

tram
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2015

A. Wang
China Telecom
B. Liu
Huawei Technologies
October 27, 2014

A Lightweight TURN Architecture and Specification (TURNLite)
draft-wang-tram-turnlite-01

Abstract

This document proposes a lightweight TURN architecture which simplifies the application provider side complexity of implementing TURN server by transferring the data relay processing to the ISP infrastructure (e.g. CGN). To achieve this goal, a new "Couple" operation using STUN message format is also defined.

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 <http://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 30, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language and Terminology	3
3. Targeted Problems	4
3.1. Application Providers Need to Deploy TURN servers individually	4
3.2. Dedicated Solutions for Different Requirements	4
4. Solution Overview	5
4.1. TURNLite Reference Architecture	5
4.2. Solution Rationale	6
5. TURNLite Communication Procedures	6
5.1. Procedures of Communication Traversing Symmetric NATs	6
5.2. Procedures of IPv4 and IPv6 Host Communication	8
6. STUN Couple Command Definition	9
6.1. Couple Opcode	10
6.2. Couple Operation Packet Format	10
7. TURNLite Features	12
8. Deployment Consideration	13
9. Security Considerations	13
10. IANA Considerations	14
11. Acknowledgements	14
12. References	14
12.1. Normative References	14
12.2. Informative References	14
Authors' Addresses	15

1. Introduction

With the depletion of public IPv4 addresses and the deployment of more and more NAT devices in real network, the communication between two hosts that located behind NAT devices is becoming more difficult. IPv6 adoption and the coexistence of IPv4 and IPv6 hosts within one network exacerbate this issue. TURN [RFC5766] technology is aim to solve these problems, but currently its deployment is very limited and most of application provider user their own platform to transfer the data between two hosts that behind NAT environment and to translate the communication packets between two hosts in different address family.

The data relay device deployed centrally by various application providers often lead to the inefficient data transmit between two hosts. The relay device must deal with complex network layered problems with which the application providers are not familiar.

On the other hand, service provider deploys many CGN devices in a distributed manner within their networks. If the service provider can use these CGN devices as the relay devices for communication between two hosts behind NATs or that from different address families, and open their data translation/forwarding capability to the application providers, the host to host communication will be easier and more efficient, and the deployment of IPv6 technology will also be accelerated.

This document proposes a lightweight TURN architecture which simplifies the application provider side complexity of implementing TURN server by transferring the data relay processing to the ISP infrastructure (mainly regarding to CGN (Carrier Grade NAT)). To achieve this goal, a new "Couple" operation using STUN message format is defined. The "Couple" operation could make the CGNs be able to couple two separate TCP/UDP connections that from the host to CGN device and relay data between two hosts. The "couple" operation could be considered as a common interface that invoked by the application providers to exploit the capabilities of CGN devices that deployed in a distributed manner by service provider.

2. Requirements Language and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] when they appear in ALL CAPS. When these words are not in ALL CAPS (such as "should" or "Should"), they have their usual English meanings, and are not to be interpreted as [RFC2119] key words.

- o Application Provider: the service providers who provide client to client communications through the Internet. E.g. VoIP service providers, instant message service providers etc.
- o TURNLite: lightweight TURN architecture. The word "lightweight" is in the perspective of an application provider.
- o TURNLite Client: the TURNLite entity that deployed in the application providers' networks; be responsible for TURNLite signaling/control interactions with the TURNLite servers.
- o TURNLite Server: the TURNLite entity that deployed in the ISP's networks; be mainly responsible for the data relay between an application providers' clients. Normally, the TURNLite servers collated with the CGNs (Carrier Grade NATs) within the service provider.

3. Targeted Problems

3.1. Application Providers Need to Deploy TURN servers individually

In real practice, TURN servers are deployed and managed by each application provider separately within their own networks. These TURN servers are responsible for all the signaling and data relay function between their clients. Thus, the application providers might face the following challenges:

- o The application providers need to be capable of dealing with large amount of relaying traffic, which is a significant burden for them.
- o At the early stage of an application/service development, normally the application provider is only able to deploy centralized TURN servers due to budget and capacity limitation. Thus, the traffic path between distributed clients through the central TURN servers might be inefficient.
- o TURN servers need to reserve a large amount of relay addresses which are expensive resources now and are difficult for the application providers to apply and manage.

