Abstract

This document proposes a new relay-based NAT traversal architecture called OARS which could simplify the data communication process.

<Wang,Liu,Uberti&Ding> Expires September 17, 2018
This document proposes a new relay-based NAT traversal architecture called OARS based on the following motivations.

1) Leverage ISPs’ infrastructures

Currently, the deployment of TURN [RFC5766] is very limited and most of the application providers use their own platform to transfer the data between two hosts that behind NATs and to translate the communication packets between two hosts in different address families.

The relay devices deployed centrally by various application providers often lead to inefficient data transmit between two hosts and it must
On the other hand, service providers have deployed many CGN/CDN nodes in a distributed manner within their networks. If the service providers can use these CGN devices/CDN nodes as the relay devices for communication between two hosts behind NATs or that from different address family, and provide their data translation/forwarding capability to the application providers, the host to host communication will be more efficient. Given most of the CGNs are capable of translating packets between IPv4 and IPv6, the adoption of IPv6 technology will also be accelerated.

2) Simplify the communication procedures

OARS needs less communication procedures than TURN of which the procedures are considered very complex to be integrated into the ISPs' infrastructure, for example:

- TURN solution has to closely interact with ICE
  Within current TURN solution, there are scenarios where the ICE or other NAT-hole punching procedures must be included for the success of communication via TURN servers. The key point is that TURN allocates different relay transport address-port pairs for different hosts.

  Each client must first use TURN allocation request to get their transport relay address-port pairs, and then must use ICE procedure (connectivity check) or other similar signaling to punch holes for these transport relay addresses on the alongside NAT devices. Or else the relayed UDP/TCP packet will be blocked.

  Even with the above procedures, there still exist some risks that the packets be rejected by TURN server due to the permission list that created by client via "CreatePermission Request" before it sending data to the peer. In order to mitigate such issues, current TURN solution requires the TURN servers only check the IP address part of the relay transport address, and ignore the port portion. But this will again introduce some attack risks because different host may share one public IP address when the CGN device is deployed within network.

- IPv4/IPv6 Relay Address/Port Reservation and Allocation
Another drawback of different relay transport addresses for different host is that the TURN server must reserve some IPv4/IPv6 address block for the allocation and plan the TCP/UDP port usage for each host. When TURN servers are deployed in a distribute manner (For example when they are incorporated into the CGN devices), there will be much coordination work to do for the address/port reservation and allocation on the TURN servers.

- Simultaneous TCP/UDP connections burden on TURN server

Current TURN solution requires the TURN servers to open and listen on many TCP/UDP ports at the same time. Under TURN solution for TCP [RFC6062], each host requires two connections to the TURN server. This will increase the burden on TURN server and the complexity to incorporate them into the CGN/CDN devices.

- Different procedures for TCP/UDP communication

Current TURN solution adopts different procedures for the TCP and UDP communication channel. So for one TURN server to provide the TCP/UDP relay function, it must implement two different procedures. This again increases the complexity of the TURN server implementation, especially in CGN devices.

- Communication complexity between two different TURN servers

Current TURN solution cannot assure two hosts select the same TURN server, and then it must deal with the communication situation between two different TURN servers. This scenario has not been covered by the current TURN related drafts. The client must reuse the XOR-PEER-ADDRESS attribute to include the relay address of the peer to reach the second TURN server.

On the other hand, because the hosts select their own TURN server, there is no mechanism to assure the relayed path is most optimal for them. The application latency will be increased when this occurs.

OARS solution will simplify the above mentioned complexity and make the TCP/UDP data relay function be easily incorporated into the existing distributed CGN devices or other kinds distributed devices i.e. the CDN nodes etc.
1.2. Relationship with TURN

This document doesn't intend to replace TURN with OARS, but consider OARS as a complementary solution along with TURN for some specific scenarios.

If one SP wants to open its infrastructure to accelerate their customers' (mainly regarding to application providers) client-to-client communications within the SP's domain, OARS could be a good candidate.

2. Conventions used in this document

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].

O Relay Selector: which is in charge of selecting a proper relay device (CGN or CDN nodes) for the communicating hosts behind NATs. Normally, the RS is a function located in the network’s management plane and possibly a part of the NMS server.

O OARS: Operator-Assisted Relay Service. Compared with the relay services that implemented independently by each TURN client, OARS can simplify the relay procedures in central control mode via the assistance of network operator.

O OARS Client: The client that initiated the Couple() message to bind two TCP/UDP sessions on one relay device or two different relay devices.

