosystem of software applications, and related edge developers' communities, is by giving them access to technology and an opportunity to showcase their applications through hackathon competition events. Staged virtually or physically at major conferences, hackathons offer developers an opportunity to innovate and understand how edge applications can be designed and implemented, based on open source solutions and consolidated international standards.

This white paper presents a set of complete MEC solutions that are designed for both virtual and face-to-face (F2F) hackathons. These systems converge typical edge workloads, including information technology (IT), operational technology (OT), and connectivity technology (CT)/networking. The described solutions were utilized for the Droidcon MEC Hackathon 2020, which took place in November 2020 in Turin, Italy.

The standards-based MEC solutions presented for use at hackathons include Intel® architecture servers from Cisco Systems as well as “bare-metal as a service” solutions and interconnect services from Equinix. The software stack is based on Open Network Edge Services Software (OpenNESS) and also includes Intel software technologies including Intel® oneAPI toolkits and the OpenVINO™ toolkit. This environment enables infrastructure and application developers to expose the virtualization, acceleration, and MEC-oriented features in the server. Furthermore, the combination of this comprehensive reference platform with the implementation of an ETSI-compliant MEC location API from LINKS Foundation represents an attractive package for developers because it is a complete set of tools for edge application development.

The paper describes the solution developed for that event and how it can be used in the future in a repeatable and scalable way for both virtual and in-person events with large numbers of software developers.

Introduction to MEC and the Benefits of Hackathons

The Multi-access Edge Computing (MEC) initiative is an Industry Specification Group (ISG) within the European Telecommunications Standards Institute (ETSI), created with the purpose of defining an open standard for edge computing. MEC servers are

hosted in micro-data centers located in base stations, points of presence (PoPs), and other edge locations. These micro-data centers store, analyze, and process data with lower latency than is possible using a connection to a data center.

MEC servers are a key platform for 5G applications, providing ultra-low latency for applications including radio access networks (RAN), content delivery networks (CDNs), vehicle-to-everything (V2X), smart city applications, and more. Overall, MEC servers improve efficiency and service quality, and they generate actionable insights faster.

Based on industry standards groups like 3GPP and ETSI,¹ MEC offers cloud-computing capabilities and an IT service environment at the edge of the network that is beneficial to both application developers and content providers.

One key priority for the market success of edge computing is the satisfaction of the needs of two stakeholder categories in the MEC ecosystem:² infrastructure owners (e.g., CoSPs and cloud providers) and software developers (e.g., application/content providers, innovators, and startups). The challenge of all new infrastructure technologies is to have enough applications available to offer high-demand services that can provide significant financial return for the investment.

MEC Hackathons are thus very suitable tools to attract the ecosystem of edge software application developers through the organization of competition events that offer developers an opportunity to innovate and understand how edge applications can be designed and implemented.

Goals and Design of a MEC Hackathon Platform

The idea of a hackathon system reference design was conceived for the [Droidcon MEC Hackathon 2020](#), and was organized by Intel in collaboration with TIM, Cisco, LINKS Foundation, and Equinix. This hackathon proposal was endorsed by ETSI and supported by GSMA and the City of Turin.

At that Droidcon MEC Hackathon, the focus of the challenge was to design edge computing applications for 5G and Android-based devices. Some of the categories of applications specified for the competition were automotive, factories of the future, drones, and consumer, media, and entertainment. In addition, as an optional challenge, developers were invited to utilize additional tools that could help them to develop applications for any of the specified use cases/blueprints. These tools included the following:

- [Data Parallel C++ or DPC++](#) cross-architecture language from the [oneAPI initiative](#) to create custom accelerators on the FPGA for a kernel algorithm.
- [Intel® Distribution of OpenVINO toolkit](#) to accelerate deep learning inference tasks for computer vision applications. The OpenVINO toolkit includes various neural network topologies and enables deep learning inference acceleration.

The MEC Hackathon platform was also designed to be utilized potentially at future events across the globe; thus, it was built with repeatability and scalability as main design assumptions. Moreover, since the COVID-19 pandemic caused many event cancellations due to health concerns

and travel restrictions, the MEC Hackathon platform was designed for virtual access for this event and future Hackathons.

A benefit of this virtual and in-person participation capability is that even larger numbers of software developers can have access. The designed platform was then adapted without eliminating any of the features of the software development environment.

The hackathon edge software development environment was architected by using well-defined open source technologies and consolidated international standards, to offer interoperable solutions that allow developer teams to create their applications according to ETSI standard architecture and APIs. The whole setup was also designed to provide a multi-tenant environment for all developer teams to have a fully separated edge cloud compute resource to ensure confidentiality in a cost-effective virtualized environment.

The development of the Hackathon solution was guided by the following main principles:

1. Edge software development environment availability:

The solution provides the latest version of OpenNESS toolkit, MEC Location API, OpenVINO toolkit, and other software tools to each participating developer/team.

2. Remote/cloud availability of the edge environment:

Remote access to the edge environment is an important criterion in the successful execution and participation of developers worldwide.

3. Isolation and privacy: The machine setup provides each developer team with its own isolated environment, each with a dedicated OS, on a single and powerful hardware platform running OpenNESS.

4. Repeatable and fast deployment: To support a new team joining the hackathon, the server is set up to instantiate new copies of the VM images. Also, in case of a version upgrade of OpenNESS, a new VM can be created to support the update once and then be reused for all developers.

F2F Hackathon Configuration with On-premises Servers

The infrastructure for F2F participation at the Droidcon MEC Hackathon 2020 was based on a cluster of six [Cisco Systems UCS Servers C240 – M5](#). UCS C240 Series rack servers are dual-socket, 2RU rack servers offering robust performance and expandability for a wide range of storage and I/O-intensive infrastructure workloads, from big data and analytics to collaboration. They can be deployed as standalone servers or as part of a cluster³ managed by a virtual infrastructure manager (VIM).

The UCS C240 Series are members of the [Cisco Unified Computing Systems \(UCS\)](#) family, which is the next generation of telco cloud infrastructure. Cisco UCS C-Series Rack Servers (see Figure 1) combine computing, networking, management, storage, and virtualization in a single solution with a self-aware, self-integrating, and intelligent infrastructure. They are advanced, modular, high storage-density rack servers that target a large number of use cases and offer high levels of density and availability.

