Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 8, 2017

C. Jennings
Cisco

July 7, 2016

ICE and STUN Timing Experiments for WebRTC
draft-jennings-ice-rtcweb-timing-00

Abstract

This draft summarizes the results in some experiments looking at the impact of proposed changes to ICE based on the latest consumer NATs. It looks and the amount of non congestion controlled bandwidth a browser can use and the impacts of that on ICE timing.

This draft is not meant to become an RFC. It is purely information to help guide development of other specifications.

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 January 8, 2017.

Table of Contents

1. Introduction

The ICE WG at IETF has been considering speeding up the rate of starting new STUN and TURN connections when doing ICE. The two primary questions that have been raised about this are 1) can the NATs create new connections fast enough to not cause problems 2) what will the impact on bandwidth usage be.

2. Background

A web page using WebRTC can form multiple PeerConnections. Each one of these starts an ICE process that initiates STUN transactions towards various IP and ports that are specified by the Javascript of the web page.

Browser do not limit the number of PeerConnections but do limit the total amount of STUN traffic that is sent with no congestion control. This draft assumes that browsers will limit this traffic to 250kbps thought right now implementation seems to exceed that when measured over an 100ms window.

Each PeerConnection starts a new STUN transaction periodically until all the iCE testing is done. [RFC5245] limits this to be 20ms or

more while [I-D.ietf-ice-rfc5245bis] proposes moves the minimum time to 5 ms. Retransmission for previous stun transaction can be happening in parallel with this.

The STUN specification [RFC5389] specifies 7 retransmission each one doubling in timeout starting with a 500ms retransmission time unless certain conditions are meant. This was put in the RFC to meet the requirements of the IESG from long ago and is largely ignored by existing implementations. Instead system do several retransmissions (6 for Firefox, 8 for Chrome) with a retransmission time starting at 100ms and doubling every retransmission. Often there is a limit of how the maximum retransmission time. Chrome for example only doubles the retransmission time up to a limit of 1600 ms for chrome where it stop increasing the time between retransmissions.

The size of STUN packets can vary based on a variety of options selected but the packets being used by browser today for IPv4 are about 70 bytes for the STUN requests. (Note: some other drafts have significantly higher numbers for this size so some investigation is likely needed to determine what the correct number is)

As the speed of the pacing is speeded up to 5ms, it increases the number of new mappings the NAT needs to create as well as increasing the non congestion controlled bandwidth used by by the browser. The rest of this draft looks at what sort of issue may or may not come out of this.

Additional information about ICE in WebRTC can be found in [I-D.thomson-mmusic-ice-webrtc].

2.1. A multi PC use case

A common design for small conferences is to have a full mesh of media formed between all participants where each participants sends their audio and video to all other participants who mix and render the results. If there are 9 people on a conference call and a 10th one joins, one design might be for the new person to in parallel form 9 new PeerConnections - one to each existing participant.

This might result in 9 ICE agent each starting a new STUN transaction every 5 ms. Assuming no retransmissions, that is a new NAT mapping every 5ms / 9 ICE agent = 0.5 ms and about 5 ms / 9 ICE agent * 70 bytes / packet * 8 bits per byte which comes to about 1 mbps. As many of the ICE candidates are expected not to work, they will result in the full series of retransmitting which will up the bandwidth usage significantly. The browser would rate limit this traffic by dropping some of it.

An alternative design would be to form these connection to the 9 people in the conference sequentially. Given the bandwidth limitations and other issues, later parts of this draft propose that if we move the pacing to 5ms, the WebRTC drafts probably need to caution developers that parallel implementation with these many peers are likely to have failures.

With the current timings, doing this in parallel often works and there are applications that do it in parallel that ill likely need to change if the timing change.

3. Nat Connection Rate Results

The first set of tests are concerned with how many new mappings the NAT can create. The 20ms limit in [RFC5389] was based on going faster than than exceeded the rate of which NATs widely deployed at that time could create new mappings.

