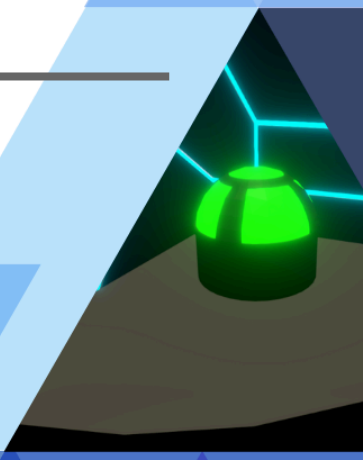
# Els Protocols de Comunicació Orientats a la Creació d'un Videojoc en Línia



**Autor:** David Didenco Pavalachi
**Tutor del TR**: Josep Ramon Juan
**Centre**: Miquel Martí i Pol
**Curs Acadèmic**: 2024/2025

“Vivim en un món on les dades són el nou petroli,
i la infraestructura és el nou camp de batalla.”

**Neelie Kroes**, excomissaria de la Unió Europea
per a l'Agenda Digital (2010, Conferència
d'Innovació de la UE).

“El ritme al qual innoven en el camp de
les comunicacions de xarxa determina el futur
de cada tecnologia que construïm sobre ell.”

**Robert Kahn**, pioner d'Internet i co-inventor del
protocol TCP (2004, Conferència de la IEEE
sobre Xarxes).

# Resum

Aquest treball de recerca estudia com s'envien i reben dades a través d'Internet en la creació de videojocs en línia, on jugadors d'arreu del món interactuen simultàniament. En aquest tipus de joc els dispositius han de comunicar-se constantment amb un servidor per enviar i rebre informació, com les posicions dels jugadors o les accions dins del joc. Aquesta comunicació segueix unes regles anomenades "protocols de comunicació".

Hi ha dos tipus principals de protocols: els fiables, que asseguren que la informació arribi completa i en ordre (TCP), i els no fiables, que prioritzen la velocitat (UDP). Aquest treball investiga quin és el millor per garantir una experiència de joc fluida i eficient.

Per poder veure quina opció és la més adequada, a part d'investigar la part teòrica, s'ha creat un videojoc multijugador utilitzant el motor de videojocs Unity. Aquest joc s'ha utilitzat com a exemple per provar com funcionen els dos protocols en diferents situacions. Així, s'ha pogut observar quin d'aquests sistemes de comunicació resulta ser més eficaç.

La recerca proporciona una comparació detallada dels dos protocols en diversos contextos, per entendre quins avantatges i desavantatges presenten en el desenvolupament d'un videojoc en línia.

# Abstract

This research project studies how data is sent and received over the Internet in the creation of online video games, where players from around the world interact simultaneously. In this type of game, devices must constantly communicate with a server to send and receive information, such as player positions or actions within the game. This communication follows a set of rules known as "communication protocols".

There are two main types of protocols: reliable ones, which ensure that information arrives complete and in order (TCP), and unreliable ones, which prioritize speed (UDP). This project investigates which of these is best suited to ensure a smooth and efficient gaming experience.

To determine which option is more appropriate, aside from the theoretical research, a multiplayer video game has been developed using the Unity game engine. This game is used as an example to test how the two protocols perform in different scenarios, allowing me to see which communication system is more effective.

The research provides a detailed comparison of the two protocols in various contexts, aiming to understand the advantages and disadvantages they present when developing an online video game.

# ÍNDEX

# Introducció

## Motivació

Personalment, m'interessa molt tot allò relacionat amb els ordinadors i la programació, per això desitjo estudiar una carrera vinculada a aquest àmbit. Una de les opcions que m'atrau és l'enginyeria en telecomunicacions, que considero una excel·lent alternativa per cursar, tot i que encara no estic completament segur d'aquesta decisió. Aquest treball, centrat en les telecomunicacions, em permetrà determinar si realment m'agrada aquest camp i si he de triar aquesta sortida acadèmica o, en canvi, optar per una altra alternativa.

## Hipòtesi

Els protocols no fiables (com UDP) són els més adients per als jocs en línia perquè aquests siguin fluids i no es quedin penjats a l'hora de jugar.

## Justificació i rellevància de la recerca

Actualment, els protocols de comunicació són fonamentals per al desenvolupament i la utilització de les tecnologies modernes. De fet, per poder fer servir qualsevol dispositiu electrònic o informàtic (com mòbils, ordinadors, videoconsoles o programes), és molt probable que calgui comunicar-se amb un servidor per al seu correcte funcionament. És en aquest context que els protocols de comunicació entren en joc, ja que la forma d'enviar la informació a través de les xarxes és molt més important del que podria semblar a primera vista.

Per exemple, si estàs jugant a un videojoc on tot succeeix a gran velocitat, el protocol ha de ser capaç d'enviar la informació tan ràpidament com sigui possible. D'altra banda, si estàs enviant un missatge privat a un amic, el protocol ha de garantir la màxima seguretat. Així doncs, segons la situació i el context en què s'hagin d'utilitzar els protocols, serà millor optar per un o altre. Aquest és el focus de la meva investigació: quan s'han d'utilitzar els diferents protocols disponibles en el context d'un videojoc en línia i per què.

## Contextualització del tema

El tema que vull investigar és sobre quin protocol de comunicació és el més adient per a la creació de videojocs en línia amb la fi que siguin el més fluid possible, centrant-me en aquells més optimitzats per aquesta tasca l'any 2024. Atès que els protocols utilitzats a Europa són els mateixos arreu del món, no és necessari especificar una zona geogràfica concreta, sinó més aviat quin subgrup es vol investigar. No em centraré en la infraestructura al darrere de la transmissió d'informació (com cables, torres…), sinó que només en la manera d'enviar aquesta informació des del *software o programari* (programes informàtics).

## Objectius del treball

L'objectiu principal d'aquest treball és analitzar i comparar els protocols utilitzats en la creació d'un videojoc en línia per determinar la seva eficàcia en diferents escenaris dins del joc, així com identificar quin és el més adequat en aquest àmbit, en l'any 2024. Es prendran en consideració factors com la fiabilitat, la velocitat i l'eficiència de la transmissió de dades per tal de proporcionar conclusions ben fonamentades sobre quin protocol és més adequat.

**Objectius específics:**

- Utilitzar d'una manera superficial el motor Unity per a l'assoliment de la part pràctica. Aquest s'utilizara principalment per crear l'apartat visual del videojoc.
- Programar i organitzar xarxes per a la comunicació de diferents dispositius electrònics, amb el fi per poder utilitzar els diferents protocols necessaris a la part pràctica (*Network programming*).

## Metodologia

A la part teòrica del meu treball de recerca, buscaré informació sobre els diferents protocols de comunicació que n'hi ha per a la creació un videojoc en línia i les seves característiques. Com a la part pràctica faré un amb el Unity, en aquesta secció també buscaré informació sobre com s'implementen els diferents protocols amb aquest motor gràfic.

A la part pràctica del treball crearé un videojoc en línia o multijugador (en el que jugues amb altres jugadors de manera simultània) amb el motor Unity. Aquest joc en línia em servirà per veure en un cas real l'eficàcia dels protocols investigats en la part teòrica, així com per entendre com s'utilitzarien aquests protocols en un projecte real. S'utilitzaran diferents protocols per assolir el mateix objectiu en una sèrie d'escenaris específics, cosa que em permet comparar-los i determinar quin ha dut a terme la tasca de manera més eficient. Totes aquestes dades seran recollides per a la posterior elaboració de la conclusió del treball de recerca.

Amb les dades recopilades en les diferents parts del treball es podran extreure les conclusions i demostrar si la hipòtesi formulada és correcta o no.

# Protocols als videojocs en línia

## Definició i funció dels protocols de comunicació

En el món de la tecnologia, la comunicació entre dispositius és crucial per al funcionament de bàsicament tots els aparells electrònics que tenim. Però com funciona aquest sistema tan important? Aquí és on els protocols de comunicació tenen una funció crucial.

Els protocols de comunicació són una sèrie de regles que permeten a dues o més entitats informàtiques (ordinadors, telèfons mòbils, etc.) comunicar-se entre si i transferir-se informació d'una manera ordenada i segura. Aquests tenen diferents formats, característiques i procediments per poder intercanviar paquets[1] depenent de la seva funció o finalitat.

És molt important mencionar que els protocols no són els cables de transmissió d'informació com a tal, són intermediaris que permeten que una aplicació informàtica (com per exemple WhatsApp) pugui enviar informació a qualsevol dispositiu arreu del món sense errors i de manera ràpida, fiable i segura.

## Els diferents nivells dels protocols de comunicació

De protocols n'hi ha de diferent tipus, tant físics (USB, Ethernet…) com digitals (TCP, HTTP…). Aquests estan organitzats en diferents nivells, anant des de l'1 fins al número 7. Cada apartat té la seva funció en específic a l'hora d'enviar un missatge des de l'emissor al destinatari. Aquesta manera d'estructurar els protocols es diu model *OSI (Open Systems Interconnection)*, que té com a objectiu estandarditzar els protocols que es creen i s'utilitzen a tot arreu. Per aquest motiu, els protocols utilitzats a Europa són els mateixos que s'utilitzen a Rússia o la Xina, és a dir, són els mateixos a tot el món.

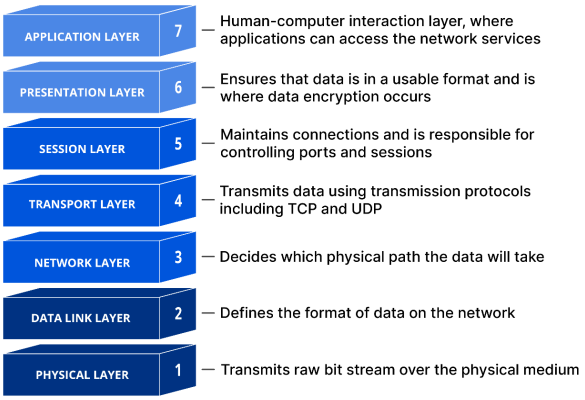



*Figura* 1: El model OSI de manera simplificada amb els 7 nivells breument explicats.

[1] ***Paquet:*** *En telecomunicacions, és un missatge gran dividit en segments més petits per optimitzar les comunicacions. L'emissor envia els paquets per separat i el receptor els combina per formar el missatge original.*

A continuació estan llistats alguns dels protocols de comunicació més importants depenent del seu nivell i la funció específica del nivell al qual pertanyen:

1. Nivell Físic: La capa física s'encarrega de les funcions materials per tal de mantenir un enllaç físic entre el receptor i l'emissor. D'una manera metafòrica, és com si fos el tipus de carretera o camí per on circulen els cotxes.
    -USB: Universal Serial Bus.
    -Ethernet: Ethernet physical layer.
    -DSL: Digital subscriber line.
    -Fibra òptica.

2. Nivell d'enllaç de dades: S'encarrega d'organitzar i regular el nivell físic. Són com les normes de circulació i els semàfors per mantenir l'ordre al carrer.
    -DCAP: Protocol d'accés del client de la commutació de la transmissió de dades.
    -FDDI: Interfície de distribució de dades en fibra.
    -HDLC: Control d'enllaç de dades d'alt nivell.
    -LAPD: Protocol d'accés d'enllaç per als canals.

3. Nivell de xarxa: S'encarrega que les dades enviades arribin al seu destinatari tot i no tenir una connexió directa. En aquest as, seria com el sistema GPS que et guia fins al teu destí quan vas amb cotxe.
    -ARP: Protocol de resolució d'adreces.
    -BGP: Protocol de frontera d'entrada.
    -ICMP: Protocol de missatge de control d'Internet.
    -IPv4: Protocol d'Internet versió 4
    -IPv6: Protocol d'Internet versió 6

4. Nivell de transport: S'encarrega de proporcionar un transport de dades fiable i un control del flux d'aquestes dades. És com la velocitat a la qual pots anar amb un vehicle i el control de trànsit d'aquests.
    -TCP: Protocol del control de la transmissió
    -UDP: Protocol de datagrames d'usuari

5. Nivell de sessió: S'encarrega d'administrar les connexions entre aplicacions locals o remotes, això també inclou establir-les i tancar-les. És com coordinar la sortida amb els teus amics per anar en cotxe.
    -NFS: Xarxa de sistema de fitxers.
    -SMB: Bloc del missatge del servidor.
    -RPC: Trucada a procediment remot.
    -SDP: Protocol directe de sockets.
    -SMB: Bloc de missatges del servidor.

6. Nivell de presentació: S'encarrega de representar les dades per al nivell d'aplicació dependent del format del missatge (text, àudio, vídeo, etc.). És com el llenguatge que utilitzes per comunicar-te amb els altres conductors.
    -TLS: Seguretat de la capa de transport.
    -SSL: Capa de connexió segura.
    -XDR: Representació de dades externes.
    -MIME: Extensions de correu de multiproposit.

7. Nivell d'aplicació: S'encarrega de proporcionar un servei de connexió per a les aplicacions. És com el servei que ofereixes als teus passatgers en el teu cotxe.
    -FTP: Protocol de transferència de fitxers
    -DHCP: Protocol de configuració dinàmica d'amfitrió
    -HTTP: Protocol de transferència d'hipertext.
    -HTTPS: Protocol segur de transferència d'hipertext.
    -SMTP: Protocol de transferència simple de correu electrònic
    -SSH: Protocol per l'accés remot a dispositius informàtics

En aquest treball de recerca em centraré en els protocols de comunicació del nivell 4, ja que són aquests els que s'utilitzen principalment per a la creació dels videojocs en línia. Aquests serien TCP i UDP. Aquests protocols en conjunt em permetran poder fer la meva part pràctica del treball, el videojoc en línia.

A continuació hi ha una petita descripció de cada un:

- **TCP *(Transmission Control Protocol)*:** Aquest és el principal protocol de comunicació utilitzat a internet, i molts protocols es basen en ell en la creació d'aquests. Aquest protocol permet mantenir una connexió oberta entre l'emissor i el receptor en tot moment durant el procés de transferència de la informació. A més, garanteix que tots els paquets transmesos arribin correctament i sense cap mena d'error al destinatari.
- **UDP *(User Datagram Protocol)*:** Aquest protocol és molt similar al TCP, no obstant això, aquest no estableix una connexió amb el receptor. El permet que la transmissió d'informació sigui més ràpida i utilitzi menys recursos.

## TCP

TCP és un protocol de comunicació que pertany al nivell 4, el Nivell de transport. És el més utilitzat a les xarxes de comunicació actual per la seva gran eficàcia i fiabilitat. I és per aquest motiu que molts altres protocols estan basats en aquest, com el famós HTTP de les pàgines web, FTP, entre d'altres.

Als annexos hi ha informació més detallada sobre aquest protocol, tanmateix, no és imprescindible.

### Funcionament

Per poder funcionar, abans de poder enviar cap mena d'informació al receptor, el protocol crea una connexió entre els dos. Aquesta connexió es mantindrà fins que el procés de transferència termini. Si es vol enviar un altre missatge després d'un temps determinat al mateix destinatari, aquesta connexió s'haurà de crear una altra vegada.

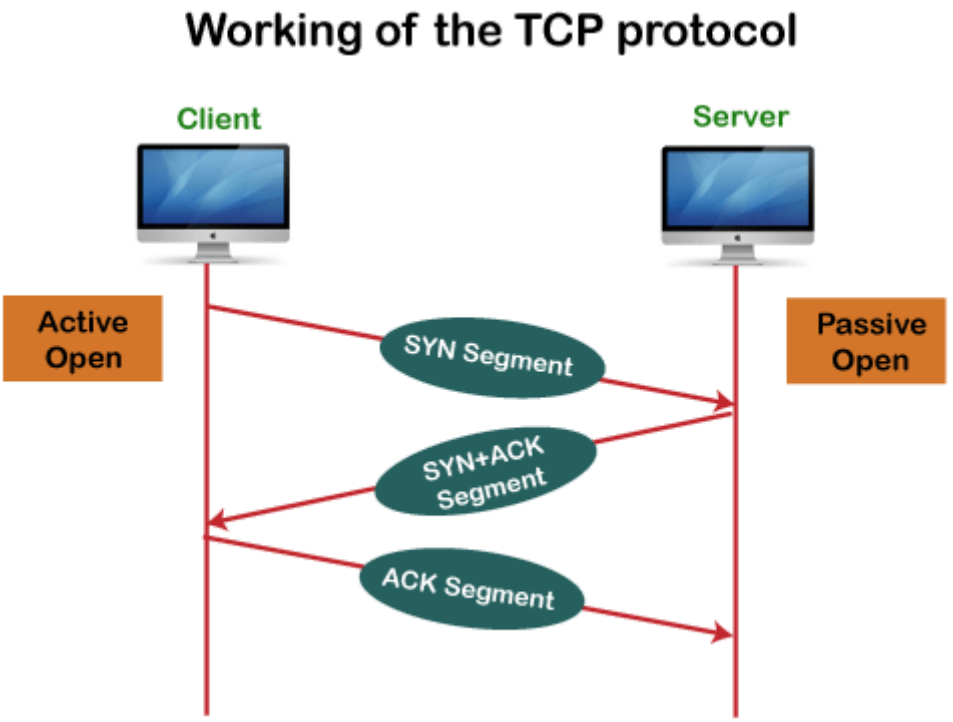



*Figura* 2. Imatge amb el funcionament de TCP. Es pot veure com el destinatari (servidor) i l'emissor (client) obren una connexió i a continuació començarien a enviar-se informació.

TCP divideix el missatge que es vol enviar en diferents trossos o paquets per fer-los més manejables pel protocol i tenir una transmissió més eficient. Addicionalment, també organitza i maneja els paquets que s'envien per la xarxa, amb el fi de fer-los arribar al destinatari de la manera més eficient possible.

TCP també ofereix un altre servei crucial, que tots els paquets tramesos arribin sense excepció al receptor. I si n'hi hagués algun cas en el qual un d'aquests paquets no arribes per qualsevol motiu (interferències, mala connexió dels cables, etc.), la connexió creada permetrà enviar-li al destinatari el paquet perdut sense complicacions.

### Avantatges i desavantatges

Tot i tenir l'avantatja de poder reenviar els paquets perduts de manera ràpida i senzilla, gràcies a tenir una connexió sempre oberta entre emissor i receptor, i poder organitzar la xarxa per a fer la transmissió més eficaç, és justament això el que fa que TCP tingui el seu major defecte, la velocitat de transmissió. Aquest temps que es passa organitzant i assegurant-se que els paquets arriben al destinatari és temps en el qual un nou paquet no s'envia al destinatari.

No obstant això, aquesta pèrdua de velocitat no és notable en la gran majoria dels casos. A l'hora de navegar per pàgines web o d'enviar missatges en qualsevol aplicació, no notaràs aquesta velocitat de transmissió reduïda. Gràcies a això, TCP es pot centrar més en la fiabilitat sense preocupar-se tant de la velocitat.

Per una altra banda, aquesta rapidesa perduda sí que es nota quan el temps de transmissió ha de ser mínim. Un exemple d'una situació com aquesta serien les plataformes de retransmissió de vídeo i àudio, com YouTube o Twitch, on la velocitat de transmissió ha de ser la més ràpida possible per evitar que l'experiència d'usuari se senti lenta i inconsistent. En aquests casos TCP no és la millor opció, ja que la cosa primordial és la rapidesa.

### Aplicacions

Com he dit abans, TCP és el protocol més utilitzat en tot l'internet. I s'utilitza quan es vol enviar informació de manera fiable.

Pot ser utilitzat directament o a través de la gran quantitat de protocols que es basen en ell, els quals afegeixen funcionalitats específiques depenent de l'entorn d'utilització de cada un. Per exemple, els protocols HTTP i HTTPS, són utilitzats més que res en pàgines web, i estan basats en TCP.

## UDP

UDP és un protocol que, igual que TCP, pertany al nivell 4, el Nivell de Transport. Aquest és una alternativa més simple i ràpida que TCP, que destaca per la seva gran velocitat a l'hora de transmetre informació. Així i tot, com que no té les funcionalitats de TCP (sense pèrdues de paquets, organització de les xarxes, etc.), el deixa propens a la pèrdua d'informació pel camí i a altres problemes.

Als annexos hi ha informació més detallada sobre aquest protocol, tanmateix, no és imprescindible.

### Funcionament

A diferència de TCP, UDP no obre ni manté una connexió constant amb el receptor, sinó que directament li envia els paquets a aquest sense previ avís i sense que el destinatari els aprovi.

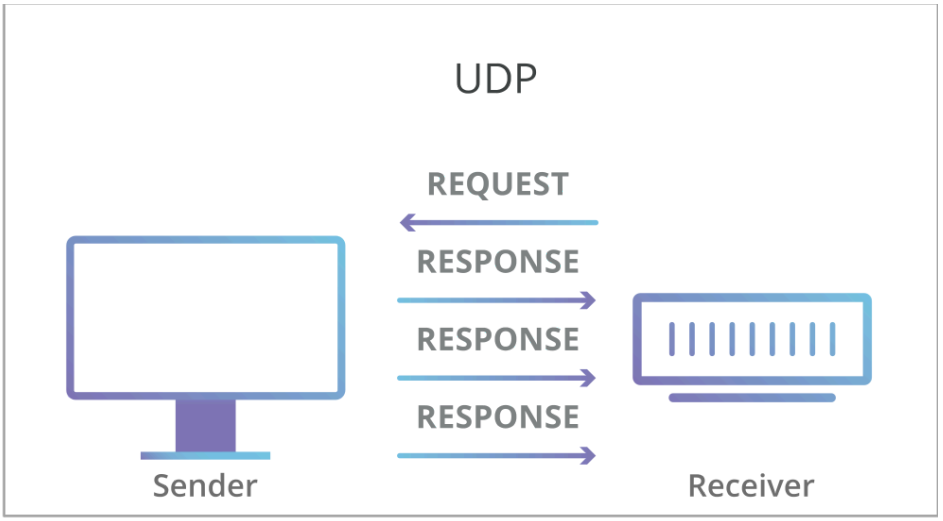



*Figura* 3. Imatge que explica el funcionament d'UDP, on els *receiver* (receptor) rep respostes o missatges del *sender* (emissor), sense obrir cap classe de connexió.

UDP divideix el missatge que es vol enviar en diferents paquets, de la mateixa forma que TCP, però no assegura l'enviament exitós d'aquests. Això vol dir que, si per qualsevol motiu un paquet no arriba al receptor, UDP no farà res per solucionar-ho.

A més, UDP tampoc organitza la xarxa quan hi ha una gran quantitat de paquets sent transmesos, a conseqüència d'això, es poden ocasionar retards en l'emissió d'aquests paquets si la xarxa està molt congestionada.

## Avantatges i desavantatges

El principal avantatge d'UDP és la seva gran velocitat de transmissió, el que el fa ideal en els casos on TCP no és prou ràpid. A part de la velocitat, la simplicitat d'utilització també és un factor a favor d'UDP, no fa falta que els clients es posin d'acord amb la informació que s'enviarà, només fa falta enviar-la i ja.

Com s'ha comentat abans, el gran problema d'aquest protocol és que no garanteix l'enviament exitós de la informació que es vol transmetre. Això fa que no sigui bona idea enviar informació de gran valor amb aquest protocol, ja que es podria perdre i no arribar mai al destinatari.

Un altre gran problema que té UDP és que pot patir atacs DDoS[2], el que pot ser una vulnerabilitat molt perillosa. Aquest fenomen es presenta a causa del nul establiment d'una connexió per a poder enviar paquets. El que ocasiona que els atacants puguin inundar un servidor o servei UDP amb paquets en blanc o amb informació fraudulenta. I tot sense haver d'obtenir primer el permís d'aquest servidor o servei, com sí que s'hauria de fer amb TCP.

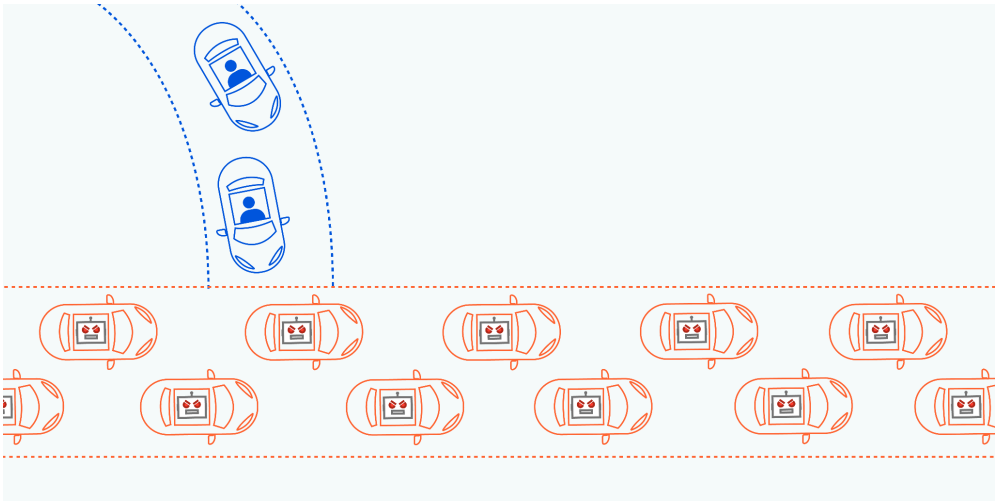


*Figura* 4. Metàfora d'un atac DDoS, que seria l'equivalent a embussar de manera intencionada una autopista amb cotxes que no tenen cap funció per tal de perjudicar els conductors legítims.

