Supporting Information-Centric Networking in SDN

Peng Li, Wu Muqing, Wang Ning, and Liu Hongbao

***Abstract*—Recently the Software-Defined Networking (SDN) has attracted increasing attention in the research community around the world. In this paper, we focus on how to modify the OpenFlow protocol, the most popular SDN instantiation, and use SDN functions to enable Information-Centric Networking (ICN). We describe the design based on CCNx (a virtual experiment network based on the concept of ICN), Floodlight (a kind of SDN controller) and a revised Open Vswitch (a virtual switch in SDN). We build a testbed that i) allows to identify the content requests, ii) realizes the routing process of ICN in SDN, iii) enables efficient content delivery, iv) is able to be deployed in existing IP networks. In addition, the proposed paradigm continues to support traditional host-to-host communication using TCP/IP. We will present how Open Vswitch should be altered to better match the Operations of CCNx and how CCNx can be implemented over the modified OpenFlow Network.**

***Index Terms*—SDN, ICN, CCNx, OpenFlow.**

I. Introduction

Most of Internet users care more about the desired contents rather than their physical addresses of the hosts. In traditional TCP/IP network, communication is based on interface addresses of both hosts that packets are sent from and transmitted to, Information-Centric Networking (ICN) shifts the communication mode to a content-to-user communication paradigm. ICN supports efficient forwarding of content by exploiting a name-based routing protocol and placing cache in appropriate nodes of network [1]. Although ICN is an advanced architecture, it is still an experimental network which is hardly to deploy due to its very different underlying structure. As a result, how to deploy ICN in current Internet has become a problem.

Software-Defined Networking (SDN) paradigm is considered a good solution to develop new applications in the network. SDN technology is based on the concept of decoupling the control plane and data plane, a centralized controller directs the management and configuration of various network devices through a standardized interface, which offers more possibilities to design and flexibly use of network resources. The programmable applications on controller abstract the underlying network infrastructure so that the data plane only need to be responsible for general forwarding function, hence the innovation of network technology will transfer to the control plane. This separation is to increase the flexibility of network interconnection equipment control, it also provides great openness and programmability that will vigorously promote the pace of network innovation. SDN enable the development of new routing and forwarding strategies with just programming. Due to these advantages mentioned above, SDN is generally considered as a viable solution to ICN deployments.

The only way to meet the requirements of various services in the traditional IP network is to increase the redundant computing resources and service deployments, which tremendously reduce the utilization of network resources and aggrandize the complexity of network architecture. Nevertheless, SDN can satisfy the above-mentioned demands since it actualizes control logic according to the complete network state information maintained by controller. By introducing the core concept of SDN into ICN architecture, a network paradigm that is easy to supervise and has good support for content distribution business is implemented. The advantages of ICN and SDN makes them bound to play a crucial role in future network, hence the continually research on ICN over SDN is of great importance.

In this paper, we represent the structure of a revised flow table in Open Vswitch. The modification refers to the CCNx protocol [2], which makes it possible to deploy ICN over a SDN framework. Hence, the rest of our paper is structured as follows. Section II introduces the background on ICN and on OpenFlow. In Section III, we explain our ICN over SDN solution and show the modified OpenFlow and the process of handling ICN messages in detail. Section IV specifically describes our preliminary realization of the approach using Floodlight SDN framework and CCNx. We conclude our work and point out our deficiency in Section V.

II. Background

In this section, the basic idea and knowledge of ICN and SDN are elaborately introduced.

A. Information-Centric Networking

Nowadays, Internet consists of multiple Autonomous Systems (ASs) which interconnect with each other all over the world. The huge scale of addresses and BGP routes are based on prefix and route aggregation. Some time ago, this way of communication could meet the public demand. However, the increasing requirement for highly scalable Network Data Objects (NDOs), such as web pages, videos, documents and other pieces of information, has promoted the development of future network towards supporting efficient distribution of data. It is obvious that ICN is proposed in such condition.

The ICN approach makes use of in-network caching, multiparty communication through replication, and interaction models decoupling senders and receivers. The goal is to provide a network infrastructure service that is better suited to today's use and more resilient to disruptions

Manuscript received September 5, 2015; revised November 11, 2015.

