BEHAVE
Internet-Draft
Expires: April 19, 2007

D. MacDonald
CounterPath Solutions, Inc.
B. Lowekamp
SIPeerior Technologies and William & Mary

October 16, 2006

NAT Behavior Discovery Using STUN
draft-macdonald-behave-nat-behavior-discovery-00

Abstract

This specification defines a usage of the Simple Traversal Underneath Network Address Translators (NAT) (STUN) Protocol that allows the applications to discover the presence and current behaviour of NATs and firewalls between them and the STUN server.

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].

Table of Contents

1. Applicability

This STUN usage does not allow an application behind NAT to make an absolute determination of the NAT's characteristics. NAT devices do not behave consistently enough to predict future behaviour with any guarantee. Applications requiring reliable reach must establish a communication channel through a NAT using another technique such as ICE [I-D.ietf-mmusic-ice] or OUTBOUND [I-D.ietf-sip-outbound]. Where reliability is not as strong a requirment, applications can use this STUN usage to select operating modes and optimizations.

2. Introduction

The Simple Traversal Underneath Network Address Translators (NAT) (STUN) [I-D.ietf-behave-rfc3489bis] provides a mechanism to discover the reflexive transport address toward the STUN server, using the Binding Request. This specification defines a usage of STUN, called the diagnostic usage, which allows applications to probe the current behaviour of the NAT/FW devices with respect to the STUN server. To accomplish this, this usage defines some new STUN attributes for the Binding Request and Binding Response.

Many NAT/FW devices do not behave consistently and will change their behaviour under load and over time. Applications requiring high reliability must be prepared for the NAT's behaviour to become more restrictive. Specifically, it has been found NATs may transtion to the most restrictive filtering and mapping behaviour and shorten the lifetime of new and existing bindings under load. In short, applications can discover how bad things currently are, but not how bad things will get.

This usage is especially useful for application entities that wish to receive incoming connections from other endpoints. For example, a P2P application can use some of these tests to deduce it if is a reasonable supernode candidate, meaning that its NAT(s) offer Address Independent Filtering. It might periodically re-run tests and would remove itself as a supernode if its NAT/FW chain lost this characteristic.

3. Overview of Operations

In a typical configuration, a STUN client is connected to a private network and through one or more NATs to the public Internet. The client is configured with the address of a STUN server on the public Internet.

OPEN ISSUE: should we have an SRV target for this usage, or piggyback on 3489-bis SRV discovery. If we had a separate SRV we wouldn't have to rely on 420 when dns is used.

The STUN NAT Behavior Discovery usage defines new attributes on the STUN Binding Request and STUN Binding Response that allow these messages to be used to diagnose the current behavior of the NAT(s) between the client and server. If the STUN server does not support the usages defined in this document a 420(Unknown Attribute) response will be returned and the client will consider this a failed test.

This section provides a descriptive overview of the typical use of these attributes. Normative behavior is described in Sections 5, 6, and 7.

OPEN ISSUE: are the UDP/TCP discussion below correct/sufficient?

3.1. Determining NAT Mapping

A client behind a NAT wishes to determine if the NAT it is behind is currently using independent, address dependent, or port dependent mapping[I-D.ietf-behave-nat-udp]. The client performs a series of tests that make use of the OTHER-ADDRESS attribute; these tests are described in detail in Section 4. These tests send binding requests to the alternate address and port of the STUN server to determine mapping behaviour. These tests can be used for UDP, TCP, or TCP/TLS connections.

3.2. Determining NAT Filtering

A client behind a NAT wishes to determine if the NAT it is behind is currently using independent, address dependent, or port dependent filtering[I-D.ietf-behave-nat-udp]. The client performs a series of tests which make use of the OTHER-ADDRESS and CHANGE-REQUEST attributes; these tests are described in Section 4. These tests request responses from the alternate address and port of the STUN server; a precondition to these tests is that no binding be established to the alternate address and port. Because the NAT does not know that the alternate address and port belong to the same server as the primary address and port, it treats these responses the same as it would those from any other host on the Internet. Therefore, the success of these binding responses sent from the alternate address and port indicate whether the NAT is currently performing independent filtering, address depending filtering, or address and port dependent filtering. This test applies only to UDP datagrams.

