Proceedings of the Asia-Pacific Advanced Network 2010 v. 30, p.37-44.
http://dx.doi.org/10.7125/APAN.30.6 ISSN 2227-3026

Improvement of Content Server with Contents Anycasting Using OpenFlow

Othman Othman M.M.[1] , Koji Okamura[2]

1 Department of Advanced Information Technology,
Graduate school of Information Science and Electrical Engineering,
Kyushu University.
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan.
+81-92-802-2722

2 Research Institute for Information Technology, Kyushu University.
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan.
+81-92-642-4030

E-Mails: omo_5@ale.csce.kyushu-u.ac.jp, oka@ec.kyushu-u.ac.jp

ABSTRACT

In this paper, we show our design and implementation method of contents anycasting using OpenFlow that is a promising future internet enabling technology. This method aims to improve the content server by enabling it serve more clients, also this method aims to make a more efficient use of the overall bandwidth available in the network by providing more flexible and dynamic redirection of contents that is done with the help of OpenFlow's new potential abilities and thus giving new opportunities to the future internet that are currently not available. This method relies on three important ideas which are; the content based networking, decision making by the network in a similar manner to anycast and the participation of user clients in providing the service. All this is done using the newly invented OpenFlow. It is done through some modifications to the content server and client, thus imposing small changes to the application layer.

Keywords

OpenFlow, Content Delivery Networks, Content based networking, Anycast.

1 INTRODUCTION

At the beginning of the Internet, it served the role of delivering contents using the simple client server model. But as time passes number of users grew fast along with that also the user's needs became broader and more diverse, causing an increase of server loads thus giving place to new content delivery models to take place like the Peer to Peer. And again as time passes with the introduction of new technologies, services and with social changes that followed the wide adoption of the Internet user's needs became more and more demanding. As shown by studies in AKARI [4] that the traffic size increases by factor of 1.7 every year. Moreover the needs became more complicated, by having wide variety of contents available on the Internet ranging from simple web pages and emails to online video and large file transfer. This lid to an increase in server loads, causing many difficulties for the current delivery models to cope with. Many solutions were proposed to solve the server overloading problem like increasing the bandwidth link, having redundant servers with load balancers or using Content Delivery Network concept which was introduced to solve the server overload problem and to provide high availability of the contents. And to enable the CDN many technologies were introduced like the Anycast [7] and Peer to Peer networks as in [5]. Anycast acts on having multiple replica servers that are carefully placed in the network acting as one logical server. It does so by efficiently directing client's requests to the nearest replica server to them. This redirection is done based on the location of both the client and the server. But anycast acts on fixed servers making it a static way of providing contents. In case of the peer to peer networks, it makes use of part of user's resources voluntary provided to contribute in providing contents to other users. But peer to peer networks impose extra communications forced by their protocols, the overlay nature of those networks and their lack of knowledge about the network topology.

All of those solutions except for peer to peer have to be implemented on the server side leaving it more complicated. While on the other hand, the clients side, clients are having stable internet connections with considerably high bandwidth specially with the introduction of the Fiber To The Building FTTB and the Fiber To The House FTTH, as an example in Japan the number of FTTH users is more than the number of DSL users as shown in

[3] (see Figure 1.). The thing that created a kind of imbalance between the server side and the client side.

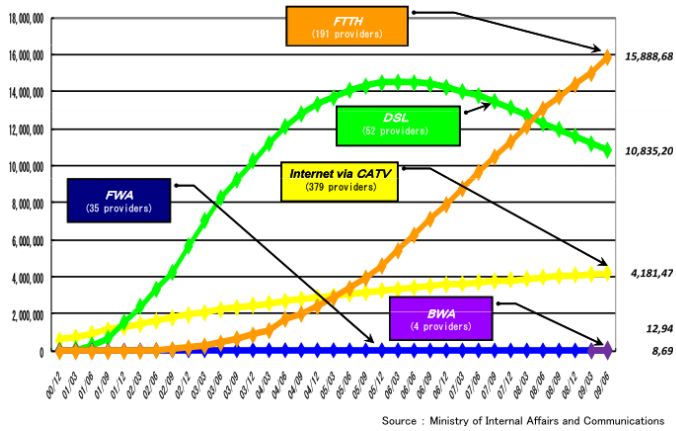

Figure 1. Number of services subscribers in Japan from 2000 to 2009

And if we take a closer look at the strength points of Anycast and Peer to Peer technologies we will find that Anycast strength lays behind that its redirection decision making is done by the network, while the strength point of the peer to peer networks lies in the contribution of users. Combining those two strength points along with idea of content based networking shown in [9, 10] looks very promising, by having the decision making done by the network based on both regular addressing and content addressing and having users contributing in providing service. Specially, that nowadays many users have a stable internet connection with great capacities.