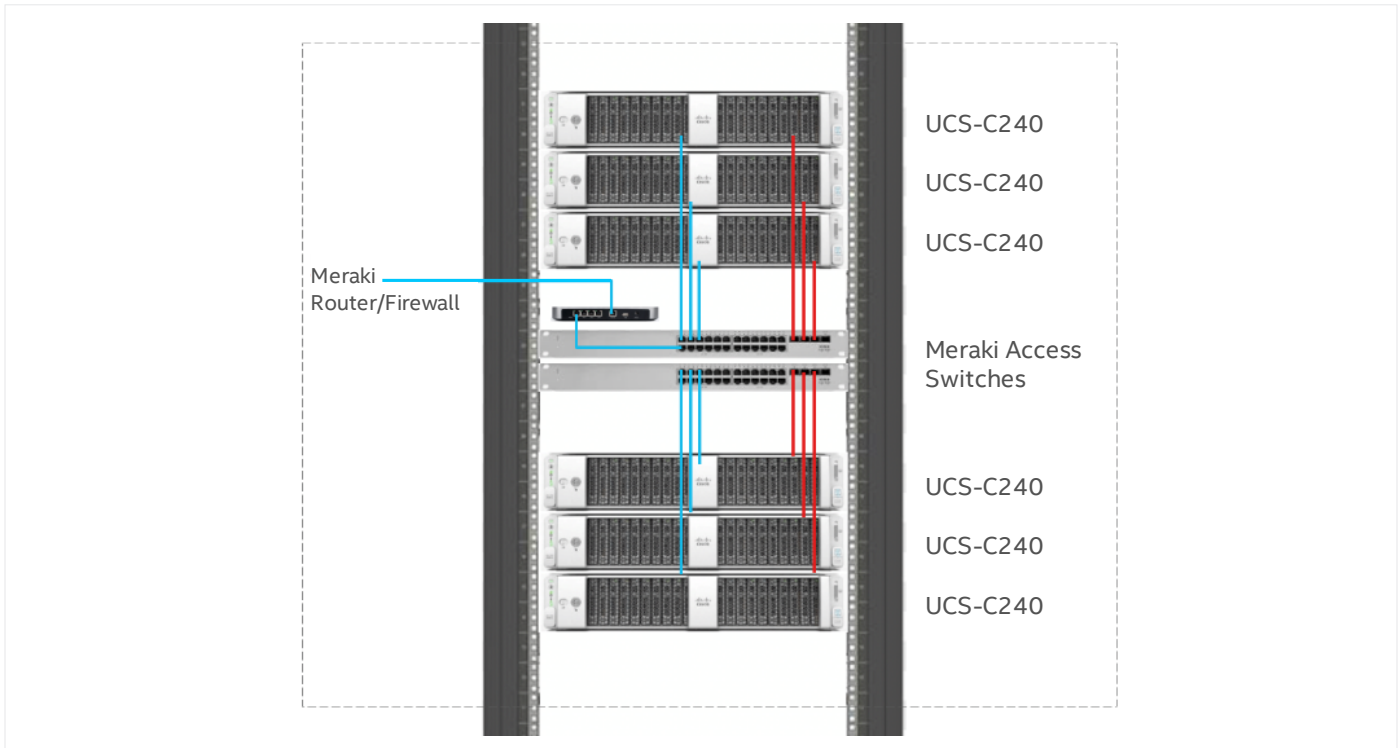


Figure 1. Cisco Systems UCS rack server

Cisco also provided a security and networking architecture via its [Cisco-Meraki](#) cloud-managed wireless, switching, security, and enterprise mobility management solutions.

Cisco-Meraki's cloud configuration and monitoring model allowed the systems to be shipped to the venue with an automated and remote configuration that was prepared in the Meraki dashboard and downloaded automatically by the networking systems as soon as they were connected to the hackathon's Cisco-Meraki cloud tenant, through a security optimized tunnel. Through the cloud, Meraki dashboard administrators could monitor all the live parameters of

clients, servers, and traffic, leveraging a real-time single management interface for detailed insight on traffic, applications, and security information.

The Meraki SD-WAN capabilities were part of the solution and supported traffic virtualization, segmentation, and slicing required by the MEC applications. The hackathon developers leveraged the integrated secure remote VPN access for direct access to the backend of their applications, creating an optimized environment representing an enhanced remote controlled and flexible architecture suitable for MEC node use cases.

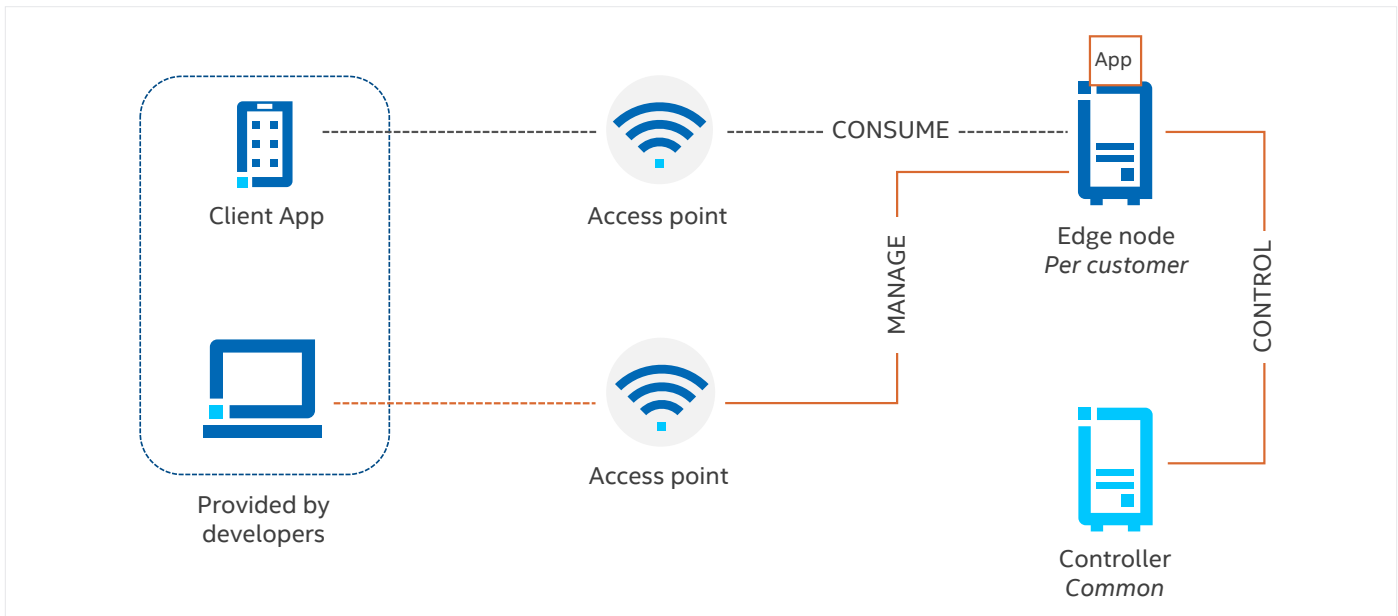


Figure 2. The logical architecture of the F2F event infrastructure

Finally, for management of the overall infrastructure, Cisco provided its [Intersight](#) software as a service (SaaS) infrastructure lifecycle management platform. Intersight delivers simplified configuration, deployment, maintenance, and support of Cisco UCS and third-party connected devices. It also provides the ability to analyze, update, fix, and automate the infrastructure environment, achieving significant TCO savings and delivering applications faster in support of new business initiatives.

