PoC Project name: Sensor Data Aggregation and Ingestion

Classification: IOWN Global Forum Recognized PoC

Stage: SSF PoC Report

Confidentiality: Public

Version: 1.0 (Phase 1)

January 24, 2024

1. Introduction

In the Area Management Security Use Case of IOWN GF, the energy efficiency of computing infrastructure needs to be drastically improved for processing data in real-time image analysis of video streams from a massive number of surveillance cameras. As a solution to this problem, we conducted this PoC employing IOWN GF architecture such as Open all-photonics network (APN), data-centric infrastructure (DCI) with data-plane acceleration (DPA). That is, this PoC demonstrates the heterogeneous computing by Kubernetes (OpenShift container platform) enabling x86 Logical Service Node with x86 CPU, smartNIC, data processing unit (DPU), and GPU for the AI inference of video streams. In addition, performance evaluations have been conducted through Open APN to show that the use case can be achieved on geographically-distributed computing infrastructure without bottlenecks.

As most IOWN GF members do not yet have a composable disaggregated infrastructure (CDI) system which is under discussion as a part of DCI Cluster in DCI-TF, this PoC report might help all IOWN GF members to see how the PoC can be started with a commercial off-the-shelf (COTS) server instead of a DCI system with CDI features. This also helps DCI-TF to learn about what type of composable Peripheral Component Interconnect Express (PCIe) device IOWN GF members are trying to use for the purpose of the use case reference implementation model (RIM) PoC.

2. PoC Project Completion Status and Project Participants

This PoC is a multi-phase project. This PoC system in the first phase is described in section 6. The performance measurements and evaluations were conducted using the demonstration system described in section 7.

- Overall PoC Project Completion Status: In progress
- PoC Stage: Significant Step Forward (SSF)
- PoC Project Name: CPS AM Security PoC-1: Sensor Data Aggregation and Ingestion (SDAI)

PoC Teams:

- Members:

Member	Company	Name
A	Fujitsu	Takayuki Uchihira, Naoki Oguchi, Jin Hase, Daisuke Matsuda
B	NTT	Rintaro Harada, Ryosuke Kurebayashi
C	NVIDIA	Hideaki Tagami, Takashi Noda, Khanh Duc, Elad Blatt
D	Red Hat	Hidetsugu Sugiyama, Erwan Gallen

3. PoC Goals Status Report

- PoC Project Goal #1: To improve energy efficiency in SDAI systems
- Goal Status: Met in 2023

4. Supported Use Case

The CPS AM Security use case PoC Reference [PoC Reference] defined the following PoC scopes to include critical parts of the AM Security Use Case RIM as illustrated in Figure 4-1, PoC-1 for Sensor Data Aggregation and Ingestion (SDAI) and PoC-2 for Data Hub and Live 4D Map. This PoC was conducted in scope of PoC-1 for SDAI.

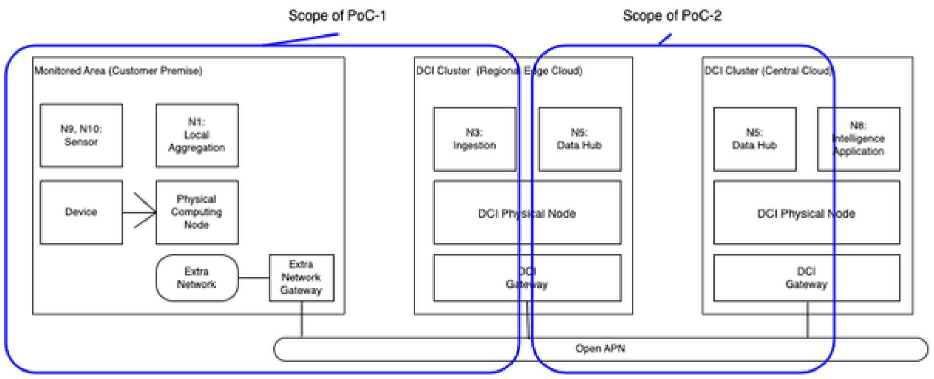


Figure 4-1: The scope of this PoC (PoC-1)

The SDAI system aggregates sensor data (e.g., videos, LiDAR data) in a Monitored Area and analyzes the data in a Regional Edge Cloud. The overview of the system is illustrated in Figure 4-2. Note that our PoC utilized only videos as sensor data at this time.

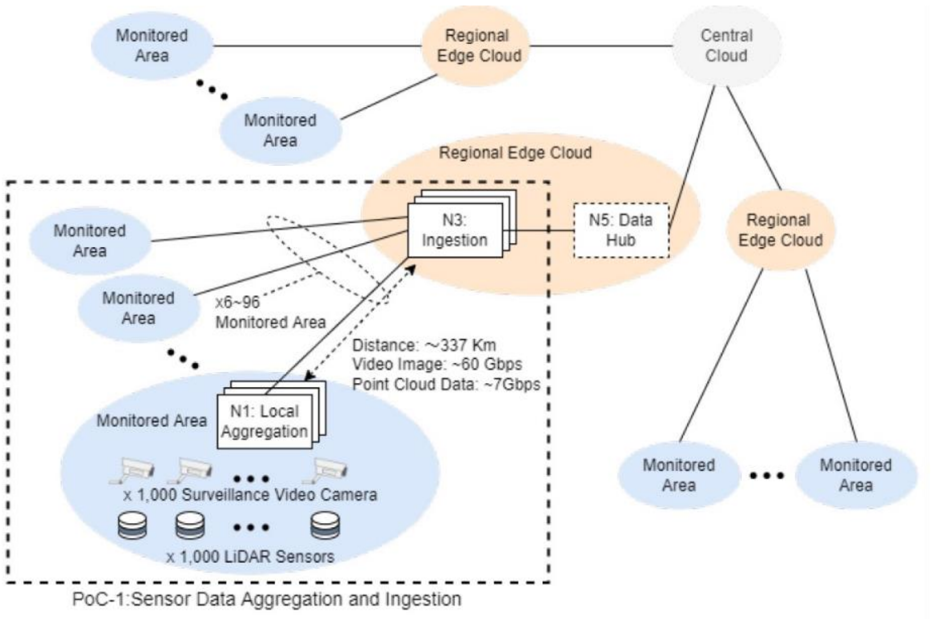


Figure 4-2: Overview of the sensor data aggregation and ingestion (SDAI) system

5. Confirmation of PoC Demonstration

Our PoC team demonstrated this PoC. In this demonstration, a video server substituting for cameras and a local aggregation node were located in NTT Yokosuka R&D Center, Kanagawa, Japan. An ingestion node and a data hub node were located in NTT Musashino R&D Center, Tokyo, Japan. Figure 5-1 illustrates the locations of these two sites, which were connected via Open APN. Setup of Open APN in this demonstration is described in Appendix A-4. Figure 5-2 is a photo of the video server and the local aggregation node. Figure 5-3 is a photo of the ingestion node and the data hub node.

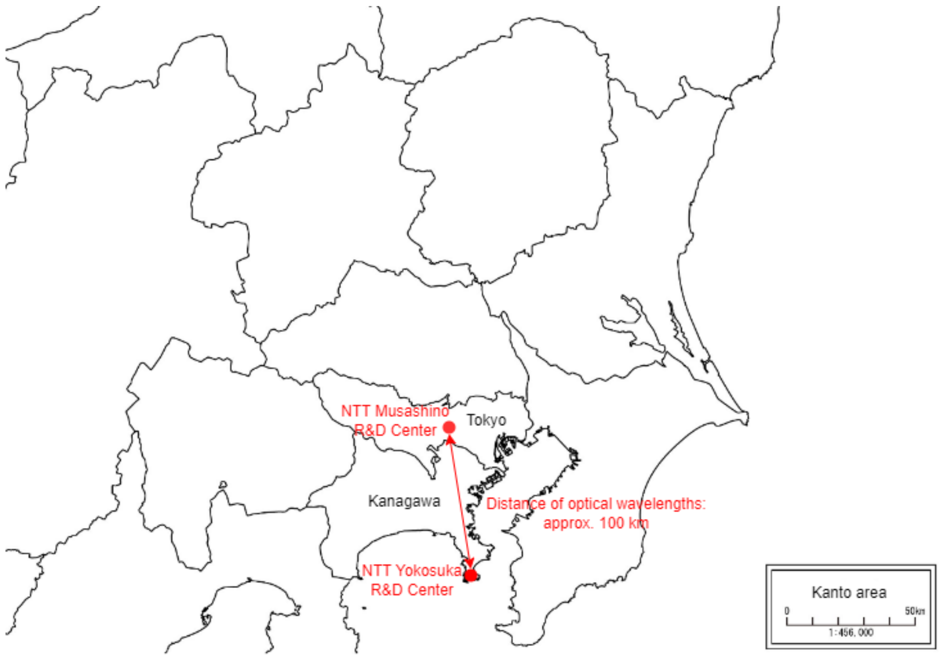


Figure 5-1: The locations of NTT Musashino R&D Center and NTT Yokosuka R&D Center

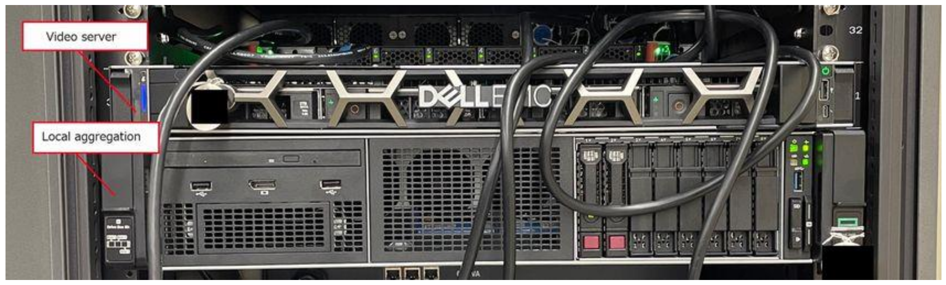


Figure 5-2: The video server and the local aggregation node located in NTT Yokosuka R&D Center