3.3. Binding Lifetime Discovery

Many systems, such as VoIP, rely on being able to keep a connection open between a client and server or between peers of a P2P system. Because NAT bindings expire over time, keepalive messages must be sent across the connection to preserve it. Because keepalives impose some overhead on the network and servers, reducing the frequency of keepalives can be useful.

Binding lifetime can be discovered by performing timed tests that use XOR-RESPONSE-ADDRESS. The client uses a second port and the STUN server's alternate address to check if an existing binding which hasn't had traffic sent on it is still open after time T. This approach is described in detail in Section 4.5. This test applies only to UDP datagrams.

3.4. Diagnosing NAT Hairpinning

STUN Binding Requests allow a a client to determine whether it is behind a NAT that support hairpinning of datagrams. To perform this test, the client first sends a Binding Request to its STUN server to determine its mapped address. The client then sends a STUN Binding Request to this mapped address from a different port. If the client receives its own request, the NAT hairpins datagrams. This test applies to UDP, TCP, or TCP/TLS connections.

3.5. Determining Fragment Handling

Some NATs exhibit different behavior when forwarding fragments than when forwarding an independent datagram. In particular, some NATs do not hairpin fragments at all and some platforms discard fragments under load. To diagnose this behavior, STUN messages may be sent with the optional PADDING attribute, which simply inserts additional space into the message. By forcing the STUN message to be dividing into multiple fragments, the NAT's behavior can be observed.

All of the previous tests can be performed with PADDING if a NAT's fragment behavior is important for an application, or only those tests which are most interesting to the application can be retested. PADDING only applies to UDP datagrams.

4. Discovery Process

The NAT Behavior Discovery usage provides primitives that allow stun checks to be made to determine the current behaviour of the NAT or NATs an application is behind. These tests can only give the instantaneous behaviour of a NAT; it has been found that NATs can

change behaviour under load and over time. An application must assume that NAT behvaiour can become more restrictive at any time. The tests described here are for UDP connectivity, NAT mapping behaviour, and NAT filtering behaviour; additional tests could be designed. Definitions for NAT filtering and mapping behaviour are from [I-D.ietf-behave-nat-udp].

4.1. Checking if UDP is Blocked

The client sends a STUN Binding Request to a server. This causes the server to send the response back to the address and port that the request came from. If this test yields no response, the client knows right away that it is not capable of UDP connectivity.

4.2. Determining NAT Mapping Behavior

This will require at most three tests. In test I, the client performs the UDP connectivity test but includes the OTHER-ADDRESS attribute in the binding request, with the attribute length set to 0. The server will return its alternate address and port in OTHER-ADDRESS in the binding response. The client examines the XOR-MAPPED-ADDRESS attribute. If this address and port are the same as the local IP address and port of the socket used to send the request, the client knows that it is not NATed and the effective mapping will be Endpoint Independent.

In test II, the client sends a Binding Request to the alternate address. If the XOR-MAPPED-ADDRESS in the Binding Response is the same as test I the NAT currently has Endpoint Independent Mapping. If not, test III is performed: the client sends a Binding Request to the alternate address and port. If the XOR-MAPPED-ADDRESS matches test II, the NAT currently has Address Dependent Mapping; if it doesn't match it currently has Address and Port Dependent Mapping.

4.3. Determining NAT Filtering Behavior

This will also require at most three tests. If these tests should be performed using a port that wasn't used for mapping or other tests as packets sent during those tests may affect results. In test I, the client performs the UDP connectivity test but includes the OTHER-ADDRESS attribute in the binding request, the attribute length set to 0. The server will return its alternate address and port in OTHER-ADDRESS in the binding response.

