WebRTC, STUN and TURN

For Puzzel Digital Engagement

Establishing high quality peer-to-peer video conferencing

puzzel.

1. Document History

Version:	Date of version:	Created by:	Note:
1.0	2017-05-19	Stefan Möhl	Created
1.1	2019-08-26	Stefan Möhl	Added illustrations
1.2	2019-09-14	Stefan Möhl	New IPs for Falkenberg/Stockholm
1.3	2020-09-21	Stefan Möhl	How Puzzel Digital Engagement connects with WebRTC

Contents

1. Document History 2

3. Introduction 6

3.1 Background 6

3.1.1 Illustration 1: Peer to peer communication 6

4. Web RTC 7

4.1 Establishing a connection 7

4.1.1 Illustration 2: WebRTC sets up the peer-to-peer transfer 7

4.2 NAT and STUN 7

4.2.1 Illustration 3: STUN sees public IP number and port 8

4.2.2 Illustration 4: STUN sees public IP number and port 8

4.3 Enterprise Level Firewalls 9

4.4 When all else fails: TURN-server 9

4.4.1 Illustration 5: Communicating via a TURN-server 9

4.5 Some security measures are just too restrictive 10

5. Puzzel Digital Engagement and WebRTC 11

5.1.1 Illustration 6: Puzzel Digital Engagement and WebRTC 12

6. Ports and transports for Puzzel Digital Engagement STUN/TURN 13

6.1.1 Illustration 7: Puzzel Digital Engagement and WebRTC 14

3. Introduction

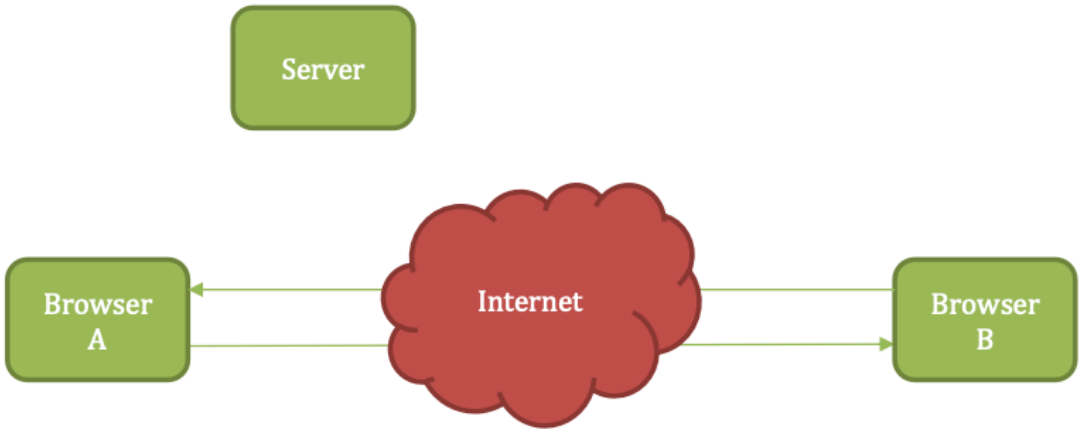
3.1 Background

WebRTC is the current standard in commonly used web browsers for real-time audio and video communication. Since real-time video conferencing requires high bandwidth (for the video signal) and low latency (to avoid awkward communication pauses) there are some important differences between WebRTC and traditional Web technologies.

The biggest difference is that WebRTC is not based on a client-server model as is usual for web pages. Instead, the preferred method is direct peer-to-peer data transfer. This means that the web browsers for both participants connect directly to each other. A server is used only to set up the connection but once meta-data has been exchanged, data is sent directly from each browser to the other.

The other main difference to traditional web technologies is that communication is done via UDP rather than TCP. This is again due to the nature of audio and video, where the human senses are much more sensitive to hiccups and variations in speed than they are to omissions. This makes waiting for retransmission of a dropped TCP packet much more disruptive than simply omitting the dropped package, and so UDP is preferred.