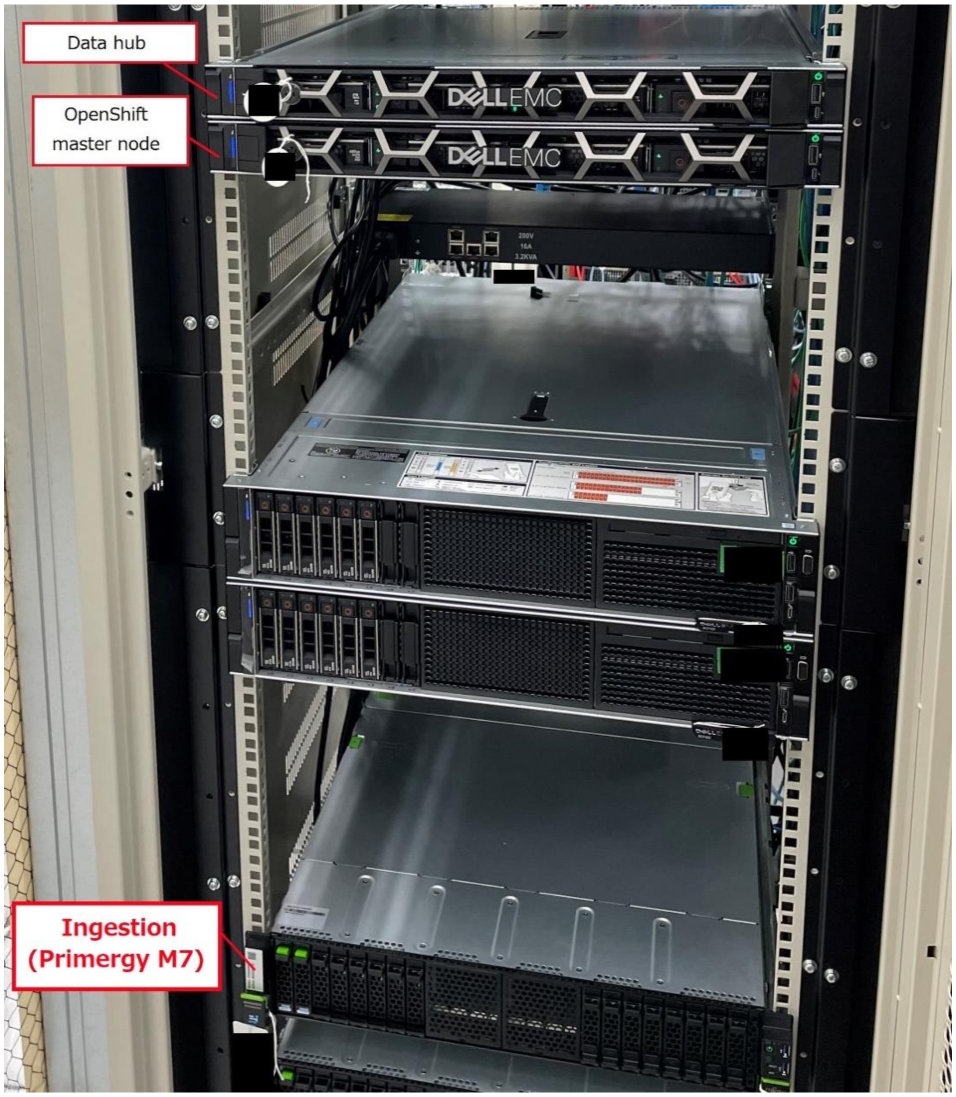


Figure 5-3: The ingestion node and the data hub node located in NTT Musashino R&D Center

6. PoC System Configuration and Implementation

6.1. Overview of implemented PoC System Configuration

This subsection describes a conventional pipeline and our implemented pipeline for a video inference system for this PoC. Figure 6-1 illustrates the PoC system configuration that employs various hardware-acceleration technologies including remote direct memory access (RDMA) for improving energy efficiency. The video data were processed outside the Kubernetes (K8s) cluster of the Ingestion node because of the RDMA interface, while the container applications deployed through K8s cluster can do application aware autoscaling per resource usage, etc. The system is composed of a video server substituting for cameras, a local aggregation node, an ingestion node, and a data hub node. In this PoC, there are two options regarding the configuration of the ingestion node. Option A (LSN-1A) employs an x86 server with a GPU as an ingestion node, whereas Option B (LSN-1B) employs a converged accelerator (DPU + GPU). Option B is expected to offload control-plane processing to the CPU on the DPU, which can further improve efficiency. Also, the converged accelerator is expected to scale out in a simpler manner. This PoC Report mainly describes Option A, though our activities for Option B are described in Appendix.

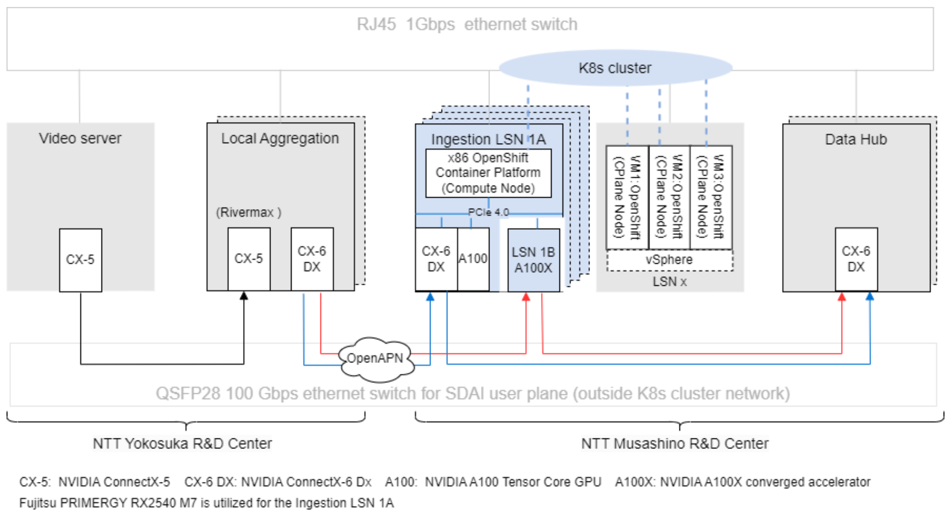

RJ45 1Gbps ethernet switch
K8s cluster
Video server
CX-5
Local Aggregation
(Rivermax)
CX-5
CX-6 DX
Ingestion LSN 1A
x86 OpenShift Container Platform (Compute Node)
PCIe 4.0
CX-6 DX
A100
LSN 1B A100X
VM1:OpenShift (CPlane Node)
VM2:OpenShift (CPlane Node)
VM3:OpenShift (CPlane Node)
vSphere
LSN x
Data Hub
CX-6 DX
OpenAPN
QSFP28 100 Gbps ethernet switch for SDAI user plane (outside K8s cluster network)
NTT Yokosuka R&D Center
NTT Musashino R&D Center
CX-5: NVIDIA ConnectX-5 CX-6 DX: NVIDIA ConnectX-6 Dx A100: NVIDIA A100 Tensor Core GPU A100X: NVIDIA A100X converged accelerator
Fujitsu PRIMERGY RX2540 M7 is utilized for the Ingestion LSN 1A

Figure 6-1: Configuration of our implemented PoC system

6.2. Cameras

A video server substitutes for cameras. This server sends multiple video streams that are encoded by MJPEG and encapsulated by RTP. The RTP packets are transmitted by UDP/IP. The frame rate of each video stream is 15 fps. The resolution of each frame is Full HD (i.e., 1920 x 1080 pixels).

6.3. Local aggregation node

A local aggregation node receives the video streams from the video server and then sends them to an ingestion node. As shown in Figure 6-2, the conventional pipeline employs simple RTP as a conventional protocol to receive and send the video streams. In this pipeline, both data plane and control plane are affected by kernel overhead. On the other hand, our implemented pipeline is based on a hardware-accelerated architecture eliminating kernel overhead in terms of data plane. As shown in Figure 6-3, ConnectXes are adopted as smartNICs. With the ConnectXes, Rivermax is utilized to put the received video data directly onto host memory without kernel overhead. Also, unified communication X (UCX) is utilized for GPU Direct RDMA to send the video data directly onto GPU memory in the ingestion node also without kernel overhead. Multiple frames are batched for RDMA, which improves signaling overhead in RDMA. By connecting sites with Open APN, almost no packet loss will occur, so long-distance RDMA is possible.

Note that some additional processing (e.g., wide-area load balancing, remapping of the cameras and the ingestion nodes, frame rate control, masking of privacy, preliminary analysis, and encryption) could be performed in the NOP (no operation) blocks in Figure 6-2 and 6-3 in a real deployment.

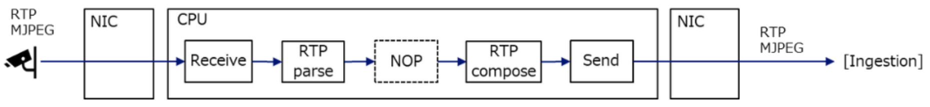


Figure 6-2: Conventional pipeline for the local aggregation node

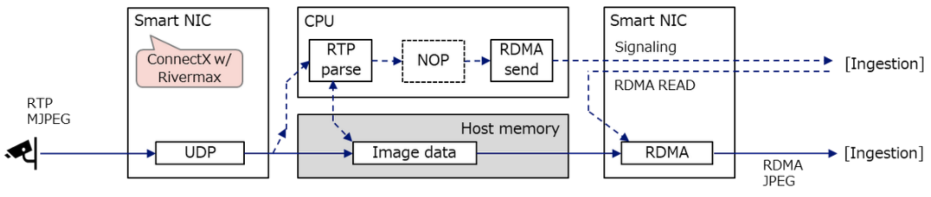


Figure 6-3: Our implemented pipeline for the local aggregation node

6.4. Ingestion node

The ingestion node receives the video data from the local aggregation node, decodes them, executes inference, and finally sends inference results as well as the video data to a data hub node. As shown in Figure 6–4, the conventional pipeline is CPU–centric, so both data plane and control plane are affected by kernel overhead. In contrast, our implemented pipeline achieves a GPU–based data–plane processing without kernel overhead. As shown in Figure 6–5, ConnectXes are adopted as smartNICs, and an A100 is adopted as a GPU. With these two hardware accelerators, several hardware–acceleration technologies are utilized as described in Table 6–1. Multiple frames are batched for RDMA, decoding and inference including pre– and post–processing, which improves signaling overhead in RDMA and efficiency of GPU resources. Note that detailed specifications of an RDMA–based interface for the data hub node will be studied in later phases.

