An ICN architecture within the framework of SDN

Suyong Eum*, Masahiro Jibiki*, Masayuki Murata†, Hitoshi Asaeda*, Nozomu Nishinaga*
* National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi, Koganei, Tokyo, 184-8795 Japan
† Osaka University, Graduate School of Information Science and Technology
1-5 Yamadaoka, Suita, Osaka, 565-0871 Japan

Abstract: The core design principle of Information Centric Networking (ICN) is in the name based routing which enables users to ask for data object by name and makes the infrastructure to deliver it to users from a nearby cache if available. Software Defined Networking (SDN) brings low-cost and low-complexity of network management by decoupling architecture from infrastructure, which promises the continuous evolution of network architectures in a flexible manner.
The synergy between ICN supporting efficient data dissemination as the norm and SDN providing flexible management framework enables the combination to be a fully controllable framework for efficient data dissemination. In this paper, we propose an ICN architecture within the framework of SDN.

Key words: Information Centric Networking (ICN), Software Defined Networking (SDN).

I. Introduction

Information Centric Networking (ICN)[1][2][3][4][5][6] has attracted much attention of network research community[1] recently due to the demand on a novel network architecture optimized for the dissemination of explosively increasing multimedia contents. For instance, the amount of traffic growth by a factor of 1.7 every year [8] and all forms of multimedia traffic (TV, video on demand [VoD], Internet, and P2P) would continue to be approximately 90% of the global consumer traffic by 2015 [9]. Software Defined Networking (SDN) is another axis of recent notable network research trend, which allows a flexible control over network infrastructures through the separation of the control plane from the data-forwarding plane in network devices. In particular, the separation of the architecture (control plane) from the infrastructure (data plane) simplifies the adoption of novel network architectures [10].

The functions of ICN can be divided into two major parts. One is an intelligent part that is aware of a user demand based on provided information, e.g., meta data. The other is a simple forwarding part that forwards user requests or responses following the path(s) defined by the intelligent part. This separation of the ICN functions is well aligned with the design principle of SDN, and it triggered the integration of ICN within the framework of SDN [11][12][13]. From a viewpoint of ICN, this integration provides several benefits. First, SDN provides the operator of ICN with a management framework that enables a full control over the distributed ICN entities. Second, SDN enables an immature ICN architecture to keep its evolution without the concern over its deployment. Third, the awareness feature of ICN can be fully boosted within the framework of SDN that provides the global view of the overall network. Unfortunately, current SDN cannot directly support the operation of ICN since it does not provide a mechanism to define a finely grained flow, e.g., based on a data or content identifier.

In this paper, we design an ICN architecture by leveraging the various benefits of SDN, especially focusing on how ICN and SDN can be integrated in a concise and concrete manner. Our special attention is given to the compatibility issue, e.g., how to deploy an ICN architecture within the framework of SDN with the least modification of the both architectural components as well as to support legacy devices running on the architectures.

This article is organized as follows. In Section II, we provide the motivation of this project and review its related works. In Section III, we describe the design consideration of the proposed architecture. In Section IV, the layout of the architecture is elaborated. The logical components of ICN are described in Section V. In Section VI, the operation of the architecture including data dissemination as well as retrieval processes are illustrated. In Section VII, we briefly discuss a performance issue of the proposal. Finally, we conclude this article in Section VIII.

II. Background

A. Motivation

This work was intended to federate legacy heterogenous networks, e.g., sensor networks through an ICN solution, and so data objects in the heterogeneous networks are efficiently shared among them based on the synergy between ICN supporting efficient data dissemination and SDN providing a flexible management framework for ICN.

Initially, we had two design options for this operational scenario. One is to use ICN as an architectural solution that provides low level ICN networking functions even for the end devices. Although, it is a clean approach, it causes the compatibility issue. For instance, off-the-shelf devices have

[1]Recent establishment of the Information Centric Networking Research Group (ICNRG) [7] within the Internet Research Task Force (IRTF).

been already deployed in the networks, e.g., its protocol stack cannot be modified easily. The other design option is to use ICN as a core transport network that provides a caching infrastructure for the networks connected through the ICN network. This deployment scheme limits the full potential of ICN within the core transport network, however, it can embrace legacy devices without any modification. This paper considers the latter case since our primary design consideration is compatibility.

