International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 11 Issue: 04 | Apr 2024 www.irjet.net p-ISSN: 2395-0072

Directed Transmission Path Strategy on SDN-Based Content Centric Networks for Efficient Caching

San-Yuan Wang,[1], Chi-Chun Chen[2], Tain-Lieng Kao[3*]

[1]Associate Professor, Department of Information Engineering, I-Shou University, Taiwan, R.O.C.
[2]Software Development Engineer in Test, FINFARE INC. 17192 Murphy Ave Irvine, CA 92623, USA
[3]Assistant professor, Department of Intelligent Network Technology, I-Shou University, Taiwan, R.O.C.*
Corresponding Author

Abstract *- With the development of technology, the Internet has become inseparable from life. The development of the 5G application and the core network architecture of the 5G will start to use the Software-Define-Network (SDN) architecture for transmission. Google Data Centers have already begun to develop and use the SDN to gain network bandwidth usage. The traditional VLANs will gradually migrate to network virtualization because excessive data traffic will inevitably lead to network saturation. Therefore, major equipment vendors have begun to focus on SDN related research, and hence data center related topics possess great potential of development. Caching is one of the most important topics. Attention will be on how to balance the cache hit rate and node replacement rate, and furthermore, to speed up the network and to make the transmission more efficient. Therefore, this research proposes a direct transmission mechanism, which can quickly confirm the connection information in the initial stage, find a better transmission path, and improve the cache hit ratio. Through exhausted simulation, the research finding include the relationship between network speed and cache hit ratio, so that the speed can be faster and more accurate in the initial stage and reduce the waste of cache resources or network performance degradation due to excessive network coverage can be avoided. From these experimental results the network speed can be improved by 5% in the initial transmission, and proposed strategy can make the cache hit ratio increase by 3-5%.*

Key Words: SDN, CCN, ICN, Cache, Controller, Switch

1. INTRODUCTION

With the flourishing development of cloud computing, mobile networking, as well as the Internet of Things (IoT) and the Internet, in today's network environment, network traffic must be heavily loaded. It's no longer based on a single server as in the past, but is gradually transitioning to multiple servers or a vast number of virtual networks to serve as bridges for communication between each other. To address the surge in network traffic today and to quickly allocate network resources based on traffic volume, as well as to make frequent and rapid configurations, the concept of the SDN architecture was introduced at the IEEE Infocom conference in 2009[19]. This concept was further exemplified by the OpenFlow technology [3][7], Google utilized OpenFlow in 2012 to create an SDN architecture for its datacenter, which increased Google's network utilization rate from 30% to 95% in one fell swoop. This substantial improvement made various network equipment manufacturers realize the importance of SDN and further motivated them to develop products related to SDN [5].

The most widely used aspects of the internet today include content access. According to Cisco Annual Internet Report (2018–2023) [22], video and other applications continue to be of enormous demand in today's home, but there will be significant bandwidth demands with the application requirements of the future. Consequently, network virtualization has become an inevitable trend. However, the internet operates on an IP protocol and a host-to-host communication model, where content delivery was not part of its initial architecture. Therefore, several Information-Centric Networking (ICN) [25] proposals have begun to emerge, offering alternatives to the traditional TCP/IP communication architecture, such as Content-Centric Networking (CCN) [24].

Among those services and applications of the Internet, the most extensively used are video streaming and P2P usage. CCN, however, proposes a model for content by storing packet addresses and non-host content along the delivery path, known as network caching [1][2]. The advantage of this approach is that any node in the network can make requests, leading to massive caching that could potentially overload the network, causing it to become paralyzed or inefficient, and wasting network resources, thereby reducing overall network performance.

As the internet continues to develop, traditional network architectures are increasingly unable to handle the growing volume of network traffic. This challenge has led to the emergence of SDN architectures. Within an SDN framework, there's a separation between traffic forwarding and routing decisions in the network. An SDN Controller manages switches and routers, effectively transforming these nodes into forwarding devices. Through the OpenFlow communication protocol, these nodes can

communicate with the Controller, enhancing network management and efficiency.

Most of the research on CCN so far has focused on caching, with only a minority addressing routing issues. Given CCN's multifaceted nature, it lacks a clear routing scheme, which is one of the main challenges for future deployments since network devices must adapt to changing demands. However, these few studies still rely on the TCP/IP stack. Therefore, proposals like NetSoft's aim to implement CCN within an SDN architecture, employing a clean-slate approach [23] without depending on TCP/IP to increase network traffic utilization.

In existing CCN schemes, there's a trade-off exemplified in caching strategies: if the cache hit rate is high, the node replacement rate is also high, which can lead to a significant load in a short period. Conversely, reducing the node replacement rate may result in a lower cache hit rate, presenting a dilemma between efficiency and performance [15][16].

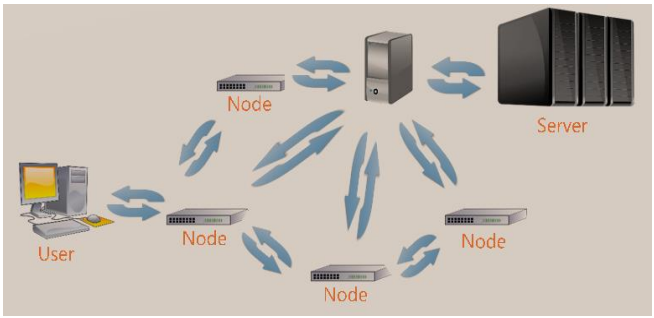


Figure 1. LCE CCN transmission strategy

Figure 1 shows the transmission strategy of Leave Copy Everywhere (LCE) CCN [20]. When user issues request to the controller, it will flood through the whole networks. Let all nodes on the path cache the request. Other nodes also transfer their requests to the controller. Then the controller issues command to let all nodes send data back to the controller, as shown in Figure 1. This strategy will cause huge cache data in continuous mass transmission, increasing the network resource waste and decreasing the performance [6].

