IOWN GLOBAL FORUM™

Recognized PoC Report – Significant Step Forward

Sensor Data Aggregation and Ingestion PoC Phase 2

MAY 2026

1. Introduction

Toward the CPS Area Management Security Use Case (CPS AM UC) in the IOWN Global Forum (IOWN GF), the energy efficiency of computing infrastructure needs to be drastically improved for real-time video inference from a massive number of surveillance cameras.

As a solution to this problem, our PoC Phase 1 employed the IOWN GF technologies such as Open All-Photonic Network (APN) and Data-Centric Infrastructure (DCI), especially with hardware-acceleration technologies. Our hardware-accelerated pipeline improved the performance and energy efficiency in data-plane processing of video inference. Evaluation results showed that latency and power consumption were reduced by approximately 40% compared with a conventional CPU-centric pipeline. You can see detailed results in our PoC Report Phase 1 [PoC Report Phase 1].

This report describes our PoC Phase 2, which builds on our PoC Phase 1 to enhance security and efficiency. We implement and evaluate two features. The first one is a hardware-accelerated secure data transfer. Offloading encryption and decryption for Remote Direct Memory Access (RDMA) from the CPU to SmartNICs is expected to enhance security without overhead, whereas the conventional CPU-based encryption and decryption may introduce overhead. The second one is offloading even control-plane processing from the CPU to SmartNICs and GPUs as much as possible to improve efficiency compared with offloading only data-plane processing as in our PoC Phase 1. For flexible and efficient scaling, applying Composable Disaggregated Infrastructure (CDI) to the CPS AM UC is also investigated, though it is not implemented or evaluated. We confirm that this report satisfies the requirement in the PoC Reference of CPS AM UC [PoC Reference].

2. PoC Project Completion Status

- Overall PoC Project Completion Status: Completed
- PoC Stage Completion Status:
    - Phase 1: Completed in 2024
    - Phase 2: Completed (this report)

3. PoC Project Participants

- PoC Project Name: CPS AM Security PoC-1: Sensor Data Aggregation and Ingestion (SDAI)
- PoC Teams:

Company	Name
1Finity	Yuji Tazaki, Kazunori Sakumoto
NTT	Rintaro Harada, Kenji Tanaka, Kazuo Ninokata, Ryosuke Kurebayashi
NVIDIA	Hideaki Tagami, Takashi Noda, Joey Hashimoto
Red Hat	Hidetsugu Sugiyama

4. Confirmation of PoC Demonstration

Our PoC Phase 2 was demonstrated in NTT Musashino R&D Center, Tokyo, Japan. Although the Open APN was not employed, it was not necessary considering the objective of our PoC Phase 2. Note that our PoC Report Phase 1 has already demonstrated that our hardware-accelerated pipeline over the Open APN can improve performance and energy efficiency compared with the conventional CPU-centric pipeline.

5. PoC Goals Status Report

- PoC Project Goal: To improve energy efficiency in SDAI systems
- Goal Status: Met in 2026

6. Supported Use Case

The PoC Reference [PoC Reference] of the CPS AM UC defined the following PoC scopes to include critical parts of the RIM as illustrated in Figure 6-1 (PoC-1 for the SDAI, and PoC-2 for IOWN Data Hub and Live 4D Map). This PoC was conducted in the scope of PoC-1. The SDAI system aggregates sensor data (e.g., video data, LiDAR data) in a Monitored Area and analyzes the sensor data in a Regional Edge Cloud. The system overview is illustrated in Figure 6-2. Note that our PoC utilized only video data as sensor data.

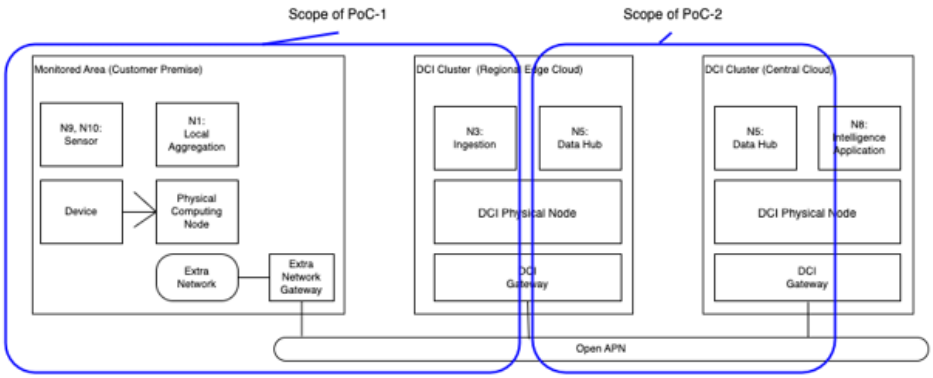


Figure 6-1: The scope of this PoC (PoC-1)