Tot això provoca que aquest servidor o servei hagi de revisar una gran quantitat de peticions invàlides, fent que la seva utilització de recursos (capacitat de processament) s'elevi exponencialment i, conseqüentment, col·lapsi.

[2] ***DDoS (denegació de serveis distribuïts):*** *Atac maliciós amb l'objectiu d'interrompre el funcionament d'un servei o xarxa mitjançat una sobrecàrrega de connexions de dispositius infectats.*

### Aplicacions

Aquest protocol, gràcies a la seva increïble velocitat de transmissió, s'utilitza en casos on la rapidesa és crucial. Alguns d'aquests casos serien:

- **La transmissió de vídeo i àudio en plataformes com YouTube o Twitch:** En aquest tipus de plataforma la velocitat de transmissió és clau per no fer que l'usuari hagi d'esperar molt de temps cada vegada que vol veure un vídeo. En aquests casos, com cada segon d'un vídeo està format normalment per 24 imatges, si es perd una imatge o dues, no passa res, ja que l'usuari final no ho nota.
- **Aplicacions de comunicació basades en l'internet:** Aplicacions com Microsoft Teams, Zoom, Discord, WhatsApp, entre d'altres, requereixen que les seves connexions siguin molt ràpides a l'hora de parlar amb algú o fer una videotrucada. En aquestes circumstàncies és preferible una conversació d'una qualitat més baixa, però ràpida, a una molt clara, però havent-hi un endarreriment bastant notable.

### Conclusions

En conclusió, tot i els seus desavantatges i la vulnerabilitat cap a atacs DDoS, UDP és un protocol molt utilitzat en una gran varietat d'indústries per la seva rapidesa i simplicitat a l'hora d'enviar informació. Des de videojocs en línia fins a pàgines de retransmissió de vídeos, on la velocitat sigui un factor crucial, UDP és una, sinó la millor opció per a utilitzar.

## Comparació de la Velocitat de transferència

Per poder comparar les velocitats de transmissió dels dos protocols que s'han descrit, s'analitzaran per separat els rendiments de cada un relacionat amb la velocitat en els diferents escenaris que poden passar a una xarxa.

### Anàlisi de dades

Primer s'analitzarà com es comporten els protocols segons la pèrdua d'aquests o *Packet Loss Rate (%)*[3]. Això vol dir que s'analitzaran diferents resultats que dependran d'aquest factor. Cal mencionar que els resultats s'han extret utilitzant **Miniset**, un servei que et permet crear una xarxa virtual on pots controlar les variables d'una xarxa, com per exemple la pèrdua de paquets, entre d'altres. Aquest tipus de programa facilita la tasca de recollir aquests valors, ja que d'una altra manera seria molt complicat i imprecís pel fet d'haver de controlar de forma tan mil·limètrica tots aquests factors. És important mencionar que en aquesta secció cada paquet enviat té una mida de 1000 bytes o 8000 bits[4].

[3] ***Pèrdua de paquets:*** *Percentatge de paquets que no arriben al seu destinatari en una transmissió per diversos factors externs, calculat com (paquets perduts / paquets enviats * 100).*

[4] ***Bit:*** *Dígit binari que pot tenir el valor de 0 o 1. El bit és la unitat mínima d'informació en informàtica, de la qual deriven altres unitats com el byte (8 bits), el megabyte (10^6 bytes), etc.*

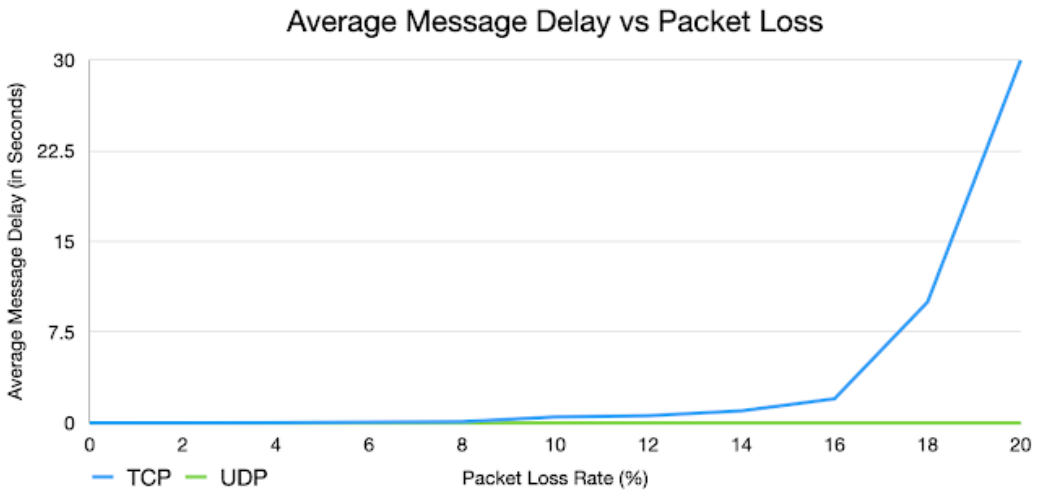



*Figura* 5. Gràfic del retard dels missatges en arribar al destinatari (en segons) depenent de la pèrdua de paquets.

Com es pot apreciar, UDP té un retard constant a l'hora enviar els paquets, això és perquè aquest no comprova si els paquets enviats arriben al destinatari o no, i no fa res si es perden. Això ocasiona que no augmenti el retard en enviar informació, és a dir, que aquest és contant, per molt alta que sigui la taxa de pèrdua. De fet, aquest retard és quasi 0, sent aquest exactament 160 μs o 0,00016 s.

Per una altra part, TCP té un retard cada vegada més gran, és un creixement exponencial. Això és deu al fet que TCP, a diferència d'UDP, sí que s'assegura que arribin els paquets al destinatari. El que fa que quan no arribin una gran quantitat d'aquests, TCP els hagi d'enviar una altra vegada fins que arribin abans d'enviar un nou missatge.

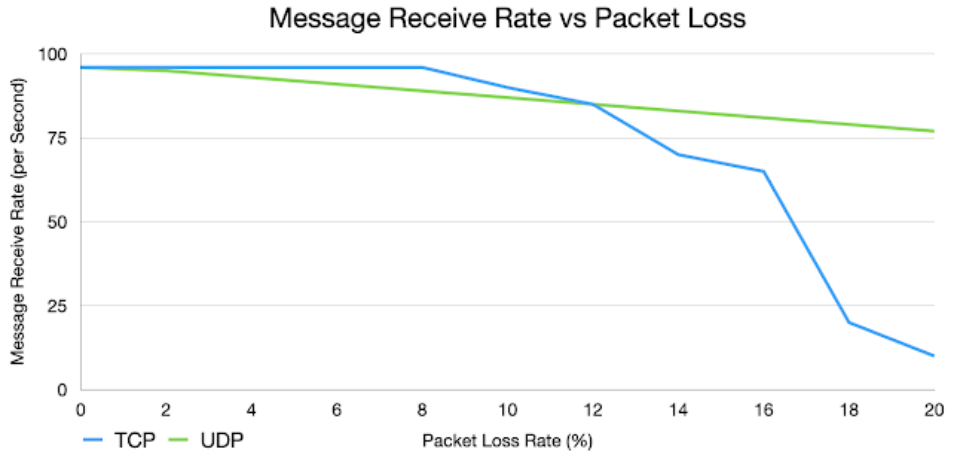



*Figura* 6. Quantitat de missatges rebuts cada segon segons la pèrdua de paquets.

UDP rep els mateixos missatges per segon que TCP amb una pèrdua de paquets del 0%, més específicament, en rep 96 per segon. Quan aquesta pèrdua va augmentant, UDP té una disminució linear del seu rendiment, ja que només es reben els missatges no es perden, com naturalment s'esperaria que passés.

En canvi, TCP es comporta d'una manera diferent, fins al 10% de pèrdua aquest aconsegueix mantenir els 96 missatges per segon, gràcies al seu sistema d'enviament de paquets que s'assegura que arribin al destinatari i al seu bon sistema de gestionament de xarxes. Tot i això, a partir d'aquest 10% de pèrdua, comença a baixar l'eficàcia fins a arribar a creuar-se amb UDP als 85 missatges per segon, quan la pèrdua és del 12%. A partir d'aquest punt, el rendiment de TCP es desploma fins a arribar als 10 missatges per segons al 20% de pèrdua de paquets. Aquesta gran baixada és deu a què el sistema d'enviament exitós de TCP es torna massa lent en aquestes condicions.

A continuació s'analitzen els protocols segons la quantitat d'informació que s'enviï al destinatari. En aquest cas, els resultats s'han obtingut mitjançant una simulació amb el programa **NS3 Simulator**.

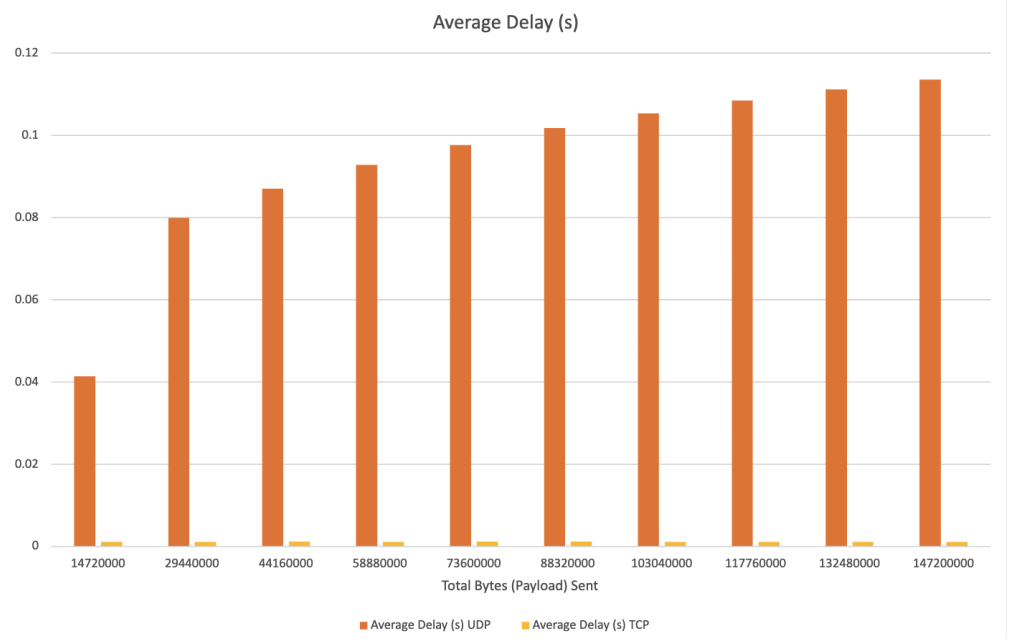



*Figura* 7. Temps que tarda el missatge a arribar al destinatari (en ms) segons la mida del missatge (en bytes).

En aquest cas TCP ha tingut un rendiment molt més alt que UDP. Aquesta gran diferència es deu a l'excel·lent sistema de descongestionament que posseeix TCP, el que li permet ser molt més eficient a l'hora d'enviar grans quantitats d'informació, tal com es pot veure a aquest gràfic.

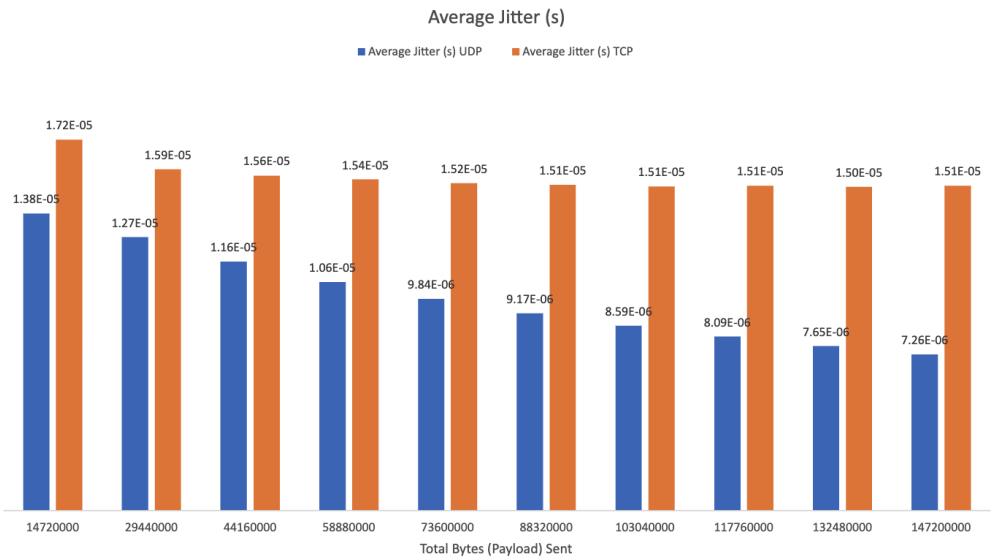



*Figura* 8. Retard que hi ha abans d'enviar el paquet (en segons) segons la mida d'aquest (en bytes).

En aquesta ocasió, UDP té el menor retard abans d'enviar la informació. Això és perquè com que no ha d'establir una connexió amb el destinatari per poder enviar paquets ni tampoc haver de mirar la xarxa per veure si està congestionada o no, dona com a resultat que TCP vagi més lent a l'hora d'haver de trametre un missatge. És important aclarir que aquest temps no és el que tarda el paquet a arribar al receptor, és el temps en el qual el protocol prepara el missatge per posteriorment ser enviat, durant aquest temps el paquet encara no s'ha enviat i segueix en el dispositiu de l'emissor.

## Conclusions

En conclusió, UDP és més ràpid que TCP en quasi tots els aspectes, ja sigui en el retard que hi ha abans d'enviar els paquets, el temps que tarden a arribar els paquets segons el percentatge de pèrdua o en la quantitat de paquets per segons depenent també de la pèrdua d'aquests. Aquesta diferència arriba a ser molt elevada depenent de les condicions de la xarxa. En l'únic aspecte en el qual és superat en velocitat és a l'hora d'haver d'enviar grans quantitats d'informació en un període de temps molt limitat, tal com s'ha vist en la tercera anàlisi.

## Comparació de la seguretat i integritat de les dades

Per poder comparar la seguretat de transmissió dels dos protocols, igual que a l'altra comparació, s'analitzaran per separat per veure les prestacions respecte a seguretat, per a després poder comparar-los.

### Anàlisi

Abans de començar, és molt important mencionar que cap dels protocols encripten els missatges per defecte. Tanmateix, això no vol dir que no tinguin la capacitat de fer-ho, ja que es poden configurar perquè ho apliquin. Això sí, les comunicacions aniran una mica més lentes pel fet d'haver d'encriptar i desencriptar el missatge.

En el cas de TCP hi ha 3 opcions principals per a poder encriptar els paquets que s'envien:

- ***Transport Layer Security (TLS)* o *Secure Sockets Layer (SSL):*** Aquestes dues tecnologies són una de les opcions preferents per a l'encriptació de TCP. TLS és simplement una versió més moderna i sofisticada que SSL, però totes dues tenen la mateixa funció. Són bastant simples d'utilitzar i molt segures, ja que en cada comunicació hi ha d'haver una autenticació per part dels dos dispositius connectats.
- ***External Security Manager (ESM):*** També està l'opció d'utilitzar un ESM, els quals són serveis oferts per empreses que donen la mateixa funcionalitat que SSL o TLS. N'hi ha de molts tipus d'ESM, com el *RACF Security Server* d'IBM o el *McAfee Enterprise Security Manager* de l'antivirus McAfee.
- ***Lightweight Directory Access Protocol (LDAP):*** Com a última opció, pots utilitzar aquest protocol especial de la capa 7 (capa d'aplicació) que també et permet tenir les mateixes prestacions que els altres sistemes d'encriptació.

En el cas d'UDP també hi ha opcions per poder encriptar paquets:

- ***Datagram Transport Layer Security (DTLS):*** És una de les opcions més utilitzades a l'hora d'encriptar paquets amb d'UDP. Utilitza la tecnologia OpenSSL, que seria la mateixa que el SSL, però oberta a tot el públic.
- ***Virtual private network (VPN) i content distribution network (CDN):*** Aquestes dues tecnologies permeten defensar als servidors UDP del perill dels atacs DDOS que es poden fer a causa dels defectes d'aquest protocol. A més de fer les connexions més segures i privades.

Si alguna d'aquestes opcions més populars no s'ajusten als interessos d'una l'empresa o desenvolupador, també es poden encriptar i posar mètodes de seguretat de manera manual i personalitzada, és a dir, creats per ells.

### Conclusions

En conclusió, la seguretat de les comunicacions amb els protocols comentats, tot i ser important, no és tan rellevant, ja que encara que ni TCP ni UDP encriptin les comunicacions per defecte, hi ha eines que es poden utilitzar per encriptar els missatges. A part que es poden crear eines de seguretat de forma manual per complir amb les necessitats d'un individu o organització. Per aquests motius, no es tomarà en compte la seguretat en el treball.

## Comparació de la fiabilitat en entorns de xarxa

En aquest apartat es compararan els protocols estudiats segons el seu grau fiabilitat a l'hora d'enviar informació a un destinatari, i com afecta aquesta fiabilitat al seu funcionament i entorn d'utilització.

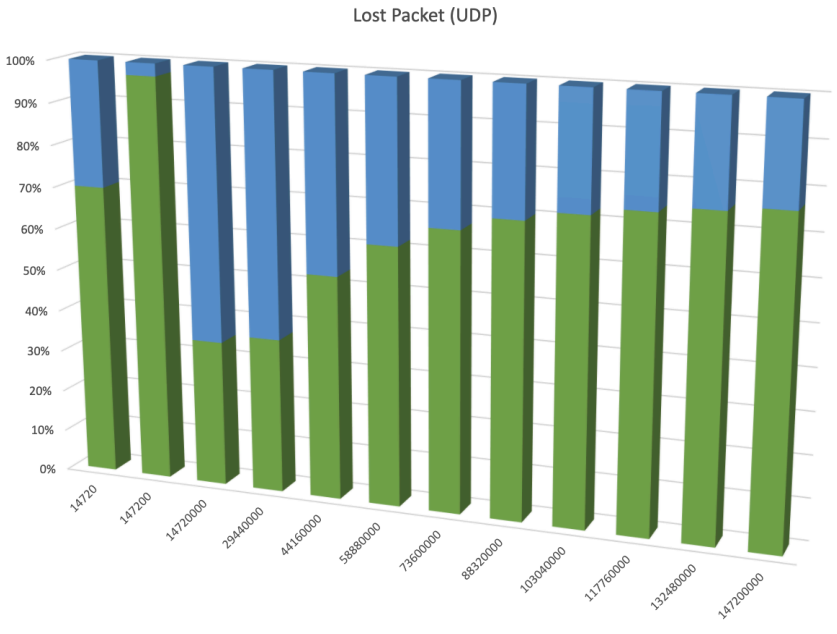
### Anàlisi

En primer lloc, TCP té una fiabilitat excel·lent gràcies al fet que té s'assegura que qualsevol paquet enviat amb aquest protocol i, evidentment dels protocols que es basen en aquest, arribi al seu destinatari.

Addicionalment, TCP també organitza el tràfic de paquets que són transmesos per la xarxa, el que fa que aquests no només arribin més ràpidament al seu destinatari, sinó que també d'una manera més eficaç.

TCP també ofereix un sistema de correcció dels possibles errors que puguin sortir a l'hora d'enviar qualsevol classe d'informació. El que vol dir que si un paquet enviat és alterat per qualsevol motiu (interferències, etc.), el que ocasiona que la informació enviada no sigui la mateixa que la que arriba al destinatari, TCP sigui capaç de corregir aquests errors.

D'una altra banda, UDP no s'assegura que els paquets tramesos arribin al receptor, si algun es perd pel camí, mai serà retransmès una altra vegada. Aquest comportament es pot veure clarament en el gràfic següent, on s'analitza la pèrdua de paquets d'UDP segons la dimensió de la informació enviada.




| | 14720 | 147200 | 14720000 | 29440000 | 44160000 | 58880000 | 73600000 | 88320000 | 103040000 | 117760000 | 132480000 | 147200000 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rx Packets | 3 | 3 | 6499 | 12581 | 13939 | 15295 | 16650 | 18008 | 19366 | 20720 | 22075 | 23432 |
| Lost packets | 7 | 97 | 3501 | 7419 | 16061 | 24705 | 33350 | 41992 | 50634 | 59280 | 67925 | 76568 |

*Figura* 9. Pèrdua de paquets d'UDP segons la dimensió del missatge que s'envia. Verd: paquets perduts, Blau: Paquets que arriben al destinatari.

Més específicament, es pot observar que com més gran sigui el missatge transmès, més gran és la pèrdua de paquets que experimenta UDP. Això fa que UDP no sigui un bon candidat a l'hora de transmetre grans quantitats de dades.

A diferència de TCP, UDP no té un control sobre la gestió del tràfic de paquets en la xarxa, el que fa que en situacions concretes on s'envia molta informació de cop, l'eficàcia d'aquestes transmissions sigui bastant baixa.

Finalment, UDP tampoc té pròpiament un sistema de correcció d'errors, sí que és capaç de detectar errors en els paquets transmesos, però simplement els descarta sense solucionar-los.

## Conclusions

En conclusió, TCP és, sense cap dubte, immensament més fiable que UDP, gràcies als seus sistemes de correcció d'errors, organització de tràfic i d'enviament exitós de tots els paquets. Aquesta excel·lent fiabilitat fa que altres protocols amb aquestes necessitats es basin en ell per al seu funcionament.

UDP, per una altra part, no és un protocol especialment fiable, com s'ha vist a l'anàlisi. Això fa que no sigui utilitzat per a molts àmbits on fa falta que la informació arribi sense errors i de manera segura.

# Funcionament d’un videojoc en línia

## Introducció al videojoc en línia

Un videojoc en línia és com qualsevol altre videojoc, amb l’única diferència que en aquest la persona que està jugant interacciona amb altres individus, siguin amics o persones desconegudes, que poden estar a una distància molt elevada. Aquestes interaccions poden ser a través de missatges, jugant en la mateixa partida, etc.

A la part pràctica d’aquest treball faré un videojoc en línia amb el fi d’utilitzar en un cas real els protocols investigats, per posteriorment poder analitzar-los i comparar-los. Aquest videojoc en línia serà molt bàsic visualment, ja que l’objectiu és analitzar el rendiment dels protocols, no centrar-se en l’apartat visual.

Aquest videojoc estarà fet amb *Unity*, el qual és un motor gràfic[5] que s’especialitza en la creació de videojocs en 3D. Aquesta elecció no influirà en la implementació dels protocols, perquè Unity només serà utilitzat per a l’apartat visual del videojoc. Tot el que té a veure amb les comunicacions es farà fora de Unity i podria ser utilitzat en altres motors gràfics. Així que de la mateixa manera que utilitzo Unity, es podria utilitzar qualsevol altre motor gràfic, com Unreal Engine, Godot, Game Maker, etc. He escollit Unity principalment perquè estic familiaritzat amb ell pel fet d’haver-lo utilitzat amb anterioritat.

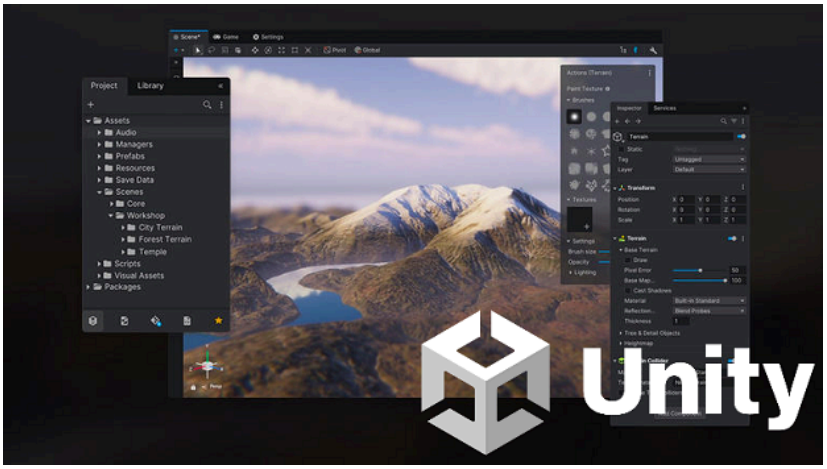



*Figura* 10. El motor de videojocs Unity.

[5] ***Motor gràfic o de videojocs:*** *Programa que proporciona eines per a la creació de videojocs, incloent-hi sistemes de física, dibuix en pantalla dels diferents elements del joc, etc.*

El videojoc serà, més concretament, un *First Person Shooter (FPS)*, un tipus de joc en primera persona fet en 3D que simula el fet d'utilitzar armes de foc. Però, a diferència d'aquest tipus de joc, en aquest treball no s'utilitzaran armes de foc, sinó de paintball. Aquest és un esport on les armes disparen pilotes de pintura. Els jugadors es podran moure i saltar pel mapa, així com utilitzar armes de paintball per intentar eliminar als altres jugadors.

