# Data Fabrics for Multi-Domain Information Systems

Pooyan Habibi, Morteza Moghaddassian, Shayan Shafaghi and Alberto Leon-Garcia
Department of Electrical and Computer Engineering
University of Toronto
Ontario, Canada
pooyan.habibi@mail.utoronto.ca, m.moghaddassian@utoronto.ca,
shayan.shafaghi@mail.utoronto.ca, alberto.leongarcia@utoronto.ca

***Abstract*—Data exchange in information systems that span multiple policy domains typically rely on network middleware that can abstract the management of underlying heterogeneous communication protocols. This also involves issues in managing interoperability, scalability, and privacy that arise in the movement of data from one domain to another information domain. The Data Fabric is an emerging approach to systematically build and design such middleware systems to support multi-domain exchange at scale. In this paper, we discuss and compare two key data-centric approaches: 1) application layer topic-based messaging and name-based networking in a multi-cloud environment. We implement and deploy these two approaches (using Kafka and NDN) and we compare the performance in terms of object transfer latency and CPU and memory utilization. We find that NDN networking has superior latency performance and lower resource usage. We believe that this advantage derives from the fact that named-based messaging operates at the network level, while topic-based messaging operates at the application level.**

***Index Terms*—Data Fabric, Kafka, Multi-domain Information Systems, Named Data Networking, Network Middleware.**

## I. Introduction

We live in a digital world awash in data, where social media, IoT devices, sensors, cameras, and the vast expanse of the Web create enormous volumes of data every second. Individuals, governments, businesses, and organizations benefit from harnessing the data available to them. Typically, intelligence is extracted from the data through analytics and machine learning and then applied to decision making in a wide range of scenarios. Applications can range from sentiment analysis and market trading to automated monitoring, protection, and management in: buildings and factories; networks and information technology systems; and in operations technology systems such as energy generation, power grids, and pipelines. All of these applications depend on the availability of the right data at the right time and the expeditious extraction of intelligence to effect timely decision making. A key issue here is the sharing of data beyond the stakeholder's domain where the data originated. Sharing data across policy domains gives rise to significant challenges in terms of interoperability, scalability, security, and privacy.

The Data Fabric is an emerging concept intended to facilitate the exchange of data in multi-domain information systems while addressing interoperability, scalability, security and privacy [1]. In general, a data fabric is a middleware architecture with associated data-centric services that together provide a unified and consistent experience across endpoints in a multi-domain information system [1], [2]. While individual information domains that use a common data fabric can impose policies and regulations to govern the sharing of data beyond their borders, the data fabric can act as a unifying layer for accessing data across all attached domains, by using abstraction to hide the underlying heterogeneity of data storage and management systems, communication protocols, and cloud environments that store, move, and process data in each domain [1], [2]. The Data Fabric needs to manage the heterogeneity of the underlying communication protocols to ensure consistency of experience as the data traverses different systems, cloud providers, and policy domains [3].

In this paper, we focus on examining data-centric approaches of sharing data across multiple information domains in the so-called Data Fabrics. Data-centric approaches allow using topics/names to organize and move data objects in the fabric. Names/Topics can simplify sharing data across multiple domains in the fabric by: 1) decoupling data exchange from the identity of the communicating endpoints and 2) allowing application names to be directly mapped to the underlying communication methods used in the fabric. In particular, we aim to measure and compare the performance of topic-based messaging systems that operate at the application layer using Kafka [4] with the performance of name-based networking methods that operate at the network layer using the Named Data Networking (NDN) architecture [5]. We believe that comparing data-centric application and network layer approaches can provide useful insights about the applicability of these approaches in various scenarios that involve sharing data across multiple information domains.