To achieve efficient caching, we have designed a direct transmission strategy, allowing for the rapid identification of optimal paths in a single network, thereby reducing the use of cache resources. We modified the controller to change the linkage paths among nodes, servers, and users, enabling direct communication between users and servers through the controller. The controller filters the nodes along the path, increasing the cache hit rate and reducing the node replacement rate due to the optimization of the path.

Finally, the performance differences are compared using data collected from traditional CCN without the direct transmission strategy and the new approach that utilizes the direct transmission strategy, against a baseline of general network transmission modes. This comparison highlights the efficiency gains achieved by our proposed method.

We constructed a system where users connect and send requests to the controller, which then communicates with servers. The controller centrally manages and identifies the optimal transmission path to make communication between users and servers more efficient. It also determines which nodes along the best path to cache, moving away from the traditional Least Recently Used (LRU) caching strategy to reduce the rate of node replacement and increase the cache hit rate. This approach leverages the advantages of centralized control within an SDN framework to optimize network performance by carefully selecting cache locations based on the flow of traffic, thereby enhancing the overall efficiency of network resource utilization.

When a user sends a request to the controller, it passes through all the nodes within the domain, necessitating that all nodes cache the data. After all the nodes receive these requests, they relay the message back to the controller, which then issues a command for all nodes to return the data or information to the user.

This round-trip method leads to a significant increase in caching, which in turn causes unnecessary resource wastage and reduces the efficiency of network resources. The extensive caching required by each node in response to a single user request can greatly inflate the network's cache storage requirements, leading to inefficiencies and potential degradation in network performance as it tries to manage the increased load and maintain the cache across all nodes.

The rest of the paper is organized as follows. Section 2 serves as the focal point of the text, detailing the setup of the experimental environments and the methods of improvement. It discusses how the proposed hypotheses were tested through simulation experiments, including network topologies, Controllers, and Switches, to find the shortest path. The aim was to enhance the cache hit rate and reduce the node replacement rate through adjustments in the environmental setup and program architecture. Section 3 describes the experimental environment configuration and the stages of the experiment. Section 4 analyzes the parameters and data obtained from the experiments. It summarizes the issues that may have arisen during the experimental process due to certain factors or methods used. Section 5 concludes the research by summarizing the findings and the data collected. It discusses the contributions and value of the

work and plans for future development, reflecting on how the processes undertaken and findings might influence further advancements in this field.

2. THE PROPOSED DIRECT TRANSMISSION PATH METHOD

The primary objective of this section is to describe the direct transmission mechanism (DT) which constructs a virtual SDN environment that leverages a Controller to manage the allocation of resources between users and servers. By directing traffic through switches or nodes, the system aims to identify the shortest path for faster communication connections. This setup seeks to bypass peripheral or infrequently used nodes, allowing various users or switches to act as nodes within the network. This approach provides a distinct networking environment from the traditional TCP/IP transmission modes, facilitating a more efficient and optimized communication pathway between users and servers.

In doing so, this architecture not only enhances the efficiency of data transmission but also improves the overall network performance by dynamically adjusting the flow of traffic based on current network conditions. This adaptability ensures that network resources are utilized in the most effective manner, contributing to the scalability and flexibility of network infrastructures in accommodating the growing demands of modern internet usage.

2.1 Controller Collect Information between user and server

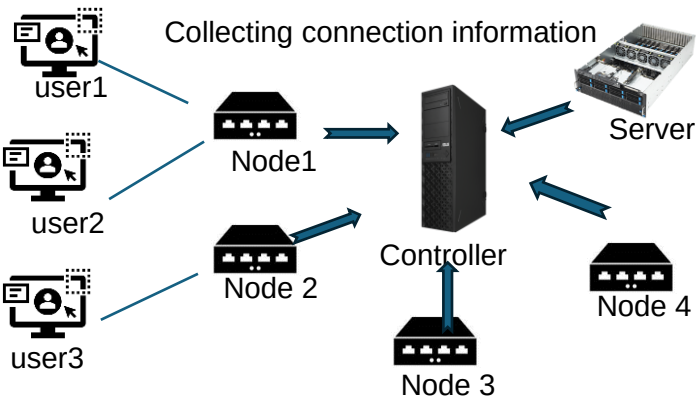


Figure 2. controller collect exchange information between user and server

Initially, the communication between users and servers is swiftly facilitated by the controller. This method utilizes the users' IP or MAC addresses for quick identification within the controller. Following this, the location of the server is pinpointed, allowing for the connection between users and servers to be pre-established within the controller, as shown in Figure 2.

By leveraging such an approach, the controller acts as an intelligent hub that efficiently manages network traffic. It streamlines the process of establishing connections by immediately recognizing devices based on their unique identifiers and mapping out the most efficient route to the server. This preemptive connection setup significantly reduces the time it takes for data to travel between the user and server, enhancing the responsiveness and overall user experience of the network.

Implementing a whitelist within the controller enables rapid identification of users' IP or MAC addresses. This system synchronously sends connection requests to the server, facilitating quicker alignment between the source and the destination.

By utilizing a whitelist, the controller can efficiently filter and prioritize legitimate devices and users, ensuring that only authorized entities are allowed to communicate through the network. This method not only enhances security by preventing unauthorized access but also optimizes the network's performance by reducing unnecessary connection attempts and streamlining the process of establishing communications. The immediate recognition and processing of connection requests from known and trusted devices expedite the data transmission process, resulting in a more efficient and responsive network infrastructure.