*Figura* 11. Persona jugant al Paintball.

Un exemple d'aquest tipus de joc seria el *Paintball War,* creat per *ShadowWolf Games*, un videojoc on controles a una persona que està participant en una partida de paintball. El teu objectiu és acabar amb l'equip contrari per guanyar la partida. El videojoc que faré serà semblant a aquest joc, però més simple pel que fa al contingut o a l'apartat visual, perquè el que interessa és el funcionament dels protocols de comunicació, no el joc en si.

Figura 12. Captura de pantalla del videojoc *Paintball War*.

Aquest tipus de videojoc és perfecte per a la comparació de protocols, ja que en un joc tan frenètic, on els reflecteixes ho son tot, la rapidesa, fiabilitat i eficàcia per transmetre la informació és essencial perquè els jugadors tinguin una bona experiència.

## Com funciona un videojoc en línia

Abans de començar amb la creació del videojoc en línia, ha de quedar ben clar com funcionen aquests i com es transmet la informació entre els diferents dispositius informàtics.

Als videojocs en línia hi ha dos tipus de dispositius:

- **El client:** Aquest és l'ordinador de l'usuari que juga al videojoc. És a dir, els jugadors.
- **El servidor:** Aquest ordinador s'encarrega de proporcionar una xarxa als clients per a poder realitzar interaccions entre ells.

El funcionament d'aquest tipus de videojoc és el següent: Imaginem que un client vol moure el seu personatge cap a la dreta, aquest li envia aquesta acció al servidor, el servidor mou el seu personatge seguint les indicacions del jugador i li envia aquesta acció a tots els altres clients, que actualitzaran la posició del personatge en les seves partides. Els clients mai es comunicaran directament entre ells, tota la informació passarà primer pel servidor. Aquest tipus de sistema és el que es coneix com a **arquitectura client-servidor**.

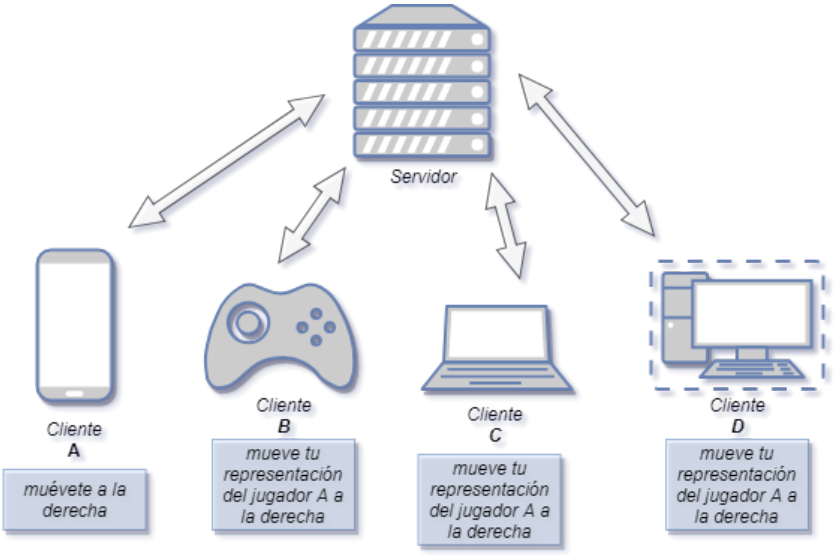



*Figura* 13. Esquema del funcionament d'un videojoc en línia.

És molt important destacar que el servidor és qui té la màxima autoritat, i aquest valida totes les accions que volen fer els jugadors. És a dir, quan un jugador mou el seu personatge cap a la dreta, el client no és qui realment mou al seu jugador, sinó que li envia al servidor la intenció de voler moure's cap a la dreta, i és aquest qui calcula en quina posició quedaria el personatge. A continuació, li envia aquesta posició calculada a tots els clients perquè l'actualitzin, incloent-hi el jugador que ha enviat la petició.

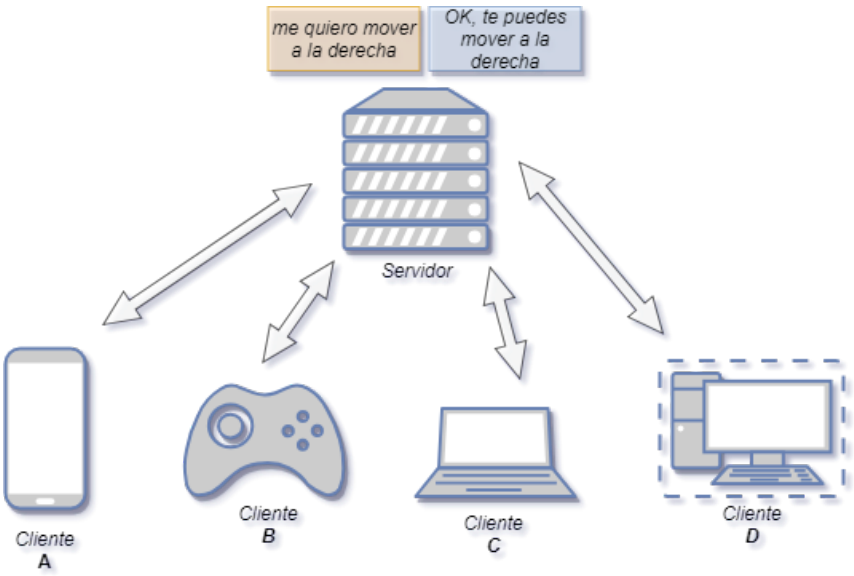



*Figura* 14. Esquema de l'autoritat que té el servidor sobre els clients.

Aquesta autoritat que té el servidor és crucial per al correcte funcionament d'un videojoc en línia, ja que és l'únic equip informàtic en el qual es pot confiar. Això és perquè qualsevol persona pot modificar el seu propi client. Si deixem que cada client calculi per si mateix les seves accions i se les enviï al servidor, hi haurà clients modificats que enviaran informació errònia o no congruent de manera intencionada amb el fi de donar una avantatja il·legítima als jugadors que han modificat el client sobre els altres jugadors lícits. Aquests avantatges poden ser diversos, però les més comunes serien: poder volar, moviment mil·limètric, invencibilitat, etc.

És a dir, un videojoc en línia es juga realment en el servidor, perquè el que s'està executant en aquest és, per dir-lo d'una manera, la partida real. Els clients simplement reprodueixen el que passa al servidor de la manera més ràpida possible perquè tot estigui molt ben sincronitzat amb el fi de què els jugadors no s'adonin d'aquest fet.

Aquest sistema de verificació per part del servidor farà que el videojoc sigui més lent. Ja que quan polses una tecla, l'acció ha d'anar al servidor, que calcularà el resultat, i tornar al teu client perquè s'actualitzi. Així i tot, és l'única manera d'assegurar el correcte funcionament del joc i evitar tramposos. Per sort n'hi ha tècniques que permeten tenir una millor experiència i que tot sigui més ràpid sense haver d'abandonar l'autoritat del servidor, però no es tocaran en aquest treball perquè no és l'objectiu d'aquest.

## Com s'implementaran els protocols al videojoc en línia

Per poder validar la hipòtesi plantejada es crearan dues versions del videojoc. Aquestes seran exactament iguals, exceptuant el protocol utilitzat per a les comunicacions. En una versió només s'utilitzarà TCP, i en l'altra UDP. D'aquesta manera serà més fàcil comparar l'eficàcia de cada protocol.

Per aconseguir implementar els protocols de comunicació al videojoc utilitzaré una llibreria[6] per al llenguatge de programació[7] C#, denominada *Riptide Networking*, creada per *Tom Weiland*. Aquesta llibreria permet fer ús dels protocols TCP i UDP en aquest llenguatge de programació. És una llibreria molt simple i personalitzable, és a dir, que només et dona les eines per enviar i rebre informació amb aquests dos protocols, i tota la resta que faci falta per a les comunicacions (quina informació enviar, sincronització, identificació de clients, etc.) s'ha de fer a part. Gràcies a això, em permetrà tenir el control de totes les comunicacions que passin al videojoc i analitzar totes les transmissions.

Realment es podria utilitzar qualsevol llibreria que et doni accés a aquests dos protocols, seria possible per exemple, utilitzar directament els protocols TCP i UDP que et dona Microsoft per al llenguatge de programació C#, o les seves respectives en els altres llenguatges de programació. *Riptide*, com altres llibreries, simplement fa que el procés de configuració d'aquests dos protocols sigui més simple i optimitzat que si s'hagués de fer des de zero.

Aquesta llibreria és *open-source* o de codi font obert, el que vol dir que qualsevol persona pot veure com està feta per dins i modificar-la al seu gust.
La pàgina oficial de *Ritptide* és la següent: https://riptide.tomweiland.net/manual/overview/about-riptide.html
I la pàgina on està el codi d'aquesta eina és la següent: https://github.com/RiptideNetworking/Riptide

*Figura* 15. Logotip de la llibreria *Riptide*.

[6] ***Llibreria:*** *Conjunt d'eines ja creades que permeten realitzar accions sense haver de crear-les des de zero. Com per exemple, les funcions trigonomètriques en una llibreria matemàtica.*
[7] ***Llenguatge de programació:*** *Eina que proporciona a l'informàtic l'habilitat de donar instruccions a una màquina de forma ordenada i entenedora per als éssers humans. Exemples: C++, C#, C, Python, Ruby, JavaScript, etc.*

# Treball de Camp: Videojoc en Línia

Abans de començar, tot el procés de creació del videojoc, així com els arxius d'aquest estan pujats a la meva compta de GitHub, un servei de núvol que et permet pujar projectes relacionats amb la programació. La pàgina està en angles, però simplement és per un tema de format. L'enllaç al projecte és el següent: https://github.com/Estikno/PaintballOnline

## Organització del videojoc

Primer de tot, s'ha d'explicar com s'organitzarà el projecte en si. Com s'ha vist amb anterioritat, l'arquitectura d'un videojoc en línia és la denominada client-servidor, i per a poder establir-la es crearan dos projectes de Unity separats.

- El primer projecte serà el **servidor**. En aquest projecte es portaran a terme totes les accions principals del videojoc, d'aquesta manera es podrà mantenir l'autoritat del servidor per sobre dels clients.
- El segon projecte serà el **client**. Aquest serà el programa que cada jugador iniciarà per poder jugar al videojoc en si. Les funcions principals del client seran transmetre el que passa al servidor i enviar-li a aquest les accions que vol fer el jugador.

Sí que és veritat que el servidor podria no ser un projecte de Unity com a tal, i que seria possible crear-lo fora d'aquest motor gràfic i fins i tot amb un altre llenguatge de programació. Ara bé, el fet de crear el servidor fora de Unity, independentment del llenguatge de programació que s'utilitzi, dificultaria innecessàriament la seva creació pel fet d'utilitzar dues eines distintes, cosa que no té sentit en aquest treball, ja que no és el seu objectiu.

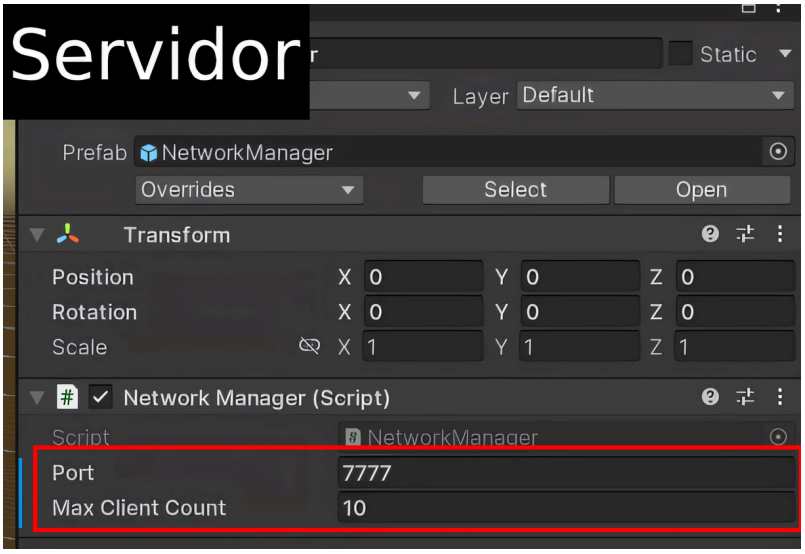
Com n'hi ha dos projectes diferents, a cada fotografia d'un d'aquests estarà indicat si es tracta del servidor o del client mitjançant una etiqueta a dalt a l'esquerra de cada captura de pantalla.


Exemple d'una captura de pantalla del projecte, en aquest cas, del servidor. Elaborat per David Didenco.

## Creació de l'administrador de xarxa

Amb el fi d'administrar d'una manera còmoda els protocols que s'utilitzaran per comunicar els clients amb el servidor, primer de tot crearé un administrador de xarxa. Aquest administrador serà l'encarregat d'organitzar les transmissions de paquets entre els clients i el servidor, així com de gestionar les connexions i desconnexions d'aquests, sincronitzar els diferents dispositius, etc. D'aquesta manera serà molt fàcil enviar paquets, ja que totes les funcionalitats estaran centralitzades en un lloc, sent fàcilment accessibles i utilitzables. Aquest administrador estarà present tant al servidor com al client. I estarà compost per un *script*[8] quasi idèntic en els dos casos.

Primer implementen el script en el servidor, que rebrà el nom de *"NetworkManager"* o administrador de xarxa en català. Dins d'aquest, a part d'altres coses defineixo les dades necessàries per a poder inicialitzar el servidor, que serien el port a on es farà la connexió i la quantitat màxima de jugadors simultanis que poden haver-hi. Addicionalment, també afegeixo les altres funcionalitats comentades al paràgraf anterior. Amb aquesta funcionalitat, ja podem inicialitzar un servidor, encara que no faci res més per ara.



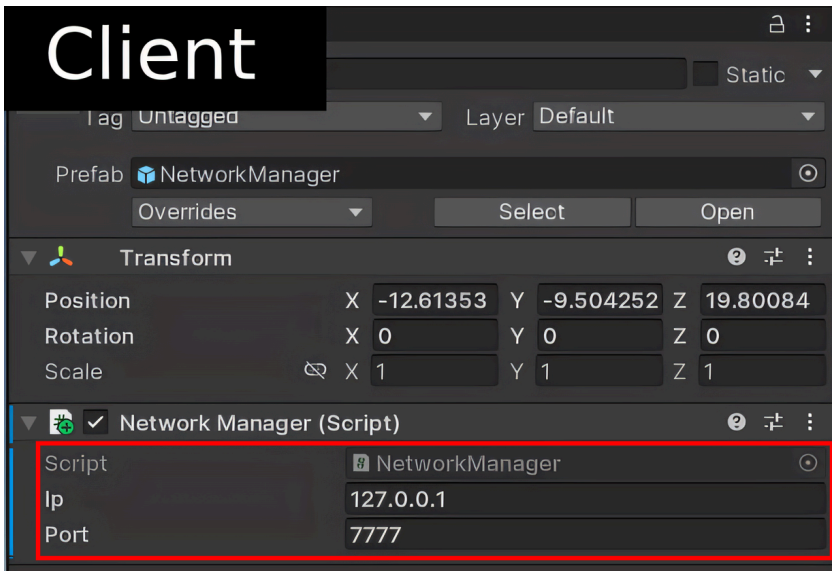

Informació necessària per poder iniciar el servidor. Elaborat per David Didenco.

Com s'ha explicat abans, el *script* és quasi el mateix per al client. El canvi més important és que ara és necessari declarar l'adreça IP[9] del servidor i el port escollit. Per la resta, el script és igual.

---

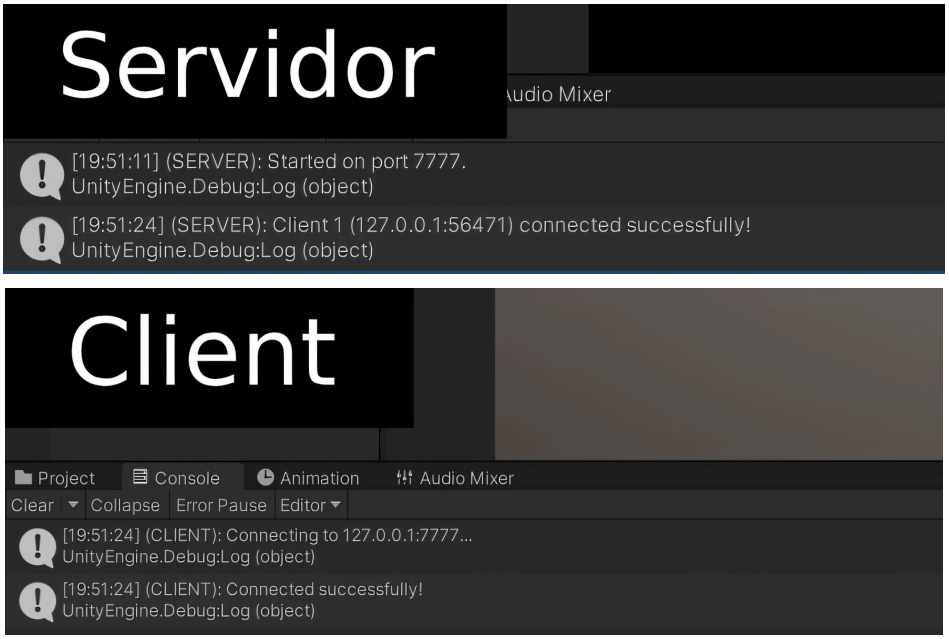
[8] ***Script:*** *Sèrie d'instruccions executables per una màquina o per un llenguatge de programació, normalment guardades com arxius separats. D'una manera més simplista, els scripts són com fulls de paper que contenen instruccions escrites en un llenguatge en programació.*

[9] ***Adreça IP****: Número d'identificació únic d'un dispositiu connectat a una xarxa, sigui privada (com la xarxa de casa teva) o pública (internet). Funcionament similar als números de telèfons, on cada dispositiu té la seva adreça.*




Informació necessària per a connectar-se al servidor. A les proves del videojoc l'adreça serà 127.0.0.1, que és el dispositiu local, però a l'hora d'executar el projecte en un cas real s'ha de canviar per la verdadera adreça IP del servidor. Elaborat per David Didenco.

Si iniciem tant el servidor com el client podrem veure un missatge que ens avisarà que tots dos s'han connectat correctament. No passa res més, però aquest petit pas ens mostra que hi ha una connexió exitosa entre client-servidor.

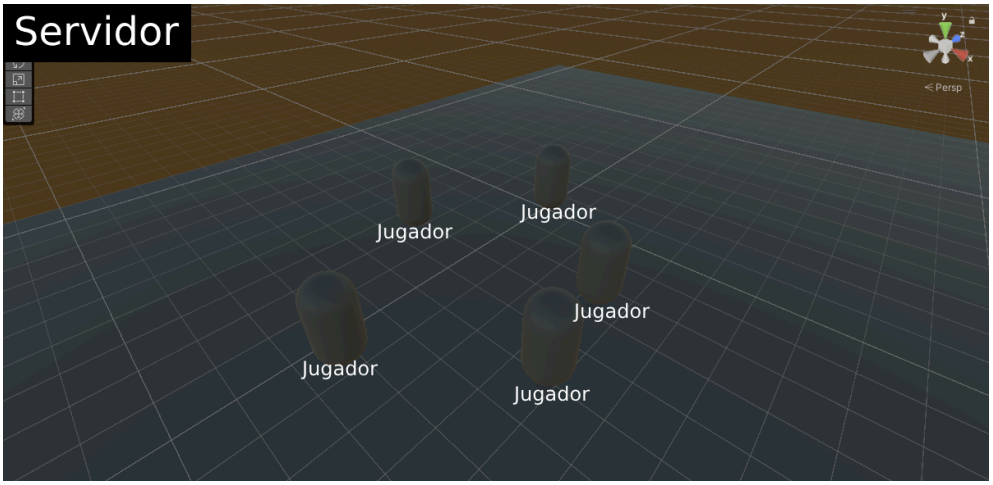



Connexió exitosa entre el servidor i un client, en aquest cas, amb UDP. Elaborat per David Didenco.

## Creació del jugador

Una vegada tenim l'administrador de xarxa funcionant, toca crear els jugadors. Aquesta tasca s'aconsegueix creant un objecte que té totes les accions que poden realitzar els clients (saltar, moure's, etc.) i clonant un de nou cada vegada que un usuari es connecta al servidor. Aquesta clonació ha de succeir tant al servidor com a tots els clients connectats. D'aquesta manera, els moviments de tots els jugadors estaran sincronitzats entre tots els dispositius.

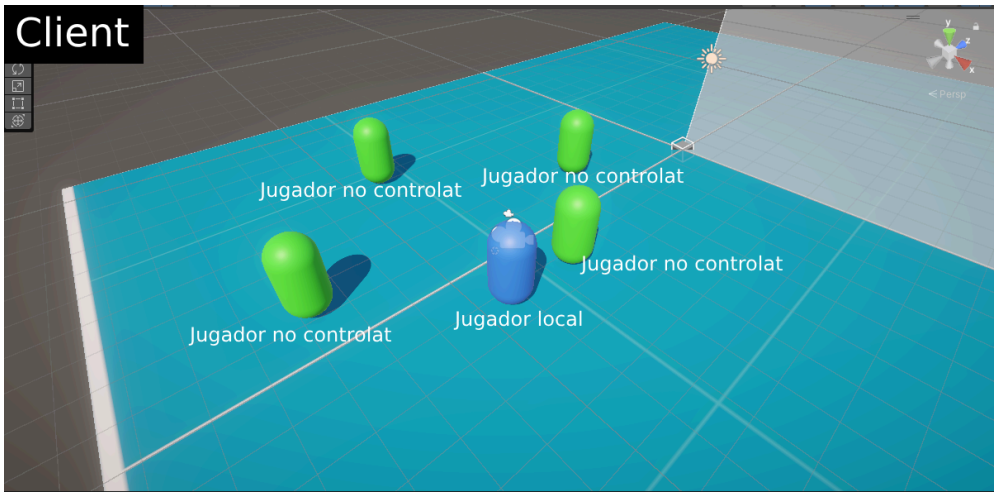
Des del punt de vista del servidor, tots els jugadors són exactament iguals, cada jugador li envia els moviments que vol fer, el servidor reprodueix les accions en el seu entorn, i passa els resultats del moviment a tots els clients perquè actualitzin les posicions de cada personatge, encara que no sigui el que controlin. Així, cada client podrà veure els moviments dels altres jugadors, igual que passa en qualsevol joc en línia.




5 jugadors connectats al servidor des de la perspectiva del servidor. Elaborat per David Didenco.

En canvi, des de la perspectiva dels clients, n'hi ha dos tipus de jugadors: el jugador local i els jugadors no controlats. Per una banda, en cada client només pot haver-hi un jugador local, ja que és aquest jugador el que el client controla mitjançant les ordres de la persona que juga al videojoc. D'una altra banda, poden haver-hi tants jugadors no controlats com clients connectats a la xarxa menys un, que seria el teu jugador local. Aquests jugadors no controlats són moguts mitjançant les ordres del servidor, sent aquestes ordres els moviments resultants de les accions de les altres persones de la xarxa.

Per poder distingir de manera visual entre el jugador local i els no controlats en el client, he pintat de color blau el local, i de color verd els no controlats.




5 jugadors connectats al servidor des del punt de vista d'un client. Hi ha 4 jugadors no controlats i un de local, que és el que controla aquest client. Elaborat per David Didenco.

## Implementació d'interpolació

Si executem el servidor i connectem uns quants clients al mateix servidor, podrem observar que el moviment dels jugadors no és fluid i se sent com si n'hi hagués un retard considerable entre cada acció. No solucionar aquest problema ocasionarà que el videojoc sigui injugable, independentment de si s'utilitza UDP o TCP. A més, aquesta sensació es veu intensificada quan controles a un jugador local, ja que a l'hora d'observar-ne a un altre no es nota tant.

Aquest fenomen es deu al fet que el servidor no pot enviar els moviments de tots els jugadors en temps real constantment, ja que això consumiria massa recursos i amplada de banda[10]. En comptes d'això, el servidor envia als clients les actualitzacions dels moviments de cada jugador un nombre determinat de vegades per segon. Aquest nombre és el *tickrate*[11] del videojoc.

