Internet Engineering Task Force (IETF)
Request for Comments: 6544
Category: Standards Track
ISSN: 2070-1721

J. Rosenberg
jdrosen.net
A. Keranen
Ericsson
B. B. Lowekamp
Skype
A. B. Roach
Tekelec

March 2012

TCP Candidates with Interactive Connectivity Establishment (ICE)

Interactive Connectivity Establishment (ICE) defines a mechanism for NAT traversal for multimedia communication protocols based on the offer/answer model of session negotiation. ICE works by providing a set of candidate transport addresses for each media stream, which are then validated with peer-to-peer connectivity checks based on Session Traversal Utilities for NAT (STUN). ICE provides a general framework for describing candidates but only defines UDP-based media streams. This specification extends ICE to TCP-based media, including the ability to offer a mix of TCP and UDP-based candidates for a single stream.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6544.

Table of Contents

1. Introduction

Interactive Connectivity Establishment (ICE) [RFC5245] defines a mechanism for NAT traversal for multimedia communication protocols based on the offer/answer model [RFC3264] of session negotiation. ICE works by providing a set of candidate transport addresses for each media stream, which are then validated with peer-to-peer connectivity checks based on Session Traversal Utilities for NAT (STUN) [RFC5389]. However, ICE only defines procedures for UDP-based transport protocols.

There are many reasons why ICE support for TCP is important. First, there are media protocols that only run over TCP. Such protocols are used, for example, for screen sharing and instant messaging [RFC4975]. For these protocols to work in the presence of NAT, unless they define their own NAT traversal mechanisms, ICE support for TCP is needed. In addition, RTP can also run over TCP [RFC4571]. Typically, it is preferable to run RTP over UDP, and not TCP. However, in a variety of network environments, overly restrictive NAT and firewall devices prevent UDP-based communications altogether, but general TCP-based communications are permitted. In such environments, sending RTP over TCP, and thus establishing the media session, may be preferable to having it fail altogether. With this specification, agents can gather UDP and TCP candidates for a media stream, list the UDP ones with higher priority, and then only use the TCP-based ones if the UDP ones fail. This provides a fallback mechanism that allows multimedia communications to be highly reliable.

The usage of RTP over TCP is particularly useful when combined with Traversal Using Relays around NAT (TURN) [RFC5766]. In this case, one of the agents would connect to its TURN server using TCP and obtain a TCP-based relayed candidate. It would offer this to its peer agent as a candidate. The other agent would initiate a TCP connection towards the TURN server. When that connection is established, media can flow over the connections, through the TURN server. The benefit of this usage is that it only requires the agents to make outbound TCP connections to a server on the public network. This kind of operation is broadly interoperable through NAT and firewall devices. Since it is a goal of ICE and this extension to provide highly reliable communications that "just work" in as broad a set of network deployments as possible, this use case is particularly important.

This specification extends ICE by defining its usage with TCP candidates. It also defines how ICE can be used with RTP and Secure RTP (SRTP) to provide both TCP and UDP candidates. This specification does so by following the outline of ICE itself and

calling out the additions and changes to support TCP candidates in ICE. The base behavior of ICE [RFC5245] remains unchanged except for the extensions in this document that define the usage of ICE with TCP candidates.

It should be noted that since TCP NAT traversal is more complicated than with UDP, ICE TCP is not generally as efficient as UDP-based ICE. Discussion about this topic can be found in Appendix A.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This document uses the same terminology as ICE (see Section 3 of [RFC5245]).

3. Overview of Operation

The usage of ICE with TCP is relatively straightforward. This specification mainly deals with how and when connections are opened and how those connections relate to candidate pairs.

When agents perform address allocations to gather TCP-based candidates, three types of candidates can be obtained: active candidates, passive candidates, and simultaneous-open (S-O) candidates. An active candidate is one for which the agent will attempt to open an outbound connection but will not receive incoming connection requests. A passive candidate is one for which the agent will receive incoming connection attempts but not attempt a connection. An S-O candidate is one for which the agent will attempt to open a connection simultaneously with its peer.

When gathering candidates from a host interface, the agent typically obtains active, passive, and S-O candidates. Similarly, one can use different techniques for obtaining, e.g., server reflexive, NAT-assisted, tunneled, or relayed candidates of these three types (see Section 5). Connections to servers used for relayed and server reflexive candidates are kept open during ICE processing.