In test II, the client sends a binding request to the primary address of the server with the CHANGE-REQUEST attribute set to change-port and change-IP. This will cause the server to send its response from its alternate IP address and alternate port. If the client receives

a response the current behaviour of the NAT is Address Independent Filtering.

If no response is received, test III must be performed to distinguish between Address Dependent Filtering and Address and Port Dependent Filtering. In test III, the client sends a binding request to the original server address with CHANGE-REQUEST set to change-port. If the client receives a response the current behaviour is Address Dependent Filtering; if no response is received the current behaviour is Address and Port Dependent Filtering.

4.4. Combining and Ordering Tests

Clients may wish to combine and paralellize these tests to reduce the number of packets sent and speed the discovery process. Test I of the filtering and mapping tests also checks if UDP is blocked. However, if the STUN server does not support the discovery usage a 420(Unknown Parameters) response will be received and the client will have to repeat the test to check if UDP is blocked. An application may not need as much detail as these sample tests provide. For example, a P2P application checking whether it is a supernode candidate may only wish to check filtering behaviour and may only wish to check for endpoint independent filtering. Mapping behaviour would not matter as ICE [I-D.ietf-mmusic-ice] would establish connectivity if an incoming packet can be received from the peer. This could be accomplished in only two tests: test I and test II of filtering discovery.

Care must be taken when parallelizing tests, as some NAT devices have an upper limit on how quickly bindings will be allocated.

4.5. Binding Lifetime Discovery

STUN can also be used to probe the lifetimes of the bindings created by the NAT. For many NAT devices, an abosulte refresh interval cannot be determined; bindings might be closed quicker under heavy load or might not behave as the tests suggest. For this reason applications that require reliable bindings should send keep-alives as frequently as required by:

OPEN ISSUE: ice specifies 15 seconds, outbound specifes 24-29..is there consensus here? Or can we say "the appropriate keep-alive interval" Some applications may wish to trade reliability for bandwidth savings. Applications can use this test to determine the best estimate of the NAT binding lifetime.

To determine the binding lifetime, the client first sends a Binding Request to the server from a particular socket, X. This creates a

binding in the NAT. The response from the server contains a MAPPED-ADDRESS attribute, providing the public address and port on the NAT. Call this Pa and Pp, respectively. The client then starts a timer with a value of T seconds. When this timer fires, the client sends another Binding Request to the server, using the same destination address and port, but from a different socket, Y. This request contains a RESPONSE-ADDRESS address attribute, set to (Pa,Pp). This will create a new binding on the NAT, and cause the STUN server to send a Binding Response that would match the old binding, if it still exists. If the client receives the Binding Response on socket X, it knows that the binding has not expired. If the client receives the Binding Response on socket Y (which is possible if the old binding expired, and the NAT allocated the same public address and port to the new binding), or receives no response at all, it knows that the binding has expired.

Because some NATs only refresh binding when outbound traffic is sent, the client must resend a binding request on the original port before beginning a second test with a different value of T. The client can find the value of the binding lifetime by doing a binary search through T, arriving eventually at the value where the response is not received for any timer greater than T, but is received for any timer less than T.

This discovery process takes quite a bit of time and is something that will typically be run in the background on a device once it boots.

It is possible that the client can get inconsistent results each time this process is run. For example, if the NAT should reboot, or be reset for some reason, the process may discover a lifetime than is shorter than the actual one. For this reason, implementations are encouraged to run the test numerous times, and be prepared to get inconsistent results.

Like the other diagnostics, this test is inherently unstable. In particular, an overloaded NAT might reduce binding lifetime to shed load. A client might find this diagnostic useful at startup, for example setting the initial keepalive interval on its connection to the server to 10 seconds while beginning this check. After determining the current lifetime, the keepalive interval used by the connection to the server can be set to this appropriate value. Subsequent checks of the binding lifetime can then be performed using the keepalives in the server connection. The STUN Keepalive Usage [I-D.ietf-behave-rfc3489bis]provides a response that confirms the connection is open and allows the client to check that its mapped address has not changed. As that provides both the keepalive action and diagnostic that it is working, it should be preferred over any

attempt to characterize the connection by a secondary technique.