Also in the future internet research where researchers from all over the world are studying challenges of the current Internet and are proposing ways to study and solve those challenges along with proposing new functionalities that were not possible on the current internet. One of the current efforts mainly aiming to enable researchers to run their experiments on the actual production networks which is OpenFlow [6].

OpenFlow is a part of Stanford University's clean slate project. It enables more freedom and flexibility in decision making by splitting the decision making or routing from packet forwarding. According to OpenFlow decision making can be done and modified freely by the OpenFlow controller according to layer 2, 3 and VLAN headers while the forwarding or routing is still done by routers or switches in addition to their original functionality. This freedom and flexibility enables it to play an important role in developing the future internet along with its main target which is running experiments.

This paper shows a design of a content anycasting mechanism aiming to tip off the imbalance between the server side and the client side and to improve the content server side by increasing the number of clients that the server can serve. This is done with the help of OpenFlow project that holds a great potential for developing systems that have more freedom in acting.

2 THE REDIRECTION SYSTEM

2.1 Motivation

Current internet relies on the server/client model and the Peer to Peer model to provide service. According to client/server model we have the server which provides the service and the client which is the user of the server who is getting the service. With the increase of the number of clients the overloading problem rouses, forcing content server to find solutions like increasing the bandwidth link, having redundant servers with load balancers, using CDN s or using peer to peer solutions. All those solutions except for the p2p are carried out by the server side leaving it more complicated. While on the other hand, the client side, the bandwidth is considerably increasing, especially with the introduction of fiber to the home (FTTH) and fiber to the building (FTTB). This causes a kind of imbalance by having more loads on content servers and increasing bandwidth for user clients, making an increase of the overall bandwidth used by the network. Moreover the anycast CDNs lacks the flexibility and the dynamicity of changing upon need. While on the other hand, the p2p networks have many limitations caused by their overlay nature the thing that imposes more communication in order to find the desired contents, locating other peers that already have that content and getting the content. So we designed our system to make a more efficient use of the overall bandwidth, in a way that is more dynamic than the anycast, with less overhead than peer to peer networks and requires some changes to the network and to the content server and client.

2.2 OpenFlow overview:

In order to elaborate our contents anycasting design, it is important to shed more light on OpenFlow which is part of Stanford University's clean slate project originally aims to enable researchers to run their experiments on their actual production network giving them better chance of getting more realistic results than that of the simulation.

According to OpenFlow, modern switches or routers contains flow tables to implement firewalls, NAT, QoS and to collect statistics. While each vendor implements the flow table differently OpenFlow identifies and exploits a common set of functions to enable its flow table and also provides the OpenFlow protocol to enable programming the flow table. Any OpenFlow system as shown in [6] (see Figure 2.) consists of the controller where the decision making or routing takes place and the OpenFlow router or switch where the packet forwarding takes place.

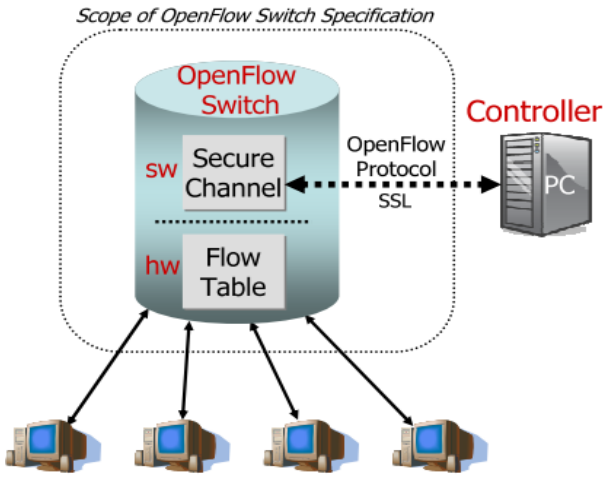

Figure 2. Idealized OpenFlow Switch. The flow table is controlled by a remote controller via the secure channel. As shown in [6]

This provides more freedom and flexibility in the decision making without overloading the routers or switches with this task. This freedom is gained by having a special controller that can be designed by developers or researchers according to their own needs and running their own code to implement or test new ideas without having to modify or change the existing networking infrastructure.