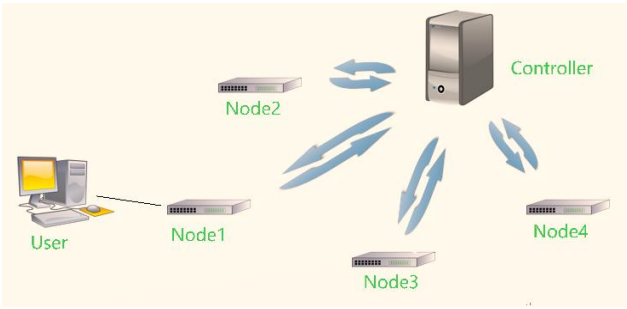


Figure 3. Controller centrally control nodes and exchange information

2.2 Centrally Control by Controller

When a user sends a request, the controller can quickly filter the nodes within the same domain to determine if there are any recently frequently used or popular data or information. It then confirms the transmission path and identifies a viable shortest path, which constitutes the direct transmission path strategy.

Figure 3 illustrates a centralized control approach, where the controller connects with each node in the network domain to analyze the resources, data, or information available at each node.

After the communication between the user and the server is completed and integrated into the entire domain,

based on the requirements and commands given by the user and server, the system filters out the data or information that meets the user's needs. It determines whether the nodes have previously cached or hold data that is popular or in high demand, aiming to synchronize the transmission of data to the user. This process also helps avoid unnecessary node replacement rates by ensuring that only relevant and efficient pathways are utilized for data transmission.

2.3 Optimal Transmission Path Computation

In Content-Centric Networking (CCN), both users and switches can act as nodes, allowing the controller to send commands to control these nodes within the domain. This enables the transmission of the data or information requested by the user, thereby shortening transmission times and reducing unnecessary network or cache resource wastage.

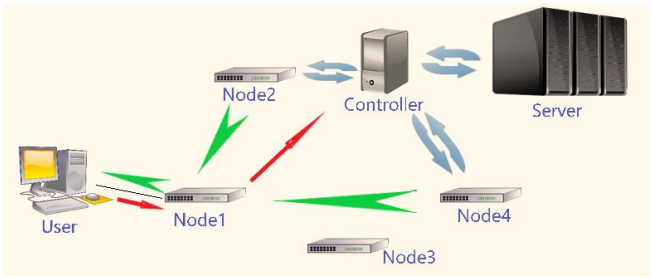


Figure 4. An example of direct transmission mechanism

As shown in Figure 4, after a user sends a request, it is directly transmitted to the controller via the connected nodes. The controller then synchronously sends a communication request to the server, completing the communication between the user and the server.

Upon analyzing the content of the request, if it is assumed that Nodes 2 and 4 have a cache of the content or it is highly popular, then the controller will command Nodes 2 and 4. Upon receiving the command, these nodes will transmit the data to the user's previous layer, Node 1, and synchronize the data transmission.

This process omits Node 3, which may not have the cache or whose content is less popular, thereby not increasing the node replacement rate and indirectly improving the cache hit rate.

Such an approach also reduces the server's performance burden when several users are transmitting the same content. If users within the same domain are transmitting different files, the controller will allocate network resources based on the cache and popularity of content relative to each user and node, thus enhancing overall network efficiency.

3. EXPERIMANTAL ENVIRONMENT

In this section, the experimental environment is firstly configured to construct a virtual SDN environment. Then, the detailed experiment design and the stages are stated.

1.1 Experimental Environment Configuration

After testing across various network topology environments, Mininet was chosen due to its support for OpenFlow, Open vSwitch, complex topologies, and customizable environment settings. It also offers a Python API for convenient multi-user collaboration, making it an excellent tool for environmental architecture.

Regarding the choice of controller, there are several options available, including OpenDaylight, OpenContrail, Floodlight, Flowvisor, and Ryu. Ryu was selected primarily for its simple architecture, cost-effectiveness in deployment, and suitability for small to medium-sized network architectures, making it convenient for traffic control and allocation.

1.1.1 Mininet

Mininet is a platform that allows the creation of virtual network topologies through the connection of virtual terminals, routers, switches, etc. The Mininet switches also support the OpenFlow protocol, enabling the creation of a Software-Defined Networking (SDN) capable local area network within a personal computer. Moreover, virtual Hosts can be utilized to simulate real computers for sending packets, allowing for comprehensive testing and development of network configurations and applications in a controlled, cost-effective environment. Figure 5 shows the steps to establish a network: (1) create 4 switches, s1,s2,s3, and s4, (2) add links between switches and hosts, (3) configure 9 hosts and one controller. After setting the network topology, one can use pingall command to let switches and hosts to exchange packages for verifying that the connections are work.

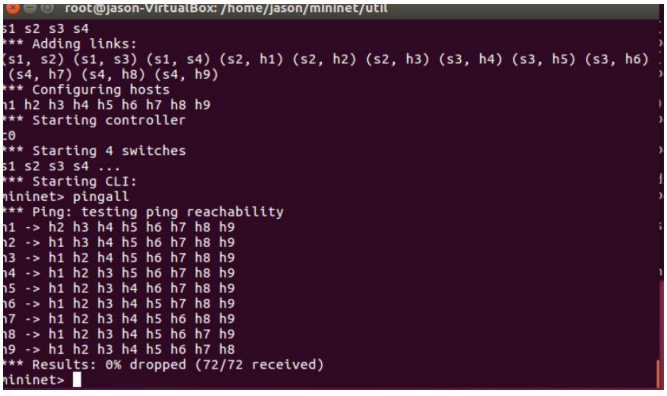


Figure 5. network topology setting and switches connecting by Mininet

1.1.2 RYU Controller

Ryu[28] is a development framework for SDN controllers, developed and designed by NTT in Japan. It includes the functionality of an OpenFlow Controller and encompasses most protocol, hence it is referred to as a Controller. In essence, its primary role is to act as a framework responsible for dispatching events. Within this framework, an application called Ofphandler is used to deliver OpenFlow messages through the framework.