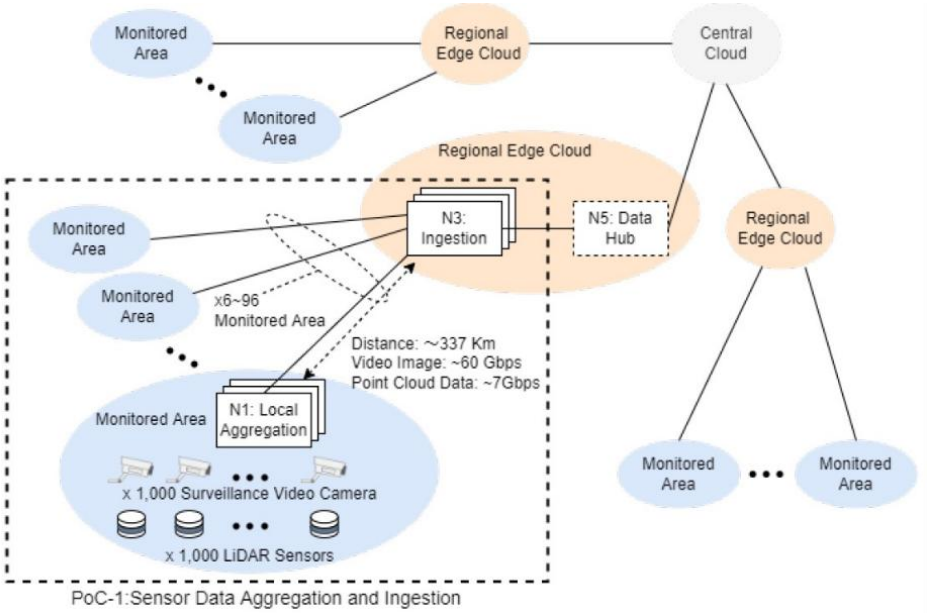


Figure 6-2: Overview of the Sensor Data Aggregation and Ingestion (SDAI) system

7. PoC System Configuration and Implementation

7.1. Overall View of Implemented PoC System Configuration

Our PoC system is composed of a video server substituting for cameras, a Local Aggregation node, an Ingestion node, and a Data Hub node as described in Appendix A-1. The SmartNICs used for the connections among the Local Aggregation node, the Ingestion node, and the Data Hub node can enable secure RDMA, i.e., RoCEv2 over IPSec/MACSec Full Offload, where the processing of IPSec/MACSec is offloaded from the CPU to the SmartNICs. This PoC focuses only on RoCEv2 over MACSec Full Offload. The Ingestion node employs a GPU for video inference.

7.2. 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).

7.3. Local Aggregation

A Local Aggregation node receives the video data from the video server and then sends them to an Ingestion node. As shown in Figure 7-1, the conventional pipeline employs simple RTP as a conventional protocol to receive and send the video data. Both data plane and control plane are affected by kernel overhead. On the other hand, our hardware-accelerated pipeline eliminates kernel overhead in data plane. As shown in Figure 7-2, the received video data are directly put onto main memory without overhead by NVIDIA Rivermax technology. Also, Unified Communication X (UCX) is utilized for GPUDirect 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.

As a new feature of our PoC Phase 2, secure RDMA (RoCEv2 over MACSec Full Offload) can be enabled in the connection with the Ingestion node.

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) could be performed inside the Local Aggregation node in a real deployment.

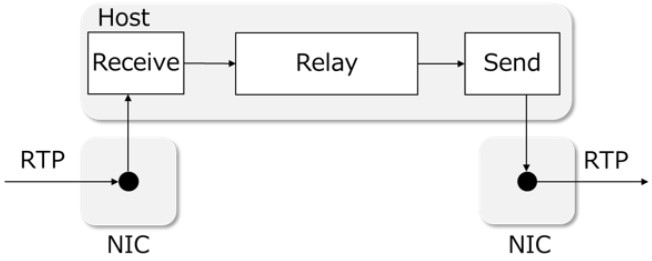


Figure 7-1: Conventional pipeline for Local Aggregation

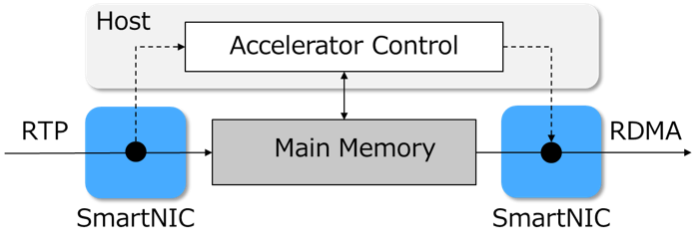


Figure 7-2: Our hardware-accelerated pipeline for Local Aggregation

7.4. Ingestion

An Ingestion node receives and decodes the video data from the Local Aggregation node, executes inference, and then sends the inference results as well as the video data to a Data Hub node. As shown in Figure 7-3, the conventional pipeline is CPU-centric, so both data plane and control plane are affected by kernel overhead. In contrast, our hardware-accelerated pipeline achieves GPU-centric data-plane processing without kernel overhead as shown in Figure 7-4 and Table 7-1. 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.

There are two new features in our PoC Phase 2 regarding the Ingestion node. The first one is secure RDMA (RoCEv2 over MACSec Full Offload) in the connection with the Local Aggregation node and the Data Hub node. The second one is applying NVIDIA CUDA Graphs and GPUNetIO for further efficiency as illustrated in Figure 7-5. CUDA Graphs integrates pre-processing, inference, post-processing, and control-plane processing for decoding into a single function. GPUNetIO offloads GPUDirect function both in data plane and control plane. It is expected to further improve efficiency compared with GPUDirect RDMA that offloads only data plane as in Figure 7-4. With the combination of these two technologies, even the control-plane processing can be offloaded from the CPU to SmartNICs and GPUs, though decoding still requires interactions with the CPU.

Note that detailed specifications of an RDMA-based interface for the Data Hub node may be studied with the PoC-2 team.

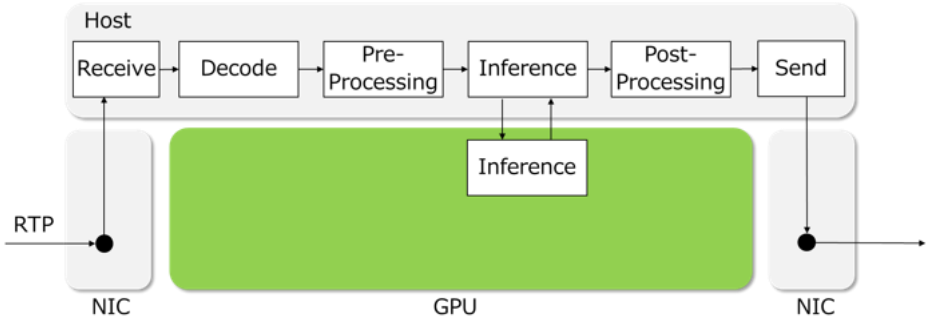


Figure 7-3: Conventional pipeline for Ingestion

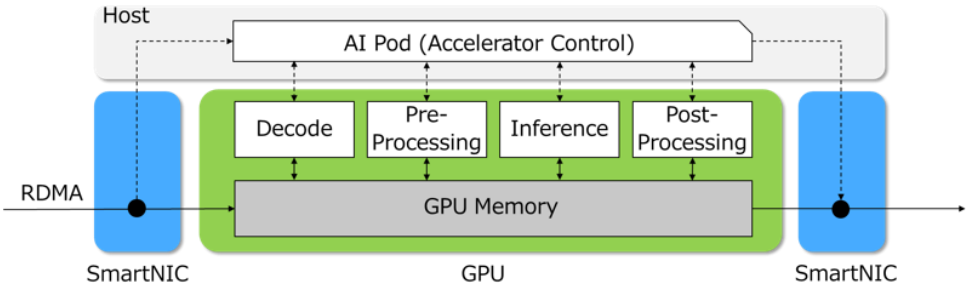


Figure 7-4: Our hardware-accelerated pipeline for Ingestion (without CUDA Graphs and GPUNetIO)

