Workgroup: Network Working Group
Internet-Draft:
draft-du-apn6-auto-encapsulation-adjustment-03
Published: 12 October 2022
Intended Status: Standards Track
Expires: 15 April 2023

Authors: Z. Du          P. Liu
         China Mobile   China Mobile

# Auto-adjustment of Encapsulation Information in APN6

## Abstract

This document introduces a method to adjust the encapsulation 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

As the development of 5G and the new emerging Internet services, such as live video streaming, the networks are facing a larger and larger SLA requirement difference. For better bearing of the user's traffic, the networks need to be intelligent and be aware of the user traffic's demand. An innovative method called APN6 is introduced in [I-D.li-apn6-problem-statement-usecases] and [I-D.li-apn-framework].

In the mechanism of APN6, the packet can carry the ID information and SLA requirements of the traffic, and network equipment can get them in each packet and handle the packet accordingly. It is one kind of network programming mechanisms on the data plane.

As the encapsulation information increases in an APN packet, some bandwidth is kindly wasted in APN6 which contains a larger overhead in every packet. On one aspect, it is believed that it is necessary for the evolution to an intelligent network; on the other aspect, it is recommended that after the network has known the requirements of the traffic and associated it with a proper policy, the traffic does not need to resend the same information in every packet again and again. This document describes the process of the later, and discusses two potential solutions for the auto-adjustment of the encapsulation information in APN.

## 2. Current Mechanism in APN6

As shown in Figure 1, the APN framework [I-D.li-apn-framework] includes Service-aware App, App-aware Edge Device, 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 APN packets each carries the application characteristic information in the encapsulation. In this document, we also call the APP or the Edge as the encapsulation node.

App-aware-process Head-End can read that information and steer the packets into a given policy which satisfies the application's SLA 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. In this document, we also call the Head-End as the mapping node.

## 3. Comparisons of Data Plane and Control Plane Programming

We can realize the same traffic steering on the control plane. The control-plane based process, as described below, includes three key components: the identity of the traffic, policies in Head-End, and the interface to notify the user requirements.

The APP or the Edge knowing the application characteristic information needs to report that information to the controller of the Head-End by some means.

The controller needs to know the traffic requirements and the status of the network, and generate a policy for the Head-End. The policy

SHOULD include the identity of the traffic and the path that the traffic should follow.

The Head-End needs to implement the policy, and steer the traffic to the proper path.

In this mechanism, we do not need to carry extra information in each packet, but need to generate control messages between the Edge and the controller, and between the Head-End and the controller.

In the situation that the traffic is small, and simple to handle, a control-layer decision-loop is not that necessary. By comparison, a date-driven method is more flexible. In this situation, the Head-End after steering the traffic needs to report the (summarized) change to the controller.

## 4. Potential Solutions for Auto-adjustment

We can find that after the Head-End has selected the policy, the extra information carried in the following APN6 packets has little use. Therefore, an auto-adjustment of encapsulation information mechanism may be helpful for the simplification of the following IPv6 packets.

According to [I-D.li-apn-framework], 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. Hence, at least, we can send only the application-aware identification information in the following APN6 packets without network performance requirements information.

Two methods to reduce the overhead of the APN packets are described below.

### 4.1. Triggered by a Timer

One straightforward method is that we firstly send full information in APN6 packets, and after several seconds, we send APN6 packets that only contain the necessary information, such as the application-aware identification information.

After receiving the first APN packet of the traffic from the encapsulation node, the mapping node can obtain the application related information from the packet. As talked before, the information includes the application-aware identification and network performance requirements, and accordingly the mapping node establishes a mapping relationship between the traffic and a proper tunnel or policy.

In this method, we believe that the mapping node can handle the policy mapping process in the several seconds. For example, it can be three seconds. The number should be a parameter that can be adjusted according to the situation of each network. In this solution, a timer is needed in the encapsulation node. It is started after the first APN packet of the traffic is sent. When the timer expires, the encapsulation node will consider that the mapping node has finished the mapping job.

If all nodes work well, after the several seconds, the mapping node will receive APN packets of the traffic from the encapsulation node that contain only the necessary information, for example the application-aware identification information. In addition, the APN related information in the packets can only contain the necessary flow ID information, which is a part of the application-aware identification information.

### 4.2. Triggered by a Notification Message

Another method is that after enabling the policy, the mapping node can notify the encapsulation node by some means. However, we do not have a notification mechanism between different nodes on the data-plane network programming now. We need to notify by using the control plane again. The control plane sends a message to the encapsulation node to adjust the encapsulation degree.

This document suggests enabling a simple notification method for the data-plane network programming if the information is not that complicated. For example, we can send a "ping" message with a specific flag to the encapsulation node. The advantage is easy to inter-operate.

In this solution, the encapsulation node will not need a timer, and instead it can receive a notification message from the mapping node. After that, the encapsulation node can make sure that the mapping node has finished the mapping job between the traffic and a proper tunnel or policy, and starts to send APN packets of the traffic with simplified information.

## 5. Deployment Consideration

In future, with the technical development of network equipments, the bandwidth may not be the bottleneck anymore, so that a full APN6 encapsulation packet may be used widely to enable the data plane intelligence. However, the auto-adjustment of encapsulation information method can help the adoption of the APN6 mechanism by providing a transit solution. Meanwhile, this document also provides a feedback mechanism for the data plane programming to enable the coordination between two nodes.

## 6. IANA Considerations

TBD.

## 7. Security Considerations

TBD.

## 8. Acknowledgements

TBD.

## 9. References

### 9.1. Normative References

**[I-D.li-apn-framework]** Li, Z., Peng, S., Voyer, D., Li, C., Liu, P., Cao, C., and G. S. Mishra, "Application-aware Networking (APN) Framework", Work in Progress, Internet-Draft, draft-li-apn-framework-06, 30 September 2022, <https://www.ietf.org/archive/id/draft-li-apn-framework-06.txt>.

**[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>.

### 9.2. Informative References

**[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)", Work in Progress, Internet-Draft, draft-li-apn6-problem-statement-usecases-01, 3 November 2019, <https://www.ietf.org/archive/id/draft-li-apn6-problem-statement-usecases-01.txt>.

## 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