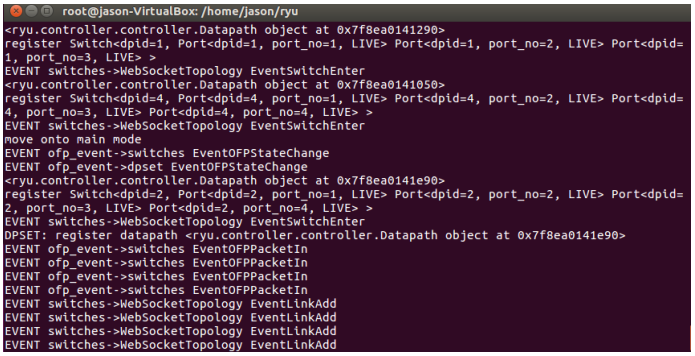


Figure 6. Datapath generated by Ryu Controller

The next layer is the Datapath: it is created by enclosing the Controller with the Ofphandler, and the Controller then generates and stores Datapath objects. The Datapath objects come into play when a new switch connects to the network. Each Datapath operates as an independent thread without interfering with others. When a switch sends an asynchronous message to it, it generates an Event object and sends it back to Ryu.

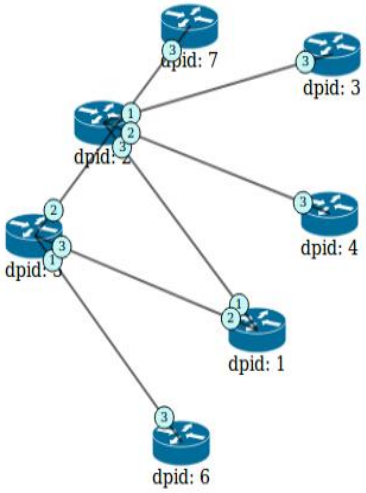


Figure 7. Combine Ryu controller and Mininet topology

Finally, on the lower level, Ryu accepts connections from specific ports based on the user's configuration of the Stream Server. These connections can be either unencrypted TCP or encrypted SSL, providing flexibility in securing the network communications.

There are two stages to configure the Ryu controller. Firstly, as shown in Figure 6, a series of commands are key in, the controller will handle the whole structure, joining switches generate and feedback Datapath. Secondly, linking Ryu controller and Mininet topology generate new topology, shown in Figure 7.

3.2 Experiment Design

Setting up a Mininet virtual network environment creates an infrastructure that supports SDN networking and OpenFlow. Installing a Ryu Controller allows the controller to use OpenFlow and communicates messages through the Datapaths generated. After establishing Mininet and the Ryu Controller and creating virtual switches based on topology structures, these switches are linked via Ryu's protocols and OpenFlow, enabling communication between the controller and the switches.

Within this setup, each switch and virtual host can act as a node. Using the full SDN architecture, the system sends and receives files from external sources, monitoring traffic flow, packet transmission, and speed. After the initial data transmission and receipt, temporary data of the same type is generated between these nodes. When these data are requested again, they can be collected from various nodes, thereby reducing the time traditionally required to re-request from the server and further increasing the speed of transmission and reception.

For comparison, traditional TCP/IP methods are used to transmit and receive files from the same source as Control Group One, and an undirected CCN mode establishes Control Group Two.

Finally, by analyzing and compiling these data, and comparing the differences in traffic and transmission speeds between the large data experimental group and control groups, it's possible to determine the efficiency improvements enabled by the SDN network architecture

3.2.1 Communicate with Ryu controller

We used the Ryu Controller to modify our assumed direct transmission mode, making configuration changes to its Address, Port, MAC, and the search mode, so that the Datapath can be more efficient and more accurate in determining when the Switch connects.

3.2.2 Whitelist in Controller

```
def __init__(self, mac, port):
    super(Host, self).__init__()
    self.port = 23
    self.mac = WhiteList[mac]
    self.ipv4 = WhiteList[ipv4]
    self.ipv6 = WhiteList[ipv6]

def to_dict(self):
    d = {'mac': self.mac,
         'ipv4': self.ipv4,
         'ipv6': self.ipv6,
         'port': self.port.to_dict()}
    return d

def __eq__(self, host):
    return self.mac == host.mac and self.port == host.port

def __str__(self):
    msg = 'Host<mac=%s, port=%s,' % (self.mac, str(self.port))
    msg += ','.join(self.ipv4)
    msg += ','.join(self.ipv6)
    msg += '>'
    return msg
```

Figure 8. Setting the addresses of User and Switch

Figure 8 shows the Host parameters selected in the code. Originally, it was to search for the location of the access port. We changed the code to execute by direct positioning and established a list controlled by the controller for management.

Figure 9 shows user information, which include MAC, IPv4, IPv6 address, and so on. Adding these information into Whitelist leads faster connect users to network.

```
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:2a:0b:8f
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::21ea:bda4:3993:5d25/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6691 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1547 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7648105 (7.6 MB)  TX bytes:108738 (108.7 KB)
```

Figure 9. confirm user address

3.2.3 Experimental Network Topology

After completing the setup of the environment, the Figure 10 shows the connection between the Controller and the Switch. By structuring virtual hosts in a virtual manner, specifically hosts with IP addresses ranging from 10.0.0.1 to 10.0.0.4, and connecting these virtual hosts to the Switch, communication can be established through the Datapaths generated. This setup interconnects the entire environment. Furthermore, these Switches and virtual hosts can all function as Nodes within the network. The SDN controller and FTP server are connected via Switch s2 and s3. This configuration facilitates a comprehensive network structure where data flow can be efficiently managed and directed, showcasing the flexibility and scalability of an SDN environment.