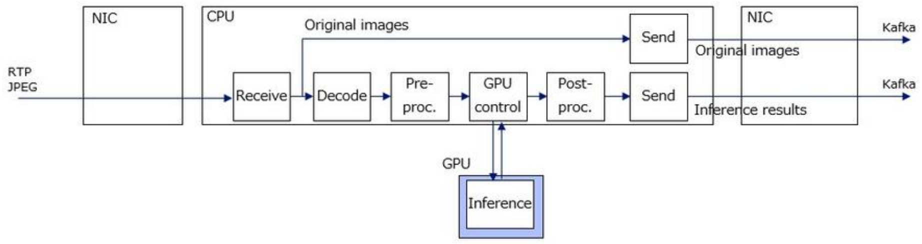


Figure 6–4: Conventional pipeline for the ingestion node

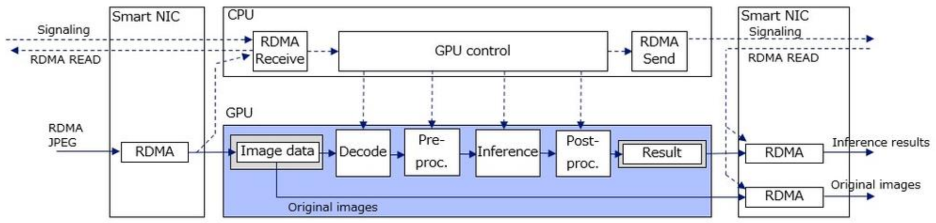


Figure 6–5: Our implemented pipeline for the ingestion node

Table 6-1: Hardware-acceleration technologies for ingestion node in our implementation

Hardware-acceleration technologies	**Purposes**
UCX	GPU Direct RDMA to send/receive the video data directly from/to GPU memory
nvJPEG	High-speed decoding using a hardware decoder named NVJPG on A100 GPU
CV-CUDA	Pre-processing (e.g., convert the type of video data for inference models) on A100 GPU
TensorRT (Note that it is used also in the conventional pipeline.)	Accelerate inference on A100 GPU by optimizing inference models
CUDA	Post-processing (e.g, pick out detected objects as bounding boxes) on A100 GPU

6.5. Data hub node

The data hub node receives the inference results and the video data from the ingestion node. The conventional pipeline employs Kafka to receive and store the inference results and the video data. On the other hand, our implemented pipeline utilizes UCX for RDMA-based reception. Since Kafka itself does not support RDMA and details of the data hub node are out of the scope in this PoC, it is implemented in a pseudo manner. An alternative solution might be Apache Arrow Flight that can support UCX/gRPC base data transfer, but the current IOWN Data Hub document does not describe any Apache Arrow in-memory solution yet.

6.6. Container platform

OpenShift is introduced as a container platform on top of an x86-based Logical Service Node. Along with the OpenShift Container Platform, Site Reliability Engineering (SRE) software of OpenShift Operators [Operator] including NVIDIA’s K8s Operator are introduced in OpenShift Kubernetes Engine for making it easier to manage cloud native life cycles of workloads with the hardware-based data transfer feature. The software stack around the OpenShift is shown in Figure 6-6.

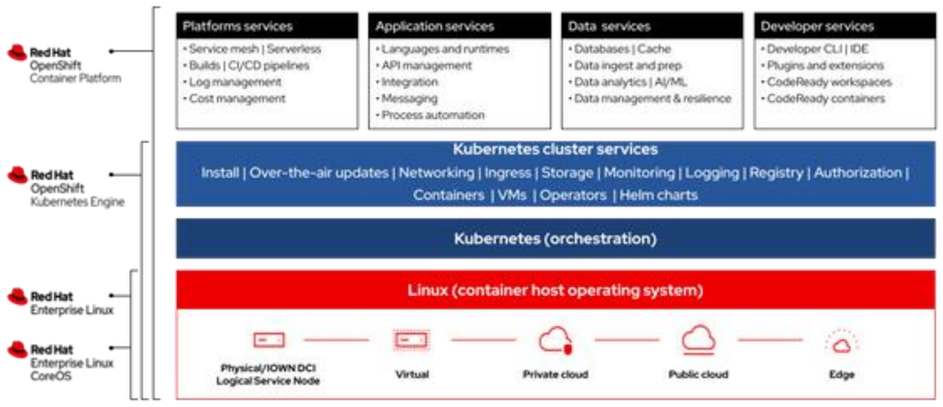

Red Hat OpenShift Container Platform
Red Hat OpenShift Kubernetes Engine
Red Hat Enterprise Linux
Red Hat Enterprise Linux CoreOS
Platforms services
• Service mesh | Serverless
• Builds | CI/CD pipelines
• Log management
• Cost management
Application services
• Languages and runtimes
• API management
• Integration
• Messaging
• Process automation
Data services
• Databases | Cache
• Data ingest and prep
• Data analytics | AI/ML
• Data management & resilience
Developer services
• Developer CLI | IDE
• Plugins and extensions
• CodeReady workspaces
• CodeReady containers
Kubernetes cluster services
Install | Over-the-air updates | Networking | Ingress | Storage | Monitoring | Logging | Registry | Authorization | Containers | VMs | Operators | Helm charts
Kubernetes (orchestration)
Linux (container host operating system)
Physical/IOWN DCI Logical Service Node
Virtual
Private cloud
Public cloud
Edge

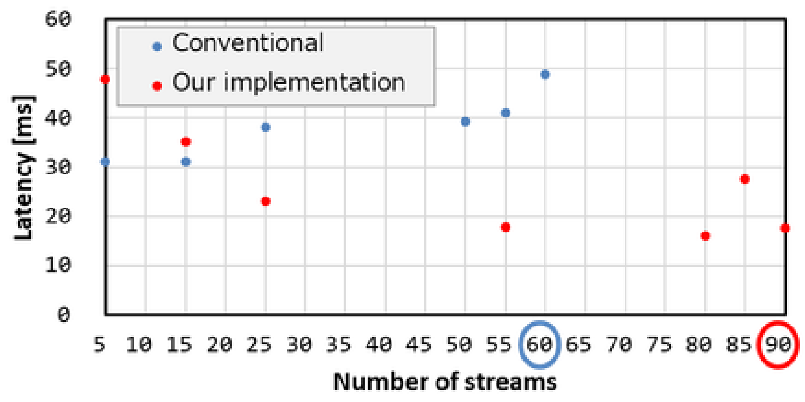
Figure 6-6: OpenShift Container Platform software stack

7. Performance Measurement and Evaluation of the PoC System

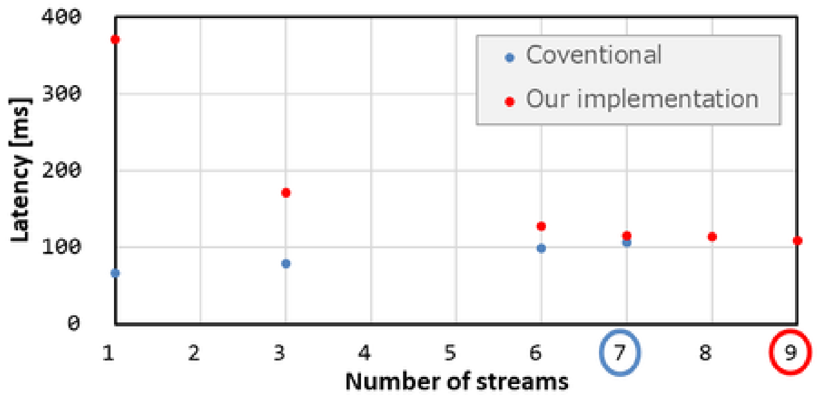
7.1. Performance measurement based on the expected benchmark

On the basis of the expected benchmark in the PoC Reference, latency, required system resources, throughput and energy efficiency (i.e., power consumption) were measured as evaluation metrics. The evaluations were for Option A, where an x86-server-based ingestion node was employed. Two different AI models were applied to the ingestion node. One is YOLOv3-tiny as a lightweight AI model. The other is YOLOv4-P6 as a heavy AI model.

Objective ID:	**1**
Description:	To demonstrate the improvement of latency (ref: 2.4.1 in [PoC Reference])
Procedure:	• Add start and end times onto each frame. • Start time is when the local aggregation node finishes receiving a frame • End time is when ingestion node completes obtaining inference results on the basis of the received frame
Lessons Learnt & Recommendations	Our hardware-accelerated pipeline improved the latency in certain cases (see Note and Figure 7-1). Hardware-acceleration technologies can contribute to latency-sensitive applications. Note: As described in section 6, multiple frames are batched in our implementation. In this PoC Report, the batching size was set to 10. Here, we call the number of batched frames as batching size. When the number of streams was less than the batching size, the latency was high because of inter-frame gaps. Therefore, the latency was improved when the number of streams increased than the batching size. Even as far as the number of streams was less than the batching size, the latency got improved as the number of streams increased. In general, a larger batching size improves throughput though latency increased. In contrast, a smaller batching size can be chosen to reduce latency though throughput is sacrificed.



