Abstract

This specification defines an experimental usage of the Simple Traversal Underneath Network Address Translators (NAT) (STUN) Protocol that discovers the presence and current behaviour of NATs and firewalls between the STUN client 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 ................................................. 4  
2. Introduction ................................................. 5  
   2.1. Diagnostic Use ........................................... 5  
   2.2. Example Use with P2P Overlays ......................... 6  
   2.3. Experimental Success ................................... 6  
3. Overview of Operations ..................................... 7  
   3.1. Determining NAT Mapping ................................. 8  
   3.2. Determining NAT Filtering ............................... 8  
   3.3. Binding Lifetime Discovery ............................. 8  
   3.4. Diagnosing NAT Hairpinning ............................ 9  
   3.5. Determining Fragment Handling ......................... 9  
   3.6. Detecting Generic ALGs ................................. 9  
4. Discovery Process ........................................... 9  
   4.1. Checking if UDP is Blocked ............................. 10  
   4.2. Determining NAT Mapping Behavior ..................... 10  
   4.3. Determining NAT Filtering Behavior ................... 11  
   4.4. Combining and Ordering Tests .......................... 11  
   4.5. Binding Lifetime Discovery ............................ 12  
5. Client Behavior ............................................. 14  
   5.1. Discovery ................................................ 15  
   5.2. Security ................................................. 15  
6. Server Behavior ............................................. 15  
   6.1. Preparing the Response ................................. 16  
7. New Attributes ............................................... 18  
   7.1. Representing Transport Addresses ...................... 19  
   7.2. CHANGE-REQUEST ......................................... 19  
   7.3. RESPONSE-ORIGIN ......................................... 20  
   7.4. OTHER-ADDRESS .......................................... 20  
   7.5. XOR-REFLECTED-FROM ..................................... 20  
   7.6. XOR-RESPONSE-TARGET .................................... 20  
   7.7. PADDING ................................................ 21  
   7.8. CACHE-TIMEOUT ........................................... 21  
8. New Response Codes ......................................... 21  
   8.1. 481 Connection does not exist ........................ 21  
   8.2. 503 Service Unavailable ................................ 22  
9. IAB Considerations ......................................... 22  
   9.1. Problem Definition ...................................... 22  
   9.2. Exit Strategy ............................................ 22  
   9.3. Brittness Introduced by STUN NAT Behavior Discovery 23  
   9.4. Requirements for a Long Term Solution ................ 23  
   9.5. Issues with Existing NAPT Boxes ...................... 24  
10. IANA Considerations ....................................... 24  
   10.1. STUN Attribute Registry ............................... 24  
   10.2. STUN Error Code Registry ............................. 24  
11. Security Considerations ................................... 25  
12. Acknowledgements .......................................... 26
13. References .................................................. 26
13.1. Normative References ..................................... 26
13.2. Informative References ................................... 26
Appendix A. Change Log ......................................... 27
A.1. from draft-macdonald-behave-nat-behavior-diagnostics-00 . 27
A.2. from draft-ietf-behave-nat-behavior-discovery-00 .......... 27
A.3. from draft-ietf-behave-nat-behavior-discovery-01 .......... 27
A.4. from draft-ietf-behave-nat-behavior-discovery-02 .......... 28
A.5. from draft-ietf-behave-nat-behavior-discovery-03 .......... 28
A.6. from draft-ietf-behave-nat-behavior-discovery-04 .......... 28
Authors’ Addresses ............................................. 29
Intellectual Property and Copyright Statements ............... 30
1. Applicability

This experimental STUN usage does not allow an application behind a 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. This STUN usage provides information about observable transient behavior; it only truly determines a NAT’s behavior with regard to the STUN server used and the particular ports used at the instant the test is run. Applications requiring reliable reach between two particular endpoints must establish a communication channel through a NAT using another technique. IETF has proposed standards including ICE [I-D.ietf-mmusic-ice] and OUTBOUND [I-D.ietf-sip-outbound] for establishing communication channels when a publicly accessible rendezvous service is available.

This usage provides techniques which are powerful diagnostic tools in the hands of a network administrator or system programmer trying to determine the causes of network failure; particularly when behavior varies by load, destination, or other factors that may be related to NAT behavior.