One question that should be answered before we progress this work is what benefits we expect from the federation of networks through ICN? As our initial stage of this work, we mainly expect ICN to provide a caching and processing infrastructure for the networks that have limited storage and processing resources. For example, data objects for unspecified end users, e.g., stock information or weather information can be tossed to the ICN domain and be pushed to end users from nearby ICN caching points. Another scenario would be to deliver the captured video from CCTV at a particular location to end users by taking advantage of the innate multicast capability of ICN. The benefits can be further extended such as mobility support or network resource saving based on a receiver-driven mechanism.

For the justification of SDN adoption, ICN is still immature, which requires further its architectural evolution. In this sense, the ICN deployment should consider its continuous architectural evolution. SDN separates the architecture from the infrastructure, and so the network architectures within the framework of SDN can evolve without worrying about the underlying infrastructure. Moreover, SDN provides an innate management framework for ICN through a well defined interface, which enables ICN to be deployed within the framework of SDN in an agile manner.

B. Related works

In [14], the authors modified TCP header to include a content identifier, IP address, and upload capability information. Thus, a SDN switch[2] detects the new information in the TCP header to redirect the TCP SYN packet to a nearby data holder, and the data holder responds to the request directly using the TCP ACK packet. A similar approach was introduced in [15][16]. The goal of these approaches is to realize ICN functions on the top of the legacy IP protocol. However, their designs mainly focus on how to identify content flows in the SDN domain without discussing logical functional blocks of ICN architectures: content naming, routing, caching, etc.

In [11][13], the authors described the deployment scenarios of their ICN architectures, namely PURSUIT [17] and CONET [18] within the framework of SDN. While the former only elaborates on the design considerations of the deployment, the latter actually deployed their ICN solution over a large scale SDN test-bed called OFELIA [19]. The main difference between them is how to define a content identifer in the SDN domain. For instance, the former constructs a fixed length bloommfilter[3] for a source routing. On the other hand, the latter introduced an identifer called *a tag* which is a fixed length identifier corresponding to the hierarchical name of data object. The use of tags in the SDN domain was also proposed in [12] where the tag represents a network path. Although, there are several deployment scenarios of ICN in the SDN domain, it is difficult to compare these proposals on an equal footing since they all used different ICN architectures and mainly aimed for demonstrating a proof of its concept.

[2]Openflow switch

[3]Encoding the identifiers of links, e.g MAC address existing between source and destination nodes.

III. Design considerations

In this Section, we elaborate on the deployment considerations of ICN in the SDN domain.

A. Awareness

Awareness means that the infrastructure of ICN recognizes the context of a user request due to the information encoded in the content packet including the name of the requested data object and its relevant meta data.

Based on the recognized information, ICN carries out two important functions. One is a caching function that caches any data object passing over the infrastructure of ICN and delivers the caching data object to the users on request. The other is a name based routing function that routes a user request and its corresponding response based on the name of the data object. From a viewpoint of a SDN switch, there is a tradeoff between the two functions. For example, when two different data objects are forwarded from the same source to the same destination nodes following the same path in the SDN domain, the forwarding tables in the SDN switches along the path only require one entry because the both follow the same forwarding path: the two different identifiers of the data objects can be merged into one identifier called a forwarding identifier for the forwarding action in the SDN domain. If possible, the forwarding identifer better be a short fixed length to be aligned with the SDN matching capability.

On the other hand, the SDN switches should be able to differentiate the two different identifiers of the data objects in order to carry out ICN related actions, e.g., caching. For this reason, the identifier of the data object better carries sufficient information, and so the infrastructure of ICN can take an appropriate action over the data object.

In summary, the proposed architecture is recommended to provide an identifier for an efficient forwarding action in the SDN domain, especially when different data objects require the same forwarding action. Thus, their identifiers better be aggregated into a single fixed length identifier. At the same time, the proposed architecture is also recommended to provide a semantic identifier which differentiates individual data objects for a particular ICN action in the SDN domain.