For measurement purposes, we create a multi-cloud network infrastructure where individual cloud network providers act as separate information domains. We measure the amount of resources required to support various cross-domain throughput, as well as the corresponding latency when data objects traverse across the information domains. The rest of this paper is organized as follows. We overview key messaging systems and name-based networking in Section II. Next, in Section III, we discuss our multi-cloud deployment environment and elaborate our performance evaluation results and experiments. In Section IV, we discuss several use-case scenarios that can benefit from data fabrics for enabling data exchange in multi-domain information systems. Finally, we provide conclusions and discuss future work in Section V.

978-3-903176-59-1 

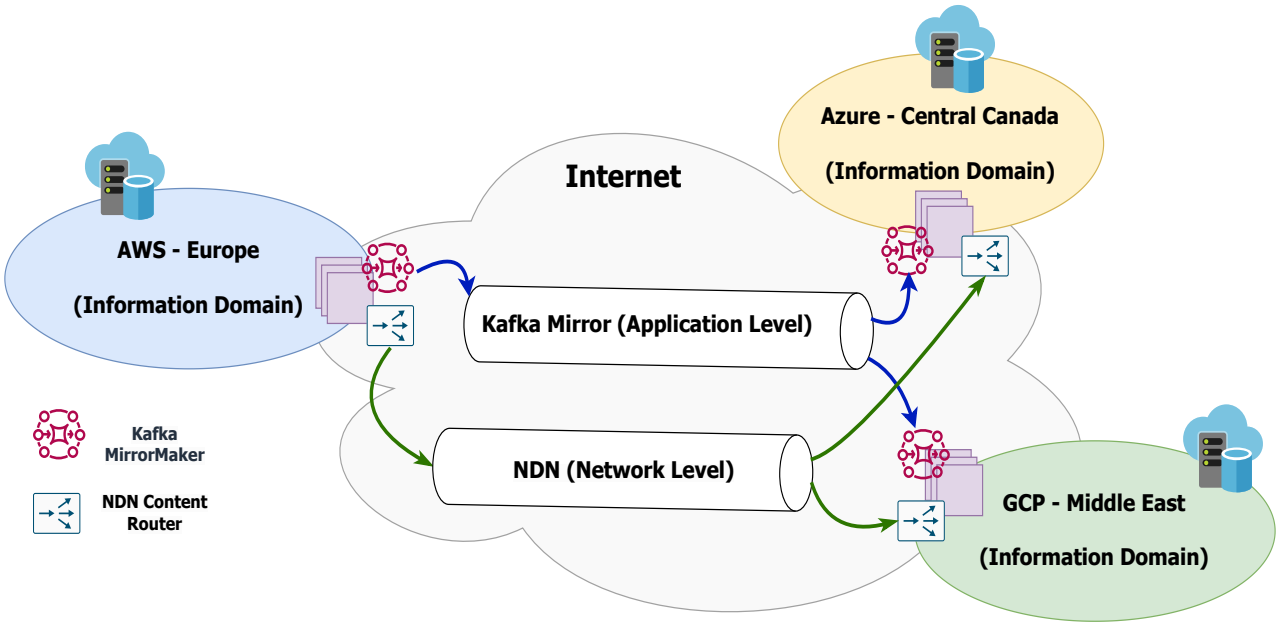


Fig. 1. Multi-cloud deployment of Kafka platform and NDN network on Azure, AWS, and GCP

## II. Background

### A. Topic-based Messaging Systems

Topic-based messaging is a form of communication in which data messages are programmed to be delivered to a topic message stream rather than a specific destination address [6]. The topic is typically an application-specified name selected by data producers before sending their data messages into the messaging system. Once a data message is sent to a topic in the messaging system, the system delivers the message to all data consumers who have active subscriptions to receive the data for the given topic [6]. A topic-based messaging system typically uses a network of distributed messaging brokers that each handle the delivery of messages to data consumers in their attention [7]. Due to the asynchronous nature of topic-based messaging, these systems are ideal for Data Fabrics as data producers and consumers across various information domains do not need to know about each others' identity and location [6].

