Operations and Management Working Group
Internet-Draft
Intended status: Informational
Expires: 4 September 2025

K. Yao, Ed.
China Mobile
W. Wu, Ed.
Peking University
J. Jeong, Ed.
Sungkyunkwan University

3 March 2025

Interfaces of In-Network Computing Functions in Data Center Networking
draft-ywj-opsawg-i2icf-data-center-networking-00

Abstract

In-network computing has gained a lot of attention and been widely investigated as a research area in the past few years, due to the exposure of data plane programmability to application developers and network operators. After several years of trials and research, some of in-network computing capabilities, or to say In-Network Computing Functions (ICF) have been proved to be effective and very beneficial for networked systems and applications, like machine learning and data analysis, and these capabilities have been gradually commercialized. However, there still lacks a general framework and standardized interfaces to register, configure, manage and monitor these ICFs. This document focuses on the applicability of ICFs in a limited domain in [RFC8799], e.g., data center networks, and describes a framework for orchastrating, managing, and monitoring these ICFs.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

In-network computing has gained a lot of attention and been widely investigated as a research area in the past few years, due to the exposure of data plane programmability to application developers and network operators. After several years of trials and research, some of in-network computing capabilities, or to say In-Network Computing Functions (ICFs) have been proved to be effective and very beneficial for networked systems and applications, like machine learning and data analysis, and these capabilities have been gradually commercialized. For example, in-network aggregation to accelerate collective communication operations like Allreduce and Broadcast,

which can be very useful in machine learning model training. Some other documents [I-D.jeong-opsawg-i2icf-problem-statement][I-D.yao-tsvwg-cco-problem-statement-and-usecases][I-D.irtf-coinrg-use-cases] also list use cases and scenarios where in-network computing can be applied.

However, there still lacks a general framework and standardized interfaces to register, configure, manage and monitor these ICFs. Interface to Network Security Functions (I2NSF) in [RFC8329] has defined a general framework for the management and orchestration of Network Security Functions (NSF). However, the framework is not sufficient to configure ICFs, since many of the in-network computing capabilities need to cooperate with endpoint computing capabilities to accelerate different applications together. Thus, it needs a strong coordination between endpoint and in-network nodes, e.g., Programmable Network Devices (PNDs). But the framework of I2NSF can be referenced and modified for the definitions of ICF interfaces.

This document focuses on the applicability of ICFs in a limited domain, e.g., data center networks, and describes a framework for registering, managing, orchastrating, and monitoring these ICFs.

2. Framework and Interfaces

This section presents the detailed design of I2ICF framework and interfaces.

2.1. I2ICF Framework

Figure 1 shows the I2ICF framework. In this framework, there are several major components and relative interfaces.

* Network Device Capability Management System. This module is for management of network device level capability, i.e. data plane programmability. Since there are differences in hardware architectures, data plane programmability and the way for programming varies between vendors. But they may support similar ICFs and can accelerate the same application together in a heterogeneous network environment, once that there is a proper way for the management and orchestration of ICFs. Commonly, the network device capability management system can be implemented within a network controller provided by a specific vendor, by following Software-Defined Networking (SDN) principles.

* Network Management System. This module is for managing the whole network infrastructure. It is usually administrated by network operators or data center service providers.

* Endpoint (EP) Device Management System. This module is for managing endpoint devices. For example, servers, acceleration units like Graphics Processing Unit (GPU) or Neural Processing Unit (NPU), and Network Interface Card (NIC). It is controlled by a single vendor.

* Endpoint (EP) Compute Management System. This module is operated by a data center service provider who controls the entire compute clusters. In heterogeneous compute clusters, compute devices may be controlled by different vendors.

* Application Development Management System. This module is for application development and management. It leverages the network and compute capabilities for application logic design.

* I2ICF User: I2ICF user typically refers to a user or an upper layer platform (e.g., application task management platform) which may use the ICFs for application acceleration, but it does not need to care about how these ICFs are implemented. For example, the user may be an Artificial Intelligence (AI) training platform, the AI training platform may allocate training tasks to the underlying network management system.

* I2ICF Analyzer: I2ICF analyzer collects monitoring data from ICFs for analyzing the behaviors of the ICFs to detect the overloading, malfunctions, and security attacks for the ICFs.

Note that the Endpoint Device Management, Endpoint Compute Management System, and Application Development Management System do not belong to the network administration domain. But these modules need to closely interact with network components to manage ICFs.

