BEHAVE	S. Perreault, Ed.	
Internet-Draft	Viagénie	
Intended status: Standards Track	G. Camarillo	
Expires: April 17, 2010	O. Novo	
	Ericsson	
	October 14, 2009	

TOC

Traversal Using Relays around NAT (TURN) Extension for IPv6
draft-ietf-behave-turn-ipv6-07

Status of this Memo

Copyright Notice

Abstract

This document adds IPv6 support to Traversal Using Relays around NAT (TURN). IPv6 support in TURN includes IPv4-to-IPv6, IPv6-to-IPv6, and IPv6-to-IPv4 relaying. This document defines the REQUESTED-ADDRESS-FAMILY attribute for TURN. The REQUESTED-ADDRESS-FAMILY attribute

allows a client to explicitly request the address type the TURN server will allocate (e.g., an IPv4-only node may request the TURN server to allocate an IPv6 address).

Table of Contents

---

1. Introduction

TOC

Traversal Using Relays around NAT (TURN) [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, “Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN),” July 2009.) is a protocol that allows for an element behind a NAT to receive incoming data over UDP or TCP. It is most

useful for elements behind symmetric NATs that wish to be on the receiving end of a connection to a single peer.
The base specification of TURN [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.) only defines IPv4-to-IPv4 relaying. This document adds IPv6 support to TURN, which includes IPv4-to-IPv6, IPv6-to-IPv6, and IPv6-to-IPv4 relaying. This document defines the REQUESTED-ADDRESS-FAMILY attribute, which is an extension to TURN that allows a client to explicitly request the address type the TURN server will allocate (e.g., an IPv4-only node may request the TURN server to allocate an IPv6 address). This document also defines and registers a new error response code with the value 440 (Address Family not Supported).

---

2. Terminology

TOC

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 [RFC2119] (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.).

---

3. Overview of Operation

TOC

When a user wishes a TURN server to allocate an address of a specific type, it sends an Allocate Request to the TURN server with a REQUESTED-ADDRESS-FAMILY attribute. TURN can run over UDP and TCP, as it allows for a client to request address/port pairs for receiving both UDP and TCP.
Assuming the request is authenticated and has not been tampered with, the TURN server allocates a transport address of the type indicated in the REQUESTED-ADDRESS-FAMILY attribute. This address is called the allocated transport address.
The TURN server returns the allocated address in the response to the Allocate Request. This response contains a XOR-RELAYED-ADDRESS attribute indicating the IP address and port that the server allocated for the client.
TURN servers allocate a single relayed-transport-address per allocation request. Therefore, Allocate Requests cannot carry more than one REQUESTED-ADDRESS-FAMILY attribute. Consequently, a client that wishes to allocate more than one address at a TURN server (e.g., an IPv4 and an IPv6 address) needs to perform several allocation requests (one allocation request per address).

A TURN server that supports a set of address families is assumed to be able to relay packets between them. If a server does not support the address family requested by a client, the server returns a 440 (Address Family not Supported) error response.

---

4. Creating an Allocation

TOC

The behavior specified here affects the processing defined in Section 6 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.).

4.1. Sending an Allocate Request

TOC

A client that wishes to obtain a transport address of a specific address type includes a REQUESTED-ADDRESS-FAMILY attribute, which is defined in Section 4.1.1 (The REQUESTED-ADDRESS-FAMILY Attribute), in the Allocate Request that it sends to the TURN server. Clients MUST NOT include more than one REQUESTED-ADDRESS-FAMILY attribute in an Allocate Request. The mechanisms to formulate an Allocate Request are described in Section 6.1 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.). Clients MUST NOT include a REQUESTED-ADDRESS-FAMILY attribute in an Allocate request that contains a RESERVATION-TOKEN attribute.

4.1.1. The REQUESTED-ADDRESS-FAMILY Attribute

TOC

The REQUESTED-ADDRESS-FAMILY attribute is used by clients to request the allocation of a specific address type from a server. The following is the format of the REQUESTED-ADDRESS-FAMILY attribute. Note that TURN attributes are TLV (Type-Length-Value) encoded, with a 16 bit type, a 16 bit length, and a variable-length value.