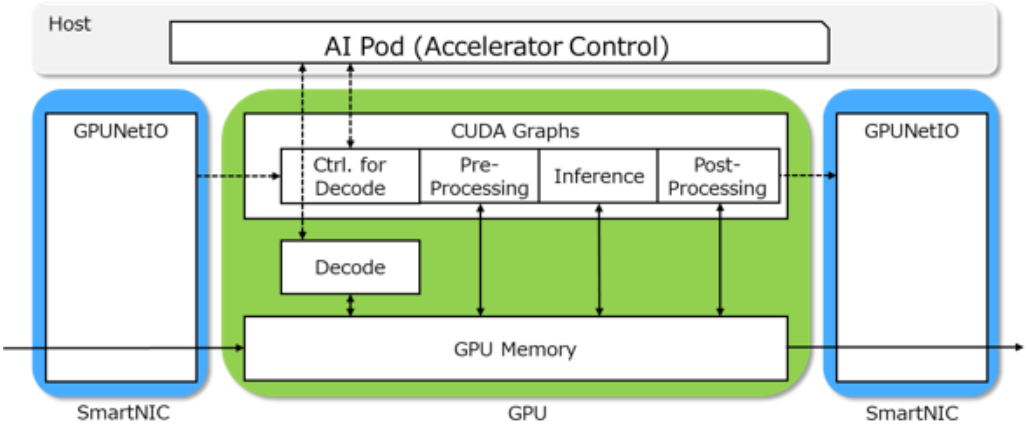


Figure 7-5: Our hardware-accelerated pipeline for Ingestion (with CUDA Graphs and GPUNetIO)

Table 7-1: Hardware-acceleration technologies for the Ingestion node in our pipeline

Hardware-acceleration technologies	Purposes
UCX	GPUDirect 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 GPU
TensorRT (Note that it is used also in the conventional pipeline.)	Accelerate inference on GPU by optimizing inference models
CUDA	Post-processing (e.g, picking out detected objects as bounding boxes) on GPU
CUDA Graphs (optional)	Integration of pre-processing, inference, and post-processing in a single function
GPUNetIO (optional)	Offload of GPUDirect function both in data plane and control plane

7.5. Data Hub

A 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 hardware-accelerated pipeline utilizes UCX for RDMA-based reception. As a new feature of our PoC Phase 2, secure RDMA (RoCEv2 over MACSec Full Offload) can be enabled in the connection with the Ingestion node. Since details of the Data Hub node are outside the scope of this PoC, it is implemented in a pseudo manner.

8. Performance Evaluations and Considerations

8.1. Performance Evaluations based on the Expected Benchmark

Same as our PoC Report Phase 1, latency, required system resources, throughput (i.e., the maximum number of accommodated video streams), and energy efficiency (i.e., power consumption) were measured as evaluation metrics on the basis of the expected benchmark in the PoC Reference. Our hardware-accelerated pipeline without CUDA Graphs and GPUNetIO had two RDMA transfer options: Write and Read. Our pipeline with CUDA Graphs and GPUNetIO had only Write-based RDMA transfer. YOLOv3-tiny was used as an inference model. Evaluation results were plotted in the following graphs only if the number of streams on the horizontal axis was able to be accommodated. Regarding the evaluation results for secure RDMA illustrated in Figure 8-1, 8-2, and 8-3, the plots of "Ours (Write, NoSec)" in orange and the plots of "Ours (Write, MACSec)" in dark orange may not be seen. This was because these plots were overlapped with the plots of "Ours (Read, NoSec)" in green and "Ours (Read, MACSec)" in dark green.

Objective ID:	1: Latency and Throughput (Secure RDMA)
Description:	To demonstrate the affects on latency and throughput when MACSec was added (ref: 2.4.1 and 2.4.3 in [PoC Reference])

Procedure:	• Latency
	  ○ 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
	• Throughput
	  ○ 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	In the conventional pipeline, adding MACSec resulted in smaller throughput and larger latency at larger number of accommodated video streams. In contrast, our hardware-accelerated pipeline prevented degradation of latency and throughput even when MACSec was added. Offloading MACSec from CPU to SmartNICs is a good approach for latency-sensitive applications that require secure RDMA transfers.

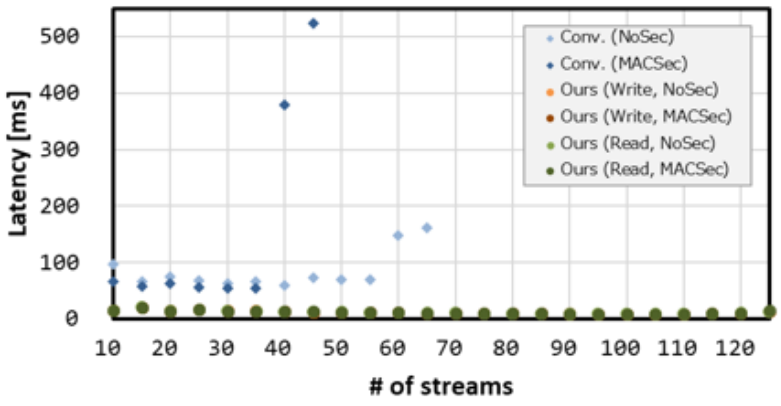
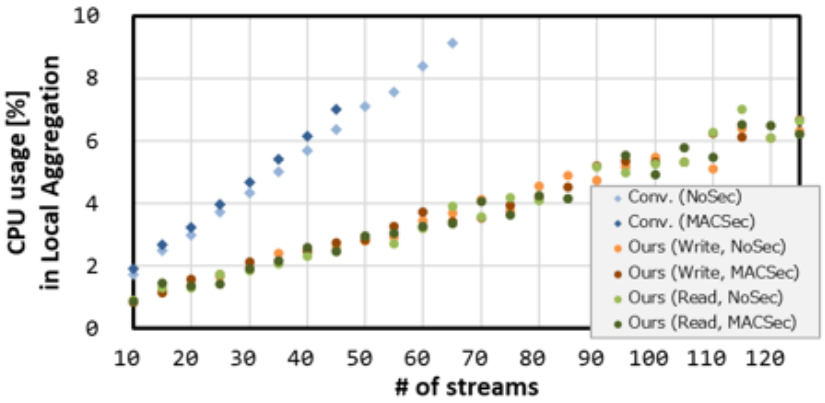
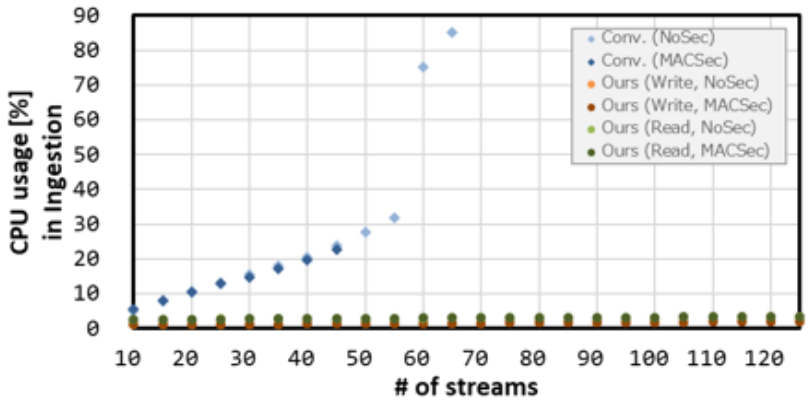