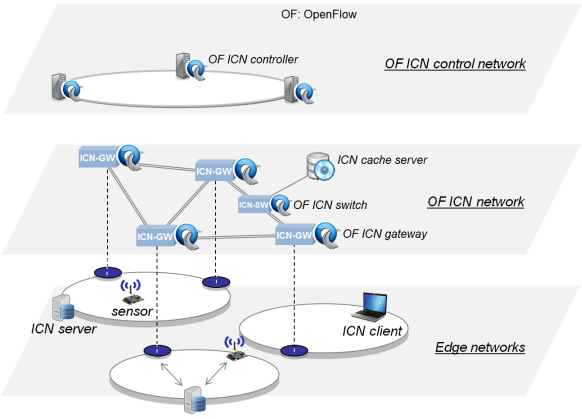

Fig. 1. Architectural layout

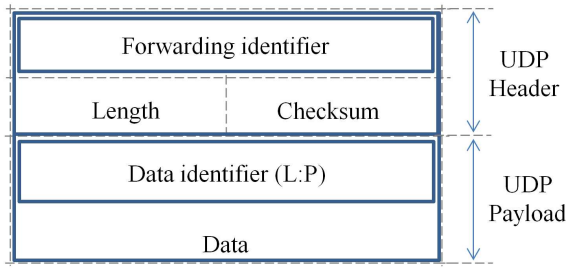

Fig. 2. ICN packet within the OF-ICN network: the data identifier and the forwarding identifier in the UDP packet.

B. Compatibility

Compatibility means the level of modification required to accommodate legacy devices in the proposed architecture.

Ideally speaking, it can be assumed that an ICN protocol is installed in every end device. However, considering off-the-shelf devices deployed widely, or some devices not allowed any modification due to a management issue, such an assumption may not be realistic. Moreover, a SDN solution such as openflow only lookups particular fields of a packet to classify them as flows. Although, the openflow protocol can be extended or modified to deal with the compatibility issue, such an extension needs to balance between the flexibility of the matching function and its performance. Also, it requires a long and tedious standardization process. The other options can be classified as a non-standard approach that abuses an existing field of the protocol to carry the identifier of the data object or something equivalent. For this option, the selection of the field should be considered carefully to enable backward compatibility.

In summary, the proposed architecture is recommended to support legacy devices with the least modification of the protocol stacks. Also, the proposed architecture is recommended to support backward compatibility with a particular SDN solution such as an old version of openflow.

IV. Architectural Layout

The proposed network architecture aims to federate networks, e.g., sensor networks using an ICN solution whose management framework is supported by SDN. We partition the layout of the proposed architecture into three major blocks as shown in Fig. 1: edge networks corresponding to individual networks, e.g., sensor networks, the openflow ICN (OF-ICN) network corresponding to the simple forwarding plane of ICN, and the OF-ICN control network corresponding to the control plane of ICN. We use the scenario of federating sensor networks through the proposed architecture as an example hereafter.

The edge networks comprise of ICN servers, ICN clients, and sensor devices. The ICN server gathers sensing data objects from ICN-enabled sensors as well as non ICN-enabled sensor devices. The ICN server selectively registers data objects to the ICN domain through the OF-ICN gateway. The ICN client retrieves a data object from the ICN domain.

The OF-ICN network comprises of openflow enabled ICN (OF-ICN) nodes. There are two types of OF-ICN nodes: OF-ICN gateway (OF-ICN GW) and OF-ICN switch (OF-ICN SW). The main role of the OF-ICN GW is to convert a general ICN protocol to the openflow based ICN protocol, and vice-versa. The OF-ICN SW is a openflow switch which provides the in-network caching function. Both OF-ICN GW and OF-ICN SW are controlled by the OF-ICN controllers in the OF-ICN control network.

The OF-ICN control network comprises of openflow controllers. Each controller includes several software modules that realize the functions of ICN: routing module, name resolution module, cache module, and security module. The routing module is responsible for maintaining the network topology and the locations of the data objects. Based on the information, it determines the routing paths of user requests and inserts the path information into the forwarding tables of the relevant OF-ICN nodes. The name resolution module is responsible for managing the mapping records of all identifiers used in the architecture. The cache module is responsible for managing caching data objects in the ICN cache servers attached to the OF-ICN SW. For instance, the cache module determines caching points for the data objects based on requests from the ICN servers, and updates the mapping information of the data objects to the name resolution module. The security module here refers the public key infrastructure (PKI) [20] that distributes the public keys of publishers.