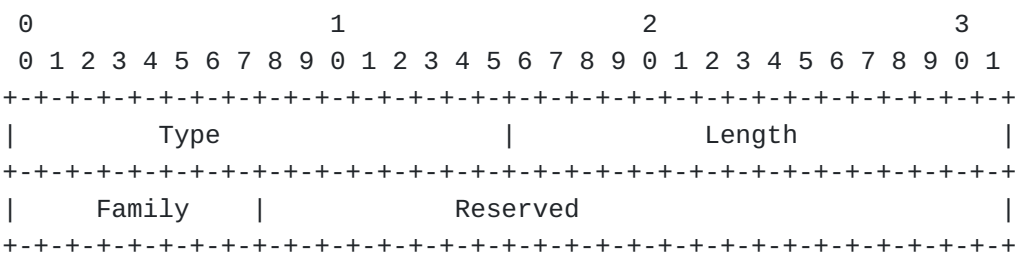

Figure 1: Format of REQUESTED-ADDRESS-FAMILY Attribute

Type: the type of the REQUESTED-ADDRESS-FAMILY attribute is 0x0017. As specified in [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)," October 2008.), attributes with values between 0x0000 and 0x7FFF are comprehension-required, which means that the client or server cannot successfully process the message unless it understands the attribute.
Length: this 16-bit field contains the length of the attribute in bytes. The length of this attribute is 4 bytes.
Family: there are two values defined for this field and specified in [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)," October 2008.): 0x01 for IPv4 addresses and 0x02 for IPv6 addresses.
Reserved: at this point, the 24 bits in the reserved field MUST be set to zero by the client and MUST be ignored by the server.
The REQUEST-ADDRESS-TYPE attribute MAY only be present in Allocate Requests.

4.2. Receiving an Allocate Request

TOC

Assuming the request is authenticated and has not been tampered with, the TURN server processes the Allocate request. Following the rules in [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)," October 2008.), if the server does not understand the REQUESTED-ADDRESS-FAMILY attribute, it generates an Allocate Error Response, which includes an ERROR-CODE attribute with response code 420 (Unknown Attribute). This response will contain an UNKNOWN-ATTRIBUTE attribute listing the unknown REQUESTED-ADDRESS-FAMILY attribute.
If the server can successfully process the request, it allocates a transport address to the TURN client, called the allocated transport address, and returns it in the response to the Allocate Request.

As specified in [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, “Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN),” July 2009.), the Allocate Response contains the same transaction ID contained in the Allocate Request and the XOR-RELAYED-ADDRESS attribute that sets it to the allocated transport address.
The XOR-RELAYED-ADDRESS attribute indicates the allocated IP address and port. It is encoded in the same way as the XOR-MAPPED-ADDRESS [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, “Session Traversal Utilities for NAT (STUN),” October 2008.).
If the REQUESTED-ADDRESS-FAMILY attribute is absent, the server MUST allocate an IPv4 transport address to the TURN client.
If the server does not support the address family requested by the client, it MUST generate an Allocate Error Response, and it MUST include an ERROR-CODE attribute with the 440 (Address Family not Supported) response code, which is defined in Section 4.2.1 (Unsupported Address Family).

4.2.1. Unsupported Address Family

TOC

This document defines the following new error response code:

> **440 (Address Family not Supported):** The server did not support the address family requested by the client.

4.3. Receiving an Allocate Error Response

TOC

If the client receives an Allocate error response with the 440 (Unsupported Address Family) error code, the client SHOULD NOT retry its request.

---

5. Refreshing an Allocation

TOC

The behavior specified here affects the processing defined in Section 7 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, “Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN),” July 2009.).

TOC

5.1. Sending a Refresh Request

To perform a binding refresh, the client generates a Refresh Request as described in Section 7.1 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.). The client MUST NOT include any REQUESTED-ADDRESS-FAMILY attribute in its Refresh Request.

5.2. Receiving a Refresh Request

TOC

If a server receives a Refresh Request with a REQUESTED-ADDRESS-FAMILY attribute, it MUST ignore the attribute and process the request as if the attribute was not there.

---

6. CreatePermission

TOC

The behavior specified here affects the processing defined in Section 9 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.).

6.1. Sending a CreatePermission Request

TOC

The client MUST only include XOR-PEER-ADDRESS attributes with addresses of the same address family as the relayed transport address for the allocation.