5. Client Behavior

Discovery of the STUN server is outside the scope of this draft. Procedures for discovering a STUN server and for obtaining a shared secret, if required, are discussed in STUN [I-D.ietf-behave-rfc3489bis]. Unless otherwise specified here, all procedures for preparing, sending, and processing messages as described in the STUN Binding Usage are followed.

If the client is interested in performing a Binding Lifetime Discovery test or other test requiring use of the RESPONSE-ADDRESS attribute, it MUST obtain a shared secret prior to beginning the test, and that shared secret must be used for all transactions during the test. If the client receives a 430 (Stale Credentials) Response to a Request containing a RESPONSE-ADDRESS, then it must acquire a new short-term credential and begin the test again.

OPEN ISSUE: The only attack made possible by the RESPONSE-ADDRESS that wasn't possibly before is using the STUN server to reflect a dos against a cient that is behind a Address-Dependent Filtering NAT and has openend permission to the STUN server. A direct DOS against the same endpoint would prevent this attack. As REFLECTED-FROM already provides traceability the shared secret requirement could be replaced with rate-limiting behaviour on the server. This would allow the deployment of STUN servers w/out shared secrets.

The client indicates it is interested in the NAT Behavior Discovery usage by including one of the attributes defined in this draft. If the client wishes to receive the alternate server address and port in the Binding Response then the OTHER-ADDRESS attribute with a length of zero is included in the Binding Request.

A client MUST be prepared for receiving a 420 (Unknown Attribute) error to its requests. This response indicates that the server does not implement that particular attribute. An unsupported attribute prevents some tests from being run, but others may work. For example, only RESPONSE-ADDRESS is necessary for binding lifetime discovery, whereas mapping and filtering discovery do not require it, but do require OTHER-ADDRESS and CHANGE-REQUEST.

All attributes listed in this specification are optional in the NAT Behavior Discovery usage. The attributes provided offer a variety of possibilities to discover the current behavior of NAT, and while suggested tests are described above, specific test algorithms are not mandated by this specification.

6. Server Behavior

Unless otherwise specified here, all procedures for preparing, sending, and processing messages as described for the STUN Binding Usage of STUN [I-D.ietf-behave-rfc3489bis] are followed.

A server implementing the NAT Behavior Discovery usage SHOULD be configured with two separate IP addresses on the public Internet. On startup, the server SHOULD allocate two UDP ports, such that it can send and receive datagrams using the same ports on each IP address (normally a wildcard binding accomplishes this). If a server cannot allocate the same ports on two different IP address, then it MUST response with a 420 (Unknown Attribute) to any Request containing an OTHER-ADDRESS or CHANGE-REQUEST attribute.

The server MUST NOT include any attribute defined in this document in the Response to a Request that does not contain at least one attribute defined in this document.

6.1. Preparing the Response

After performing all authentication and verification steps, and after adding the MAPPED-ADDRESS or XOR-MAPPED-ADDRESS, the server begins processing specific to this Usage if the Request contains any request attributes defined in this document: RESPONSE-ADDRESS, CHANGE-REQUEST, OTHER-ADDRESS, or PADDING.

If the Request contains an OTHER-ADDRESS or CHANGE-REQUEST attribute and the server does not have an alternate address and port as described above, the server MUST generate an error response of type 420.

If the Request contains a RESPONSE-ADDRESS attribute, but the message does not contain a MESSAGE-INTEGRITY attribute and a USERNAME, the server MUST generate an error response of type 401. If RESPONSE-ADDRESS is included, then the server must verify that it has previously received a binding request from the same address as is specified in RESPONSE-ADDRESS. If it has not, or if sufficient time has passed that it no longer has a record of having received such a request due to limited state, it MUST respond with an error response of type 430.