The authors are with the Beijing University of Posts and Telecommunications, China (e-mail: pengli1107@bupt.edu.cn, wumuqing@bupt.edu.cn, wangning321@bupt.edu.cn, liuhongbao@bupt.edu.cn).

and failures [3]. However, a number of crucial issues and challenges related to name based routing are yet to be addressed in order to successfully realize a content oriented networking model for the future Internet. The crux of these problems is that ICN requires Internet routers to maintain a large amount of routing state (for the content names are not as aggregatable as IP addresses), which seems impossible with current technology.

B. Software-Defined Networking

To support new network protocols without destroying industrial chain of the legacy IP network, SDN supports user programmability and controllability by decoupling the data forwarding plane from the control plane. The SDN-based approaches have been deployed in some areas such as Data Centers and Cloud Computing.

In SDN, the management functions are stripped from the physical switching/routing equipments, and be realized by programming on the terminals. As a result, the physical switching equipments will be simplified as "no-brainer" controlled by programmable software.

After years of research and exploration, the Open Network Foundation (ONF) puts forward the SDN architecture [4]. SDN is divided into the Infrastructure Layer, the Control Layer and the Network Applications. Softwares in the Control Layer interact with the Infrastructure Layer through the Southbound Interface, it also interacts with the applications through Northbound Interface (Open APIs). OpenFlow is one of the Infrastructure Layer deployment scheme, and the corresponding Southbound Interface operates the OpenFlow protocol. OpenFlow switching is comprised of flow table, secure channel and OpenFlow protocol [5]. Fig. 1 shows the OpenFlow pipeline in Open Vswitch. A flow table consists of several flow entries, each flow entry contains match fields, counters and instructions. The flow tables of an OpenFlow switch are serial numbered, starting at 0. When a packet arrives, it first matches against entries of flow table 0, if the packet matches a flow entry in flow table, the corresponding instructions in the flow entry will be executed. At the same time, the action set is updated. The instructions may explicitly direct the packet to another flow table, where the same process is repeated again (the instruction is named Goto). If the matching flow entry does not direct packets to another table, the pipeline processing stops at this table, and the packet is usually forwarded on the basis of the action set. If the packet does not match any flow entry in a flow table, the default behavior is to send packets to the controller via the secure channel, another option is to drop the packet. The packet processing can also be set to continue, in this case the packet is processed by the next sequentially numbered table.

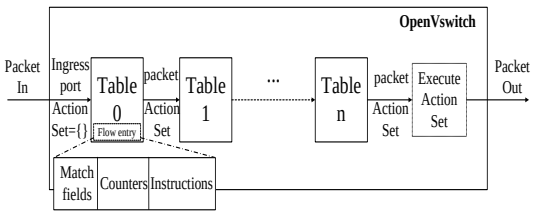

Fig. 1. OpenFlow pipeline.

C. Related Works

Recently, there have been several implementations of combining SDN with ICN. N.Blefari-Melazzi et al. proposed the coCONET framework [6], they utilize SDN for the realization of an ICN architecture called CONET. The coCONET supports content delivery and caching in a name-based manner. It leverages the programmability of OpenFlow to process packets according to the content names. However, coCONET cannot provide the name-based content delivery service to legacy IP hosts.

Dimitris Syrivelis *el al*. proposed an OpenFlow framework which providing name-based routing by using the Publish/Subscribe system [7]. It simplifies the ICN node and network architecture by using LIPSIN identifiers. But it cannot support name-based caching and efficient content delivery.

Dukhyun Chang *el al*. proposed a SDN-based content delivery framework called C-flow [8], which supports name-based routing and caching. The SDN functionalities in C-flow enhance the efficiency of content deliveries by dynamic re-routing, parallel transmission and cache management. But C-flow is different from our strategy, we put forward an Open Vswitch revision plan and realize it in our testbed. Our scheme is able to reduce the heavy load of the controller.

III. Proposed Approach

In this section we first describe the origin of our thoughts. We revise the flow table and flow entry depending on the forwarding process of CCNx. Then we will introduce our architecture in detail.