This draft also proposes experimental applications of NAT Behavior Discovery STUN for real-time selection of parameters for protocols in situations where a publicly accessible rendezvous service is not available. One such application is role selection in P2P networks based on statistical experience with establishing connections and diagnosing NAT behavior with a variety of peers. The experimental question is whether such a test is useful. If a node trying to join an overlay as a full peer when its NAT prevents sufficient connectivity and then withdrawing is expensive or leads to unreliable or poorly performing operation, then even if the behavior discovery check is only "correct" 75% of the time, its relative cheapness may make it very useful for optimizing the behavior of the overlay network. Section 2.2 describes this experimental application in more detail and discusses how to evaluate its success or failure.

The applications of this STUN usage are very different than the original use of RFC3489 [RFC3489], which was intended for static determination of device behavior. The NAT Behavior Discovery STUN usage makes an explicit statement that it is not, and cannot be, correct 100% of the time, but is still very useful. It is submitted to the Internet community as an experimental protocol that, when applied with appropriate statistical underpinnings and application behavior that is ultimately based on experienced connectivity patterns, can lead to more stability and increased performance than is available without the knowledge it provides.
2. Introduction

The Session Traversal Utilities for NAT (STUN) [RFC5389] provides a mechanism to discover the reflexive transport address toward the STUN server, using the Binding Request. This specification defines the NAT Behavior Discovery STUN usage, which allows a STUN client to probe the current behaviour of the NAT/FW devices between the client and the STUN server. This usage defines 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 that under load NATs may transition to the most restrictive filtering and mapping behaviour and shorten the lifetime of new and existing bindings. In short, applications can discover how bad things currently are, but not how bad things will get.

Despite this limitation, instantaneous observations are often quite useful in troubleshooting network problems, and repeated tests over time, or in known load situations, may be used to characterize a NAT’s behavior. In particular, in the hands of a person knowledgeable about the needs of an application and the nodes an application needs to communicate with, it can be a powerful tool.

2.1. Diagnostic Use

Applications that work well in the lab, but fail in a deployment, are notoriously common within distributed systems. There are few systems developers who have not had the experience of searching to determine the difference in the environments for insight as to what real-network behavior was missed in the testing lab. The behavior discovery usage offers a powerful tool that can be used to check NAT and firewall behavior as the application is running.

As they are being used to detect instantaneous behavior for analysis by an experienced developer or administrator, there are relatively few concerns about this application of the NAT Behavior Discovery STUN usage. However, the user should be aware that

- adding new traffic to new destinations (STUN servers) has the potential to itself change the behavior of a NAT and

- the user must be careful to select a STUN server that is appropriately located, ideally collocated (or even integrated) with the communication partners of the application in question, for the results to be applicable to the network conditions.
2.2. Example Use with P2P Overlays

An application could use Behavior Discovery in a Peer-to-Peer (P2P) protocol to determine if a particular endpoint is a reasonable candidate to participate as a peer or supernode (defined here as a peer in the overlay that offers services, including message routing, to other members or clients of the overlay network). This P2P network application is willing to select supernodes that might be located behind NATs to avoid the cost of dedicated servers. A supernode candidate requires that its NAT(s) offer(s) Endpoint-Independent Filtering. It might periodically re-run tests and would remove itself as a supernode if its NAT/FW chain lost this characteristic. These tests could be run with other supernodes acting as STUN servers as well as with dedicated STUN servers. As many P2P algorithms tolerate non-transitive connectivity between a portion of their peers, guaranteed pair-wise reliable reach might be sacrificed in order to distribute the P2P overlay’s load across peers that can be directly contacted by the majority of users.

Use of Behavior Discovery for such an application requires:

- Specification of protocols capable of offering reliable end-user performance using unreliable links between peers.
- The application is deployed behind NATs that provide Endpoint-Independent Filtering and that remain in this mode for an amount of time sufficient for the application to identify their behavior, distribute this information to the rest of the overlay, and provide useful work for the application.

This draft is experimental as deployed applications implementing open protocols have yet to be deployed in such environments to demonstrate that these two requirements have been met. However, apocryphal evidence suggests that NATs targeted at households and small businesses have stable behaviour, especially when there are few clients behind them. Numerous P2P applications have been deployed that appear to have these properties, although their protocols have not yet been subjected to rigorous evaluation by standards bodies.