Figure 8-1: Latency and throughput

Objective ID:	2: Required System Resources (Secure RDMA)
Description:	To demonstrate the affects on required system resources when MACSec was added (ref: 2.4.2 in [PoC Reference])
Procedure:	• Introduce servers, hardware accelerators (i.e., smartNICs and GPUs), and network equipment (switches, transceivers, etc.) to configure our system as illustrated in Figure A-1. • Use "sar" command to obtain CPU usage.
Lessons Learnt & Recommendations	In the conventional pipeline, adding MACSec resulted in slight increases in CPU usage at the Local Aggregation node. In contrast, our hardware-accelerated pipeline prevented CPU usage from increasing even when MACSec was added. Unused CPU resources can be allocated to additional processing such as processing for other applications.



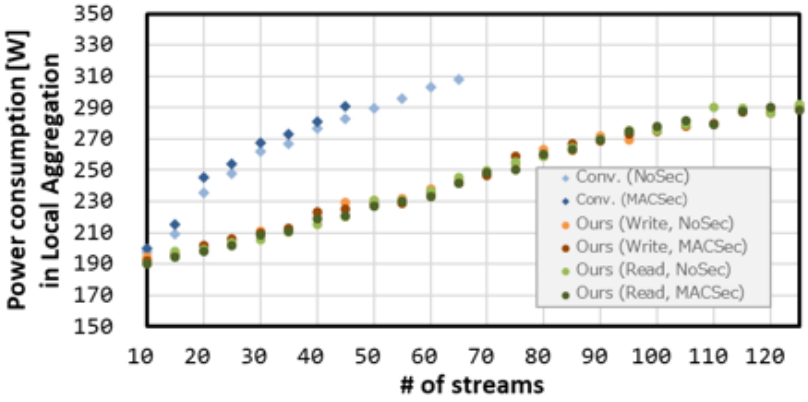
(a) CPU usage in Local Aggregation



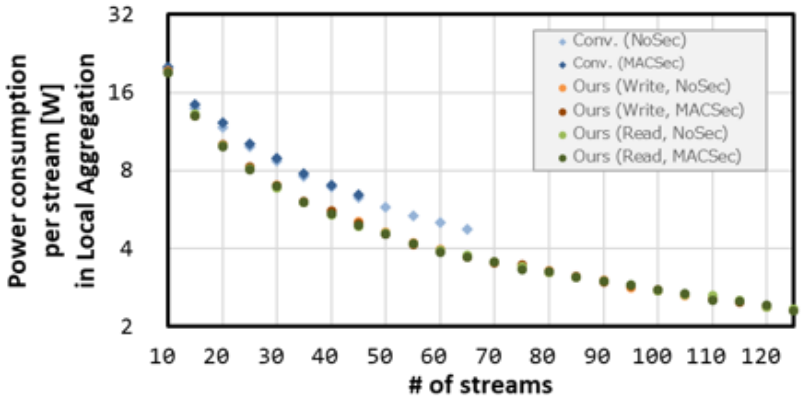
(b) CPU usage in Ingestion

Figure 8-2: CPU usage

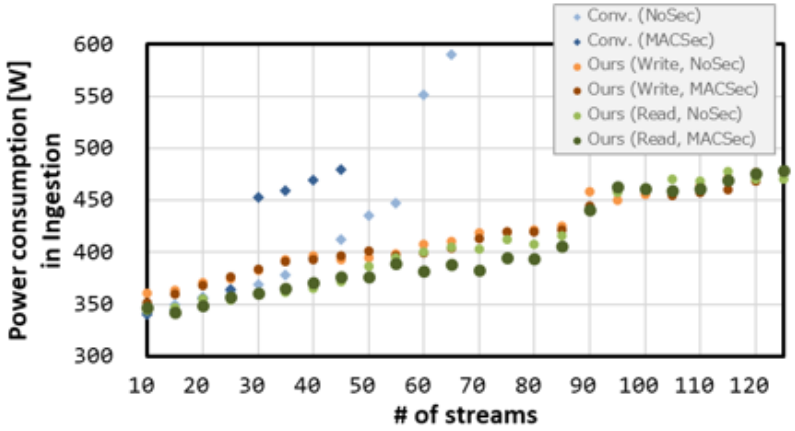
Objective ID:	3: Energy Efficiency (Secure RDMA)
Description:	To demonstrate the affects on energy efficiency when MACSec was added (ref: 2.4.4 in [PoC Reference]) Note: Power consumption evaluated in this PoC Report was that of the Local Aggregation node and the Ingestion node. The power consumption of the other nodes and networking infrastructure (e.g., switches) was not included.
Procedure:	Use “ipmitool” command.
Lessons Learnt & Recommendations	In the conventional pipeline, adding MACSec led to increased power consumption especially at the Ingestion node. In contrast, our hardware-accelerated pipeline maintained power consumption even when MACSec was added. Through Objective ID 1-3, we confirmed that it was possible to add MACsec to our hardware-accelerated pipeline without overhead from the perspective of performance and energy efficiency.