O OARS Server: The server that implemented the Couple() message to bind two TCP/UDP sessions on one relay device or two different relay devices.

3. Solution Overview

3.1. Reference Architecture of OARS

```
+---------------+      +---------------+      +---------------+
|                |      |            RS  |      |                |
|                |      | (Relay Selector)|      |                |
+---------------+      +---------------+      +---------------+
```

<Wang,Liu,Uberti&Ding> Expires September 17, 2018 [Page 5]
As depicted in above figure, the application clients that located on hosts act as the OARS clients while the CGNs act as OARS Servers. There is a Relay Selector (RS) for choosing a proper CGN to relay traffic between the two hosts. In practice, the RS could be a dedicated server or a function located in the management plane servers such as NMS server or SDN Controller.

RS has the intelligent route selection capability to choose a proper CGN for a given host pair. It generate the communication token upon the request of Application server, transfer these tokens to the relay devices to authenticate and authorize the communication between OARS clients(host) and OARS server(Relay/CGN devices).

BEHAVE compliant and non-BEHAVE compliant NAT traverse [RFC4787] [RFC5382] is supported in OARS. IPv6 and IPv4 host communication is also supported.

3.2. Solution Rationale

The solution could be briefly described in the following sections.
3.2.1. Get_RS_Token()

When clients register to their server, they will get the ip address of appropriate Relay Selector from the service provider, together with the token for the further usage of relay (CGN device) service. The token is generated dynamically via the communication between application server and RS, which is out the scope of this draft.

3.2.2. Get_Optimal_Relay()

Upon receiving the RS information, the clients will send Get_Optimal_Relay(RS, AP Reflex_Pair, Token) message to the RS, with the AP Reflex_Pair (client’s reflective address to the App server) and allocated token as the parameters, to let the RS select on optimal relay device for the clients.

3.2.3. Get_Relay_Reflex()

Client will send the Get_Relay_Reflex (Optimal_Relay, Token) message to the optimal relay, get its reflective address to this relay, and exchange such information with the peer via the Application server.

3.2.4. Couple()

Client then send the Couple(Relay_A, Reflex Relay_A, Relay_B, Reflex Relay_B, Token) message to the optimal relay, build the mapping table on such relay, with the relay address, the reflective address to them and token as the parameter.

3.2.5. Data()

Client can now send data directly to relay via Data(Relay, Token) message. Upon receiving the user data, the relay will change the source and destination address of the data packet respectively according to the mapping table built by previous Couple message. The return data will follow the same procedure.

The procedure is same regardless of the communication peer are located in the same SP or different SPs.
4. Detailed Example

4.1. Procedures of Communication between two IPv4 hosts

When one of the communication hosts located behind the symmetric NAT device, the host-to-host communication must via one relay device. Below are the key procedures of OARS solution, we use the Fig3-1 to describe the example.

1) If Host A and Host B want to communicate with each other, they will send Get_RS_Token() message to the application server, to get the IPv4 address/port of RS and the token for further communication with the relay device.

2) Host A and Host B will request the RS to select one optimal relay device for their communication, based on the host’s reflective address to the Application server. In this example, we assume the CGN-1 is selected for Host A and CGN-2 is selected for Host B.

3) Host A and Host B will send Get_Relay_Reflex message to CGN-1 and CGN-2 respectively, get their relay address to CGN[REFLX-Relay] and exchange them via the Application server.

4) Host will send the Couple message to the optimal relay, which includes mainly the [REFLX-Relay] addresses of Host A and Host B and their communication protocol, here we assume they use TCP to communicate.

5) Upon receiving the Couple message, the CGN device will form one forwarding table that look like below:

<table>
<thead>
<tr>
<th>Reflex_A</th>
<th>Relay_1</th>
<th>T_Relay_1</th>
<th>Reflex_B</th>
<th>Relay_2</th>
<th>T_Relay_2</th>
<th>Protocol</th>
</tr>
</thead>
</table>

Table 5-1: Couple Table Example (symmetric case)

6) Host A will send the application data to the relay transport address in CGN-1(Relay_1).

7) CGN-1 device will look up the Couple table, use the source address of received packet(Reflex_A in this example) to get the reflex IPv4 address of Host B.
8) It then will change the source address of the packet to the relay transport address in CGN-2 device (Relay_2), the destination address of this packet to the IPv4 reflex address of Host B(Reflex_B), then encapsulated in the tunnel, with T_Relay_1 as the tunnel source address and T_Relay_2 as tunnel destination address. The translated and tunneled packet will be forwarded to CGN-2(Relay 2).