And in more details any switch or router that supports OpenFlow must have all of the following OpenFlow components; a flow table, a secure channel to connect the switch or router to the OpenFlow controller and an OpenFlow protocol which is used as a protocol of communication between the switch or router and the controller over the secure channel. First of all the flow table, which consists of flow entries (see Figure 3),

Header Fields	Counters	Actions

Figure 3. Flow table entry.

Where each flow entry consists of header fields to which the header of the incoming packet is matched against (see Figure 4),

Ingress Port	Ether source	Ether dst	Ether type	VLAN id	VLAN priority	IP src	IP dst	IP protocol	IP ToS	TCP/UDP src port	TCP/UDP dst port

Figure 4. Header fields.

counters to provide statistics about the flow entry and actions to be performed to the matched incoming packet. The actions can be either forwarding the packet to physical port or ports, enqueue the packet in a queue attached to a physical port, dropping the packet or modifying incoming packet's header fields which include modifying fields shown in Figure 4.

2.3 System overview

The proposed system consists of content server that provides contents, anycast manager, OpenFlow routers or switches and user clients (see Figure 5).

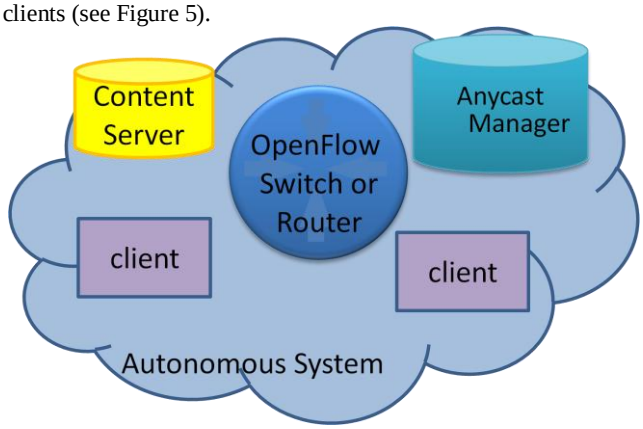

Figure 5. System overview.

2.3.1 Content Server

The content server is almost a regular content server that provides contents like large files or videos. Usually servers keep track of current user clients that are getting contents, and the content server must be modified to keep track of its current user clients upload capabilities. The use of the upload capabilities shows in finding out the number of new user clients that the current user client can serve. The content server must be modified so that it sends redirection request when some threshold is reached for example number of user clients or available bandwidth for the server. This redirection request includes the contents ids and the list of user clients that are currently getting this content and the number of redirections for each one of them (See Figure 6.). After that this redirection request is sent to the anycast manager in order to initiate the redirection. Also each content must have a unique identifier and the content server must manage the set of identifiers for the contents it serves.

Redirection request header		
Content id: 12345	192.168.10.1	1
	192.169.1.1	2
	192.168.10.4	3
⋮		
Content id: 33333	192.200.10.1	2
	192.200.1.1	1
	192.210.10.4	1

Figure 6. Redirection Request example.

2.3.2 Anycast Manager

The anycast manager is responsible for creating redirections and managing those redirections. The redirection system might require

more than one anycast manager for example one for each autonomous system, which means one anycast manager for each network of independent organization that implements BGP (Border Gateway Protocol) (see Figure 7.).