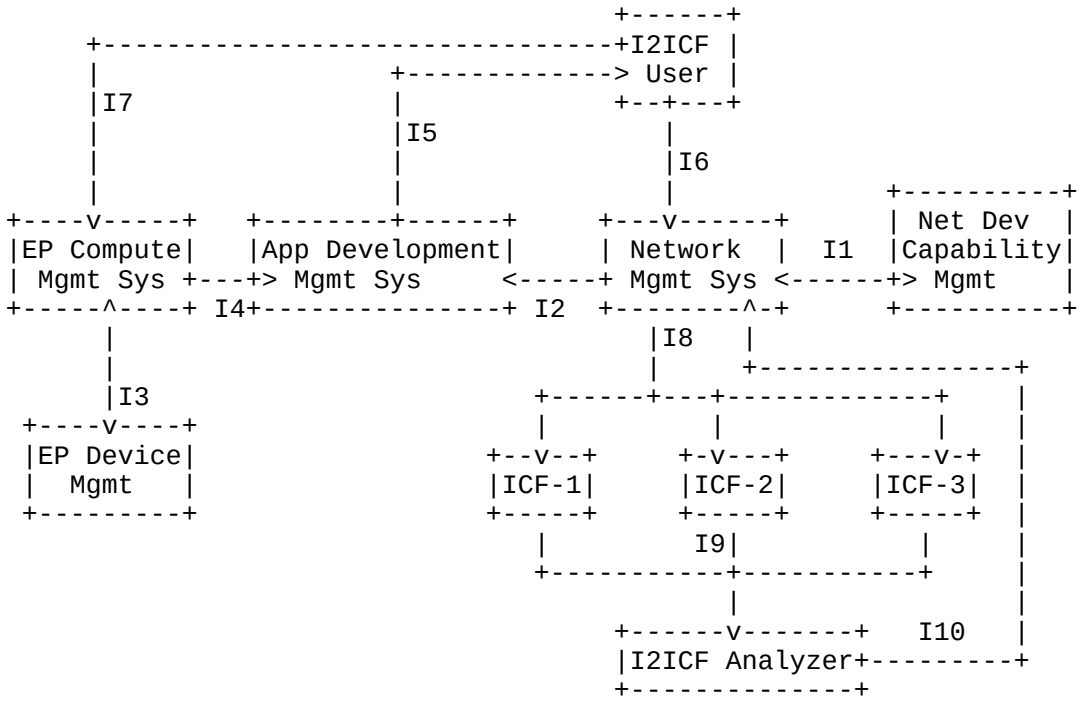

Figure 1: I2ICF Framework and Interfaces

2.2. Interfaces

According to the framework described in the previous section, there are major interfaces that I2ICF should define.

Interface 1 (I1): This is the registration interface between network device capability management system and network management system. This interface is designed for different network device vendors to report their network device capabilities. These capabilities include the object that network devices can process, e.g., packet, the data structures that network devices can support, e.g., array or map, and the primitives that network devices can support, e.g., get, write, and clear. The implementation of this interface can be bidirectional, which means the network device capability management system can report to network management system, and it can also be inquired by the network management system.

Interface 2 (I2): This is the in-network computing capabilities exposure interface. the network management system exposes the in-network computing capabilities to application development management system. These capabilities are dependent on the programmability of network devices of vendors.

Interface 3 (I3): This is the registration interface of endpoint compute capabilities. This interface is for different endpoint compute device vendors to report their capabilities. For example, these devices are GPUs. The capabilities include the objects that these endpoint devices can operate and process, like data vectors and key-value pairs, the data structures that the devices can support, like array and list, and the compute operations that the devices support, like get, write, clear, and convolution. This interface can be realized in bidirectional way similar to interface 1.

Interface 4 (I4): This is the endpoint computing capabilities exposure interface. The endpoint compute management system uses this interface to tell application development management system what compute capabilities it can use to realize some application logics.

Interface 5 (I5): This is the notification interface from application development management system to the I2ICF user. When application development system finishes the application programming, it will notify the I2ICF user, like application task management system. The application task management system will determine which task will be implemented in a network domain, and which should be implemented in a compute domain.

Interface 6 (I6): This is the configuration interface from task management platforms to network management system. Different application task platform may use this interface to deliver a high-level policy to the network management system so that it can implement different in-network jobs through appropriate ICFs.

Interface 7 (I7): This is the configuration interface from task management platforms to the endpoint compute management system. The tasks that should be executed in endpoints will be downloaded via this interface.

Interface 8 (I8): This is the configuration interface for ICFs with a low-level policy that is translated from the high-level policy by the network management system (through a policy translator). When the network management system grabs different in-network jobs, e.g., in-network key-value aggregation for map-reduce task and in-network vector aggregation for machine learning training, it will compile them together, based on the heterogeneous programmability provided by different network device vendors. After the intermediate compilation

and program synthesis, multiple ICFs are generated by a life cycle management system (i.e., network device capability management system). These ICFs are configured via this interface.

