RTCWEB
Internet-Draft
Intended status: Informational
Expires: April 25, 2019

Y. Fablet
Apple Inc.
J. de Borst
J. Uberti
Q. Wang
Google

October 22, 2018

Using Multicast DNS to protect privacy when exposing ICE candidates
draft-ietf-rtcweb-mdns-ice-candidates-01

Abstract

WebRTC applications collect ICE candidates as part of the process of creating peer-to-peer connections. To maximize the probability of a direct peer-to-peer connection, client private IP addresses are included in this candidate collection. However, disclosure of these addresses has privacy implications. This document describes a way to share local IP addresses with other clients while preserving client privacy. This is achieved by obfuscating IP addresses with dynamically generated Multicast DNS (mDNS) [RFC6762] names.

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 https://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 25, 2019.

Table of Contents

1. Introduction

As detailed in [IPHandling], exposing client private IP addresses by default maximizes the probability of successfully creating direct peer-to-peer connection between two clients, but creates a significant surface for user fingerprinting. [IPHandling] recognizes this issue, but also admits that there is no current solution to this problem; implementations that choose to use Mode 3 to address the privacy concerns often suffer from failing or suboptimal connections in WebRTC applications. This is particularly an issue on unmanaged networks, typically homes or small offices, where NAT loopback may not be supported.

This document proposes an overall solution to this problem by registering ephemeral mDNS names for each local private IP address, and then providing those names, rather than the IP addresses, to the web application when it gathers ICE candidates. WebRTC implementations resolve these names to IP addresses and perform ICE processing as usual, but the actual IP addresses are not exposed to the web application.

2. Principle

This section uses the concept of ICE agent as defined in [RFC8445]. In the remainder of the document, it is assumed that each browsing context (as defined in Section 7.1 of [HTMLSpec]) has its own ICE agent.

2.1. ICE Candidate Gathering

For any host candidate gathered by an ICE agent as part of [RFC8445] section 5.1.1, the candidate is processed as follows:

1. Check whether the ICE agent has a usable registered mDNS hostname resolving to the ICE candidate's IP address. If one exists, skip ahead to Step 6.

2. Generate a unique mDNS hostname. The unique name MUST consist of a version 4 UUID as defined in [RFC4122], followed by ".local".

3. Register the candidate's mDNS hostname as defined in [RFC6762].

4. If registering of the mDNS hostname fails, abort these steps. The candidate is not exposed.

5. Store the mDNS hostname and its related IP address in the ICE agent for future reuse.

6. Replace the IP address of the ICE candidate with its mDNS hostname, and expose the candidate as usual.

An ICE agent can implement this procedure in any way so long as it produces equivalent results to this procedure.

An implementation may for instance pre-register mDNS hostnames by executing steps 3 to 5 and prepopulate an ICE agent accordingly. By doing so, only step 6 of the above procedure will be executed at the time of gathering candidates.

An implementation may also detect that mDNS is not supported by the available network interfaces. The ICE agent may skip steps 2 and 3 and directly decide to not expose the host candidate.

This procedure ensures that a mDNS name is used to replace only one IP address. Specifically, an ICE agent using an interface with both IPv4 and IPv6 addresses MUST expose a different mDNS name for each address.

2.2. ICE Candidate Processing

For any remote ICE candidate received by the ICE agent, the following procedure is used:

1. If the connection-address field value of the ICE candidate does not end with ".local" or if the value contains more than one ".", then process the candidate as defined in [RFC8445].

2. Otherwise, resolve the candidate using mDNS.

3. If it resolves to an IP address, replace the mDNS hostname of the ICE candidate with the resolved IP address and continue processing of the candidate.

4. Otherwise, ignore the candidate.

An ICE agent may use a hostname resolver that transparently supports both Multicast and Unicast DNS. In this case the resolution of a ".local" name may happen through Unicast DNS, see [RFC6762] section 3.

An ICE agent that supports mDNS candidates MUST support the situation where the hostname resolution results in more than one IP address. In this case, the ICE agent MUST take exactly one of the resolved IP addresses and ignore the others. The ICE agent SHOULD, if available, use the first IPv6 address resolved, otherwise the first IPv4 address.