V. Architectural Components

Based on the design considerations in the previous Section, we elaborate on the logical components of the proposed architecture, especially from an ICN perspective, and then explain how the local components are realized within the openflow framework.

A. Naming

To address the first design consideration - awareness, we define two identifiers for the proposed architecture: data identifier[4] and forwarding identifier. While the data identifier is used in the general ICN domain, e.g., within an edge network, the forwarding identifier is used in the openflow based ICN domain, e.g., within the OF-ICN network.

The data identifier mainly comprises of the "*label* (L)" of the data object and its "*principal* (P)" information. The label of the data object (L) is given by the publisher which is unique under the publisher domain with which the data object is logically associated. It includes meta data which provides the attributes of the data object. The principal (P) information identifies the publisher of the content, which is the cryptographic hash of the publisher's public key. In the case of the sensor network, the principal information becomes an identifier of one autonomous sensor network domain. Thus, the combination of (L:P) ensures the globally unique name of the data object. Due to the globally unique name, this identifier is used to carry out ICN related functions such as caching in the ICN domain.

The forwarding identifier has a short fixed length (4 bytes) and is used within the OF-ICN network. The short fixed length forwarding identifier enables even legacy openflow switch (v1.0) to be aware of a data flow[5] and to enhance its matching capability based on the implementation at the hardware with a classical parallelization process. Fig. 2 shows how the data and forwarding identifiers are embedded in the UDP packet. Further discussion is given in the following Section.

B. Routing & Mapping identifiers

The name based routing is carried out by the routing module in the OF-ICN control network. Given the global view of the network topology and the locations of data objects, the routing module calculates routing path(s) to the closest caching point(s) based on a routing algorithm such as the Dijkstra algorithm in OSPF [21] or PBR [6]. Then, the OF-ICN controller inserts the path information into the forwarding tables of the relevant OF-ICN SWs.

As mentioned previously, the forwarding action in the OF-ICN domain is carried out based on the forwarding identifier. A problem of using the forwarding identifier is that the total number of identifiable data objects based on the 4 bytes field ($2^{32} \sim 10^{10}$) is not large enough to accommodate all available data objects[6]. In other words, a forwarding identifier and a data identifier cannot have one-to-one mapping. For this reason, the forwarding identifier should be dynamically mapped to the data identifier, and the mapping information is managed by the name resolution module in the OF-ICN[4] controller.

C. Caching

Each OF-ICN SW is associated with an external ICN cache server. External cache means that the caching function is not integrated to the OF-ICN SW hardware.

There are two different caching operations: content distribution and retrieval. The latter is known as cache aware routing [6] that has been explained in the previous Section. Thus, we here only explain the former case how to distribute the copies of contents to the ICN cache servers attached to the OF-ICN SWs.

In a general ICN architecture, the caching decision on an ICN entity is preferably made while a data object is passed through its associated node since each ICN element is aware of user requests and their counterpart responses. However, this operation is not possible within the OF-ICN network since the forwarding identifier cannot provide enough information to identify individual contents[7]. For this reason, the caching decision in the OF-ICN network is made by the caching module in the OF-ICN controller in a centralized manner. The problem of selecting ICN cache servers to cache popular contents is somewhat similar to that of surrogate server placement in CDN in a sense that the selection of proper spots over the network yields an optimum performance [22][23][24].

D. Access control

ICN generally deals with this issue of access control by encrypting each content to discourage unauthorized users from accessing the network: everyone is allowed to retrieve any content in ICN but only authorized user can decrypt the content in principle. Although, this encryption mechanism corresponds to the design principle of ICN, it introduces the concerns over security and network management. Malicious users may flood large number of fake requests to multiple access points with the intention of interrupting them. Unfortunately, such a flooding attack is hard to be detected since the source of the traffic cannot be identified due to non-existence perpetual connectivity between end-to-end hosts in ICN. Thus, it has been required that individual ICN entities support a complicated access control mechanism which may be unrealistic considering the capability of each ICN entity.