Figure 2 shows how a client application (or another management device) from a developer team could access the edge node present in the F2F event infrastructure.

To enable F2F participation, two types of server pod configurations (Figure 3) were used:

- Pod configuration #1 used three Cisco servers, each having an OpenNESS compute node, OpenNESS control node, and client simulator respectively (note: the client simulator was offered to developers also for testing purposes).
- Pod configuration #2 used two Cisco servers. One of these servers had the OpenNESS installation with a combined controller and edge node,⁴ and one server was running the client simulator.

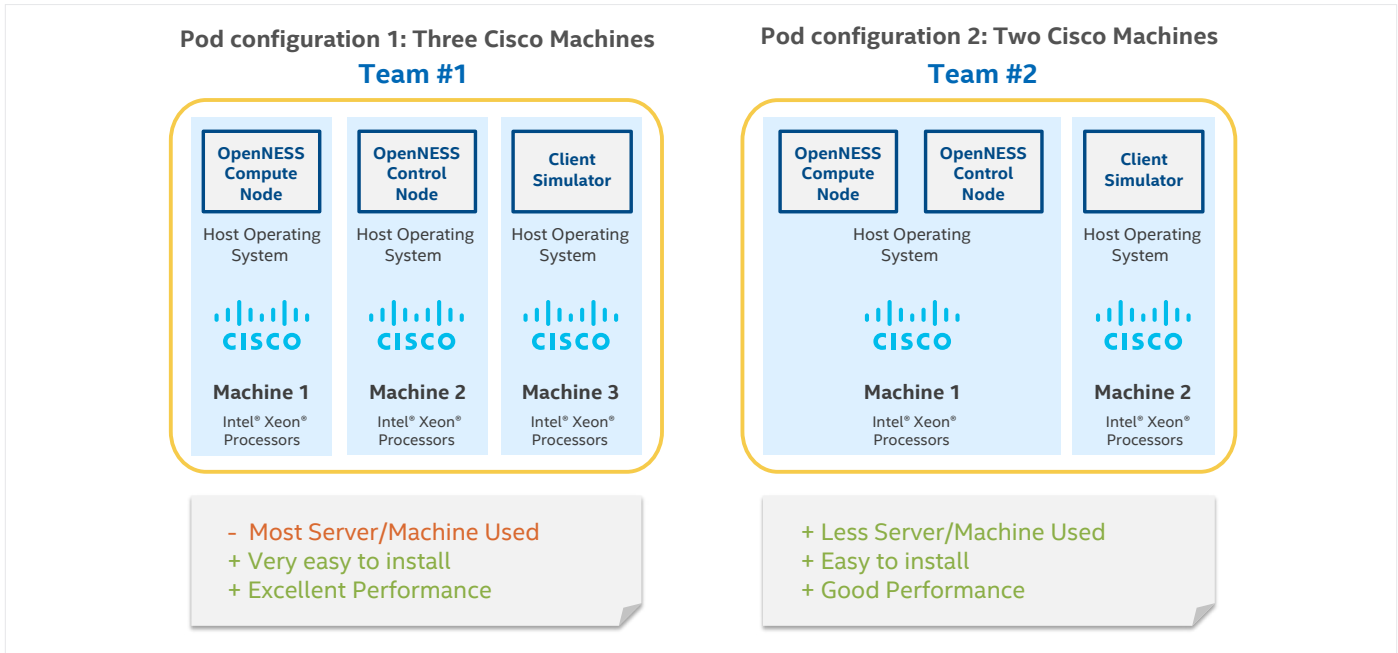


Figure 3. Pod configurations for F2F participation (example for two developers teams on different Cisco servers)

Remote Hackathon Configuration Using Bare-Metal PaaS

The remote version of the hackathon hardware environment was offered in collaboration with Equinix and was based on the [Equinix Metal](#) bare-metal server platform as a service (PaaS) product (the [n2.xlarge configuration](#) was used). The Equinix Metal portfolio offers a choice of hardware platforms for different cost-performance points offering a variety of CPU, memory, and storage options.

The Equinix Metal service comes with several preselected operating systems (including VMware ESXi), but installing a custom image is also an option. Other common capabilities, like installing Kubernetes images, are also possible. Equinix Metal boasts several cloud-adjacent Kubernetes-based workloads, specifically using the [c3.small](#) topology, which features the Intel® Xeon® E-2278G processor with an embedded graphics card. This processor is well suited for cloud native implementations that need the flexibility of a dedicated hardware platform as well as the adjacency to the public cloud.

Furthermore, the Equinix Metal team has pioneered a way to provision and manage bare metal in a similar way to what

Kubernetes has done for containers. The company’s suite of microservices, which has been made open source, is called [Tinkerbell](#).

Equinix Metal platforms are interconnected using the Equinix Fabric service that provides immediate vicinity to tier-one public cloud services. This high-bandwidth and high-availability setup makes Equinix Metal eminently suitable for workloads that have to live at the cloud-edge or “in-between” clouds. Equinix Fabric services are designed to achieve typical latencies (between Equinix Metal machines and the closest public cloud services) in the range of 1-2 ms, using private connections like Azure Express Route or AWS Direct Connect.

The remote machines were virtualized for use by the hackathon teams as shown in an example in Figure 4. A single machine was required for each remote pod serving multiple teams. The example in Figure 4 shows a deployment used by two developer teams. Team one has two VMs assigned (i.e., VM 1 has both controller and compute node, and VM 2 has a client simulator).² Team two also has two VMs assigned—VM 3 and VM 4—where the VM 3 includes not only OpenNESS, but also oneAPI and OpenVINO toolkit installations.