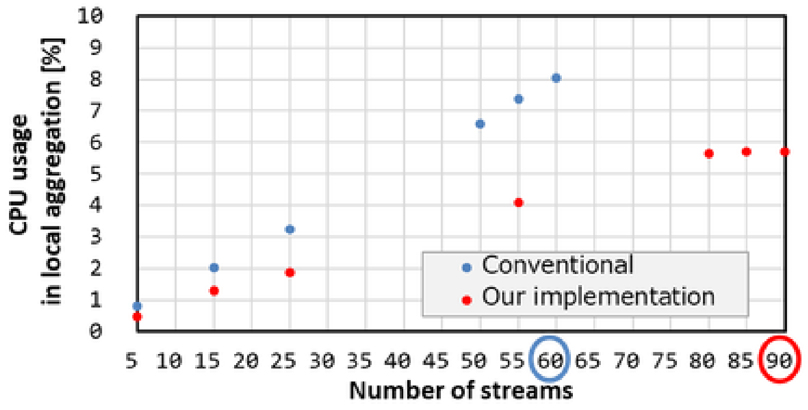
(a) Latency with YOLOv3-tiny



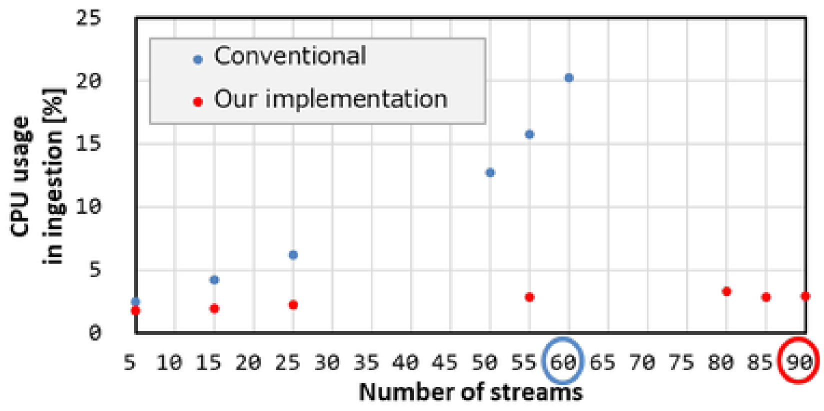
(b) Latency with YOLOv4-P6

Figure 7-1: Latency

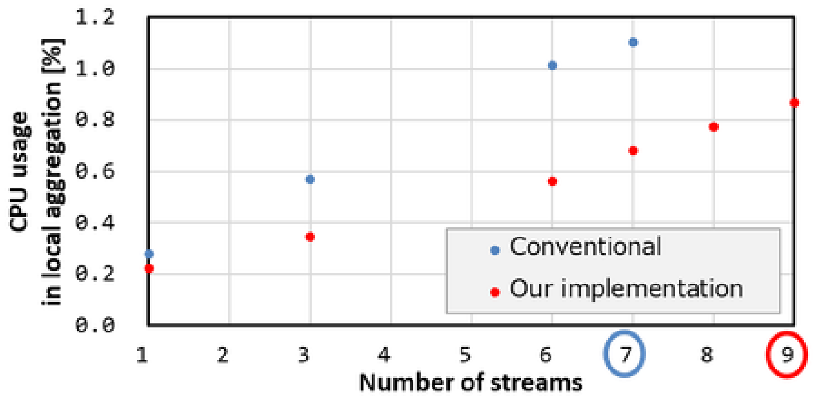
Objective ID:	**2**
Description:	To demonstrate the improvement of required system resources (ref: 2.4.2 in [PoC Reference])
Procedure:	• Introduce servers, hardware accelerators (smartNICs, a GPU and a converged accelerator), and network equipment (switches, transceivers, etc.) to physically configure our system. • Introduce a container platform to easily manage workloads with hardware-acceleration features. • Use "sar" command and "Kepler" (see Appendix A-3) to obtain CPU usage.
Lessons Learnt & Recommendations	Our hardware-accelerated pipeline for the performance evaluations was configured as illustrated in Figure 6-1. Our pipeline improved the CPU usage (see Figure 7-2). Unused CPU resources can be allocated to additional video streams or other applications. Required system resources for accommodating 1,000 streams are discussed in subsection 7.2.



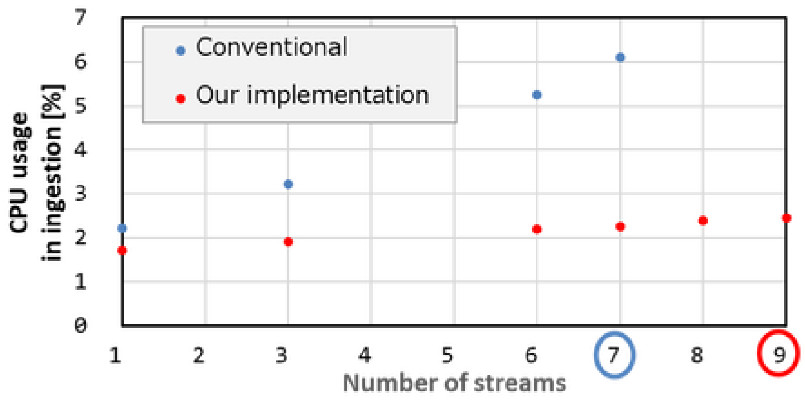
(a) CPU usage in local aggregation with YOLOv3-tiny



(b) CPU usage in ingestion with YOLOv3-tiny



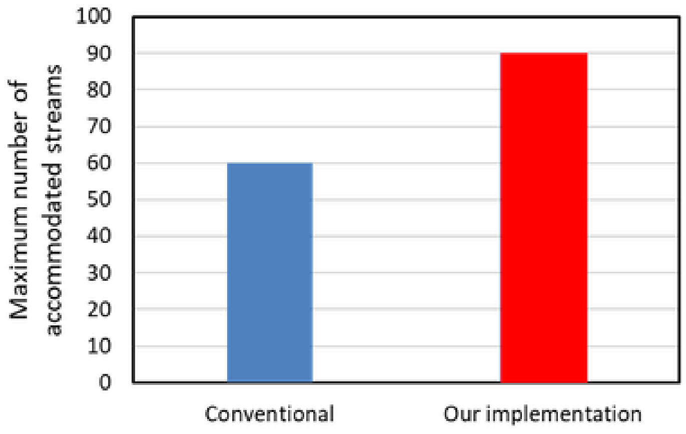
(c) CPU usage in local aggregation with YOLOv4-P6



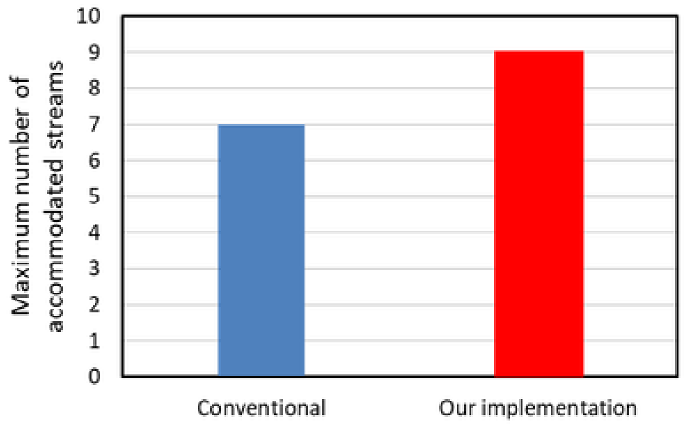
(d) CPU usage in ingestion with YOLOv4-P6

Figure 7-2: CPU usage

Objective ID:	**3**
Description:	To demonstrate the improvement in throughput, i.e., the maximum number of accommodated video streams (ref: 2.4.3 in [PoC Reference])
Procedure:	• Change the number of input streams. • Check the maximum number of streams where packet loss is not significantly increasing and the latency is less than and equal to 1 sec.
Lessons Learnt & Recommendations	Our hardware-accelerated pipeline improved the throughput (see Figure 7-3). This reduces the required number of nodes for the local aggregation and the ingestion. We confirmed that the efficiency of data transfer was not degraded because of packet loss in RDMA over Open APN.



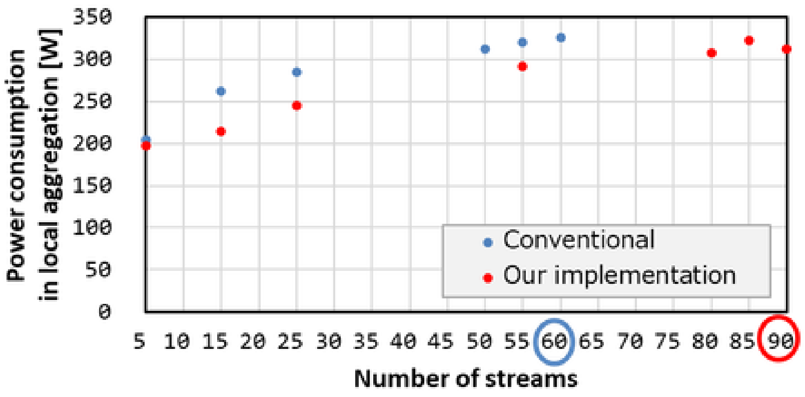
(a) Throughput with YOLOv3-tiny



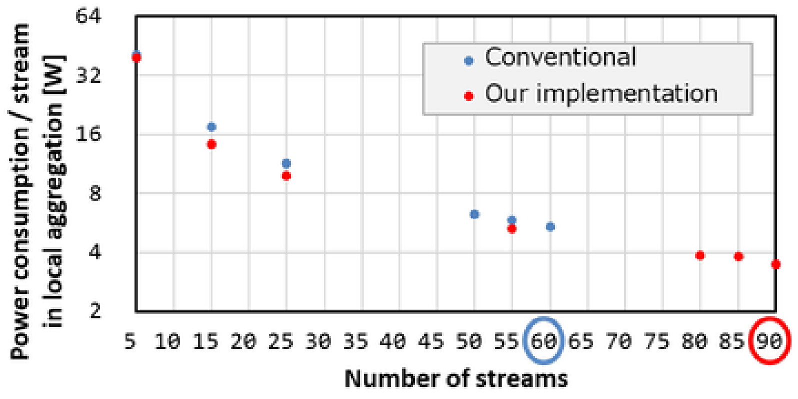
(b) Throughput with YOLOv4-P6

Figure 7-3: Maximum number of accommodated streams (i.e., throughput)