2.3. Experimental Success

The criteria for an application to successfully demonstrate use of the NAT Behavior Discovery STUN usage would include:
An implementation that relies on this usage to determine its runtime behavior, most likely using it to determine an initial choice of options that are then adjusted based on experience with its network connections.

The implementation must either demonstrate its applicability in environments where it is realistic to expect a provider to deploy dedicated STUN servers with multiple IP addresses, or it must demonstrate duplicating the behavior of such a dedicated STUN server with two nodes that share the role of providing the address-changing operations required by this usage.

Experimental evidence that the application of this usage results in improved behavior of the application in real-world conditions. The exact metrics for this improvement may vary, some possibilities include: faster convergence to the proper parameters, less work to set up initial connections, fewer reconfigurations required after startup, etc.

A protocol specification that defines how the implementation applies this usage.

The P2P scenario described above is a likely experimental test case for this usage, but others applications are possible as well.

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. The Behavior Discovery usage makes use of SRV records so that a server may use a different transport address for this usage than for other usages. This usage does not provide backward compatibility with RFC3489 [RFC3489] for either clients or servers. Implementors of clients that wish to be compliant with RFC3489 servers should see that specification. Implementors of servers SHOULD NOT include support for RFC3489 clients as the original uses of that protocol have been deprecated.

Because the STUN forbids a server from creating a new TCP or TCP/TLS connection to the client, many tests apply only to UDP. The applicability of the various tests is indicated below.

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.
This section provides a descriptive overview of the typical use of these attributes. Normative behavior is described in Sections 5, 6, 7, and 8.

3.1. Determining NAT Mapping

A client behind a NAT wishes to determine if the NAT it is behind is currently using endpoint-independent, address-dependent, or address and port-dependent mapping [RFC4787]. 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 endpoint-independent, address-dependent, or address and port-dependent filtering [RFC4787]. The client performs a series of tests that 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. Ensuring this precondition is difficult, therefore the client must either use a random port or ensure that no traffic has previously occurred on the selected 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 the binding responses sent from the alternate address and port indicate whether the NAT is currently performing endpoint-independent filtering, address-dependent 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-TARGET. XOR-RESPONSE-TARGET allows the client to allocate two ports and request that responses to queries sent from
one port be delivered to the other. The client uses its second port and the STUN server’s alternate address to check if an existing binding that 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 client to determine whether it is behind a NAT that supports hairpinning of connections. 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 connections. 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 a single-frame 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 PADDING attribute, which simply inserts additional space into the message. By forcing the STUN message to be divided 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. PADDING can not be used with XOR-RESPONSE-TARGET.

3.6. Detecting Generic ALGs

A number of NAT boxes are now being deployed into the market which try to 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 behavior can be detected because the STUN server returns both the MAPPED-ADDRESS and XOR-MAPPED-ADDRESS in the same response. If the result in the two does not match, there is a NAT with a generic ALG in the path. This test applies to UDP and TCP, but not TLS over TCP connections.

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 behaviour 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 using this usage’s mechanisms. Definitions for NAT filtering and mapping behaviour are from [RFC4787].

Because mapping behavior can vary on a port-by-port basis, an application should perform its tests using the source port intended for use by the application whenever possible. If it intends to use multiple source ports, it should repeat these tests for each source port. Such tests should be performed sequentially to reduce load on the NAT.

This section provides a descriptive overview of how the primitives provided by the STUN attributes in this specification may be used to perform behavior tests. Normative specifications for the attributes is defined in later sections.

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. This test requires only STUN [RFC5389] functionality.

As with all tests, this test is only deterministic for connectivity with the particular STUN server and source port used. A client should be configured with multiple STUN servers for redundancy and to protect against the configuration specifying an incorrect address for the STUN server.

4.2. Determining NAT Mapping Behavior

This will require at most three tests. In test I, the client performs the UDP connectivity test. The server will return its alternate address and port in OTHER-ADDRESS in the binding response. If OTHER-ADDRESS is not returned, the server does not support this usage and this test cannot be run. 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, but primary port. 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. Because prior traffic may affect results, the client needs to either use a source port that is known to have been idle for at least 15 minutes prior to running the test or use a random source port for these tests. A client performing these tests immediately on startup will have no knowledge of prior use of these ports and, therefore, needs to use a random source port to ensure correct results.

