Abstract

Various protocols are deployed in today’s networks, such as BGP / ISIS / OSPF etc. Link neighbor state changes and protocol neighbor state changes are the most important network events that need to be processed with the highest priority. In particular, the SDN controller needs to quickly sense the link neighbor & protocol neighbor state change information in the network. Thus, the various policies applied by the SDN controller to the network can quickly match the current state of the network. This document discusses some possible scenarios and the relevant requirements.

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2019.
1. Introduction

Various protocols are deployed in today’s networks, such as BGP / ISIS / OSPF / LDP / BFD etc. When managing a network, one of the most important things to monitor is changes to the various protocols’ neighbor states. Many times a protocol neighbor state change is indicative of a problem on the network, and it is an important basis for the SDN controller to deploy the traffic steering policies. There are several ways to monitor these state changes, e.g. we can use command-line interface (CLI) to get them from the devices, but typically it’s done with either SNMP based polling and/or SNMP traps. For BGP, we can use BMP (BGP Monitoring Protocol) [RFC7854] to collect BGP neighbor state change information.

Link neighbor state changes and protocol neighbor state changes are the most important network events that need to be processed with the highest priority. In particular, the SDN controller needs to quickly sense the link neighbor & protocol neighbor state change information in the network. Thus, the various policies applied by the SDN controller to the network can quickly match the current state of the network.
The problem of the current real-time data collection method: Collecting protocol neighbor state will also collect many other large amounts of information data associated with it and have a significant impact on the reception/processing of high priority protocol neighbor state data. E.g., the processing of the BMP Peer Up/Down message is not real-time, and is affected by the receiving and processing of other BMP messages, especially a large number of route monitoring messages.

At present, the SDN controller uses a single channel to receive real-time data from the network, and then classifies the data and processes it in order, which causes the delay of the neighbor state information processing to grow; and the neighbor information data structure of different protocols are different; these cases will increase the delay in which the SDN controller processes neighbor state data.

2. Requirements and Options

Summary of requirements are as follows:

Requirement 1: Network event prioritization, Set Link neighbor state changes and protocol neighbor state changes as the most important network events.

Requirement 2: The structure of the link neighbor & protocol neighbor state change information needs to use a normalized format, such as a unified TLV.

Requirement 3: Link Neighbor & Protocol Neighbor State Change Information requires a separate transport channel to be separated from other low priority data.

Requirement 4: SDN controller implements the convergence mechanism of the existing network protocol in milliseconds/second.

Some options to be discussed:


2) Consider processing all neighbor states into LS information, flooding them through IGP, and collecting them on the controller through BGP-LS.

3) Consider processing all neighbor states into LS information, importing them to the BGP-LS Local-RIB, and collecting them on the controller through BMP.
4) gRPC + YANG Model.

5) To be added...

3. Neighbor State Information Format

At present, the neighbor information data structure of different protocols are different. In order to speed up processing in controller or collector, this document proposes to use a normalized format as following:

```
+---------------------------------------+
<table>
<thead>
<tr>
<th>Protocol Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node-IP Address</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Local-IP Address</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Peer-IP Address</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Neighbor State</td>
</tr>
<tr>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Timestamp</td>
</tr>
</tbody>
</table>
+---------------------------------------+
```

Figure 1 Link Neighbor/Protocol Neighbor State Information Format

Where:

Protocol Type: 1: ISIS / 2: OSPF / 3: BGP / 4: LDP / 5: BFD etc.

Node-IP Address: The IP Address of the monitored node, usually the router ID.

Local-IP Address: Local-IP Address of the Neighbor

Peer-IP Address: Peer-IP Address of the Neighbor

Neighbor State: The state of the Neighbor

Timestamp: The timestamp of the moment of the event

4. Acknowledgements

The authors would like to thank Haibo Wang, Zhongjia Wang for their contributions to this work.
5. Contributors

Mach Chen
mach.chen@huawei.com

6. IANA Considerations

TBD.

7. Security Considerations

TBD.

8. Normative References


Authors’ Addresses

Feng Xu
Tencent
Guangzhou
China

Email: oliverxu@tencent.com