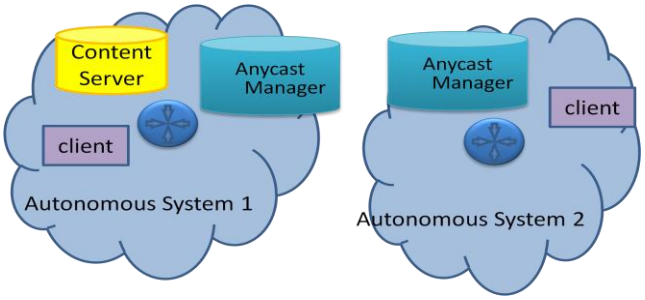

Figure 7. An example of a redirection system with two autonomous systems and tow anycast managers

The first function that the anycast manager provides is managing redirection requests which mean deciding which is the best anycast manager to handle the redirection request and forwarding the redirection request to it. The second function it provides is creating the redirection by sending instructions to the OpenFlow routers or switches to redirect the content requests that are originally sent to the content server to other user clients that are near to the requesting client. In more details for the second function, the anycast manager works as an OpenFlow controller. Through acting as an OpenFlow controller the anycast manager will be able to add the redirections to the OpenFlow routers or switches, enabling them to redirect packets according to those redirections. And about redirections themselves, they are OpenFlow's flow-entries that are carried out by the OpenFlow routers or switches to which they were sent and stored in the flow tables.

It is also important to show in more details how the anycast manager carries out its tasks in the case of having multiple anycast managers, where each anycast managers is located in a different autonomous system (see Figure 7.). First, we will explain the internal structure of the anycast manager. The anycast manager consists of the IP prefix locator, redirection analyzer and the redirection controller (see Figure 8.).

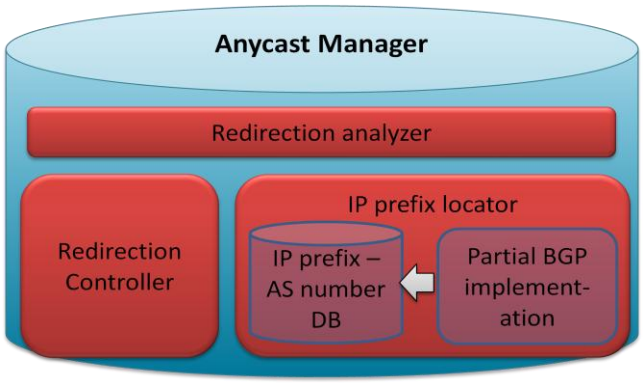

Figure 8. The internal structure of the anycast manager.

First, the IP prefix locator where its first part is the IP prefix – AS number data base that is used to map the autonomous system number which is the unique identifier to the autonomous system to the list of IP prefixes that are originated in that autonomous system. Where he IP prefix – AS number data base is a vital component that is used to find the appropriate anycast manager to handle the redirection request or part of it, where the appropriate anycast manager is the anycast manager that is located in the same autonomous system as the IP address of the client to where the content request must be redirected.

The second part of the IP prefix locator which is the partial BGP implementation that as the name declares is not a full implementation of the BGP. It implements vital BGP protocol messages that are required to establish and maintain a BGP session like the Open message and the Keepalive message. The main role of the partial BGP implementation is to be able to receive the Update messages, through receiving the update messages the IP prefix locator will be able to know in which autonomous system the IP prefix is located and this will be stored in the IP prefix – AS number data base. In more details if the partial BGP implementation receives an Update message it will first look into the Path Attributes field to find the Attribute Type and Value. If the Attribute Type is equal to the Origin then the Value is taken as the autonomous system number that originates the IP prefixes in the Network Layer Reachability Information field and this will be stored in the IP prefix – AS number data base. And if the type is equal to AS_Path then the last number found in the path that is found in the Value field which is the autonomous system number where the IP prefixes in the Network Layer Reachability Information field are located and then this is stored in the IP prefix – AS number data base.

Second, the redirection analyzer which is responsible of receiving the redirection request that is sent by the content server and analyzing it using the IP prefix – AS number data base (see Figure 9.).