[10] ***Amplada de banda:*** *Quantitat màxima de dades que es pot transmetre a través d'una connexió de xarxa en un temps determinat, normalment mesurada en bits/s, kilobytes/s, megabytes/s o gigabytes/s.*
[11] ***Tickrate:*** *Nombre de vegades per segon que un servidor envia als clients informació de qualsevol mena.*

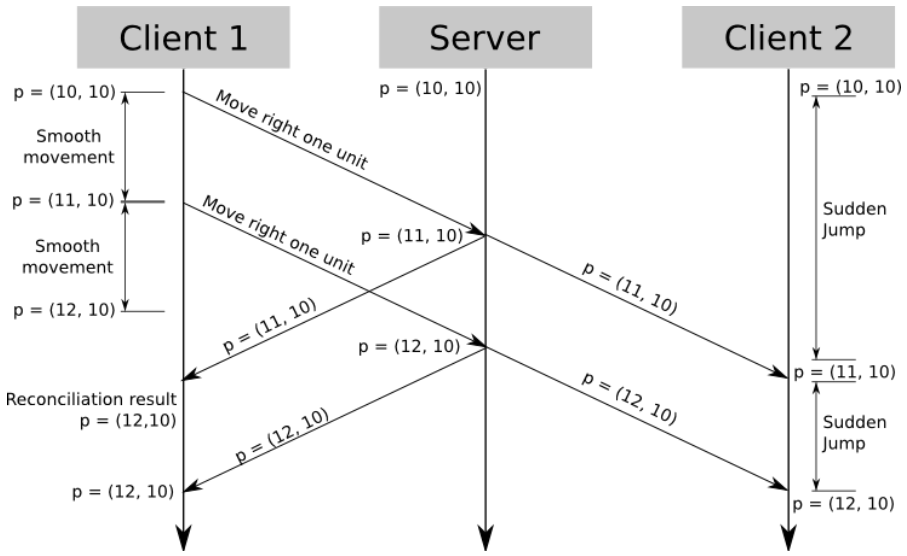



*Figura* 16. Falta de fluïdesa que se sent en els clients pel fet de no rebre missatges del servidor en cada instant. Si ens fixem en el client 2, podem veure com d'un moment a un altre el jugador 1 passa de la posició (11,10) a la (12,10), el que fa que no se senti fluid el videojoc.

Com més alt sigui el *tickrate*[12], menor serà el retard en l'execució de les accions, però això implicarà un major consum de recursos per part del servidor. Normalment, el *tickrate* està compres entre un valor de 20 i 120. Un valor superior no seria rendible a causa del gran consum de recursos que requeriria el servidor, i un valor inferior tampoc seria adequat perquè el retard seria massa elevat.

Tot i que un *tickrate* de 60 o fins i tot de 120 pugui semblar alt, no és suficient per fer que el joc es percebi fluid. Per aconseguir aquesta sensació de fluïdesa, es realitza una aproximació de les posicions dels jugadors enviades pel servidor als clients. Aquesta tècnica es coneix com a **interpolació**.

La idea darrere d'aquesta solució és la següent: suposem que el servidor té un **tickrate** de 40. Això vol dir que, com a màxim, el servidor ens enviarà 40 posicions diferents dels altres jugadors cada segon. Aquest màxim depèn del protocol utilitzat, ja que pot ser que no arribin les 40 posicions. El que fa cada client és mostrar una aproximació del que ha passat entre les posicions que rep de cada jugador.

La interpolació té altres aplicacions a part d'aproximar els moviments dels jugadors, també pot ser utilitzada per aproximar qualsevol objecte en moviment, així com per estimar rotacions i canvis de dimensions.

[12] ***Tickrate:*** *Nombre de vegades per segon que un servidor envia als clients informació de qualsevol mena.*

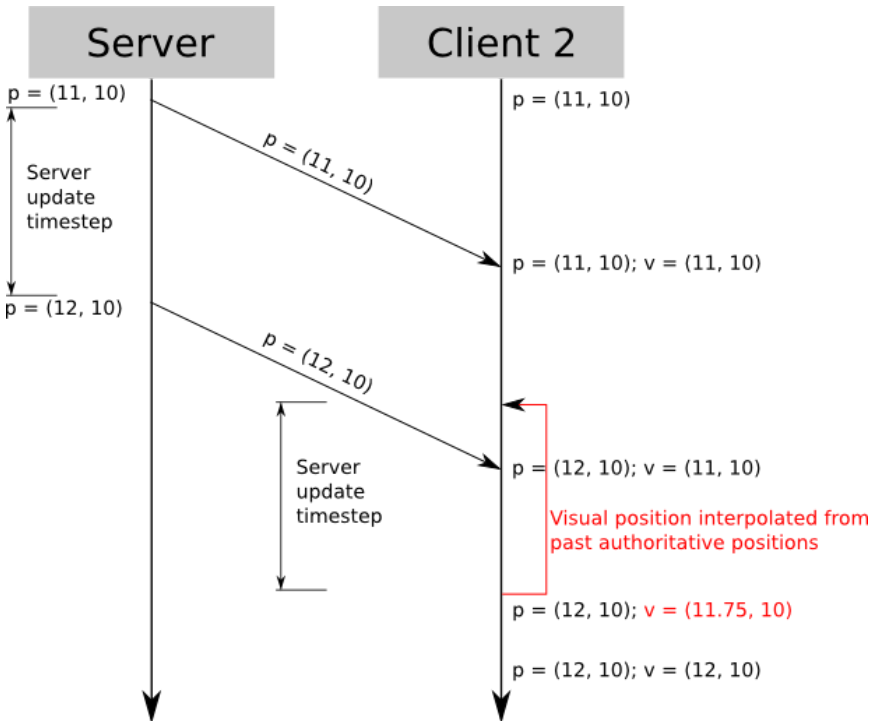



*Figura* 17. Exemple del funcionament de la interpolació. Ara el client 2 mostra els moviments del jugador 1 des de la posició (11,10) fins a la (12,10), el que fa que el videojoc se senti fluid.

Per exemple, si en un moment determinat rebem la informació que el jugador 1 es troba en una posició específica i, 100 mil·lisegons més tard, ens arriba una nova posició que indica que s'ha desplaçat lleugerament cap a la dreta, el que es visualitzarà en cada client és una representació del moviment que ha tingut lloc entre aquestes dues posicions. Això significa que el jugador 1 es mostrarà movent-se de la primera posició a la segona. És important subratllar que aquestes aproximacions o interpretacions del moviment només es produeixen en els clients; el servidor no aplica cap mena d'interpolació.

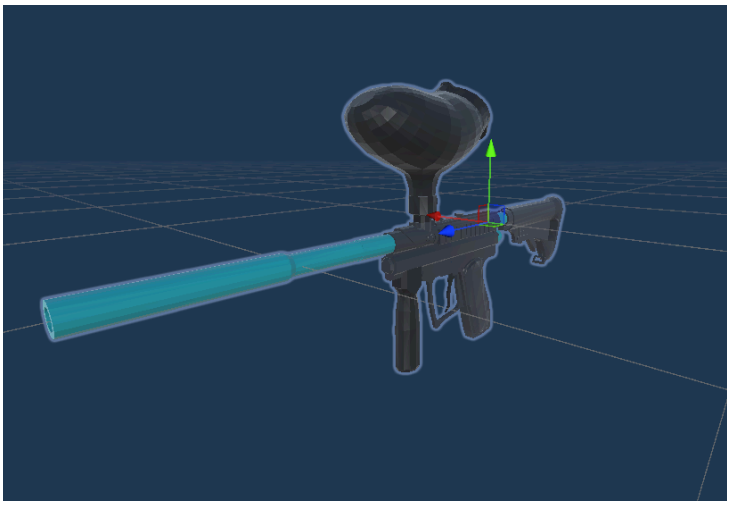
En aquest projecte concret, utilitzaré un *tickrate* de 40, ja que és similar al que utilitzen la gran majoria de videojocs en línia. Aquest valor no és ni massa petit ni massa gran, cosa que permet que el servidor no consumeixi excessius recursos, alhora que manté un bon equilibri de fluïdesa gràcies, en gran part, a la interpolació.

## Creació de l'arma de paintball

La creació de l'arma de paintball és bastant semblant a la del jugador. Cada jugador tindrà només una arma i aquesta serà clonada de la mateixa manera que els jugadors. Fent que tu només puguis moure la teva arma, alhora que veus en quina direcció els altres jugadors mouen la seva.

Com s'ha explicat amb anterioritat el servidor és qui realment realitza les accions dels jugadors per un tema de seguretat. Tanmateix, en el cas de disparar, com el projectil una vegada no canvia de direcció ni de velocitat, cada client pot calcular la trajectòria de cada projectil per ell mateix, fent que el servidor no hagi d'enviar aquesta informació, el que millorarà el rendiment d'aquest.

Sí que és veritat que en aquest cas, un client podria modificar la trajectòria dels projectils perquè vagin a on vulgui, però realment no serviria per a res, ja que és el servidor qui comunica dels efectes del projectil disparat. És a dir, si un jugador dispara una bala de pintura a un altre, encara que un client modifiqui la trajectòria d'aquesta, el servidor calcularà per ell mateix la trajectòria real i si al final li dona a l'altre jugador, enviarà als clients que tal jugador ha sigut eliminat, fent que aquesta modificació il·legítima sigui inútil.



Arma de paintball en el client. Elaborat per David Didenco.

## Control de la salut del personatge

Cada projectil en el videojoc causa una quantitat de dany determinada cap als jugadors. Això vol dir que si per exemple un jugador té 100 punts de vida i cada projectil treu 25 punts per impacte, el jugador serà eliminat quan li donin 4 projectils.

Per poder implementar aquest sistema, la vida dels jugadors sempre es calcularà en el servidor per evitar que els jugadors facin trampes modificant la seva vida per a no ser eliminats. Quan hi hagi un canvi d'aquest valor en el servidor per qualsevol motiu, aquest enviarà a cada client la vida actualitzada. I finalment, cada client actualitzarà els valors que tinguin amb el del servidor.

## Eliminació i reaparició

Fins aquest moment, no passa res quan la vida d'un jugador arriba a zero, excepte per un missatge que ens ho comunica. Per poder acabar amb la part jugable del videojoc s'ha d'implementar el sistema de reapareixement, és a dir, que una vegada eliminat el jugador, aquest aparegui en una part del mapa específica amb la vida completa.

Implementar això és molt simple, una vegada la vida d'un jugador arriba a zero, el servidor envia un missatge informant a tots els clients de l'eliminació d'aquest jugador i el seu nou punt de reaparició. A continuació els clients simplement eliminaran a aquest jugador i el restauraran en la seva nova posició.

Per una altra part, el jugador eliminat rebrà un missatge per pantalla que l'informarà que ha sigut eliminat i serà preguntat si vol reaparèixer per continuar jugant o sortir del joc.

## Les proves que es realitzaran

Per poder extreure els resultats que em permetrà concloure quin protocol és millor per a la creació de videojocs en línia, s'han de fer unes proves amb el videojoc desenvolupat.

### Servidor

Primer de tot, per a permetre a qualsevol persona jugar al videojoc, aquest s'ha d'allotjar en el núvol. Per aconseguir això, he utilitzat la plataforma de Google Cloud per llogar un servidor situat a Madrid. La localització d'aquest és important, ja que com més lluny estigui, més temps tardarà la informació a arribar a aquest, independentment del protocol utilitzat, fent que l'experiència sigui dolenta. Per aquest motiu una localització a prop dels usuaris és recomanable.

### Clients

Cada client té una amplada de banda i velocitat d'internet diferent, el que farà que puguem tenir dades del que passaria amb un públic amb equips i connexions variades, que és el que passa en la realitat.

En cada client s'analitzarà quan disparin una bala amb l'arma de paintball quin retard hi ha entre que polsen el botó fins que veuen el patró, per tal d'observar l'experiència dels clients amb més fidelitat.

### Wireshark

Per poder obtenir informació addicional per examinar els protocols utilitzaré el programa Wireshark. Aquest programa permet analitzar tots els protocols que arriben i surten d'un ordinador, així com mostrar la informació que porta cada protocol, el retard, etc. Utilitzaré aquest programa al servidor, ja que és qui rep i envia tots els paquets, fent que sigui perfecte en aquesta ocasió.

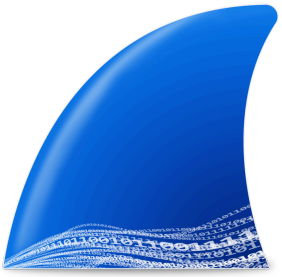


*Figura* 18. Logo del programa Wireshark.

### Proves que es faran

Per extreure els resultats necessaris es faran 3 proves:

1. En la primera prova els usuaris només executaran el videojoc, fent que aquest pugui utilitzar lliurement tota l'amplada de banda disponible de l'usuari.
2. En la segona prova els usuaris hauran de tenir un vídeo de la plataforma de YouTube executant-se en segon pla per limitar l'amplada de banda del videojoc.
3. En últim lloc, en la tercera prova en lloc de tenir un vídeo en segon pla els usuaris hauran de tenir un arxiu descarregant-se en tot moment, no importa l'arxiu que es descarrega, només importa el fet que això reduirà encara més l'amplada de banda que tindrà disponible el videojoc.

Aquestes 3 proves es faran tant amb TCP com amb UDP, i ajudarà a veure l'eficàcia de cada un en diferents situacions. Addicionalment, en cada prova s'utilitzarà Wireshark per analitzar tots els paquets enviats i rebuts del servidor. Finalment, també s'analitzarà el retard que experimenta cada client en veure l'acció de disparar executant-se en la seva pantalla.

## Resultats obtinguts

Com s'ha explicat en l'apartat anterior, els clients tenen una amplada de banda diferent, més específicament, amb les següents característiques:

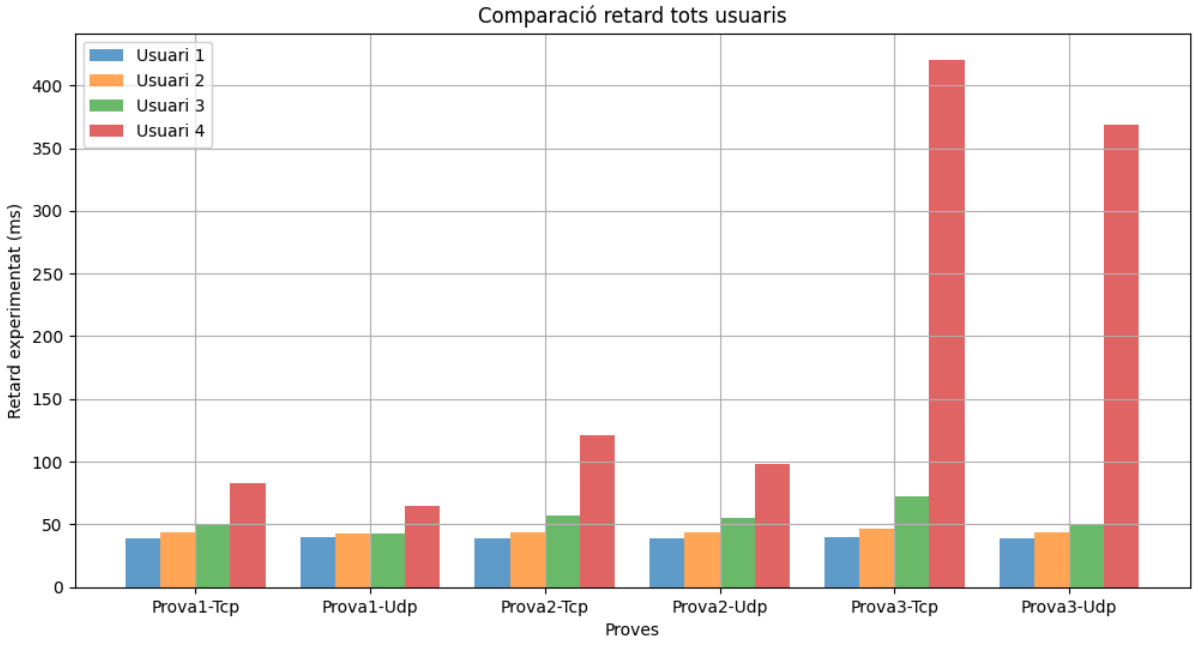
- Usuari 1 → **Millor** amplada de banda
- Usuari 2 → Amplada de banda **molt bona**
- Usuari 3 → Amplada de banda **mitjana alt**
- Usuari 4 → Amplada de banda **mitjana baix**

Després de dur a terme les proves explicades en l'apartat anterior aquests són els resultats captats. Dividits en l'objecte d'estudi: velocitat, fiabilitat i càrrega de dades o amplada de banda.

Tots els recursos addicionals utilitzats per a fer l'anàlisi que, no apareixen en aquest apartat, estan col·locats i detallats als annexos.

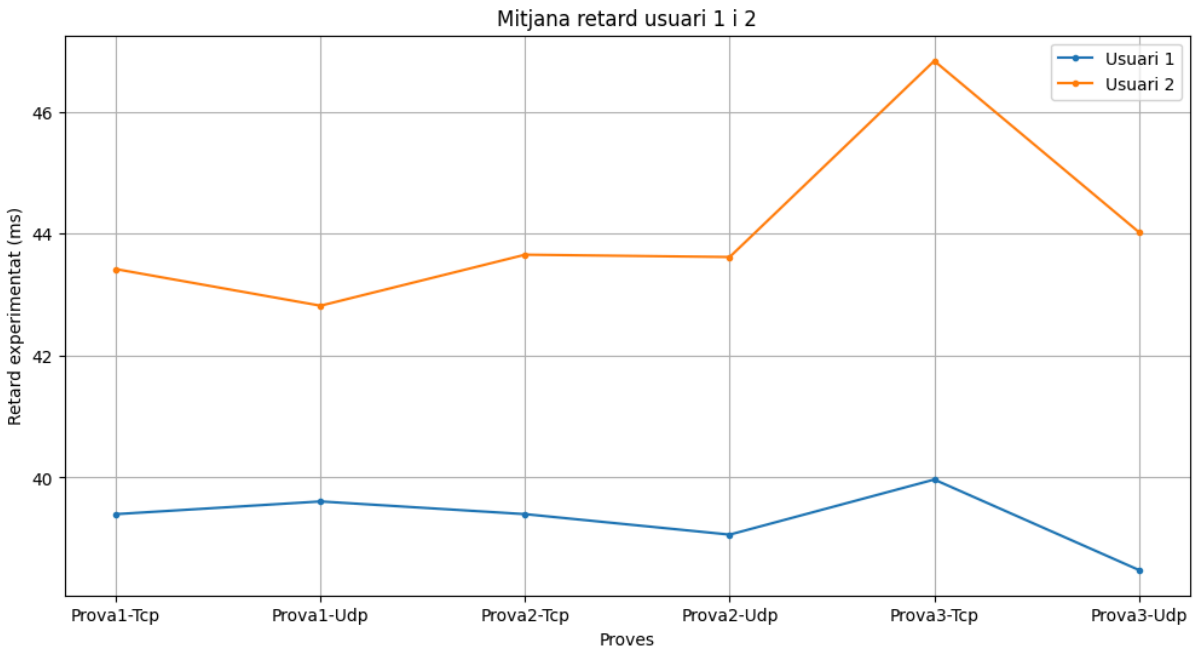
### Velocitat de la connexió

La velocitat és una de les parts més importants a l'hora de decidir quin protocol és millor en l'àmbit del videojoc. Això és perquè que si n'hi ha massa retard en les transmissions, l'experiència d'usuari se sentirà frustrant i pobre en general.



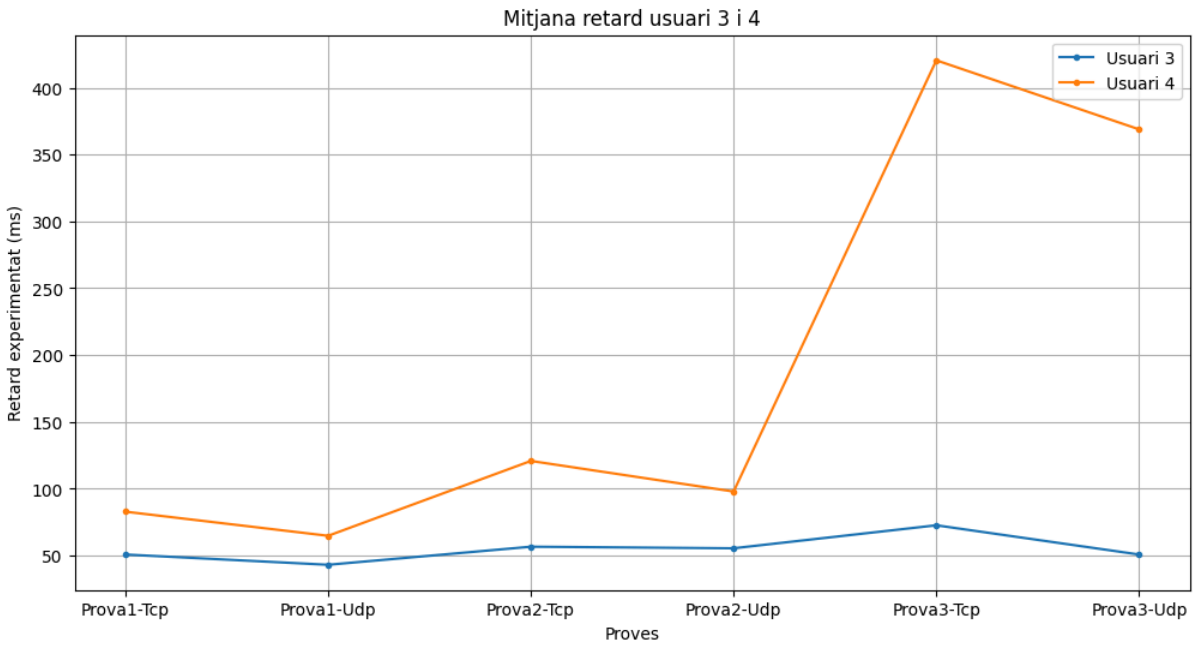

Gràfic que representa el retard experimentat per cada usuari en les diferents proves. Elaborat per David Didenco.

En el gràfic anterior es pot observar el retard de cada usuari depenent de la prova. Es pot apreciar com els usuaris un i dos, que tenen una molt bona amplada de banda) quasi no tenen un canvi notable en la latència, independentment del protocol utilitzat.




Mitjana del retard dels usuaris un i dos. Elaborat per David Didenco.

Aquest comportament és visible d'una manera més clara en aquest gràfic. L'usuari un té un retard que va des dels 38 ms fins als 40 ms com a màxim, és a dir, que la diferència és tan petita que no es pot notar. Un comportament quasi igual passa amb el segon usuari, però aquesta vegada va des dels 43 ms fins als 47 ms de retard. Tot i això, UDP té retard menor retard, per molt que aquesta diferència sigui imperceptible per a l'usuari a causa de la seva petitesa.




Mitjana del retard dels usuaris 3 i 4. Elaborat per David Didenco.

En canvi, amb els usuaris tres i quatre no passa el mateix. Per una banda, l'usuari tres sí que té un canvi notable en la seva latència al llarg de les proves. Aquesta va des dels 44 ms fins als 73 ms aproximadament. Encara que aquest continuï sent un canvi bastant petit i difícil de notar en certes situacions, sí que es notaria a l'hora de jugar un joc competitiu, on el temps de reacció és crucial, ja que tindries uns 23 ms menys per reaccionar. Això ocasionaria que l'experiència d'usuari se senti frustrant i injusta. En aquest cas, sí que és notable la diferència entre utilitzar TCP i UDP, sent UDP l'opció més ràpida.

Per una altra banda, l'usuari quatre, que té la pitjor amplada de banda, sí que nota un gran canvi depenent del protocol utilitzat. Mentre que les dues primeres proves segueixen el mateix comportament que als altres usuaris, en la tercera prova és on la latència es dispara, i on, conseqüentment, la diferència entre els dos protocols també ho fa. En aquesta tercera prova el retard va des dels 420 ms amb TCP fins als 370 ms amb UDP.

Per concloure l'àmbit de la velocitat, UDP ha obtingut en totes les proves un retard menor a TCP.

## Fiabilitat

No només la velocitat de transmissió és important per a un videojoc en línia, la fiabilitat també ho és. De res serveix que la informació s'enviï ràpidament si aquesta no arriba al seu destinatari.

| | Packet Loss Usuari 1 | Packet Loss Usuari 2 | Packet Loss Usuari 3 | Packet Loss Usuari 4 |
|---|---|---|---|---|
| Test 1 UDP | 2,96% | 3,90% | 3,40% | 6,68% |
| Test 3 UDP | 3,65% | 6,98% | 10,14% | 19,91% |

Taula amb la pèrdua de paquets de cada usuari a les proves 1 i 3 d'UDP. Elaborat per David Didenco.

Si analitzem la pèrdua de paquets[13] en la primera i tercera prova, que són les més significatives, es pot observar una pèrdua de paquets al voltant del 4,42% de mitjana, una xifra baixa. No obstant això, si analitzem la pèrdua en la prova 3, els valors són més variats entre els usuaris. Els que tenen una amplada de banda excel·lent (Usuari un i dos) experimenten una pujada de la pèrdua molt baixa, mentre que els altres dos usuaris mostren una pujada considerable, arribant fins al 19,91% en el cas de l'Usuari quatre.

Amb TCP aquest problema no existeix, perquè aquest protocol garanteix que cada transmissió arribi al seu destinatari, encara que sigui més lent. A més, com s'ha comentat amb anterioritat, TCP també garanteix la correcció d'errors que poden tenir els paquets que arriben al destinatari, el que fa que no s'hagi d'enviar una altra vegada, optimitzant la utilització de recursos del sistema. UDP, per una altra banda, no té aquestes prestacions, el que ocasiona que hi hagi una pèrdua de paquets en tot moment, encara que aquesta pugui arribar a ser molt baixa.