3.1.1 Illustration 1: Peer to peer communication

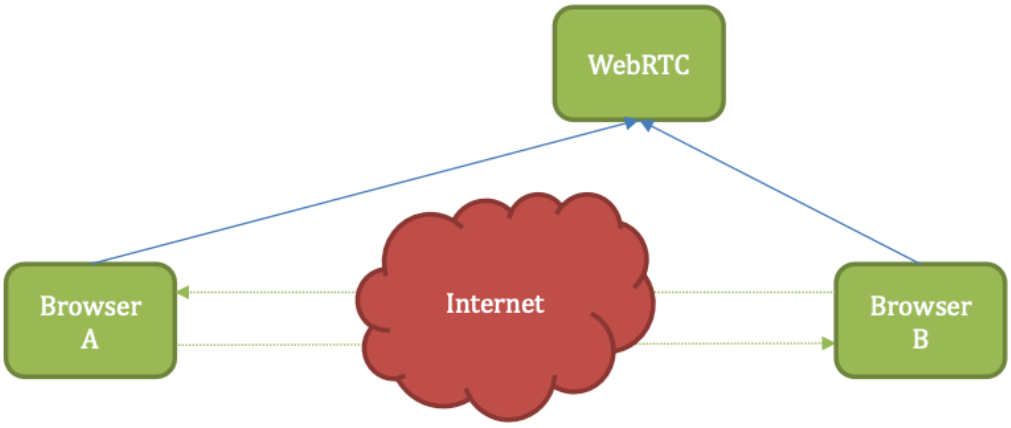


4. Web RTC

4.1 Establishing a connection

To establish direct data transfer from one peer to another, the peers need to be aware of each other to start with. This is done by a WebRTC signalling server. The signalling server is a normal web server that both peers connect to for session authorization and initiation. Among other data related to the data stream, the peers exchange their IP addresses and port numbers via the WebRTC server, enabling them to connect directly to each other.

4.1.1 Illustration 2: WebRTC sets up the peer-to-peer transfer



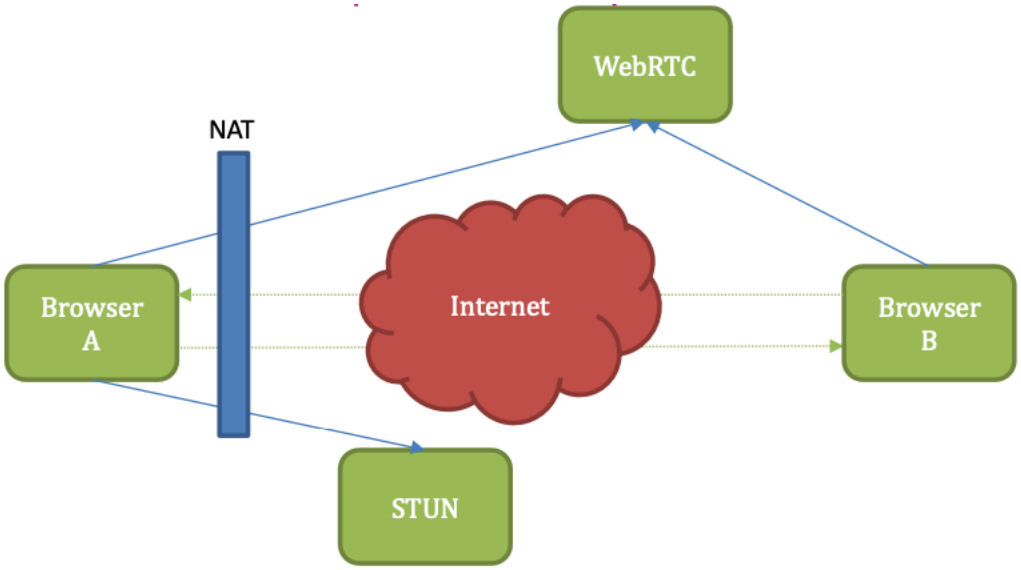
4.2 NAT and STUN

Almost all users under IPv4 are located behind a NAT, so two participants at different locations usually cannot establish a peer-to-peer connection directly. The IP address visible from within the local machine is only the local network IP and not the public IP address and port number usable by the remote partner.