These challenges can be mitigated by utilizing the CGN devices that deployed within the service provider's network to fulfill the data relay function needed by the application provider. The service provider is expert in the network/transport layer packet processing and has large amounts of resources to use and schedule. They can provide such capability to the application provider and alleviate them from the complicate and laborious work of deploying TURN server.

3.2. Dedicated Solutions for Different Requirements

Different client-to-client communication scenarios require dedicated adaption mechanisms respectively in current TURN solution. For example, for the following three communication scenarios, there are specific TURN extensions under developing.

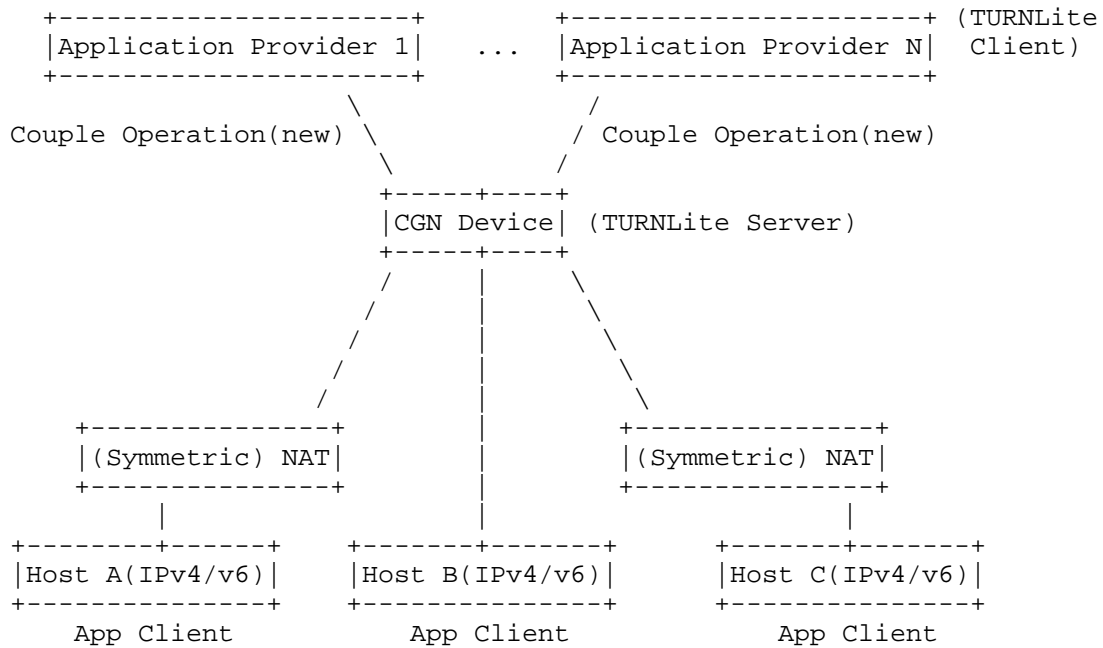
- o TCP-based client-to-client communication, see [RFC6062]
- o IPv6 client-to-client communication, see [RFC6156].
- o Host mobility, see [I-D.wing-tram-turn-mobility].

These dedicated extension mechanisms will obviously increase the complexity of implementing the TURN server. The complexity would become blocking factor of the TURN adoption by real practice.

Thus, there should be one lightweight TURN solution to cope with these challenges and provide one unified solution to fulfill the client-to-client communication under various scenarios.

4. Solution Overview

4.1. TURNLite Reference Architecture



As the figure shows, the main difference between the TURN and TURNLite is the data relay function of original TURN server is offloaded to the CGN device. the TURNLite Server/CGN device provides such capabilities to the Application Provider via the newly defined Couple operation. The Application Provider does not need to deploy a full TURN server individually, they only need to apply their requirements to the TURNLite Server/CGN device via the common control interface. The subsequent data between the application clients will be relayed by the CGN devices according to the couple information provided in the Couple operation.

Symmetric NAT traverse is supported in TURNLite. IPv6 and IPv4 host communication is also supported.

