DMM Working Group
Internet-Draft
Intended status: Informational
Expires: November 5, 2022

Jaehwoon Lee
Dongguk University

May 6, 2022

Network-based mobility management in Dyncast network environment
draft-jaehwoon-dmm-dyncast--mobility-00

Abstract

Dynamic anycast (Dyncast) network architecture is to choose the best edge computing server by considering both the network environment and available computing/storage resources of the edge computing server. This draft describes the mechanism in which service continuity is provided even when the client moves and connects to a new ingress Dyncast anycast Node (DAN) by using the PMIPv6-based mobility management method in the Dyncast-based edge computing networking environment.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

Cloud computing provides powerful computing and nearly unlimited storage resources to client devices connected over the Internet. However, if the number of client, such as Internet of Things (IoT) devices is quite large, traffic exchange between the client and the cloud computing server is also large and it can cause congestion over the Internet. When congestion occurs on the path between a client and the cloud computing server, the client transmitting service request may experience long response time in receiving the result of the service request, or the service request itself may be lost.

In edge computing, even though edge computing server provides smaller computing and storage resources compared to the cloud computing server, multiple number of edge computing servers can be located near client devices and the client sending service request can benefit from shorter response time. In the edge computing environment, one way for a client to find a suitable edge computing server is to choose the nearest edge server based on the location of the client inferred from the client's source IP address. Another way is to choose one of the several edge servers by utilizing the round-robin method. However, in such cases, if the available resource in the chosen server is insufficient or congestion occurs on the path between the client and the chosen server, the client may experience longer response time or service request may be lost.

Dynamic anycast (Dyncast) network architecture is proposed in choosing the best edge computing server by considering both the networking environment and available computing/storage resources of the edge computing server[1]. Here, a service is represented by an anycast IP address. Assume that there is a client trying to receive a service provided by a specific service instance. In this case ingress Dyncast anycast node (DAN) acts as a gateway for the client. In addition, egress DAN is connected to the edge computing server in which specific service instance is installed. Assume that there are

N edge servers providing a specific service. Each edge server is connected to a different egress DAN. The client transmits a service request message with anycast address as a destination IP address. Ingress DAN chooses the best egress DAN by using the combination of the network metric such as delay, and computing metric such as available computing/storage resource of edge servers. The ingress DAN establishes a tunnel with the chosen egress DAN and then transmits a service request through the tunnel. After which egress DAN transmits the service request to the service instance in the edge computing server. The result of the service request is in turn transmitted from the edge server to the client through the egress DAN and the ingress DAN.

When a client transmits a service request and then moves to another network before receiving the service result, the client can no longer receive the result of the service request. Even when the client moves and connects to a new ingress DAN, host-based mobility management method such as Mobile IPv6 (MIPv6) can be used to maintain end-to-end connectivity[2]. In this case however, the destination IP address of the service request message sent by the client is the anycast IP address. Which means that the new ingress DAN cannot know the egress DAN connected to the edge server providing service to the client which uses the anycast IP address as the destination IP address. Therefore, host-based mobility management cannot be used in the Dyncast networking environment. That being said, network-based mobility management mechanism such as Proxy MIPv6 (PMIPv6) can be used in the Dyncast networking environment if the new ingress DAN knows the address of the egress DAN connected to the edge server providing service to the client[3]. In this case, service continuity is ensured for the client.

This draft describes the mechanism in which service continuity is provided even when the client moves and connects to a new ingress DAN by using the PMIPv6-based mobility management method in the Dyncast-based edge computing networking environment.

2. Conventions and Terminology

2.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL","SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [4].

2.2 Terminology

TBD.

3. Protocol Operation

Fig. 1 show the message exchange procedure to provide service continuity proactively when a client moves to another network in Dyncast networking environment. If the client transmits service request message with anycast address as a destination IP address, an ingress DAN (that is, old ingress DAN) chooses the best egress DAN by using the combination of the network metric and computing metric. The old ingress DAN establishes the tunnel with the chosen DAN and then transmits the service request message through the tunnel. The egress DAN transmits the service request message to the corresponding service instance in the edge computing server.