6.2. Receiving a CreatePermission request

TOC

If an XOR-PEER-ADDRESS attribute contains an address of an address family different than the relayed transport address for the allocation, the server MUST generate an error response with the 443 (Peer Address Family Mismatch) response code, which is defined in Section 6.2.1 (Peer Address Family Mismatch).

TOC

6.2.1. Peer Address Family Mismatch

This document defines the following new error response code:

> **443 (Peer Address Family Mismatch):** A peer address was of a different address family than the relayed transport address of the allocation.

---

7. Channels

TOC

The behavior specified here affects the processing defined in Section 11 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.).

7.1. Sending a ChannelBind Request

TOC

The client MUST only include a XOR-PEER-ADDRESS attribute with an address of the same address family as the relayed transport address for the allocation.

7.2. Receiving a ChannelBind Request

TOC

If the XOR-PEER-ADDRESS attribute contains an address of an address family different than the relayed transport address for the allocation, the server MUST generate an error response with the 443 (Peer Address Family Mismatch) response code, which is defined in Section 6.2.1 (Peer Address Family Mismatch).

---

8. Packet Translations

TOC

The TURN specification [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.) describes how TURN relays should relay traffic consisting of IPv4 packets (i.e., IPv4-to-IPv4 translations). The relay translates the IP addresses and port numbers of the packets based on the allocation's state data. How to translate other header fields is also specified in

[I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.). This document addresses IPv4-to-IPv6, IPv6-to-IPv4, and IPv6-to-IPv6 translations. TURN relays performing any translation MUST translate the IP addresses and port numbers of the packets based on the allocation's state information as specified in [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.). The following sections specify how to translate other header fields.

As discussed in Section 2.6 of [I-D.ietf-behave-turn] (Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," July 2009.), translations in TURN are designed so that a TURN server can be implemented as an application that runs in userland under commonly available operating systems and that does not require special privileges. The translations specified in the following sections follow this principle.

The descriptions below have two parts: a preferred behavior and an alternate behavior. The server SHOULD implement the preferred behavior. However, if that is not possible for a particular field, then the server SHOULD implement the alternative behavior.

> Note that the use of the behaviors specified in the following sections is at the "should" level. Having its use at the "should" level instead of at the "must" level makes it possible to use different translation algorithms that may be developed in the future.

8.1. IPv4-to-IPv6 Translations

TOC

Flow Label

> Preferred behavior: The relay sets the Flow label to 0. The relay can choose to set the Flow label to a different value if it supports [RFC3697] (Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification," March 2004.).
>
> Alternative behavior: the relay sets the Flow label to the default value for outgoing packets.

Hop Limit

>

> Preferred behavior: as specified in Section 3.1 of [RFC2765] (Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," February 2000.).
>
> Alternative behavior: the relay sets the Hop Limit to the default value for outgoing packets.

Fragmentation

> Preferred behavior: as specified in Section 3.1 of [RFC2765] (Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," February 2000.).
>
> Alternative behavior: the relay assembles incoming fragments. The relay follows its default behavior to send outgoing packets.
>
> If present, the DONT-FRAGMENT attribute MUST be ignored by the server.

Extension Headers

> Preferred behavior: the relay sends outgoing packet without any IPv6 extension headers, with the exception of the Fragmentation header as described above.
>
> Alternative behavior: same as preferred.

8.2. IPv6-to-IPv6 Translations

TOC

Flow Label

The relay should consider that it is handling two different IPv6 flows. Therefore, the Flow label [RFC3697] (Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification," March 2004.) SHOULD NOT be copied as part of the translation.

> Preferred behavior: The relay sets the Flow label to 0. The relay can choose to set the Flow label to a different value if it supports [RFC3697] (Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification," March 2004.).
>
> Alternative behavior: the relay sets the Flow label to the default value for outgoing packets.

Hop Limit

>

> Preferred behavior: the relay acts as a regular router with respect to decrementing the Hop Limit and generating an ICMPv6 error if it reaches zero.
>
> Alternative behavior: the relay sets the Hop Limit to the default value for outgoing packets.