4.2. Solution Rationale

The solution could be briefly described as the following stages:

1) The hosts communicate with the application server respectively. The communication might be based on the application private protocol. At this stage, the key points are:

- The application server learns the reflex address of the hosts.
- The application server learns that Host A wants to communicate with Host C. Thus, it could correlates the reflex addresses of the two hosts.
- The application server selects a proper CGN based on the pair of reflex addresses and indicates the CGN address to host A and C.

2) Host A and C punch a hole in the NAT to the given CGN respectively (this could be a standard STUN [RFC5389] process, for example, sending the STUN binding messages to the given CGN). And report their reflex addresses that bound to the CGN's well-known transport address to the application server.

3) The application server use the newly defined "Couple" operation to couple the two newly acquired reflex addresses bound to the CGN reported by the hosts. Then the hosts could directly send the data to the CGN, and the CGN could relay the data correctly according to the couple information.

5. TURNLite Communication Procedures

5.1. Procedures of Communication Traversing Symmetric NATs

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 TURNLite solution, we use the similiar figure that described in [I-D.ietf-tram-turnbis] for comparison.

Please note that the figure does not include the application server which acts as the TURNLite client. The communication procedures between the hosts and the application server are not included as well, since they might be private communication protocol/mechanism developed by the application provider.

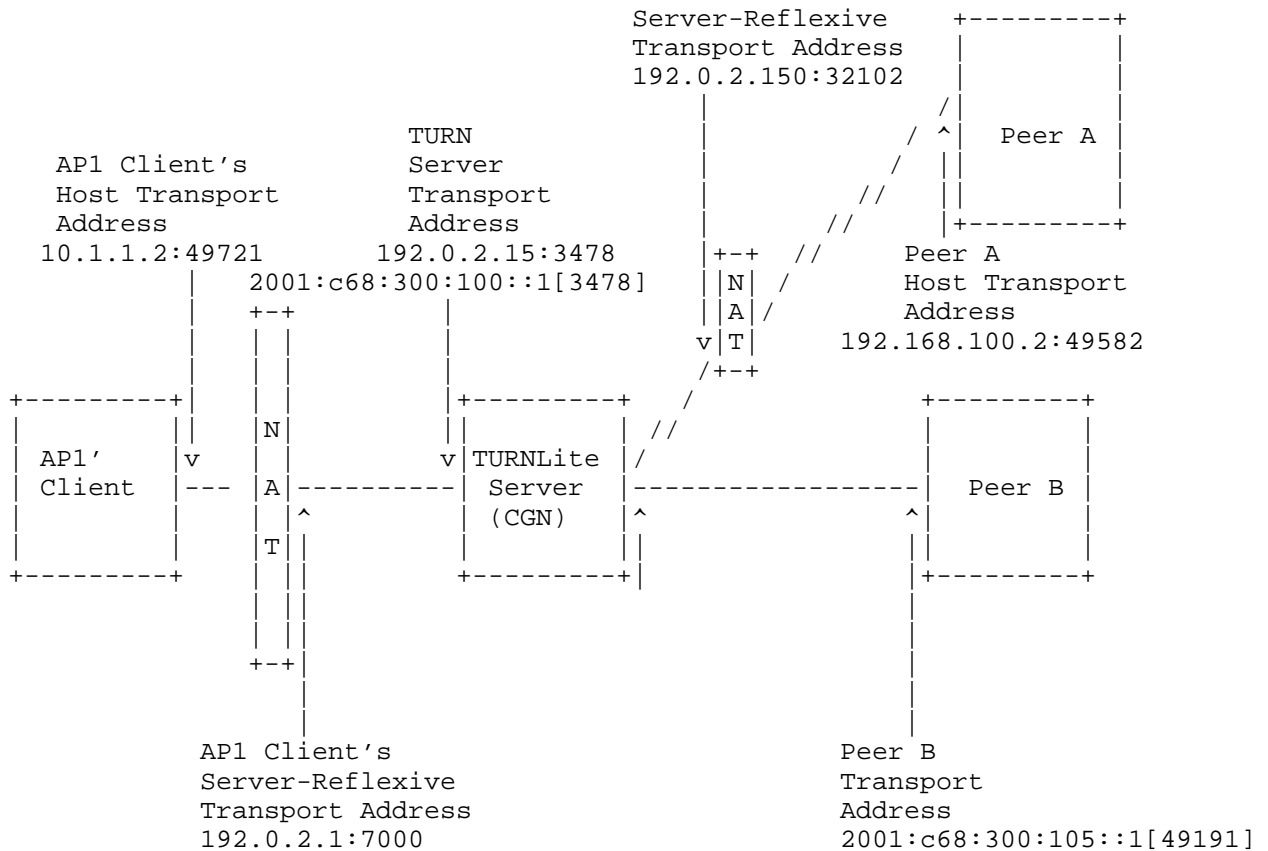


Figure 5-1 Example of TURNLite Communication Scenarios

Communication Procedures:

(Note: before the following step one, the two API's clients both have communicated with the application server. The communication might be based on the application private protocol. And the application server needs to select a proper CGN for the clients based on their reflex addresses the server learned. Currently, this document doesn't specifically consider these problems.)

1. If API's Client and Peer A want to communicate with each other, they will send STUN binding message to the well-known public IPv4 relay address/port (192.0.2.15:3478 in this example, and it is just the selected CGN's address), get their reflex public addresses to this relay address(that is 192.0.2.1:7000 and 192.0.2.150:32102 respectively).

2. AP1 client and Peer A should report their reflex address to Application Provider 1.
3. Application Provider 1 will use the mapped address of AP1 Client and Peer A to formulate one Couple message that defined in this draft, send it to the CGN device, let the CGN device to build one Couple table for AP1 Client and Peer A. The Couple table looks like below:

Reflexive transport address of AP1's Client	Reflexive transport address of Peer A	Transport Protocol
192.0.2.1:7000	192.0.2.150:32102	TCP/UDP

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

4. AP1 Client will send the AP1 application data to the well-known public IPv4 relay address/port(192.0.2.15:3478 in this example) of the CGN device.
5. CGN device will look up the Couple table that formulated in the step c), use the source address of received packet (192.0.2.1:7000 in this example) to get the reflex IPv4 address of Peer A. It then will change the source address of the packet to the well-known public IPv4 relay address of CGN device, the destination address of this packet to the IPv4 reflex address of Peer A.
6. The translated packet will be forwarded to the Peer A, processed by the AP1 client located on it.
7. The return traffic will also be sent to the well-known IPv4 relay address/port of CGN device, converted by the CGN device, and sent back to the Application Provider1's Client.

5.2. Procedures of IPv4 and IPv6 Host Communication

1. If AP1 clients and Peer B want to communicate with each other, they will send STUN binding message to the well-known public IPv4 relay address/port (192.0.2.15:3478 in this example), get their reflex public addresses to this relay address (that is 192.0.2.1:7000 and 2001:c68:300:105::1[49191]respectively).
2. AP1 client and Peer B should report their reflex address to Application Provider 1.

3. Application Provider 1 will use the mapped address of AP1 Client and Peer B to formulate one Couple message that defined in this draft, send it to the CGN device, let the CGN device to build one Couple table for AP1 Client and Peer B. The Couple table looks like below:

Reflexive transport address of AP1's Client	Reflexive transport address of Peer B	Transport Protocol
192.0.2.1:7000	2001:c68:300:105::1[49191]	TCP/UDP

Table 5-2: Couple Table Example (Case of different address families)

4. AP1 Client will send the AP1 application data to the well-known public IPv4 relay address/port(192.0.2.15:3478 in this example) of the CGN device.
5. CGN device will look up the Couple table that formulated in the step c), use the source address of received packet (192.0.2.1:7000 in this example) to get the reflex IPv6 address of Peer B. It then will change the source address of the packet to the well-known public IPv6 relay address of CGN device, the destination address of this packet to the IPv6 reflex address of Peer B.
6. The translated IPv6 packet will be forwarded to the Peer B, processed by the Application Provider1's client located on it.
7. The return traffic will also be sent to the well-known IPv6 relay address/port of CGN device, converted by the CGN device, and sent back to the AP1 Client.

The AP1 client and Peer B will not notice the other end is located in different address families. The CGN device finishes all the network/transport layer related translation work.

Such procedures are all same, regardless of the application using TCP or UDP transport, located in IPv4 environments or in IPv6 environments.

