DERECHOS
DIGITALES
América Latina

IT’S NOT MAGIC

Interfaces and protocols for video calls

Martu Isla
cacu
la_jes
Juliana Guerra

Written by Martu Isla, Juliana Guerra, la_jes (Sursiendo), cacu (Tierra Común Cooperativa).

Cover and publication design by Constanza Figueroa.
English Translation by Selene Cruz.
Edited by Vladimir Garay.

This publication was made possible with the support of Article 19.

i. In isolation, can we stay close?

As a result of the Covid-19 pandemic, in recent months we have faced the promise on with internet was built. From one moment to the next, much of the world turned to digital environments. In many cities around the world, only the “essential” activities of care, cleaning and surveillance remained in place, while a large part of the population was forced into confinement.

In addition to the imminent crisis generated by the suspension of economic activities, in regions such as Latin America and the Caribbean, there was also evidence of the persistent gap in internet access, in its multiple dimensions. The figures are very unequal between countries, but also internally, between urban and rural contexts. But what does it mean to get access to the internet? Perhaps, it means to have access to a device, a tablet or a cell phone, even if it is not the same as a computer, and it is not the same to connect to fixed or mobile broadband, also because the cost is not the same.

The cost depends on the infrastructure; for instance, in poorer countries it is more expensive to connect and connections are slower. Hence, it relies on how much infrastructure is available and how robust it is, which in turn also depends on the materials that are used to make the connection and the technologies that run devices. But this is only one dimension of access: once we can connect, our technical, linguistic, and cultural skills also determine our ability to ‘navigate’.

But for now, let’s go back to the promise of the internet.

In 1989, at the European Center for Nuclear Research (CERN) a tool for collaboration and information exchange was proposed to become the World Wide Web[1], and materialized in the HTTP protocol (Hypertext Transfer Protocol) that we use today to navigate the internet. The proposal consisted of a distributed hypertext system, legible for people and where information was connected in an unlimited way, not based on a fixed hierarchical system.

From the beginning the Web was designed to organize and streamline the work of a specific scientific community. It was conceived as a universal interconnected information system, supporting different platforms and extensible to new data formats. With that purpose, between 1993 and 1994, it began to be an attractive product for the market and little by little, it was entering government offices, companies and homes, in an expansive process that has continued until today.

Almost ten years later, the transmission of video over the internet greatly increased the traffic of digital information, while the development of 3G technology in mobile telephony, which allowed connection to some internet services, skyrocketed the level of connections. Nowadays, we not only connect to work, learn or do administrative procedures, our emotions and

[1] Tim Berners-Lee. CERN. 1989-1990. Information Management: A Proposal. https://www.w3.org/History/1989/proposal.html

feelings are also connected. Thus, in a pandemic, our circles of affection, trust, and political organization are necessarily mediated by internet technologies.

Collaborative editing, file sharing, and audio and video streaming are perhaps the most useful digital tools in times of confinement, but why do video calls and video conferencing became so popular in recent months? Despite consuming many resources and that communication is often not fluent or understandable, we choose to see each other: in class; in meetings with few or many people; in presentations and workshops; in the parties; in sex.

Beyond the reasons that lead us to prefer real-time audio and video tools, or commercial alternatives and their characteristics in terms of quality, security or privacy, in this document we want to understand how this communication is technically possible, and we wonder if access to video calls and video conferencing is universal. In other words, what does it depends on being able to use these services optimally?

ii. Contact Limitations

When the physical distancing measures began to stop the contagion curve for Covid-19, from many places we demanded to strengthen the social meeting, and there was the internet to satisfy us. According to the ECLAC[2], during the first half of 2020, the consumption of broadband communication services in Latin America and the Caribbean increased dramatically: the use of teleworking solutions increased 324%, distance learning 62%, and E-commerce and delivery services 157%.

This meant an increase in traffic and greater demands on capacity and resilience for the networks operating in the region, although the potential for connectivity remains quite limited. ECLAC says that in order to guarantee effective participation in digital environments in the region, including access to health, education and work, as well as to shopping, banking and entertainment services, it is first necessary to expand fixed broadband coverage and improve mobile broadband connection speed. In addition, for the provision of online health services, it calls the attention the need to guarantee access to digitized medical information and interoperability of services, as well as data privacy and security.

Nonetheless, beyond access to digital goods and services, or the figures on the quality of connection to fixed or mobile broadband, in confinement it has become evident how in digital environments we have experiences and initiate relationships in which we are necessarily embodying multiple identities and realities. This is recognized by the Feminist Principles

[2] The Economic Commission for Latin America, ECLA. 2020. Universalizing access to digital technologies to address the consequences of COVID-19. https://repositorio.cepal.org/bitstream/handle/11362/45938/4/S2000550_es.pdf

for the Internet[3], which for several years have called for "universal, acceptable, affordable, unconditional, open, meaningful and equal" access, especially for women and queer people.

The possibility of accessing the internet is crossed by multiple dimensions and while the industry seeks profitable solutions to connect the other half of the world's population, it is-rapidly advancing towards cutting-edge technologies, which are increasingly complex and require better infrastructures to function optimally. In point of fact, for this reason, it is imperative to work so expansion of coverage is done with criteria of quality and dignity for users, since it is about connecting communities that have traditionally been marginalized and subjected to different types of violence.

During the first months of 2020, different organizations published guides to lead the proper use of video calling platforms and applications, some addressed to wide audiences,[4] others at critical groups such as journalists,[5] school teachers[6] or activists[7]. Security and privacy analysts turned their eyes to the most popular platforms, and many of these had to update their policies, designs and settings, to respond to the needs of the moment.

Zoom's case is paradigmatic. This company located in Silicon Valley, since 2013 was trying to position itself as a competition against Google, Apple or Microsoft, offering a simple and friendly interface, while guaranteeing a stable transmission of audio and video. As early lockdown measures began to take effect, Zoom became the most popular video conferencing option in businesses, State entities and schools. Thus, it went from 10 million participants per day in December 2019, to 300 million in April 2020.[8]

As early as March, a series of criticisms began to be published regarding the vulnerabilities in the platform and the misleading discourse with which it was advertised. Already in 2019, it was denounced its ability to "bypass browser security settings and remotely enable a user's

[3] Feminist Principles of the Internet. Statements that offer a gender and sexual rights perspective on critical Internet-related rights. 2014-2015. https://feministinternet.org/en

[4] In English https://foundation.mozilla.org/en/privacynotincluded/categories/video-call-apps/ y https://videoconferencing.guide/, among other sources. In Spanish, and for Latin America.

[5] Choosing the right video conferencing tool for the job. https://freedom.press/training/blog/videoconferencing-tools/

[6] Protecting Students in Virtual Classrooms: Considerations for Educators. https://cdt.org/insights/protecting-students-in-virtual-classrooms-considerations-for-educators/

[7] Guide on safe tools for conferences and group chats https://www.frontlinedefenders.org/es/resource-publication/guide-secure-group-chat-and-conferencing-tools

[8] Data published on the blog. Available on: https://blog.zoom.us/a-message-to-our-users/ y https://blog.zoom.us/90-day-security-plan-progress-report-april-22/

web camera without the knowledge or consent of the user"[9], to what it was added criticism for the ***attention tracking***; which allowed the hostess see if any attendees do not have the desktop client or mobile app in focus for more than 30 seconds.[10]

Alerts were also raised for the so-called *Zoom Bombing*,[11] for the data that the platform sent to Facebook to notify when someone opened the application,[12] and for the data filtering of those who subscribed with email accounts in other servers different than the most popular ones like Gmail, Hotmail or Yahoo.[13] Then, it came analysis on the pre-installation mechanisms implemented in macOS,[14] the implementation of "what the company calls end-to-end encryption",[15] its routing alternatives, using servers in China since the pandemic began,[16] and a vulnerability in the waiting room of a meeting.[17]