We take advantage of the packet inspection capability of openflow switch to react to external events by changing forwarding polices, e.g., dropping or redirecting traffic, especially the controller expresses policies to compose policies at the OF-ICN GW that is able to monitor traffic flows coming into the network, and dynamically provides the event-based network access control mechanism similar to [25].

[4]Data name or data identifier are used interchangeably in this paper.

[5]A legacy openflow switch (v1.0) only allows the inspection of transport protocol header of UDP and TCP packets whose size is only 4 bytes [13].

[6]According to [1], the number of originally published content files that ICN is expected to support was estimated as 10^{11} back in 2007, and there are still many people in ICN research community who believe the number should be much larger such as 10^{15}.

[7]Since the openflow (v1.0) API and its corresponding switches do not support it.

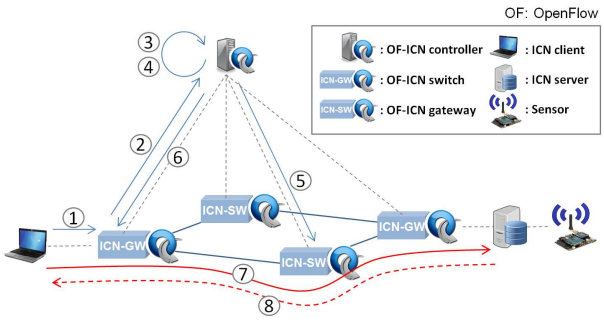

Fig. 3. Data retrieval process

E. Security

Security is built into the data object itself rather than a connection or a device, which is the basic policy in ICN. For example, assuming that a user requests a data object with its name, and receives the requested data object with its signature and the public key of the publisher. Then, the user can authenticate whether the data object is from the publisher by comparing the hash of the received public key with the principal (P) information in the name of the data object. After the authentication, the user hashes the received data object into a message digest and compares it with the message digest from the decrypted signature (using the received public key). If the message digests are same, the user confirms that the data object has not been modified since it was signed. This is called "self-certifying" approach. To verify the signature of a data object, individual users should know the publisher's public key so that they can verify the origin and integrity of the data object. For this reason, we incorporate an external third party authority, e.g., Public Key Infrastructure (PKI) [20] with the architecture to distribute the public keys of publishers, which is the responsibility of the security module in the OF-ICN controller.

VI. Operational view of the architecture

This Section elaborates on the communication process of each device in the architecture including data registration and retrieval processes.

A. Data object registration

In each edge network, we assume that there is a data storage node called ICN server that can talk to ICN-enabled as well as non ICN-enabled devices. Its role is to gather sensing data objects from non-ICN sensors which are under the administration domain. The ICN server registers selected sensing data through the OF-ICN GW. For doing that, first the ICN server names the data objects as described in Section. V-A and passes them to the OF-ICN GW. Then, OF-ICN GW registers the names [key: data identifier, value: locator[8]] through the name resolution module running in the OF-ICN control network.

Optionally, the ICN server can register data object itself and let the OF-ICN controller distribute the data object to several ICN cache servers in the OF-ICN network.

B. Data object retrieval

We here assume that an ICN client knows the identifier of the data object which intends to be retrieved.

① An ICN client sends a request to OF-ICN GW to retrieve a data object. ② When the OF-ICN GW does not have a matching flow in the forwarding table, the first packet is forwarded to the OF-ICN controller. ③ The OF-ICN controller determines the forwarding identifier that corresponds to the requested data object. ④ Based on the global view of the topology as well as the locators of the requested data object from the name resolution module, the routing module in the OF-ICN controller determines the routes to the data object that is either in caches or an ICN server. ⑤ The OF-ICN controller populates OF-ICN SWs' forwarding tables with appropriate entries. ⑥ The OF-ICN controller sends the forwarding identifier to the OF-ICN GW and then the OF-ICN GW encapsulates the ICN request packet into the UDP packet whose source/destination port field is replaced with the forwarding identifier as shown in Fig. 2. ⑦ The ICN request packet is forwarded to the data object following the route set by the OF-ICN controller. ⑧ Lastly, the corresponding data packet is back to the ICN client following the reverse path.