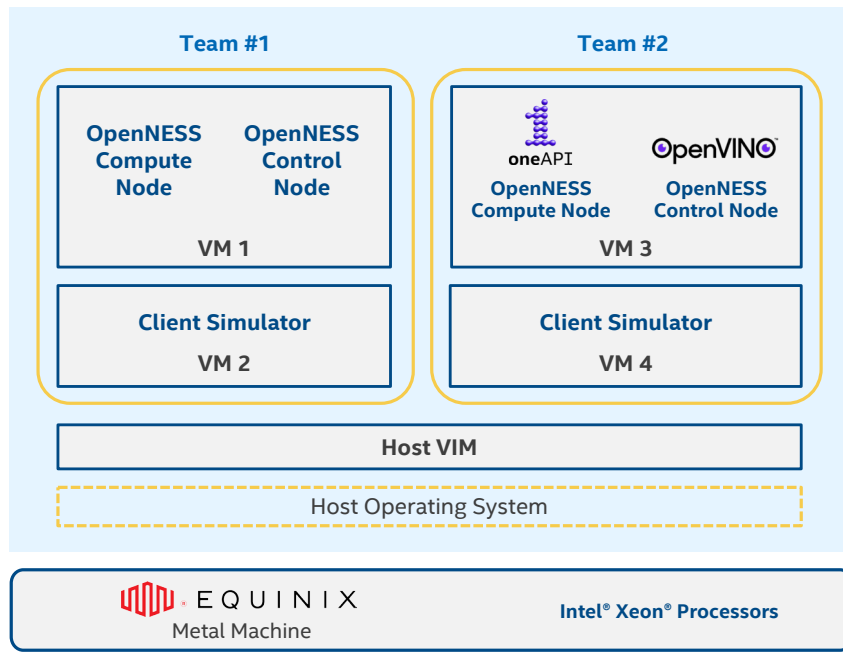


Figure 4. Remote pod setup with OpenNESS installation

Intel® Xeon® Scalable Processors

Industry-standard Intel Xeon Scalable processor platforms, used in cloud-optimized 5G networks, support the convergence of key workloads such as applications and services, control plane processing, high-performance packet processing, and signal processing that take place on edge networks. This delivers a virtualized, software-defined infrastructure to enable cloud capabilities for agile service delivery throughout the network.

With an open architecture that scales and adapts with ease to handle the demands of emerging applications, the Intel Xeon Scalable processor provides a future-ready foundation for agile networks that can operate with cloud economics, be

highly automated and responsive, and support rapid and more secure delivery of new and enhanced services.⁵

Software Platform for Edge Developers

The MEC hackathon solution included a carefully selected software stack to help developers exploit the benefits of MEC technology for their applications.

OpenNESS

OpenNESS (see Figure 5) is an edge computing software toolkit that enables highly optimized and performant edge platforms to onboard and manage applications and network functions with cloud-like agility across any type of network.⁶

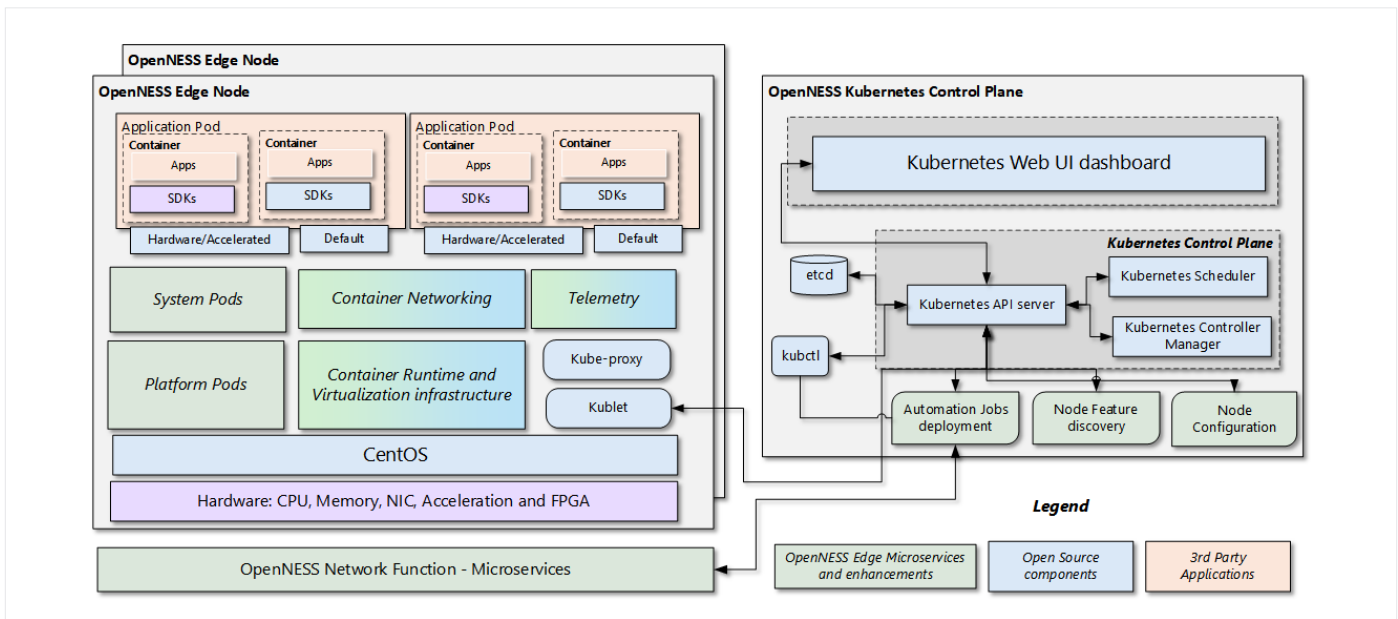


Figure 5. Overview of OpenNESS – Open Network Edge Services Software

OpenNESS is generally deployed as a cluster, but there is a minimal deployment that can be done using a single machine working as an edge node and an edge controller. OpenNESS has a modular, cloud native architecture that is based on Docker and Kubernetes. The modular architecture includes various building blocks as can be seen in Figure 6.

The benefits of OpenNESS to application developers include the following:

- The application microservices/building blocks of OpenNESS let developers enable their applications on OpenNESS as a standalone edge application or as a producer application or a consumer application, depending on the business logic of the application.
- Application developers can leverage OpenNESS APIs to register their applications as a producer and publish

notifications to consuming applications. Similarly, for a consumer application developers can use OpenNESS APIs and subscribe to notifications from producer applications.

- Application developers can offload the processing of application functionality to various OpenNESS accelerator services when using field programmable gate array (FPGA), visual cloud accelerator card for analytics (VCAC-A), or high density deep learning (HDDL) cards. The application needs to integrate with the relevant building blocks and can be especially useful for applications that leverage OpenVINO for AI workloads.
- Application developers can similarly leverage various OpenNESS building blocks to integrate with various data plane technologies or network access protocol technologies.