OPEN-ISSUE: is this too strong? There is no amplification attack here. Maybe either require a shared secret, but don't track addresses, or track addresses that have done binding requests but not relate them to shared secret or just put in a rate limit and leave it at that?

The source address and port of the Binding Response depend on the value of the CHANGE-REQUEST attribute and on the address and port the Binding Request was received on, and are summarized in Table 1.

Let Da represent the destination IP address of the Binding Request (which will be either A1 or A2), and Dp represent the destination port of the Binding Request (which will be either P1 or P2). Let Ca represent the other address, so that if Da is A1, Ca is A2. If Da is A2, Ca is A1. Similarly, let Cp represent the other port, so that if Dp is P1, Cp is P2. If Dp is P2, Cp is P1. If the "change port" flag was set in CHANGE-REQUEST attribute of the Binding Request, and the "change IP" flag was not set, the source IP address of the Binding Response MUST be Da and the source port of the Binding Response MUST be Cp. If the "change IP" flag was set in the Binding Request, and the "change port" flag was not set, the source IP address of the Binding Response MUST be Ca and the source port of the Binding Response MUST be Dp. When both flags are set, the source IP address of the Binding Response MUST be Ca and the source port of the Binding Response MUST be Cp. If neither flag is set, or if the CHANGE-REQUEST attribute is absent entirely, the source IP address of the Binding Response MUST be Da and the source port of the Binding Response MUST be Dp.

Flags	Source Address	Source Port	OTHER-ADDRESS
none	Da	Dp	Ca:Cp
Change IP	Ca	Dp	Ca:Cp
Change port	Da	Cp	Ca:Cp
Change IP and Change port	Ca	Cp	Ca:Cp

Table 1: Impact of Flags on Packet Source and OTHER-ADDRESS

The server MUST add a SOURCE-ADDRESS attribute to the Binding Response, containing the source address and port used to send the Binding Response.

If it supports an alternate address and port and the Binding request contained a zero-length OTHER-ADDRESS attribute the server MUST add an OTHER-ADDRESS attribute to the Binding Response. This contains the source IP address and port that would be used if the client had set the "change IP" and "change port" flags in the Binding Request. As summarized in Table 1, these are Ca and Cp, respectively, regardless of the value of the CHANGE-REQUEST flags.

Next the server inspects the Request for a RESPONSE-ADDRESS attribute. If the RESPONSE-ADDRESS attribute is included, then the server inspects the magic cookie field to determine the version of the protocol. If it matches the magic cookie from STUN [I-D.ietf-behave-rfc3489bis] then it includes an XOR-REFLECTED-FROM attribute with the source address the Request was received from. Otherwise, it includes a REFLECTED-FROM attribute.

If the Request contained a PADDING attribute, then the server SHOULD insert a PADDING attribute of the same length into its response, but no longer than 64K.

Following that, the server completes the remainder of the processing from STUN [I-D.ietf-behave-rfc3489bis], including adding the SERVER, MESSAGE-INTEGRITY, and FINGERPRINT attributes as appropriate. When it sends the response datagram, it is sent from the source address as determined above and to the destination address determined from the RESPONSE-ADDRESS, or to the source address of the Request if not specified.

7. New Attributes

This document defines several STUN attributes that are required for NAT Behavior Discovery. These attributes are all used only with Binding Requests and Binding Responses.

OPEN ISSUE: are these considered "new" attributes or not?