When the old ingress DAN detects the movement of the client before completing transmission of all service results, it transmits the mobility notification message including the IP addresses of the client and the egress DAN to one or more candidate new ingress DANs that clients may connects to. The format of the mobility notification message is TBD. Here, how the old ingress DAN can know the movement of the client is out of scope. One method is to use the signal strength of the client. Moreover, how the old ingress DAN can know which is the new ingress DAN that the client moves and connects to is TBD. One method is for the old ingress DAN to broadcast the mobility notification message to neighbor ingress DANs. Another method is to find some candidate ingress DANs by using the GPS information of the client. A new ingress DAN having received mobility notification message establishes the tunnel with the old ingress DAN. Moreover, it establishes the tunnel with the egress DAN. When the client moves and connects to a new ingress DAN, the new ingress DAN transmits mobility indication message including the IP address of the client to the old ingress DAN and the egress DAN. The format of the mobility indication message is TBD. From now on, the old ingress DAN and the egress DAN transmit all services results to the client through the new ingress DAN.

Fig. 2 show the message exchange procedure to provide service continuity reactively to the client. If the client moves and connects to a new ingress DAN, the new ingress DAN transmit mobility request message including the IP address of the client to the old ingress DAN. The format of the mobility request message is TBD. Here, how the new ingress DAN can know the address information of the old ingress DAN is TBD. Moreover, how the new ingress DAN can know whether the connected client needs service continuity or not is TBD. The old ingress DAN transmits the mobility notification message and establishes the tunnel with the new ingress DAN. The new ingress DAN transmits the mobility indication message to the egress DAN and establishes the tunnel with the egress DAN. From now on, the old ingress DAN and egress DAN transmit all service results to the client through the new ingress DAN.

```
      Client   old ingress DAN   new ingress DAN   egress DAN   Service
                                                                instance
       |              |               |                 |               |
       |<--connect -->|               |                 |               |
       |              |<=====     est. tunnel      ====>|               |
       |-service req->|               |                 |               |
       |              |------ service request --------->|               |
       |              |               |                 |-service req ->|
   (movement)         |               |                 |               |
       |  (client move detection)     |                 |               |
       |              |- notify msg ->|                 |               |
       |<-----  connect    ---------->|                 |               |
       |              |<-- ind. msg --|                 |               |
       |              |<=est. tunnel=>|                 |               |
       |              |               |                 |<- svc result--|
       |              |<----     service result    -----|               |
       |              |- svc result ->|                 |               |
       |<---      svc result      ----|                 |               |
       |              |               |--- ind. msg --->|               |
       |              |               |<= est. tunnel =>|               |
       |              |               |                 |<- svc result--|
       |              |               |<-- svc result --|               |
       |<---      svc result      ----|                 |               |
```

Figure 1: Message exchange procecure - proactive method

```
      Client   old ingress DAN   new ingress DAN   egress DAN   Service
                                                                instance
       |              |               |                 |               |
       |<--connect -->|               |                 |               |
       |              |<=====     est. tunnel      ====>|               |
       |-service req->|               |                 |               |
       |              |------ service request --------->|               |
       |              |               |                 |-service req ->|
   (movement)         |               |                 |               |
       |<-----  connect    ---------->|                 |               |
       |              |<-- req. msg --|                 |               |
       |              |- notify msg ->|
       |              |<=est. tunnel=>|                 |               |
       |              |               |                 |<- svc result--|
       |              |<----     service result    -----|               |
       |              |- svc result ->|                 |               |
       |<---      svc result      ----|                 |               |
       |              |               |--- ind. msg --->|               |
       |              |               |<= est. tunnel =>|               |
       |              |               |                 |<- svc result--|
       |              |               |<-- svc result --|               |
       |<---      svc result      ----|                 |               |
```

Figure 2: Message exchange procecure - reactive method

4. Security Considerations

TBD

5. IANA Considerations

TBD

6. References

[1] Y. LI, L. Iannone, D. Trossen, P. Liu and C. Li, "Dynamic-Anycast Architecture", draft-li-dyncast-architucture-03 (work in progress, Mar. 7, 2022.

[2] D. Johnson, C. Perkins and J. Arkko, "Mobility Support in IPv6", IETF RFC 3775, June 2004.

[3] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury and B. Patil, "Proxy Mobile IPv6", IETF RFC 5213, Aug. 2008.

[4] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Author's Address

Jaehwoon Lee
Dongguk University
30, Pildong-ro 1 gil, Jung-gu
Seoul 04620, KOREA
Email: jaehwoon@dongguk.edu