Interface 9 (I9): This is the monitoring interface via which monitoring data is collected from ICFs to I2ICF analyzer. The interface can be used to deliver notifications of ICFs to I2ICF analyzer for reporting I2ICFs' alarms and events to I2ICF analyzer.

Interface 10 (I10): This is the analytics interface via which policy reconfigurations or feedback information is delivered from I2ICF analyzer to the network management system. The results of monitoring data analysis at I2ICF analyzer are reported to the network management system for further actions, such as the policy reconfiguration for the target I2ICFs or the execution of an action for the feedback information.

Note that interfaces 3, 4, 5, and 7 may not be within IETF's scope. But they are required to show the entire procedure for ICFs definition, orchestration, and configuration.

3. Use Cases

This section breifly introduces some I2ICFs use cases within limited domain.

* In-network machine learning. Collective communications are typical pattern for large scale AI training. Allreduce, as one of the most import operations in collective communication, can be accelerated by in-network data vector aggregation. Parameters from multiple endpoints gather at an in-network node for computation, and the result will be broadcasted to all of these endpoints. This is essential for saving bandwidth and acclerate training.

* In-network distributed data analysis. Distributed data analysis systems usually contain several major building blocks, data collection, data storage, and data processing. In the procedure of data processing, a job is usually described as an execution plan, normally Directed Acyclic Graph (DAG). Each node in the DAG is an operator, and each edge means the data trasmission between operators. During the execution, key-value pairs follow the DAG for processing. In some main stream processing schemes, for example, in MapReduce, the Reduce operator can be accelerated by in-network computing.

* In-network caching. Caching is an important action for many distributed systems, for example, distributed transaction systems. Key-value store can be offloaded to PNDs for acceleration. There are two major operations when applying in-network caching such as Read operation and Write operation. The in-network caching usually needs some coordination mechanism to guerantee the caching consensus.

4. Security Considerations

I2ICF can benefit many applications, but it indeed introduce some security issues, because it requires the network management system to expose in-network capabilities to application development system. To ensure the overall security of the entire system, here are some sugguestions. First, the application development system should be controlled by the same services providers who own the network and compute infrastructure, for example, cloud service provider or operators. Second, vendors can pre-set some security zones within their devices for isolation, so it will not influence other traffic.

5. IANA Considerations

TBD.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8329] Lopez, D., Lopez, E., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security Functions", RFC 8329, DOI 10.17487/RFC8329, February 2018, <https://www.rfc-editor.org/info/rfc8329>.

[RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <https://www.rfc-editor.org/info/rfc8799>.

6.2. Informative References

[I-D.irtf-coinrg-use-cases]
Kunze, I., Wehrle, K., Trossen, D., Montpetit, M., de Foy, X., Griffin, D., and M. Rio, "Use Cases for In-Network Computing", Work in Progress, Internet-Draft, draft-irtf-coinrg-use-cases-07, 4 December 2024, <https://datatracker.ietf.org/doc/html/draft-irtf-coinrg-use-cases-07>.

[I-D.jeong-opsawg-i2icf-problem-statement]
Jeong, J. P., Shen, Y., Ahn, Y., Kim, Y., Jr., E. P. D., and K. Yao, "Interface to In-Network Computing Functions (I2ICF): Problem Statement", Work in Progress, Internet-Draft, draft-jeong-opsawg-i2icf-problem-statement-00, 3 March 2025, <https://datatracker.ietf.org/api/v1/doc/document/draft-jeong-opsawg-i2icf-problem-statement/>.

[I-D.yao-tsvwg-cco-problem-statement-and-usecases]
Yao, K., Shiping, X., Li, Y., Huang, H., and D. KUTSCHER, "Collective Communication Optimization: Problem Statement and Use cases", Work in Progress, Internet-Draft, draft-yao-tsvwg-cco-problem-statement-and-usecases-00, 23 October 2023, <https://datatracker.ietf.org/doc/html/draft-yao-tsvwg-cco-problem-statement-and-usecases-00>.

Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Ministry of Science and ICT (MSIT) (No. RS-2024-00398199 and RS-2022-II221015).

Authors' Addresses

Kehan Yao (editor)
China Mobile
Beijing
100053
China
Email: yaokehan@chinamobile.com

Wenfei Wu (editor)
Peking University
Beijing
100053
China
Email: wenfeiwu@pku.edu.cn

Jaehoon Paul Jeong (editor)
Department of Computer Science & Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea
Phone: +82 31 299 4957
Email: pauljeong@skku.edu
URI: http://iotlab.skku.edu/people-jaehoon-jeong.php