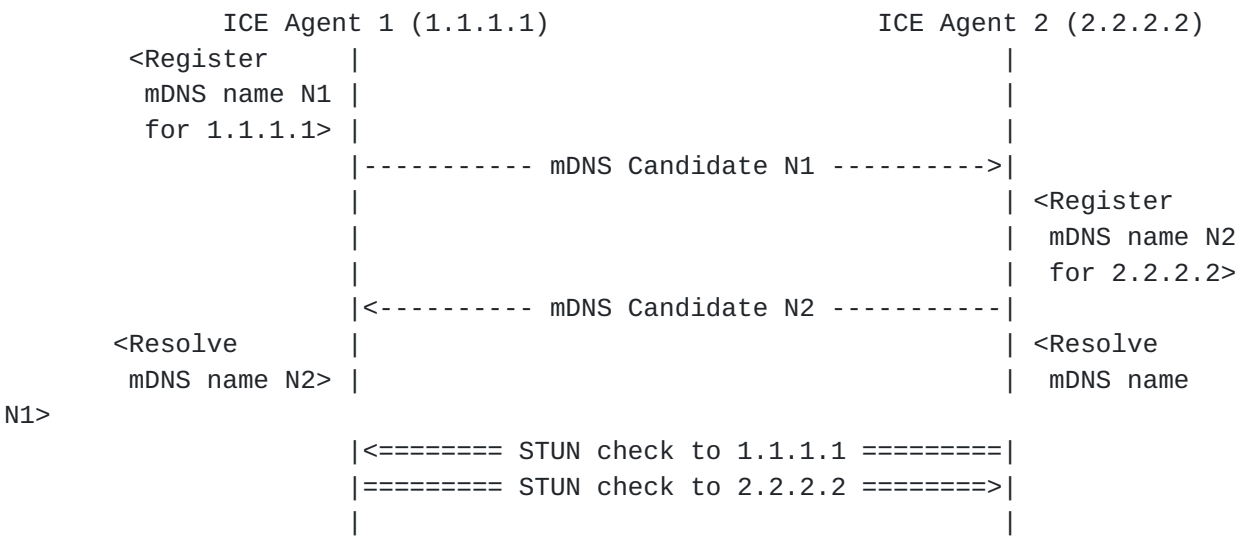
2.2.1. Handling of Peer-Reflexive Remote Candidate

A peer-reflexive remote candidate could be learned and constructed from the source transport address of the STUN Binding request as an ICE connectivity check. The peer-reflexive candidate could share the same address as a remote mDNS candidate that is in the process of being signaled or name resolution.

In addition to the elimination procedure of redundant candidates defined in Section 5.1.3 of [RFC8445], which could remove constructed peer-reflexive remote candidates, the address of any existing peer-reflexive remote candidate should not be exposed to Web applications by ICE agents that implement this proposal, as detailed in Section 4.

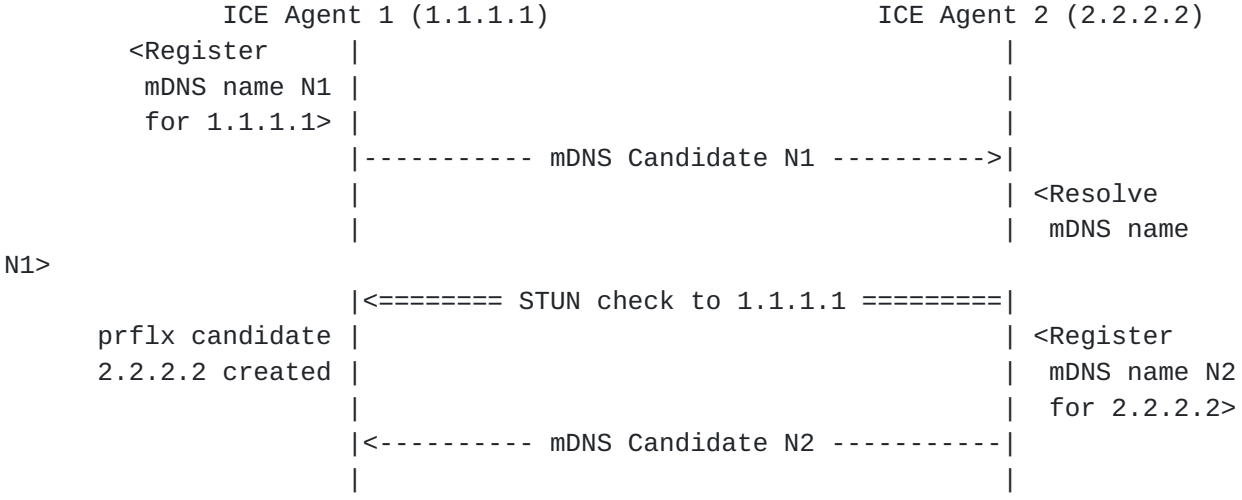
3. Examples

In this example, mDNS candidates are exchanged between peers and resolved to obtain the corresponding IP addresses.

