Service Affinity Solution for TCP based Application in Anycast Situation

[draft-wang-tcpm-tcp-service-affinity-option]

Wei Wang (China Telecom)

Aijun Wang (China Telecom)

IETF 116@Yokohama, Mar. 2023

- Modifications after IETF115
- Considerations on MPTCP and TCP-EDO
- Further Actions

Modifications after IETF115

As the network status and computing ~~power~~ resources are constantly changing, different customers may be scheduled to different service nodes when accessing the same service. For customers who have established connections, the service node providing services must remain unchanged. Otherwise, a large number of state synchronization between service nodes are required to maintain the consistency of application data in the process of two-way connection communication.

The traditional solutions have two main methods:

* Maintain the customer-based connection status table in each router along the path. This table will not change dynamically with the change of network status and computing ~~power~~ resources, so that the subsequent packets will be transmitted along the same path.

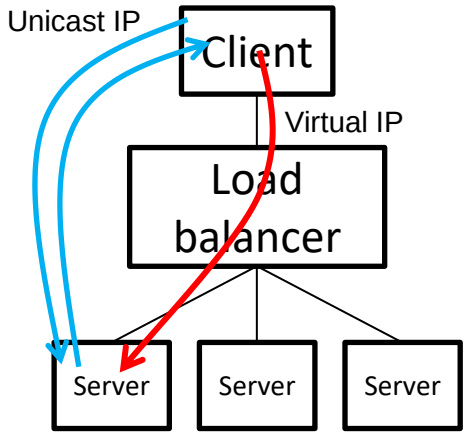
* Maintain the customer-based connection status table in ingress and egress routers. The packets need to be forwarded through tunnels on the intermediate routers.

The above solutions based on the connection status table are lack of flexibility and extensibility. The network devices should keep large amounts of status table to keep the service affinity for every customer flow. For large-scale service deployment, if the network status changes, it is easy to affect the customer experience.

Besides, in the load balance scenario, a load balancer is usually put in front of all the physical servers so that all the packets sent and received by the physical servers should pass through the load balancer. This deployment may lead to the load balancer become the bottleneck when the traffic increases. Direct traffic redirection and traffic scheduling between the client and server can avoid the bottleneck of load balancer.

MPTCP enables hosts to send packets belonging to one connection over different paths, but it is confined to the MPTCP framework. We want to find one solution that can meet such requirements in more general manner for TCP based application.

We propose a solution for the service affinity between client and server based on one newly defined TCP Option, which can realize the comprehensive scheduling based on real-time status of network and computing ~~power~~ resources. This solution eliminates the need to maintain customer-based connection status tables for network devices, and improves the feasibility and extensibility of large-scale deployment of ~~computing power~~ **computing-aware traffic steering** network.



- Add the load-balance scenario, the typical load balancer could become the bottleneck of the scenario. Our solution can alleviate this problem.
- Add the description of MPTCP, and the reason why we do not use MPTCP to meet our demands.

Modifications after IETF115

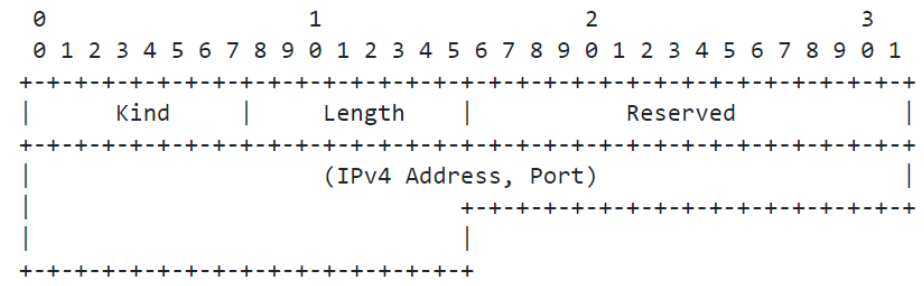


Figure 4: IPv4 Service Affinity option

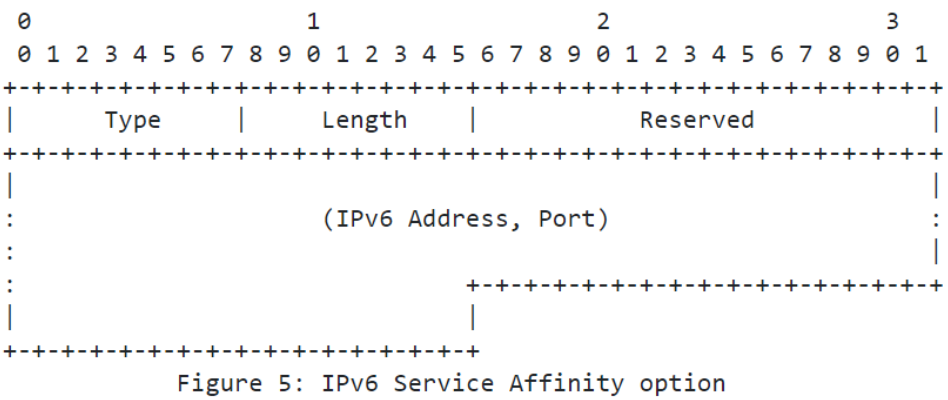


Figure 5: IPv6 Service Affinity option

- Split Service Affinity option into IPv4 Service Affinity option and IPv6 Service Affinity option.
- This modification can reduce the length of Service Affinity option and avoid wasting TCP header space with invalid fields.
- Which option to use depends on whether the server receives an IPv4 anycast address or an IPv6 anycast address.

Considerations on MPTCP and TCP-EDO

MPTCP

- MPTCP is a TCP extension enables a host send packets belonging to a single connection through different paths.
- MPTCP provides fast handover and bandwidth aggregation.

Our considerations

- In terms of function, MPTCP can solve the anycast traffic scheduling in the scenario we proposed. But MPTCP defines a new MPTCP framework, and we prefer to perform fast service path handover based on the traditional TCP three-way handshake process. The MP-TCP based solution requires establishing a connection before the path handover.
- The solution we proposed just set the "SAF" flag in SYN packet when sending to the anycast IP address, identifying it can support Service Affinity Option. If the server receives the SYN packet and it support Service Affinity Option, it will send a TCP FIN packet contains its unicast IP address in IPv4/IPv6 Service Affinity Option to the sender. The sender can establish the connection to the server via the unicast IP address. This process can be completed during three-way handshake.

Considerations on MPTCP and TCP-EDO

TCP-EDO

- TCP supports headers with a total length of up to 60 bytes. The default TCP header occupies 20 bytes, so that the length of options cannot exceed 40 bytes.
- Multiple options were defined to realize various capabilities, which may cause the total length of options exceeds 40 bytes when several options are carried in the same TCP packet. TCP-EDO extends the space available in TCP header for non-SYN segment.

Our considerations

- The TCP header extension based on TCP-EDO requires establishing a connection at first, which may affect the efficiency of fast handover.

Considerations on MPTCP and TCP-EDO

- We prefer to define a solution could support fast handover.
  - Both MPTCP and TCP-EDO need to establish a TCP connection first.
  - Service Affinity option can complete path choice during TCP connection establishment, which is more efficiency.

- For the extension of TCP header, we suggest the Service Affinity option and EDO supported option can be carried in SYN/ACK and FIN. The other options that are not related to fast handover can be carried in the extended part. This allows for both fast handover and TCP option available space extension.

Further Action

- Comments?

wangw36@chinatelecom.cn
wangaj3@chinatelecom.cn
IETF116@Yokohama