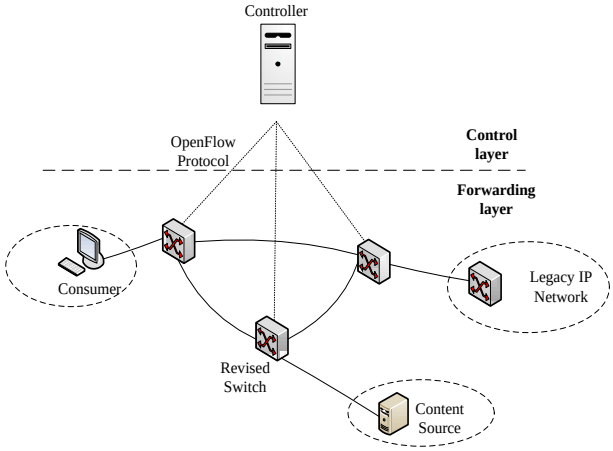

Fig. 2. Overall architecture.

Our target is using the SDN mechanism to operate the communication process of ICN, that is to detect the CCNx requests in our testbed and to forward these requests and corresponding response messages. SDN sets up forwarding rules through generating a flow entry whose match fields matching the coming IP packets. Current OpenFlow Specifications allow to identify TCP/UDP ports, but it is difficult to deeply analyze the payload of the packets, as it is important for efficient transport of CCNx packets. Therefore, the heart of the matter is how to modify traditional OpenFlow protocol so as to deal with non-IP packets, such as the CCNx content packets we use. In order to run CCNx over OpenFlow,

we need to revise the structure of flow entry and flow table, meanwhile the new-type SDN still supports the IP packets forwarding. When a packet arrives at a face, Open Vswitch needs to identify the packet type and transmits the packet in accordance with the corresponding forwarding process.

The network architecture is logically devided into content forwarding layer and control layer, as shown in Fig. 2. The content forwarding layer contains the revised Open Vswitches, content sources, consumers and legacy IP nodes. We choose Floodlight as our control layer, which controls the forwarding process of Interest packet and IP packet in the content forwarding layer. The control layer also has the function of cache management and disturbance switching. These two layers communicate with each other through the OpenFlow protocol.

A. CCNx Forwarding Process

Communication in CCNx is driven by the consumers. There are two types of packets in CCNx: Interest and Data.

In order to get a Data, the consumer broadcasts an Interest packet, which contains content name (such as / home/ pic/ abc.jpg) that identifies the requirement. A router marks the interface from which the Interest packet comes in, and then a longest-match lookup on content name will be done [9]. The lookup process is ordered so that Content Store (CS) match will be preferred over Pending Interest Table (PIT) match which will be preferred over Forwarding Information Base (FIB) match.

If there is a Data packet in the CS that matches the Interest, it will be sent out where the Interest arrives.

Otherwise, if the Interest matches PIT, its arrival interface will be added to the corresponding PIT entry. (It means an Interest in the same Data has already arrived, when a consumer requests the Data packet, a copy of the packet will be sent out according to the written face.)

Otherwise, if the Interest matches FIB, it should to be sent towards the Data. If the outgoing face is not empty, the arrival face is removed from the FIB entry, and a new PIT entry is created depending on the Interest and its arrival face.

B. Open Vswitch Revision

We revise the structure of Open Vswitch to realize CCNx over OpenFlow. On the basis of CCNx forwarding process, we append three tables to the original flow table. These three tables correspond to the CS, PIT and FIB in CCNx forwarding process.

In Section II-B, we mention that each flow entry contains match fields, counters and instructions. The original match fields in the flow entry are listed in [5]. It shows the fields which an incoming packet is compared against. Each entry contains a specific value in order to match a certain packet. The switch should apply the instructions and update the associated counters of the highest-priority flow entry matching the coming packet.

Fig. 3 depicts our flow table structure and the modified flow entry in the switch. Fig. 4 contains details of the fields in our flow entry. The content name is transformed into a hash value, which is used to identify a certain Data packet. The consumer sends an Interest packet that includes a hash value. After a series of forwarding, the Interest packet is responded by a content source that suits the needs. The content source returns the required Data packet which has the same hash value and data payload as shown in Fig. 5.