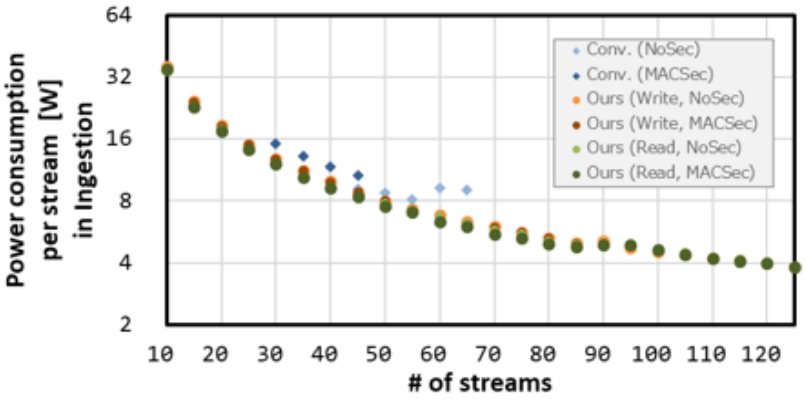
(a) Power consumption in Local Aggregation



(a') Power consumption per stream in Local Aggregation



(b) Power consumption in Ingestion



(b') Power consumption per stream in Ingestion

Figure 8–3: Power consumption

Objective ID:	4: Latency and Throughput (CUDA Graphs and GPUNetIO)
Description:	To demonstrate the improvements of latency and throughput by applying CUDA Graphs and GPUNetIO (ref: 2.4.1 and 2.4.3 in [PoC Reference])
Procedure:	Same as in Objective ID 1
Lessons Learnt & Recommendations	For our hardware-accelerated pipeline without CUDA Graphs and GPUNetIO, latency was measured from the time when the Local Aggregation node finished receiving the whole data of a frame to the time when the Ingestion node completed to produce labeled objects from the frame as described in [PoC Reference]. On the other hand, for our hardware-accelerated pipeline with CUDA Graphs and GPUNetoIO, the end time of latency measurement was when the Data Hub node finished the reception from the Ingestion node due to the implementation of GPUNetIO. The values of latency for our pipieline with CUDA Graphs and GPUNetIO in the Figure 8-4 included the duration of transmission between the Ingestion node and the Data Hub node, which was not included for our pipieline without CUDA Graphs and GPUNetIO. What we could say when we looked at Figure 8-4 considering the above was applying CUDA Graphs and GPUNetIO improved the latency at 120 or more video streams. Throughput was also improved from 125 to 140. The larger throughput led to improved energy efficiency because the standby power of nodes can be divided by larger number of accommodated video streams.

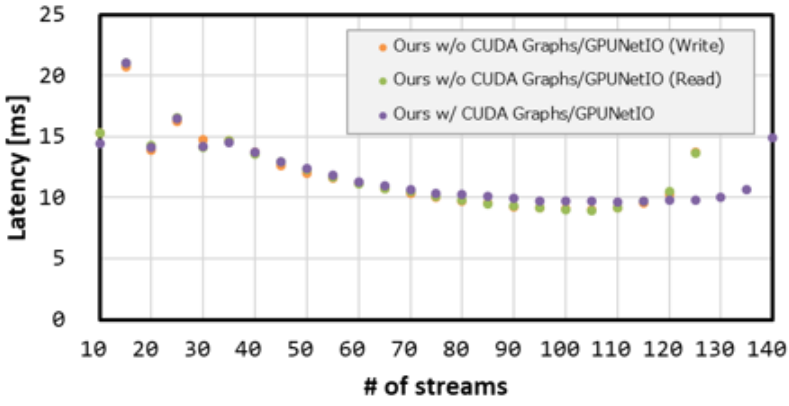
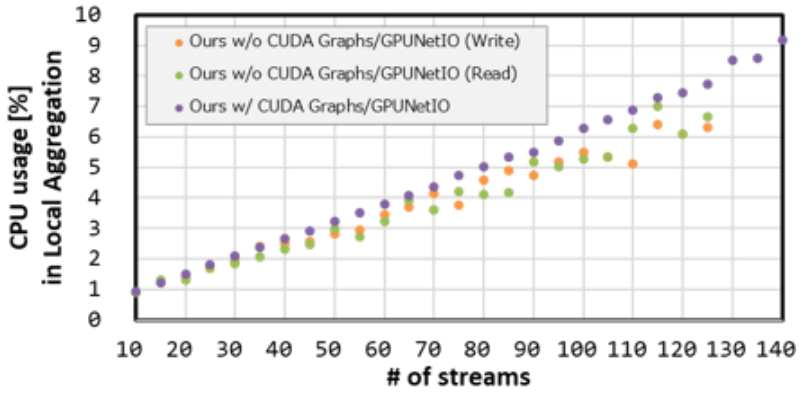
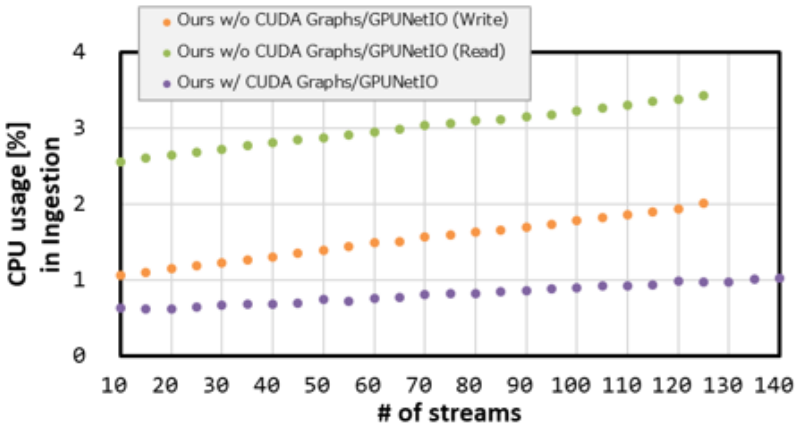


Figure 8-4: Latency and throughput

Objective ID:	5: Required System Resources (CUDA Graphs and GPUNetIO)
Description:	To demonstrate the improvement of required system resources by applying CUDA Graphs and GPUNetIO (ref: 2.4.2 in [PoC Reference])
Procedure:	Same as in Objective ID 2
Lessons Learnt & Recommendations	CPU usage at the Ingestion node was improved by applying CUDA Graphs and GPUNetIO. As described in Objective ID 2, unused CPU resources can be allocated to additional processing such as processing for other applications.