VII. A performance issue

The number of forwarding identifiers, e.g., (4 bytes $\approx 10^{10}$) is much less than that of data identifiers, e.g., 10^{15}. For this reason, we expect a collision when a forwarding identifier is requested while all forwarding identifiers are fully occupied. To analyze the system behavior, we adopt a simple queueing model that has been used in the telephony network to derive the collision probability.

In the system model, a forwarding identifier is considered as a telephone line, and so it is occupied by a user who is arrived with the rate of λ_i and holds the resource with the rate of μ_i. There are total k forwarding identifiers, and q buffer spaces where a user's request waits for a forwarding identifier when it is not available. The ratio C between λ and μ shows the offered load to the system, e.g., $C = (\lambda/\mu)$. Equ. (1) shows the collision probability that a request for a forwarding identifier experiences a collision, e.g., request rejection. We omit its derivation due to space limitation.

$$P_c = \frac{C^k}{k!}\left(\frac{C}{k}\right)^q \left[\sum_{i=0}^{k-1}\frac{C^k}{k!} + \frac{C^k}{k!}\left(\frac{1-\left(\frac{C}{k}\right)^{q+1}}{1-\left(\frac{C}{k}\right)}\right)\right]^{-1} \quad (1)$$

[8]Here the locator means that the identifier of ICN-SW (caching server) or ICN-GW (ICN server) which can be identifiable in the topology and so the routing module can use the information to setup a route for data retrieval.

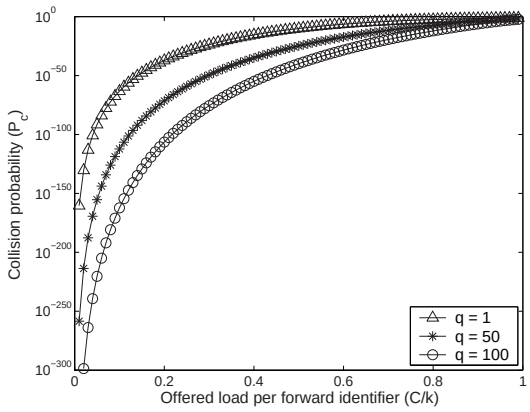

Fig. 4. Collision probability P_c as the offered load per forward identifier increases.

Fig. 4 plots the collision probability as a function of offered load per forward identifier (C/k). C/k being equal to 0.2 implies that the time interval between two consecutive requests for a forwarding identifier is five times larger than the time interval that the forwarding identifier is being occupied. This primitive result suggests two possible approaches to deal with the shortage of forwarding identifiers: increase buffer size and reduce the holding time of each forwarding identifier by users. This is a starting point to understand its problem, which will be further explored in our future works.

VIII. Conclusion

We have designed an ICN architecture within the framework of SDN to federate networks, e.g., sensor networks. ICN is expected to provide a caching and processing infrastructure for networks that have limited storage and processing powers. Moreover, the awareness feature of ICN enables the infrastructure to carry out traffic engineering in the level of content flow - fine granularity.

Two design issues have been considered: awareness and compatibility. To address the former design consideration, we introduced a new short fixed length identifier called a forwarding identifier that enhances the matching capability of openflow through its implementation at the hardware with a classical parallelization process. For the latter design issue, we embedded the forwarding identifier in the source/destination field of the UDP packet and so even legacy openflow switch (v1.0) can identify the data flow.

References

[1] T. Koponen, A. Ermolinskiy, M. Chawla, K. H. Kim, I. Stoica, B. gon Chun, and S. Shenker, "A data-oriented (and beyond) network architecture," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '07)*, Kyoto, Japan, August 2007.

[2] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking Named Content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies (ACM CoNEXT '09)*, Rome, Itary, December 2009.

[3] C. Dannewitz, "NetInf: An Information-Centric Design for the Future Internet," in *Proc. 3rd GI/ITG KuVS Workshop on The Future Internet*, Munich, Germany, May 2009.

[4] M. Ain et al, "D2.3 - Architecture Definition, component Descriptions, and Requirements," Deliverable, PSIRP 7th FP EU-funded project, Feb.," 2009.