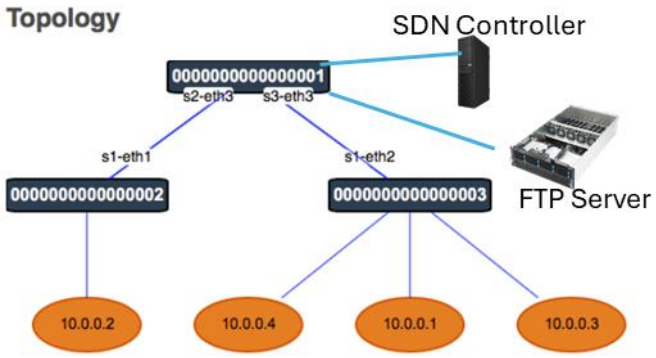


Figure 10. Experiment network topology

3.2.4 Networking Model and Experiment Stages

The experiment engages mass files transfer by FTP between users under 4G, Wi-Fi, and Ethernet networking models, respectively. Three transmission strategies are used: legacy TCP/IP, LCE CCN, and the proposed directly transmission path (DT) mechanism on CCN. Transmission speed and cache hit ratio are used for comparing the performance of these three transmission strategies. Using the open source SpeedNet FTP to test the transmission strategies, the transmission speed in MB per time are recorded. For get more precision values, the connections between users do not go through public networks. The study builds caching system by Memcache, which can set caching period and caching sources. Then we can get the cache hit ratio for comparing.

During the transmission process, we employed the widely used or default Least Recently Used (LRU) caching algorithm and derived three types of results from fixed file transmissions over 4G and Ethernet.

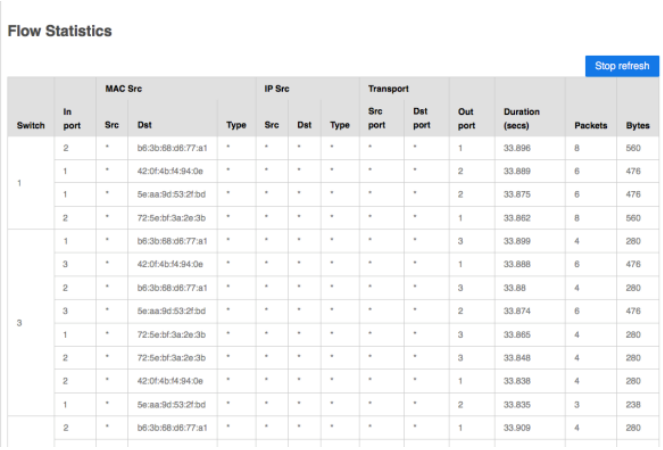
LRU, a default caching strategy in many Information-Centric Networking (ICN) contexts, involves caching data at all nodes along the path as data returns. This approach can lead to the wasteful use of caching resources, reducing the diversity of content that the caching system can hold due to the duplication of the same large files at multiple nodes.

The experiment shows that in the control group not using the CCN architecture, the transmission speed remains stable and the consumption of network resources is consistent.

For the group using the CCN architecture without a directed mode, the LCE algorithm caches at all nodes, so with an increase in the number of transmissions, the network speed gradually improves over time but ultimately stabilizes due to environmental limitations.

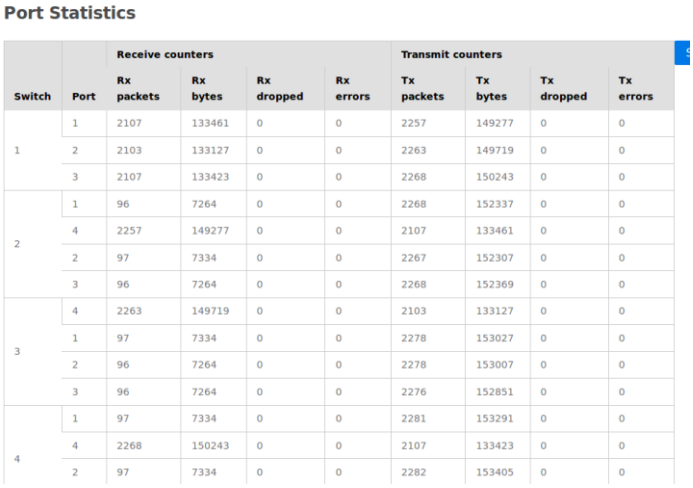
In the experimental group using the CCN with a directed mode, the system can quickly find the optimal path between users and servers from the outset due to

precise positioning. This method does not require caching at every node along the path, thus improving the cache hit rate for faster and more efficient transmission, reducing the consumption of network resources.



Switch	In port	MAC Src	Dst	Type	IP Src	Dst	Type	Transport Src port	Dst port	Out port	Duration (secs)	Packets	Bytes
1	2	*	b6:3b:68:d6:77:a1	*	*	*	*	*	*	1	33.896	8	560
	1	*	42:0f:4b:f4:94:0e	*	*	*	*	*	*	2	33.889	6	476
	1	*	5e:aa:9d:53:2f:bd	*	*	*	*	*	*	2	33.875	6	476
	2	*	72:5e:bf:3a:2e:3b	*	*	*	*	*	*	1	33.862	8	560
3	1	*	b6:3b:68:d6:77:a1	*	*	*	*	*	*	3	33.899	4	280
	3	*	42:0f:4b:f4:94:0e	*	*	*	*	*	*	1	33.888	6	476
	2	*	b6:3b:68:d6:77:a1	*	*	*	*	*	*	3	33.88	4	280
	3	*	5e:aa:9d:53:2f:bd	*	*	*	*	*	*	2	33.874	6	476
	1	*	72:5e:bf:3a:2e:3b	*	*	*	*	*	*	3	33.865	4	280
	2	*	72:5e:bf:3a:2e:3b	*	*	*	*	*	*	3	33.848	4	280
	2	*	42:0f:4b:f4:94:0e	*	*	*	*	*	*	1	33.838	4	280
	1	*	5e:aa:9d:53:2f:bd	*	*	*	*	*	*	2	33.835	3	238
	2	*	b6:3b:68:d6:77:a1	*	*	*	*	*	*	1	33.909	4	280