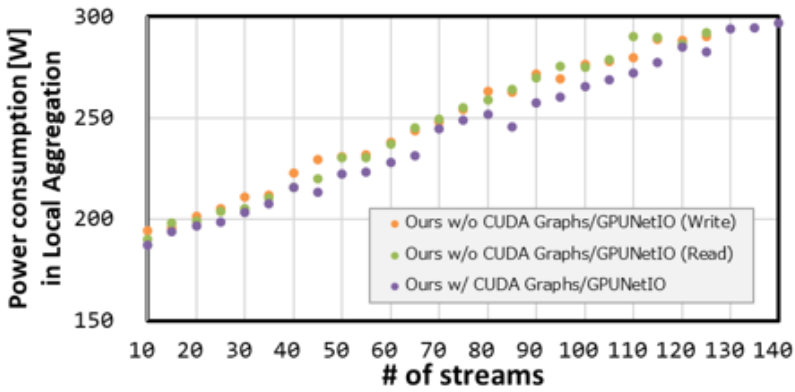
(a) CPU usage in Local Aggregation



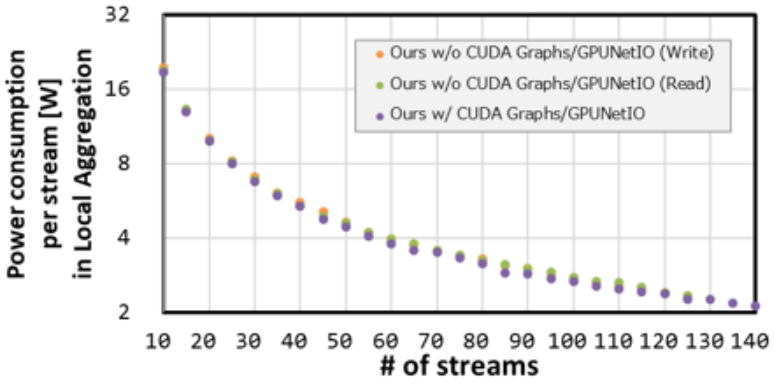
(b) CPU usage in Ingestion

Figure 8-5: CPU usage

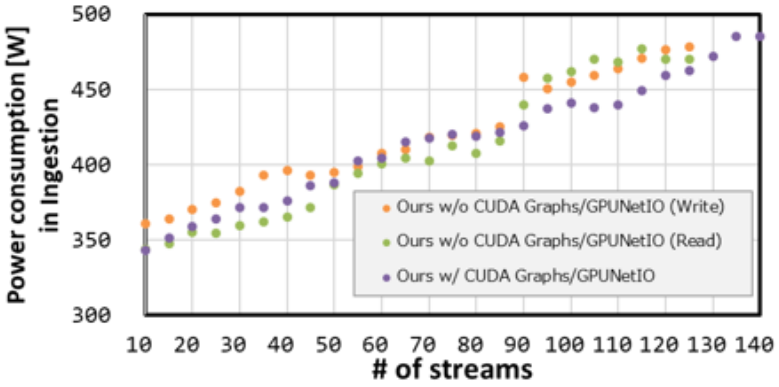
Objective ID:	6: Energy Efficiency (CUDA Graphs and GPUNetIO)
Description:	To demonstrate the improvement of energy efficiency by applying CUDA Graphs and GPUNetIO (ref: 2.4.4 in [PoC Reference]) Note: Same as in Objective ID 3, the power consumption evaluated in this PoC Report was that of the Local Aggregation node and the Ingestion node.
Procedure:	Same as in Objective ID 3
Lessons Learnt & Recommendations	The power consumption of our hardware-accelerated pipeline with CUDA Graphs and GPUNetIO was improved especially at larger number of accommodated video streams, compared with our pipeline without CUDA Graphs and GPUNetIO. For example, let's look at the power consumption per stream at the Ingestion node illustrated in Figure 8-6 (b'). The power consumption of our pipeline without CUDA Graphs and GPUNetIO at 125 streams (the maximum number of accommodated video streams) was 3.8 W, whereas the power consumption of our pipeline with CUDA Graphs and GPUNetIO at 140 streams (the maximum number of accommodated video streams) was 3.5 W. Through Objective ID 1-3, we confirmed that applying CUDA Graphs and GPUNetIO simultaneously achieved high performance and high energy efficiency. This helps us design high-performing and sustainable AI server and data center infrastructure.



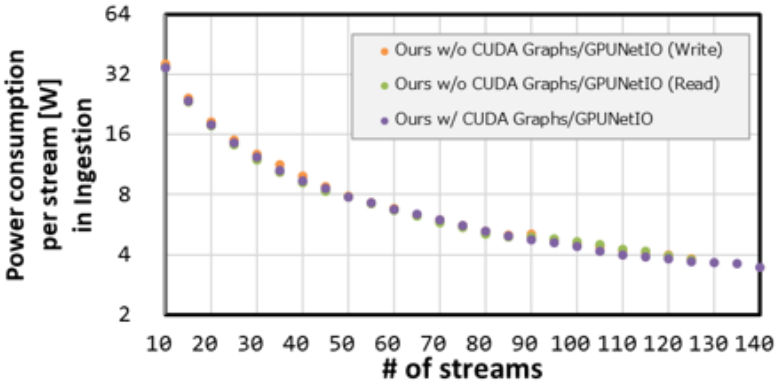
(a) Power consumption in Local Aggregation



(a') Power consumption per stream in Local Aggregation



(b) Power consumption in Ingestion



(b') Power consumption per stream in Ingestion

Figure 8-6: Power consumption

8.2. Scalability Considerations with Composable Disaggregated Infrastructure (CDI)

Composable Disaggregated Infrastructure (CDI) has caught attention in recent years to achieve flexible and efficient scaling of computing infrastructure. IOWN GF published the DCI Cluster RIM [DCI Cluster RIM], which describes the overview of CDI, use cases that the CDI supports, and RIM of the DCI Cluster with CDI. The CPS AM UC is a listed use case that the CDI supports.

