Network Working Group Z. Du
Internet-Draft P. Liu
Intended status: Standards Track China Mobile
Expires: December 26, 2021 June 24, 2021

Path Information Detection in Application-aware IPv6 Networking
draft-du-apn6-path-infomation-detection-01

Abstract

This document introduces a method to detect path information in Application-aware IPv6 Networking.

Requirements Language

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 RFC 2119 [RFC2119].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Application-aware IPv6 Networking is a kind of self-identified mechanism per packet. In the mechanism of APN6 [I-D.li-apn6-problem-statement-usecases], an IPv6 packet can carry the APP ID information and SLA requirements of the traffic in its Extension Headers. Therefore, the network equipment can analyze them in each packet and handle the packet accordingly.

This novel mechanism can enable the negotiation between the user traffic and the network. The network can supply proper treatment for different kinds of user traffic. As a result of this flexible on-demand SLA mechanism, the user can get a better experience, and the network resource can be scheduled more efficiently.

However, the current mechanism in APN6 only enables a unidirectional information notification, i.e., from the APP/user to the network. The APP/user is not aware of the path information in the network. In some cases, it is not sufficient. A bidirectional information negotiation mechanism can enable a more powerful APN6 platform.

This document introduces the process of the path information detection mechanism in APN6 by extending some IPv6 Extension Headers.

2. Current Mechanism in APN6

As shown in Figure 1, the APN framework [I-D.li-apn-framework] includes App (Client and Server), App-aware Edge, App-aware-process Head-End, App-aware-process Mid-Point, and App-aware-process End-Point.

```
Client                                                         Server
+-----+                                                        +-----+
|App x|-\                                                   /->|App x|
+-----+ |   +-----+ +---------+   +---------+   +---------+ |  +-----+
         \->|App- | |App-aware|-A-|App-aware|-A-|App-aware|-/
User side   |aware|-|process  |-B-|process  |-B-|process  |
         /->|Edge | |Head-End |-C-|Mid-Point|-C-|End-Point|-\
+-----+ |   +-----+ +---------+   +---------+   +---------+ |  +-----+
|App y|-/                                                   \->|App y|
+-----+           ---------  Uplink   ---------->              +-----+
```

Figure 1: Framework and Key Components in APN6

The data-driven process of APN6 is described below.

The APP or the APP-aware Edge will generate an APN packet which carries the application characteristic information in the encapsulation. The information may include application-aware identification, such as SLA level, application ID, user ID, flow ID, etc., and network performance requirements, such as bandwidth, latency, jitter, packet loss ratio, etc. The former is recorded in the Application-aware ID Options, and the latter is recorded in the Service-Para Options defined in the [I-D.li-apn-framework].

App-aware-process Head-End can read that information and steer the packet into a given policy which satisfies the application requirements. It is supposed that a set of paths, tunnels or SR policies, exist between the App-aware-process Head-End and the App-aware-process End-Point. App-aware-process Head-End can find one existing path or establish a new one for the traffic.

3. Path Latency Information Detection

In the APN architecture, the user/APP can give a latency requirement to the network, and the network will provide a low latency path for the traffic. The path may be the one with the lowest latency among the set of paths between the Head-End and the End-Point, or may be one of the paths with a lower latency than the value carried in the latency requirement TLV. However, the users/APPs do not know how well the network handle the requirements, especially when the APPs give multiple requirements.

An APP can easily obtain a bidirectional E2E latency, i.e., the sum of the bidirectional network latency and the server latency, but it does not know the exact network latency. The mechanism proposed in this document can provide this information to the APP, and the APP can confirm the network's SLA guarantee activity if needed.

In details, the APP can add a new Service-Para Option named Timestamp Request TLV into the packet to indicate that it needs the timestamps of the headend and the endpoint in the network. The headend and the endpoint can read the TLV and add two new Timestamp TLVs in the IPv6 extension header, which contain the time that the packet reach the headend and the endpoint respectively. Then, the server can receive this specific packet with the Timestamp Request TLV and the two Timestamp TLVs. The server can record the timestamps, and encapsulate them into a packet that is about to be sent to the APP. This packet can also include a Timestamp Request TLV. On the converse direction, the headend and the endpoint in the network can add another two Timestamp TLVs into the packet. Hence, the APP can get four timestamps in total, and know the forwarding path latency and the reverse path latency of the network.

4. Other Path Information Detection

The APP can also request other information by extending more Request TLVs and Information TLVs in the IPv6 extension header. For example, the APP can request to obtain the SR policy BSID in the Path Information TLV. In this case, only the headend on the forwarding direction needs to add information into the packet. As the information needs to be handled by the server, the BSID can be stored in DOH (Destination Options Header) of the packet.

When the APP has obtained the BSID, it can add it into the SID list contained in the packet or into a new Service-Para Option. The headend can directly steer the traffic to a specific SR Policy according to that information. Therefore, it only needs to analyze the several packets in the traffic at the beginning, and does not need to analyze the following packets with BSID in the traffic in details.

Another example is about dynamic load balance. Nowadays, the load balance in the network, such as the ECMP or weighted-ECMP, is based on pre-configured weight. As an example, in an SR policy, different candidate paths may have different weights[I-D.ietf-spring-segment-routing-policy]. We assume two SID lists, List1 and List2, have weight 1 and weight 5 respectively. In this static load balance, more traffic will be steered to List2 because it has a large weight. However, even if the candidate path following List2 is congested, and the candidate path following List1 is light-loaded, no mechanism can enable new-coming traffic to use List1 more than before. In other words, we can not adjust the weight ratio from 1:5 to 1:3. It is because that if we change it, some traffic used to follow List2 will be moved to the path following List1, and misorder may take place.

With the help of the path information detection methods in this document, a flow can be aware of the candidate path it follows, and carry an ID of the candidate path in the IPv6 header. The load balance point in the network can use it to steer the flow traffic directly, bypassing the load balance process. In this situation, for the dynamic path-attached traffic, the load balance point can use a dynamic weight, which may be influenced by the traffic load in the candidate paths. When a new flow comes, it can be load-balanced by using the dynamic weight.

5. IANA Considerations

TBD.

6. Security Considerations

TBD.

7. Acknowledgements

TBD.

8. References

8.1. Normative References

[I-D.li-apn-framework]
Li, Z., Peng, S., Voyer, D., Li, C., Liu, P., Cao, C., Ebisawa, K., Previdi, S., and J. N. Guichard, "Application-aware Networking (APN) Framework", draft-li-apn-framework-02 (work in progress), February 2021.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

8.2. Informative References

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-11 (work in progress), April 2021.

[I-D.li-apn6-problem-statement-usecases]
Li, Z., Peng, S., Voyer, D., Xie, C., Liu, P., Liu, C., Ebisawa, K., Previdi, S., and J. N. Guichard, "Problem Statement and Use Cases of Application-aware IPv6 Networking (APN6)", draft-li-apn6-problem-statement-usecases-01 (work in progress), November 2019.

Authors' Addresses

Zongpeng Du
China Mobile
No.32 XuanWuMen West Street
Beijing 100053
China

Email: duzongpeng@foxmail.com

Peng Liu
China Mobile
No.32 XuanWuMen West Street
Beijing 100053
China

Email: liupengyjy@chinamobile.com