The test for this draft were run on the very latest models NATs from Asus, DLink, Netgear, and Linksys. These four vendors were selected due to the large market share they represent. This is not at all representative of what is actually deployed in the field today but represents what we will be seeing widely deployed in the next 3 to 7 years as this generation of NATs moves into the marketplace as well as the lower end NATs in the product lines. It is also clear that in some geographies, a national broadband provider may use some globally less common NAT causing that vendors NAT to prevalent in a given country even if it is not common world wide.

Test were only run using wired interfaces and consisted of connecting both sides of the NAT to two different interfaces on the same computer and using a single program to send packet various direction as well as measure the exact arrival times of packets. Key results were verified using Wireshark to look wire captures made on a separate computer. The first test was normal tests made to classify the type of the NAT for the cases when 1, 2, and 3 internal clients all have the same source port. The second test created many new mappings to measure the maximum rate mapping could reliably be made.

The conclusion of the first test was that all of the NATs tested were behave complaint (for UDP) with [RFC4787] with regards to mapping and filtering allocations. This is great news as well as a strong endorsement on the success of the BEHAVE WG. The fact that we see a non trivial percentage of non behave compliant NATs deployed in the field does highlight that this sample set of NATs tested is not a representative sample of what is deployed. It does suggest that we should see a reduced use of TURN servers over time.

On the second test, all the NATs tested could reliably create new mapping in under 1ms - often more like several hundred micro seconds. The NATs do drop packets if the rate of new mapping gets too high but for all the NATs tested, this rate was faster than 1000 mappings per second. Looking at the code of one NAT, this largely seems to be due to large increase in clock speed of the CPUs in the NATs tested here vs the speed in the NATs tested in 2005 in [I-D.jennings-behave-test-results].

This implies that as long as there or less than 5 or 10 PC doing ICE in parallel in a given browser, we do not anticipate problems on the texted NATs moving the ICE pacing to 5ms.

4. ICE Bandwidth Usage

4.1. History of RFC 5389

At the time [RFC5389] was done, the argument made was it was OK for STUN to use as much non congestion controlled bandwidth as RTP audio was likely to do as the STUN was merely setting up a connection for an RTP phone call. The premise was the networks that IP Phones were used on were designed to have enough bandwidth to reasonable work with the audio codecs being used and that the RTP audio was not elastic and not congestion controlled in most implementations. There was a form of "User congestion control" in that if your phone call sounded like crap because it was having 10% packet loss, the user ended the call, tried again, and if it was till bad gave up and stopped causing congestion.

Since that time the number of candidates used in ICE has significantly increased, the range of networks ICE is used over has expanded, and uses have increased. We have also seem much more widespread use of FEC that that allows high packet loss rate with no impact on the end user perception of media quality. In WebRC there applications such as file sharing and background P2P backup that form data channel connecting using ICE with no human interaction to stop if the packet loss rate is high. ICE in practical usage has expanded beyond a tool for IP phones to become the preferred tool on the internet for setting up end to end connection.

4.2. Bandwidth Usage

To prevent things like DDOS attacks on DNS servers, WebRTC browser limit the non congestion controlled bandwidth of STUN transaction to an unspecified number but seems that browsers currently plan to set this to 250 kbps. An advertisement running on a popular webpage can create as many PeerConnections as it wants and specify the IP and port to send all the STUN transaction to. Each Peer Connection

objects sends UDP traffic to an IP and port of specified in the JavaScript which the browser limits by dropping packets that exceed the global limit for the browser.

It seems that the current plans for major browsers would allow the browser to 250 kbps of UDP traffic when there was 100% packet loss. Currently they send more than this. As far as I can tell there is specification defining what this limit should be in this case.