Our team considered scenarios to apply CDI to the Ingestion node in the CPS AM UC as follows. First, our team listed parameters related to scaling: the number of cameras that one Ingestion node accommodates, the resolutions of video data (e.g., the number of pixels, frame rate, and bit depth), and the number of analysis targets. These three parameters determine the required resources in the Ingestion node. Then, our team considered example triggers where each of the above parameters changes. Finally, scaling approaches for each of the above parameters are considered. Table 8-1 summarized the considerations.

Table 8-1: Scalability Considerations with CDI

<table>
<tr><th>Parameters related to scaling</th><th>Example triggers to change the parameters</th><th>Scaling approaches</th></tr>
<tr><td>The number of cameras that one Ingestion node accommodates</td><td rowspan="2">• Time of day (e.g., Daytime vs Night) [DCIaaS PoC Report]
• Ordinary time vs When on-site event is held [CPS AM RIM]</td><td rowspan="3">1. Change the number of attached GPUs to an LSN.
2. Change the performance of GPUs attached to an LSN.
3. Change the number of LSNs.</td></tr>
<tr><td>The number of analysis targets</td></tr>
<tr><td>The resolutions of video data (e.g., the number of pixels, frame rate, and bit depth)</td><td>• Ordinary time vs When anomaly occurs</td></tr>
</table>

While several vendors have already introduced CDI products, vendor-specific interfaces and siloed operations remain significant challenges. To address the challenges, collaborations with other technology communities are necessary. For example, OpenFabrics Alliance (OFA) [OpenFabrics Alliance] is developing a framework named Sunfish [Sunfish] based on Redfish in Distributed Management Task Force (DMTF) [Distributed Management Task Force] and Swordfish in Storage Networking Industry Association (SNIA) [Storage Networking Industry Association]. The Sunfish facilitates easier composition of LSNs with hardware from multiple vendors. Cloud Native Computing

Foundation (CNCF) [Cloud Native Computing Foundation] has a sandbox project named Composable Hardware in Disaggregated Infrastructure (CoHDI) [CoHDI] initiated by Fujitsu, IBM, NTT, Red Hat, etc. It aims to deploy and operate container platforms with hardware from multiple vendors. Some components in CoHDI are designed to collaborate with Sunfish.

9. PoC's Contribution to IOWN GF

Contribution	WG/TF	Study Item / Work Item	Comments
Performance and efficiency improvements	RIM-TF EES-TF	SIs/WIs including processing a large amount of sensor data with less energy	Latency, required system resources, throughput, and energy efficiency have been improved. Evaluations with newer hardware-accelerated technologies are expected in the future.
Other UCs	RIM-TF OAF-TF	Same as above	A part of the above improvements can be applied to other UCs such as CPS Mobility Management and Fiber Sensing.
Cybersecurity	RIM-TF DSDT-TF	SIs/WIs related to cybersecurity	Secure RDMA was achieved without overhead.
Design of AI servers and data centers	RIM-TF DCS-TF EES-TF	SIs/WIs related to sustainable AI infrastructure	Our findings help to design sustainable and high-performing AI servers and data centers in the IOWN era.

10. PoC Suggested Action Items and Next Steps

10.1. Gaps Identified in Relevant Standardization

Gap Identified	Forum/SDO	Affected WG/TF	Gap Details and Status
Interoperability between software and hardware	CNCF	RIM-TF	An appropriate combination of software and hardware for hardware-accelerated systems with workload management platforms is difficult to find. Dynamic Resource Allocation (DRA) in CNCF can be a part of a solution.
Interoperability in RDMA	Linux, UCX	RIM-TF	Major RDMA libraries depend somewhat on specific vendors. This dependency prevents interoperability between RDMA applications on RDMA-capable NICs from different vendors.
Cybersecurity	ISO/IEC	RIM-TF DSDT-TF	Although secure RDMA was achieved in this PoC, overall system architecture and configuration for robust cybersecurity including resilience and post-quantum cryptography should be discussed in the future.
Multiple-vendor CDI	OFA, CNCF, OCP	RIM-TF DCS-TF	As described in subsection 8.2, while several vendors have already introduced CDI products, vendor-specific interfaces and siloed operations remain significant challenges. We must build a truly multiple-vendor CDI ecosystem through the CNCF CoHDI sandbox project, integrating Fujitsu's PG-CDI, IBM's Sunfish-based CDI, and other community-driven initiatives.

10.2. PoC Suggested Action Items

To achieve hardware-accelerated systems more efficiently and flexibly, IOWN GF is expected to help fill the gaps described in subsection 10.1.

10.3. Next Steps

The following items will be considered as next steps.

- Collaborations with related TFs, especially DCS-TF, DSDT-TF, EES-TF, and OAF-TF
- Joint discussions with other communities listed in 10.1

11. Conclusion

In our PoC Phase 2 for the CPS AM UC, we achieved secure RDMA without overhead and further efficiency including control-plane processing with NVIDIA CUDA Graphs and GPUNetIO on our hardware-accelerated pipeline in our PoC Phase 1, while satisfying the requirements in the PoC Reference [PoC Reference]. Since the evolution of hardware accelerators and hardware-acceleration technologies continues, including CDI, we will continue to seek a better architecture of the system and pipeline through collaboration with other communities as necessary.