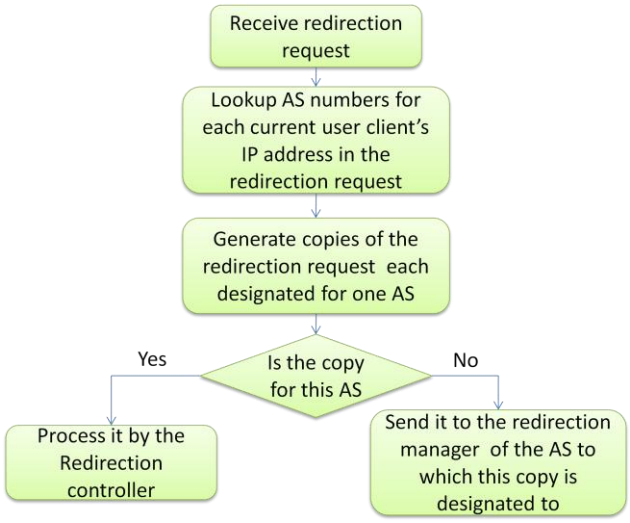

Figure 9. The steps of the redirection analysis done by the redirection analyzers.

This is done by looking up the IP prefixes of the current user clients that that are received in the redirection request to find their

AS numbers. Then the redirection analyzer will generate multiple copies of the of redirection request where each one of those copies is designated for a specific autonomous system by having only the IP addresses of the current user clients that belongs to the same autonomous system then. Then the redirection analyzer checks if any of those copies belongs to itself to be processed by the redirection controller and sends the rest of those copies each to the anycast manager that belongs to the autonomous to which this copy is designated to.

Finally, redirection controller which is the component that is responsible for creating the flow table entries and sending them to the OpenFlow switches or routers within the same autonomous system, actually it is an OpenFlow controller. And to explain more, this entry is designed so that it will match the incoming packets destination IP address to that of the server IP address , the protocol number to be that of our modified TCP protocol and both of the TCP/UDP source and destination port numbers will be used together to match the content id. (see figure 10.)

Header Fields											counters	Actions
Ingress Port	Ether source	Ether dst	Ether type	VLAN id	VLAN priority	IP src	IP dst	IP protocol	IP ToS	TCP/UDP src port / TCP/UDP dst port		
X	X	X	X	X	X	X	Content server IP	Modified TCP #	X	Cont-ent ID		Modify IP dst to be the current user client IP address.

Figure 10. An example of the flow table entry generated by the redirection controller.

2.3.3 OpenFlow routers or switches

The role of the OpenFlow routers or switches is to perform the redirections that where created by the anycast manager. As mentioned before that the redirections are OpenFlow's flow table entries, where each incoming packet is checked if its destination IP address and its content id matches the content server IP address and the content id in one of the flow table entries. And if a match occurs the OpenFlow router or switch uses one of the Openflow functionalities to change the destination IP address in the first packet of the content request from the IP address of the content server to the IP address of the nearest client. By doing so, the first packet of the content request will be delivered to the nearest client without any effort done by the user clients.

2.3.4 User Clients

User clients in our system have a modified behavior that they will act as servers for the contents they are currently getting from the content server. Also the method of getting the content is different than the normal method as described below.

2.4 Requesting content

In order to make use of our system a special method for getting the content is required. This new method is divided into two phases; the first one is related to finding the content id, while the second is related to sending the content request.

2.4.1 Phase 1

Before this phase begins the client browses the content server's web pages looking for the contents that he is interested in. and when the client finds the desired content and sends the request, and then the content server will send the content's id to the client so that it can be used in the second phase. As an example when using the HTTP protocol when the client sends a GET request the content server will respond with a special response that contains the content id.

2.4.2 Phase 2: Content Request

This phase takes place after the client gets the content id and wants to start getting the content. In this phase a 3 way handshake is used. And in order to enable this, a modified TCP protocol is required (see Figure 11 and Figure 12).

Bit offset	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0	Source Port	Destination Port
32	Sequence Number	
64	Acknowledgement Number	
96	Data Offset / Reserved / Flags	Window Size
128	Checksum	Urgent Pointer
160	Options	

Figure 11. The original TCP header.

Bit offset	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0	Content id	
32	Source Port	Destination Port
64	Sequence Number	
96	Acknowledgement Number	
128	Data Offset / Reserved / Flags	Window Size
160	Checksum	Urgent Pointer
192	IP address	
224	Upload capabilites	
256	Options	

Figure 12. The modified TCP header, where the highlighting shows the modified fields.