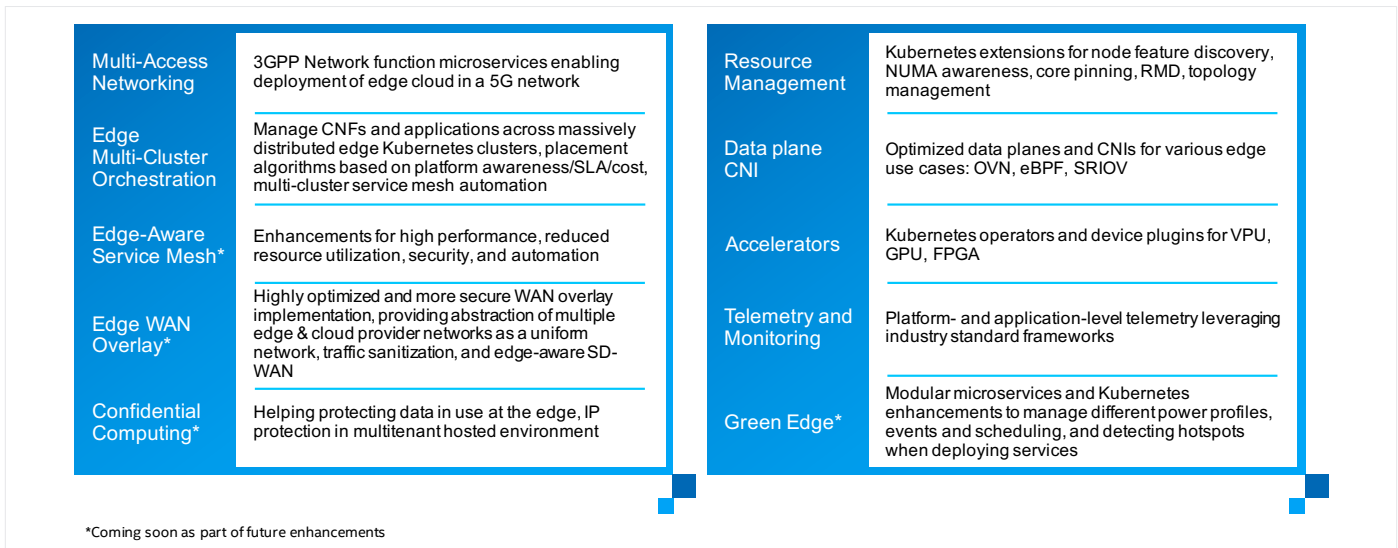


Figure 6. Building blocks displaying the modular architecture of OpenNESS

The OpenNESS toolkit can also give infrastructure developers a jump start in building a MEC platform⁷ for provisioning edge services across vertical markets such as retail, industrial, healthcare, smart cities, and more.

The approach presented in the context of this white paper for an edge hackathon can be leveraged by any infrastructure developer to extend it to practical scenarios. OpenNESS provides a quick start through various experience kits and Intel® Edge Software Hub recipes, including the [OpenNESS Experience Kits](#) and the [Intel® Edge Software Hub – Edge Computing Software and Packages](#).

Intel® Distribution of OpenVINO™ Toolkit

The Intel Distribution of OpenVINO toolkit provides improved neural network performance on Intel processors and enables the development of applications and solutions that solve a variety of tasks, including emulation of human vision, automatic speech recognition, natural language processing, recommendation systems, and many others.

The OpenVINO toolkit:

- Enables deep learning inference.
- Supports heterogeneous execution across multiple platforms: Intel® architecture CPUs, Intel® GPUs, Intel®

FPGAs, Intel® Movidius™ Neural Compute Stick, and Intel® Movidius™ vision processing units (VPUs). This provides implementations spanning cloud architectures to edge devices.

- Speeds time to market via an easy-to-use library of computer vision functions and preoptimized kernels.
- Includes optimized calls for computer vision standards, including OpenCV and OpenCL.

The OpenNESS toolkit enables developers and businesses to easily onboard deep learning solutions that were optimized using the OpenVINO toolkit. Visual inference applications using the OpenVINO toolkit are onboarded by OpenNESS for accelerated and low-latency execution on the edge.

MEC Location API

LINKS Foundation⁸ provided MEC location services for both the F2F and virtual platforms. These services are designed to provide information about the location of user equipment (UE). The ETSI MEC group specified the API of the MEC location service in ETSI GS MEC 013.⁹ The location API is used by MEC applications to interact with the location service for retrieving the desired information.

The location service API is registered to the OpenNESS platform as a producer application and provides location-related information to other software running on the edge computing platform. Therefore, the location service is first authenticated and activated, and then registered on the OpenNESS platform to be discovered from other MEC consumer applications. Once discovered, the location service can interact via location API with consumers. It has a graphical user interface, enabling developers to simulate mobile users' movements by feeding the simulator with a GPS track in GPX format.

From a consumer point of view, the location API appears as a RESTful web service that exposes ETSI standards-compliant endpoints. The location API is used by MEC applications to interact with the location service for retrieving the desired information. The specifications, procedures, and the retrievable location information details are available in the ETSI GS MEC 013 document.

Consumer Registration

The MEC Location Service is discoverable if the consumer application is correctly registered on the OpenNESS platform. Different programming languages can be used to register on the OpenNESS platform. This allows authenticating through a certificate signed request (CSR), which must include all the distinguishing name information such as country name, state, and a common name with the format "<application-name>:<application-type>".

The format is essential for being accepted by the platform. If the CSR is accepted from the edge application agent (EAA), the consumer is authenticated and the EAA returns a certificate needed for the next steps, such as discovering and registering to MEC services.

Location API Discovering and Registration

Once authenticated on the OpenNESS platform, a consumer can look for the location service through the edge application APIs. Below are a few steps the consumer follows for the authentication procedure:

1. Prepare a JSON request including the generated CSR PEM-encoded certificate;
2. Send the authentication request to the OpenNESS platform;
3. Save the received certificate;
4. Send a secure GET request to the EAA including the certificate, pointing to the services endpoint;
5. Save the response of which services are included in EAA in a .JSON file.

In this way, the consumer application gets all the information needed for contacting the location API service. Indeed, when a producer is registered on the EAAs, it inserts all the information about the kind of services it exposes.

The location API service has the searched information in the "info" field, where there is useful information for the consumer to start requesting information and be informed about the location of users. Therefore, once that information is retrieved, the consumer can start requesting data as any other ETSI-standard MEC application by following all the implemented HTTP requests.

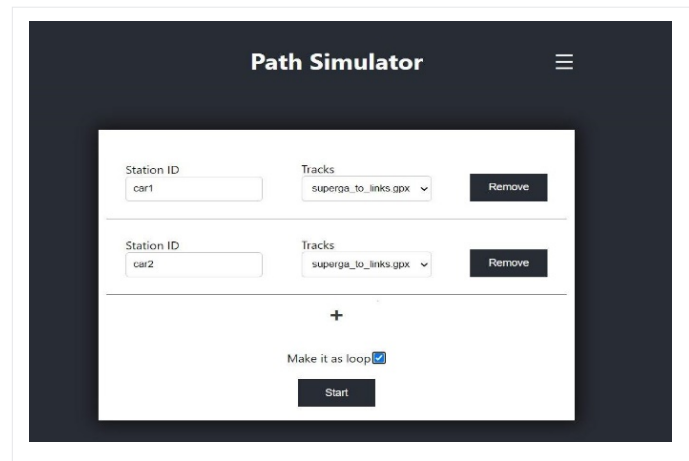


Figure 7. GUI simulator

How to Use Location API – Simulator

The location service has to be fed by simulated location data to use its functions. To fulfill this scope, it exposes a simulation service to simulate the users' movement according to a provided path.