[13] ***Pèrdua de paquets:*** *Percentatge de paquets que no arriben al seu destinatari en una transmissió per diversos factors externs, calculat com (paquets perduts / paquets enviats * 100).*

### Càrrega de Dades i Amplada de Banda

Un altre apartat crucial és la quantitat d'informació que s'envia a través de la xarxa per fer que un joc en línia funcioni. Encara que la quantitat exacta varia depenent del videojoc i de com està fet, aquesta sempre tendeix a ser minúscula.

En les següents aules estan tots els *bytes* transmesos entre client i servidor, així com la mida mitjana dels paquets, tant per a TCP com UDP.

| Informació enviada utilitzant TCP (escenari Prova 1, duració de 5 minuts aproximadament | | | | | | | |
|---|---|---|---|---|---|---|---|
| | Bytes totals | Bytes A → B | Bytes B → A | Paquets A → B | Paquets B → A | Paquets | Bytes/Paquet |
| Usuari 1 | 3519658 | 1209671 | 2309987 | 18659 | 24625 | 43284 | 81,315 |
| Usuari 2 | 3919459 | 1456365 | 2463094 | 22620 | 26494 | 49114 | 79,803 |
| Usuari 3 | 3908865 | 1446014 | 2462851 | 22488 | 26535 | 49023 | 79,735 |
| Usuari 4 | 3822438 | 1392818 | 2429620 | 21490 | 25851 | 47341 | 80,743 |

Taula resumida de les interaccions del servidor i els usuaris a la prova 1 utilitzant TCP. B: representa el servidor i A: l'usuari de cada fila. Elaborat per David Didenco.

| Informació enviada utilitzant UDP (escenari Prova 1, duració de 7 minuts aproximadament | | | | | | | |
|---|---|---|---|---|---|---|---|
| | Bytes totals | Bytes A → B | Bytes B → A | Paquets A → B | Paquets B → A | Paquets | Bytes/Paquet |
| Usuari 1 | 4514567 | 957408 | 3557159 | 17407 | 56755 | 74162 | 60,874 |
| Usuari 2 | 4237486 | 818138 | 3419348 | 14731 | 54348 | 69079 | 61,343 |
| Usuari 3 | 4289546 | 848821 | 3440725 | 15332 | 54872 | 70204 | 61,101 |
| Usuari 4 | 4177808 | 776460 | 3401348 | 11177 | 52388 | 63565 | 65,725 |

Taula resumida de les interaccions del servidor i els usuaris a la prova 1 utilitzant UDP. B: representa el servidor i A: l'usuari de cada fila. Elaborat per David Didenco.

Encara que sembli que s'hagi enviat molta informació, en realitat és poca, ja que si observem els Bytes per paquets de cada un, podem veure com TCP envia de mitjana 80,399 Bytes per paquet, en canvi, UDP envia 62,261 Bytes per cada un. TCP ha de trametre més informació per paquet per poder oferir l'enviament exitós d'aquest, correcció d'errors, etc.

És cert que, en la prova d'UDP, s'ha remès més informació en total, però això és perquè aquesta ha durat més temps, així i tot, la diferència de quantitat total enviada respecte de TCP no és tan alta, com si hauria sigut si la situació hagués passat a l'inrevés.

Si convertim els Bytes totals enviats de cada usuari a *MegaBytes (MB)*, una unitat més comuna i coneguda, veiem que s'ha enviat una mitja de 3,79 MB en total amb TCP i 4,30 MB en total amb UDP, el que equival aproximadament a una imatge d'alta qualitat en format JPEG, un format estàndard d'imatges.

# Conclusions

## Velocitat de la connexió

Com s'ha pogut observar en els resultats obtinguts, UDP ha mantingut sempre una latència inferior a la de TCP. Tot i que la diferència de latència entre TCP i UDP no és perceptible per als dos primers usuaris, que disposen d'una connexió a internet de millor qualitat, sí que ho és per als dos últims, els quals, amb una amplada de banda inferior, experimenten una diferència màxima de 23 ms i 50 ms, respectivament. A més, considerant que no tothom té una connexió tan bona com la dels dos primers usuaris, o que sovint comparteixen l'ús de l'internet, cosa que limita l'amplada de banda, queda clar que en termes de velocitat, UDP és, indiscutiblement, la millor opció.

## Fiabilitat

Com s'ha observat en els resultats, UDP experimenta una pèrdua de paquets a causa de la manca d'un mecanisme que en garanteixi l'entrega exitosa, a diferència de TCP. Aquesta pèrdua oscil·la entre el 3% i el 6% en la primera prova, i entre el 7% i el 20% en la tercera prova, fet que demostra que com menor és l'amplada de banda disponible, major és la pèrdua de paquets, tal com s'ha evidenciat amb més claredat en el cas de l'usuari quatre.

Tanmateix, això no implica que UDP no es pugui utilitzar per enviar paquets amb la garantia que arribaran al seu destinatari, però aquest procés ha de ser implementat manualment pel desenvolupador o l'empresa. Per exemple, la llibreria que he utilitzat per desenvolupar la part en línia del meu videojoc, *Riptide*, disposa d'una funció que assegura l'enviament exitós dels paquets UDP necessaris. No obstant això, aquesta funció fa que l'enviament de la informació sigui una mica més lent en comparació amb enviar-la directament sense cap mena de garantia, tot i que no arriba a ser tan lent com utilitzar TCP.

També és important aclarir que en l'àmbit dels videojocs en línia no és necessari que tots els paquets arribin de manera exitosa. Per exemple, si el servidor envia 40 actualitzacions de les posicions dels jugadors cada segon, com en el meu videojoc, no hi ha problema si algunes d'aquestes posicions no arriben als clients. És preferible que aquestes arribin al més ràpidament possible per oferir una bona experiència d'usuari, encara que algunes es perdin pel camí, en lloc d'assegurar que totes les posicions arribin a tots els clients, però amb un retard significatiu, fet que faria que l'experiència del videojoc es percebés lenta i frustrant.

Tot i així, hi ha informació que és imprescindible que arribi al receptor per al correcte funcionament del joc. Per exemple, quan es dispara, és essencial que l'acció es transmeti sense errors, ja que aquesta acció es pot realitzar una vegada cada certs segons, depenent de l'usuari, a diferència de les 40 actualitzacions per segon de les posicions. Si aquesta acció no es transmetés, l'experiència seria frustrant i injusta, ja que el jugador no hauria disparat quan havia polsat el botó a causa de la pèrdua del paquet, fet que escapa al seu control.

Si ens centrem exclusivament en la fiabilitat, tant UDP com TCP són opcions vàlides, perquè tot i que UDP no garanteix l'enviament exitós de paquets de manera predeterminada, aquesta funcionalitat es pot afegir manualment. Per tant, pel que fa a la fiabilitat, no importa quin protocol s'utilitzi, ja que ambdós poden arribar a garantir l'enviament correcte dels paquets.

## Càrrega de Dades i Amplada de Banda

Com s'ha observat, als videojocs en línia cal remetre molt poca informació. En el cas del meu videojoc, és necessari enviar uns 4,30 MB per jugar 7 minuts amb UDP, que, si ho convertim a MB/s, equival a 0,0102 MB/s, una quantitat realment insignificant. Amb TCP cal enviar més informació en comparació amb UDP, tot i que la diferència és tan petita que no és perceptible.

El fet que calgui enviar tan poca informació fa que UDP sigui més eficient, ja que no necessita gestionar grans volums de dades. Tal com s'ha vist en apartats anteriors, TCP és més adequat per enviar grans quantitats d'informació, però en aquest no és el cas.

## Conclusió General

UDP destaca com la millor opció de protocol de comunicació per a videojocs en línia gràcies a la seva menor latència i eficiència en l'enviament de dades. Aquest avantatge és especialment rellevant per als usuaris amb connexions a internet més lentes, on la diferència de latència entre TCP i UDP pot ser significativa. Tot i que TCP ofereix una major fiabilitat en l'entrega de paquets i correcció d'errors de manera predeterminada, aquestes funcionalitats es poden afegir manualment a UDP, i a més, no són necessàries per a la majoria dels paquets transmesos.

Per als usuaris amb una amplada de banda excel·lent, la diferència entre TCP i UDP és mínima, tot i que UDP ofereix una lleugera millora en termes de latència. En canvi, per als usuaris amb connexions més lentes, UDP és clarament superior gràcies a la seva capacitat per reduir la latència i millorar la resposta del joc. A més, UDP és més eficient pel que fa a l'ús de l'amplada de banda, fet que resulta avantatjós en entorns amb una capacitat de xarxa limitada.

En conclusió, els resultats obtinguts demostren que UDP és la millor opció per a videojocs en línia en la gran majoria de situacions analitzades, oferint una millor experiència de joc gràcies a la seva menor latència i major eficiència en l'enviament de dades. Per tant, es pot concloure que la hipòtesi plantejada a l'inici del treball és correcta.

# Assessorament per part de ETSETB

Finalment, vull expressar el meu profund agraïment a l'Escola Tècnica Superior d'Enginyeria de Telecomunicacions de Barcelona (ETSETB) per haver-me brindat l'oportunitat d'obtenir assessorament per al meu treball de recerca. En particular, vull agrair sincerament a un professor de telemàtica que, amb gran generositat, m'ha ofert la seva ajuda.

Gràcies al seu coneixement, experiència i expertesa, el meu treball ha estat assessorat i revisat per un professional del sector, garantint així la seva qualitat, la precisió de la informació presentada i la correcta anàlisi de les dades obtingudes.

# Glossari

*Paquet (definició): En telecomunicacions, és un missatge gran dividit en segments més petits per optimitzar les comunicacions. L'emissor envia els paquets per separat i el receptor els combina per formar el missatge original.*

*Adreça IP (definició): Número d'identificació únic d'un dispositiu connectat a una xarxa, sigui privada (com la xarxa de casa teva) o pública (internet). Funcionament similar als números de telèfons, on cada dispositiu té la seva adreça.*

*Datagrama (definició): Mínim bloc d'informació en una xarxa de commutació per datagrames, el qual és un mètode per enviar els paquets al destinatari, aquest és utilitzat per protocols com TCP, UDP, entre altres.*

*Bit (definició): Dígit binari que pot tenir el valor de 0 o 1. El bit és la unitat mínima d'informació en informàtica, de la qual deriven altres unitats com el byte (8 bits), el megabyte (10^6 bytes), etc.*

*DDoS (denegació de serveis distribuïts) (definició): Atac maliciós amb l'objectiu d'interrompre el funcionament d'un servei o xarxa mitjançat una sobrecàrrega de connexions de dispositius infectats. Seria l'equivalent a embussar de manera intencionada una autopista amb cotxes que no tenen cap funció per tal de perjudicar els conductors legítims.*

*Port (definició): Punt virtual d'un ordinador per gestionar les connexions, encara que aquestes connexions arribin per la mateixa connexió a internet. N'hi ha molts ports per a una multitud de connexions distintes. Només els protocols del nivell 4, el nivell de transport, com TCP o UDP, poden especificar a quin port volen enviar un paquet.*

*Pèrdua de paquets (definició): Percentatge de paquets que no arriben al seu destinatari en una transmissió per diversos factors externs, calculat com (paquets perduts / paquets enviats * 100).*

*Motor gràfic o de videojocs (definició): Programa que proporciona eines per a la creació de videojocs, incloent-hi sistemes de física, dibuix en pantalla dels diferents elements del joc, etc.*

*Llibreria (definició): Conjunt d'eines ja creades que permeten realitzar accions sense haver de crear-les des de zero. Com per exemple, les funcions trigonomètriques en una llibreria matemàtica.*

*Llenguatge de programació (definició): Eina que proporciona a l'informàtic l'habilitat de donar instruccions a una màquina de forma ordenada i entenedora per als éssers humans. Exemples: C++, C#, C, Python, Ruby, JavaScript, etc.*

*Script (definició): Sèrie d'instruccions executables per una màquina o per un llenguatge de programació, normalment guardades com arxius separats. D'una manera més simplista, els*

*scripts són com fulls de paper que contenen instruccions escrites en un llenguatge en programació.*

*Amplada de banda (definició): Quantitat màxima de dades que es pot transmetre a través d'una connexió de xarxa en un temps determinat, normalment mesurada en bits/s, kilobytes/s, megabytes/s o gigabytes/s.*

*Tickrate (definició): Nombre de vegades per segon que un servidor envia als clients informació de qualsevol mena.*

# Webgrafia


AgendaPro Salud (13/09/2022) Tipos de protocolos de comunicación, qué son y para qué sirven [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://blog.agendapro.com/centros-de-salud/tipos-de-protocolos-de-comunicacion>

KIO (08/2019) ¿Qué son y para qué sirven los protocolos de comunicación de redes? [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.kio.tech/blog/data-center/protocolos-de-comunicación-de-redes>

Comunicare (03/10/2023) QUÉ ES UN PROTOCOLO DE COMUNICACIÓN [En linia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.comunicare.es/que-es-un-protocolo-de-comunicacion/>

Codename Noreste (22/02/2024) Protocolo de comunicaciones [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://es.wikipedia.org/wiki/Protocolo_de_comunicaciones>

Cloudflare (2023) ¿Qué es un paquete? | Definición de paquete de red [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/network-layer/what-is-a-packet/>

Fortinet (28/03/2023) ¿Qué es un modelo TCP/IP de protocolo de control de transmisión? [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.fortinet.com/lat/resources/cyberglossary/tcp-ip#:~:text=TCP%20significa%20Protocolo%20de%20control,en%20comunicaciones%20de%20red%20digitales.>

Diego V. (30/8/2023) Protocolo TCP: definición y funcionamiento [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.hostinger.es/tutoriales/protocolo-tcp>

Cloudflare (2024) ¿Qué es el UDP? [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/ddos/glossary/user-datagram-protocol-udp/>

IBM Corporation (12/04/2021) Protocolos TCP/IP [En línia] [Consultat: 19/03/2024] Disponible a Internet: <https://www.ibm.com/docs/es/aix/7.2?topic=protocol-tcpip-protocols>

Wikipedia Project (04/01/2022) Datagrama [En línia] [Consultat: 19/03/2024] Disponible a Internet: <https://es.wikipedia.org/wiki/Datagrama>

Erik Rodriguez (04/01/2022) TCP vs. UDP [En línia] [Consultat: 19/03/2024] Disponible a Internet: <https://www.skullbox.net/tcpudp.php>

IONOS (2024) UDP: ¿qué es el protocolo UDP? [En línia] [Consultat: 22/03/2024] Disponible a Internet: <https://www.ionos.es/digitalguide/servidores/know-how/udp-user-datagram-protocol/>

IBM Corporation (03/03/2021) User Datagram Protocol [En línia] [Consultat: 22/03/2024] Disponible a Internet: <https://www.ibm.com/docs/es/aix/7.1?topic=protocols-user-datagram-protocol>

Cloudflare (2024) ¿Qué es un ataque DDoS? [En línia] [Consultat: 22/03/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/ddos/what-is-a-ddos-attack/>

JavaTpoint (2021) What is Transmission Control Protocol (TCP)? [En línia] [Consultat: 24/03/2024] Disponible a Internet: <https://www.javatpoint.com/tcp>

Cloudflare (2024) ¿Qué es un puerto de ordenador? | Puertos en la red [En línia] [Consultat: 24/03/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/network-layer/what-is-a-computer-port/>

Wikipedia Project (2024) Modelo OSI [En línia] [Consultat: 25/03/2024] Disponible a Internet: <https://es.wikipedia.org/wiki/Modelo_OSI>

Cloudflare (2024) ¿Qué es el modelo OSI? [En línia] [Consultat: 03/04/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/ddos/glossary/open-systems-interconnection-model-osi/>

Peter (14/02/2022) UDP vs TCP for real-time streaming telemetry [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://blog.sflow.com/2022/02/udp-vs-tcp-for-real-time-streaming.html>

Wargaming (13/09/2022) ¿Qué son el ping, la latencia y la pérdida de paquetes? [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://eu.wargaming.net/support/es/products/wows/article/10253/#:~:text=La%20pérdida%20de%20paquetes%20se,inferior%20al%202%2C5%20%25.>

Pico (2024) UDP vs TCP [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://www.pico.net/kb/udp-vs-tcp/>

alfredtso (22/04/2021) Performance Analysis of TCP and UDP [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://github.com/alfredtso/ns-3-project>

Kate Vogel (28/07/2023) What is jitter? How to test and reduce internet jitter [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://www.ringcentral.com/us/en/blog/what-is-jitter/#:~:text=Jitter%20is%20when%20there%20is,congestion%2C%20and%20sometimes%20route%20changes.>

IBM Corporation (24/03/2023) TCP/IP security [En línia] [Consultat: 09/04/2024] Disponible a Internet: <https://www.ibm.com/docs/en/aix/7.2?topic=network-tcpip-security>

Anders (5/10/2018) Encryption of TCP communication? [En línia] [Consultat: 09/04/2024] Disponible a Internet: <https://security.stackexchange.com/questions/195111/encryption-of-tcp-communication>

IBM Corporation (11/4/2023) TCP/IP security [En línia] [Consultat: 09/04/2024] Disponible a Internet: <https://www.ibm.com/docs/en/zvm/7.2?topic=control-tcpip-security>

NXLog Docs (2024) Send logs to McAfee Enterprise Security Manager (ESM) [En línia] [Consultat: 09/04/2024] Disponible a Internet:

<https://docs.nxlog.co/integrate/mcafee-esm.html>

Arun Chand (06/04/2011) Is LDAP a TCP or a UDP protocol? [En línia] [Consultat: 09/04/2024] Disponible a Internet: <https://stackoverflow.com/questions/5913941/is-ldap-a-tcp-or-a-udp-protocol>

o3de (2024) UDP with Datagram Transport Layer Security (DTLS) [En línia] [Consultat: 09/04/2024] Disponible a Internet: <https://docs.o3de.org/docs/user-guide/networking/aznetworking/encryption/>

David Jacobs (20/10/2023) What to know about UDP vulnerabilities and security [En línia] [Consultat: 09/04/2024] Disponible a Internet: <https://www.techtarget.com/searchnetworking/answer/Are-there-any-inherent-security-problems-with-UDP>

zenarmor (2024) TCP vs UDP: A Simplified Comparison of Vital Network Protocols [En línia] [Consultat: 12/04/2024] Disponible a Internet: <https://www.zenarmor.com/docs/network-basics/tcp-vs-udp#what-is-the-difference-between-tcp-and-udp-in-terms-of-reliability>

Unity Technologies (2024) Go Create [En línia] [Consultat: 17/04/2024] Disponible a Internet: <https://unity.com>

Félix Palazuelos (2015) Qué son los motores gráficos y cuáles son los más populares [En línia] [Consultat: 17/04/2024] Disponible a Internet: <https://blogthinkbig.com/motores-graficos/>

Jorge Moreno Aguilera (03/10/2019) Multiplayer en UE4 [En línia] [Consultat: 20/04/2024] Disponible a Internet: <https://www.aprendeunrealengine.com/multiplayer-en-ue4/>

Wikipedia Project (13/04/2024) Library (computing) [En línia] [Consultat: 21/04/2024] Disponible a Internet: <https://en.wikipedia.org/wiki/Library_(computing)>

Tom Weiland (2022) About Riptide [En línia] [Consultat: 21/04/2024] Disponible a Internet: <https://riptide.tomweiland.net/manual/overview/about-riptide.html>

Wikipedia Project (11/04/2024) Programming language [En línia] [Consultat: 22/04/2024] Disponible a Internet: <https://en.wikipedia.org/wiki/Programming_language>

Carles Pairot Gavaldà, Rubén Mondéjar Andreu (2019) Juegos multijugador en red [En línia] [Consultat: 24/04/2024] Disponible a Internet: <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiGzYeCldFAxUOSfEDHUiRCtUQFnoECBcQAQ&url=https%3A%2F%2Fopenaccess.uoc.edu%2Fbitstream%2F10609%2F148676%2F3%2FJuegosMultijugadorEnRed.pdf&usg=AOvVaw2wW9fDRF5M15AxDiyEspXf&opi=89978449>

Brave Software (24/08/2023) What is a script? [En línia] [Consultat: 24/04/2024] Disponible a Internet: <https://brave.com/glossary/script/#:~:text=A%20script%20is%20a%20sequence,and%20executed%20in%20real%20time.>

Tom Weiland (04/01/2022) How to Make a Multiplayer Game in Unity | Connecting Clients to a Server - Part 1 [En línia] [Consultat: 26/04/2024] Disponible a Internet: <https://www.youtube.com/watch?v=6kWNZOFcFQw&t=531s>

Gabriel Gambetta (2024) Fast-Paced Multiplayer (Part III): Entity Interpolation [En línia] [Consultat: 30/04/2024] Disponible a Internet: <https://gabrielgambetta.com/entity-interpolation.html>

Flo Networks (2024) ¿Qué es el ancho de banda? [En línia] [Consultat: 30/04/2024] Disponible a Internet: <https://flo.net/es/que-es-el-ancho-de-banda/#:~:text=El%20ancho%20de%20banda%20se,pueden%20transmitir%20en%201%20segundo.>

Wireshark Foundation (2024) The world's most popular network protocol analyzer [En línia] [Consultat: 03/07/2024] Disponible a Internet: <https://www.wireshark.org/>

Wireshark Foundation (13/09/2022) Capture Lifecycle with Tshark [En línia] [Consultat: 03/07/2024] Disponible a Internet: <https://tshark.dev/>

## Figures:

Figura 1: Cloudflare, Inc (2024) Il·lustració del model OSI [En línia] [Consultat: 03/04/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/ddos/glossary/open-systems-interconnection-model-osi/>

Figura 2: JavaTpoint (2021) Il·lustració del funcionament del protocol TCP [En línia] [Consultat: 24/03/2024] Disponible a Internet: <https://www.javatpoint.com/tcp>

Figura 3: Cloudflare (2024) Il·lustració del funcionament del protocol UDP [En línia] [Consultat: 12/03/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/ddos/glossary/user-datagram-protocol-udp/>

Figura 4: Cloudflare (2024) Il·lustració metaforica sobre que és un atac DDOS [En línia] [Consultat: 22/03/2024] Disponible a Internet: <https://www.cloudflare.com/es-es/learning/ddos/what-is-a-ddos-attack/>

Figura 5: Peter (14/02/2022) Gràfic del retard dels missatges en arribar al destinatari (en segons) depenent de la pèrdua de paquets. [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://blog.sflow.com/2022/02/udp-vs-tcp-for-real-time-streaming.html>

Figura 6: Peter (14/02/2022) Quantitat de missatges rebuts cada segon segons la pèrdua de paquets [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://blog.sflow.com/2022/02/udp-vs-tcp-for-real-time-streaming.html>

Figura 7: alfredtso (22/04/2021) Temps que tarda el missatge a arribar al destinatari (en ms) segons la mida del missatge (en bytes). En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://github.com/alfredtso/ns-3-project>

Figura 8: alfredtso (22/04/2021) Retard que hi ha abans d'enviar el paquet (en segons) depenent de la mida d'aquests (en bytes). [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://github.com/alfredtso/ns-3-project>

Figura 9: alfredtso (22/04/2021) Pèrdua de paquets d'UDP segons la dimensió del missatge que s'envia. [En línia] [Consultat: 06/04/2024] Disponible a Internet: <https://github.com/alfredtso/ns-3-project>

Figura 10: Microsoft (2024) Il·lustració del motor de videojocs Unity [En línia] [Consultat: 17/04/2024] Disponible a Internet: <https://dotnet.microsoft.com/es-es/apps/games/unity>

Figura 11: ARCHERY TAG BARCELONA (2016) Fotografia d'una persona jugant al paintball [En línia] [Consultat: 17/04/2024] Disponible a Internet: <https://www.archerytagbarcelona.com/blog/como-funciona-una-pistola-de-paintball/>

Figura 12: ShadowWolf Games (16/12/2020) Paintball War [En línia] [Consultat: 17/04/2024] Disponible a Internet: <https://store.steampowered.com/app/757130/Paintball_War/>

Figura 13: Jorge Moreno Aguilera (03/10/2019) Esquema del funcionament d'un videojoc en línia. [En línia] [Consultat: 20/04/2024] Disponible a Internet: <https://www.aprendeunrealengine.com/multiplayer-en-ue4/>

Figura 14: Jorge Moreno Aguilera (03/10/2019) Esquema de l'autoritat que té el servidor sobre els clients. [En línia] [Consultat: 20/04/2024] Disponible a Internet: <https://www.aprendeunrealengine.com/multiplayer-en-ue4/>

Figura 15: Tom Weiland (2022) Logotip de la llibreria Riptide [En línia] [Consultat: 21/04/2024] Disponible a Internet: <https://riptide.tomweiland.net/manual/overview/about-riptide.html>

Figura 16: Gabriel Gambetta (2024) Esquema on es mostra la falta de fluïdesa al videojoc [En línia] [Consultat: 30/04/2024] Disponible a Internet: <https://gabrielgambetta.com/entity-interpolation.html>