As explained by The Intercept at the end of March,[18] until that moment in Zoom, only the connection between the client and the platform was encrypted, in the same way as navigation on a website that has HTTPS is encrypted. The communication was not encrypted end-to-end (E2E) but only the chat, that is, text messages. According to the report by CitizenLab,[19] Zoom implemented its own transport protocol, with some modifications over the existing RTP (Real-Time Transport Protocol) standard, and all media traffic was encrypted and de-

[9] EPIC Files Complaint with FTC about Zoom https://epic.org/2019/07/epic-files-complaint-with-ftc-.html

[10] Working From Home? Zoom Tells Your Boss If You're Not Paying Attention https://www.vice.com/en_us/article/qjdnmm/working-from-home-zoom-tells-your-boss-if-youre-not-paying-attention

[11] Beware of 'ZoomBombing': screensharing filth to video calls https://techcrunch.com/2020/03/17/zoombombing/

[12] Zoom iOS App Sends Data to Facebook Even if You Don't Have a Facebook Account https://www.vice.com/en_us/article/k7e599/zoom-ios-app-sends-data-to-facebook-even-if-you-dont-have-a-facebook-account. This feature was quickly removed according to the same source. https://www.vice.com/en_us/article/z3b745/zoom-removes-code-that-sends-data-to-facebook

[13] Zoom is Leaking Peoples' Email Addresses and Photos to Strangers https://www.vice.com/en_us/article/k7e95m/zoom-leaking-email-addresses-photos

[14] https://twitter.com/c1truz_/status/1244737675191619584

[15] Move Fast and Roll Your Own Crypto. A Quick Look at the Confidentiality of Zoom Meetings https://citizenlab.ca/2020/04/move-fast-roll-your-own-crypto-a-quick-look-at-the-confidentiality-of-zoom-meetings/

[16] Zoom admits some calls were routed through China by mistake https://techcrunch.com/2020/04/03/zoom-calls-routed-china/

[17] Zoom's Waiting Room Vulnerability https://citizenlab.ca/2020/04/zooms-waiting-room-vulnerability/

[18] Zoom Meetings Aren't End-to-End Encrypted, Despite Misleading Marketing https://theintercept.com/2020/03/31/zoom-meeting-encryption/

[19] Move Fast and Roll Your Own Crypto... https://citizenlab.ca/2020/04/move-fast-roll-your-own-crypto-a-quick-look-at-the-confidentiality-of-zoom-meetings/

crypted with a single AES-128 key (Advanced Encryption Standard-128 bits), generated and distributed by the platform's server to the participants, in ECB (Electronic codebook) mode, considered as very weak within existing standards.

In order to be brief on Zoom, it is worth saying that the company made a commitment to the privacy of its users and in April launched a 90-day plan to repair errors and vulnerabilities.[20] However, the payment services of this and other platforms such as Meet (Google), Teams (Microsoft) and Webex (Cisco) continue to offer a better service in terms of quality, stability and privacy.[21] Perhaps, that is why, with the advance of the pandemic, these were the companies that best answered to the institutional demand for video calling and video conferencing services and they are the ones who dominate the market today. But what about the organizations, movements, groups and individuals who cannot afford access to the services offered by the greats of the internet?

Regarding difficulties and risks associated with the increasing use of free digital platforms, some organizations shared recommendations for remote work[22] based on free and privacy-friendly tools, where Jitsi Meet appeared as one of the best options for video calls[23]. Jitsi is an open source project that in 2003 began developing a desktop application for voice and text messaging over the internet. Over the years, it has been implementing different technologies to integrate video and guarantee fluid communication, which does not require so many resources from end customers.[24]

Since its code is open, it is possible to install your own instances and many organizations

[20] CEO Report: 90 Days Done, What's Next for Zoom https://blog.zoom.us/ceo-report-90-days-done-whats-next-for-zoom/

[21] Zoom is Making Privacy and Security a Luxury https://foundation.mozilla.org/en/blog/zoom-making-privacy-and-security-luxury/ and Google Meet acabará con el 'gratis total' de los últimos meses, ¿cuándo?

[22] Conectadas y seguras en tiempos de cuarentena https://blog.torproject.org/Conectadas-seguras-tiempos-cuarentena, Recomendaciones de software libre para usar en contexto de distanciamiento físico (pero no social) https://www.vialibre.org.ar/2020/05/04/recomendaciones-de-software-libre-para-usar-en-contexto-de-distanciamiento-fisico-pero-no-social/, Recomendaciones para una mejor experiencia en línea https://ranchoelectronico.org/recomendaciones-cuarentena/, among others.

[23] Vídeollamadas con Jitsi: la alternativa a las plataformas comerciales https://labekka.red/novedades/2020/04/21/jitsi.html, Alternativas a las reuniones en vivo https://mayfirst.coop/es/post/2020/node-167915/, ¿Qué está pasando con Zoom? https://sursiendo.org/blog/2020/05/que-esta-pasando-con-zoom/

[24] Jitsi User FAQ https://jitsi.org/user-faq/

did so during the pandemic.[25] For instance, in Argentina, it was developed Jitsimeter,[26] a comparison of the quality of the instances and the privacy conditions in which they operate, based on the use of intermediate servers, owned by large companies in the data market such as Amazon, Google or Microsoft. It is important to mention that the infrastructure behind a video call is much more complex than setting up an instance.

ii. The gears of infrastructure

The possibility of communicating with audio and video in real-time is a project that began in the late 1980s, when the internet was a tool to connect computers that could exchange digital information between them, with a mainly military and academic use. But the logic of the internet is changing and not only the web allowed it to become a communication tool worldwide; perhaps the commercial deployment of fiber optics was the most important factor in the exponential growth of the internet, since it allowed transporting increasingly large volumes of traffic, at lower costs compared to copper cables.

Optical fibre allowed not only the transport of data, but also the transmission of high-quality audio and, years later, video.[27] If at the beginning the internet has allowed the exchange of emails, since 2010 most of the traffic on the internet

Although the user base has also grown exponentially, the internet market is in the hands of fewer and fewer companies, which not only develop tools with which we interact on a daily basis (search engines, social media or collaborative work platforms) but also collect, host and process our data, at the same time they develop and standardize the rules with which the infrastructure operates, to ensure that all this information remains available on the internet, as we have become used to practically everything being hosted in “the cloud.”

But the internet is not a cloud, it is not ethereal, it is material and solid. Although the devices with which we connect are increasingly smaller and information travels at very high speeds, it deals with a huge technical and commercial complex. Hence, in times of the pandemic, when an important part of our lives is spent on different screens, and “connecting the other half” is a priority for companies and governments, questions about sovereignty over our information - and about our own autonomy when we interact online – become urgent.

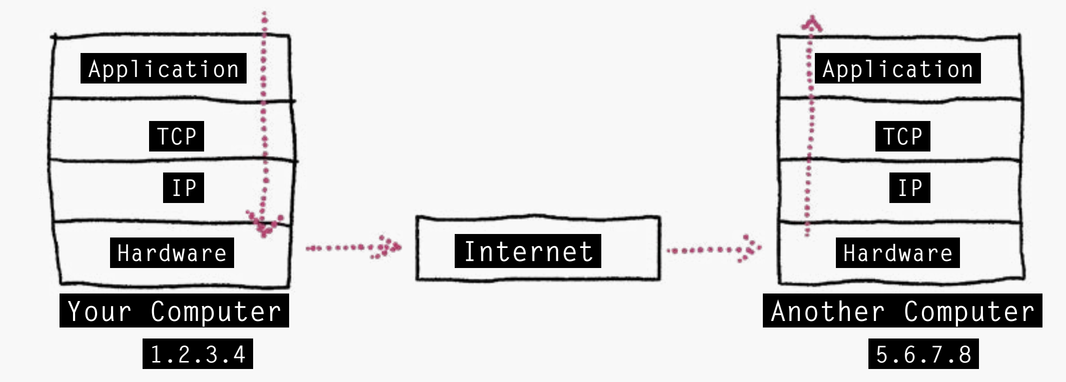
[25] Maadix is an infrastructure provider that offers online work tools while guaranteeing the autonomy, security and privacy of its users. At the beginning of April, they had the service of installing their own Meet Jitsi instances. https://maadix.net/es/instala-jitsi-meet-con-un-clic and published a series of recommendations to optimize its performance. https://maadix.net/es/optimizar-rendimiento-jitsi

[26] Jitsimeter ¿Qué instancia de Jitsi me conviene usar? https://ladatano.partidopirata.com.ar/jitsimeter/

[27] Clark, D. 2018. Designing an Internet. Massachusetts Institute of Technology.