The required modifications are done by inserting a 32 bit field before the source port number which contains the content id. This content id is used in the process of the redirection that is done by the OpenFlow routers or switches. Also an IP address field is added so that the packet sender uses this field to tell the other part of the communication about its IP address. And the final modification is adding the upload capabilities field which the sender uses to inform the other part of the communication about its upload bandwidth. This is especially useful in the case of starting a download from the server because it lets the server know about the clients upload bandwidth so that it will use this

information to decide about incorporating clients to provide service or not.

According to the modified TCP, the client initiates this modified TCP connection by sending a TCP SYN packet that contains the content id in it. This packet will be redirected by the OpenFlow switches or routers to the nearest client that is currently getting the same content from the content server. This redirection is done if the destination IP address and the content id in the header of the received packet are matched against the content servers IP address in the destination IP address and the same content id in the content id in the header of one of the flow table entries of the switch or router. Then the switch or router changes the destination IP address to be that of the current client instead of the content servers IP address and send it out on one of its ports to be delivered to the nearest client that is currently getting the same content. And in response to this initiation the other client that received the packet will respond with SYN/ACK packet that contains its own IP address and port number to be used later. At this time the new client realizes that it has been redirected to another client and continues the rest of the TCP connection using the IP address of the other client directly without relying on the redirection system. Figure 13 shows the modified TCP connection initiation in case of a matching (see Figure 13).

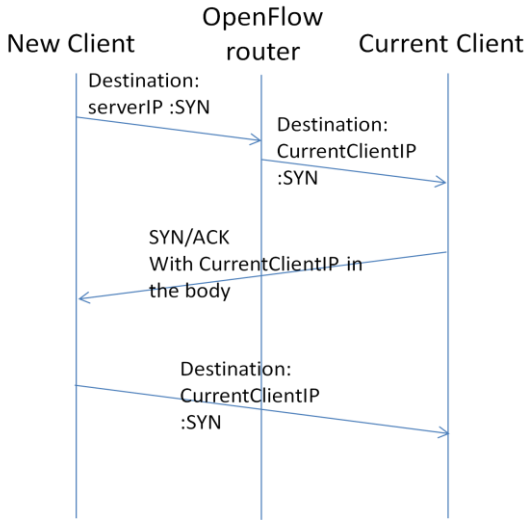

Figure 13. Content request in case of redirection.

Using this way the role of the redirection system is minimized to modifying one packet for each content request that has a matching redirection, otherwise the system will treat the content request in the traditional manner and send it directly to the content server (see Figure 14).

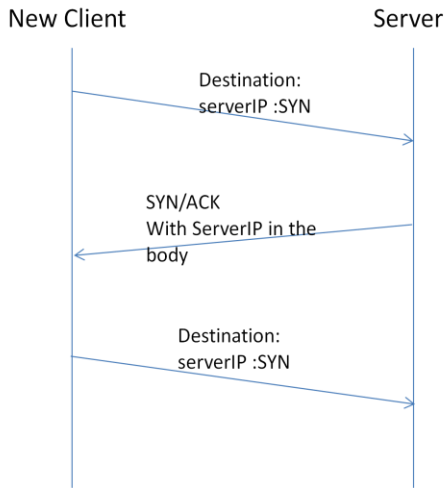

Figure 14. Content request in case of no redirection.

2.5 Usage Scenario

The following scenario helps to give a better understanding of whole system.

First of all, Figure 15 show the content server, anycast manager, client A which is currently downloading the desired content, client B which is also a new client that is intersected in downloading the same content and an OpenFlow router shown as the circle in the figure.

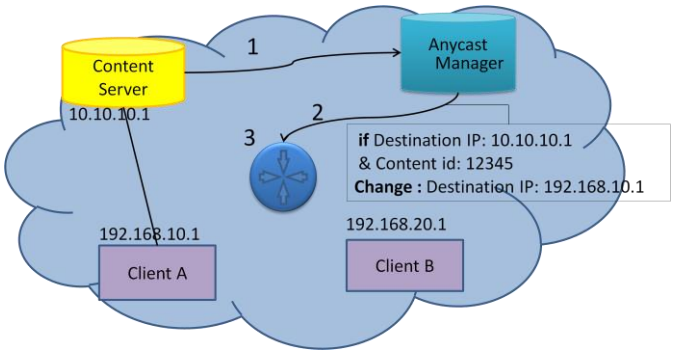

Figure 15. System initiation

In step number 1 the content server reaches some threshold limit, and so sends a redirection request to the anycast manager. In step number 2 the anycast manager analyzes the request and prepares the required redirections and sends them to the OpenFlow router. And in step 3 the OpenFlow router saves the redirection, which is an OpenFlow table entry to its flow table.

