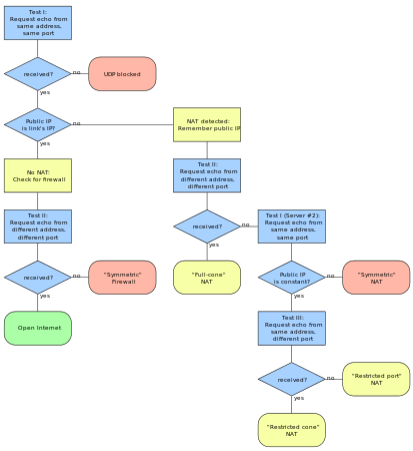
STUN

STUN (**Session Traversal Utilities for NAT**; originally **Simple Traversal of User Datagram Protocol (UDP) through Network Address Translators**) is a standardized set of methods, including a network protocol, for traversal of network address translator (NAT) gateways in applications of real-time voice, video, messaging, and other interactive communications.

STUN is a tool used by other protocols, such as Interactive Connectivity Establishment (ICE), the Session Initiation Protocol (SIP), and WebRTC. It provides a tool for hosts to discover the presence of a network address translator, and to discover the mapped, usually public, Internet Protocol (IP) address and port number that the NAT has allocated for the application's User Datagram Protocol (UDP) flows to remote hosts. The protocol requires assistance from a third-party network server (STUN server) located on the opposing (public) side of the NAT, usually the public Internet.

STUN was first announced in RFC 3489;[1] the title was changed in a specification of an updated set of methods published as RFC 5389, retaining the same acronym.[2]

History



Original NAT characterization algorithm of RFC 3489

STUN was first announced in RFC 3489.[1] The original specification specified an algorithm to characterize NAT behavior according to the address and port mapping behavior. This algorithm is not reliably successful and only applicable to a subset of NAT devices deployed. The algorithm consists of a series of tests to be performed by an application. When the path through the diagram ends in a red box, UDP communication is not possible and when the path ends in a yellow or green box, communication is possible. The methods of RFC 3489 proved too unreliable to cope with the plethora of different NAT implementations and application scenarios encountered in production networks. The STUN protocol and method were updated in RFC 5389, retaining many of the original specifications as a subset of methods, but removing others.

The title was changed in a specification of an updated set of methods published as RFC 5389, retaining the same acronym.[2]

Design

STUN is a tool for communications protocols to detect and traverse network address translators that are located in the path between two endpoints of communication. It is implemented as a light-weight client–server protocol, requiring only simple query and response components with a third-party server located on the common, easily accessible network, typically the Internet. The client side is implemented in the user's communications application, such as a Voice over Internet Protocol (VoIP) phone or an instant messaging client.

The basic protocol operates essentially as follows: The client, typically operating inside a private network, sends a *binding request* to a STUN server on the public Internet. The STUN server responds with a *success response* that contains the IP address and port number of the client, as observed from the server's perspective. The result is obfuscated through exclusive or (XOR) mapping to avoid translation of the packet content by application layer gateways (ALGs) that perform deep packet inspection in an attempt to perform alternate NAT traversal methods.

STUN messages are sent in User Datagram Protocol (UDP) packets. Since UDP does not provide reliable transport, reliability is achieved by application-controlled retransmissions of the STUN requests. STUN servers do not implement any reliability mechanism for their responses.[2] When reliability is mandatory, the Transmission Control Protocol (TCP) may be used, but induces extra networking overhead. In security-sensitive applications, STUN may be transported and encrypted by Transport Layer Security (TLS).

An application may automatically determine a suitable STUN server for communications with a particular peer by querying the Domain Name System (DNS) for the *stun* (for UDP) or *stuns* (for TCP/TLS) server (SRV) resource record, e.g., _stun._udp.example.com. The standard listening port number for a STUN server is 3478 for UDP and TCP, and 5349 for TLS. Alternatively, TLS may also be run on the TCP port if the server implementation can de-multiplex TLS and STUN packets. In case no STUN server is found using DNS lookups, the standard recommends that the destination domain name should be queried for address records (A or AAAA), which would be used with the default port numbers.[2]

In addition to using protocol encryption with TLS, STUN also has built-in authentication and message-integrity mechanisms via specialized STUN packet types.

When a client has evaluated its external address, it can use this as a candidate for communicating with peers by sharing the external NAT address rather than the private address, which is not reachable from peers on the public network.

If both communicating peers are located in different private networks, each behind a NAT, the peers must coordinate to determine the best communication path between them. Some NAT behavior may restrict peer connectivity even when the public binding is known. The Interactive Connectivity Establishment (ICE) protocol provides a structured mechanism to determine the optimal communication path between two peers. Session Initiation Protocol (SIP) extensions are defined to enable the use of ICE when setting up a call between two hosts.

Limitations

Network address translation is implemented via a number of different address and port mapping schemes, none of which is standardized.

STUN is not a self-contained NAT traversal solution applicable in all NAT deployment scenarios and does not work correctly with all of them. It is a tool among other methods and it is a tool for other protocols in dealing with NAT traversal, most notably Traversal Using Relay NAT (TURN) and Interactive Connectivity Establishment (ICE).

STUN works with three types of NAT: full cone NAT, restricted cone NAT, and port restricted cone NAT. In the cases of restricted cone or port restricted cone NATs, the client must send out a packet to the endpoint before the NAT will allow packets from the endpoint through to the client. STUN does not work with symmetric NAT (also known as bi-directional NAT) which is often found in the networks of large companies. Since the IP address of the STUN server is different from that of the endpoint, in the symmetric NAT case, the NAT mapping will be different for the STUN server than for an endpoint. TURN offers better results with symmetric NAT.

See also

- Internet Gateway Device Protocol
- Port Control Protocol
- UDP hole punching

References

1. *RFC 3489 (https://datatracker.ietf.org/doc/html/rfc3489)*
2. *RFC 5389 (https://datatracker.ietf.org/doc/html/rfc5389)*

External links

- STUNTMAN - Open source STUN server software (http://www.stunprotocol.org/)
- Yahoo VoIP STUN (https://www.youtube.com/watch?v=9MWYw0fltr0) on YouTube
- STUNT: TCP NAT traversal (https://web.archive.org/web/20170911122644/http://nutss.gforge.cis.cornell.edu/stunt.php) at the Wayback Machine (archived 2017-09-11)

- What are Session Traversal Utilities for NAT (STUN) and Traversal Using Relays around NAT (TURN)? (https://www.callstats.io/blog/what-are-stun-and-turn) at callstats.io
- Session Traversal Utilities for NAT (STUN) (https://help.hcltechsw.com/sametime/11.0.2/admin/session_traversal_utilities.html) at HCL Software

---

Last edited 4 months ago by BlackcurrantTea