9) CGN-2 device will decapsulate the packet, send the decapsulate packet to Reflex_B. This packet will pass the NAT device, be translated by the NAT and then be forwarded to the Host2.

10) The return traffic will be sent to CGN-2(Relay_2). Upon receiving the return packet, CGN-2(Relay_2) will change the source address to Relay_1, the destination address to Reflex_A, according to the mapping table, and then encapsulate it into one tunnel packet, with the T_Relay_2 as tunnel source address and T_Relay_1 as tunnel destination address.

11) Relay_1 will decapsulate the packet, send the decapsulate packet to Reflex_A. This packet will pass the NAT device, be translated by the NAT and then forward to the Host1.

4.2 Procedures of IPv4 and IPv6 Host Communication

If Host A is one IPv4 node and Host B is one IPv6 node. The communication process is similar, except that the relay address of Host A is the IPv4 address of the CGN-1 and the relay address of the Host B is the IPv6 address of the CGN-2. Host A and Host B will not notice that they are talking to one node that in different address family.

The mapping table is same except that the Reflex_A/Relay_1 is belong to IPv4 address family and the Reflex_B/Relay_2 is belong to IPv6 address family. The T_Relay_1 and T_Relay_2 should be in the same address family, and are determined by the operator themselves.

5. OARS Benefits

Comparing to TURN, OARS could provide following benefits:

- Decoupled from ICE

TURN is tightly coupled with ICE. Operations like NAT punching, create permission .etc all require ICE connectivity check packets. Benefited from the couple operation, OARS doesn’t necessarily need ICE interaction.
OARS doesn’t need to allocate different address-port pair for each session initiated from the hosts. Thus, it could obviously reduce the resource consumption and the human burden for planning the resource allocation.

- Unified solution for TCP/UDP and IPv4(6)-IPv6(4) data relay
  OARS supports the data relay for the communication between two hosts, uses same mechanism for TCP and UDP transport protocol, even for the communication between different address families.

- Support for optimal relay selection
  Because of the central deployed RS could have the whole network’s routing/path knowledge, OARS is more likely to achieve an optimal relay (OARS server) selection based on various information such as the reflective transport addresses of the two communicating peers, the link usage information between two peers and the load status of the candidate TURN-Lite servers etc.

With the RS's knowledge, OARS is also more likely to achieve better relay selection for some specific requirements. For example, if one peer wants to hide its ip address to protect its privacy, the RS in OARS architecture could possibly select one OARS server that located far away from the host.

6. OARS Deployment Considerations

The OARS Server can be deployed in distributed manner. The most appropriate devices for incorporating this function are the CGN devices that have been deployed distributed by the service provider. Each distributed OARS Server has one unique public IPv4/IPv6 transport address.

The RS can select the appropriate OARS Server based on the proximity of the OARS server with the communication hosts and can also use other criteria to influence the selection procedure, as described in Section 3.

7. Security Considerations

The additional requirement of OARS is authenticating the messages between the OARS Client, RS and Relay device. Currently, we use the
RS allocated token to accomplish this task. Because such token is allocated dynamically, such security risks can be mitigated accordingly.

8. IANA Considerations

TBD.

9. Conclusions

Currently, the communication between two hosts that located behind NAT devices, especially that behind the symmetric NAT devices is emerging. With the development of IPv6 technology, the communication between two hosts that in different address families needs also be considered. Under the OARS architecture, the communication requests for IPv4/IPv4, IPv4/IPv6 scenario can be met in one general solution. Such solution can alleviate the burden of various CP/SP to deploy the TURN server by themselves, exploit and open the capabilities of CGN device that deployed by service provider to the third party(CP/SP), make the host-to-host communication more efficient.

10. Acknowledgements

Many valuable comments were received from Brandon Williams, Oleg Moskalenko, Jonathan Rosenberg, and Dan Wing etc.

11. References

11.1. Normative References


11.2. Informative References


Authors’ Addresses

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing, 102209
Email: wangaj.bri@chinatelecom.cn

Bing Liu
Huawei Technologies
Q14, Huawei Campus, No.156 Beiqing Road, Hai-Dian District
Beijing, 100095
P.R. China
Email: leo.liubing@huawei.com

Justin Uberti
Google
747 6th Ave S
Kirkland, WA 98033
USA
Email: justin@uberti.name