When encoding these candidates into offers and answers, the type of the candidate is signaled. In the case of active candidates, both IP address and port are present, but the port is meaningless (it is there only for making encoding of active candidates consistent with the other candidate types and is ignored by the peer). As a consequence, active candidates do not need to be physically allocated

at the time of address gathering. Rather, the physical allocations, which occur as a consequence of a connection attempt, occur at the time of the connectivity checks.

When the candidates are paired together, active candidates are always paired with passive, and S-O candidates with each other. When a connectivity check is to be made on a candidate pair, each agent determines whether it is to make a connection attempt for this pair.

The actual process of generating connectivity checks, managing the state of the check list, and updating the Valid list works identically for TCP as it does for UDP.

ICE requires an agent to demultiplex STUN and application-layer traffic, since they appear on the same port. This demultiplexing is described in [RFC5245] and is done using the magic cookie and other fields of the message. Stream-oriented transports introduce another wrinkle, since they require a way to frame the connection so that the application and STUN packets can be extracted in order to differentiate STUN packets from application-layer traffic. For this reason, TCP media streams utilizing ICE use the basic framing provided in RFC 4571 [RFC4571], even if the application layer protocol is not RTP.

When Transport Layer Security (TLS) or Datagram Transport Layer Security (DTLS) is used, they are also run over the RFC 4571 framing shim, while STUN runs outside of the (D)TLS connection. The resulting ICE TCP protocol stack is shown in Figure 1, with (D)TLS on the left side and without it on the right side.

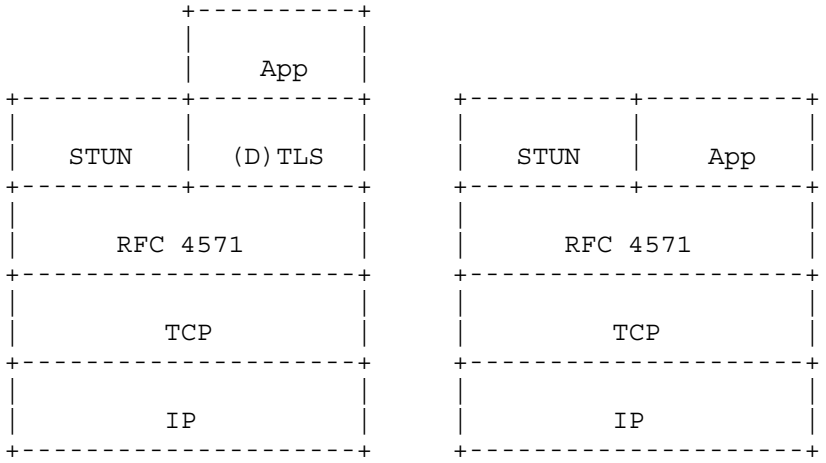

Figure 1: ICE TCP Stack with and without (D)TLS

The implication of this is that, for any media stream protected by (D)TLS, the agent will first run ICE procedures, exchanging STUN messages. Then, once ICE completes, (D)TLS procedures begin. ICE and (D)TLS are thus "peers" in the protocol stack. The STUN messages are not sent over the (D)TLS connection, even ones sent for the purposes of keepalive in the middle of the media session.

4. Sending the Initial Offer

For offerers making use of ICE for TCP streams, the procedures below are used. The main differences compared to UDP candidates are the new methods for gathering candidates, how TCP candidates are prioritized, and how they are encoded in the Session Description Protocol (SDP) offer and answer.

4.1. Gathering Candidates

Providers of real-time communications services may decide that it is preferable to have no media at all rather than to have media over TCP. To allow for choice, it is RECOMMENDED that it be possible to configure agents to either obtain or not obtain TCP candidates for real-time media.

Having it be configurable, and then configuring it to be off, is far better than not having the capability at all. An important goal of this specification is to provide a single mechanism that can be used across all types of endpoints. As such, it is preferable to account for provider and network variation through configuration instead of hard-coded limitations in an implementation. Besides, network characteristics and connectivity assumptions can, and will, change over time. Just because an agent is communicating with a server on the public network today doesn't mean that it won't need to communicate with one behind a NAT tomorrow. Just because an agent is behind a NAT with endpoint-independent mapping today doesn't mean that tomorrow it won't pick up its agent and take it to a public network access point where there is a NAT with address- and port-dependent mapping properties or one that only allows outbound TCP. The way to handle these cases and build a reliable system is for agents to implement a diverse set of techniques for allocating addresses, so that at least one of them is almost certainly going to work in any situation. Implementors should consider very carefully any assumptions made about deployments before electing not to implement one of the mechanisms for address allocation. In particular, implementors should consider whether the elements in the system may be mobile and connect through different networks with different connectivity. They should also consider whether endpoints that are under their control, in terms of location and network connectivity, would always be under their control. In environments

where mobility and user control are possible, a multiplicity of techniques is essential for reliability.