References

- [PoC Report Phase 1] IOWN Global Forum, "Sensor Data Aggregation and Ingestion," 2024.
- [PoC Reference] IOWN Global Forum, "PoC Reference: Reference Implementation Model for the Area Management Security Use Case," 2022.
- [DCI Cluster RIM] IOWN Global Forum, "Data-Centric Infrastructure Cluster Reference Implementation Models," 2025.
- [DCIaaS PoC Report] IOWN Global Forum, "An Implementation of Heterogeneous and Disaggregated Computing for DCI as a Service," 2023.
- [CPS AM RIM] IOWN Global Forum, "Reference Implementation Model (RIM) for the Area Management Security Use Case," 2022.
- [OpenFabrics Alliance] https://www.openfabrics.org/ofa-overview/
- [Sunfish] https://www.openfabrics.org/openfabrics-management-framework/
- [Distributed Management Task Force] https://www.dmtf.org/
- [Storage Networking Industry Association] https://www.snia.org/
- [Cloud Native Computing Foundation] https://www.cncf.io/
- [CoHDI] https://github.com/CoHDI

Document History

Version	Date	By	Description of Change
1.0	May. 29th, 2026	Members listed in section 3	Initial version

Appendix:

A-1: Hardware Configuration

Figure A-1 illustrates the hardware configuration in this PoC. A video server substitutes for cameras. Each node in Figure A-1 is connected via 100Gbps Ethernet for data-plane processing. The Local Aggregation node employs an NVIDIA ConnectX-5 SmartNIC for receiving RTP video streams from the video server in a hardware-accelerated manner utilizing NVIDIA Rivermax. ConnectX-7 SmartNICs are utilized in the Local Aggregation, the Ingestion, and the Data Hub nodes for RDMA and secure RDMA. The Ingestion node employs an NVIDIA A100 GPU for video inference. NVIDIA has released newer GPUs than the A100, but such newer GPUs are not necessary to evaluate the design of our hardware-accelerated pipeline in this PoC. Although the Local Aggregation node and the Data Hub node are not supposed to employ GPUs, these nodes employ NVIDIA A100 GPUs in this PoC due to the implementation of GPUNetIO. Considering this, power consumption from the GPU in the Local Aggregation node is excluded from the evaluation results in subsection 8.1.

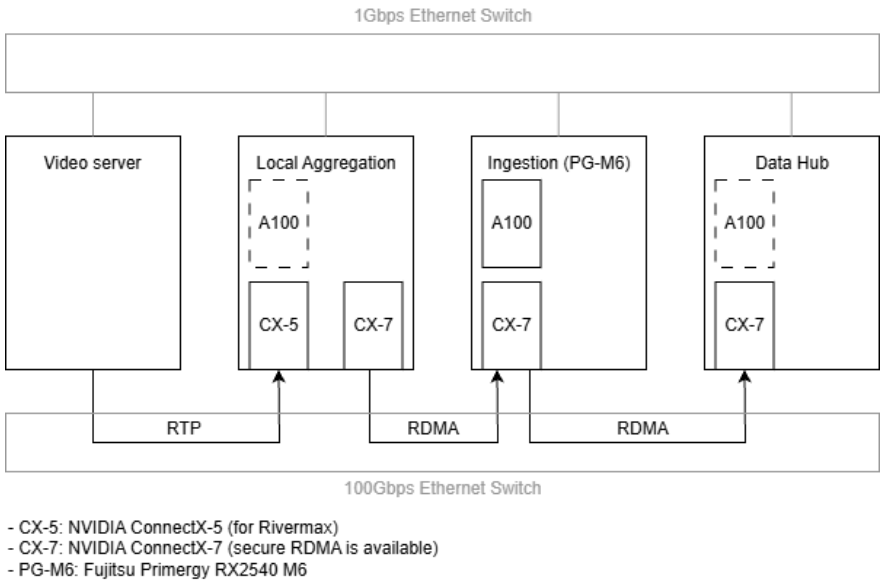


Figure A-1: Hardware configuration

A-2: Software Configuration

Table A-1 lists the software configuration in this PoC.

Table A-1: Software configuration

(a) Software configuration in host

Type	Version
OS	Ubuntu 24.04.3 LTS
Kernel	6.8-generic
DOCA	3.1.0
DOCA_OFED	OFED-internal-25.07-0.9.7
GPU Driver	580.95.05 (580.105.95 in Ingestion)

(b) Software configuration in container

Type	Version
Base image	nvcr.io/nvidia/tensorrt:25.08-py3
OS	Ubuntu24.04.2 LTS
Kernel	6.8-generic
DOCA	3.1.0
DOCA_OFED	OFED-internal-25.07-0.9.7
TensorRT	10.13.2.6-1
Gstreamer	1.24.2
UCX	1.21.0
CUDA	13.0
Rivermax SDK	1.71.30

A-3: Offloading Control-Plane Processing in Ingestion

We originally planned to utilize a converged accelerator NVIDIA A100X (Bluefield-2 DPU + A100 GPU) to offload both data-plane and control-plane processing from the CPU to the hardware accelerators. We confirmed that an A100X-based Ingestion node achieved almost the same throughput as the Ingestion node implemented in our PoC Phase 1. Afterward, such offloading has become enabled with CUDA Graphs and GPUNetIO on a DOCA-capable SmartNIC and a GPU as in this report. This is a more versatile configuration, and SmartNICs and GPUs are easier to be used across various applications than converged accelerators.

A-4: Applying NVIDIA Inference Xfer Library (NIXL)

NVIDIA Inference Xfer Library (NIXL) is a library to achieve high-capacity and low-latency data transfer for inference. Though it is mainly utilized for LLM inference, it can also be applied to video inference such as in CPS AM UC. As an advanced activity, we applied the NIXL to our PoC system and confirmed that performance and energy efficiency were almost the same regardless of applying the NIXL or not. This means that applying the NIXL introduces almost no overhead. Compatibility between NIXL and RoCEv2 over MACSec Full Offload, CUDA Graphs, and GPUNetIO was also confirmed. These results will help us to design secure and efficient LLM inference applications.

A-5: Requirements and Expectations in PoC Reference

Features	Section in PoC Reference	Status in this Report
Sensor device	2.2.1, 2.2.4	Met (see 7.2)
Local Aggregation	2.2.2, 2.2.4, 2,2,5	Met (see 7.3)
Ingestion	2.2.3, 2.2.5	Met (see 7.4)
Communication between sensor devices and Local Aggregation	2.2.4	Met (see 4, 7.2, 7.3, and A-1)
Communication between Local Aggregation and Ingestion	2.2.5	Met (see 4, 7.3, 7.4, and A-1)
Latency	2.2.6	Met (see 8.1)
Scalability	2.2.7	Met (see 8.1 and 8.2)
Advanced optional features	2.3	Met (we implemented and evaluated a hardware-accelerated pipeline with secure RDMA and offload of both data and control planes, which were advanced optional features.)
Expected benchmarks	2.4	Met (see 8.1)
Other Considerations	2.5	Met (see 8.1 and 8.2)