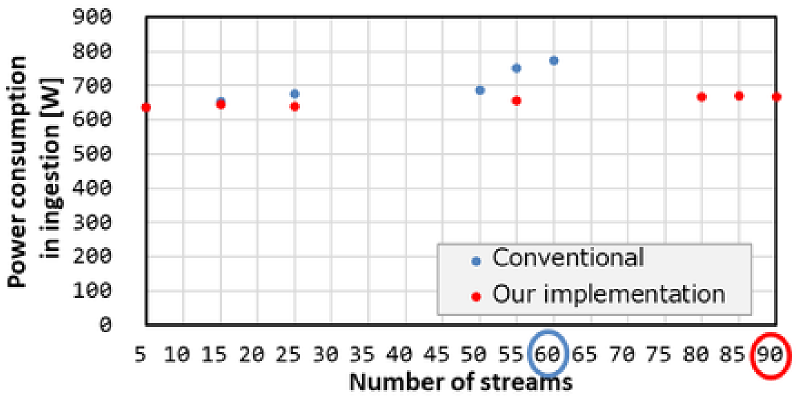
Objective ID:	**4**
Description:	To demonstrate the improvement of energy efficiency, i.e., power consumption (ref: 2.4.4 in [PoC Reference]) Note: Power consumption evaluated in this PoC Report was that of the local aggregation node and the ingestion node. The power consumption of the other nodes and networking infrastructure (e.g., switches) was not included.
Procedure:	Use “ipmitool“ command and “Kepler” (see Appendix A-3)
Lessons Learnt & Recommendations	Our hardware-accelerated pipeline improved the energy efficiency (see Figure 7-4). According to (a’), (b’), (c’), and (d’) of Figure 7-4, standby power accounted for the majority of power consumption when the number of accommodated streams was smaller. Let us see the results with YOLOv3-tiny in (a), (a’), (b), and (b’) in Figure 7-4. The power consumption per stream when the largest number of streams (i.e., 60 for conventional, 90 for our implementation) was accommodated was improved from 5.4 W to 3.5 W in the local aggregation, and from 12.9 W to 7.4 W in the ingestion node by applying our implementation. Next, look over the results with YOLOv4-P6 in (c), (c’), (d), and (d’) in Figure 7-4. The power consumption per stream when the largest number of streams (i.e., 7 for conventional, 9 for our implementation) was accommodated was improved from 30.4 W to 22.9 W in the local aggregation, and from 127.9 W to 93.7 W in the ingestion node by applying our implementation. Analyses from the evaluation results show that our hardware-accelerated pipeline improves the energy efficiency (see section 7.2).



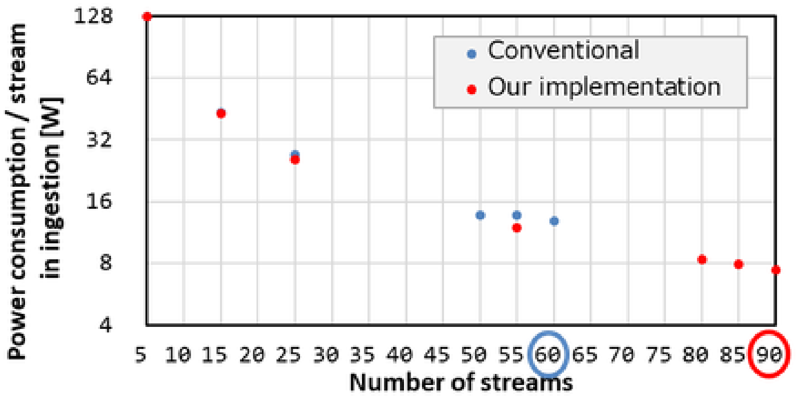
(a) Power consumption in local aggregation with YOLOv3-tiny



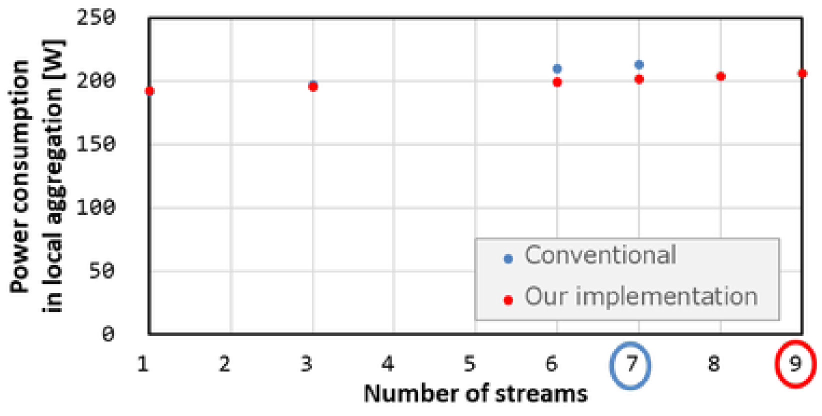
(a') Power consumption per stream in local aggregation with YOLOv3-tiny



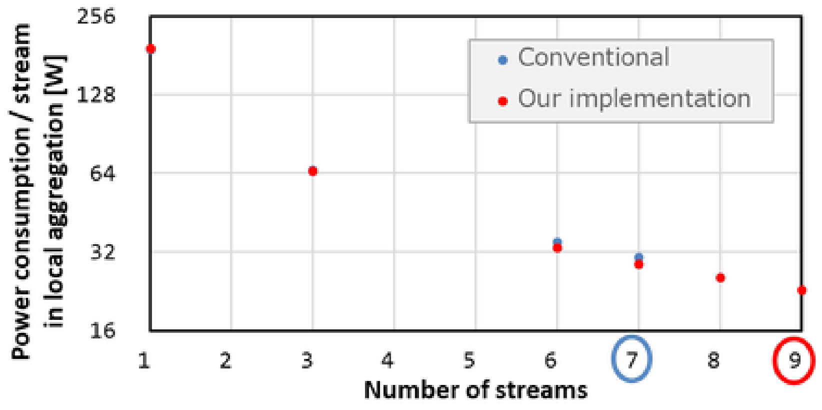
(b) Power consumption in ingestion with YOLOv3-tiny



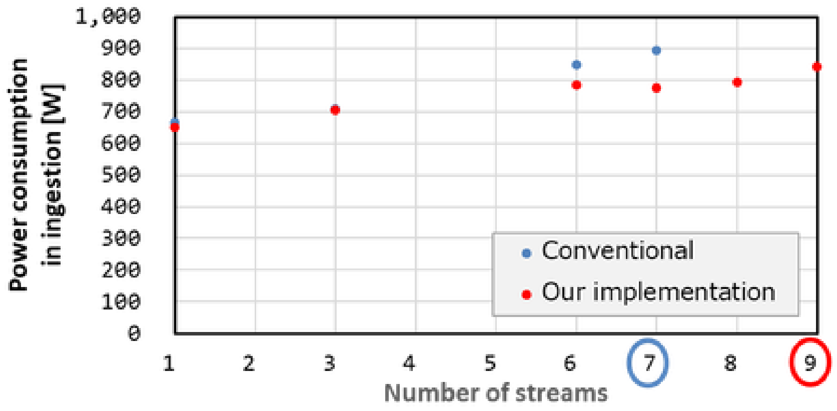
(b') Power consumption per stream in ingestion with YOLOv3-tiny



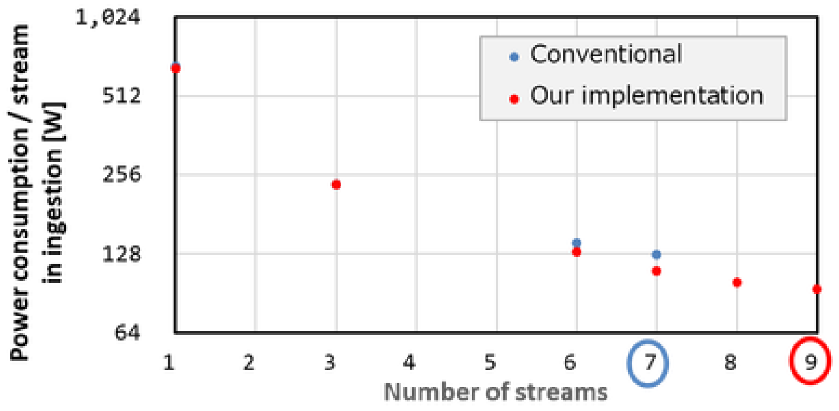
(c) Power consumption in local aggregation with YOLOv4–P6



(c') Power consumption per stream in local aggregation with YOLOv4–P6



(d) Power consumption in ingestion with YOLOv4–P6



(d') Power consumption per stream in ingestion with YOLOv4-P6

Figure 7-4: Power Consumption

7.2. Scalability consideration for 1,000 cameras accommodation

The previous subsection showed the evaluation results with only one GPU. Scaling up with multiple GPUs inside a node can be achieved by employing the latest GPU servers and PCIe bus extension technologies such as CDI. This scaling up increases the number of streams per node and relatively reduces the rate of standby power. In our implementation, CPU usage is improved and video data are transferred just one round trip between a NIC and a GPU in the ingestion node. Therefore, the number of GPUs inside a node can be increased without CPU and PCIe bus becoming bottlenecks.

On the basis of the above, we estimated the required system resources and power consumption for accommodating 1,000 streams as shown in Figure 7-5. To estimate of the required number of nodes, the maximum number of accommodated streams by one node was assumed to be the number of streams where its CPU usage reached 100% based on the linear extrapolation of evaluation results. Especially for the ingestion nodes, considering the limitation of the latest PCIe bus extension technology, the number of GPUs was assumed to be 22 or less. Based on that, the maximum number of accommodated streams by one ingestion node should be up to the product of 22 and the measured maximum number of accommodated streams by one GPU. To estimate the total power consumption, the power consumption of each node was calculated on the basis of the linear extrapolation of evaluation results. The required number of GPUs in each ingestion node was estimated based on the assumed number of accommodated streams in the ingestion node and measured maximum number of accommodated streams by one GPU. As illustrated in Figure 7-5, our implementation may improve the efficiency of resources and energy, leading to lower initial and operation costs.