How much do we know about the information about us that is captured and exchanged every time we make a video call? Who owns the networks through which it is transmitted? Who installs, maintains and accesses those networks? These questions can exceed our interests and capacities, if we only want to hold a meeting that cannot be done in person. However, because of the need in which this context puts us, we consider relevant to look beyond the free software options or alternative infrastructures and understand better standards and protocols that govern the operation of the Internet.

Communication between machines



Basado en Shuler, R. 2018. How Does the Internet Work? Standford University.

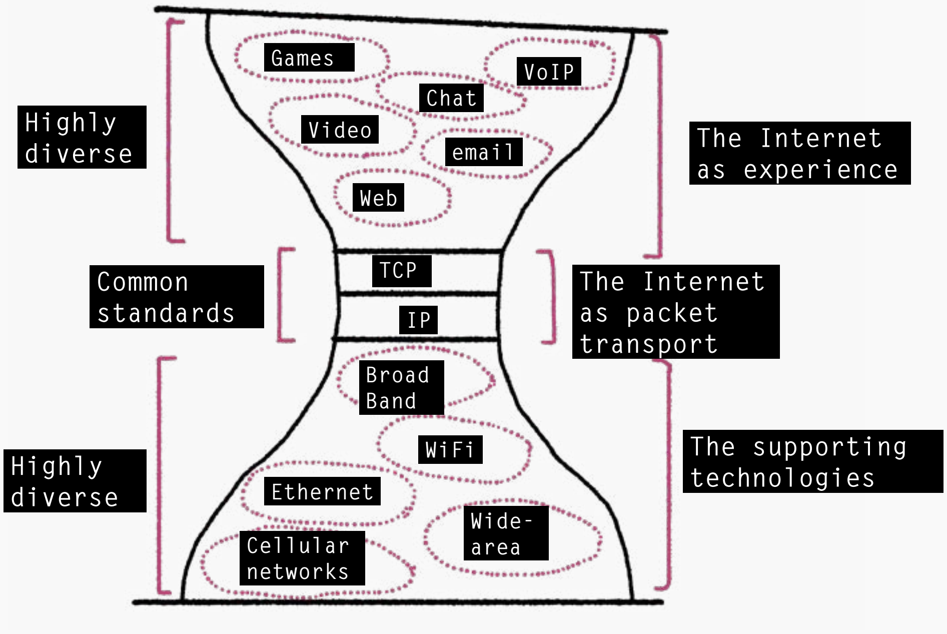
To access a web platform, different layers need to communicate each other. In the application layer runs the HTTP or HTTPS[28] protocol; in the transport one runs the TCP (Transport Control Protocol), which is responsible for directing the information using different ports (for example, port 80 for HTTP and 443 for HTTPS). Then, at the network layer, each connected device obtains an IP (Internet Protocol) address that identifies it. Finally, the hardware converts all the connection information into binary code. [29]

We could say that the network (IP) is until today the basis of how the internet works. Together with TCP, they have been responsible for ensuring that very different types of applications communicate each other, also they work together using different communication technologies, like this:

[28] HTTPS adds a layer of encryption to the HTTP protocol. Through the generation of an SSL (Secure Sockets Layer) certificate, the integrity of the information shared with a specific website is guaranteed, as well as the identity of the site and privacy in the information. Best explained in this comic https://howhttps.works

[29] Shuler, R. 2018. How Does the Internet Work? Standford University. https://web.stanford.edu/class/msande91si/www-spr04/readings/week1/InternetWhitepaper.htm

Communication model of the internet



Based on Clark, D. 2018. Designing an Internet. Massachusetts Institute of Technology.

IP is supposed to be in everything that happens on the internet, because everything runs over IP. But as we will see later, TCP is not the only transport protocol that can be used. TCP works in the end nodes: it is information carried by the packets, but it should not be checked by the intermediate routers, which should only look at the IP information. The idea of layered structure is that most work happens at the endpoints and not on the network.[30]

In practice, while increasingly complex internet functionalities are being developed, such as streaming real-time audio and video, web developers do not need to know the details of each technology they use. Protocols and standards are common agreements to make the work at the endpoints compatible with the network functioning.

WebRTC is the standard for real-time audio, video and data communications on the web and is perhaps one of the clearest examples of how something very complex in its internal logic, is easy to implement. Both for those who develop video call applications (for example, Jitsi), for those who develop and maintain browsers (such as Firefox or Chrome) and for the people who use those applications. Henceforth, we are going in depth throughout the process of a video call and the protocols involved.

Although technical protocols usually refer to "clients" as those with the power to communicate, we will make a great effort to differentiate them from users, who interact with those clients (a video call application or a navigation one) and that, finally, are interested in establishing communication.

[30] Although this principle is not currently fulfilled, because the network is full of intermediaries who analyze the information to forward it.

Audio, video and data in real-time

WebRTC is an open source project initially promoted by Google in 2011. Its main objective is to allow the transmission in real-time of audio, video and generic data between browsers, guaranteeing quality and privacy in communications. The benefit that WebRTC offers to an end user is that she can establish a communication from their browser, without having to create a profile, install an application or download *add-ons* or *plug-ins*.

To achieve interoperability between different browsers (even belonging to different companies, they can communicate with each other) the project is based on open standards, developed in the World Wide Web Consortium[31] at the API (Application Programming Interface) level, and in the Internet Engineering Task Force[32] at protocols level.

Currently, the most used browsers support WebRTC, that is, they have the necessary API to establish communication between peers. This does not mean that browsers have this capacity themselves, because to guarantee good quality real-time communications it is necessary to have high information processing speeds, among other resources that normally do not have a home device such as computers, tablets or cell phones.

On the other hand, even if its main objective is to establish a ***peer-to-peer*** (P2P) communication between two or more browsers, WebRTC can also be implemented in an independent application that can also be integrated with other existing communication systems, such as VoIP (voice over IP), clients SIP (Session Initiation Protocol or Session Initiation Protocol) or PSTN (Public Switched Telephone Network or Public Switched Telephone Network), traditionally used in digital telephone service. This is why WebRTC is not only about bringing real-time communications to the browser, but also about bringing the competences of the web to the world of telecommunications.

In the WebRTC model, the browser is expected to have the ability to work in conjunction with backup servers that have sufficient resources to implement the required functions. Therefore, before starting any transmission, it must be made a signaling between devices. In other words, identifying themselves as the end points that will establish a P2P communication using the internet.

Once the devices are identified a WebRTC session is opened between *peers*, that does not use TCP for transportation but UDP (User Datagram Protocol) since, as it is real-time media, it is more important that the information is transmitted immediately, and not that each package is reliable. Unlike TCP, UDP does not offer any promise about the reliability or order of the data.