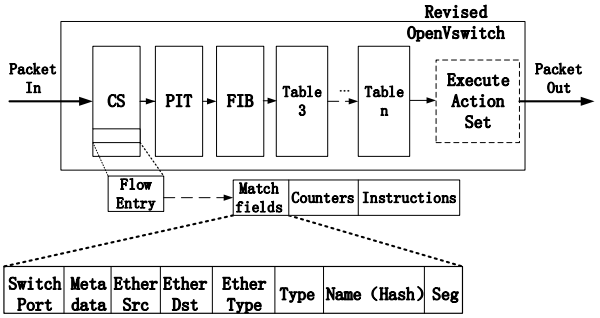

Fig. 3. Revised OpenFlow pipeline and match fields.

Field	Bits	When applicable	Function
Switch Port	32	All ICN packets	Record the incoming port (in CS and PIT) and forwarding port (in FIB)
Metadata	64	Table 1 and above	Pass information between tables
Ethernet source address	48	All packets on enable ports	Record the MAC address of consumers
Ethernet destination address	48	All packets on enable ports	Record the MAC address of content sources
Ethernet type	16	All packets on enable ports	Ethernet type of packet payload
Packet type	8	All ICN packets	Recognize the type of ICN packet
Content name	128	All ICN packets	Record the hashed name of ICN packets
Segment	64	All ICN packets	Record the number of ICN segmented packets

Fig. 4. Fields lengths and the way they must be applied to flow entries.

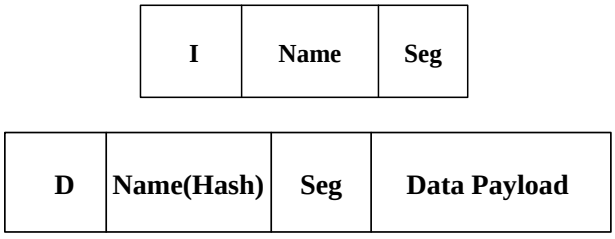

Fig. 5. Interest packet (above) and data packet (below).

C. Interest Packet Forwarding

The switch starts by performing a table lookup in the first table, and likely to perform table lookup in other flow tables following behind according to the last result. An Interest packet matches a flow table entry if the values in the match fields used for the lookup match those defined in the flow table. (Besides, if a flow table field has a value of ANY, it matches all possible values in the header.)

If the packet is a legacy IP packet, it cannot match any flow entry in the CS table (as the first table in pipeline process), the packet should be processed based on our table configuration. We make it be sent to the table behind FIB table (the packet will be sent to flow Table III). The packet starts to look for the most matching flow entry in flow Table III. If such an entry exists, the packet applies the instructions by which the action set is updated. The process is repeated until the last matching entry do not have a Goto Instruction, and the packet is

processed with its associated action set and usually forwarded to another switch. In this case, the matching process is the same as the original matching process of Open Vswitch.

If the packet is an Interest packet, it first looks for a highest-priority flow entry in the CS table. If the type, name and seg fields are completely matched, the Load content (an extended instruction) should be executed, and the content stored in the cache is forwarded through the corresponding switch port.

If the Interest does not match the CS table, it should look up the PIT table. If the switch port, type, name and seg fields are matched, it means that this port has already received a same Interest, and the Interest packet ought to be dropped. If only the switch port field is unmatched, it means that a same Interest has already received by another port of the switch. In this case, the switch generates a new PIT entry in which records the corresponding port, and the Interest packet will be dropped.

If the Interest packet does not match the PIT table, it should refer to the FIB. (The switch generates a new PIT entry recording the informations of this Interest beforehand.) The FIB entries are generated by the routing component (we name it the FIB Constructor) in the controller. Only type and name fields will be matched. If there is a matching entry in FIB table, the Interest packet should be forwarded to another switch in accordance with the switch port field of the FIB entry. If there is no matching entry, the Interest should be dropped.

D. Data Packet Forwarding

The Interest packet repeats the above-mentioned matching process until it is dropped. If the CS table of a certain switch is matched (the switch can be considered as a content source), the Data packet is extracted from the cache and straightly forwarded. The Data is forwarded to the previous switch where the Interest comes and then looks up the PIT table.

According to the switch port field of all matched PIT entries, the Data is replicated and forwarded. The process is repeated until the requested Data packet reaches all consumers. A content request may be satisfied by the cache of any switch along the path that Interest packet passes by.