Figure 11. Flow statistics between Switch and Host



Switch	Port	Receive counters Rx packets	Rx bytes	Rx dropped	Rx errors	Transmit counters Tx packets	Tx bytes	Tx dropped	Tx errors
1	1	2107	133461	0	0	2257	149277	0	0
	2	2103	133127	0	0	2263	149719	0	0
	3	2107	133423	0	0	2268	150243	0	0
2	1	96	7264	0	0	2268	152337	0	0
	4	2257	149277	0	0	2107	133461	0	0
	2	97	7334	0	0	2267	152307	0	0
	3	96	7264	0	0	2268	152369	0	0
3	4	2263	149719	0	0	2103	133127	0	0
	1	97	7334	0	0	2278	153027	0	0
	2	96	7264	0	0	2278	153007	0	0
	3	96	7264	0	0	2276	152851	0	0
4	1	97	7334	0	0	2281	153291	0	0
	4	2268	150243	0	0	2107	133423	0	0
	2	97	7334	0	0	2282	153405	0	0

Figure 12. port statistics: throughput of TX/RX

As illustrated in Figure 4, a directed transmission method is constructed. Initially, users can make requests to the centrally controlled network controller through connected nodes or switches. The controller then forwards the user's access or request to the server. Once direct transmission communication is established between the two parties, the controller filters the necessary caching resources for transmission, avoiding unnecessary waste of caching resources.

4. EXPERIMENTAL RESULTS AND ANALYSIS

This section collects the experimental data for the performance analysis between transmission strategies on CCN.

4.1 Traffic Transmission Confirm

After establishing the network environment, traffic is initiated using Ping to confirm whether there is a connection between the Switch and the virtual host. As shown in Figure 11, we can confirm that the connection between the two is established. Then, generating traffic messages is used to check the state of sending and receiving. The Tx and Rx throughput can be seen, as shown in Figures 12. This can confirm user can communicate with the server. It is interchangeable.

4.2 Transmission Performance Analysis

In the experiment, we used control groups and an experimental group to analyze the data shown in Figure 13, 14, 15, and 16. The blue line represents Control Group One, which follows a traditional network transmission architecture, TCP/IP. The orange line represents Control Group Two, which does LCE on the CCN. The grey line represents the Experimental Group, which employs the CCN direct transmission strategy (DT).

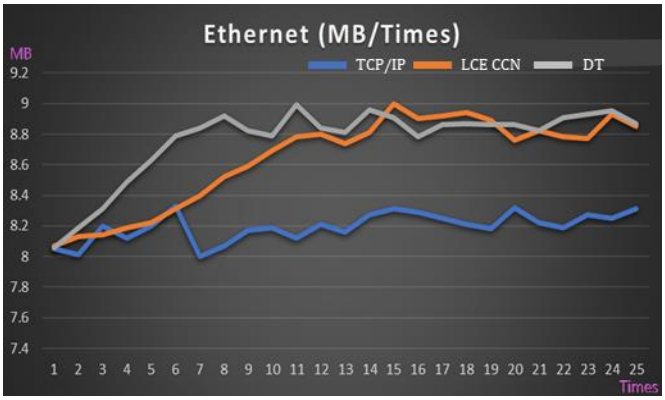


Figure 13. transmission throughput under Ethernet

Within the same network environment, we processed requests to the same file source (Speed.net FTP). Figure 13 shows the transmission throughput between the three transmission strategies under Ethernet networking environment: TCP/IP without CCN, LCE CCN and direct transmission path on CCN. As you can see, the throughput are very closed to each other. However, as the transfer times increased, the throughput are different according to the cache hit ratio. The TCP/IP without CCN strategy keep speed around a lower 8.2 MB. Between LCE CCN an DT, although the final transmission speeds were similar, the addressing method in DT improved cache hit rates and reduced node replacement rates. This allowed for quicker identification of the optimal transmission path, thereby reducing the utilization of cache resources to enhance network performance.

The LCE CCN has a worse performance since the leave copy everywhere mechanism in the beginning stages. However, the speed will increase fast than TCP/IP just

because the leave copy everywhere mechanism. On the other hand, the proposed DT has good performance since it firstly compute a shortest path and cache on the direct path. These methods leads a faster speed on the initial stages. The speed increased bounded around 8.9MB due to network speed limitations. The proposed DT approach has 5% improvement from LCE CCN in the begging stages but almost the same after a larger times of requests for transferring mass FTP files.

Figure 14 demonstrates the transmission throughput among the three transmission strategies under the 4G/LTE networking environment. The trends are almost the same as Figure 13, except the numbers. The proposed DT also has good performance in the beginning stages since it just cache content on the nodes on the shortest path. The proposed DT approach has 5% improvement from LCE CCN in the begging stages but almost the same after a larger times of requests for transferring mass FTP files.

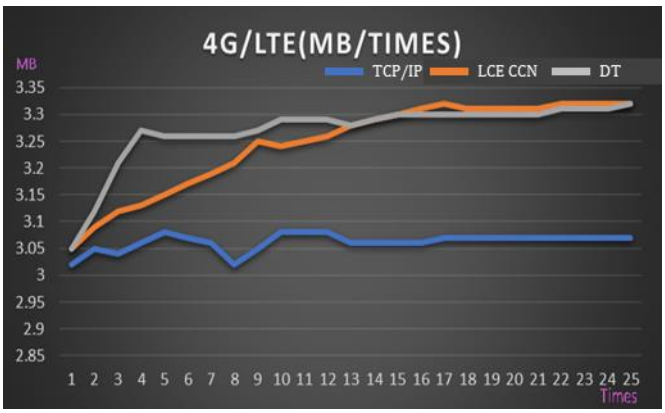


Figure 14. transmission throughput under 4G/LTE

Figure15 shows the same experiment as Figure 13 and Figure 14 except that the under networking environment is WiFi. Due to unstable wireless signal on a noise office environment, as you can see, DT outperforms other schemes in the beginning stages but has a smaller gate between them.