[31] WebRTC 1.0: Real-Time Communication Between Browsers https://www.w3.org/TR/webrtc/

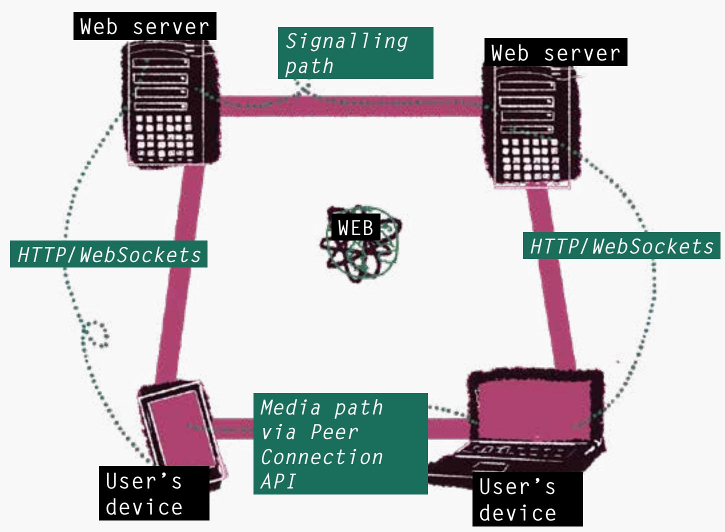
[32] Internet Engineering Task Force https://ietf.org/

1.Signaling

To exemplify, suppose a group of people are about to start a meeting and, by connecting at the agreed time, their devices will send a signal to identify themselves via the URL. This signaling process consists of searching for an intermediate server that allows establishing a direct transmission channel between browsers, then they will start a P2P communication flow.

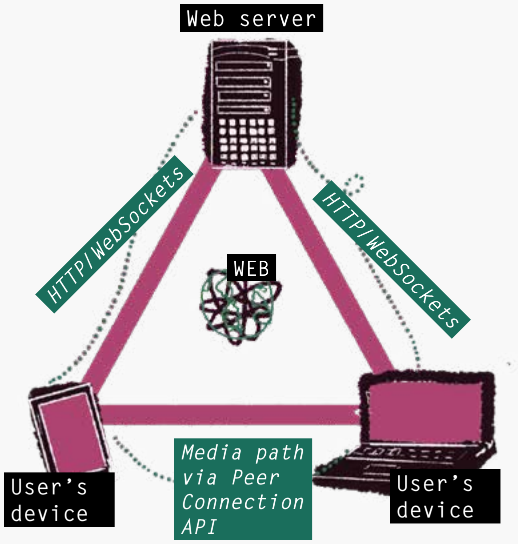
This process is not part of the WebRTC standards, but it is necessary as a preliminary step for establishing a P2P connection. To that extent, a mechanism for the transport of information between connected browsers has not been defined, since the intermediary server does not have the capacity to interpret the data content.[33] The information exchange will take place through RTCPeerConnection, once the session profile has been created with SDP (Session Description Protocol). Meanwhile, different mechanisms such as SIP over WebSockets, XMPP, MQTT or proprietary solutions[34] can be used for signaling. This process can be done using one or two intermediate servers, but the most common model is the triangular one.

SIP trapezoidal model



Both devices run a web application from different servers. Based on https://www.tutorialspoint.com/webrtc/index.htm

SIP triangular Model



Both devices run a single web application from the same server. Based on https://www.tutorialspoint.com/webrtc/index.htm

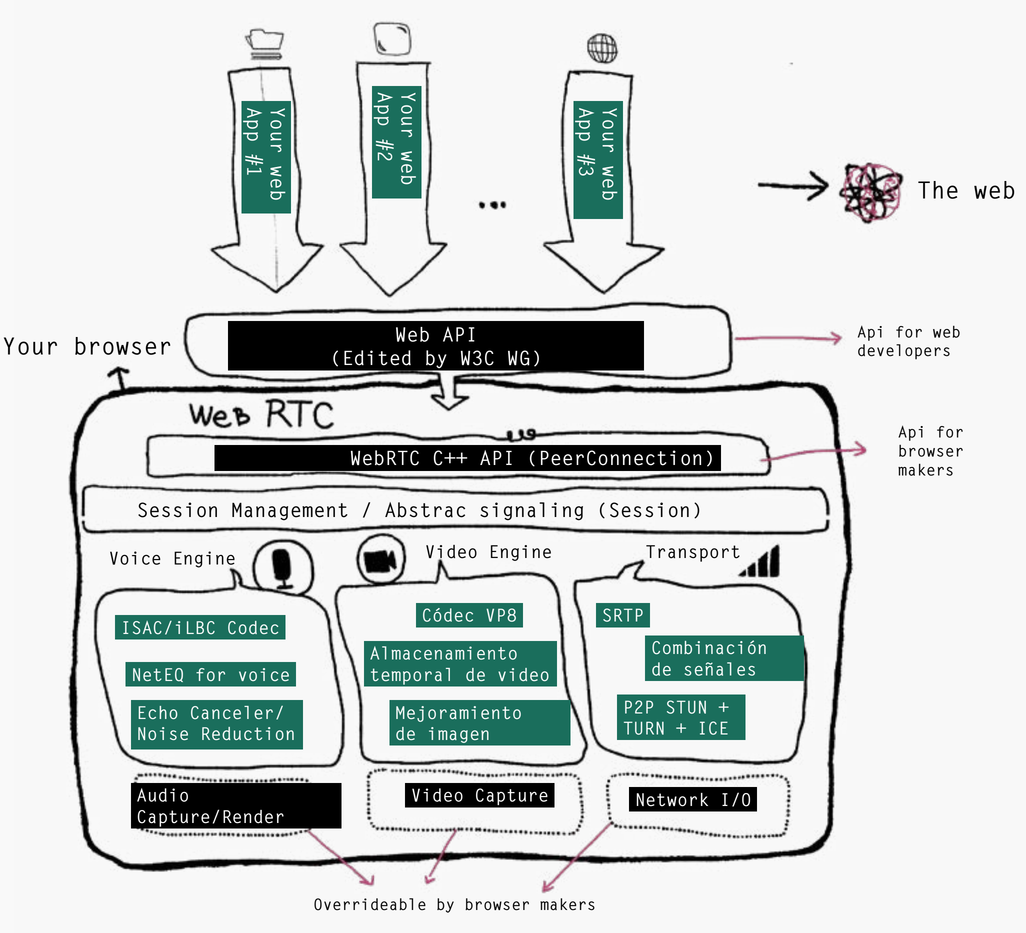
[33] Signaling and video calling https://developer.mozilla.org/en-

[34] Sobre los servidores que intervienen en una sesión WebRTC https://bloggeek.me/webrtc-server/

2. WebRTC architecture

Once the devices have been identified among themselves, the API is executed,[35] and their pieces fulfill different tasks to establish the information and media flows in real-time, directly between browsers. The quality of the transmission that users can enjoy depends on the way the API is implemented, both in browser (for example, Firefox or Chrome) and applications (for example, Jitsi or Zoom), specifically by the configuration of codecs or formats in which the audio and video will be transferred.

WebRTC API architecture



Based on https://webrtc.github.io/webrtc-org/architecture/

Besides establishing, managing and maintaining the session or P2P communication channel (RTCPeerConnection) between browsers, WebRTC API fulfills two other main tasks: 1) the capture, from the browser, of the audio and video tracks that will be transmitted (MediaStream) and 2) the transmission of data, different from audio and video (RTCDataChannel).

[35] WebRTC - Architecture https://www.tutorialspoint.com/webrtc/webrtc_architecture.htm

Moreover, the parameters for the transport of data in real-time are established.

2.1. RTCPeerConnection

Much of what is considered WebRTC is in the establishment of P2P: the processing of SDP and ICE protocols, which will be described in the next section, managing a UDP connection with another user; the ability to communicate with a WebRTC session through a phone call; the opening of a data channel; the identity verification of connected *peers;* the connection maintenance and monitoring, as well as the closing of the connection once it is no longer needed; and the statistics report.[36]

2.2. MediaStream

The MediaStream allows capturing, from the local browser, both camera and microphone of the device, asking the user beforehand if allows this access; and, in case there is more than one camera or microphone, it also permits or not to access any of them. MediaStream interface can consist of multiple video or audio tracks, if it is a multi-participant session. The flow is opened with the session description using an intermediate server, but once opened, the media is shared through the RTCPeerConnection, bypassing the server.