Note that a client wishing to perform these tests using the source port it will use for its application traffic, as recommended above, must either use a random source port for its application traffic or ensure that the intended source port is not used for at least 15 minutes prior to performing the test.

In test I, the client performs the UDP connectivity test. The server will return its alternate address and port in OTHER-ADDRESS in the binding response. If OTHER-ADDRESS is not returned, the server does not support this usage and this test cannot be run.

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 Endpoint-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 parallelize these tests to reduce the number of packets sent and speed the discovery process. For example,
test I of the filtering and mapping tests also checks if UDP is blocked. Furthermore, an application or user may not need as much detail as these sample tests provide. For example, establishing connectivity between nodes becomes significantly more difficult if a NAT has any behavior other than endpoint-independent mapping, which requires only test I and II of Section 4.2. An application determining its NAT does not always provide independent mapping might notify the user if no relay is configured, whereas an application behind a NAT that provides endpoint-independent mapping might not notify the user until a subsequent connection actually fails or might provide a less urgent notification that no relay is configured. Such a test does not alleviate the need for ICE [I-D.ietf-mmusic-ice], but it does provide some information regarding whether ICE is likely to be successful establishing non-relayed connections.

Care must be taken when parallelizing tests, as some NAT devices have an upper limit on how quickly bindings will be allocated. Section 5 restricts the rate at which clients may begin new STUN transactions.

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 absolute 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 must send keep-alives as frequently as required by all NAT devices that will be encountered. Suggested refresh intervals are outside the scope of this document. ICE [I-D.ietf-mmusic-ice] and OUTBOUND [I-D.ietf-sip-outbound] have suggested refresh intervals.

Determining the binding lifetime relies on two separate source ports being used to send STUN Binding Requests to the STUN server. The general approach is that the client uses a source port X to send a single Binding Request. After a period of time during which source port X is not used, the client uses a second source port Y to send a Binding Request to the STUN server that indicates the response should be sent to the binding established to port X. If the binding for port X has timed out, that response will not be received. By varying the time between the original Binding Request sent from X and the subsequent request sent from Y, the client can determine the binding lifetime.

To determine the binding lifetime, the client first sends a Binding Request to the server from a particular source port, 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 source port, Y. This request contains an XOR-RESPONSE-TARGET address attribute, set to (Pa,Pp), to request the response be delivered 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, (Pa,Pp), if it still exists. If the client receives the Binding Response on port X, it knows that the binding has not expired. If the client receives the Binding Response on port 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 bindings when outbound traffic is sent, the client must resend a binding request from the original source 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. Note also that the binding refresh behavior (outbound only or all traffic) can be determined by sending multiple Binding Requests from port Y without refreshes from the original source port X.

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. Binding lifetime may also be dependent on the traffic load on the NAT. 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-sip-outbound] 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

Unless otherwise specified here, all procedures for preparing, sending, and processing messages as described in the STUN Binding Usage [RFC5389] are followed.

If a client intends to utilize an XOR-RESPONSE-TARGET attribute in future transactions, as described in Section 4.5, then it MUST include a CACHE-TIMEOUT attribute in the Request with the value set greater than the longest time duration it intends to test. The server will also include this attribute in its Response, modified with its estimate of how long it will be able to cache this connection. Because the returned value is only an estimate, the client must be prepared for the value to be wrong, and therefore to receive a 481 response to its subsequent Requests with XOR-RESPONSE-TARGET.

Support for XOR-RESPONSE-TARGET is optional due to the state cost on the server. Therefore, a client MUST be prepared to receive a 420 (Unknown Attribute) error to requests that include XOR-RESPONSE-TARGET or CACHE-TIMEOUT. Support for OTHER-ADDRESS and CHANGE-REQUEST is optional, but MUST be supported by servers advertised via SRV, as described below. This is to allow the use of PADDING and XOR-RESPONSE-TARGET in applications where servers do not have multiple IP addresses. Clients MUST be prepared to receive a 420 for requests that include CHANGE-REQUEST when OTHER-ADDRESS was not received in Binding Response messages from the server.