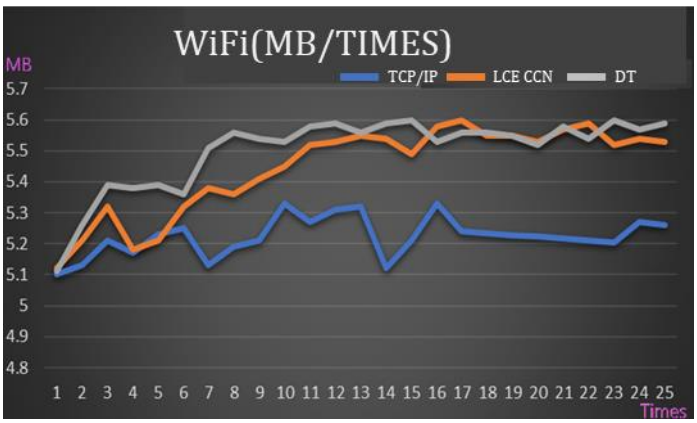


Figure 15. Transmission throughput under WiFi

Regarding cache hit rates, the non-CCN method(TCP/IP) showed lower efficiency compared to architectures that utilized CCN. By comparing the CCN architecture with and without addressing, DT and LCE CCN, respectively, it's evident that addressing significantly improves the cache hit rate initially. This prevents the excessive buildup of cache and access to nodes with no relevant data, which would otherwise increase the miss rate. The DT approach has 3%-5% improvement from LCE CCN.

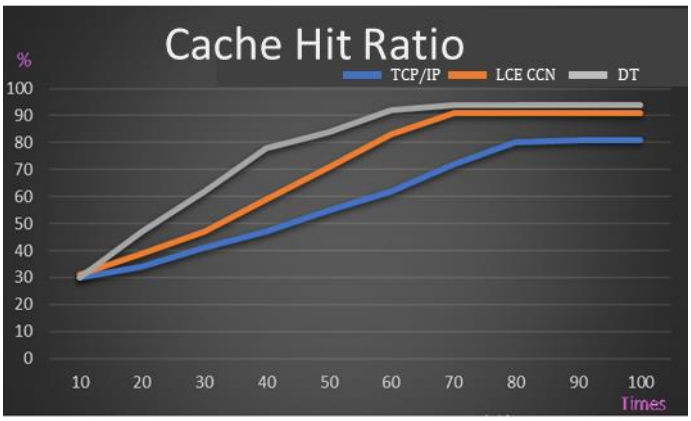


Figure 16. Cache hit ratio comparison

The proposed DT approach improved transmission speed, efficiency, and cache hit rate, while also reducing the node replacement rate for missed nodes.

4.3 Summary

In these experiments, it was discovered that various environmental factors could potentially influence the outcomes, necessitating methods to eliminate such variables to maintain the integrity of the experiments. Factors such as public networks, wireless signal strength, and weather conditions were considered. To ensure data integrity and relevance, the experiments utilized private networks and long-duration transmissions, with simulations conducted at the same time, location, and network segment.

Wireless networks were particularly challenging due to uneven signal strengths, obstructions by buildings, and the presence of various signal sources in the environment, leading to fluctuating transmission rates and consequently, data variability.

A large volume of data was needed to calculate an overall average, as external factors often impact preliminary results. For instance, high network speeds with very small file sizes could easily skew accuracy.

Comparing these experiments with control groups, a clear difference was evident. The use of a directed CCN architecture demonstrated that transmitting an unknown file between users and servers could facilitate faster

communication, thereby reducing network resource consumption and efficiently increasing the cache hit rate. This approach also minimized the burden typically caused by caching along the route, showcasing the benefits of optimizing network protocols and structures to improve overall network performance and efficiency.

5. CONCLUSIONS

In anticipation of future internet usage trends, driven by the explosive growth of cloud computing, mobile networking, and the Internet of Things (IoT), Software-Defined Networking (SDN) has emerged as a pivotal technology. Within this domain, various solutions are continually proposed, among which Content-Centric Networking (CCN) has gained notable recognition. Despite the innovative approaches within CCN, including early strategies like Least Recently Used (LCE) caching to later methods like Maximal Path Caching (MPC), there remains room for improvement. Thus, efforts have shifted towards exploring more balanced and rapid strategies on a smaller scale, leading to the experimental hypothesis of a direct transmission strategy.

This research highlights the challenges of previous node-to-node transmission methods, which, while aimed at increasing cache hit rates, inadvertently raised node replacement rates, producing counterproductive outcomes. The proposed direct transmission mechanism adds whitelist on SDN controller and set the cache remanding time by Memcache. The DT mechanism introduced an adjustment by increasing the list of users and servers within the domain's virtual machines, prioritizing domain communications. This approach leveraged the benefits of direct transmission strategies, enabling algorithms to calculate optimal paths that lower node replacement rates and enhance cache hit rates, thereby improving network utilization and transmission speed.

Through exhausted experiments, the research setup SDN networks on different networking environments: TCP/IP without CCN, LCE CCN and the proposed direct path transmission (DT) on CCN. Trough mass FTP files transferring between users, collecting transmission throughput and cache hit ratios for comparison. The experiment results showed that the proposed direct transmission scheme has 5% improvement on transmission throughput and 3%-5% improvement on cache hit ratio.

Challenges identified during the experiments include communication barriers between different domains, potentially due to network protocol incompatibilities or authentication issues. The experiments, conducted in virtual machines and environments, demonstrate the feasibility of executing these strategies within a complete network architecture. However, application is currently limited to smaller local area networks (LANs). For broader domains, future developments could explore more comprehensive algorithms capable of automatically selecting the best paths and adaptable to various communication protocols. The current experimental setup does not integrate different domains and protocols.