Today, a variety of messaging systems support topic-based messaging protocols. Message Queuing Telemetry Transport Protocol (MQTT) [8] is one of the principal and widely used messaging protocols on the Internet. MQTT is a lightweight messaging protocol designed for machine-to-machine communication in high latency and low bandwidth network conditions [9]. Advanced Message Queuing Protocol (AMQP) is another widely used messaging protocol on the Internet [10]. AMQP focuses mainly on supporting process-to-process communication over IP networks [11]. Both the MQTT and AMQP protocols are widely used to support communication between system components for the Internet of Things [12], [13] and in machine learning applications [13]. Both protocols are also supported by a wide range of messaging brokers that range from micro-scale to cloud-scale performance [7], [13].

The extended messaging and presence protocol (XMPP) [14] is another messaging protocol that is widely used on the Internet. Due to its verbose and network-agnostic nature [15], the XMPP protocol is used in many applications, including IoT [16], instant messaging [17], [18], and real-time voice and video communications [14]. Much like MQTT, the Data Distribution Service (DDS) is used to support machine communication in real-time systems [19]. However, unlike MQTT and other messaging systems, DDS features a brokerless messaging protocol [19] which is widely used in specific contexts such as smart grids [20]. Finally, we would like to highlight Kafka messaging [4] as an important messaging system that features robust messaging at cloud-scale [7]. Kafka messaging leverages a high-performance TCP-based protocol that is optimized for message delivery at scale [21]. Due to its high-performance, Kafka messaging is considered ideal for use in large-scale messaging environments such as Data Fabrics [22].

### B. Name-based Networking

Name-based networking uses names to organize data, much as topic-based messaging systems use topics to organize its flows. A fundamental difference, however, is that names in name-based networking replace network layer IP addresses [23]. In other words, names are used not only to organize the data but also to directly identify data in the network. Consequently, there is no need for names in the application layer (i.e., topics) [24]. The potential of name-based networking to simplify the network stack required for communications in a

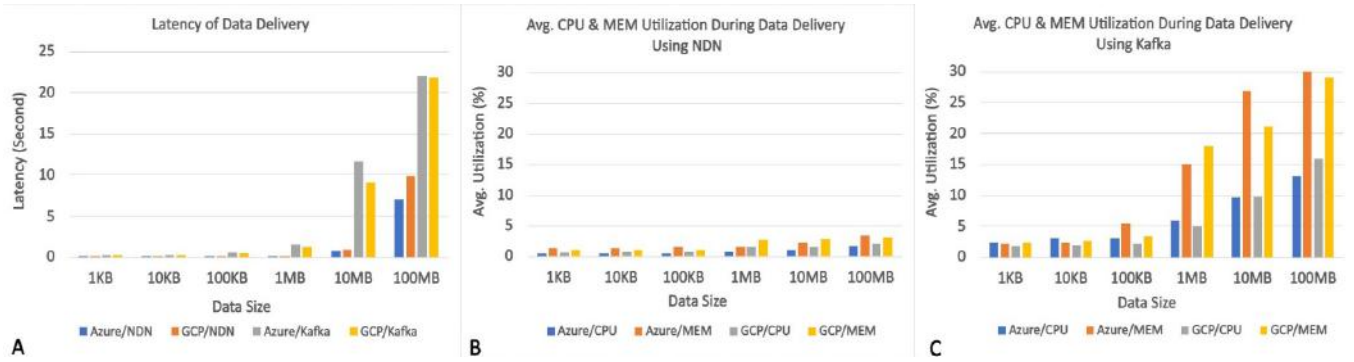


Fig. 2. **(A)** Average latency of data delivery from AWS to Azure and AWS **(B)** Average CPU utilization **(C)** Average Memory utilization

Data Fabric, makes it an ideal solution for use in multi-domain information systems.