If an application makes use of the NAT Behavior Discovery STUN usage by multiplexing it in a flow with application traffic, a FINGERPRINT attribute SHOULD be included unless it is always possible to distinguish a STUN message from an application message based on their header.

When PADDING is used, it SHOULD be longer than the MTU of the outgoing interface.

Clients SHOULD ignore an ALTERNATE-SERVER attribute in a response unless they are using authentication with a provider of STUN servers that is aware of the topology requirements of the tests being performed.
A client SHOULD NOT generate more than ten new STUN transactions per second and should pace them such that the RTOs do not synchronize the retransmissions of each transaction.

5.1. Discovery

Unless the user or application is aware of the transport address of a STUN server supporting the NAT Behavior Discovery usage through other means, a client is configured with the domain name of the provider of the STUN servers. The domain is resolved to a transport address using SRV procedures [RFC2782]. The mechanism for configuring the client with the domain name of the STUN servers or of acquiring a specific transport address is out of scope for this document.

For the Behavior Discovery Usage the service name is "stun-behavior". The protocol can be "udp", "tcp" or "tls". Other aspects of handling failures and default ports are followed as described in STUN [RFC5389].

5.2. Security

Servers MAY require authentication before allowing a client to make use of its services. This is particularly important to requests used to perform a Binding Lifetime Discovery test or other test requiring use of the XOR-RESPONSE-TARGET attribute. The method for obtaining these credentials, should the server require them, is outside the scope of this usage. Presumably, the administrator or application relying on this usage should have its own method for obtaining credentials. If the client receives a 401 (Unauthorized) Response to a Request, then it must either acquire the appropriate credential from the application before retrying or report a permanent failure. Procedures for encoding the MESSAGE-INTEGRITY attribute for a request are described in STUN [RFC5389].

6. Server Behavior

Unless otherwise specified here, all procedures for preparing, sending, and processing messages as described for the STUN Binding Usage of STUN [RFC5389] 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 a pair of ports for each of the UDP, TCP, and TCP/TLS transport protocols, such that it can send and receive datagrams using the same ports on each IP address (normally a wildcard binding accomplishes this). TCP and TCP/TLS MUST use different ports. If a server cannot allocate the same ports on two
different IP address, then it MUST NOT include an OTHER-ADDRESS attribute in any Response and MUST respond with a 420 (Unknown Attribute) to any Request with a CHANGE-REQUEST attribute. A server with only one IP address MUST NOT be advertised using the SRV service name "stun-behavior".

6.1. Preparing the Response

After performing all authentication and verification steps the server begins processing specific to this Usage if the Request contains any request attributes defined in this document: XOR-RESPONSE-TARGET, CHANGE-REQUEST, CACHE-TIMEOUT, or PADDING. If the Request does not contain any attributes from this document, OTHER-ADDRESS and RESPONSE-ORIGIN are still included in the response.

The server MUST include both MAPPED-ADDRESS and XOR-MAPPED-ADDRESS in its Response.

If the Request contains 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 CACHE-TIMEOUT attribute, then the server SHOULD include a CACHE-TIMEOUT attribute in its response indicating the duration (in seconds) it anticipates being able to cache this binding request in anticipation of a future Request using the XOR-RESPONSE-TARGET attribute. The CACHE-TIMEOUT response value can be greater or less than the value in the request. If the server is not prepared to provide such an estimate, it SHOULD NOT include the CACHE-TIMEOUT attribute in its Response. The server SHOULD NOT provide a CACHE-TIMEOUT length longer than the amount of time it has been able to cache recent requests.

If XOR-RESPONSE-TARGET is included in a Request, then the server MUST verify that it has previously received a binding request from the same address as is specified in XOR-RESPONSE-TARGET. 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 481.

If the Request contains a XOR-RESPONSE-TARGET attribute and the server is authenticating such requests, then the server checks the message for a MESSAGE-INTEGRITY attribute and a USERNAME. If they are not present the server MUST generate an error response of type 401.