Figura 17: Gabriel Gambetta (2024) Esquema del funcionament de la interpolació [En línia] [Consultat: 30/04/2024] Disponible a Internet: <https://gabrielgambetta.com/entity-interpolation.html>

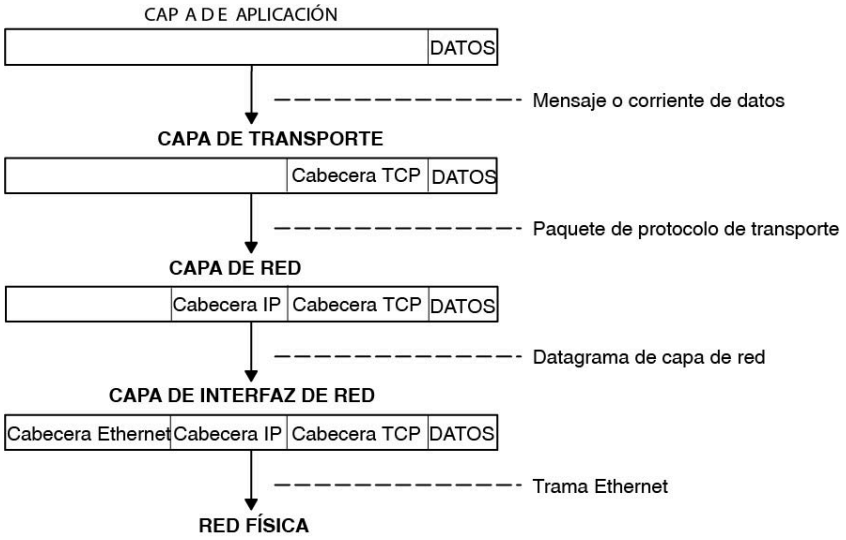
Figura 18: Wireshark Foundation (09/05/2023) Logo del programa Wireshark [En línia] [Consultat: 30/04/2024] Disponible a Internet: <https://commons.wikimedia.org/wiki/File:Wireshark_icon_new.png>

# Annexos

## Tcp

Per poder enviar la informació al destinatari, TCP divideix el missatge que es vol enviar en diferents paquets d'informació que s'enviaran per separat, cada paquet haurà de passar per quatre capes. Aquest procés que prepara cada paquet per la seva transmissió es diu *encapsulació,* aquest procés crearà el *datagrama*[14] que serà enviat al destinatari:

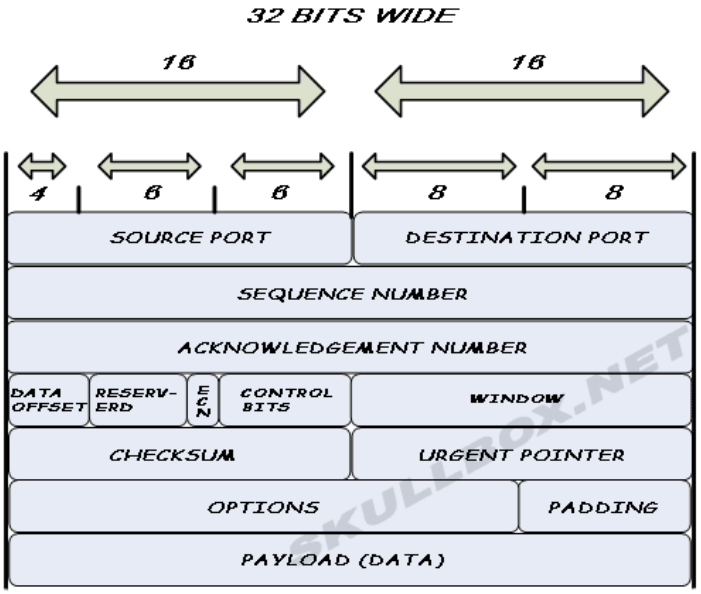
1. **Capa d'aplicació:** L'aplicació crea el missatge que es vol enviar al destinatari
2. **Capa de transport:** TCP divideix aquest missatge en diferents paquets i se'ls s'hi posa a cada un una direcció de destí
3. **Capa de xarxa:** Es posa el paquet en un datagrama d'IP (Internet Protocol), se li posa la capçalera i la cola del datagrama, per últim es decideix on s'ha d'enviar aquest datagrama.
4. **Capa d'interfície de xarxa:** Aquesta capa accepta els datagrames creats en la capa anterior i els envia al destinatari a través d'un equip de transmissió específic, com per exemple per un cable Ethernet. A partir d'aquí els paquets són transmesos per les xarxes físiques (cables de fibra òptica, cables submarins, etc.).




Les diferents capes per les quals ha de passar un protocol per poder ser enviat al destinatari. Font: https://www.ibm.com/docs/es/aix/7.2?topic=protocol-tcpip-protocols

[14] ***Datagrama:*** *Mínim bloc d'informació en una xarxa de commutació per datagrames, el qual és un mètode per enviar els paquets al destinatari, aquest és utilitzat per protocols com TCP, UDP, entre altres.*

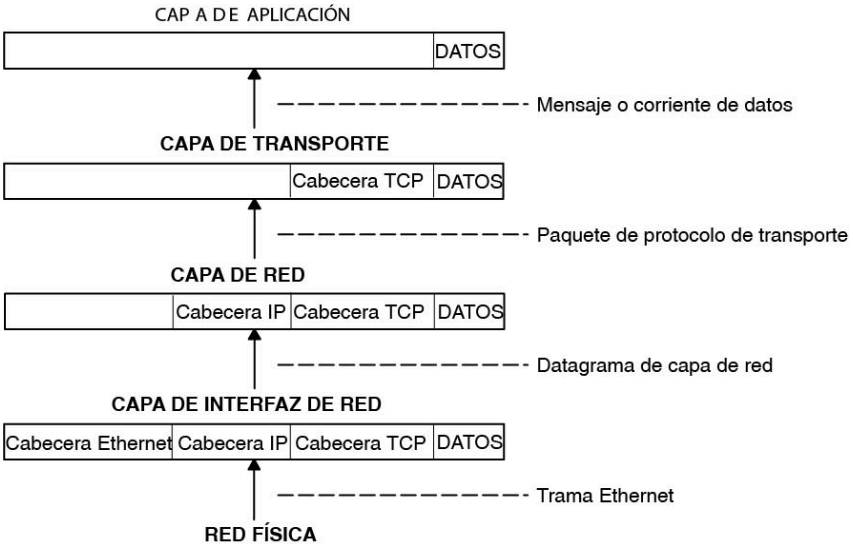
Després de passar per aquest procés d'encapsulació cada paquet d'informació que s'enviï al destinatari tindrà un aspecte semblant al següent:




Aspecte de cada paquet d'informació o datagrama amb la informació del receptor, el port, la informació… de TCP. Font: https://www.skullbox.net/tcpudp.php

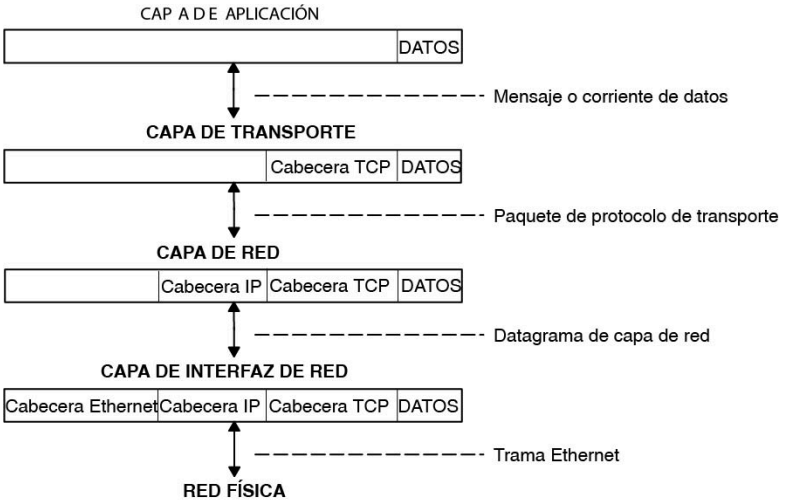
El datagrama té una longitud de 32 bits en total i com es pot apreciar l'estructura d'aquest és bastant complexa, això és deu més que res al sistema de reenviament de missatges de TCP. El *PAYLOAD* és realment el missatge que es vol enviar al destinatari, i la resta són paràmetres necessaris per a que el paquet arribi al seu destinatari i que en cas que no ho faci TCP ho pugui reenviar una altra vegada.

Quan el destinatari rep els paquets enviats per l'emissor, aquests han de travessar una altra vegada les quatre capes d'abans, però aquesta vegada en sentit contrari.




Les diferents capes per les quals ha de passar un protocol per poder ser rebut pel destinatari. Origen: https://www.ibm.com/docs/es/aix/7.2?topic=protocol-tcpip-protocols

Els sistemes principals d'una xarxa envien i reben constantment informació, fins i tot simultàniament. Per aquest motiu aquests processos es van fent en paral·lel per a no alentir la transmesa d'informació.




**Nota:** las cabeceras se añaden y separan en cada capa de protocolo a medida que los host transmiten y reciben datos.

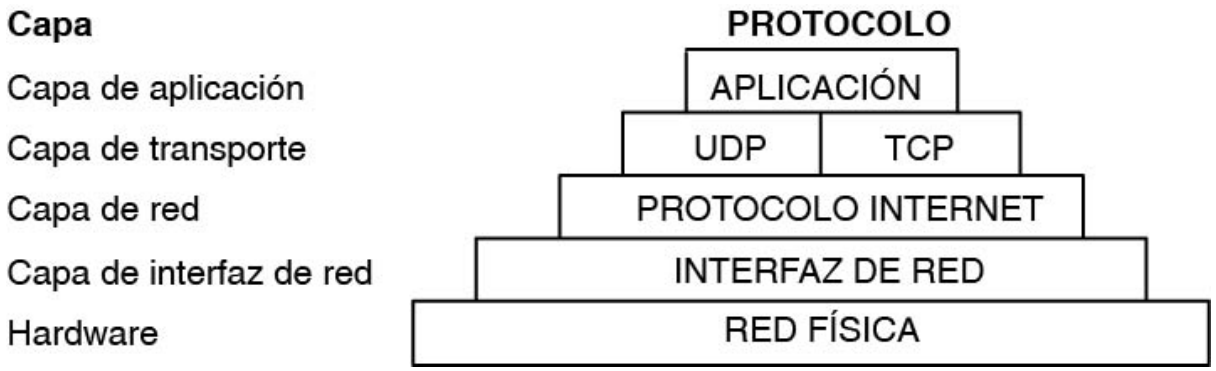
El treball en paral·lel que fan possible la transmissió d'informació de manera simultània en un dispositiu Origen: https://www.ibm.com/docs/es/aix/7.2?topic=protocol-tcpip-protocols

## Udp

Igual que TCP, UDP utilitza el sistema de datagrames com a forma d'organitzar la informació que es vol enviar al receptor.

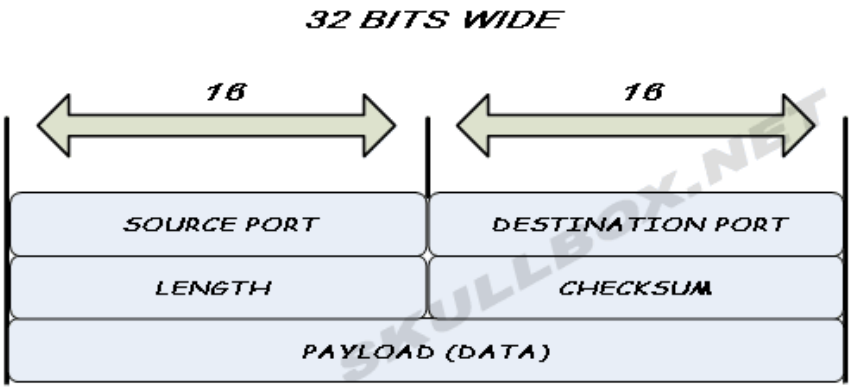
El procés d'encapsulament és el mateix que en el de TCP, però és més simple i ràpid al no haver-hi de posar informació addicional que ofereixen les funcionalitats de TCP. UDP també ofereix la capacitat de rebre i enviar paquets de manera instantània, com fa TCP. Aquest procés més simplificat seria:

1. **Capa d'aplicació:** L'aplicació crea el missatge que es vol enviar al destinatari
2. **Capa de transport:** En aquest cas, UDP, divideix aquest missatge en diferents paquets i se'ls s'hi posa a cada un una direcció de destí
3. **Capa de xarxa:** Es posa el paquet en un datagrama d'IP (Internet Protocol) y se li posa la direcció de destí.
4. **Capa d'interfície de xarxa:** Aquesta capa accepta els datagrames creats en la capa anterior i els envia al destinatari a través d'un equip de transmissió específic, com per exemple per un cable Ethernet. A partir d'aquí els paquets són transmesos per les xarxes físiques (cables de fibra òptica, cables submarins, cables d'Ethernet, etc.).




Les diferents capes per les quals ha de passar un paquet en els protocols TCP i UDP. Font: https://www.ibm.com/docs/es/aix/7.2?topic=protocol-tcpip-protocols

El que si canvia és l'estructura del datagrama després de l'encapsulament de la informació. Aquest datagrama tindria la següent forma:




Aspecte de cada paquet d'informació o datagrama amb la informació del receptor, el port, la informació… d'UDP. Font: https://www.skullbox.net/tcpudp.php

El datagrama té una longitud de 32 bits i conte la informació que s'envia en el *PAYLOAD*, exactament igual que en el TCP. Però el datagrama que genera UDP, a part d'aquests atributs, només conté la destinació del paquet, res més. Això és degut al fet que no és necessari en aquest cas, a diferència del TCP, que necessita posar més atributs per poder dur a terme altres tasques com l'enviament exitós dels paquets, entre d'altres.

## HTTP / HTTPS

### Introducció

Hypertext Transfer Protocol (HTTP) (o protocol de transferència d'Hipertext) és un protocol que pertany a la capa d'aplicació, i està basat en TCP. Això vol dir que agafa totes les funcionalitats que té TCP i les augmenta per poder centrar-se en un camp en específic. Encara que també es pot fer que en lloc de TCP utilitzi qualsevol altre protocol de la capa de transport, com pot ser UDP.

HTTP se centra en la transmissió de documents d'hipermèdia als navegadors, com (HTML), que és el que s'utilitza per mostrar les pàgines web. Però aquest no és el seu únic propòsit, ja que també es pot utilitzar per administrar les connexions d'un servidor web.

Per una altra banda, HTTPS és simplement la versió segura d'HTTP, com indica el seu nom: protocol de transferència d'Hipertext. L'única diferencia entre el HTTP i la versió segura és que aquesta encripta tots els missatges que s'envien, el que és especialment important a l'hora d'enviar dades confidencials.

### Funcionament

És important mencionar que HTTP i, conseqüentment HTTPS, és un protocol sense estat, això vol dir que el protocol analitza les dades que rep sense revisar les que hagi rebut prèviament, i cada nova sol·licitud que rep el servidor és com si fos una de nova. Això ocasiona que si s'envia més d'un paquet a la vegada, una petició no sap que l'altra existeix, ja que el servidor no emmagatzema cap dada. Aquesta característica fa que no sigui un protocol especialment interactiu amb l'usuari final.

Per poder saber a on s'ha d'enviar la informació, el protocol utilitza les *urls* o enllaços webs. Una *url* és simplement una cadena de text que indica cap a on ha d'anar la informació que es vol enviar.


L'estructura bàsica d'un URL o enllaç web. Font: https://www.azion.com/es/blog/que-es-http-y-como-funciona/

Com es pot apreciar en la imatge anterior, un enllaç es pot dividir en tres parts:

1. **El protocol (HTTP:// o HTTPS://):** Indica al navegador o servidor web quin dels dos protocols ha d'utilitzar per enviar i rebre la informació desitjada.
2. **El domini (en aquest cas "azion.com") i el subdomini (en aquest cas "blog.":** És el nom amb el qual es coneix el lloc web o servidor web. Aquí també s'inclou el *Top Level Domain* (TDL), que indica la categoria, aquestes poden ser: .com, .org, .net, .tv, etc.
3. **La ruta (en aquest cas "/edge"):** Dona informació sobre a quina pàgina en específic ha d'anar la sol·licitud.

Com s'ha mencionat abans, HTTPS encripta tota la informació enviada i rebuda. Aquest procediment es realitza mitjançant un procés d'encriptació que es diu *Transport Layer Security* (TLS), o com es coneixia abans, *Secure Sockets Layer* (SSL). Aquest mètode utilitza dues claus per poder assegurar la seguretat del missatge:

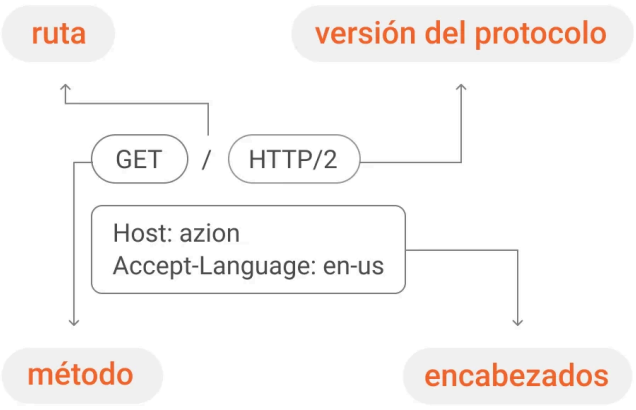
- **La clau privada**: És una clau que només el propietari coneix, i que serveix per decodificar el missatge.
- **La clau publica**: Aquesta està disponible per a qualsevol persona que vulgui interaccionar amb el servidor de manera segura, però la informació encriptada només pot ser desxifrada pel propietari d'aquest servidor, ja que per dur a terme aquesta acció necessites la clau privada.

L'ús d'HTTPS és recomanable a les pàgines web per sobre d'HTTP. De fet, molts navegadors actuals com CHROME (que és de GOOGLE), indiquen com a pàgines insegures les que no utilitzin aquest protocol, i les que sí, les marca com a segures amb un petit cadenat a dalt a l'esquerra de la pàgina.


Notificació de si una pàgina utilitza o no HTTPS a la majoria dels navegadors actuals. Font: https://dinahosting.com/blog/web-insegura-http/

En últim lloc, per poder enviar una petició amb aquests dos protocols, has d'especificar el mètode que vols utilitzar per obtenir la informació. Aquests mètodes estan preestablerts i són els següents:

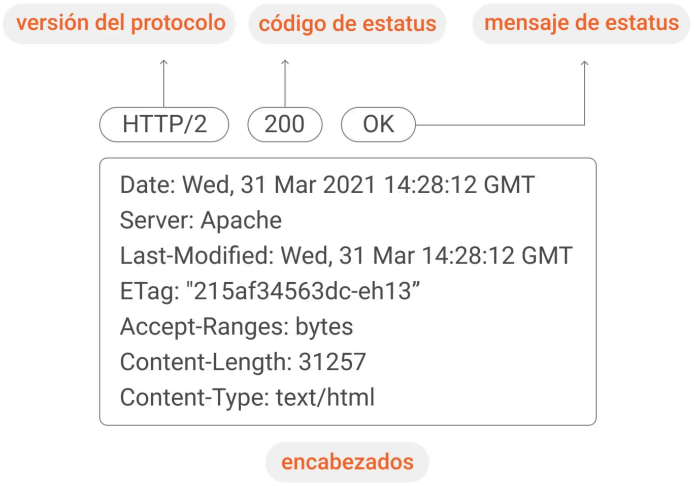
- **GET:** Per obtenir recursos o recuperar dades des del servidor; és el més comú.
- **POST:** Per enviar dades al servidor; per exemple, en registrar un formulari.
- **HEAD:** Per resumir la línia de resposta i les capçaleres.
- **PUT:** Per enviar fitxers al servidor o fer una actualització completa d'un recurs.
- **PATCH:** Per fer una actualització parcial d"un recurs.
- **DELET:** Per eliminar documents al servidor.
- **OPTIONS:** Per consultar quines ordres estan disponibles per a un usuari determinat.
- **TRACE:** Per depurar sol·licituds i tornar la capçalera d'un document.




Exemple de sol·licitud HTTP o HTTPS (HTTP en aquest cas). Font: https://www.azion.com/es/blog/que-es-http-y-como-funciona/

Amb això, el servidor t'enviarà el missatge resultant amb la teva informació i un codi d'estatus que t'informarà de com ha anat la teva petició. De codis n'hi ha un munt, ja que cada un defineix una situació en concret, però els més habituals són els següents:

- **200:** La sol·licitud ha sigut resposta de manera exitosa.
- **401:** La sol·licitud no ha sigut ser autoritzada pel servidor.
- **404:** La sol·licitud no ha pogut ser trobada pel servidor.
- **500:** Error intern del servidor.




Exemple d'una resposta d'un servidor web. Font: https://www.azion.com/es/blog/que-es-http-y-como-funciona/

## Avantatges i desavantatges

Un avantatge d'HTTP i HTTPS és que són protocols molt simples i fàcils a l'hora d'enviar i rebre informació. També, gràcies al fet que està basat en TCP, té totes les excepcionals funcionalitats d'aquest, el que el fa un protocol molt fiable. HTTPS soluciona l'únic problema d'HTTP, la falta de seguretat, encriptant els missatges que s'envien i es reben. A més, el fet de ser sense estat fa que les comunicacions siguin més ràpides, ja que el servidor no ha d'emmagatzemar dades sobre les sol·licituds.

El desavantatge més gran d'aquests dos protocols és que són bastant lents i tenen un ús molt limitat, que seria en les pàgines web i servidors web. Això ocasiona que siguin uns protocols molt puntuals i que estiguin restringits a una àrea en concret de la informàtica.

## Aplicacions

Com he comentat abans, les principals aplicacions per aquests dos protocols és a l'àmbit de les pàgines i servidors web. On destaquen per la seva gran simplicitat i facilitat d'ús, i són sense dubte la millor opció en aquests casos.

Però, no és l'única opció. Com he mencionat abans, HTTP/HTTPS també es pot utilitzar per transmetre informació a servidors web, fent que en una aplicació, quan passi determinada acció, li puguis enviar al teu servidor una petició per actualitzar un valor en concret, i aquest actualitzarà aquest valor per a tots els clients. Un exemple real d'això és el funcionament dels comptes en qualsevol servei. Per exemple, quan tu inicies en la teva compta de GMAIL, aquest envia mitjançant una connexió HTTPS als seus servidors que tu has iniciat sessió, fent que tu puguis veure aquesta acció que tu has fet des de qualsevol dispositiu (telèfon, altre ordinador, etc.), per aquest motiu de vegades se t'envia una notificació al teu telèfon mòbil preguntant si has sigut tu qui ha iniciat sessió.

## Conclusions

En conclusió, HTTP i HTTPS són protocols molt simples i d'una utilitat restringida, però això no fa que no sigui important per al món actual. En l'actualitat, HTTP i HTTPS són dels protocols més utilitzats, principalment per a les pàgines web i per a projectes on no cal enviar informació molt ràpidament i que aquests estiguin relacionades amb serveis webs.

# FTP

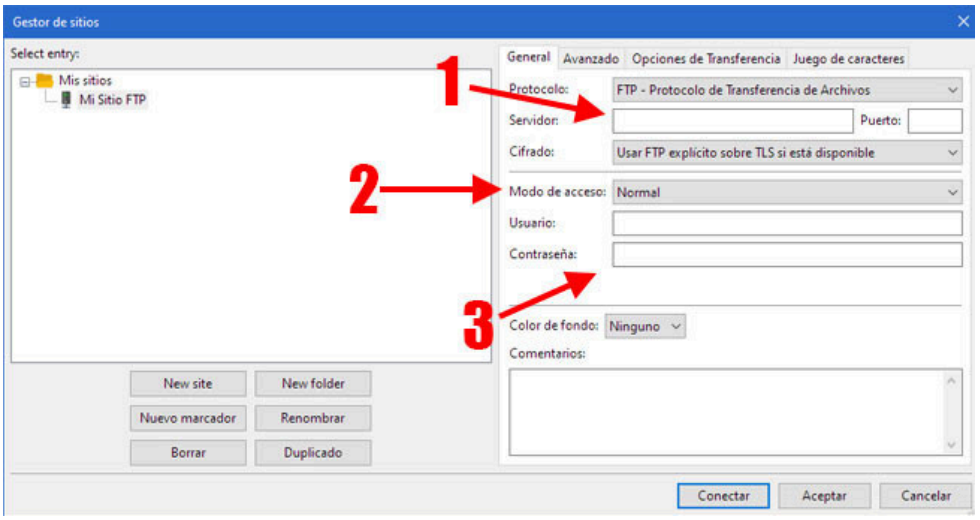
## Introducció

*File Transfer Protocol (FTP)* és un protocol que pertany al nivell 7, el nivell d'aplicació, que s'especialitza en la transmissió d'arxius entre diferents sistemes informàtics. Està basat en TCP, com molts altres, això fa que sigui un protocol fiable i efectiu. I encara que avui dia a poc a poc van sorgint alternatives que a poc a poc el substituiran en un futur pròxim, com per exemple els sistemes *Peer To Peer (P2P)*, aquest segueix estant molt utilitzat.