To find the public IP address and port number for a peer, a STUN server is used. This is a simple server on the public Internet that essentially responds back with the external IP address and port numbers that are visible to it. The peer behind a NAT can then let the other

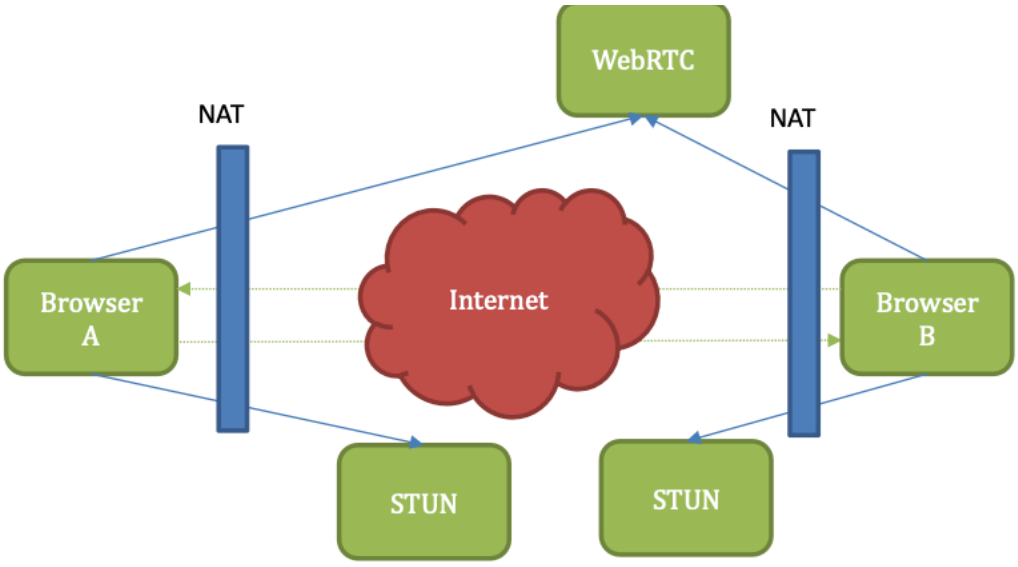
peer know about its public IP address and port number through the WebRTC signaling server.

4.2.1 Illustration 3: STUN sees public IP number and port



Though the NAT is only shown on one side here, the other side is usually also behind a NAT, and uses a STUN server (possibly the same one) in the same way.

4.2.2 Illustration 4: STUN sees public IP number and port



4.3 Enterprise Level Firewalls

For home networks, STUN is usually sufficient to traverse the NAT used for Internet connectivity. Enterprise firewalls are often more restrictive.

For normal TCP connections, a bidirectional session is established that lets both the client and the server send data to each other. The server is typically exposed on a well-known port number, letting the client send its request. The server response is usually sent to a random port number given by the client, conventionally a high-numbered port (also known as dynamic port or ephemeral port). IANA suggests the port range 49152 to 65535 for high-numbered ports.

Enterprise level firewalls are typically stateful (SPI) firewalls. This allows an administrator to close all ports except the ones that are specifically exposed for the public services on the local network, and the firewall inspects TCP traffic to deduce what high-numbered port to temporarily open for the response.

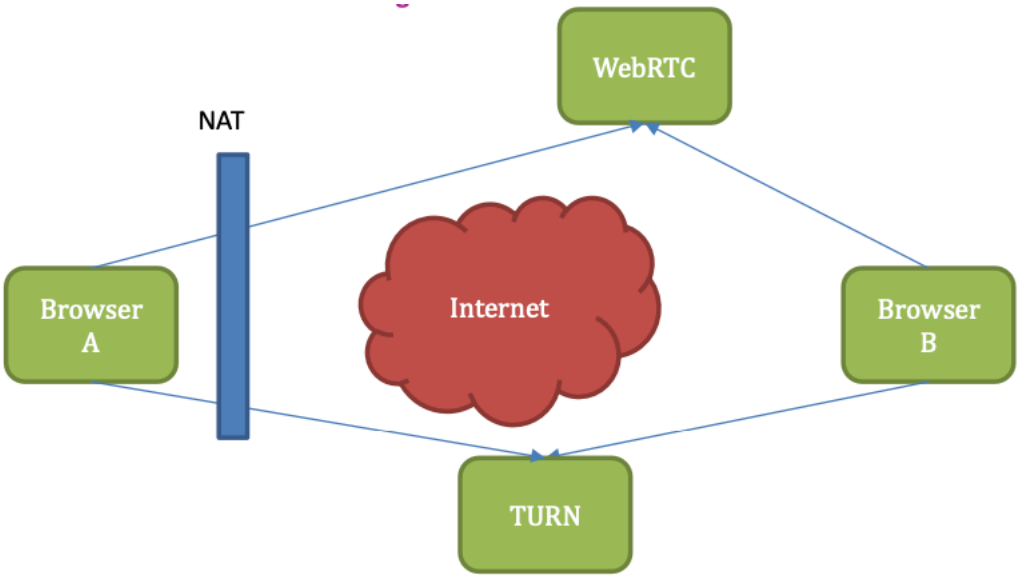
Since WebRTC uses UDP, the situation is different. UDP is session-less and data is transmitted unidirectionally without expecting a response. Thus, the firewall will usually not open a port for UDP packets from the other side. However, a modern firewall will be aware of STUN. The implementation is vendor-specific, e.g. Cisco calls it STUN Inspection, Juniper calls it Persistent NAT, and other vendors have similar features. These give roughly the same functionality to UDP with STUN as TCP does, temporarily opening UDP ports as needed.