Because XOR-RESPONSE-TARGET offers the potential for minor indirection attacks, a server MUST either authenticate the users
requesting its use or rate-limit its response to those requests. Rate-limiting is RECOMMENDED for responses to authenticated requests unless the server is deployed for an application that requires more frequent responses. If the Request contains a XOR-RESPONSE-TARGET attribute and the server is rate-limiting such requests, it MUST ensure that it does not generate a Response on a particular address more often than one per second. If the server receives requests whose responses are being rate-limited more often than one per second, it MUST generate a 503 (Service unavailable) Response to the Request.

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 A1 and A2 be the two IP addresses used by the server and P1 and P2 be the ports used by the server. 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.

<table>
<thead>
<tr>
<th>Flags</th>
<th>Source Address</th>
<th>Source Port</th>
<th>OTHER-ADDRESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>Da</td>
<td>Dp</td>
<td>Ca:Cp</td>
</tr>
<tr>
<td>Change IP</td>
<td>Ca</td>
<td>Dp</td>
<td>Ca:Cp</td>
</tr>
<tr>
<td>Change port</td>
<td>Da</td>
<td>Cp</td>
<td>Ca:Cp</td>
</tr>
<tr>
<td>Change IP and</td>
<td>Ca</td>
<td>Cp</td>
<td>Ca:Cp</td>
</tr>
<tr>
<td>Change port</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

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

If the server supports an alternate address and port 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 XOR-RESPONSE-TARGET attribute. If the XOR-RESPONSE-TARGET attribute is included, then it includes an XOR-REFLECTED-FROM attribute with the source address the Request was received from.

If the Request contained a PADDING attribute, PADDING MUST be included in the Binding Response. The server SHOULD use a length of PADDING equal to the MTU on the outgoing interface. If the Request also contains the XOR-RESPONSE-TARGET attribute the server MUST return an error response of type 400.

Following that, the server completes the remainder of the processing from STUN [RFC5389]. If authentication is being required, the server MUST include a MESSAGE-INTEGRITY and associated attributes as appropriate. A FINGERPRINT attribute is only required if the STUN messages are being multiplexed with application traffic that requires use of a FINGERPRINT to distinguish STUN messages.

An ALTERNATE-SERVER attribute MUST NOT be included with any other attribute defined in this specification.

When the server sends the Response, it is sent from the source address as determined above and to the destination address determined from the XOR-RESPONSE-TARGET, or to the source address of the Request otherwise.

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. CHANGE-REQUEST was originally defined in RFC3489 [RFC3489] but is redefined here as that document is obsoleted by [RFC5389].
7.1. Representing Transport Addresses

Whenever an attribute contains a transport IP address and port, it has the same format as MAPPED-ADDRESS. Similarly, the XOR-attributes have the same format as XOR-MAPPED-ADDRESS[RFC5389].

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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A B 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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. RESPONSE-ORIGIN

The RESPONSE-ORIGIN attribute is inserted by the server and 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.

7.4. OTHER-ADDRESS

The OTHER-ADDRESS attribute is used in Binding Responses. 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
server has a second IP address.

OTHER-ADDRESS uses the same attribute number as CHANGED-ADDRESS from
RFC3489 [RFC3489] because it is simply a new name with the same
semantics as CHANGED-ADDRESS. It has been renamed to more clearly
indicate its function.

7.5. XOR-REFLECTED-FROM

The XOR-REFLECTED-FROM attribute is present only in Binding Responses
when the Binding Request contained a XOR-RESPONSE-TARGET attribute.
The attribute contains the transport 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 anonymous denial-of-
service attacks.

The XOR-REFLECTED-FROM attribute is used in place of RFC3489’s
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.

7.6. XOR-RESPONSE-TARGET

The XOR-RESPONSE-TARGET attribute contains an IP address and port.
The XOR-RESPONSE-TARGET 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 XOR-RESPONSE-TARGET that does not contain
MESSAGE-INTEGRITY. The XOR-RESPONSE-TARGET attribute is optional in
the Binding Request.

XOR-RESPONSE-TARGET is used in place of RFC3489’s RESPONSE-ADDRESS.
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.

7.7. PADDING

The PADDING attribute allows for the entire message to be padded to
force the STUN message to be divided into IP fragments. PADDING
consists entirely of a freeform string, the value of which does not
matter. PADDING can be used in either Binding Requests or Binding
Responses.