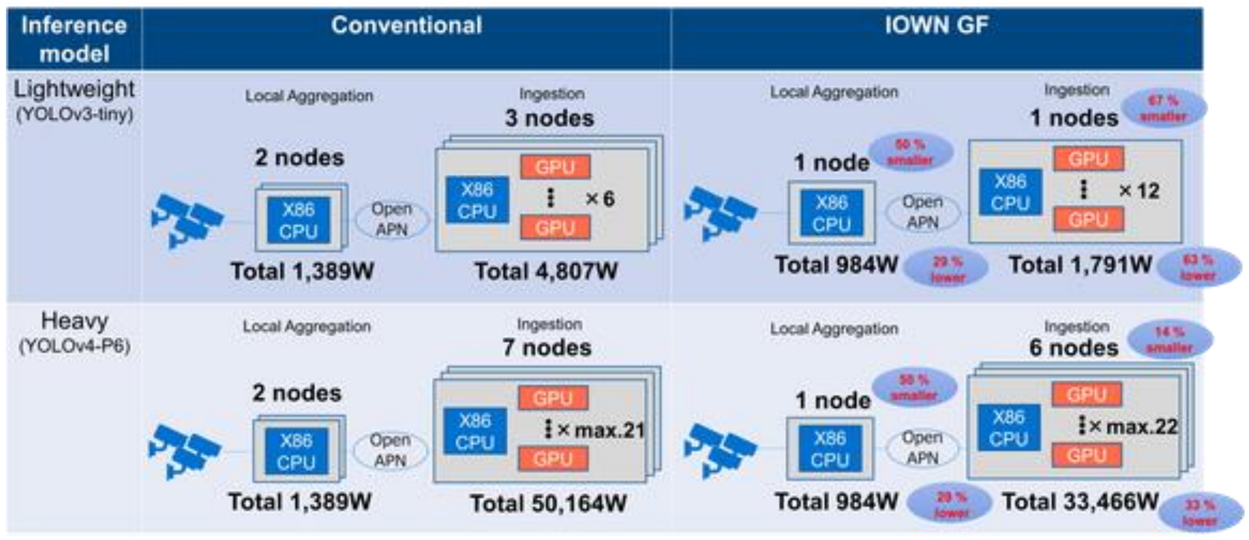


Figure 7-5: Resource reduction for accommodating 1,000 cameras (estimated from the evaluation results)

7.3. Comparative consideration in terms of the inference models

This subsection focuses on the difference between the results with YOLOv3-tiny (lighter-weight) and those with YOLOv4-P6 (heavier). In general, heavier inference models need more computation cost than lighter-weight inference models, so the evaluation metrics described in subsection 7.1 are better with lighter-weight inference models. The results in subsection 7.1 and the estimation in subsection 7.2 have this tendency as expected. Also, the effectiveness of hardware-acceleration becomes more pronounced as the inference models get lighter.

8. PoC's Contribution to IOWN GF

Contribution	WG/TF	Study Item / Work Item	Comments
Performance improvements	DCI/RIM-TF	n/a	Latency, required system resources, throughput, and energy efficiency have been improved. Evaluations with newer hardware-accelerated technologies are expected in the future.
Flexibility in geographical distribution of system resources	DCI/RIM-TF	n/a	RDMA over Open APN allows GPU resources to be aggregated in Remote Edge Clouds (i.e., nodes in Monitored Areas do not have to employ GPUs).
Efficiency improvement for accommodating 1,000 cameras, which is the target of this PoC	RIM-TF	n/a	According to analyses based on the evaluations, required system resources for accommodating 1,000 cameras can be reduced. See Figure 7-5.

9. PoC Suggested Action Items and Next Steps

9.1. Gaps identified in relevant standardization

9.1.1. Interoperability

Gap Identified	Forum/SDO	Affected WG/TF	Gap Details and Status
Interoperability between software and hardware	CNCF	DCI/RIM-TF	An appropriate combination of software and hardware for hardware-accelerated systems with workload management platforms is difficult to find.
Interoperability regarding RDMA	Linux, UCX, Open Fabric Alliance (OFA)	DPA in DCI-TF (DPA)	Major RDMA libraries depend somewhat on specific vendors. This dependency prevents interoperability between RDMA applications on RDMA NICs from different vendors.
GPU Direct RDMA or RDMA with GPU memory over DMA-BUF	Linux, UCX, OFA, CUDA	RIM-TF	Since peer-to-peer support via DMA-BUF with a focus on GPU support has been released in Linux 5.12+, DMA-BUF interop should be tested as described in Appendix A-2-2, in addition to NVIDIA's GPU Direct RDMA.

9.1.2. Strategy for creating, scaling and deleting LSNs

Gap Identified	Forum/SDO	Affected WG/TF	Gap details and Status
Flexibility of device configuration	CNCF (K8s), OPI (Open Programmable Infrastructure)	DCI/RIM-TF	An appropriate device configuration for an application is difficult to predict in terms of business growth, variety of workloads, progress of technologies and devices. DCI cluster aiming at a flexible computing architecture is a solution for that, but more concrete studies should be done for creating, scaling and deleting LSNs.
DPU LSN level composability and life cycle management, including security patch in DPU OS.	OPI	DCI/RIM-TF	Even the dedicated Linux OS runs inside DPU/IPU, the Linux Operating System software must support long term life cycle management as a standard distributed OS while keeping a Linux Open Source license.
x86 LSN scale Out/In, x86 LSN scale Up/Dn (Device PlugIn/Out) with DPU	CNCF, OFA, CUDA, OPI	DCI/RIM-TF	K8s Device Resource Allocation feature is missing, and interoperability with CDI system will be needed as proposed in Appendix A-2-2 for the next step of this PoC.

9.1.3. Desired solution for data transfer in IOWN GF Data Hub

Through conducting this PoC, we figured out that the Apache Arrow in-memory analytics solution is available in platform agnostic including GPU memory (CUDA) and Linux x86 CPU memory. In addition to Apache Kafka, IOWN GF Data Hub Task Force should consider new data transfer design using Apache Arrow on gRPC/UCX and DCI Task Force DPA team should consider including Apache Arrow Flight on gRPC/UCX as a PoC scenario of RDMA over Open APN.

9.2. PoC suggested action items

To achieve hardware-accelerated systems more efficiently and flexibly, IOWN GF is expected to help to fill the gaps described in subsection 9.1.

9.3. Next step?

This first phase of our PoC demonstrated the improvements in latency, required system resources, throughput and energy efficiency. Evaluations for Option B are planned in the next phase. In addition, following items will be considered.

- Evaluations integrated with other IOWN GF technologies such as Mobile Network and Data Hub
- Further efficiency improvements (e.g., workload optimization with an event-driven approach among local aggregation nodes and ingestion nodes)
- Joint discussions with other communities such as OPI for considering AI and security use cases with DPU-based LSNs
- Secure and efficient transport for RDMA
- Utilizing CDI for scaling the workloads

10. Conclusion

In this first phase of PoC focused on Sensor Data Aggregation and Ingestion (SDAI), we designed heterogeneous computing to evaluate offloading accelerator performance and obtained good results including improved end-to-end latency, throughput, lower CPU consumption, and energy efficiency. Through the evaluations with Open APN, we found that a CPS AM Security use case can be achieved on geographically distributed computing infrastructure.

In the middle of hardware evolution, we will continue to evaluate with the latest device to obtain even better results to meet the IOWN GF goal. Any members who can propose new PCIe devices are welcome to join this PoC.

References

- [PoC Reference] IOWN Global Forum, “PoC Reference: Reference Implementation Model for the Area Management Security Use Case,” 2022.
- [Operator] https://github.com/cncf/tag-app-delivery/blob/main/operator-whitepaper/v1/CNCF_Operator_WhitePaper_v1-0_20210715.pdf
- [Kepler] https://sustainable-computing.io/
- [Open APN] IOWN Global Forum, “Open All-Photonic Network Functional Architecture,” ver. 2.0, 2023.

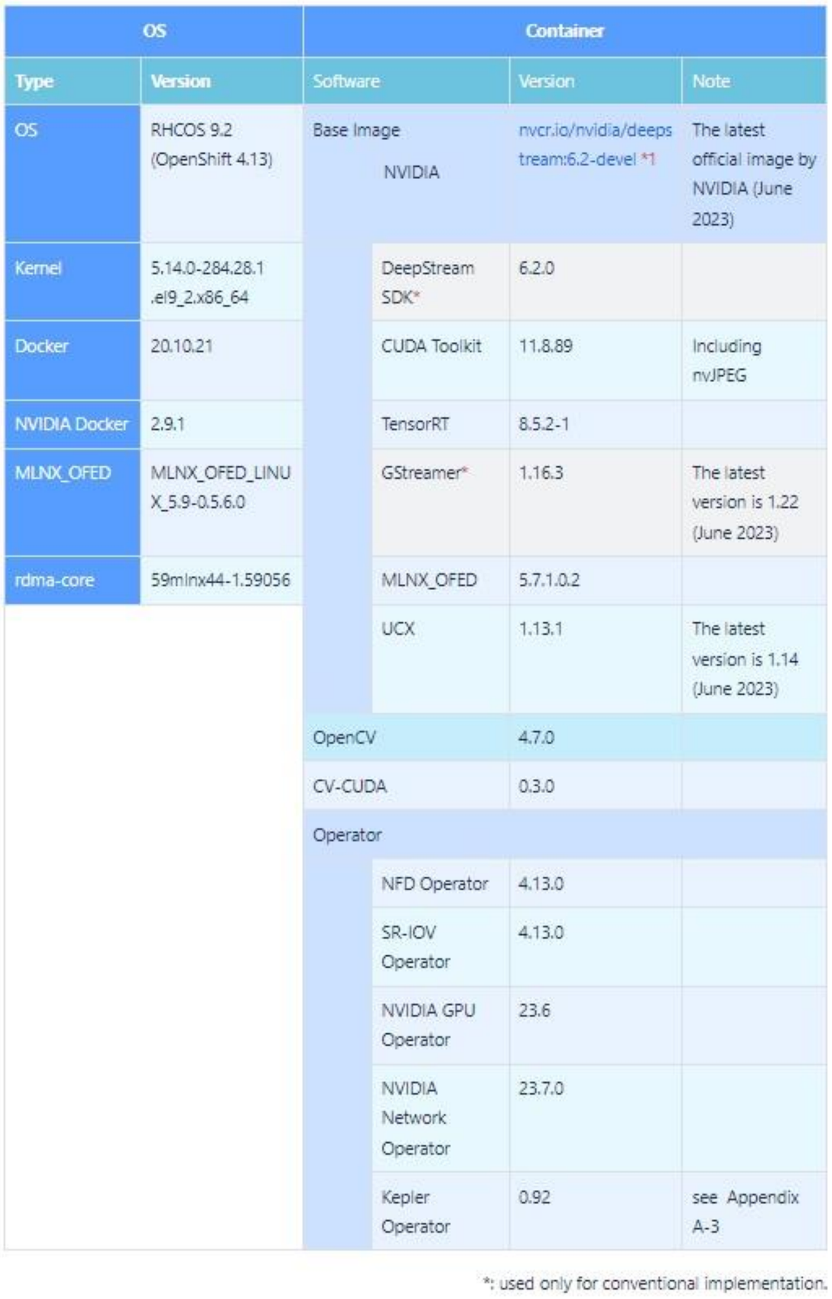
Document History

Version	Date	By	Description of Change
1.0	Jan. 10, 2024	Members listed in section 2	Initial version

Appendix:

A-1: Bill of Materials (BoM)

BoM for Option A in this PoC is described in Figure A-1-a. Note that one process is assigned for one stream in the conventional implementation.