2.3. RTCDataChannel

In addition to P2P media , WebRTC can also send data bi-directionally that does not require codec negotiation or stream synchronization. The main task of RTCDataChannel is to create a channel that comes from an existing RTCPeerConnection object. It uses the same API as WebSockets and has very low latency.

2.4. Codecs

In the context of WebRTC, a codec is a piece of software whose function is to compress and decompress a digital media stream, discarding all information that is not perceptible to the eye or ear of a person, in order to encode and decode in the shortest possible time (that is, with low latency), but taking care that the transmission is clear for the participants.

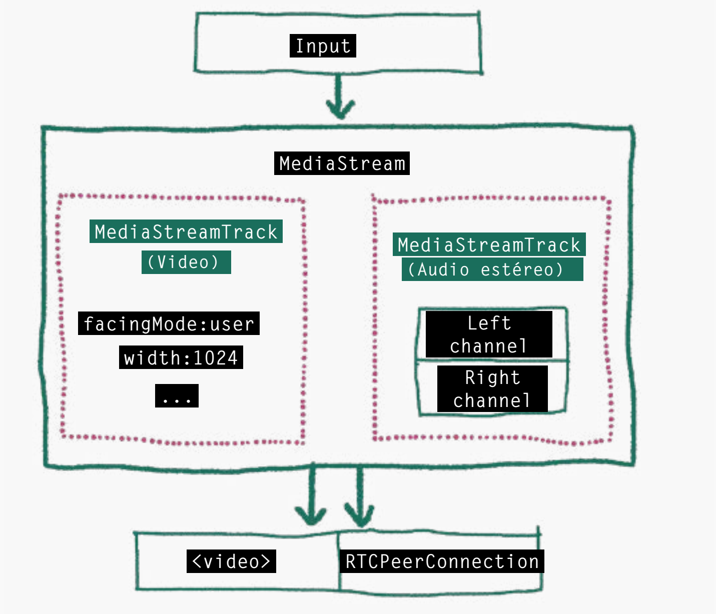
In general, for media streams to be stored or transmitted is necessary to encapsulate them together in containers, what for users are file formats or extensions as mpg, .avi, .mov, .mp4,. rm, .ogg, .mkv. For the transmission in real-time, the objective is that the different tracks can be synchronized, then the same user might want to share their camera and their screen at the same time, or several users could be sharing their camera while they listen to each other.

Even if the success of a WebRTC transmission depends to a large extent on the participants'

[36] RTCPeerConnection https://developer.mozilla.org/es/docs/Web/API/RTCPeerConnection

connection quality,[37] the codec configuration in the browsers and in the applications allows to have a better transmission quality using the minimum amount of resources, as well as the codec negotiation that is done through SDP.

Stream coding in WebRTC



MediaStream synchronizes multiple audio and video tracks (MediaStreamTrack). Based on https://hpbn.co/webrtc/

Although codecs have evolved, the most widely used audio standard in WebRTC today is Opus, according to RFC 7874,[38] but the use of additional codecs is contemplated for greater interoperability, according to RFC7875.[39] Opus is designed to support interactive audio applications such as VoIP, video conferencing and voice chat in games, among others. This, like the other codecs used in WebRTC, are characterized by having loss, that is, they do not retain all the original information.

Today, the video standard that is most widely used in WebRTC, VP8 + H264, was developed by Google and is an open source, and that is why it has been adopted in different applications, not just those developed with the WebRTC standard. Currently, Chrome browser, owned by Google, has implemented the VP9 codec, the same used on YouTube, also owned by Google.

[38] WebRTC Audio Codec and Processing Requirements https://tools.ietf.org/html/rfc7874

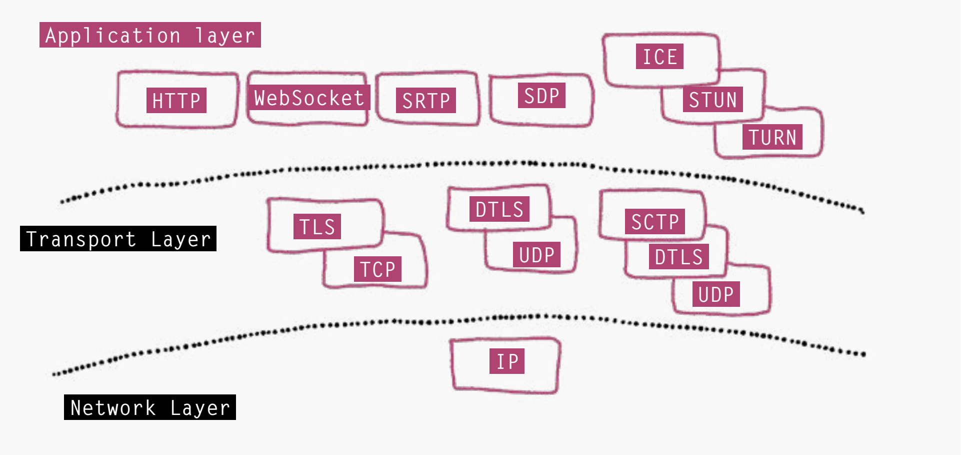
[39] Additional WebRTC Audio Codecs for Interoperability https://tools.ietf.org/html/rfc7875

3. Protocols for WebRTC

So far, we have reviewed the WebRTC API, which allows one to capture audio and video in each of the browsers that will be within a session, with the authorization of their respective users, to start a media stream. In other words, to recognize and to synchronize the tracks (for instance, there can be two of video and four of audio) with the best capabilities that each browser has, and to attach the media tracks to be broadcasted. Furthermore, once the connection is established, it allows the exchange and agreement of communication details between browsers (codecs, information on bandwidth and IP addresses). Besides -in parallel to media transmission- it also enables opening a data channel between browsers.

To make this process possible, different protocols are used at the application as well as at transport layers, which have not been specifically developed for WebRTC, due to the fact that before its invention there were infrastructures for IP telephony and unidirectional audio and video transmission, among other functionalities. For this reason, necessary extensions have been developed to support real-time audio and video, under the conditions defined by the WebRTC standard.

WebRTC protocols



Based on WebRTC Tutorial, IETF 100. 2017. https://youtu.be/viZC1G4tmVM

Despite WebRTC runs over the UDP transport protocol, pre-signaling is done over TCP. With NAT, STUN and TURN protocols, a P2P connection is established and maintained over UDP, but as we will see later, ICE is the process by which this interaction between browsers is possible, because it processes the connection establishment requests that registers each browser with the RTCPeerConnection object. Once that process is complete, SDP offer is generated and the signaling channel is used to reach *peers*.

Parallel to media transmission, a data channel between browsers will be opened. This channel uses the SCTP transport protocol to control flow and congestion, and measure the quality of service, bearing in mind that transport over UDP is not reliable (unlike TCP), but is based on the *best effort* principle. Additionally, the TLS, DTLS and SRTP protocols are presented throughout the process, to guarantee the security and privacy of the transmitted information.

3.1. Application Layer

One of the biggest benefits that WebRTC offers is that all management of a live stream is done from the web browser. For users, this enables to access, learn and use this type of system, and perhaps that is why the gamer community is the one that has contributed the most to its development and implementation. From a technical point of view, this implies that capabilities must be developed in the browser that allow opening and maintaining stable communication flows through the internet infrastructure. That's the job of application protocols.

3.1.1. NAT - Network Address Translation

RFC 2663 https://tools.ietf.org/html/rfc2663

NAT exists to manage the limited number of IP addresses in the version 4 of IP protocol (IPv4). When we connect, our device makes a request to the internet service provider (ISP); throughout the router managed by that ISP we can access a public IP address and navigate. Using a single port, NAT will translate a private IP address into a public one.

For security reasons, today many home routers serve as firewalls and NAT devices.[40] In addition, there are different types of NAT configurations, depending on the restrictions to communicate the devices of the local private network with the external devices.

