ICE Working Group
Internet-Draft
Updates: 8445 (if approved)
Intended status: Standards Track
Expires: April 15, 2020

C. Holmberg
Ericsson
J. Uberti
Google

October 13, 2019

Interactive Connectivity Establishment Patiently Awaiting Connectivity (ICE PAC) draft-ietf-ice-pac-03

Abstract

During the process of establishing peer-to-peer connectivity, ICE agents can encounter situations where they have no candidate pairs to check, and, as a result, conclude that ICE processing has failed. However, because additional candidate pairs can be discovered during ICE processing, declaring failure at this point may be premature. This document discusses when these situations can occur and proposes a way to avoid premature failure. This document updates RFC 8445 and RFC XXXX.

[RFC EDITOR NOTE: Please replace RFC XXXX with the RFC number of draft-ietf-ice-trickle once it has been published. Please also indicate that this specification updates RFC XXXX.]

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 15, 2020.

Copyright Notice

Table of Contents

1. Introduction

[RFC8445] describes a protocol, Interactive Connectivity Establishment (ICE), for Network Address Translator (NAT) traversal for UDP-based communication.

When using ICE, endpoints will typically exchange ICE candidates, form a list of candidate pairs, and then test each candidate pair to see if connectivity can be established. If the test for a given pair fails, it is marked accordingly, and if all pairs have failed, the overall ICE process typically is considered to have failed.

During the process of connectivity checks, additional candidates may be created as a result of successful inbound checks from the remote peer. Such candidates are referred to as peer-reflexive candidates, and once discovered, will be used to form new candidate pairs which will be tested like any other. However, there is an inherent race condition here; if, before learning about any peer-reflexive candidates, an endpoint runs out of candidate pairs to check, either because it has none, or it considers them all to have failed, it will prematurely declare failure and terminate ICE processing. This race condition can occur in many common situations.

This specification updates [RFC8445], by simply requiring that an ICE agent wait a minimum amount of time before declaring ICE failure, even if there are no candidate pairs to check, or if all candidate pairs have failed. This delay provides enough time for the discovery of peer-reflexive candidates, which may eventually lead to ICE processing completing successfully.

2. Conventions

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Relevant Scenarios

As noted above, the core problem this specification attempts to address is the situation where even after local gathering and remote candidate signaling has completed, the ICE agent immediately ends up with no valid pairs and no candidate pairs left to check, resulting in a premature ICE failure. This failure is premature because not enough time has elapsed to allow for discovery of peer-reflexive candidates from inbound connectivity checks; if discovered, these candidates are very likely to result in a valid pair.

In most ICE scenarios, the lengthy timeouts for connectivity check transactions, typically tens of seconds, will prevent this problem from occurring. However, there are certain specific cases where this problem will frequently occur.

3.1. No Candidates From Peer

It is entirely legal for an ICE agent to provide zero candidates of its own. If the agent somehow knows that the remote endpoint is directly reachable, gathering local candidates is unnecessary and will only cause delays; the peer agent can discover the appropriate local candidate via connectivity checks.

However, following the procedures from [RFC8445] strictly will result in immediate ICE failure, since the checklist at the peer agent will be empty.

3.2. All Candidates Discarded

Even if the ICE agent provides candidates, they may be discarded by the peer agent if it does not know what to do with them. For example, candidates may use an address family that the peer agent

does not support, (e.g., a host candidate with an IPv6 address in a NAT64 scenario), or may not be usable for some other reason.

In these scenarios, when the candidates are discarded, the checklist at the peer agent will once again be empty, leading to immediate ICE failure.

3.3. Immediate Candidate Pair Failure

Section 7.2.5.2 of [RFC8445] describes several situations in which a candidate pair will be considered to have failed, well before the connectivity check transaction timeout.

As a result, even if the ICE agent provides usable candidates, the pairs created by the peer agent may fail immediately when checked, e.g., a check to a non-routable address that receives an immediate ICMP error.

In this situation, the checklist at the peer agent may contain only failed pairs, resulting in immediate ICE failure.

4. Update to RFC 8445

In order to avoid the problem raised by this document, the ICE agent needs to wait enough time to allow peer-reflexive candidates to be discovered. Accordingly, when a full ICE implementation begins its ICE processing, as described in [RFC8445], Section 6.1, it MUST set a timer, henceforth known as the PAC timer, to ensure ICE will run for a minimum amount of time before determining failure.