Next in Figure 16 shows phase 1 of the content request.

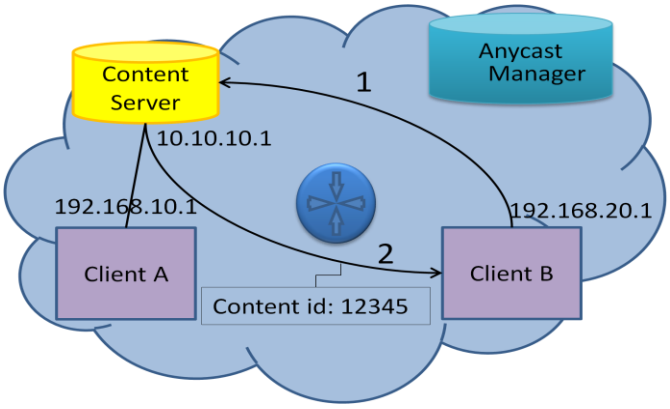

Figure 16. Content request, phase 1

In Figure 16 step 1, client B sends a request to get the content id, and receives the content id in step 2.
While in Figure 17 phase 2 of the content request is shown.

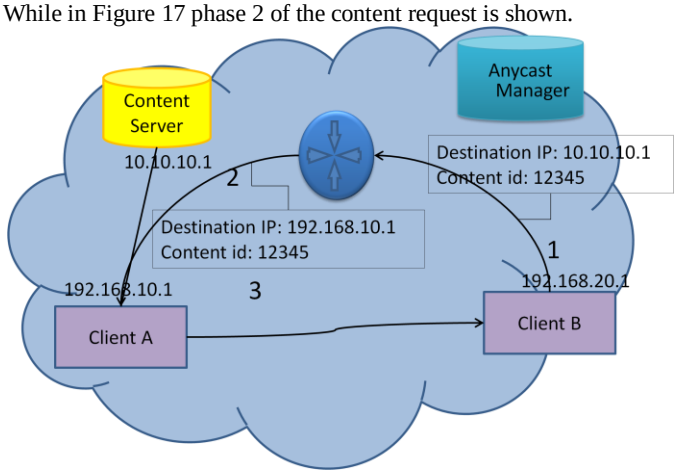

Figure 17. Content request, phase 2

In Figure 17 step 1 client B starts the modified TCP handshake that contains the content id. In step 2 the OpenFlow router redirects the packet sent by client B to go to client A instead of the content server after matching it to the redirection it received from the anycast manager by changing the destination IP to be that of client A and sending it on the right port to be delivered to client A. But in case of not finding a matching redirection the router will process the packet in the conventional way by sending the packet to the content server. Finally in step 3 the client B and client A completes the handshake and client A will start sending the content to client B directly without using the redirection on the router.

3 IMPLEMENTATION

Currently we are in the phase of building our testbed that includes five PCs with multiple Ethernet interfaces running Ubuntu 9.10 in which the reference OpenFlow switch version 1.0 is running. And a anycast manager to be implemented on a PC running Ubuntu 9.10, in which we are currently building the redirection controller which is a special OpenFlow controller, redirection analyzer and the IP prefix locator. The redirection controller is responsible for creating the redirections and sending them to the OpenFlow routers or switches, while the redirection analyzer is responsible for managing redirections and organizing them with other anycast managers. Also our testbed will include a modified content server that implements the two phases of requesting the content and multiple user clients that are also modified to meet the two phases and have the ability to serve other clients.

Firstly, we will run experiment in which we will build a CDN using our system and also use it to find the best tuning values of the redirections. Second we will make a second duplicate testbed, and using the two testbeds we will run experiment in which we will run a single CDN over the two parts of the testbed in order to experiment the cooperation between multiple anycast managers.