## Funcionament

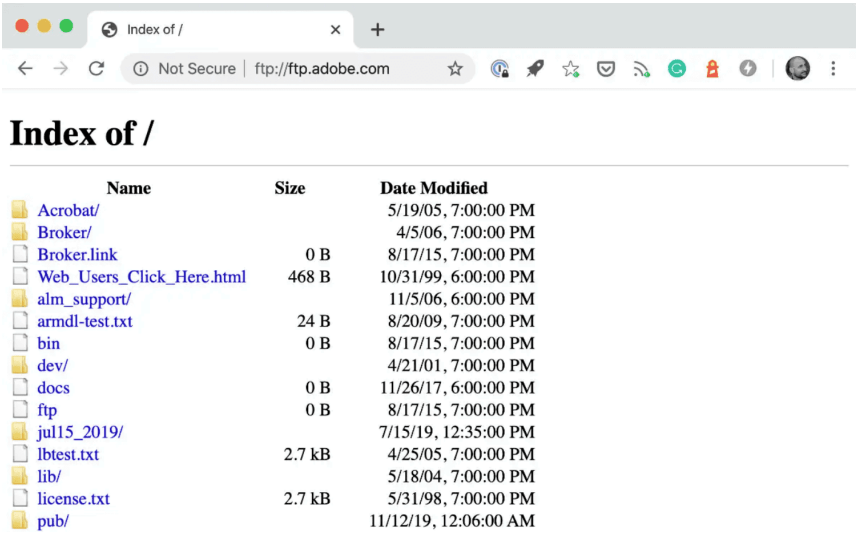
FTP envia els arxius de forma directa al destinatari i sense intermediaris, això permet que no hi hagi un màxim a la grandària dels arxius que es volen transmetre i a més fa que el procés sigui molt simple.

Abans de poder enviar o rebre cap mena d'informació, el receptor s'ha de connectar mitjançant un usuari i una contrasenya al servidor.




Programa extern que et permet connectar-te a un servidor FTP. Font: https://www.xataka.com/basics/ftp-que-como-funciona

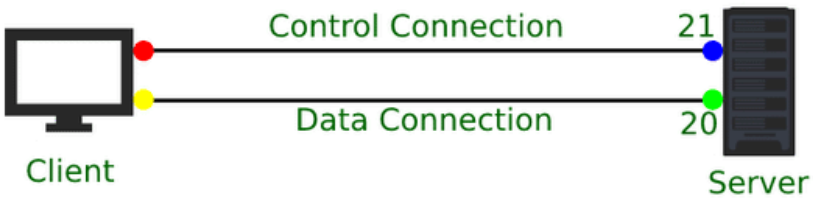
Si l'autenticació és correcta, el servidor li dona permís a l'usuari per mirar els arxius disponibles, esborrar, descarregar i pujar els seus propis arxius a aquest servidor. El procés és molt similar a com si estiguessis utilitzant Google Drive o qualsevol altre servei d'emmagatzemament en el núvol.




Accedeixes a FTP mitjançant un navegador web. Font: https://kinsta.com/es/base-de-conocimiento/que-es-el-ftp/

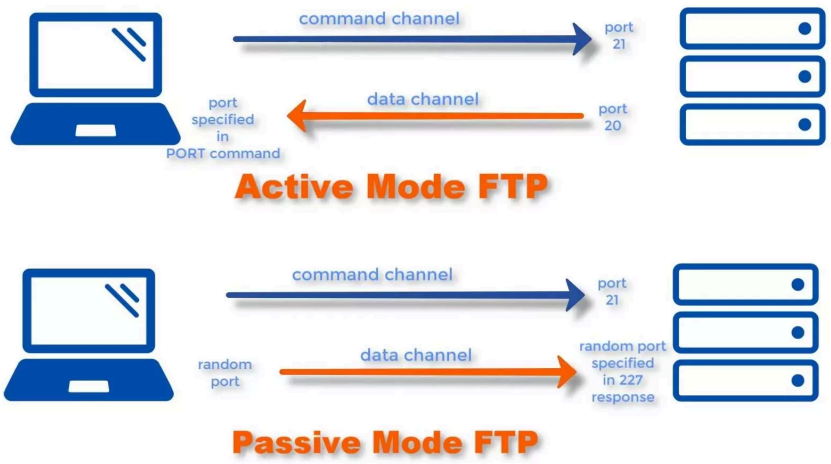
FTP per si sol no és un protocol segur, ja que tota la informació que s'envia i es rep està sense xifrar, el que fa que un atacant només amb escoltar les comunicacions pugui accedir a tota la informació sense cap mena d'esforç i apropiar-se dels arxius que es troben al servidor. Per solucionar aquest defecte s'han creat noves alternatives com per exemple el protocol FTPS, que xifra totes les dades que es transmeten per la xarxa per evitar que informació personal pugui acabar en mans dels atacants.

Per poder transmetre tota la informació necessària per al seu funcionament, FTP utilitza dos ports per aquesta tasca. El port 21 s'utilitza per transmetre els comandaments que ha de seguir el servidor (eliminar, pujar, descarregar, etc.) i el port 20 és pel que passen les dades dels arxius que es volen transmetre. Aquests ports es poden canviar de manera manual, però no és recomanable perquè ja estan estandarditzats.




Ports utilitzats per FTP. Font: https://kinsta.com/es/base-de-conocimiento/que-es-el-ftp/

A l'hora d'enviar dades a través de la xarxa, FTP utilitza dues variants. Està la manera per una part l'activa i per altra la passiva. La manera activa, que és la més utilitzada, és en la que l'usuari que vol pujar o descarregar arxius del servidor és qui especifica el port per fer aquesta transmissió de dades. En canvi, en la passiva, el port per realitzar aquesta transmesa és escollit de manera aleatòria, el que solucionar els problemes de transmissió en rars casos on l'ordinador detecta falsament la connexió com a fraudulenta.




Modes FTP actiu i passiu. Font: https://cloudzy.com/blog/ftp-active-vs-passive-which-one-is-right-to-your-needs/

## Avantatges i desavantatges

El principal avantatge d'FTP és que és molt simple d'utilitzar i molt efectiu en la seva tasca d'enviar i rebre arxius, sent també molt ràpid i fiable, gràcies a estar basat en TCP.

No obstant això, FTP també té bastants defectes, sent un d'aquests el no xifrar la informació, semblant a HTTP, que tampoc ho fa. Però això se soluciona utilitzant una alternativa que sí que ho fa, com el ja mencionat abans FTPS o SCP. Tanmateix, el principal problema d'FTP és que cada vegada es va fent més obsolet, ja que es va crear fa uns 50 anys i no s'ha millorat molt des de llavors, i cada vegada van sortint millors opcions que el més possible és que el reemplaçaran en un futur, opcions que tenen les mateixes característiques que FTP, però millor.

## Aplicacions

La principal aplicació d'FTP és la transmissió privada d'arxius, és a dir, un servei en el qual només un grup petit de persones específiques poden accedir-hi als arxius. Un exemple d'això seria tenir en una oficina un servidor FTP per poder compartir ràpidament arxius entre els treballadors, sense haver de pujar-ho al núvol i descarregar-ho, que seria molt tardat, o encara pitjor, haver de connectar cables entre ordinadors o USB.

Una altra aplicació perfecta per a FTP estaria orientada a un àmbit més privat. En aquest cas hi hauria un servidor a casa teva en el que posaràs arxius que vulguis compartir amb qualsevol dispositiu de la teva xarxa local. Amb això, tu des de qualsevol dispositiu (mòbils, ordinadors o fins i tot televisions intel·ligents) et podràs connectar al servidor i gestionar aquests arxius. Això fa que hagis d'estar sempre amb cables o connectant dispositius USB.

FTP també podria ser utilitzat com a sistema d'emmagatzemat en el núvol, encara que és més limitat que les noves opcions que van apareixent. Tanmateix, podries tenir perfectament el teu servidor FTP en qualsevol lloc del món i en el que puguis pujar, baixar i organitzar els teus arxius de manera senzilla, sense haver d'estar utilitzant una memòria d'emmagatzemament externa.

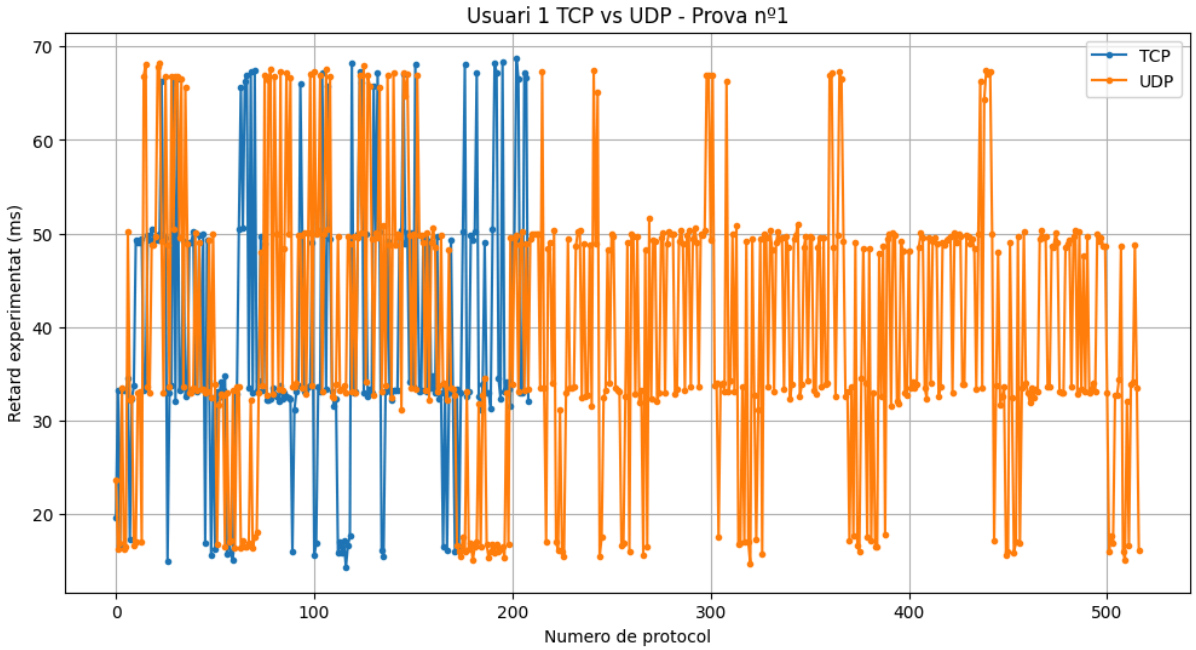
### Conclusions

En conclusió, FTP és un protocol que ha sigut i segueix sent molt important per a la indústria tecnològica actual. A més, el fet de ser tan simple i accessible fa que pugui ser utilitzat per un públic major, el qual el pot utilitzar en àmbits privats de manera senzilla. I encara que vagin sortint alternatives més modernes i eficaces, encara segueix sent una peça crucial en l'àmbit professional.

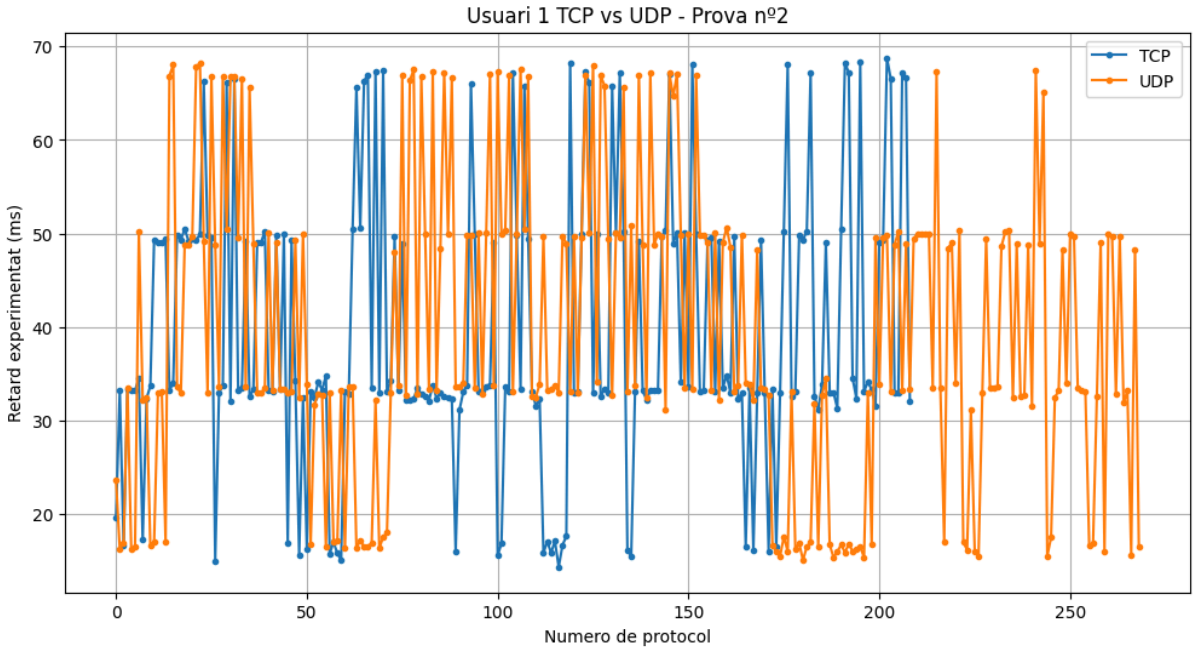
## Resultats obtinguts

Gràfics i taules addicionals que s'han extret de les proves i utilitzats per a la formulació de les conclusions.

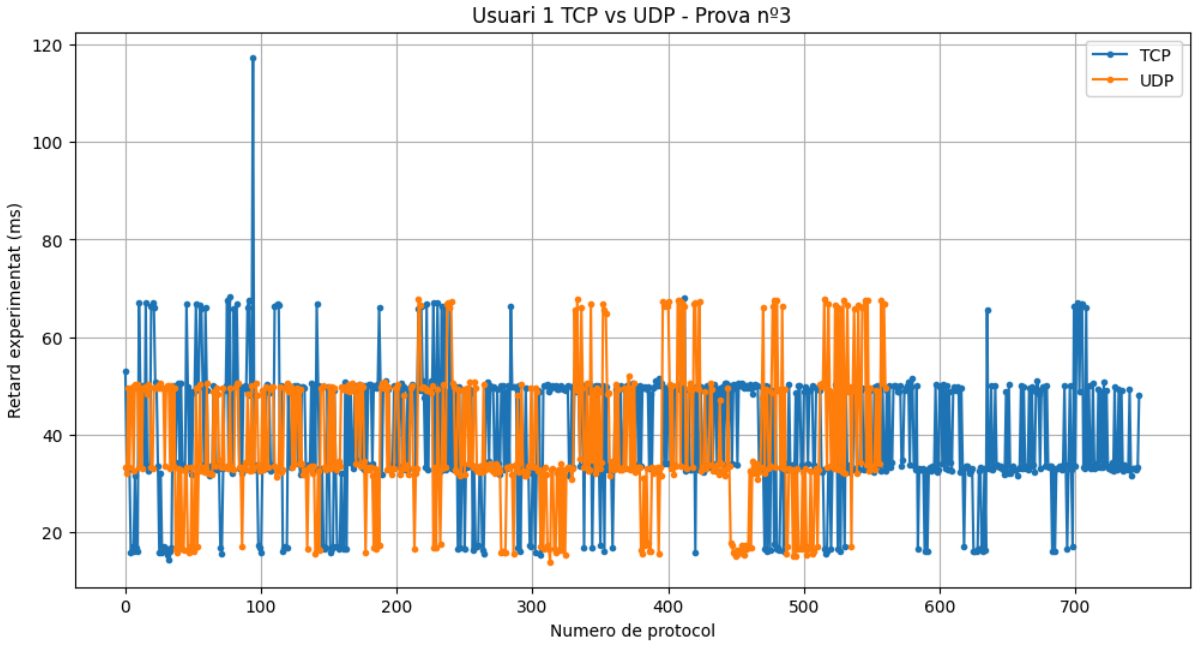
### Usuari núm. 1



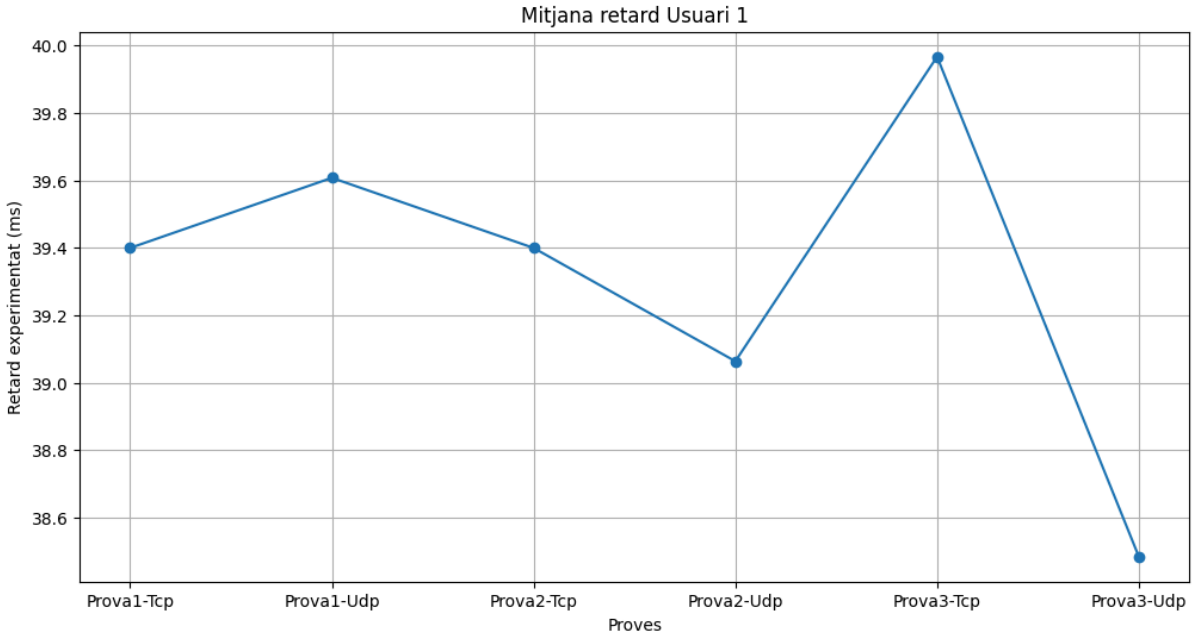

Comparació del retard entre TCP i UDP en la primera prova. Usuari núm. 1. Elaborat per David Didenco.




Comparació del retard entre TCP i UDP en la segona prova. Usuari núm. 1. Elaborat per David Didenco.

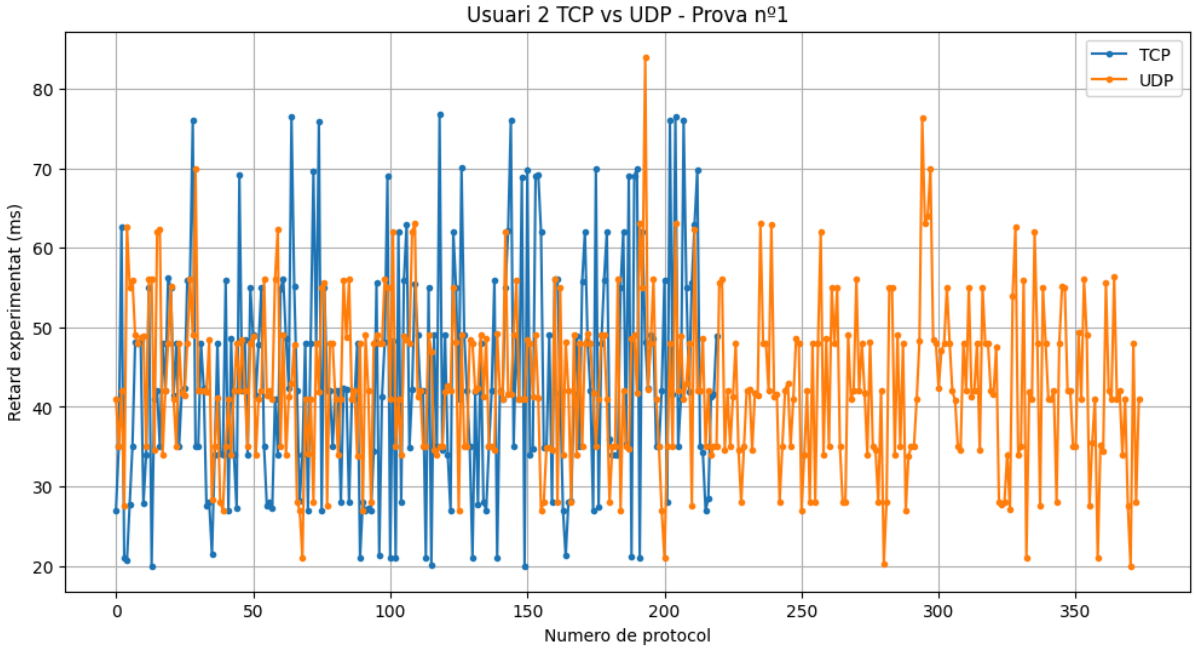



Comparació del retard entre TCP i UDP en la tercera prova. Usuari núm. 1. Elaborat per David Didenco.

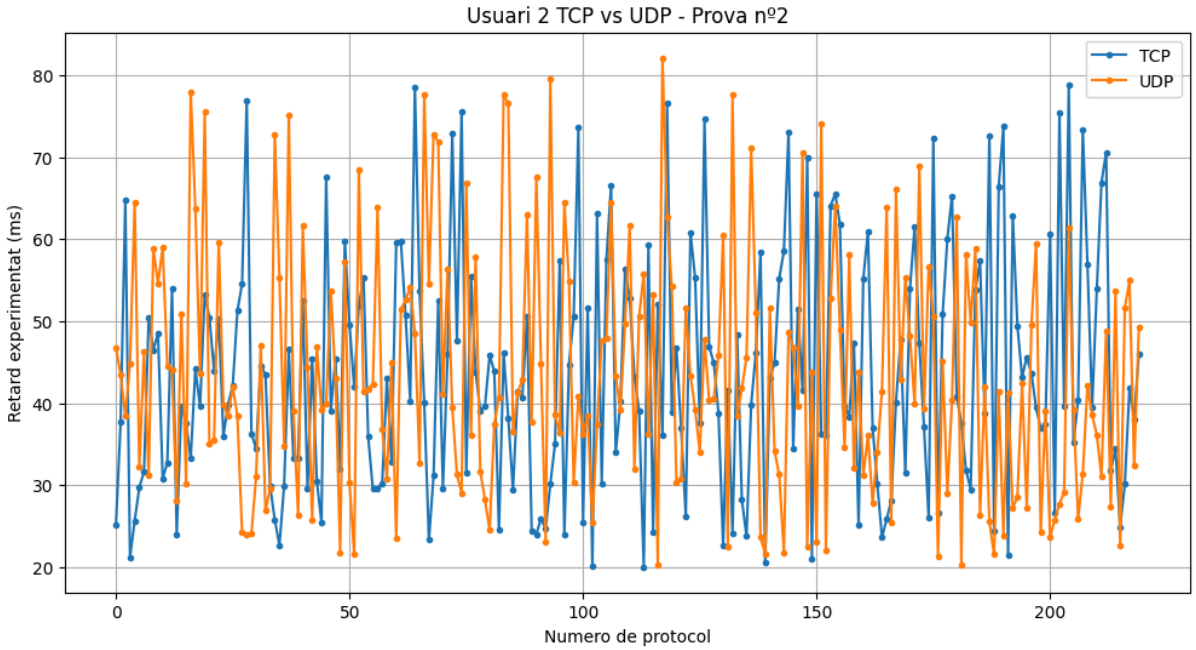



Mitjana del retard en totes les proves. Usuari núm. 1. Elaborat per David Didenco.