For the interaction establishment through ICE, it is necessary to send and receive packets between internal and external devices, and this is done through STUN, but the symmetric NAT configuration does not support that protocol, due to the fact that the translation from a private IP address to a public one is conditioned by the destination IP address to which one wants to send the traffic. That's what TURN is used for.

STUN - Session Traversal Utilities for NAT

RFC 5389 https://tools.ietf.org/html/rfc5389

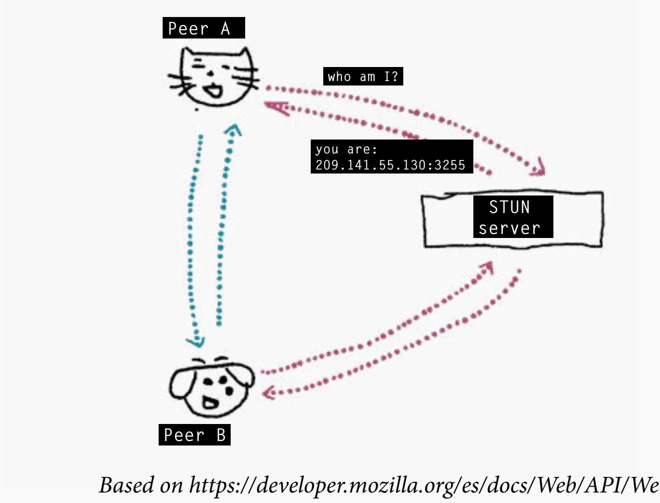
This protocol allows a user to know the public IP address whenever surfing the internet. It works under the *client/server* model, since it allows NAT clients (like a browser) to find its public IP address, the type of NAT it is in and the internet port associated with the local port through NAT.

In the context of WebRTC, this information is used to configure an UDP communication between two devices that are behind NAT routers. The software must incorporate a STUN

[40] Clark, D. 2018. ibid, p. 25.

client that sends requests to a STUN server, which informs the client of its public IP and which port has been opened by NAT to allow incoming traffic to the client's network. Different types of NAT handle incoming UDP packets, although it is usually done through port 3478 over UDP.

An STUN server discovers the public IP of the client



Based on https://developer.mozilla.org/es/docs/Web/API/WebRTC_API

3.1.2. TURN - Traversal Using Relays around NAT

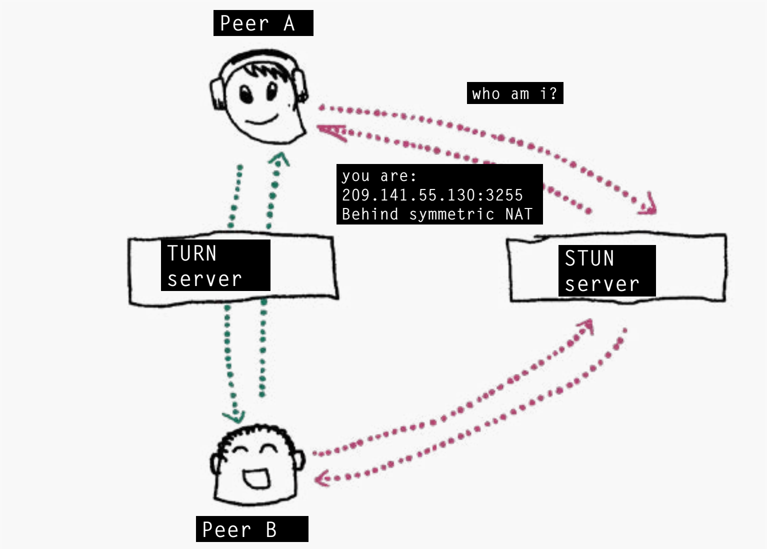
RFC 5766 https://tools.ietf.org/html/rfc5766

When a router uses symmetric NAT or a firewall system, the traversal protocol uses repeaters around NAT. TURN is designed to avoid these restrictions and uses a third server to relay all messages between two clients. For this purpose, the client must connect to the TURN server and that server connects to the destination on its behalf, relaying the packets.

Even though this process is more resource intensive and therefore only as a last resort used, today most connections have security protections, so nearly any WebRTC service must support the use of TURN. To optimize resources, COTURN[41] servers have also been developed, that act as TURN and STUN at the same time.

[41] Free open source implementation of TURN and STUN Server https://github.com/coturn/coturn

A TURN server solves the symmetric NAT restriction



Based on https://developer.mozilla.org/es/docs/Web/API/WebRTC_API

3.1.3. ICE - Interactive Connectivity Establishment

RFC 5245 https://tools.ietf.org/html/rfc5245

The protocol for the establishment of interactive connectivity is a process that facilitates a web browser to connect with others, identifying a reliable route to do so. The connectivity offer that is set in the RTCPeerConnection object contains a list of candidates IP, as well as port titles available to a remote entity. With this, the ICE agent can verify the connectivity conditions to see if it can reach the other entity.

The process runs nearly in the following way: the ICE agent sends a STUN join request that the other entity must acknowledge with a successful STUN response. If this is completed, a path for the P2P connection will be opened. If the candidates fail, it can happen that the RTCPeerConnection is marked as failed or that the connection falls to a TURN relay server to establish the connection.

The ICE agent automatically sorts and prioritizes the order in which connection checks are performed for candidate entities: local IP addresses are checked first, then public IPs via STUN, and TURN is used as a last resort. After the connection is established, the ICE agent continues to issue periodic STUN requests to the other peer. This is to keep the connection alive and to see if a better performance can be delivered through an alternate route.

3.1.4. SDP - Session Description Protocol

RFC 4566 https://tools.ietf.org/html/rfc4566

A session is described with a series of attributes. Each attribute is on a line, illustrated in the following way:

v = (protocol version)
o = (source and session identifier)
s = (session name)
i = * (session information)
u = * (URI description
e = * (email)
p = * (telephone number)
c = * (login information)
b = * (zero or more lines with bandwidth information)
One or more lines of time description (See below "t =" and "r =")
z = * (time zone settings)
k = * (encryption key)
a = * (zero or more lines of session attributes)
Zero or more media descriptions

SDP is responsible for describing a session profile. This protocol is widely used for real-time transmission, and within the framework of WebRTC it is used together with SIP, mainly to define how to encode the media that will later be transmitted using SRTP (Secure Real-Time Transport Protocol).

This protocol allows negotiating under the *offer/answer* model by recognizing the capabilities of each one of the entities that will participate to establish the parameters to open a session. Its function is not to deliver content, but to enter into a negotiation to define which codecs to use, what bandwidth can be linked to the connection and who the IP candidates to connect with are.

As soon as the RTCPeerConnection has been created, SDP *offer/answer* text string needs to be created for the calling and receiving entity. When both entities are already recognized, the server that made this connection possible loses control over the session, and the direct P2P connection over UDP starts. ICE takes care of this. Communication will last as long as there is data flow.

3.2.Transport layer

In the context of WebRTC, the transport layer fulfills the functions of signaling, congestion control and management in network traffic, in order to guarantee quality of service.

Although a whole work area has been developed in this layer in real-time media transport (which are normally encapsulated in UDP), the protocols that intervene in WebRTC must support a real-time communications service that runs on top of from the web, which in turn runs over TCP.

3.2.1. TCP – Transport Control Protocol

RFC 793 https://tools.ietf.org/html/rfc793

This protocol is as old as the internet. It was designed to meet the specific needs of communication systems in the military field, is means, in an environment susceptible to be attacked. That is why TCP is connection-oriented: to run, it requires prior synchronization of the parts to be communicated. In addition, it is designed so that information is transmitted reliably end-to-end, and for that reason it employs a verification system, each time a packet is sent and received. Applications such as the web, email, FTP (for file sharing) or SSH (for remotely connecting to a server) run over TCP.