4.3. What should global rate limit be

It is clear that sending 250 kbps on 80 kbps edge cellular connection severely impacts other application on that connection and is not even remotely close to TCP friendly. In the age of cellular wifi hot spots and highly variable backhaul, the browser has very little idea of what the available bandwidth is.

This draft is not in anyway suggesting what the bandwidth limit should be but it is looking at what are the implication to ICE timing based on that number. The limit has security implication in that browser loading Javascript in paid advertisements on popular web sides could use this to send traffic to DDOS an server. The limit has transport implication in how it interacts with other traffic on the networks that are close to or less than this limit.

More information on this topic can be found in [I-D.ietf-tsvwg-rfc5405bis] and [I-D.ietf-avtcore-rtp-circuit-breakers].

4.4. Rate Limits

Having a global bandwidth limit for the browser, which if exceeded will drop packets, means that applications need to stay under this rate limit or the loss of STUN packets will cause ICE to start mistakenly thinking there is no connectivity on flows which do not work. Consider the case with two NICs (cellular and wifi), each with an v4 and v6 address, and a reachable TURN server on each. This gives 12 candidates and if the other side is the same there are six v6 addresses matching on the other side so 36 pairs for v6 and the same for v4 resulting in 72 pairs for ICE to check (assuming full bundle, RTCP mux etc). The number of pairs we will see in practice in the future is a somewhat controversial topic and the 72 here was a number pulled out of a hat and not based on any real tests. There is probably a better number to use.

A simple simulation on this where none of the connections works suggests that the peak bandwidth in 100ms windows is about 112kbps if the pacing is 20 ms while it goes to about 290kbps if the pacing is 5

ms. This is the bandwidth used by a single ICE agent and there could easily be multiple ICE agents running at the same time in the same tab or across different tabs in the browser.

The point I am trying to get at with this is that if the global rate limit would need to be much higher than 250 kbps to move to a 5 ms pacing and have it reliably work with multiple things happening at the same time in the browser.

4.5. ICE Synchronization

NATs and firewalls create very short windows of time where a response to an outbound request is allowed to in and allowed to create a new flow. Though this draft did not test these timing on major firewalls, some information indicates these windows are being reduced as time goes on to possible provide better a short attack window for certain types of attacks. ICE takes advantage of both one side sending a suicide packet that will be lost but will create a short window of time where if the other side sends a packet it will get in a the window created by the suicide packet and allow a full connection to form. To make this work, the timing of the packets from either side needs to be closely coordinated. Most the complexity of the ICE algorithm comes from trying to coordinate both sides such that they send the related packets at similar times.

A key implication of this is that if several ICE agent are running in single browser, what is happening in other ICE agent can't change the timing of what a given ICE agent is sending. Or at least the amount of skew introduced can't cause the packets to fall outside the timing widows from the NATs and Firewalls. So any solution that slowed down the transmission in one Peer Connection if there were lots of other simultaneous Peer Connection may have issues unless the far side also knows to slow down.

Figuring out how much the timing can be skewed between the two sides requires measuring how long the window is open on the NATs and firewalls. Currently we do not have good measurements of this timing and it is not possible to evaluate how much this is an issue without that information.

5. Recommendations

The ICE and RTCWeb Transport documents should specify a clear upper bound on the amount of non congestion controlled traffic an browser or applications should be limited to. The transport and perhaps security area should provide advice on what that number should be. WebRTC basically application work better the larger that number is at

the expense of other applications running on the same congested links.

There is no way for a JavaScript application to know how many other web pages or tabs in the browser are also doing stun yet all of these impact the global rate limit in the browser. If the browser discards STUN packets due to the global rate limit being exceeded, it results in applicant failures that look like network problems which are in fact just an artifact of other applications running the browser at the same time. This is critical information to understanding why applications are failing. The recommendation here is that the WebRTC API be extended to provide a way for the browsers to inform the application using a given PeerConnection object if STUN packets that PeerConnection is sending are being discarded by the browser.