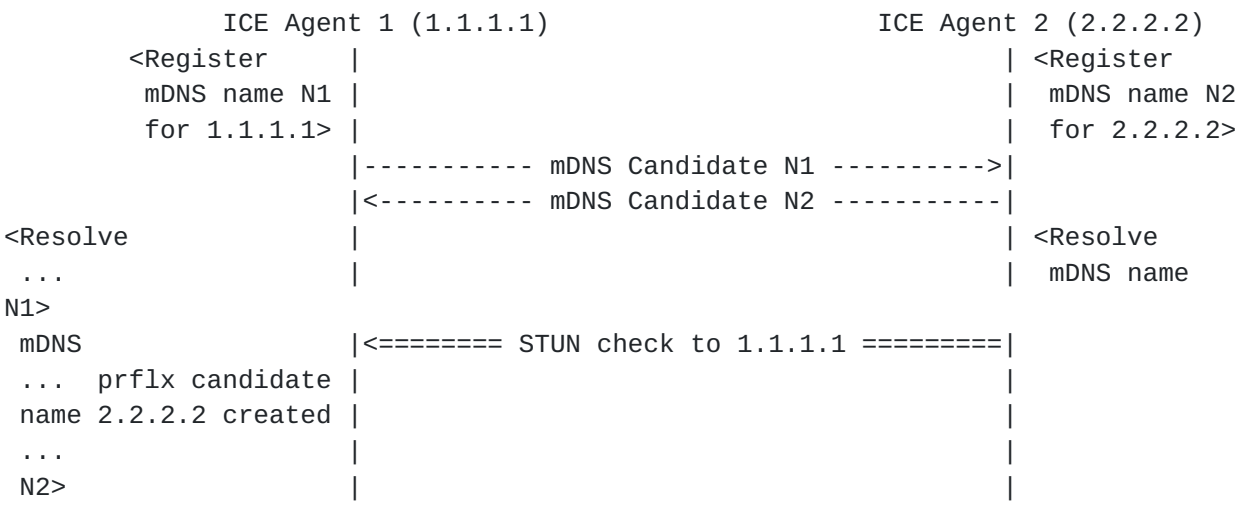


The following two examples indicate how peer-reflexive candidates for host IP addresses can be created due to timing differences.

In this example, a peer-reflexive candidate is generated because the mDNS candidate is signaled after the STUN checks begin.



In this example, a peer-reflexive candidate is generated because the mDNS resolution for name N2 does not complete until after the STUN checks are received.



4. Privacy Guidelines

4.1. APIs Leaking IP Addresses

When there is no user consent, the following filtering should be done to prevent private IP address leakage:

1. ICE candidates with an IP address are not exposed as ICE candidate events.

2. Server reflexive ICE candidate raddr field is set to 0.0.0.0 and rport to 0.

3. SDP does not expose any a=candidate line corresponding to an ICE candidate which contains an IP address.

4. Statistics related to ICE candidates MUST NOT contain the resolved IP address of a remote mDNS candidate or the IP address of a peer-reflexive candidate, unless that IP address has already been learned through other means, e.g., receiving it in a separate server-reflexive remote candidate.

4.2. Interactions With TURN Servers

When sending data to a TURN [RFC5766] server, the sending client tells the server the destination IP and port for the data. This means that if the client uses TURN to send to an IP that was obtained by mDNS resolution, the TURN server will learn the underlying host IP and port, and this information can then be relayed to the web application, defeating the value of the mDNS wrapping.

To prevent disclosure of the host IP address to a TURN server, the ICE agent MUST NOT form candidate pairs between its own relay candidates and remote mDNS candidates. Note that the converse is not

an issue; the ICE agent MAY form candidate pairs between its own mDNS candidates and remote relay candidates, as in this situation host IPs will not be sent directly to the TURN server.

This restriction has no effect on connectivity; in the cases where host IP addresses are private and need to be wrapped with mDNS names, they will be unreachable from the TURN server, and as noted above, the reverse path will continue to work normally.

4.3. Generated Names Reuse

It is important that use of registered mDNS hostnames is limited in time and/or scope. Indefinitely reusing the same mDNS hostname candidate would provide applications an even more reliable tracking mechanism than the private IP addresses that this specification is designed to hide. The use of registered mDNS hostnames SHOULD be scoped by origin, and SHOULD have the lifetime of the page.

4.4. Specific Browsing Contexts

As noted in [IPHandling], privacy may be breached if a web application running in two browsing contexts can determine whether it is running on the same device. While the approach in this document prevents the application from directly comparing local private IP addresses, a successful local WebRTC connection can also present a threat to user privacy. Specifically, when the latency of a WebRTC connection latency is close to zero, the probability is high that the two peers are running on the same device.