For transport, TCP organizes and sends each byte individually, ensuring that all of them arrive at their destination, in order and without errors. In addition, with the port system, it allows data from different applications to be transported simultaneously. Initially, TCP and IP were the same basic internet protocol. But, given its complexity, TCP often had a delay in sending packets, what was not useful, for example for audio transmission. That is why in the late 1970s the network (IP) layer was separated from the transport (TCP) layer and the UDP protocol began to be developed.[42]

3.2.2. UDP – User Datagram Protocol

RFC 768 https://tools.ietf.org/html/rfc768

Originally published in 1980, UDP is transaction-oriented, it means that it does not require prior establishment of a connection to run. Application protocols such as DNS (for domain name resolution), DHCP (for assigning private IP addresses in local networks) or RIP (with information for packet routing) work over UDP, since it requires a minimum of the network to run.

As its name suggests, UDP works with messages, that is, with datagrams or packets of bytes, not with individual bytes. This allows one to be more agile, because it does not guarantee the order in which the packages arrive at their destination, nor if they arrive.

RTP (Real-Time Transport Protocol) packets travel encapsulated in UDP. Firstly published in 1996, RTP has served the development of communication and media transmission systems, including IP telephony and television systems, among other systems. WebRTC standard uses SRTP protocol since encryption is mandatory.

[42] Clark, D. 2018. ibid.

3.2.3. SCTP - Stream Control Transmission Protocol

RFC 4960 https://tools.ietf.org/html/rfc4960

SCTP runs directly over IP, it was developed for transport signaling in public switched telephone networks (PSTN) and was first published in 2000. At the transport layer, it is an alternative to TCP and UDP, as it is connection-oriented, provides reliability, flow control, and packet sequencing, like TCP. But similarly to UDP, it uses message delimiters, not bytes, to guarantee the arrival of all the information, allowing it to be sent in disorder, what makes transportation more efficient.

It is about a much less complex protocol than TCP and despite that, it allows congestion control and improve fault tolerance when sending packets, offering multihoming support (simultaneous connection to several networks) and multistreaming (several data streams with the same port, so that communication is not blocked if there is any failure.). Guaranteeing greater security in communication with a four-way handshake that includes an authentication cookie and a mandatory verification tag in the header of each sent packet.

Parallel to the P2P media transmission, WebRTC opens a data channel between browsers. This channel uses SCTP to queue management and congestion control, but here SCTP connects through a DTLS tunnel to guarantee information confidentiality. Likewise, DTLS runs over UDP, what provides transport through NAT once a channel has been opened through ICE.

4. WebRTC Security and privacy

WebRTC has been developed with ease of access to real-time media streaming. That is why the standard proposes that it runs on the web and does not require any application or *plug-in* to be installed. Taking into account the risks associated with this ease of access for users, encryption is a mandatory feature in WebRTC; and therefore, security is based mainly on DTLS and SRTP protocols and requires browsers to implement access authorization management to the camera and microphone.

Even if WebRTC is careful in setting up security and privacy for streaming media, the prior signaling process fell outside the standard. However, it is based on the exposure of local capacities and flows by end customers in browsers and devices. This supposes an exercise of trust, for those who develop applications and also for those who use them, since third parties (in this case servers) with access to the information of the participants necessarily take action in connection establishment.

The development of extensions and protocols for the implementation of WebRTC has been a permanent problem. At IETF, a Working Group was formed to improve privacy in RTP based conferencing,[43] which works specifically on the SRTP protocol, but also on SIP (a protocol widely used in signaling WebRTC applications). Security considerations related to the set of APIs and protocols used by WebRTC are described in an Internet-Draft, which will be published soon as an RFC.[44]

4.1. DTLS - Datagram Transport Layer Security

RFC 6347 https://tools.ietf.org/html/rfc6347

This protocol provides privacy in communications and prevents its interception and manipulation. It is based on TLS (Transport Layer Security), an extended security protocol for communications. The main difference between these two protocols is that TLS runs over TCP and DTLS over UDP. Currently, DTLS runs at version 1.2, published in 2012.

During the ICE process, data is encrypted using DTLS that must be integrated into all browsers that support WebRTC. This is used to secure all P2P data transfers.

4.2. SRTP - Secure Real-Time Transport Protocol

RFC 3711 https://tools.ietf.org/html/rfc3711

Similarly to DTLS, WebRTC also encrypts media via SRTP to ensure that unauthorized third parties listen to or view broadcasts, and to minimize the risks of attacks such as denial of service. Released in 2004, SRTP establishes an encryption and authentication system for traffic in the RTP and RTCP protocols.

RTP was initially published in 1996 and updated in 2003. It is one of the technical foundations of VoIP, hence it is implemented in many other communication systems besides WebRTC. RTP runs over UDP and is used in conjunction with RTCP (Real-Time Control Protocol), which allows tracking and monitoring the sending of packets. While RTP streams content, RTCP captures stream statistics and quality of service, and helps to synchronize multiple streams.

4.3. Permissions in browser or web application

According to RFC7478 regarding use cases and requirements in real-time communications on the web,[45] the browser that establishes a communication through WebRTC should provide several mechanisms to guarantee consent of access to camera, microphone and screen, by users. This is usually implemented through a message where access can be accepted or denied. Furthermore, browsers should implement some mechanism to report when camera

[43] Privacy Enhanced RTP Conferencing (perc) https://datatracker.ietf.org/wg/perc/about/

[44] WebRTC Security Architecture. https://tools.ietf.org/html/draft-ietf-rtcweb-security-arch-20

[45] Web Real-Time Communication Use Cases and Requirements. https://tools.ietf.org/html/rfc7478

and microphone are being used, what is usually done through an icon. In addition, users should be able to review and revoke this permission at any time and, for that, applications that implement WebRTC should ensure that their users consent the establishment of communication between them, in order to receive and send any data flow.

4.4.About end-to-end encryption

WebRTC can be used for communication between two people or between larger groups, and the implementation of DTLS and SRTP security protocols will be different in each case. On the one hand, in P2P communications are encrypted E2E using DTLS and SRTP, as detailed in RFC5763,[46] even if the packet is sent through intermediaries, for example TURN servers.

On the other hand, when communications are between more than two people (multiparty sessions), the encryption layer provided by DTLS and SRTP is removed when packets pass through intermediate servers. Some video calling services, such as Jitsi, are testing an E2E encryption deployment in these multiparty sessions,[47] on the basis of WebRTC Insertable Streams API.[48]

5. WebRTC in Browsers

Following the layer structure, browsers should have these functionalities for the execution of applications using real-time protocols:[49]

Support to the local system	They do not need to be uniformly specified since each participant can choose how to do it, without affecting the transmission. For example: echo cancellation, local authentication and authorization mechanisms, access to the operating system and ability to record locally.
Presentation and control	They guarantee that interactions do not behave in a surprising way, cooperation of participants is required. Many applications have been built without standardized interfaces for these functions. For instance, ground control, screen layout, voice activation of image switching, among others.
Connection management	Establishment of connections, agreement on data formats, changes in data formats during a call. To this category belong signaling protocols such as SDP, SIP, and Jingle / XMPP.

[46] Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS) https://tools.ietf.org/html/rfc5763

[47] This is what end-to-end encryption should look like! https://jitsi.org/blog/e2ee/

[48] WebRTC Insertable Streams https://www.chromestatus.com/feature/6321945865879552
Avances para cifrado punto a punto en https://webrtchacks.com/true-end-to-end-encryption-with-webrtc-insertable-streams/ and https://webrtchacks.com/you-dont-have-end-to-end-encryption-e2ee/

[49] Overview: Real Time Protocols for Browser-based Applications
https://datatracker.ietf.org/doc/draft-ietf-rtcweb-overview/

Data formats	Codec specifications for audio and video, as well as format and functionality for data that passes between systems.
Data framing	Protocols like RTP, SRTP, DTLS, and others. They serve as containers and guarantee data integrity and confidentiality.
Data transport	Protocols such as TCP, UDP, SCTP and means to establish safe connections between participants and functionalities to decide when to send data: management of congestion, bandwidth and estimation.