[5] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Commun, Mag.*, pp. 26–36, 2012.

[6] S. Eum, K. Nakauchi, M. Murata, Y. Shoji, and N. Nishinaga, "CATT: Potential Based Routing with Content Caching for ICN," in *Proceedings of the SIGCOMM 2012 ICN workshop*, Helsinki, Finland, August 2012.

[7] ICNRG: Information Centric Networking Research Group. [Online]. Available: http://irtf.org/icnrg/

[8] M. Hirabaru, M. Inoue, H. Harai, and et. al, "New Generation Network Architecture AKARI Conceptual Design (ver. 1.1). AKARI Architecture Design Project," in *http://akari-project.nict.go.jp/eng/concept-design/AKARI_fulltext_e_translated_version_1_1.pdf*, October 2008.

[9] Cisco Visual Netowrking Index: Forecast and Methodology, 2010-2015. [Online]. Available: http://www.cisco.com/

[10] B. Raghavan, M. Casado, T. Koponen, S. Ratnasamy, A. Ghodsi, and S. Shenker, "Software-defined Internet Architecture: Decoupling Architecture from Infrastructure," in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, Redmond, Washington, USA, October 2012.

[11] D. Syrivelis, G. Parisis, D. Trossen, P. Flegkas, V. Sourlas, T. Korakis, and L. Tassiulas, "Pursuing a Software Defined Information-centric Network," in *Software Defined Networking (EWSDN), 2012 European Workshop on*, Darmstadt, Germany, October 2012.

[12] B. J. Ko, V. Pappas, R. Raghavendra, Y. Song, R. B. Dilmaghani, K.-w. Lee, and D. Verma, "An information-centric architecture for data center networks," in *Proceedings of the second edition of the ICN workshop on Information-centric networking*, Helsinki, Finland, August 2012.

[13] S. Salsano, N. Blefari-Melazzi, A. Detti, G. Morabito, and L. Veltri, "Information centric networking over SDN and OpenFlow: Architectural aspects and experiments on the OFELIA testbed," *Computer Networks*, pp. 3207–3221, 2013.

[14] O. M. M. Othman and K. Okamura, "Design and Implementation of Application Based Routing Using OpenFlow," in *Proceedings of the 5th International Conference on Future Internet Technologies*, Seoul, Korea, June 2008.

[15] Y. Sakurauchi, R. McGeer, and H. Takada, "OpenWeb: Seamless Proxy Interconnection at the Switching Layer," in *Proceedings of the First International Conference on Networking and Computing*, Hiroshima, Japan, November 2010.

[16] A. Chanda and C. Westphal, "ContentFlow: Mapping Content to Flows in Software Defined Networks," *arXiv preprint arXiv:1302.1493*, 2013.

[17] PURSUIT: Pursuing a Pub/Sub Internet. [Online]. Available: http://www.fp7-pursuit.eu/

[18] A. Detti, N. B. Melazzi, S. Salsano, and M. Pomposini, "CONET: A Content Centric Inter-Networking Architecture," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, Toronto, Ontario, Canada, August 2011.

[19] OFELIA project. [Online]. Available: http://www.fp7-ofelia.eu

[20] IETF RFC 5280, Internet X.509 Public Key Infrastructure Certificate Revocation List (CRL), 2008.

[21] J. Moy, "OSPF Version 2," *RFC 2328*, April 1998.

[22] P. Radoslavov, R. Govindan, and D. Estrin, "Topology-Informed Internet Replica Placement," in *Proceedings of Sixth International Workshop on Web Caching and Content Distribution*, Boston, USA, June 2001.

[23] E. Cronin, S. Jamin, C. Jin, A. R. Kurc, D. Raz, and Y. Shavitt, "Constrained Mirror Placement on the Internet," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 1369–1382, 2002.

[24] S. Jin and L. Wang, "Content and service replication strategies in multi-hop wireless mesh networks," in *Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, Montreal, Canada, October 2005.

[25] A. K. Nayak, A. Reimers, N. Feamster, and R. Clark, "Resonance: Dynamic Access Control for Enterprise Networks," in *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*, Barcelona, Spain, August 2009.