6. STUN Couple Command Definition

In order to let the CGN device to build one Couple item upon the request of Application Provider, it is needed to define one Couple message to transfer the related information.

6.1. Couple Opcode

The Couple request defines the relationship between two TCP or UDP half-connections, the translation rule that converts both the source address and destination address of pass through packet in both directions.

Couple Opcode: It defines how to bind two half-connections that ends at the CGN well-known transport address together. When CGN device receives the Couple request, it will create one map table item that includes the reflex IP address/port of both hosts that lies behind the NAT device and the protocol that the host will use to communicate.

When the CGN device receives the packet from one host, it will use the reflex source address/port to lookup the map table item; converts the source address/port of this packet to the well-known PCP server/port and converts the destination address/port of this packet to the address/port that results from the map table lookup action.

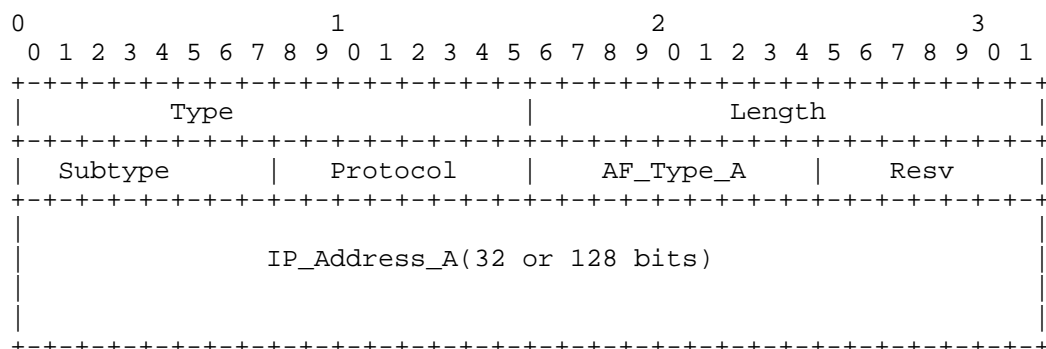
The converted packet will be routed to NAT side of the other host, NATED by the NAT device and then to the other host. The return packet will be delivered to the well-known IP address/port of CGN device and be converted in reverse accordingly.

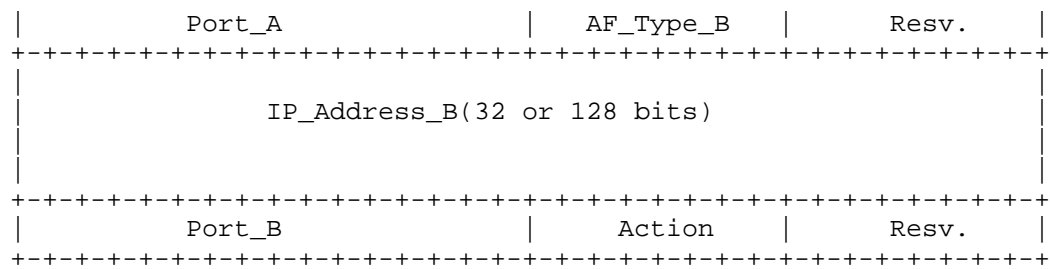
6.2. Couple Operation Packet Format

The Couple Opcode allows a TURNlite client to create a new explicit couple table on the CGN device (TURNlite Server), instructs the CGN device to accomplish the related translation work.

The following diagram shows the Opcode layout for the Couple Opcode request/response format.

Fig.7-1: Couple Opcode Request/Response Format





Type	The value should be newly applied from IANA.
Length	The total length of Couple packet.
Subtype	This field will differentiate the Couple Request/Response packet. The value 0x01 indicates it's a Couple Request packet; The value 0x01 indicates it's a Couple Response packet.
Protocol	Upper-layer protocol associated with this Opcode. Values are taken from the IANA protocol registry [proto_numbers]. For example, this field contains 6 (TCP) if the Opcode is intended to create a TCP mapping. This field contains 17(UDP) if the Opcode is intended to create a UDP mapping. The value 0 has a special meaning for 'all protocols'.
AF_Type_A	The reflex address family of host A. It will be 4 when the host's reflex address is IPv4 and will be 16 when the host's reflex address is IPv6. It actually represents the length of following ?IP_Address_A? field.
IP_Address_A	This field is the value of host A's reflex address. Specially, in symmetric NAT environment, the reflex address is related to the well-known IP address/port of TURNLite server. For IPv6 host, the reflex address is often same as the local IPv6 address of host A.
AF_Type_B	The reflex address family of host B. It will be 4 when the host's reflex address is IPv4 and will be 16 when the host's reflex address is IPv6. It actually represents the length of following ?IP_Address_B? field.
IP_Address_B	This field is the value of host B's reflex