6. Future work

It would be nice to collect measurements on how long NATs and Firewalls keep mapping with no response open. It would be nice to simulate how much global pacing would introduce skew the timing of ICE packets and if that would reduce non relay connectivity success rates.

7. Conclusions

The combination of a low ICE pace timing, lots of Peer Connections, and many candidates will cause problems. The optimal way to balance this depends on the factors such as what how much non congestion controlled bandwidth we should assume is available.

The speed of NATs mapping creation going forward in the future is likely adequate to move the pacing to 5ms. However applications that create parallel peer connections or situations where more than a handful of PeerConnections are forming in parallel in the same browser (possibly in different tabs or web pages) need to be avoided.

From a bandwidth limit point of view, if the bandwidth is limited at 250 kbps, a 5ms timing will work for a single PeerConnection but not much more than that. The specification should make developers aware of this limitation. If the non congestion controlled bandwidth limit is less than 250 kbps, a 5ms timing is likely too small to work reliably particularly with multiple ICE agents running in the browser.

8. Acknowledgments

Many thanks to review from Eric Rescorla for review and simple simulator and to Ari Keraenen, and Harald Alvestrand.

9. References

9.1. Normative References

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, DOI 10.17487/RFC5245, April 2010, <http://www.rfc-editor.org/info/rfc5245>.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <http://www.rfc-editor.org/info/rfc5389>.

9.2. Informative References

[I-D.ietf-avtcore-rtp-circuit-breakers]
Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", draft-ietf-avtcore-rtp-circuit-breakers-16 (work in progress), June 2016.

[I-D.ietf-ice-rfc5245bis]
Keraenen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", draft-ietf-ice-rfc5245bis-04 (work in progress), June 2016.

[I-D.ietf-tsvwg-rfc5405bis]
Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", draft-ietf-tsvwg-rfc5405bis-15 (work in progress), June 2016.

[I-D.jennings-behave-test-results]
Jennings, C., "NAT Classification Test Results", draft-jennings-behave-test-results-04 (work in progress), July 2007.

[I-D.thomson-mmusic-ice-webrtc]
Thomson, M., "Using Interactive Connectivity Establishment (ICE) in Web Real-Time Communications (WebRTC)", draft-thomson-mmusic-ice-webrtc-01 (work in progress), October 2013.

[RFC4787] Audet, F., Ed. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, DOI 10.17487/RFC4787, January 2007, <http://www.rfc-editor.org/info/rfc4787>.

Appendix A. Appendix A - Bandwidth testing

The following example web page was used to measure how much bandwidth a browser will send to an arbitrary IP and port when getting 100% packet loss to that destination. It creates 100 Peer Connections that all send STUN traffic to port 10053 at 10.1.2.3. It them creates a single data channel for each one and starts the ICE machine by creating an offer setting that to be the local SDP.

```
<!DOCTYPE html>
<html>

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <meta charset="utf-8">
    <title> STUN traffic demo </title>
</head>
<body>
    <h2> This is STUN traffic demo </h2>
    <p>
        grab traffic with:
        sudo tcpdump -i en1 -s 0 -w /tmp/dump.pcap 'port 10053'
        </p>

    <script>
        var pc = new Array(100);
        var i = 0;

        function setupPC(lpc) {
            lpc.createDataChannel("myData");
            lpc.createOffer().then(function(offer) {
                return lpc.setLocalDescription(offer);
            });
        }

        var configuration = {
            iceServers: [{
                urls: 'stun:10.1.2.3:10053'
            }]
        };
        for (i = 0; i < pc.length; i += 1) {
            if (navigator.mozGetUserMedia) {
                pc[i] = new RTCPeerConnection(configuration);
            } else { // assume it is chrome
                pc[i] = new webkitRTCPeerConnection(configuration);
            }

            setupPC(pc[i]);
        }
    </script>

</body>

</html>
```

Author's Address

Cullen Jennings
Cisco

Email: fluffy@iii.ca