IV. Implementation and Evaluation

In this section we first provide an overview of our initial prototype implementation. Then we briefly report on our evaluation of the approach. Our implementation is based on Floodlight and revised OpenFlow over Open Vswitch.

A. Implementation

The Floodlight controller [10] is extended to support the OpenFlow-based Information-Centric Networking. Fig. 6 shows the reference model of our controller for this framework. The newly added functionalities consist of Topology Manager, Cache Manager and FIB Constructer.

Topology Manager receives the link status from Open Vswitch by using the OpenFlow protocol, and it also detects any link problems in the network to maintain the network topology.

Cache Manager controls the replications of Data packet and keeps track of the locations where content is cached. It exchanges messages with all switches in the network to synchronize the cache list. The cache placement strategy is deployed by Cache Manager. The controller decides the cache placement nodes according to nodes' degree.

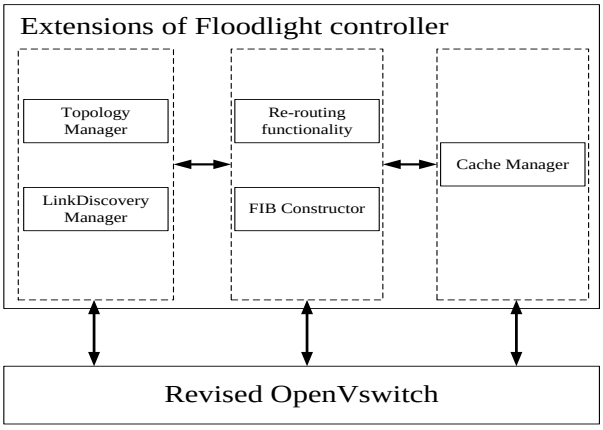

Fig. 6. The reference model of Floodlight.

Fig. 7 shows how the FIB constructor works. Content sources advertise their contents to the Floodlight. FIB Constructor calculates the optimal paths between consumers and locations which contain these contents. (Controller contains the global topological information so that it calculates the paths easily based on minimum hop routing algorithm.) Then FIB constructor advertises the FIBs to all switches under its control. The FIB tables are successfully constructed with these steps. The Re-routing functionality is activated when the LinkDiscovery Manager detects a link problem. Then it calculates an alternative path (or an alternative content source if it needs) and updates the FIB tables of the relevant switches.

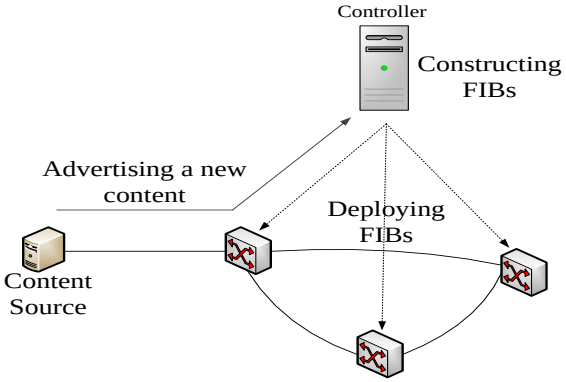

Fig. 7. Content advertisement and FIB construction.

Open Vswitch is also extended for caching and retrieving the Data packets. We defined new actions called "OFPFC_CACHE" and "OFPFC_RETRIEVE". If the former action is invoked, the Data packet is stored into the cache by the switch, which creates the corresponding CS entry at the same time. The latter action is invoked when the CS entry is matched. Then the switch forwards the Data from the cache to the incoming port of Interest.

B. Evaluation

Our prototype shows some benefits represented in Section II-C. ICN is difficult to be deployed on existing IP network, we use SDN to support the complex communication between

ICN nodes. As a result of in-network caching and route-by-name approach, ICN realizes efficient content delivery. Our prototype provides the possibility to resolve current IP problems such as traffic explosion and security in a highly feasible network.

To analyze the applicability of our concept, we set up an experimental network environment to run some traffic with our ICN over SDN approach in comparison to ICN overlay approach (CCNx). Fig. 8 shows the variation of download time during the simulation. The content download time in our approach is distinctly shorter than that using a flooding mechanism in the CCNx.

Although this is a result in a small experimental platform, it demonstrates that our approach is viable in the legacy IP network and superior to some extent. However our result is primary and we plan to apply various cache placement strategies, to further refine the network performance.