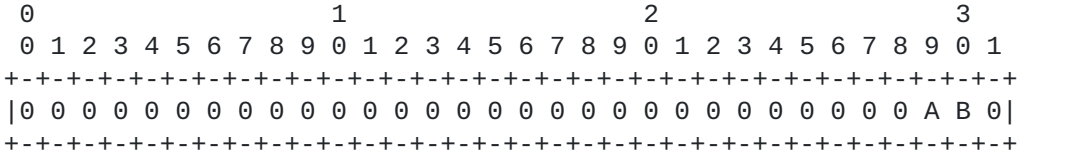
0x0x0002: RESPONSE-ADDRESS
0x0003: CHANGE-REQUEST
0x0004: SOURCE-ADDRESS
0x0005: OTHER-ADDRESS
0x000b: REFLECTED-FROM
0x0023: XOR-REFLECTED-FROM

7.1. RESPONSE-ADDRESS

The RESPONSE-ADDRESS attribute contains an IP address and port. The RESPONSE-ADDRESS attribute can be present in the Binding Request and indicates where the Binding Response is to be sent. When not present, the server sends the Binding Response to the source IP address and port of the Binding Request. The server MUST NOT process a request containing a RESPONSE-ADDRESS that does not contain MESSAGE-INTEGRITY. The RESPONSE-ADDRESS attribute is optional in the Binding Request.

7.2. CHANGE-REQUEST

The CHANGE-REQUEST attribute contains two flags to control the IP address and port the server uses to send the response. These flags are called the "change IP" and "change port" flags. The CHANGE-REQUEST attribute is allowed only in the Binding Request. The "change IP" and "change port" flags are useful for determining the current filtering behavior of a NAT. They instruct the server to send the Binding Responses from the alternate source IP address and/or alternate port. The CHANGE-REQUEST attribute is optional in the Binding Request. The attribute is 32 bits long, although only two bits (A and B) are used:



The meanings of the flags are:

A: This is the "change IP" flag. If true, it requests the server to send the Binding Response with a different IP address than the one the Binding Request was received on.

B: This is the "change port" flag. If true, it requests the server to send the Binding Response with a different port than the one the Binding Request was received on.

7.3. SOURCE-ADDRESS

The SOURCE-ADDRESS attribute indicates the source IP address and port the response was sent from. It is useful for detecting twice NAT configurations. It is only present in Binding Responses. SOURCE-ADDRESS MUST NOT be inserted into a Binding Response unless the Binding Request contained an attribute defined in this specification.

7.4. OTHER-ADDRESS

The OTHER-ADDRESS attribute is used in both Binding Requests and Binding Responses. In a Binding Request it indicates that the client wishes to use the NAT Behavior Discovery usage, and the server should include attributes involved in this usage in its response. When used by the server, It informs the client of the source IP address and port that would be used if the client requested the "change IP" and "change port" behavior. OTHER-ADDRESS MUST NOT be inserted into a Binding Response unless the Binding Request contained an attribute defined in this specification.

7.5. REFLECTED-FROM

The REFLECTED-FROM attribute is present only in Binding Responses when the Binding Request contained a RESPONSE-ADDRESS attribute. The attribute contains the identity (in terms of IP address) of the source where the request came from. Its purpose is to provide traceability, so that a STUN server cannot be used as a reflector for denial-of-service attacks. Its syntax is identical to the MAPPED-ADDRESS attribute.

REFLECTED-FROM is included for compatibility with legacy applications, only. New implementations should use XOR-REFLECTED-FROM.

7.6. XOR-REFLECTED-FROM

The XOR-REFLECTED-FROM attribute is used in place of the REFLECTED-FROM attribute. It provides the same information, but because the NAT's public address is obfuscated through the XOR function, It can pass through a NAT that would otherwise attempt to translate it to the private network address. XOR-REFLECTED-FROM has identical syntax to XOR-MAPPED-ADDRESS.

7.7. PADDING

The PADDING attribute allows for the entire message to be padded to force the STUN message to be divided into UDP fragments. PADDING consists entirely of a freeform string, the value of which does not matter. When PADDING is used, it SHOULD be 1500 bytes long, unless a more appropriate length is known based on the MTU of the path. PADDING can be used in either Binding Requests or Binding Responses. If PADDING is present in the Binding Request and the server supports it, PADDING MUST be present in the Binding Response. The server SHOULD use the same length PADDING as was used in the Binding Request, but it MAY use another length if it knows what length is required to cause fragmentation along the return path, or it MAY use a length of zero to indicate that the field is understood but the server is ignoring it.