To avoid this issue, browsers SHOULD NOT register mDNS names for WebRTC applications running in a third-party browsing context (i.e., a context that has a different origin than the top-level browsing context), or a private browsing context.

5. Security Considerations

5.1. mDNS Message Flooding

The implementation of this proposal requires the mDNS querying capability of the browser for registering mDNS names or adding remote ICE host candidates with such names. It also requires the mDNS responding capability of either the browser or the operating platform of the browser for registering, removing or resolving mDNS names. In particular,

- the registration of name requires optional probing queries and mandatory announcing responses ([RFC6762], Section 8), and this is performed at the beginning of ICE gathering;

- the addition of remote ICE host candidates with mDNS names generates mDNS queries for names of each candidate;

- the removal of names could happen when the browsing context of the ICE agent is destroyed in an implementation, and goodbye responses should be sent to invalidate records generated by the ICE agent in the local network ([RFC6762], Section 10.1).

A malicious Web application could flood the local network with mDNS messages by:

- creating browsing contexts that create ICE agents and start gathering of local ICE host candidates;

- destroying these local candidates soon after the name registration is done;

- adding fictitious remote ICE host candidates with mDNS names.

[RFC6762] defines a per-record multicast rate limiting rule, in which a given record on a given interface cannot be sent less than one second since its last transmission. This rate limiting rule however does not mitigate the above attacks, in which new names, hence new records, are constantly created and sent. A browser-wide mDNS message rate limit MUST be provided for all messages that can be indirectly dispatched by a web application, namely the probing queries, announcement responses, resolution queries, and goodbye responses associated with mDNS.

5.2. Malicious Responses to Deny Name Registration

If the optional probing queries are implemented for the name registration, a malicious endpoint in the local network, which is capable of responding mDNS queries, could send responses to block the use of the generated names. This would lead to the discarding of this ICE host candidate as in Step 5 in Section 2.1.

The above attack can be mitigated by skipping the probing when registering a name, which also conforms to Section 8 in [RFC6762], given that the name is randomly generated for the probabilistic uniqueness (e.g. a version 4 UUID) in Step 3 in Section 2.1. However, a similar attack can be performed by exploiting the negative responses (defined in [RFC6762], Section 8.1), in which NSEC resource records are sent to claim the nonexistence of records related to the gathered ICE host candidates.

The existence of malicious endpoints in the local network poses a generic threat, and requires dedicated protocol suites to mitigate, which is beyond the scope of this proposal.

5.3. Monitoring of Sessions

A malicious endpoint in the local network may also record other endpoints who are registering, unregistering, and resolving mDNS names. By doing so, they can create a session log that shows which endpoints are communicating, and for how long. If both endpoints in the session are on the same network, the fact they are communicating can be discovered.

As above, mitigation of this threat is beyond the scope of this proposal.

6. Specification Requirements

The proposal relies on identifying and resolving any mDNS-based ICE candidates as part of adding/processing a remote candidate. [ICESDP] section 4.1 could be updated to explicitly allow mDNS names in the connection-address field.

The proposal relies on adding the ability to register mDNS names at ICE gathering time. This could be described in [ICESDP] and/or [WebRTCSpec].

The proposal allows updating [IPHandling] so that mode 2 is not the mode used by default when user consent is not required. Instead, the default mode could be defined as mode 3 with mDNS-based ICE candidates.

7. Informative References

[HTMLSpec] "HTML Living Standard", n.d., <https://html.spec.whatwg.org>.

[ICESDP] Keranen, A., "Session Description Protocol (SDP) Offer/ Answer procedures for Interactive Connectivity Establishment (ICE)", April 2018, <https://tools.ietf.org/html/draft-ietf-mmusic-ice-sip-sdp>.

[IPHandling] Shieh, G., "WebRTC IP Address Handling Requirements", April 2018, <https://tools.ietf.org/html/draft-ietf-rtcweb-ip-handling>.

[IPLeak] "IP/DNS Detect", n.d., <https://ipleak.net>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/info/rfc4122>.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, DOI 10.17487/RFC5766, April 2010, <https://www.rfc-editor.org/info/rfc5766>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <https://www.rfc-editor.org/info/rfc6762>.

[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>.

[WebRTCSpec] Bruaroey, J., "The WebRTC specification", n.d., <https://w3c.github.io/webrtc-pc/>.

Authors' Addresses

Youenn Fablet
Apple Inc.

Email: youenn@apple.com

Jeroen de Borst
Google

Email: jeroendb@google.com

Justin Uberti
Google

Email: juberti@google.com

Qingsi Wang
Google

Email: qingsi@google.com