Furthermore, this foundation could inspire the development of new algorithms that accommodate temporary users with dynamic MAC and IP addresses, automatically assessing their security and controllability. Such advancements could enable a transition to direct transmission strategy modes, facilitating faster, simpler, and more efficient communication between servers and users.

6. REFERENCES

[1] Marc Mendonca, Bruno Astuto A. Nunes, Xuan-Nam Nguyen, Katia Obraczka, and Thierry Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," in submission 2013.

[2] Sakir Sezer, Barbara Fraser, David Lake, Jim Finnegan, Niel Viljoen, Marc Miller, and Navneet Rao, " Are We Ready for SDN? Implementation Challenges for Software-Defined Networks," on IEEE Communications Magazine, issue 7, vol.51, p.36-43, Jul. 2013.

[3] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner, "OpenFlow: Enabling Innovation in Campus Networks," on ACM SIGCOMM Computer Communication Review, vol.38, issue 2, pp.69-74, Apr. 2008.

[4] Boris Koldehofe, Frank Durr, Muhammad Adnan Tariq, and Kurt Rothermel, "The power of software-defined networking: line rate content-based routing using OpenFlow," on Proceedings of the 7th Workshop on Middleware for Next Generation Internet Computing, Article No. 3, 2012.

[5] Advait Dixit, Fang Hao, Sarit Mukherjee, T.V. Lakshman, and Ramana Kompella, “ElastiCon: An elastic distributed SDN controller,” ACM/IEEE Symposium on Architectures for Networking and Communications Systems(ANCS 14), pp.17-28, October 2014.

[6] Elian Aubry, Thomas Silverston, and Isabelle Chrisment, “SRSC: SDNbased routing scheme for CCN,” IEEE Conference on Network Softwarization (NetSoft 15), April 2015.

[7] Bruno Nunes Astuto, Marc Mendonça, Xuan Nam Nguyen, Katia Obraczka, and Thierry Turletti. A

Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. Research Report hal00825087, INRIA - UCSC, October 2013.

[8] X.N. Nguyen. Software defined networking in wireless mesh network. Msc. thesis, INRIA, UNSA, August 2012.

[9] Xuan-Nam Nguyen, Damien Saucez, and Thierry Turletti. Efficient caching in Content-Centric Networks using OpenFlow. In 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pages 67–68. IEEE, April 2013.

[10] L. Cui, F. R. Yu, and Q. Yan, "When big data meets software-defined networking (SDN): SDN for big data and big data for SDN," IEEE Network, vol. 30, no. 1, pp. 58–65, Jan. 2016.

[11] A. Shah S, J. Faiz, A. Farooq, et al. "An architectural evaluation of SDN controllers," IEEE. 2013 IEEE International Conference on IEEE. Budapest: ICC, 013: 3504-3508.

[12] A. Cahn, J. Hoyos, M. Hulse, et al. "Software-defined energy communication networks: from substation automation to future smart grids," IEEE. 2013 IEEE International Conference on IEEE. Vancouver: SmartGridComm, 2013: 558-563.

[13] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding metamorphosis: fast programmable match-action processing in hardware for SDN," in ACM SIGCOMM, 2013.

[14] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," White paper, Open Networking Foundation, Palo Alto, CA, USA, April 2012.

[15] D. Bernardini, T. Silverston and O. Festor, "Cache management strategy for CCN based on content popularity,". Berlin: Springer Berlin Heidelberg, 2013.

[16] I. Psaras, W.K. Chai , G. Pavlou. "Probabilistic in-network caching for information-centric networks, Edition of the ICN Workshop on Information-Centric Networking," August 13-17, 2012, Helsinki, Finland, New York: ACM Press, 2012:55-60.

[17] W.K. Chai, D. He, I. Psaras, G. Pavlou, et al. "Cache "less for more" in information-centric networks," International Conference on Research in Networking , NETWORKING 2012 pp 27-40

[18] Eberhart, R.C. and Shi, Y., "Particle Swarm Optimization: Developments, Applications and Resources," Proc. IEEE Int. Conf. On Evolutionary Computation, Vol. 1, pp. 81-86, 2001.

[19] M. Srivatsa, A. Iyengar, and Ling Liu. Privacy in voip networks: A k-anonymity approach. In INFOCOM 2009, IEEE, pages 2856– 2860, April 2009

[20] I. Abdullahi, S. Arif, and S. Hassan, "Survey on caching approaches in information centric networking," J. Netw. Comput. Appl., vol. 56, no. C, pp. 48–59, Oct. 2015

[21] J. Kennedy, R. Eberhart Particle swarm optimization. Neural Networks, 1995. Proceedings., IEEE International Conference on (IEEE). 1995, 4: 1942–1948

[22] Cisco Annual Internet Report (2018–2023) White Paper, March, 2024

[23] Orenstein, David (March 14, 2007). "A broad-based team of Stanford researchers aims to overhaul the Internet". Stanford News. Retrieved 2024-3-31.

[24] Van Jacobson, "A New Way to look at Networking" https://www.youtube.com/watch?feature=player_embedded&v=8Z685OF-PS8, Retrieved 2024-3-31

[25] C. Liang, F. Yu, X. Zhang, "Information-Centric Network Function Virtualization over 5G Mobile Wireless Networks." IEEE Network, vol. 29, no. 3, pp. 68-74, 2015

[26] Geoff Huston, "Web Caching", The Internet Protocol Journal, vol. 2, no. 3. Retrieved 2024-3-31.

[27] J. Elizabeth O'Neil, E. Patrick O'Neil, and Weikum Gerhard, "The LRU-K Page Replacement Algorithm for Database Disk Buffering," Proceedings of the 1993 ACM SIGMOD, ACM SIGMOD Record, Volume 22, Issue 2, pp 297–306, https://doi.org/10.1145/170036.170081, Retrieved 2024-3-31

[28] Ryu, https://ryu-sdn.org/, Retrieved 2024-3-31