WebRTC promises interoperability and ease for establishing video and audio connections between browsers, but we did not find up-to-date documentation on how well we can connect from certain operating systems and web browsers. According to Wikipedia, and at this point, it is important to say that the technical information on real-time communications is very clear and complete there, especially in English.[50] WebRTC is compatible with the following browsers:

Computers	Android devices	iOS devices
Microsoft Edge 12+	Google Chrome 28+	MobileSafari/WebKit (iOS 11+)
Google Chrome 28+	Mozilla Firefox 24+	
Mozilla Firefox 22+	Opera Mobile 12+	
Safari 11+		
Opera 18+		
Vivaldi 1.9+		

Nonetheless, it seems to be an outdated and under-referenced information that contradicts data from other projects such as caniuse.com[51] and the tests carried out by ourselves. caniuse.com offers a more detailed comparison and specifically reports that WebRTC is not supported by Internet Explorer, UC Browser and Opera Mini browsers. Other references on the internet[52] specify that WebRTC is compatible with Chrome, Mozilla Firefox, Safari, Opera and other Chrome-based browsers, without giving much more detail.

To provide more up-to-date data on browsers' WebRTC support, we have prepared the following table.

[50] WebRTC https://en.wikipedia.org/wiki/WebRTC#support

[51] WebRTC Peer-to-peer connections https://caniuse.com/#feat=rtcpeerconnection

[52] Who Supports WebRTC? https://www.3cx.com/webrtc/which-browsers-support-webrtc/; Which web browsers are currently supporting WebRTC? https://support.pexip.com/hc/en-us/articles/216077528-Which-web-browsers-are-currently-supporting-WebRTC

Table 1. WebRTC support in browsers

Browser *(usage rate*)*	Chrome *(63,97 %)*		Safari *(16,96%)*		Firefox *(4,44%)*		Samsung Internet *(3,39%)*		UC Browser *(2,69 %)*		Opera *(2,2%)*		Edge *(2,11%)*		IE *(1,79%)*	
Operating system (OS)	Test	Version	T	V	T	V	T	V	T	V	T	V	T	V	T	V
Windows 10	🟢 ❶❷❸	83	⊗	⊗	🟢 ❶❷❸	78	⊗	⊗	🔴 ❶❷❸	13	🟢 ❶❷❸	69	🟢 ❶❷❸	83	🔴 ❶❷❸	11
Debian 10	🟢 ❶❷❸	83	⊗	⊗	🟢 ❶❷❸	68	⊗	⊗	⊗	⊗	🟢 ❶❷❸	69	⊗	⊗	⊗	⊗
MacOS	🟢 ❶❷	83	🟡 ❶❷	13	🟢 ❶❷		⊗	⊗	⊗	⊗	🟡 ❶❷	69	🟢 ❶❷	83	⊗	⊗
Android	🟢 ❶❷	83	⊗	⊗	🟢 ❶❷	68	🟢 ❶❷	12	🔴 ❶❷	13	🟡 ❶❷	Touch 2	🟢 ❶❷	45	⊗	⊗
iOS	🔴 ❶❷	83	🟢 ❶❷	13	🔴 ❶❷	28	⊗	⊗	🔴 ❶❷	13	🟡 ❶❷	Touch 2	🔴 ❶❷	45	⊗	⊗
Chromium based	*Yes*		*No*		*No*		*Yes*		*No*		*Yes*		*Yes*		*No*	

Leyend:

❶ Jitsi call test

❷ bbb call test

❸ wpt test

⊗ No browser-OS compatibility

Color code:

Green	Working
Yellow	Working with errors
Red	Not working

Web-platform-tests offers public test development for web standards[53] that can be run in the browser of your choice. There are a series of tests for WebRTC[54], which we carry out following indications[55]and are collected in tables 2 and 3.

Table 2. web-platform-tests on Windows 10

Browser	Version	Approved Tests	Failed Test
Firefox	78	1058	706
Chrome	83	1288	304
Edge	83	1270	314
Opera	69	1244	313
UC Browser	6	87	17

[53] The Web platform: Browser technologies https://platform.html5.org/

[54] Directory listing for /webrtc/ https://wpt.live/webrtc/

[55] Running Tests from the Local System. https://web-platform-tests.org/running-tests/from-local-system.html

Table 3. Web-platform-tests on Debian 10

Browser	Version	Approved Tests	Failed Test
Firefox	68	Error al ejecutar	Error al ejecutar
Chrome	83	1334	332
Chromium	83	1221	310
Opera	69	1264	307

From the data obtained in our tests, we can conclude that WebRTC is not fully supported by the most used browsers worldwide, while Chrome and the browsers based on its code underscored their compatibility.

Given these results, it seems important to know more about what difficulties browser developers are encountering to make them compatible with WebRTC, but also why Chrome-based browsers better support WebRTC and how compatibility, which Chrome and other browsers (based on its code) offers, is translated into consumption.

Considering that Google was the brand that promoted WebRTC as an open standard, what today continues to be one of the main project promoters, and that its main video calling services (Google Hangouts, Google Meets and Google Duo) are based on WebRTC, we wonder: How are large corporations influencing the development of standards like WebRTC? How does this affect our freedom and autonomy as users, when choosing which software to use to make video calls?

6. All this for what?

In May 2020, IETF resumed work started in 2017 on encrypt E2E media streams when necessarily there must be an intermediate server, such as group video calls. Until now, some encryption solutions have been implemented in specific applications and some browsers have supported it, but there is still no open standard in this regard.[56] The conversation can be followed on an open mailing list,[57] where it would be good to have a more diverse participation.

Joining a technical conversation is difficult, especially if there are disagreements and differences. However, to transform something it is necessary to understand how it works, or at least to wonder about it. This document and our previous exercise of inquiry and understanding have that purpose.

The web is plenty of information on how WebRTC works. Almost all in English and aimed

[56] Secure Frames (SFrames): end-to-end media encryption with #webrtc now in chrome. https://webrtcbydralex.com/index.php/2020/03/30/secure-frames-sframes-end-to-end-media-encryption-with-webrtc-now-in-chrome/

[57] Frame-based end-to-end encryption of real-time media https://www.ietf.org/mailman/listinfo/sframe

at developers interested in implementing or adapting the standard to their needs. This document, with its failures and successes, is the result of an exercise that sought to gather all this information and explain it coherently for us, users with very different technical capacities, hoping that it may be of interest and utility to our colleagues.

Because if during the pandemic we were able to continue with our organizational processes thanks to the use of digital tools, especially video calls, the exposure of our voices and bodies on the screens also made us an object of attacks. [58] This, of course, is not a new scenario. The pandemic only made the violence to which women and gender diversities are exposed more evidently. Violence that, of course, is intensified due to conditions of race, class, abilities, age and geographical location.[59]

If the tools we use to work, organize and disseminate ideas are the same with which we maintain affective relationships of different types in the distance, claiming privacy necessarily implies claiming control over our information. If personal issues are political, will it also be public? If we make a secure video call, how sure are we that our information is protected? Protected from whom, or from what? In whose hands are we allotting that protection?

With or without a pandemic, there are many strategies that we can work on[60] to eradicate violence based on traditional systems of oppression. Understanding how they work allows us to imagine other systems, where submission is not the rule, neither in use nor in development.

The design of highly complex communication systems puts us further and further away from this possibility. Is it possible to communicate with less complex systems? Is it possible to make its complexity more visible and legible?

We leave these questions open.

[58] Trolls pandémicos https://www.pikaramagazine.com/2020/05/trolls-pandemicos/

[59] La otra pandemia: internet y violencia de género en América Latina https://www.derechosdigitales.org/14716/la-otra-pandemia-internet-y-violencia-de-genero-en-america-latina/

[60] Emergencia.Acoso.Online. Available materials to know what to do in case of non consensual pornographic content in platforms or any other type of online gender-based violence https://acoso.online/cl/emergencia/

DERECHOSDIGITALES
Derechos Humanos y Tecnología en América Latina