Fragmentation

> Preferred behavior: If the incoming packet did not include a Fragment header and the outgoing packet size does not exceed the outgoing link's MTU, the relay sends the outgoing packet without a Fragment header.
>
> If the incoming packet did not include a Fragment header and the outgoing packet size exceeds the outgoing link's MTU, the delay drops the outgoing packet and send an ICMP message of type 2 code 0 ("Packet too big") to the sender of the incoming packet. If the packet is being sent to the peer, the relay reduces the MTU reported in the ICMP message by 48 bytes to allow room for the overhead of a Data indication.
>
> If the incoming packet included a Fragment header and the outgoing packet size (with a Fragment header included) does not exceed the outgoing link's MTU, the relay sends the outgoing packet with a Fragment header. The relay sets the fields of the Fragment header as appropriate for a packet originating from the server.
>
> If the incoming packet included a Fragment header and the outgoing packet size exceeds the outgoing link's MTU, the relay MUST fragment the outgoing packet into fragments of no more than 1280 bytes. The relay sets the fields of the Fragment header as appropriate for a packet originating from the server.
>
> Alternative behavior: the relay assembles incoming fragments. The relay follows its default behavior to send outgoing packets.
>
> If present, the DONT-FRAGMENT attribute MUST be ignored by the server.

Extension Headers

> Preferred behavior: the relay sends outgoing packet without any IPv6 extension headers, with the exception of the Fragmentation header as described above.
>
> Alternative behavior: same as preferred.

8.3. IPv6-to-IPv4 Translations

TOC

Type of Service and Precedence

> Preferred behavior: as specified in Section 4 of [RFC2765] (Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," February 2000.).
>
> Alternative behavior: the relay sets the Type of Service and Precedence to the default value for outgoing packets.

Time to Live

> Preferred behavior: as specified in Section 4.1 of [RFC2765] (Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," February 2000.).
>
> Alternative behavior: the relay sets the Time to Live to the default value for outgoing packets.

Fragmentation

> Preferred behavior: as specified in Section 4 of [RFC2765] (Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," February 2000.). Additionally, when the outgoing packet's size exceeds the outgoing link's MTU, the relay needs to generate an ICMP error (ICMPv6 Packet Too Big) reporting the MTU size. If the packet is being sent to the peer, the relay SHOULD reduce the MTU reported in the ICMP message by 48 bytes to allow room for the overhead of a Data indication.
>
> Alternative behavior: the relay assembles incoming fragments. The relay follows its default behavior to send outgoing packets.
>
> If present, the DONT-FRAGMENT attribute MUST be ignored by the server.

---

9. Security Considerations

TOC

The attribute and error response code defined in this document do not have any special security considerations beyond those for other attributes and Error response codes. All the security considerations applicable to STUN [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)," October 2008.) and TURN are applicable to this document as well.

10. IANA Considerations

TOC

The IANA is requested to register the following values under the STUN Attributes registry and under the STUN Response Code Registry.

10.1. New STUN Attribute Registry

TOC

0x0017: REQUESTED-ADDRESS-FAMILY

10.2. New STUN Response Code Registry

TOC

440 Address Family not Supported

11. Acknowledgements

TOC

The authors would like to thank Alfred E. Heggestad, Rémi Denis-Courmont, and Philip Matthews for their feedback on this document.

12. Normative References

TOC

[I-D.ietf-behave-turn]	Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," draft-ietf-behave-turn-16 (work in progress), July 2009 (TXT).
[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC2765]	Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," RFC 2765, February 2000 (TXT).
[RFC3697]	Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification," RFC 3697, March 2004 (TXT).
[RFC5389]	Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)," RFC 5389, October 2008 (TXT).

Authors' Addresses

TOC

	Simon Perreault (editor)
	Viagénie
	2600 boul. Laurier, suite 625
	Québec, QC G1V 4W1
	Canada
Phone:	+1 418 656 9254
Email:	simon.perreault@viagenie.ca
URI:	http://www.viagenie.ca
	Gonzalo Camarillo
	Ericsson
	Hirsalantie 11
	Jorvas 02420
	Finland
Email:	Gonzalo.Camarillo@ericsson.com
	Oscar Novo
	Ericsson
	Hirsalantie 11
	Jorvas 02420
	Finland
Email:	Oscar.Novo@ericsson.com