PADDING MUST NOT be longer than the length that brings the total IP
datagram size to 64K and SHOULD be an even multiple of four bytes.
Because STUN messages with PADDING are intended to test the behavior
of dUDP fragments, they are an exception to the usual rule that STUN
messages be less than the MTU of the path.

7.8. CACHE-TIMEOUT

The CACHE-TIMEOUT is used in Binding Requests and Responses. It
indicates the time duration (in seconds) that the server will cache
the source address and USERNAME of an original binding request that
will later be followed by a request from a different source address
with a XOR-RESPONSE-TARGET asking that a response be reflected to the
source address of the original binding request. A server SHOULD NOT
send a response to a target address requested with XOR-RESPONSE-
TARGET unless it has cached that the same USERNAME made a previous
binding request from that target address. The client inserts a value
in CACHE-TIMEOUT into the Binding Request indicating the amount of
time it would like the server to cache that information. The server
responds with a CACHE-TIMEOUT in its Binding Response providing a
prediction of how long it will cache that information. The response
value can be greater than, equal to, or less than the requested
value. If the server is not able to provide such an estimate or the
information in the response would be meaningless, the server should
not include a CACHE-TIMEOUT attribute in its response.

8. New Response Codes

This draft defines new STUN response code.

8.1. 481 Connection does not exist

This code is generated when a server has received an XOR-RESPONSE-
TARGET, but the server has no record of having received a prior
Binding Request from the address specified in XOR-RESPONSE-TARGET.
The client SHOULD send a new Binding Request from the address it intends to specify in an XOR-RESPONSE-TARGET. This new Binding Request SHOULD include a CACHE-TIMEOUT attribute with the value set to the desired duration. If the server’s response includes a CACHE-TIMEOUT duration that is shorter than the client’s requested duration, the server is unable to satisfy the caching time requested by the client and the client SHOULD NOT continue to retry the request.

8.2.  503 Service Unavailable

This response is generated when a server receives Requests specifying a particular address in their XOR-RESPONSE-TARGET attribute more often than one per second.

9.  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 [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.

9.1.  Problem Definition

From [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 instantaneous 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 and an appropriate statistical model of the behavior required for the application to succeed.

9.2.  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 for v4 NATs. At the time of this writing, it appears some sort of NAT will be necessary between v6 clients and v4 servers, but this specification will not be necessary with those v6 to v4 NATs, because the IETF is planning to adequately describe their operation. This specification will be of no interest for v6 to v6 connectivity.

9.3. 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. This draft is experimental because the extent to which brittleness is introduced to an application relying on the Behavior Discovery usage is unclear and must be carefully evaluated by the designers of the protocol making use of it. The experimental test for this protocol is essentially determining whether an application can be made less brittle through the use of behavior-discovery information than it would be if attempted to make use of the network without any awareness of the NATs its traffic must pass through.

9.4. 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.

As long as v4 NATs are present, means of adapting to their presence will be required. As described above, well-behaved v6 to v4 NATs and direct v6 to v6 connections will not require behavior characterization.
9.5. 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-FROM and XOR-RESPONSE-TARGET attributes instead of the older REFLECTED-FROM and RESPONSE-ADDRESS attributes.

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.

10. IANA Considerations

This specification requests that IANA make additions to the "STUN Attributes Registry" and "STUN Error Code Registry". IANA is also requested to add an SRV registration for "stun-behavior" for this STUN usage.

10.1. STUN Attribute Registry

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.

0x0003: CHANGE-REQUEST
0x0027: XOR-RESPONSE-TARGET
0x0028: XOR-REFLECTED-FROM
0x0026: PADDING
0x8027: CACHE-TIMEOUT
0x802b: RESPONSE-ORIGIN
0x802c: OTHER-ADDRESS

10.2. STUN Error Code Registry

This specification defines two new STUN error response codes.
481: Connection does not exist
503: Service Unavailable

11. Security Considerations

This usage inherits the security considerations of STUN [RFC5389]. 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 Response can completely hide the STUN server from the client.

XOR-RESPONSE-TARGET allows a STUN server to be used as a reflector for denial-of-service attacks. It does not provide any amplification of the attack. 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. XOR-RESPONSE-TARGET is rate-limited or uses pre-existing credentials to alleviate this threat. Server caching previous contacts before directing a response to a XOR-RESPONSE-TARGET further eliminates the threat, although it introduces the complexity of state into a STUN server. CACHE-TIMEOUT is used to reduce the amount of additional state required.