If the firewall cannot be made aware of STUN, the best option is to open all UDP ports in the range 49152 to 65535 for both inbound and outbound traffic or to upgrade the firewall.

4.4 When all else fails: TURN-server

If no method for opening a direct peer-to-peer connection can be used, the final option is to send the audio and video traffic to a dedicated TURN server. This entails sending the audio and video traffic from one client to the TURN server, sending on a copy of the same traffic from the TURN server to the other client and vice versa for the other direction. Unfortunately, this unavoidably introduces latency since at least parts of the data stream will be buffered in the TURN server, resulting in “long distance phone call” problems where speakers speak over each other. It might also result in pops and crackles in the audio stream due to random latencies being introduced, as well as artifacts and deformations in the video image. Further, the significant bandwidth used by video streams to and from the TURN server, as well as the TURN server itself introduce extra costs. The advantage is that an Enterprise firewall only has to open ports to one or just a few (for redundancy) remote hosts for the audio and video data.

4.4.1 Illustration 5: Communicating via a TURN-server



4.5 Some security measures are just too restrictive

Finally, there are some situations where security measures simply do not permit any form of real-time connection. For instance, this happens when the security policy requires a web-proxy to be used at all times. A web-proxy is a server that fetches individual web pages on behalf of the user, meaning that the user never connects to the other side at all; the web-proxy establishes all connections on behalf of the user. Unless the web-proxy has a special mode to allow Web Sockets and WebRTC, a traditional web-proxy will simply not let the data streams through at all. Unfortunately, since audio and video are bi-directional, both sides need to have a data stream, so it might very well be the other side that uses a web proxy or other such restrictive security measures, making it impossible to solve without an agreement between the two parties about opening security measures.