In a broader sense, name-based networking also provides an approach to uniquely identify information content in the network, which is also known as Information-Centric Networking (ICN) [25]. A Data Fabric that uses ICN can recognize the uniqueness of the contents in the network and move the contents in the fabric based on its type and substance [25]. The same can be also achieved by application-layer content-based messaging systems [6]. However, an ICN architecture achieves this goal at the network level [26]. Currently, name-based networking is supported by several ICN architectures, including CCN [27], NDN [5], PSIRP [28], PURSUIT [29], and NetInf [30]. Some of these were developed as future Internet architecture projects [31] for use at the Internet scale.

## III. Experiment Methods and Environments

### A. Multi-cloud Deployment Setup

We created a multi-domain environment by using resources from three different public cloud service providers, Azure, Amazon Web Service (AWS), and Google Cloud Platform (GCP), as shown in Fig. 1. Logically, each cloud service provider is a separate policy domain governing the data that is generated in that domain. A Data Fabric facilitates data exchange among the separate policy domains. We implemented the two different approaches for building the given data fabric: 1) multi-cloud Kafka messaging and 2) name-based data networking using NDN. To provide a fair performance comparison, we allow Kafka [4] and NDN [5] to each use a separate virtual node as a data gateway for each policy domain. The gateway's role is to mirror outgoing data from a given policy domain to other domains in the data fabric. Each data gateway node is powered by 2 virtual CPUs and 4GB of RAM. In this deployment, we use the AWS domain as a data producer, and Azure and GCP as data consumer domains.

### B. Kafka Messaging Deployment

We deployed Kafka in our multi-cloud information system using Apache Kafka 3.5 [4]. In each policy domain, we utilized a separate Kafka cluster. We employed MirrorMaker 2.0 (MM2) [32] to enable cross-domain replication. MM2 functions as a multi-domain data replication engine in our system and is built on the Kafka Connect framework [33]. Each MM2 instance encapsulates both Apache Kafka source and sink connectors [34] that are compatible with each cloud provider. These connectors are responsible for automatically detecting new topics and partitions in each domain and ensuring that topic configurations are synchronized among registered domains. This approach allows us to read data from the topics in the source information domain and write it to topics with the same name in the target information domain.

### C. NDN Deployment

We build a Data Fabric that is based on NDN [5], that uses the latest version of Named Data Forwarding (NFD) [35] on each data gateway. NFDs are programmed to interconnect using Faces [35] that mutually connect them. For data exchange, we used the NDN Chunk tool [36] from the NDN tools to transfer a predefined amount of data from AWS policy domain to Azure and GCP policy domains simultaneously.

### D. Performance Comparison Results

We compared the performance of Kafka messaging and the NDN network, both of which were deployed in our multi-cloud environment. We produced data objects of varying sizes in the AWS policy domain and allowed these data objects to be replicated to other policy domains (Azure and GCP) via Kafka and NDN deployments. We measured the latency of data distribution (Fig. 2, A) and the CPU and memory utilization of the data gateways in the Azure and GCP policy domains (Fig. 2, B and C) for each data object produced.

In Fig. 2(A), we compare the latencies of Kafka messaging with NDN messaging when distributing data objects from the AWS policy domain to the Azure and GCP policy domains. Data objects ranged in size from 1KB to 100MB. Across all object sizes, NDN messaging exhibited much lower latency than Kafka messaging. We observed that the NDN network consistently experienced retransmissions of data packets and timeouts when using NDN Chunk for data generation, which could have influenced the results. We anticipate that latency can be further reduced for larger data objects in NDN networks when operating in more optimized networking environments. A significant factor contributing to NDN's lower latency, in

TABLE I
NDN VS KAFKA UNDER 1GB DATA OBJECT SIZE

| Cloud/Param | Latency (Seconds) | CPU (%) | Memory (%) |
|---|---|---|---|
| NDN/Azure | 101.455 | 18.25 | 9.63 |
| Kafka/Azure | 216.526 | 46.73 | 49.31 |
| NDN/GCP | 147.237 | 29.37 | 13.8 |
| Kafka/GCP | 215.051 | 49.47 | 43.31 |

comparison to Kafka, is its operation at the network layer, while Kafka messaging functions at the application layer.