A user interface is available to upload paths (in GPX format) that will simulate the movement of mobile users. This is done through a simple web page available at the 8081 port.

The uploaded paths will remain available for future use. With the GUI the user can:

1. Choose an identifier for the UE;
2. Choose the path that the user will follow;
3. Add multiple users;
4. Start a simulation, which triggers the availability of dynamic user location accessible through the location API. The user can choose either to start a simulation running in a loop or only once.

Further information can be retrieved from the official OpenNESS repository.¹⁰

How to Use Location API – Practical Use

Once the simulation is started, the location API has all the required information, and the consumer retrieves the location information by following the RESTful ETSI-compliant requests (for further details see ETSI GS MEC 013 (v2.1.1, Section 5.3 – Sequence Diagrams). The location service API supports both queries and subscriptions (using the pub/sub mechanism).

From the perspective of the location service consumer, below is a list of examples in which the location API plays the central role. (Refer also to the Location API Discovering and Registration section above.)

- **GET the user list.** This request will obtain all the general information about users that are known by the location service through the RESTful API, and therefore retrieves information about the IP address, the associated zone, or the access point.
- **GET detailed information about a single user.** This request is for detailed information about a single user obtained by exploiting the received user list, in which there is an IP address associated with that user.

- **POST a subscription.** Using this request initiates interest in single or multiple users over a period of time. Using this method, the location API periodically reports the requested information. If the request goes well, the response will include the generated subscriptionId. The location API will start to send messages to the indicated callback in the JSON request according to the data that the user is interested in.
- **DELETE a subscription.** Finally, this request is used when a consumer does not need to receive information from a subscription; it can be simply deleted by sending a DELETE request including the subscriptionId. Further details can be found in ETSI GS MEC 013 (v2.1.1, Section 7 – API Definition).

The MEC Location API Service is valuable for developing MEC applications, such as tracking, location-based service recommendation, and others (for more examples, see ETSI GS MEC 002).¹¹ The location service is platform independent per ETSI standards.

The MEC Location API Service enables other procedures for acquiring users' movements in a geographic area, the distance to a specific location, or another user. For an in-depth view of additional use cases, see ETSI GS MEC 002.

Detailed Description of the MEC Hackathon Server Solution

As described in the previous sections, two different setups were offered to developers, depending on their likelihood to attend the Droidcon MEC Hackathon 2020 F2F or virtually. Each of these two setups is described in detail in the following sections.

Setup for F2F Participation

For the developers who wanted to access an end-to-end environment F2F, the system included the Cisco UCS C240-M5 servers with the following configuration:

- Dual Intel® Xeon® Gold 6132 processors
- 256 GB DDR4-2666-MHz RAM
- Two 1 TB 2.5-inch enterprise value 6 GB SATA SSDs
- One Intel® Ethernet Converged Network Adapter X710 with dual 10G SFP+ ports
- Redundant power supply for high availability

Figure 8 shows the final configuration of the end-to-end environment developed for the F2F participation in the Droidcon MEC Hackathon 2020. In the provided setup, the client application is running on consumer UEs that are connected to the edge node through the Wi-Fi access point.

Setup for Virtual Participation

For the developers who wanted to attend the hackathon competition virtually, the setup used the following Equinix Metal server specifications:

- Dual Intel® Xeon® Gold 5120 CPUs @ 2.20 GHz
- 384 GB DDR4-2666-MHz RAM
- Two 120 GB SATA SSDs
- One 3.8 TB NVME
- Four 10 Gbps bonded ports