The first two experiments aims to find and measure performance related measurements like the communication cost of our contents anycasting system that means the bandwidth consumed by the extra phases in the content request, the redirection request sent by the server and management and sending the redirections to the routers or switches. Also the response time or our contents anycasting system will be measured, which means the time required since issuing the redirection request by the content server and using those redirection on the OpenFlow routers or switches. Moreover we will run experiments to test how efficient our system will response to different load patterns, especially to the flash crowd loads. And also we will measure the benefit that the content server will gain by using our system by measuring the increase in the number of user clients getting the content.

Also finally we will run experiments on a network simulator in order to get the same results but on a simulation environment with large network the thing that will help us to get a more realistic vision on how our system will react on the actual world network.

Finally and after getting all the previous results we will run a comparison in which we will compare our system with other solutions like the wide spreading BitTorrent Peer to Peer and also we will compare our system to the anycast.

4 CONSIDERATIONS AND FUTURE WORK

It is important to have a more in-depth understanding of many details related to our contents anycasting system like the best implementation coverage of Openflow among the whole network which means studying about the optimal number of OpenFlow routers or switches in the network along with studying the best places for those router or switches to be located within the network in order to get good redirection performance.

It is also important to study in more detail the way that the anycast manager uses to get information about the network topology.

Moreover studying the anycast managers load limits and the way our system affects the network. And finally putting a business model is also important towards the completing the study of our new system.

5 CONCLUSION

Providing new opportunities to the future internet is very important. But this requires the creation and adoption of new

technologies. In this paper we described our new design for content anycasting, which aims to improve the content server by increasing the number of clients it serves, also it aims to make a more efficient use of the overall bandwidth in the network and providing more flexibility and dynamicity in a way that is exceeds that of the current internet. Through this paper we showed our design that makes use of OpenFlow in order to perform the required redirection of packets. Along with the use of OpenFlow our method requires the use of an anycast manager to create and organize redirections. In our system the redirection relies on both the destination address and the content id to take the redirection decision. And in order to support the content id, we showed our content request that consists of two phases, one is related to getting the id of the content and the other is a modified 3 way handshake that incorporates the content id. This also requires modifications to both the content server and the client to adopt the new system. And in order to show the flexibility of this system we are currently implementing a testbed to run experiments such as running a content delivery network with multiple anycast managers.

6 REFERENCES

[1] BitTorrent Protocol Specification v1.0. http://wiki.theory.org/BitTorrentSpecification

[2] Cerf, V., Dalal, Y., Sunshine, C. SPECIFICATION OF INTERNET TRANSMISSION CONTROL PROGRAM. December 1974.

[3] FUJINO, M. National Broadband Policies: 1999--2009, Japan. October 2009. http://www.soumu.go.jp/main_sosiki/joho_tsusin/eng/presentation/pdf/091019_1.pdf .

[4] Hirabaru, M., Inoue, M., Harai, H., et. al. October 2008. New Generation Network Architecture AKARI Conceptual Design (ver1.1). AKARI Arechitecture Design Project. http://akari-project.nict.go.jp/eng/concept-design/AKARI_fulltext_e_translated_version_1_1.pdf .

[5] Karagiannis, T., Rodriguez, P., and Papagiannaki, K. 2005. Should internet service providers fear peer-assisted content distribution?. In Proceedings of the 5th ACM SIGCOMM Conference on internet Measurement (Berkeley, CA, October 19 - 21, 2005). Internet Measurement Conference. USENIX Association, Berkeley, CA, 6-6.

[6] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. 38, 2 (March 2008), 69-74. DOI=10.1145/1355734.1355746 http://doi.acm.org/10.1145/1355734.1355746 .

[7] Partridge, C., Mendez, T., and Milliken, W., RFC 1546 – Host Anycasting Service. November 1993.

[8] Rekhter, Y., Li, T. and Hares, S., RFC 4271 – A Border Gateway Protocol 4 (BGP-4). January 2006.

[9] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. 2007. A data-oriented (and beyond) network architecture. SIGCOMM Comput. Commun. Rev. 37, 4 (August 2007), 181-192. DOI=10.1145/1282427.1282402 http://doi.acm.org/10.1145/1282427.1282402

[10] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. 2009. Networking named content. In Proceedings of the 5th international conference on Emerging networking experiments and technologies (CoNEXT '09). ACM, New York, NY, USA, 1-12. DOI=10.1145/1658939.1658941 http://doi.acm.org/10.1145/1658939.1658941 .