The only attack possible with the PADDING attribute 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.

CHANGE-REQUEST provides no attacks, but adds three more reflection sources for the XOR-RESPONSE-TARGET reflection attacks. It provides no additional amplification and the security mechanisms for XOR-RESPONSE-TARGET are deemed sufficient.

RESPONSE-ORIGIN, CACHE-TIMEOUT and XOR-REFLECTED-FROM do not provide any additional attacks.
12. 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. Thanks to Dan Wing, Cullen Jennings, and Magnus Westerlund for detailed comments.

13. References

13.1. Normative References


13.2. Informative References


Appendix A. Change Log

RFC-EDITOR: Please remove this entire Change Log section while formatting this document for publication.

A.1. from draft-macdonald-behave-nat-behavior-diagnostics-00

- Only OTHER-ADDRESS, CHANGE-ADDRESS, and XOR-RESPONSE-TARGET support is optional; support for PADDING and SOURCE-ADDRESS is now mandatory
- PADDING is now a mandatory attribute
- OTHER-ADDRESS is returned in all binding responses if the server has a second IP address

A.2. from draft-ietf-behave-nat-behavior-discovery-00

- Clarified that only servers with two IP addresses should have an SRV entry
- Removed support for backward compatibility with 3489 clients by removing non-XOR forms of attributes. Language states that backward compatibility with 3489 clients is SHOULD NOT. Compatibility with 3489 servers is left unspecified.
- PADDING is mandatory and language has been changed to indicate that if a server supports PADDING it must either actually provide the padding or return an error (can’t support it but refuse to do it)
- Require both MAPPED-ADDRESS and XOR-MAPPED-ADDRESS to be returned to support detection of generic ALGs

A.3. from draft-ietf-behave-nat-behavior-discovery-01

- Changed proposed status to experimental
- Made significant changes to the introduction and applicability statements to reflect the experimental status
- Fixed the New Attributes and IANA considerations not listing the same attribute numbers.
o Removed mandatory shared secret credentials in favor of the option of rate limiting or credentials. Specified that credentials must be obtained from the user or parent application.

o Made OTHER-ADDRESS and SOURCE-ADDRESS optional to address compatibility with 3489bis clients. Renamed SOURCE-ADDRESS as RESPONSE-ORIGIN to avoid conflicts with 3489.

o Renamed XOR-RESPONSE-ADDRESS to XOR-RESPONSE-TARGET

o Added discussion of FINGERPRINT and ALTERNATE-SERVER for compliance with 3489bis stun usage definition requirements.

A.4. from draft-ietf-behave-nat-behavior-discovery-02

o fix terminology for endpoint-independent, address-dependent, and address and port-dependent from rfc4787

o define the ALG detection to apply to UDP and TCP

o fix >From typo in 9.5

o added exception to single MTU size restriction for PADDING

o removed OPEN ISSUE about CHANGE-REQUEST IANA registry based on the belief that we need to list that definition here now that 3489bis is dropping it.

A.5. from draft-ietf-behave-nat-behavior-discovery-03

o moved semantics of PADDING usage into behavior sections rather than attributes section

o removed reference to SERVER attribute

o removed Open Issues section

o Updated IAB considerations

A.6. from draft-ietf-behave-nat-behavior-discovery-04

o Clarified that behavior may vary by port used as well as by destination IP/particular STUN server, and therefore specified that all tests should be performed using the port the application will use

o Added additional text on selecting random port/ensuring port has been unused prior to starting filtering tests
specified limit to start rate of tests and that tests retransmissions should not synchronize

additional explanatory text for XOR-RESPONSE-TARGET

added SRV entry to IANA section and subdivided to match STUN registries from 5389

clarified that test combinations are non-normative

Numerous clarifications

Changed PADDING to default to interface MTU, and changed maximum length to not make IP datagram exceed 64K

Added text that server should allocate TCP and TCP/TLS

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
5251-18 John Tyler Highway #330
Williamsburg, VA 23185
USA

Email: bbl@lowekamp.net