address. Specially, in symmetric NAT environment, the reflex address is related to the well-known IP address/port of TURNLite server. For IPv6 host, the reflex address is often same as the local IPv6 address of host B.

Port_B	This field is the value of host B's reflex port. Specially, in symmetric NAT environment, the reflex address is related to the well-known IP address/port of TURNLite server.
Action	It distinguishes the add/delete action that the TURNLite client wants to apply to the TURNLite server. The value 0x00 indicates deleting the related Couple item; 0x01 indicates adding the related Couple item.
Reserved	Reserved byte, MUST be sent as 0 and MUST be ignored when received.

7. TURNLite Features

The TURNLite mainly aims to deal with the problems that are described in Section 3. TURNLite also supports some other good features as the following.

o Support of Symmetric NAT Traverse

Due to every host communicate with the well-known relay transport address, there is no additional requirement for punching holes in the NAT devices, which is indispensable for the current TURN solution.

o Native Support of Host Mobility

Since the host communicates with the well-known relay transport address, it will not sense the address change of the corresponding peer, then it support node mobility naturally. When the node move and application server detects it, the moving node only needs to resend the STUN binding command and report it to the application server. The application server just resends the Couple command to the TURNLite server; renew the Couple item within TURNLite server. There is no new command need to be defined.

o Less Relay Address Resource Consumption

It can conclude that the TURNLite solution simplifies the communication procedure apparently, and alleviate the burden of reserving and allocating large amount of relay address/port in TURN server.

o Simplified Procedures

Theoretically, TURNLite requires only two commands to accomplish the relay function, compared with over eight commands that required by TURN solution. Please see the figure below for detail.

	TURN Solution	TURNLite Solution
Required Commands	1. Binding 2. Allocate 3. Send 4. Data 5. Channel Bind 6. Connect 7. ConnectionBind 8. ConnectionAttempt	1. Bindiiing 2. Couple

8. Deployment Consideration

The TURNLite 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 TURNLite Server has one unique well-known IPv4/IPv6 transport address. These addresses can be obtained via the well-known FQDN of the TURNLite Server.

The application server can select the appropriate TURNLite Server based on the proximity of it with the communication hosts. The TURNLite Server selection policy can be controlled by the application server or service provider and is out of the scope of this document.

9. Security Considerations

The additional requirement of TURNLite is authenticating the couple operation from the TURNLite client to the TURNLite Server.

However, in theory the OAuth mechanism defined in current TURN solution could be reused in TURNLite. (Detailed considerations need further study.)

10. IANA Considerations

This draft requires IANA to allocate one new type value of STUN methods for the Couple command. (TBD)

11. Acknowledgements

This document was produced using the xml2rfc tool [RFC2629].

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.
- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.
- [RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, April 2010.

12.2. Informative References

- [I-D.ietf-tram-turnbis]
Reddy, T., Johnston, A., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", draft-ietf-tram-turnbis-00 (work in progress), August 2014.
- [I-D.wing-tram-turn-mobility]
Wing, D., Patil, P., Reddy, T., and P. Martinsen, "Mobility with TURN", draft-wing-tram-turn-mobility-02 (work in progress), September 2014.
- [RFC6062] Perreault, S. and J. Rosenberg, "Traversal Using Relays around NAT (TURN) Extensions for TCP Allocations", RFC 6062, November 2010.
- [RFC6156] Camarillo, G., Novo, O., and S. Perreault, "Traversal Using Relays around NAT (TURN) Extension for IPv6", RFC 6156, April 2011.

Authors' Addresses

Aijun Wang
China Telecom
China Telecom Coporation Limited Beijing Research Institute
No.118,Xizhimenneidajie,Xicheng District,Beijing, 100035
P.R. China

Email: wangaj@ctbri.com.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