Next, we compare the CPU and memory performance between Kafka messaging and NDN messaging. By contrasting Figure 2(B) with Figure 2(C), it becomes evident that data gateways equipped with NDN data forwarders surpass those using Kafka. This is evidenced by their consistent use of significantly less memory and CPU capacity when receiving data objects across all object sizes. It is important to note that the slight variations in latency, CPU, and memory utilization among data gateways in Azure and GCP policy domains could stem from differing networking conditions and configurations inherent to each cloud service provider's infrastructure.

### *E. Meeting The Gigabyte Scale*

To extend our evaluation to higher scales, we measured the performance of the Kafka and NDN deployments in experiments where a 1GB data object is replicated from the AWS policy domain to the Azure and GCP policy domains simultaneously. The latencies for NDN increased by more than a factor of 10 as the object size went from 100 MB to 1 GB. The latencies for Kafka increased by less than a factor of 10, but were still nearly twice that of NDN. We again observed that data packet retransmission occurs frequently in the NDN network while using the NDN Chunk data generator. In particular, we observed 408 data packet retransmissions for the data object sent from the AWS data gateway to the data gateway in the Azure policy domain, and 2098 data packet retransmissions for the data object that was sent from the AWS data gateway to the data gateway in GCP. This behavior requires further investigation. In terms of CPU utilization, NDN continued to show lower usage than Kafka. Furthermore, NDN showed a higher proportional increase in usage relative to Kafka, showing that its resource use increases according to load, whereas Kafka has a higher but more fixed usage level.

## IV. OPPORTUNITIES AND USE CASES

**Federated Learning:** Federated machine learning environments serve as an exemplary case of a multi-domain information system. In these environments, a decentralized training model replaces the traditionally used centralized model that operates on a single global server collecting data from all sources across multiple information domains [37]. In federated learning, domains with shared interests collaborate by training their individual local models using their own data. These domains can enhance their model accuracy while safeguarding their privacy. This is achieved by only sharing locally generated model parameters with the central server. This central server then utilizes these local models to construct a comprehensive model that integrates the insights from various information domains [37].

In such a framework, the Data Fabric ensures the accuracy of the model by facilitating the exchange of model parameters. Moreover, it manages the versions of machine learning models across the participating entities from different domains [38]. This guarantees that the most current and relevant models are available to all participants, thereby amplifying the efficiency and success of the Federated Learning process.

**Healthcare:** Data Fabric systems offer a powerful solution to the challenges associated with sharing and integrating patient-generated data from wearable devices, sensors, and health records [39], [40]. As patients increasingly use health monitoring devices, the data produced becomes invaluable for global analysis. This information offers insights into current health trends and aids in responding to pandemics like COVID-19 [41]. The Data Fabric serves as a unified data layer, seamlessly integrating information from varied sources such as wearable devices, electronic health records, and research data [39]. Leveraging a Data Fabric allows healthcare organizations to securely gather and disseminate vast amounts of data across different information domains. They can do so while adhering to strict policy governance, ensuring both data privacy and security [39]. This methodology promotes cross-institutional collaboration and research without jeopardizing patient privacy or data integrity. Thanks to the scalability and real-time data access offered by the Data Fabric, healthcare professionals can analyze and interpret data more effectively, which leads to enhanced patient care, improved public health outcomes, and a deeper understanding of global health trends [39].