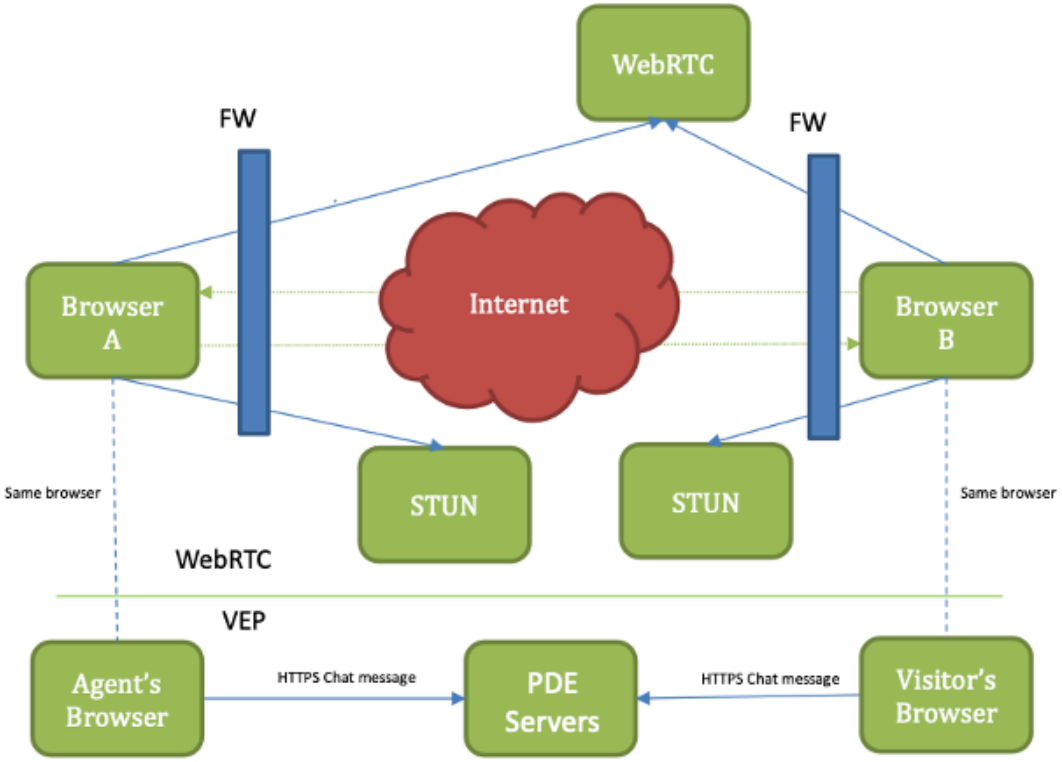
5. Puzzel Digital Engagement and WebRTC

WebRTC is a common public technology in modern web browsers. It has been integrated into Puzzel Digital Engagement to allow both voice and video calls through Puzzel Digital Engagement without the need for any party to install applications or plugins. Voice and video can be used in Puzzel Digital Engagement with both parties using only a standard web browser.

The way WebRTC is used by Puzzel Digital Engagement is similar to how Microsoft Skype or Teams set up a call using e-mail. In a Skype call, an e-mail is sent to the other party to set up the call through Skype, so, though e-mail is used, Skype is not a part of e-mail. Likewise for a Puzzel Digital Engagement audio or video call, a chat message is sent to the other party to set up the call through WebRTC. This means that the involvement of the Puzzel Digital Engagement servers is mainly limited to passing along a special chat message inviting both clients to initiate a WebRTC channel setup. This chat message is passed along the normal Puzzel Digital Engagement chat between the agent and visitor and nothing special from the point of view of Puzzel Digital Engagement. Once the WebRTC session has been established, the two browsers communicate directly, peer-to-peer, without any involvement from Puzzel Digital Engagement.

In those cases when fall-back to a TURN server is required, Puzzel Digital Engagement also establishes access tokens for the TURN server that are sent to the Puzzel Digital Engagement users, allowing us to prevent unrelated Internet users from spending bandwidth on the TURN server.

5.1.1 Illustration 6: Puzzel Digital Engagement and WebRTC



6. Ports and transports for Puzzel Digital Engagement STUN/TURN

Transport	Port	Target	Service	Direction
TCP+UDP	80	194.54.166.44, 194.54.166.45, 194.54.166.46, 194.54.166.47, 46.21.96.236, 46.21.96.237, 46.21.96.238, 46.21.96.239	STUN/TURN + WebRTC Signaling server	Outgoing
TCP+UDP	81	194.54.166.44, 194.54.166.45, 194.54.166.46, 194.54.166.47, 46.21.96.236, 46.21.96.237, 46.21.96.238, 46.21.96.239	STUN RFC-5780	Outgoing
TCP+UDP	443	194.54.166.44, 194.54.166.45, 194.54.166.46, 194.54.166.47, 46.21.96.236, 46.21.96.237, 46.21.96.238, 46.21.96.239	STUN/TURN over TLS + WebRTC Signaling Server	Outgoing
TCP+UDP	444	194.54.166.44, 194.54.166.45, 194.54.166.46, 194.54.166.47, 46.21.96.236,	STUN over TLS RFC-5780	Outgoing

		46.21.96.237, 46.21.96.238, 46.21.96.239		
UDP	49152-65535	ANY	WebRTC dynamic ports	Incoming
UDP	ALL	ANY	WebRTC dynamic ports	Outgoing

Each STUN/TURN server requires two public IP addresses for NAT traversal, and two servers are used for redundancy per datacenter. There are two datacenters. UDP is preferable to TCP (see 3.1 Background).

6.1.1 Illustration 7: Puzzel Digital Engagement and WebRTC