First, agents SHOULD obtain host candidates as described in Section 5.1. Then, each agent SHOULD "obtain" (allocate a placeholder for) an active host candidate for each component of each TCP-capable media stream on each interface that the host has. The agent does not yet have to actually allocate a port for these candidates, but they are used for the creation of the check lists.

The agent SHOULD then obtain server reflexive, NAT-assisted, and/or UDP-tunneled candidates (see Section 5.2, Section 5.3, and Section 5.4). The mechanisms for establishing these candidates and the number of candidates to collect vary from technique to technique. These considerations are discussed in the relevant sections.

Next, agents SHOULD obtain passive (and possibly S-O) relayed candidates for each component as described in Section 5.5. Each agent SHOULD also allocate a placeholder for an active relayed candidate for each component of each TCP-capable media stream.

It is highly RECOMMENDED that a host obtains at least one set of host candidates and one set of relayed candidates. Obtaining additional candidates will increase the chance of successfully creating a direct connection.

Once the candidates have been obtained, the agent MUST keep the TCP connections open until ICE processing has completed. See Appendix B for important implementation guidelines.

If a media stream is UDP-based (such as RTP), an agent MAY use an additional host TCP candidate to request a UDP-based candidate from a TURN server (or some other relay with similar functionality). Usage of such UDP candidates follows the procedures defined in ICE for UDP candidates.

Like its UDP counterparts, TCP-based STUN transactions are paced out at one every Ta milliseconds (see Section 16 of [RFC5245]). This pacing refers strictly to STUN transactions (both Binding and Allocate requests). If performance of the transaction requires establishment of a TCP connection, then the connection gets opened when the transaction is performed.

4.2. Prioritization

The transport protocol itself is a criteria for choosing one candidate over another. If a particular media stream can run over UDP or TCP, the UDP candidates might be preferred over the TCP

candidates. This allows ICE to use the lower latency UDP connectivity if it exists but fallback to TCP if UDP doesn't work.

In Section 4.1.2.1 of [RFC5245], a recommended formula for UDP ICE candidate prioritization is defined. For TCP candidates, the same formula and candidate type preferences SHOULD be used, and the RECOMMENDED type preferences for the new candidate types defined in this document (see Section 5) are 105 for NAT-assisted candidates and 75 for UDP-tunneled candidates.

When both UDP and TCP candidates are offered for the same media stream, and one transport protocol should be preferred over the other, the type preferences for the preferred transport protocol candidates SHOULD be increased and/or the type preferences for the other transport protocol candidates SHOULD be decreased. How much the values should be increased or decreased depends on whether it is more important to choose a certain transport protocol or a certain candidate type. If the candidate type is more important (e.g., even if UDP is preferred, TCP host candidates are preferred over UDP server reflexive candidates) changing type preference values by one for the other transport protocol candidates is enough. On the other hand, if the transport protocol is more important (e.g., any UDP candidate is preferred over any TCP candidate), all the preferred transport protocol candidates SHOULD have type preference higher than the other transport protocol candidates. However, it is RECOMMENDED that the relayed candidates are still preferred lower than the other candidate types. For RTP-based media streams, it is RECOMMENDED that UDP candidates are preferred over TCP candidates.

With TCP candidates, the local preference part of the recommended priority formula is updated to also include the directionality (active, passive, or simultaneous-open) of the TCP connection. The RECOMMENDED local preference is then defined as:

```
local preference = (2^13) * direction-pref + other-pref
```

The direction-pref MUST be between 0 and 7 (both inclusive), with 7 being the most preferred. The other-pref MUST be between 0 and 8191 (both inclusive), with 8191 being the most preferred. It is RECOMMENDED that the host, UDP-tunneled, and relayed TCP candidates have the direction-pref assigned as follows: 6 for active, 4 for passive, and 2 for S-O. For the NAT-assisted and server reflexive candidates, the RECOMMENDED values are: 6 for S-O, 4 for active, and 2 for passive.

The preference priorities listed here are simply recommendations that try to strike a balance between success probability and the resulting path's efficiency. Depending on the scenario where ICE TCP is used,