## V. SUMMARY AND FUTURE WORK

This paper has investigated two data-centric approaches for building Data Fabric systems that support data exchange in multi-domain information systems. The paper examined Topic-Based Messaging Systems and Name-Based Data Networking as approaches to design Data Fabrics. The two approaches were implemented and deployed on three public clouds: AWS, GCP, and Azure. Experimental measurements of object delivery latency and gateway CPU and memory usage were conducted for the two approaches. The results show that Name-Based Data Networking (implemented using NDN) outperforms Topic-Based Messaging (implemented using Kafka). NDN has much lower latencies than Kafka, and its CPU and memory utilization are lower as well. We attribute this superior performance to the fact that NDN operates at the network layer while Kafka operates at the application layer. We also noted an opportunity to further improve NDN Chunk tool performance by investigating its data object segmentation algorithm with a view to reducing packet retransmission.

## REFERENCES


[1] J. CaraDonna and A. Lent, "Netapp data fabric architecture fundamentals," 2016.

[2] V. Sharma, B. Balusamy, J. J. Thomas, and L. G. Atlas, *Data Fabric Architectures: Web-Driven Applications*. Walter de Gruyter GmbH & Co KG, 2023.

[3] E. Hechler, M. Weihrauch, and Y. Wu, "Data fabric within an enterprise architecture," in *Data Fabric and Data Mesh Approaches with AI: A Guide to AI-based Data Cataloging, Governance, Integration, Orchestration, and Consumption*. Springer, 2023, pp. 257–275.

[4] "Apache Kafka — kafka.apache.org," https://kafka.apache.org/, [Accessed 04-08-2023].

[5] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[6] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM computing surveys (CSUR)*, vol. 35, no. 2, pp. 114–131, 2003.

[7] V. John and X. Liu, "A survey of distributed message broker queues," *arXiv preprint arXiv:1704.00411*, 2017.

[8] "MQTT - The Standard for IoT Messaging — mqtt.org," https://mqtt.org/, [Accessed 04-08-2023].

[9] S. Quincozes, T. Emilio, and J. Kazienko, "Mqtt protocol: fundamentals, tools and future directions," *IEEE Latin America Transactions*, vol. 17, no. 09, pp. 1439–1448, 2019.

[10] "Home — AMQP — amqp.org," https://www.amqp.org/, [Accessed 04-08-2023].

[11] "CloudAMQP - Queue starts here. — cloudamqp.com," https://www.cloudamqp.com/, [Accessed 04-08-2023].

[12] M. B. Yassein, M. Q. Shatnawi *et al.*, "Application layer protocols for the internet of things: A survey," in *2016 International Conference on Engineering & MIS (ICEMIS)*. IEEE, 2016, pp. 1–4.

[13] P. K. Donta, S. N. Srirama, T. Amgoth, and C. S. R. Annavarapu, "Survey on recent advances in iot application layer protocols and machine learning scope for research directions," *Digital Communications and Networks*, vol. 8, no. 5, pp. 727–744, 2022.

[14] "XMPP — The universal messaging standard — xmpp.org," https://xmpp.org/, [Accessed 04-08-2023].

[15] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," RFC 6120, Mar. 2011. [Online]. Available: https://www.rfc-editor.org/info/rfc6120

[16] N. Nikolov, "Research of mqtt, coap, http and xmpp iot communication protocols for embedded systems," in *2020 XXIX International Scientific Conference Electronics (ET)*. IEEE, 2020, pp. 1–4.

[17] D. Gupta, J. Shivankar, and S. Gugulothu, "Instant messaging using xmpp," in *Journal of Physics: Conference Series*, vol. 1913, no. 1. IOP Publishing, 2021, p. 012126.

[18] A. Hornsby and R. Walsh, "From instant messaging to cloud computing, an xmpp review," in *IEEE international symposium on consumer electronics (ISCE 2010)*. IEEE, 2010, pp. 1–6.

[19] G. Pardo-Castellote, "Omg data-distribution service: Architectural overview," in *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings.* IEEE, 2003, pp. 200–206.