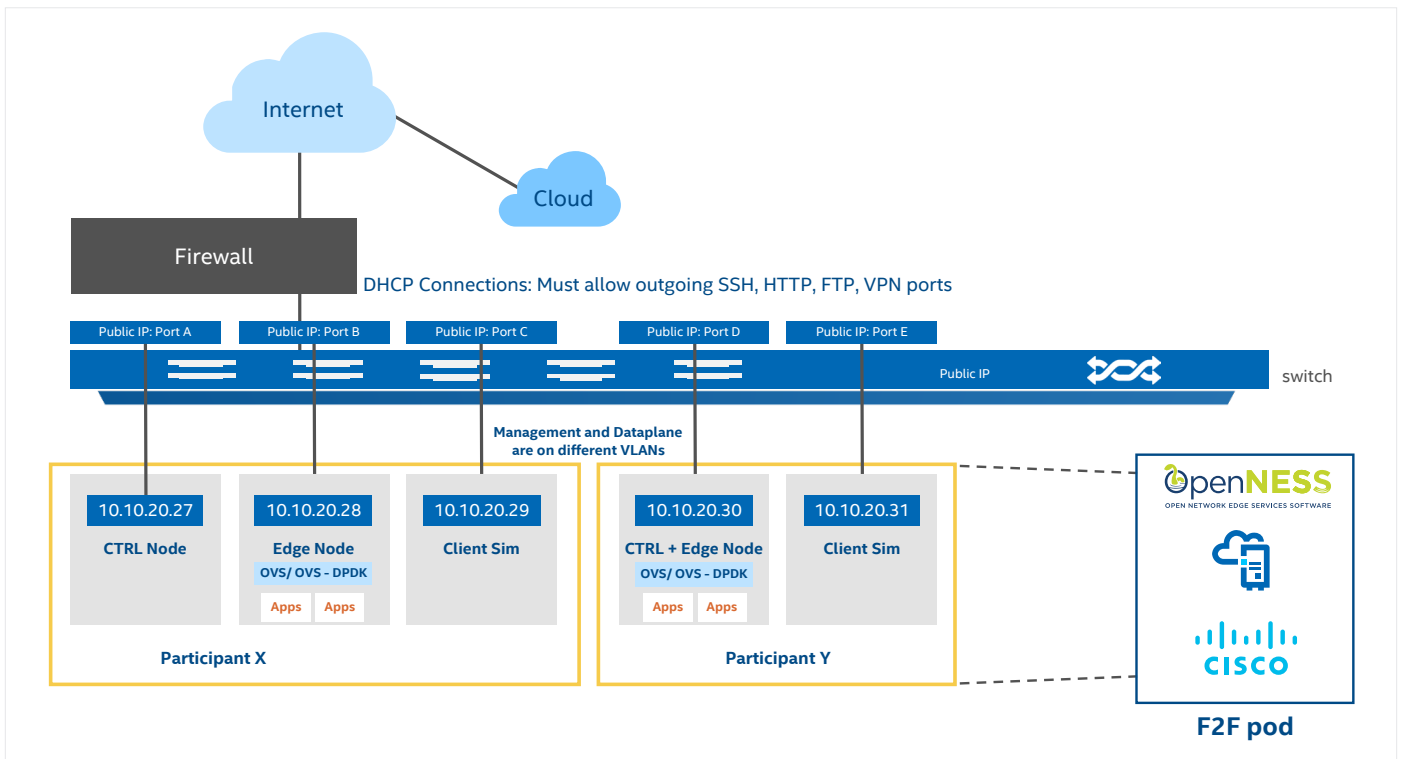


Figure 8. Complete face-to-face pod setup

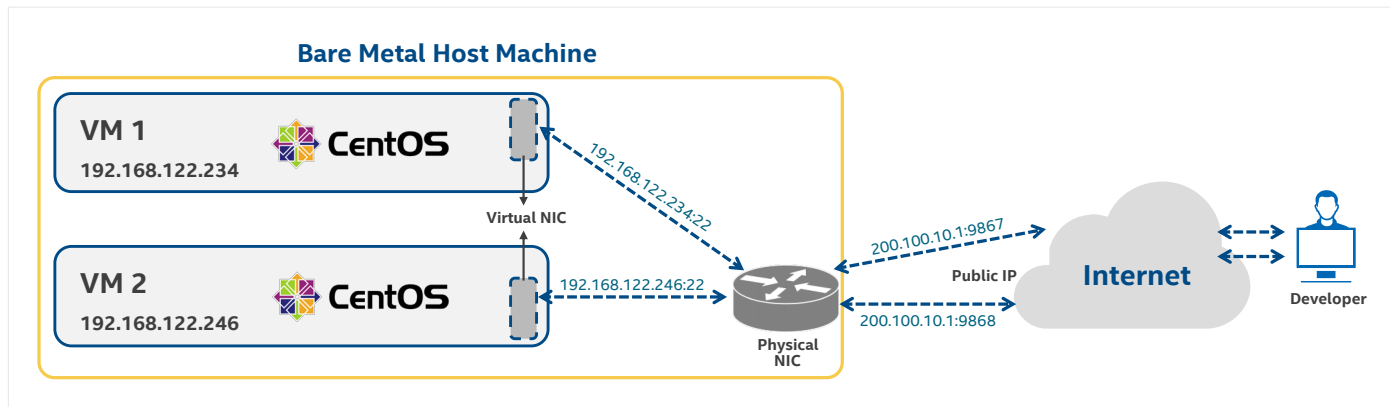


Figure 9. POD access to the external environment

The following steps were taken to set up the remote environment:

- 1. Provide Global Server Availability:** Server access was made available to developers worldwide by leveraging the global presence of Equinix. The worldwide server availability ensured low latency for the application developers. Taking advantage of low latency communication between device and client application (managed by application developers) and MEC application (running on the Equinix server) is one of the key benefits of edge deployments, and it is also an important requirement for application development.
- 2. Create VM Environment:** Virtualization infrastructure, including libvirt, virt-manager, and qemu-kvm, was installed on the bare-metal servers. In addition, VMs were created, as shown in Figure 9, by following the configuration described on the OpenNESS website for the hardware requirement for the OpenNESS installation kit.⁷

While a standard Equinix metal deployment comes with a single static IP address to the bare-metal host system, port forwarding can provide each team with direct access to the target VMs assigned to them, therefore allowing global anytime access to the target servers. For example, Table 1 shows the mapping of internal IP addresses to external IP address mapping for two VMs. Virtual hackathon teams only had access to their VMs and had no access to the bare-metal host system.

- 3. Initiate OpenNESS:** OpenNESS Release 20.06³ was used, including both controller node and compute node running on a single VM. A client simulator containing the client application was instantiated in a separate VM running on the same Equinix machine.
- 4. Finalize configuration:** The end-to-end environment provided isolated environments for each of the participating teams, each with its own OS on a single and powerful platform.

INTERNAL NETWORK - PRIVATE IP ADDRESS	EXTERNAL NETWORK - PUBLIC IP ADDRESS
192.168.122.234:22	200.100.10.1:9867
192.168.122.246:22	200.100.10.1:9868

Table 1. IP Address Mapping



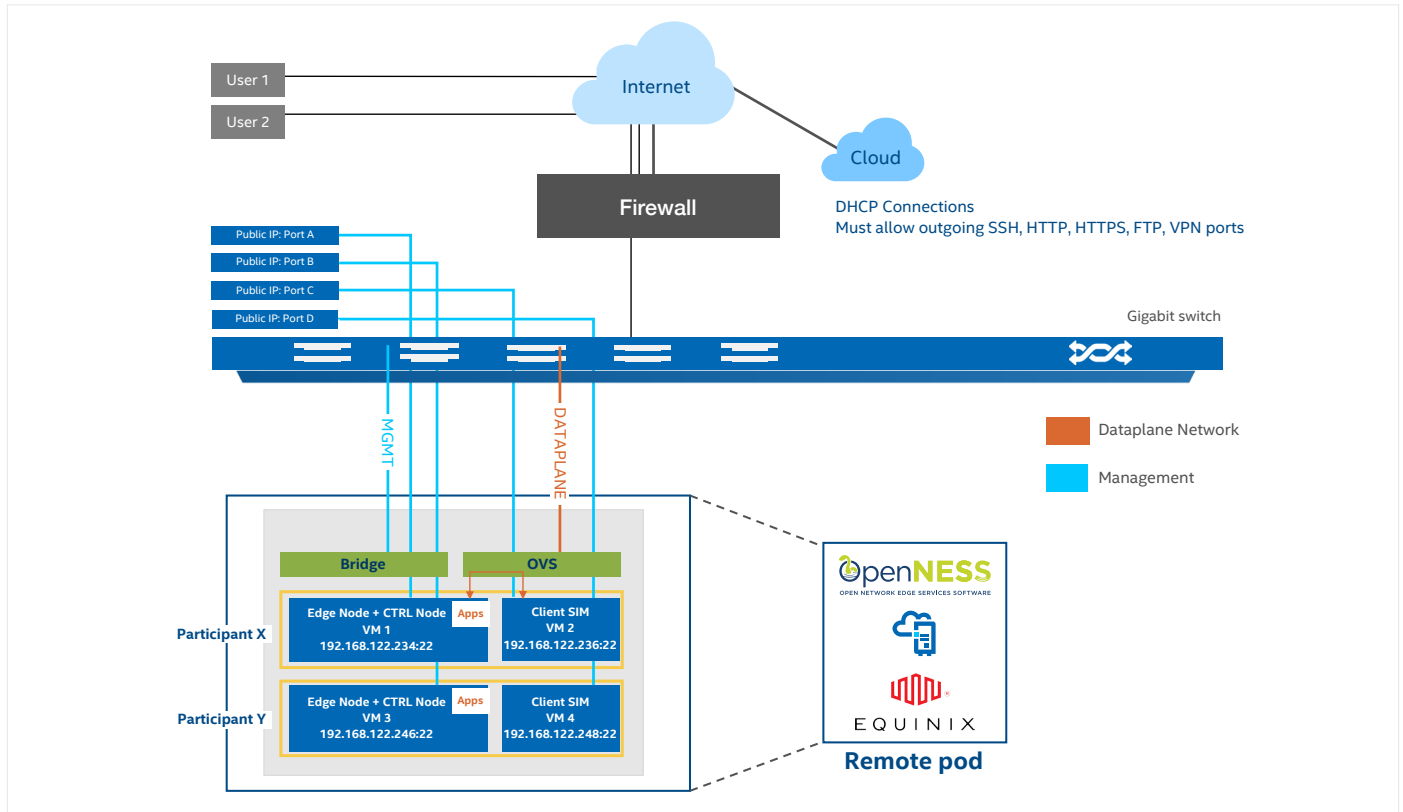


Figure 10. Complete remote pod setup

Figure 10 shows the final configuration of the end-to-end virtual hackathon environment for application software developers, which included the following:

- A client application running either on customer laptops or on the client simulator (for example, for debugging/testing purposes).
- Customer laptops connected to the edge node through the internet.
- The client simulator that could be used to test a local OpenNESS implementation without the data plane going through the internet. This configuration was handy in the case of internet connectivity, low throughput, and high latency issues.

Conclusion

The paper describes an end-to-end hardware and software environment that Intel and its partners made available to application software developers of the Droidcon MEC Hackathon 2020, and possibly also for future participation in similar MEC hackathons—either F2F or virtually. The main advantages of the designed solution include the following:

- It is specifically designed to address common challenges with edge deployments;
- it leverages virtualization technologies to install OpenNESS so that edge computing infrastructure can be shared, replicated, and installed quickly on other MEC nodes;
- it enables more resources to be allocated based on the requirement. More hardware resources such as memory

or SSD space, can be allocated depending on the developer’s application usage. The infrastructure will help in a faster deployment and upgrade of individual VMs with the latest OpenNESS software. As more teams join the hackathon, no new servers are needed, thanks to the ability to instantiate new copies of the VM images on the machine.

Standards-based MEC solutions enable agile and efficient networks with faster service deployment, high return on investment (ROI), lower total cost of ownership (TCO), and improved scalability. Furthermore, the combination of this comprehensive MEC reference platform with the implementation of an ETSI-compliant MEC Location API, represents an attractive package for developers by offering a complete set of tools for helping edge application development.

The solution, developed by Intel in collaboration with Cisco, Equinix, and LINKS Foundation, was first adopted as a flexible reference design for the Droidcon MEC Hackathon 2020, as a means to provide remote access of an end-to-end environment to application developers at the hackathon competition. Now, this system design is available as a scalable and cost-effective solution that is suitable for bigger developer events, and provides the needed flexibility to allow possible remote participation. Robust and flexible edge computing software, based on open-source reference implementations aligned with standards, can accelerate innovation within the industry. This hackathon system plays an important role in giving developers an opportunity and a motivation to develop new applications suited for the edge.

Authors

Intel

Anish Rawat, Dario Sabella, Purvi Thakkar, Xavier Simonart

LINKS Foundation

Maurizio Florida, Edoardo Bonetto, Daniele Brevi, Riccardo Scopigno

Cisco

Mirko Berlier, Alessandro Breccia

Equinix

Guido Coenders, Jacob Smith

Appendix 1: Abbreviations

3GPP	3rd Generation Partnership Project
5G	5th Generation
API	Application Programming Interface
CDN	Content Delivery Network
CSR	Certificate Signed Request
CT	Connectivity Technology
DHCP	Dynamic Host Configuration Protocol
EAA	Edge Application Agent
ETSI	The European Telecommunication Standards Institute
F2F	Face to Face
FPGA	Field Programmable Gate Array
GPS	Global Positioning System
GPX	GPS Exchange Format
GUI	Graphical User Interface
HDDL	High Density Deep Learning
ISG	Industry Specification Group
IT	Information Technology
JSON	JavaScript Object Notation
MEC	Multi-access Edge Computing
OpenNESS	Open Network Edge Services Software
OpenVINO toolkit	Open Visual Inference and Neural Network Optimization toolkit
OT	Operational Technology
PaaS	Platform as a Service
PoP	Point of Presence
RAN	Radio Access Network
SDK	Software Development Kit
TCO	Total Cost of Ownership
UCS	Unified Computing System (i.e., the Cisco UCS platform)
UE	User Equipment
V2X	Vehicle to Everything
VCAC-A	Visual Cloud Accelerator Card for Analytics
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VPU	Vision Processing Unit

References

- ¹ ETSI White Paper “Harmonizing standards for edge computing - A synergized architecture leveraging ETSI ISG MEC and 3GPP specifications”, July 2020, Link: https://www.etsi.org/images/files/ETSIWhitePapers/ETSI_wp36_Harmonizing-standards-for-edge-computing.pdf
- ² Intel White Paper “Edge Computing: from standard to actual infrastructure deployment and software development”, November 2019; Link: <https://builders.intel.com/docs/networkbuilders/edge-computing-from-standard-to-actual-infrastructure-deployment-and-software-development.pdf>
- ³ A single-node cluster scenario, that is, a single machine working as a control plane and node, was developed for both the F2F and virtual pod setups. This type of installation is very effective for resource-constrained edge deployments. Also, co-locating the control plane improves reliability by removing a point of failure.
- ⁴ OpenNESS website - <https://www.openness.org/news-and-events/openness-20-06-released>
- ⁵ Intel Xeon Scalable processors product brief - <https://www.intel.com/content/www/us/en/products/docs/processors/xeon/2nd-gen-xeon-scalable-processors-brief.html>
- ⁶ Intel Network Builders: “Introduction to Open Network Edge Services Software (OpenNESS)”, Link: <https://networkbuilders.intel.com/university/course/introduction-to-open-network-edge-services-software-openness>
- ⁷ OpenNESS hardware and software compatibility: https://github.com/open-ness/specs/blob/master/openness_releasenotes.md#hardware-and-software-compatibility
- ⁸ <https://linksfoundation.com/en/links-foundation/>
- ⁹ ETSI GS MEC 013 V2.1.1 (2019-09), Multi-access Edge Computing (MEC) Location API, Link: https://www.etsi.org/deliver/etsi_gs/mec/001_099/013/02.01.01_60/gs_mec013v020101p.pdf
- ¹⁰ <https://github.com/open-ness/edgeapps/tree/master/applications/location-api>
- ¹¹ ETSI GS MEC 002 V2.1.1 (2018-10), Multi-access Edge Computing (MEC) - Phase 2: Use Cases and Requirements, Link: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/02.01.01_60/gs_MEC002v020101p.pdf



Notices & Disclaimers

Intel technologies may require enabled hardware, software or service activation.


No product or component can be absolutely secure.

Your costs and results may vary.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

0321/DO/H09/PDF

 Please Recycle

345231-002US