OS	
Type	Version
OS	RHCOS 9.2 (OpenShift 4.13)
Kernel	5.14.0-284.28.1 .el9_2.x86_64
Docker	20.10.21
NVIDIA Docker	2.9.1
MLNX_OFED	MLNX_OFED_LINUX_5.9-0.5.6.0
rdma-core	59mlnx44-1.59056

Container			
Software		Version	Note
Base Image	NVIDIA	nvcr.io/nvidia/deepstream:6.2-devel *1	The latest official image by NVIDIA (June 2023)
	DeepStream SDK*	6.2.0	
	CUDA Toolkit	11.8.89	Including nvJPEG
	TensorRT	8.5.2-1	
	GStreamer*	1.16.3	The latest version is 1.22 (June 2023)
	MLNX_OFED	5.7.1.0.2	
	UCX	1.13.1	The latest version is 1.14 (June 2023)
OpenCV		4.7.0	
CV-CUDA		0.3.0	
Operator			
	NFD Operator	4.13.0	
	SR-IOV Operator	4.13.0	
	NVIDIA GPU Operator	23.6	
	NVIDIA Network Operator	23.7.0	
	Kepler Operator	0.92	see Appendix A-3

*: used only for conventional implementation.

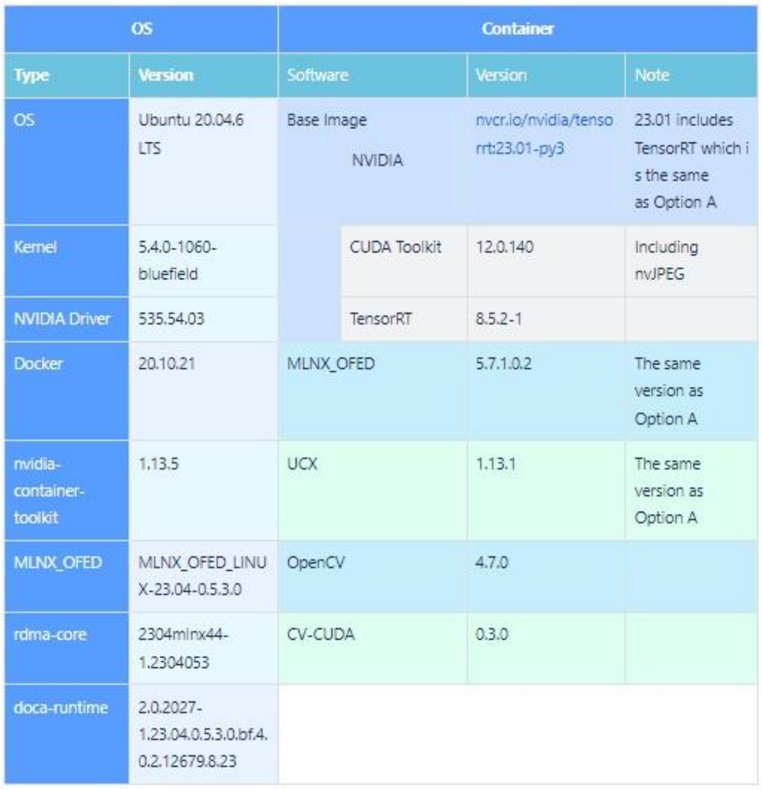
Figure A-1-a: BoM for Option A

A-2: Activities for Option B

Option B employs a converged accelerator-based ingestion node. In this PoC, A100X is utilized as the converged accelerator. Although performance evaluations for Option B have not been conducted in the first phase of this PoC, we have been preparing as follows.

A-2-1: Workload implementation on a converged accelerator with DPU vendor's default OS

By enabling NVIDIA BlueField-X mode to run a Linux OS on an ARM processor that are internally connected with both A100 GPU and ConnectX of smartNIC in NVIDIA A100X converged accelerator card, we found that the workload for CPS AM Security use case was successfully executed as an ingestion node. BoM for the current implementation in Option B is described in Figure A-2-1-a.



OS		Container		
Type	Version	Software	Version	Note
OS	Ubuntu 20.04.6 LTS	Base Image NVIDIA	nvcr.io/nvidia/tensorrt:23.01-py3	23.01 includes TensorRT which is the same as Option A
Kernel	5.4.0-1060-bluefield	CUDA Toolkit	12.0.140	Including nvJPEG
NVIDIA Driver	535.54.03	TensorRT	8.5.2-1	
Docker	20.10.21	MLNX_OFED	5.7.1.0.2	The same version as Option A
nvidia-container-toolkit	1.13.5	UCX	1.13.1	The same version as Option A
MLNX_OFED	MLNX_OFED_LINUX-23.04-0.5.3.0	OpenCV	4.7.0	
rdma-core	2304mlnx44-1.2304053	CV-CUDA	0.3.0	
doca-runtime	2.0.2027-1.23.04.0.5.3.0.bf4.0.2.12679.8.23			

Figure A-2-1-a: BoM for the current implementation in Option B

Note that NVIDIA BlueField default OS ver DOCA 2.0.2 (Ubuntu 20.04 base) with CUDA 12.0.140 was installed on the A100X at this stage for a feasibility check, while Red Hat is working in progress together with the NVIDIA development team to enable RHEL-based Kubernetes that will be called the ARM version of "MicroShift" or "Red Hat Device Edge".

The implementation status of Red Hat MicroShift open source base A100X converged accelerator card is described in the next subsection.

A-2-2: MicroShift on BlueField-X and OpenShift host

OpenShift on NVIDIA BlueField-2/3 was in completed at the Developer Preview stage and showcased at Red Hat summit 2023 for accelerating Kubernetes Hybrid Clouds with BlueField DPUs and OpenShift. This OpenShift DPU is a solution coexisting with x86 CPU base OpenShift host and ARM CPU base OpenShift DPU running Open Virtual Network (OVN) based applications. This was the initial plan for Option-B PoC, but we figured out that the OpenShift DPU does not easily fit on the current spec of NTT's inference engine without extra features because UCX API and GPU resources are needed. For supporting the API of NTT's inference engine, MicroShift development on the BlueField in the A100X converged accelerator, which is a new approach to enable minimal Kubernetes features on RHEL, is a work in progress.

The BlueField has two modes. One is Standard mode, which installs workloads on the x86 CPU host connecting with the BlueField DPU card such as the Option-A PoC described in section 7. Another is BlueField-X mode, which installs every workload in its own DPU. The goal of implementation for Option-B PoC is to enable NTT's inference engine on MicroShift as BlueField-X mode into the A100X converged accelerator card, which is the same combination of ConnectX smartNIC and A100 GPU implemented on the x86 OpenShift compute host for Option-A. GPU was successfully enabled with the NVIDIA GPU suite (nvidia-driver-deamonset, nvidia-container-toolkit, nvidia-device-plugin, etc) on MicroShift in the A100X as shown in Figure A-2-2-a. The deployment for NTT's inference engine will be tested on MicroShift as LSN-1B at Option-B PoC.

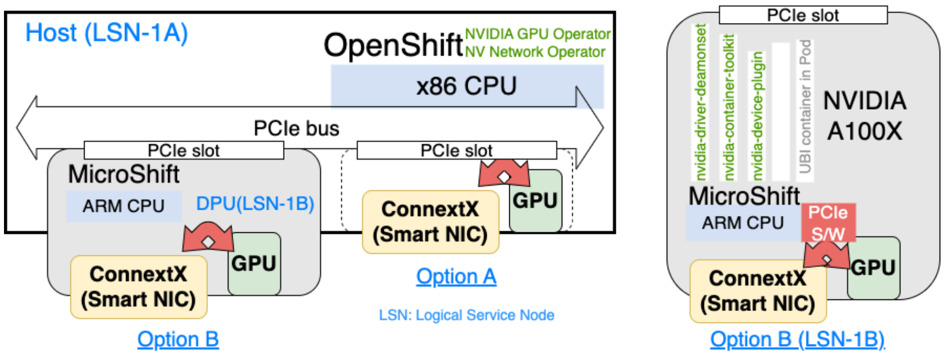


Figure A-2-2-a: Ingestion & AI Inference node in OpenShift (LSN-1A) or MicroShift (LSN-1B)

This MicroShift on the BlueField DPU card will be able to coexist with not only x86 OpenShift compute host but also x86 OpenStack compute host. This will be good for IOWN DCI, which is computing host platform agnostic, as long as the computing platform software can be composable with CPU and other PCIe cards. MicroShift DPU in LSN-1B (Option B) will be able to communicate with LSN-1A (Option A) running x86 OpenShift (or x86 OpenStack) via OVN on PCIe as needed. For the next step of this PoC project, between the CPU host board and PCIe bus expansion chassis, LSN-1A running IOWN Data Hub/ Intelligence application on x86 OpenShift along with LSN-1B running Ingestion with NTT's inference engine in the A100X converged accelerator will be expected for flexible workload deployment. In addition, the CDI feature should be adopted into the PoC system such as Figure A-2-2-b at the next step of PoC for a composability test as DCIaaS PoC. Since UCX and CUDA support DMA-Buff in addition to GPU Direct RDMA, RDMA interoperability test also will be considered at the next step of PoC.