[20] A.-M. Basem and H. Ali, "Data distribution service (dds) based implementation of smart grid devices using ansi c12. 19 standard," *Procedia Computer Science*, vol. 110, pp. 394–401, 2017.

[21] N. Narkhede, G. Shapira, and T. Palino, *Kafka: the definitive guide: real-time data and stream processing at scale*. " O'Reilly Media, Inc.", 2017.

[22] J. Yongguo, L. Qiang, Q. Changshuai, S. Jian, and L. Qianqian, "Message-oriented middleware: A review," in *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE, 2019, pp. 88–97.

[23] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, "Named data networking (ndn) project," *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, vol. 157, p. 158, 2010.

[24] B. Mishra and A. Kertesz, "The use of mqtt in m2m and iot systems: A survey," *IEEE Access*, vol. 8, pp. 201 071–201 086, 2020.

[25] X. Jiang, J. Bi, G. Nan, and Z. Li, "A survey on information-centric networking: rationales, designs and debates," *China Communications*, vol. 12, no. 7, pp. 1–12, 2015.

[26] A. Carzaniga, M. Papalini, and A. L. Wolf, "Content-based publish/subscribe networking and information-centric networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, 2011, pp. 56–61.

[27] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves, "Content-centric networking," *Whitepaper, Palo Alto Research Center*, pp. 2–4, 2007.

[28] D. Lagutin, K. Visala, and S. Tarkoma, "Publish/subscribe for internet: Psirp perspective," in *Towards the Future Internet*. IoS Press, 2010, pp. 75–84.

[29] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, "Developing information networking further: From psirp to pursuit," in *Broadband Communications, Networks, and Systems: 7th International ICST Conference, BROADNETS 2010, Athens, Greece, October 25–27, 2010, Revised Selected Papers 7*. Springer, 2012, pp. 1–13.

[30] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of information (netinf)–an information-centric networking architecture," *Computer Communications*, vol. 36, no. 7, pp. 721–735, 2013.

[31] "NSF Future Internet Architecture Project — nets-fia.net," http://www.nets-fia.net/, [Accessed 04-08-2023].

[32] "KIP-382: MirrorMaker 2.0 - Apache Kafka - Apache Software Foundation — surl.li," http://surl.li/jtarl, [Accessed 04-08-2023].

[33] "Kafka Connect — Confluent Documentation — surl.li," http://surl.li/jtarx, [Accessed 04-08-2023].

[34] "Kafka Connectors — Confluent Documentation — surl.li," http://surl.li/jtasy, [Accessed 04-08-2023].

[35] "NFD Overview x2014; Named Data Networking Forwarding Daemon (NFD) 22.12 documentation — docs.named-data.net," https://docs.named-data.net/NFD/current/overview.html, [Accessed 05-08-2023].

[36] "ndn-tools/tools/chunks at master · named-data/ndn-tools — github.com," https://github.com/named-data/ndn-tools/tree/master/tools/chunks, [Accessed 05-08-2023].

[37] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE signal processing magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[38] E. Hechler, M. Weihrauch, and Y. Wu, "Data fabric and data mesh for the ai lifecycle," in *Data Fabric and Data Mesh Approaches with AI: A Guide to AI-based Data Cataloging, Governance, Integration, Orchestration, and Consumption*. Springer, 2023, pp. 195–228.

[39] P. Rahi, M. Dandotiya, S. P. Sood, and M. Tiwari, "Sayed sayeedi 7 an open-source data fabric platform: Features, architecture, applications, and key challenges in public healthcare," *Data Fabric Architectures: Web-Driven Applications*, p. 127, 2023.

[40] D. A. M. Budida and R. S. Mangrulkar, "Design and implementation of smart healthcare system using iot," in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. Ieee, 2017, pp. 1–7.

[41] S. Selvaraj and S. Sundaravaradhan, "Challenges and opportunities in iot healthcare systems: a systematic review," *SN Applied Sciences*, vol. 2, no. 1, p. 139, 2020.