Specifically, the ICE agent will start its timer once it believes ICE connectivity checks are starting. This occurs when the agent has sent the values needed to perform connectivity checks (e.g., the Username Fragment and Password denoted in [RFC8445], Section 5.3) and has received some indication that the remote side is ready to start connectivity checks, typically via receipt of the values mentioned above. Note that the agent will start the timer even if it has not sent or received any ICE candidates.

The RECOMMENDED duration for the timer is equal to the agent's connectivity check transaction timeout, including all retransmissions. This timeout value is chosen to roughly coincide with the maximum possible duration of ICE connectivity checks from the remote peer, which, if successful, could create peer-reflexive candidates. Because the ICE agent doesn't know the exact number of candidate pairs and pacing interval in use by the remote side, this timeout value is simply a guess, albeit an educated one. Regardless, for this particular problem, the desired benefits will be realized as

long as the agent waits some reasonable amount of time, and, as usual, the application is in the best position to determine what is reasonable for its scenario.

While the timer is running, the ICE agent MUST NOT set the state of a checklist to Failed, even if the checklist has no pairs left to check. As a result, the ICE agent will not remove any data streams or set the state of the ICE session to Failed as long as the timer is running.

When the timer eventually elapses, the ICE agent MUST resume typical ICE processing, including setting any checklists containing only Failed pairs to the Failed state, as usual, and handling any consequences as indicated in [RFC8445], Section 8.1.2. Naturally, if there are no such checklists, no action is necessary.

One consequence of this behavior is that in cases where ICE should fail, e.g., where both sides provide candidates with unsupported address families, ICE will no longer fail immediately, and only fail when the PAC timer expires. However, because most ICE scenarios require an extended period of time to determine failure, the fact that some specific scenarios no longer fail fast should have minimal application impact, if any.

Note also that the PAC timer is potentially relevant to the ICE nomination procedure described in [RFC8445], Section 8.1.1. That specification does not define a minimum duration for ICE processing prior to nomination of a candidate pair, but in order to select the best candidate pair, ICE needs to run for enough time in order to allow peer-reflexive candidates to be discovered and checked, as noted above. Accordingly, the controlling ICE agent SHOULD wait a sufficient amount of time before nominating candidate pairs, and it MAY use the PAC timer to do so. As always, the controlling ICE agent retains full discretion, and MAY decide, based on its own criteria, to nominate pairs prior to the timer elapsing.

5. Update to RFC XXXX

[RFC EDITOR NOTE: Please replace RFC XXXX with the RFC number of draft-ietf-ice-trickle once it has been published.]

Trickle ICE [I-D.ietf-ice-trickle] considers a similar problem, namely whether an ICE agent should allow a checklist to enter the Failed state if more candidates might still be provided by the remote peer. The solution, specified in [I-D.ietf-ice-trickle], Section 8, is to wait until an end-of-candidates indication has been received before determining ICE failure.

However, for the same reasons described above, the ICE agent may discover peer-reflexive candidates after it has received the end-of-candidates indication, and so the solution proposed by this document MUST still be used even when the ICE agent is using Trickle ICE.

Note also that sending an end-of-candidates indication is only a SHOULD-strength requirement, which means that ICE agents will need to implement an backup mechanism to decide when all candidates have been received, typically a timer. Accordingly, ICE agents MAY use the PAC timer to also serve as an end-of-candidates fallback.

6. Security Considerations

The security considerations for ICE are defined in [RFC8445]. This specification only recommends that ICE agents wait for a certain time of period before they declare ICE failure, and does not introduce new security considerations.

7. IANA considerations

This specification makes no requests to IANA.

8. Acknowledgements

9. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018, <https://www.rfc-editor.org/info/rfc8445>.

[I-D.ietf-ice-trickle] Ivov, E., Rescorla, E., Uberti, J., and P. Saint-Andre, "Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol", draft-ietf-ice-trickle-21 (work in progress), April 2018.

Authors' Addresses

Christer Holmberg
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: christer.holmberg@ericsson.com

Justin Uberti
Google
747 6th St W
Kirkland 98033
USA

Email: justin@uberti.name