PADDING MUST be no longer than 64K and SHOULD be an even multiple of four bytes.

8. IAB Considerations

The IAB has studied the problem of ``Unilateral Self Address Fixing'', which is the general process by which a client attempts to determine its address in another realm on the other side of a NAT

through a collaborative protocol reflection mechanism RFC 3424 [RFC3424]. The STUN NAT Behavior Discovery usage is an example of a protocol that performs this type of function. The IAB has mandated that any protocols developed for this purpose document a specific set of considerations. This section meets those requirements.

8.1. Problem Definition

>From RFC 3424 [RFC3424], any UNSAF proposal must provide:

> Precise definition of a specific, limited-scope problem that is to be solved with the UNSAF proposal. A short term fix should not be generalized to solve other problems; this is why "short term fixes usually aren't".

The specific problem being solved by the STUN NAT Behavior Discovery usage is for a client, which may be located behind a NAT of any type, to determine the characteristics of that NAT in order to either diagnose the cause of problems experienced by that or other applications or for an application to modify its behavior based on the current behavior of the NAT.

8.2. Exit Strategy

From [RFC3424], any UNSAF proposal must provide:

> Description of an exit strategy/transition plan. The better short term fixes are the ones that will naturally see less and less use as the appropriate technology is deployed.

The STUN NAT Behavior Discovery usage does not itself provide an exit strategy. Instead, that is provided by other protocols. Specifically, the Interactive Connectivity Establishment (ICE) [I-D.ietf-mmusic-ice] mechanism allows two cooperating clients to interactively determine the best addresses to use when communicating, regardless of the type of NAT involved. BEHAVE is currently considering proposals for protocols that allow clients to determine the location of and control the behavior of NATs through direct interaction with the NAT. STUN NAT Behavior Discovery is no longer needed once NATs that can be communicated with directly are in use. Finally, as NATs phase out and as IPv6 is deployed, STUN NAT Behavior Discovery will no longer be of any interest.

8.3. Brittleness Introduced by STUN NAT Behavior Discovery

From [RFC3424], any UNSAF proposal must provide:

> Discussion of specific issues that may render systems more "brittle". For example, approaches that involve using data at multiple network layers create more dependencies, increase debugging challenges, and make it harder to transition.

The STUN NAT Behavior Discovery usage allows a client to determine the current behavior of a NAT. This information can be quite useful to a developer or network administrator outside of an application, and as such can be used to diagnose the brittleness induced in another application. When used within an application itself, STUN NAT Behavior Discovery allows the application to adjust its behavior according to the current behavior of the NAT. While this can be helpful in improving the performance of an application, an improperly written application could use information from this usage and assume that the NAT will always behave in the same manner, and thus failing to work properly when the NAT changes its behavior. Regardless of whether an application makes use of NAT Behavior Discovery or not, if it does not use techniques such as ICE [I-D.ietf-mmusic-ice] or OUTBOUND [I-D.ietf-sip-outbound] it exposes itself to the inherent instability of NAT.

8.4. Requirements for a Long Term Solution

>From [RFC3424]}, any UNSAF proposal must provide:

> Identify requirements for longer term, sound technical solutions -- contribute to the process of finding the right longer term solution.

Our experience with STUN NAT Behavior Discovery continues to validate our belief in the requirements outlined in Section 14.4 of STUN [I-D.ietf-behave-rfc3489bis].

8.5. Issues with Existing NAPT Boxes

>From [RFC3424], any UNSAF proposal must provide:

> Discussion of the impact of the noted practical issues with existing, deployed NA[P]Ts and experience reports.