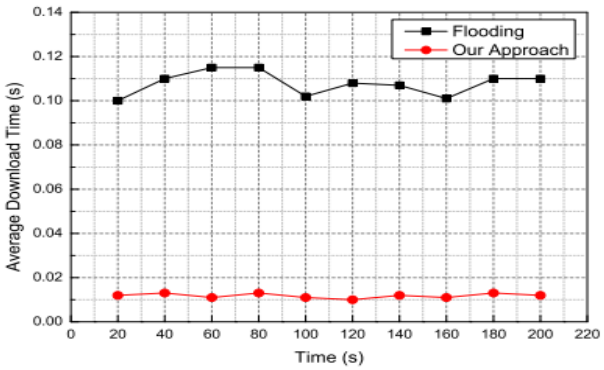

Fig. 8. Average download time.

V. Conclusion and Future Work

In this paper, we propose a mechanism to deploy ICN over SDN in IP networks. Crucial here is the revision of flow tables in the Vswitches, which we refer to the packet forwarding process of CCNx. The revision makes it possible to achieve efficient content delivery in a existing IP network. In addition, content sources advertise new contents to controller to calculate the FIBs, avoiding the flooding route in ICN. This design works out the network congestion in ICN.

We have tentatively demonstrated the practicability and benefits of our approach, but the experimental environment is a miniature network. We do not assume the result will be a common case when the network becomes a large-scale one because of the content advertising process. All content sources should advertise their contents to controller, and then controller calculates the FIBs. As future work we intend to utilize multiple controllers to monitor a large-scale network collaborately, and to extend the evaluation in more practical framework.

Acknowledgment

This work has been supported by the National Science and Technology Major Projects of China under Grant No.2011ZX03001-007-03 and by the Fundamental Research Funds for the central universities under Grant No.G470527.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Trans. Communication Magazine*, vol. 50, no. 7, pp. 26-36, 2012.
[2] CCNx OpenSource project. (2014). [Online]. Available: http://www.ccnx.org/
[3] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. 5th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, pp. 1-12, 2009.
[4] Open Networking Foundation. (2014). [Online]. Available: https://www.opennetworking.org/
[5] OpenFlow Switch Specification, Version 1.1.0. (2011). [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.1.0.pdf/
[6] N. Blefari-Malazzi, A. Detti, G. Mazza, G. Morabito, S. Salsano, and L. Veltri, "An openflow-based testbed for information centric networking," *Future Network Mobile Summit (FutureNetw)*, pp. 1-9, 2012.
[7] D. Syrivelis *et al*., "Pursuing a software defined information-centric network," *Software Defined Networking (EWSDN)*, pp. 103-108, 2012.
[8] C. Dukhyun, M. Kwak, N. Choi, T. Kwon, and Y. Choi, "C-flow: An efficient content delivery framework with OpenFlow," in *Proc. 2014 International Conference on Information Networking (ICOIN)*, pp. 270-275.
[9] *Named Data Networking Project*, Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC, 2010.
[10] Project Floodlight. (2014). [Online]. Available: http://www.projectfloodlight.org/floodlight/.

Peng Li was born in November 1991, and received his Bachelor's degree in telecommunication engineering from Beijing University of Posts and Telecommunications, P.R.China, in 2013. Now he is a master student in Beijing University of Posts and Telecommunications, P.R. China. His main research interest includes future network of cache distribution and routing mechanism.

Wu Muqing was born in July 1963. He is a Ph.D. professor of Beijing University of Posts and Telecommunications (BUPT), senior membership of China institute of communications. Now, he was interested and researching in ad hoc wireless network, UWB, high-speed network traffic control and performance analysis, GPS locating and services, teaching basic theories of telecom networks.

Wang Ning received her bachelor's degree in telecommunication engineering from Beijing University of Posts and Telecommunications, P.R.China, in 2013. Now she is a master student in Beijing University of Posts and Telecommunications, P.R. China. Her main research interest includes future network of cache distribution and routing mechanism.

Liu Hongbao born in August 1989, and received his bachelor's degree in telecommunication engineering from Xidian University, in 2011. Now he is a master student in Beijing University of Posts and Telecommunications, P.R. China. His main research interest includes future network of cache distribution and routing mechanism.