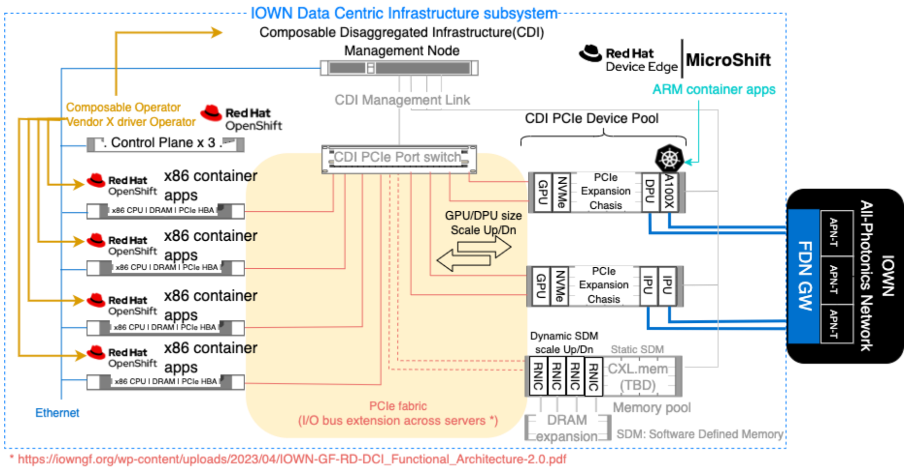


Figure A-2-2-b: Data Centric Infrastructure conceptual view

A-3: Resource Monitoring with Kepler

Kepler (Kubernetes-based Efficient Power Level Exporter) [Kepler] was introduced to monitor resource usage in this PoC. Kepler is an OSS solution for Kubernetes clusters that collects and visualizes data of various energy consumption metrics at machine/device/container levels. Figure A-3-a shows captured images of Kepler during this PoC execution.

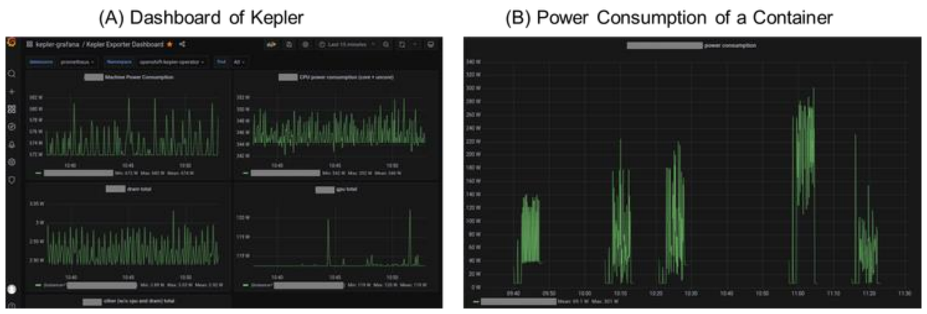

(A) Dashboard of Kepler
(B) Power Consumption of a Container
Machine Power Consumption
CPU power consumption (core + uncore)
dram total
gpu total
power consumption

Figure A-3-a: Resource monitoring with Kepler

A-4: Open APN Setup

Setup of Open APN for our demonstration described in section 5 is as follows.

- Optical transmission/reception specification
  - W 100-200G 31.6 Gbaud of Open ROADM MSA Optical Specification Version 5.1 [Open APN]
  - Bandwidth: 100 Gbps
- Network configuration is illustrated in Figure A-4-a.

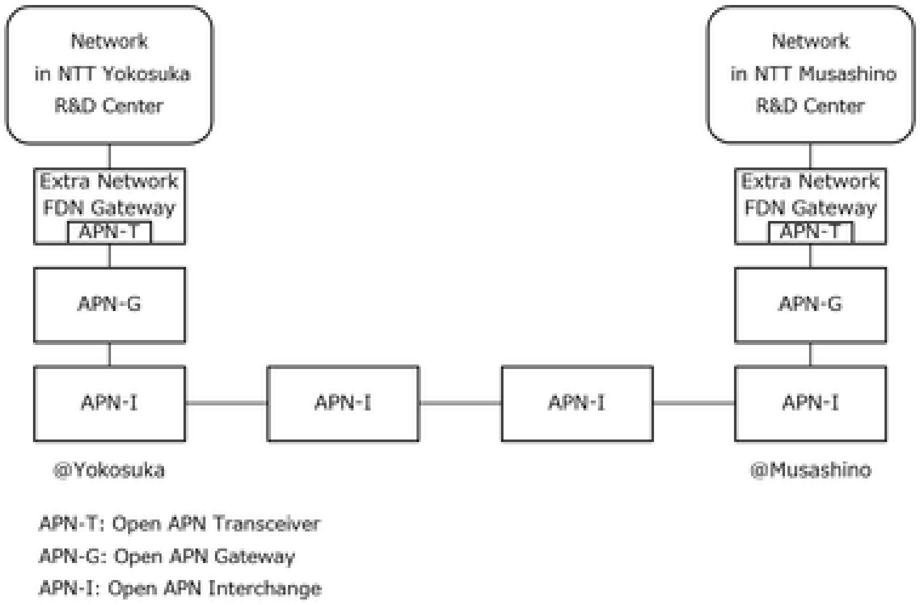

Network in NTT Yokosuka R&D Center
Network in NTT Musashino R&D Center
Extra Network FDN Gateway
APN-T
Extra Network FDN Gateway
APN-T
APN-G
APN-G
APN-I
APN-I
APN-I
APN-I
@Yokosuka
@Musashino
APN-T: Open APN Transceiver
APN-G: Open APN Gateway
APN-I: Open APN Interchange

Figure A-4-a: Configuration of Open APN setup

A-5: Requirements and Expectations in the PoC Reference

Features	Requirements and Expectations in the PoC Reference	Section in the PoC Reference	Status in this Report
Sensor device	• Video cameras can be substituted for video delivery servers. • Frame size is Full HD or higher, and frame rate is 15 fps or higher.	2.2.1, 2.2.4	Met (see 6.2)
Local aggregation	• Accepting and aggregating video streams from one or more sensor devices. • Hardware-based data transmission of aggregated video image streams to the peered ingestion node	2.2.2, 2.2.4, 2,2,5	Met (see 6.3)
Ingestion	• Hardware-based data reception of video image streams from one or more peered local aggregation nodes • Data replication and delivery of accepted video image streams to multiple data consumers • Video decoding and CNN-based image recognition to produce labeled objects in a real-time manner • Data-plane accelerated by hardware-based data transfer • Utilization of heterogeneous accelerators optimal for each portion of the workload	2.2.3, 2.2.5	Met (see 6.4)

Communication between sensor devices and local aggregation	• Network type is not specified but PoC Teams should take the following points into account. ○ Example network types are Ethernet, SDI (Serial Digital Interface), etc. ○ To allow the heterogeneous networking environments of the monitored area, the AM-S RIM document assumes that sensor devices are connected via the best available network for the monitored area. • Communication distance is not specified but PoC Teams should take the following points into account. ○ In a deployment option in the AM-S RIM document, the communication distance can be up to 1km. ○ Evaluation in a local experiment environment is acceptable since the sensor devices are connected with local aggregation nodes in the same monitored area. • Protocol is not specified but PoC Teams should take the following points into account. ○ Example communication protocols are RTP, HTTP, QUIC, etc.	2.2.4	Met (see 6.2 and 6.3)

Communication between local aggregation and ingestion	• Network type is not specified but PoC Teams should take the following point into account. ○ It is recommended to apply Open APN, extra network gateway, and DCI gateway to the underlying network. The extra network gateway and DCI gateway should support Flexible bridging Service. • Communication distance is not specified but PoC Teams should take the following points into account. ○ In a deployment option in the AM-S RIM document, the communication distance can be up to 337 km. ○ It is recommended to evaluate the changes in performance with respect to changes in distance. The communication distance may be emulated by network emulators. ○ To evaluate the basic performance of the SDAI, evaluation in a local experiment environment is also acceptable. • RoCEv2 (RDMA over Converged Ethernet v2) (UDP/IP/Ethernet) should be applied, at least, to the data transfer of video image streams. ○ The AM-S RIM document recommends RDMA as a communication protocol for inter-node interconnect since its protocol stack can be fully offloaded to hardware on the NIC.	2.2.5	Met (see 5, 6.3 and 6.4)
Latency	• The mandatory E2E response time of the AM Security UC is expected to be less than 1 sec. Therefore, the latency of the SDAI should be kept around hundreds of milliseconds.	2.2.6	Met (see 7)

Scalability	• The PoC Team shall choose an implementation strategy that scales well and choose an implementation size for the PoC that is sufficiently large to allow extrapolation of the required system resources and the system cost to the size of an actual production system.	2.2.7	Met (see 7)
Advanced optional features	• PoC Teams are encouraged to implement additional features described in the AM-S RIM document and demonstrate them. Examples of such features include: ○ Workload optimization with an event-driven approach among local aggregation nodes and ingestion nodes. ○ Utilization of heterogeneous accelerators with accelerator pooling and auto-scaling. ○ Advanced AI analysis employing sensor fusion techniques with multiple types of sensor devices.	2.3	For Option A, container platform (i.e., OpenShift) is introduced for making it easier to manage life cycles of workloads with the hardware-based data transfer feature. See 6.6, 9.3 and A-2.
Expected benchmarks	• The key KPIs for the benchmark defined in this PoC Reference are latency, system resources and configuration, throughput, and energy efficiency.	2.4	Met (see 7)
Other Consideratio ns	• The PoC Reports should include considerations regarding the following items: ○ Qualitative and quantitative analysis by comparing IOWN technologies with existing technologies. ○ Scalability.	2.5	Met (see 6, 7 and 9.1.2)