A number of NAT boxes are now being deployed into the market which try and provide "generic" ALG functionality. These generic ALGs hunt for IP addresses, either in text or binary form within a packet, and rewrite them if they match a binding. This usage avoids that problem by using the XOR-REFLECTED-ADDRESS attribute instead of the REFLECTED-ADDRESS.

This usage provides a set of generic attributes that can be assembled

to test many types of NAT behavior. While tests for the most commonly known NAT box behaviors are described, the BEHAVE mailing list regularly has descriptions of new behaviors, some of which may not be readily detected using the tests described herein. However, the techniques described in this usage can be assembled in different combinations to test NAT behaviors not now known or envisioned.

9. IANA Considerations

This specification defines several new STUN attributes. This section directs IANA to add these new protocol elements to the IANA registry of STUN protocol elements. The code for OTHER-ADDRESS renames this code from CHANGED-ADDRESS to OTHER-ADDRESS for clarity, the semantics remain the same.

OPEN ISSUE: does IANA consider these new attributes or are there in forever from original 3489?

0x0002: RESPONSE-ADDRESS
0x0003: CHANGE-REQUEST
0x0004: SOURCE-ADDRESS
0x0005: OTHER-ADDRESS
0x000b: REFLECTED-FROM
0x0023: XOR-REFLECTED-FROM
0x8024: PADDING

10. Security Considerations

This usage inherits the security considerations of STUN [I-D.ietf-behave-rfc3489bis]. This usage adds several new attributes; security considerations for those are detailed here.

OTHER-ADDRESS does not permit any new attacks; it provides another place where an attacker can impersonate a STUN server but it is not an interesting attack. An attacker positioned where it can compromise the Binding Request can completely hide the STUN server from the client.

RESPONSE-ADDRESS allows a STUN server to be used as a reflector for denial-of-service attacks. The XOR-REFLECTED-FROM mitigates this by providing the identity (in terms of IP address) of the source where the request came from. Its purpose is to provide traceability, so that a STUN server cannot be used as an anonymous reflector for denial-of-service attacks. Authenticating the RESPONSE-ADDRESS using shared secrets alleviates this threat.

The only attack possible with the PADDING attribue is to have a large padding length which could cause a server to allocate a large amount of memory. As servers will ignore any padding length greater than 64k so the scope of this attack is limited. In general, servers should not allocate more memory than the size of the received datagram. This attack would only affect non-compliant implementations.

11. Acknowledgements

The authors would like to thank the authors of the original STUN specification [RFC3489] from which many of the ideas, attributes, and description in this document originated.

12. References

12.1. Normative References

[I-D.ietf-behave-nat-udp]
Audet, F. and C. Jennings, "NAT Behavioral Requirements for Unicast UDP", draft-ietf-behave-nat-udp-08 (work in progress), October 2006.

[I-D.ietf-behave-rfc3489bis]
Rosenberg, J., "Simple Traversal Underneath Network Address Translators (NAT) (STUN)", draft-ietf-behave-rfc3489bis-04 (work in progress), July 2006.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3424] Daigle, L. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.

12.2. Informative References

[I-D.ietf-mmusic-ice]
Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-mmusic-ice-11 (work in progress), October 2006.

[I-D.ietf-sip-outbound]
Jennings, C. and R. Mahy, "Managing Client Initiated

Connections in the Session Initiation Protocol (SIP)", draft-ietf-sip-outbound-04 (work in progress), June 2006.

[RFC3489] Rosenberg, J., Weinberger, J., Huitema, C., and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)", RFC 3489, March 2003.

Authors' Addresses

Derek C. MacDonald
CounterPath Solutions, Inc.
Suite 300, One Bentall Centre, 505 Burrard St
Vancouver, BC V7X1M3
Canada

Phone: +1-604-320-3344
Email: derek@counterpath.com

Bruce B. Lowekamp
SIPeerior Technologies and William & Mary
3000 Easter Circle
Williamsburg, Virginia 23188
USA

Phone: unlisted
Email: lowekamp@sipeerior.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.