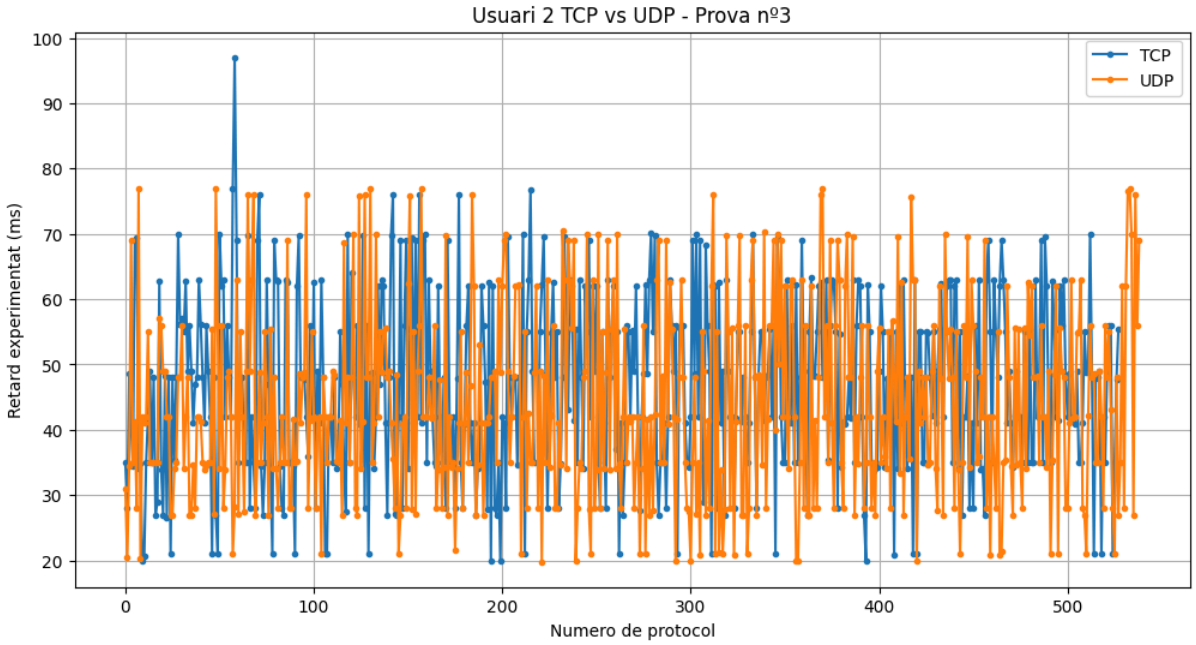
## Usuari núm. 2



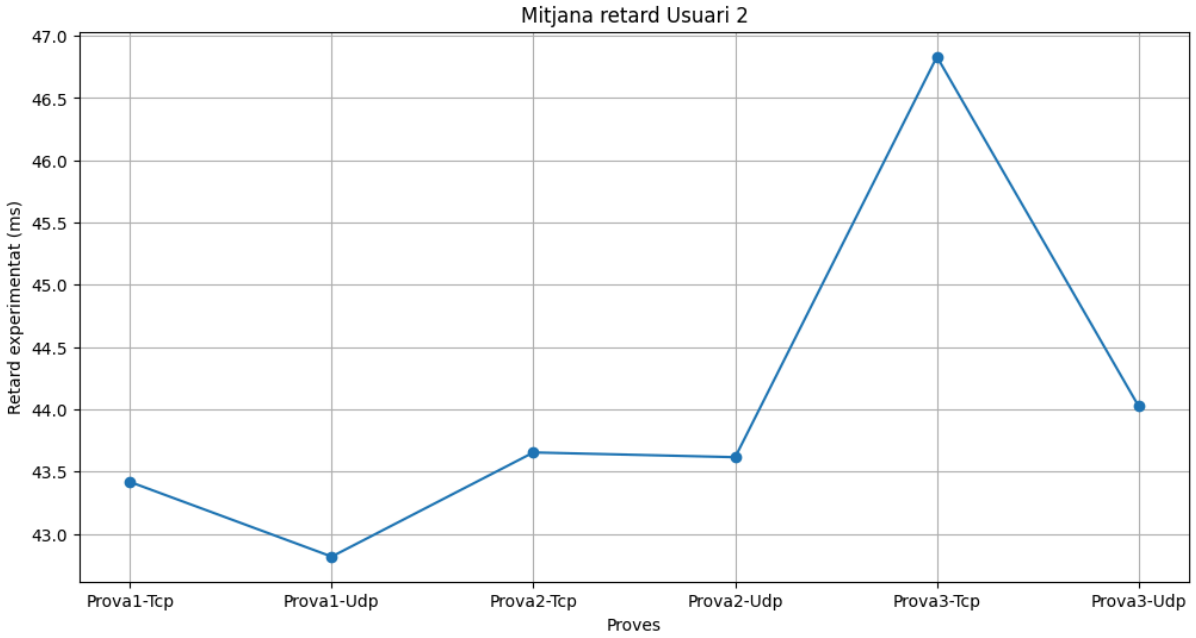

Comparació del retard entre TCP i UDP en la primera prova. Usuari núm. 2. Elaborat per David Didenco.




Comparació del retard entre TCP i UDP en la segona prova. Usuari núm. 2. Elaborat per David Didenco.

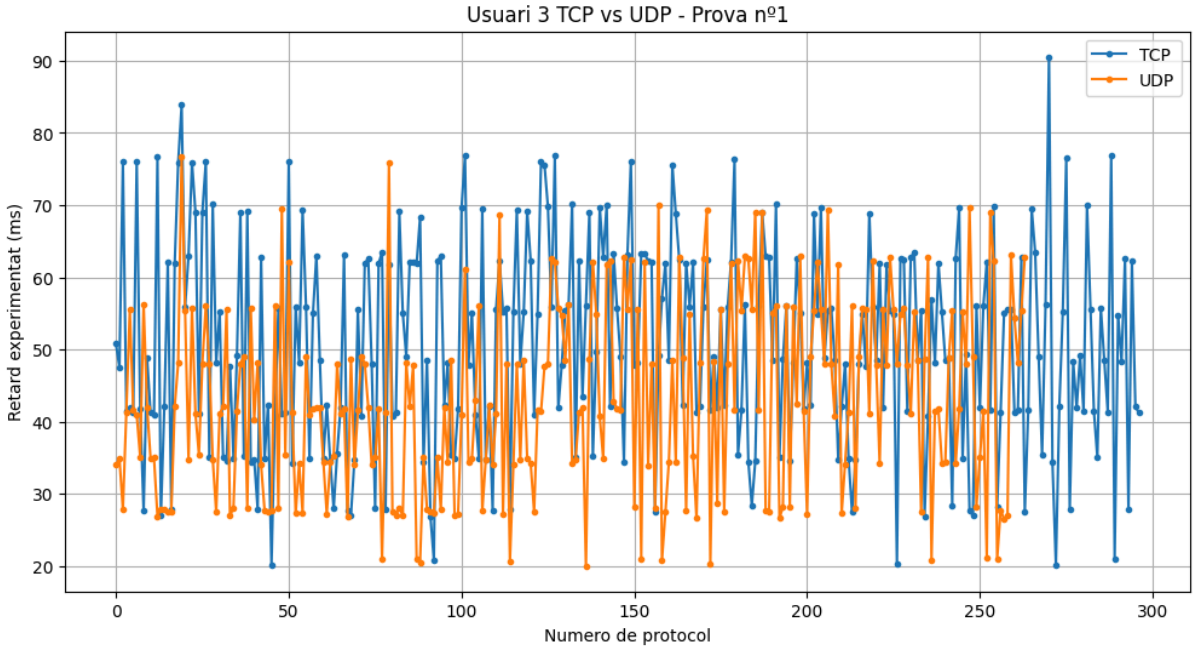



Comparació del retard entre TCP i UDP en la tercera prova. Usuari núm. 2. Elaborat per David Didenco.

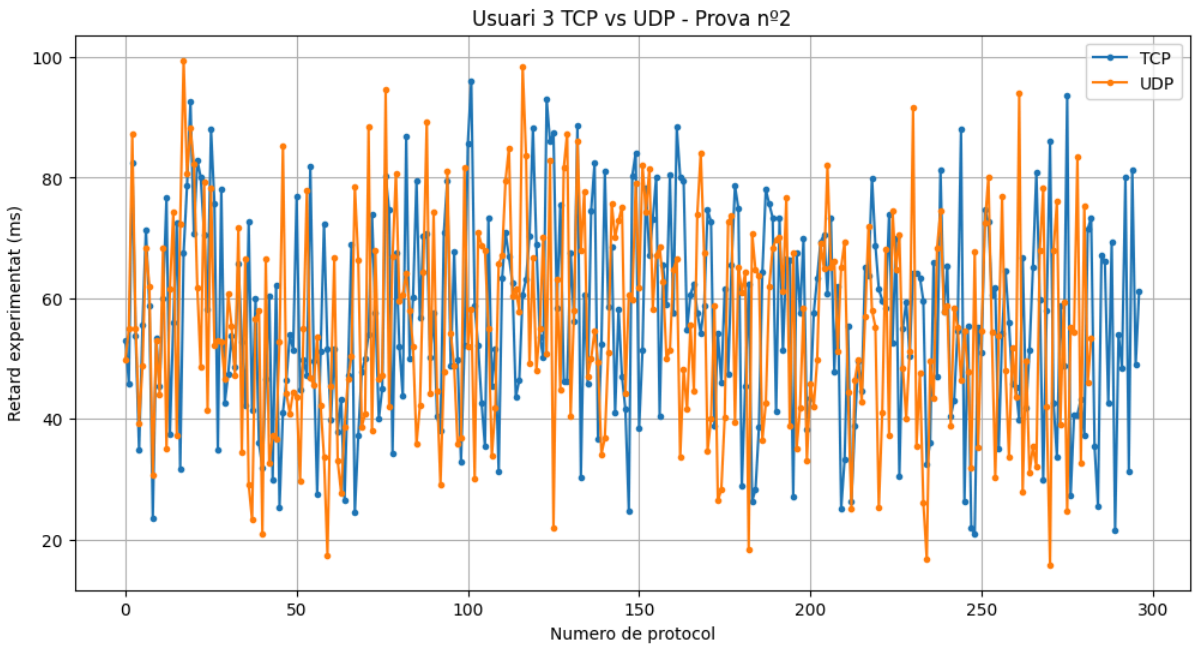



Mitjana del retard en totes les proves. Usuari núm. 2. Elaborat per David Didenco.

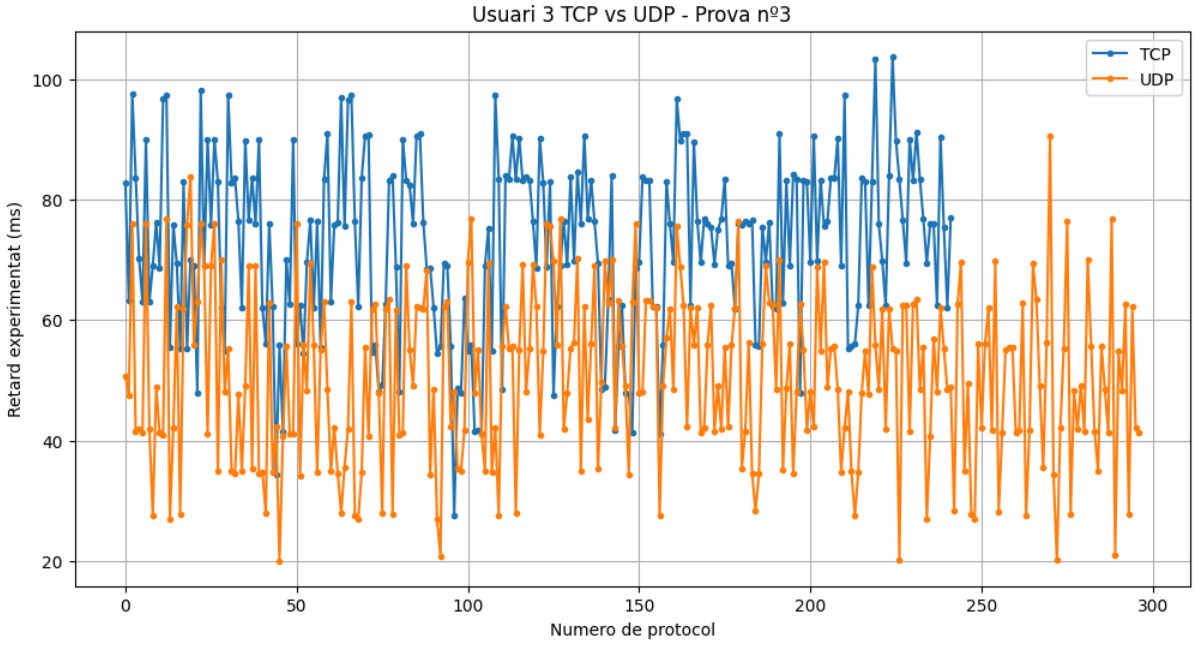
## Usuari núm. 3



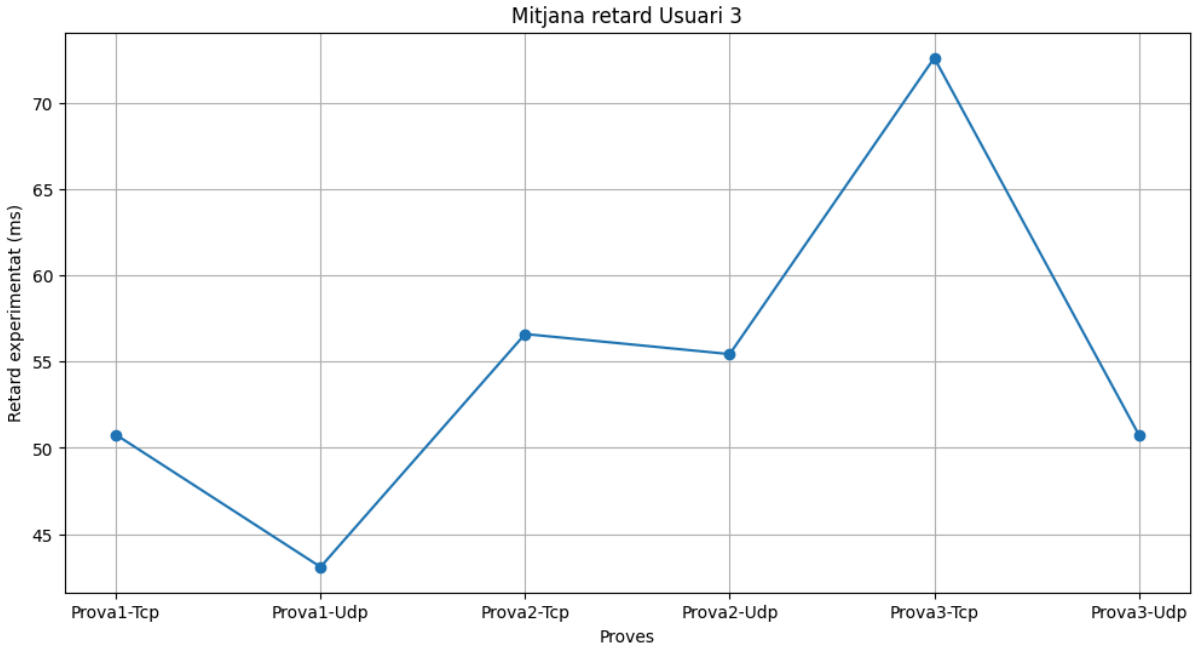

Comparació del retard entre TCP i UDP en la primera prova. Usuari núm. 3. Elaborat per David Didenco.




Comparació del retard entre TCP i UDP en la segona prova. Usuari núm. 3. Elaborat per David Didenco.

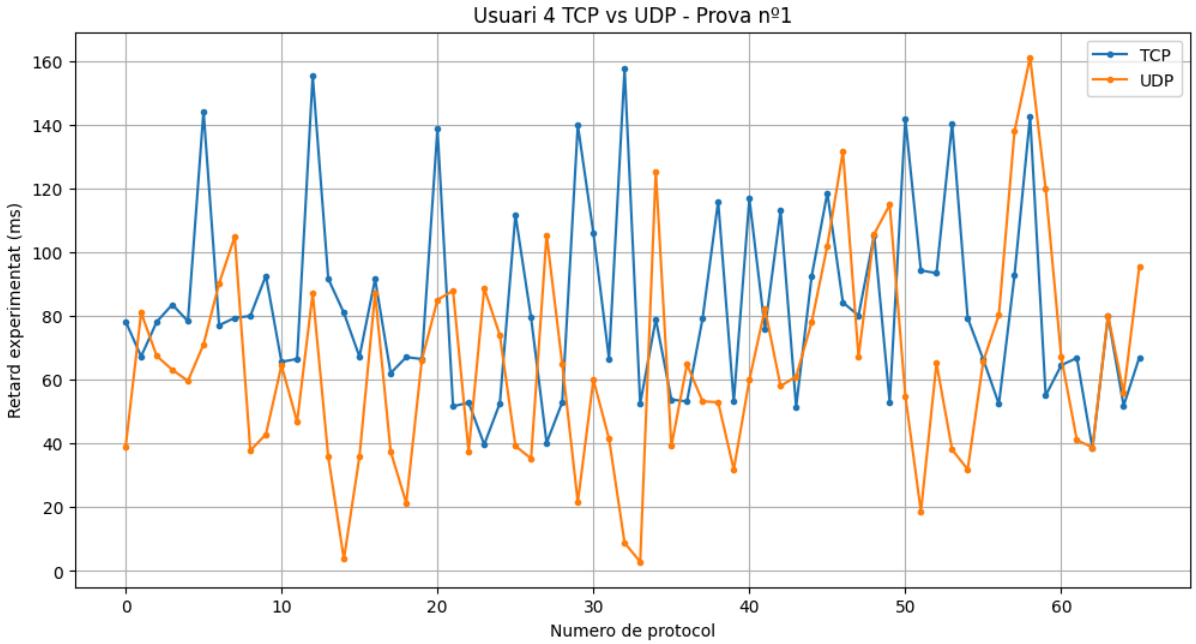



Comparació del retard entre TCP i UDP en la tercera prova. Usuari núm. 3. Elaborat per David Didenco.

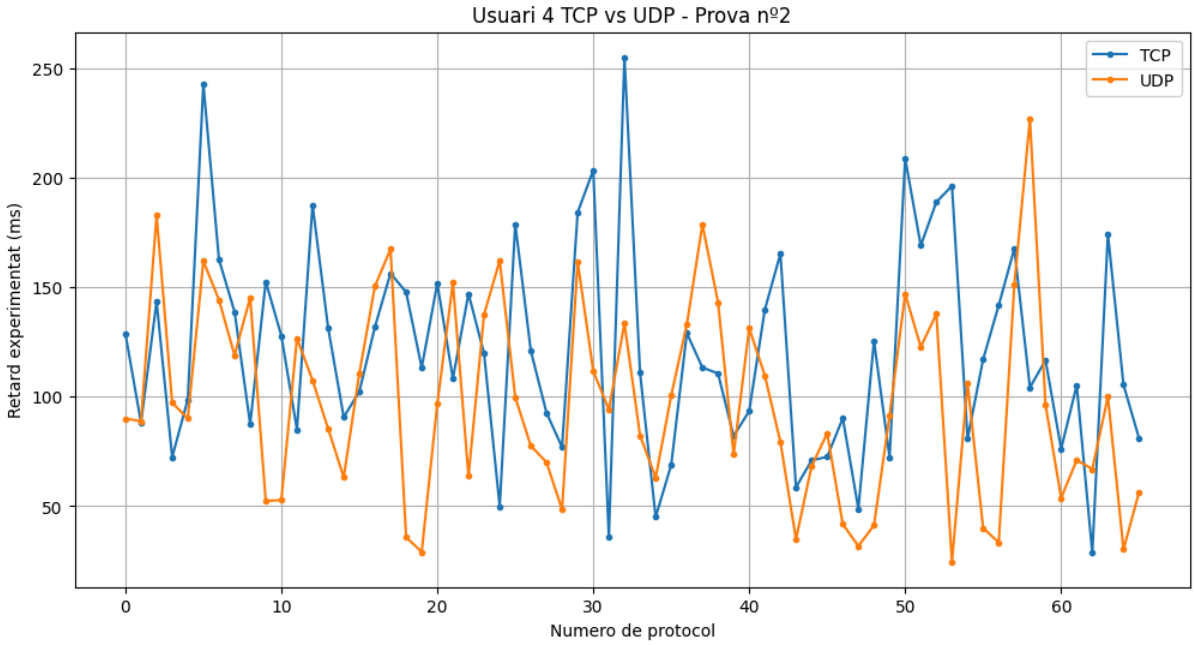



Mitjana del retard en totes les proves. Usuari núm. 3. Elaborat per David Didenco.

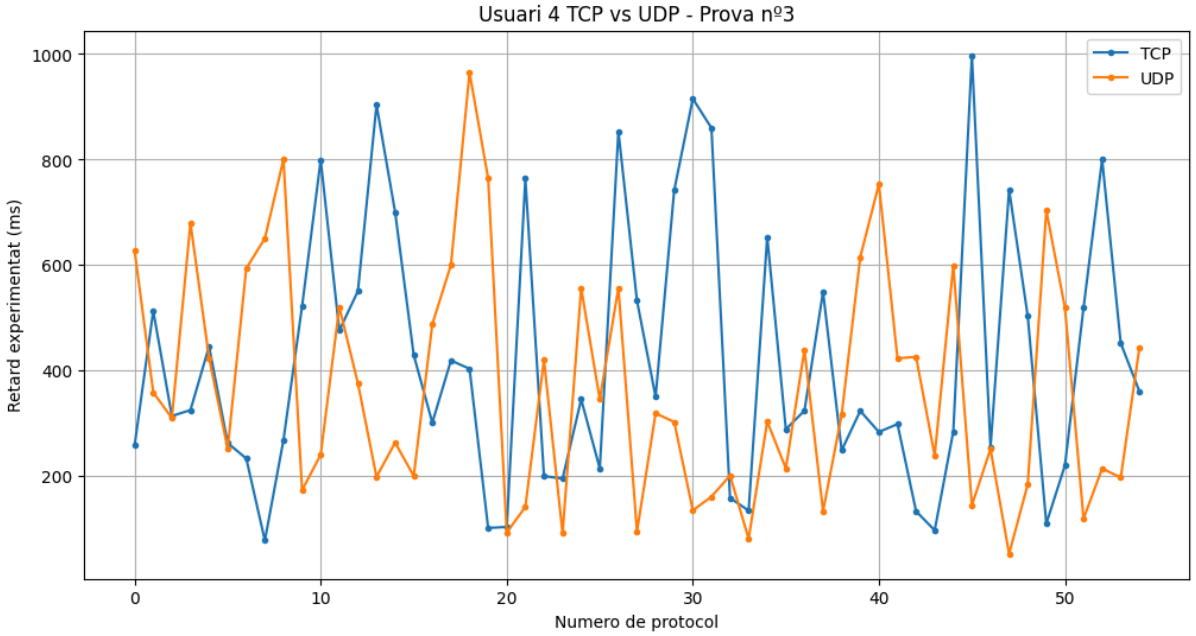
## Usuari núm. 4



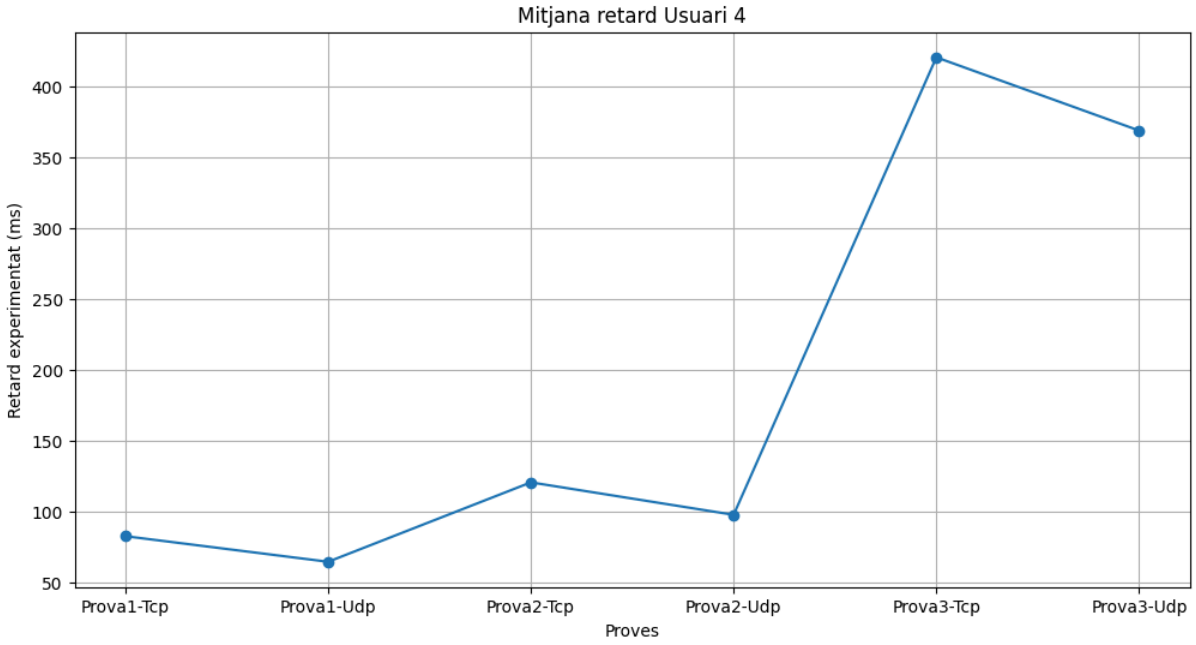

Comparació del retard entre TCP i UDP en la primera prova. Usuari núm. 4. Elaborat per David Didenco.




Comparació del retard entre TCP i UDP en la segona prova. Usuari núm. 4. Elaborat per David Didenco.




Comparació del retard entre TCP i UDP en la tercera prova. Usuari núm. 4. Elaborat per David Didenco.




Mitjana del retard en totes les proves. Usuari núm. 4. Elaborat per David Didenco.

## Taules Wireshark

| | Paquets Usuari 1 | Paquets Usuari 2 | Paquets Usuari 3 | Paquets Usuari 4 | Packet Loss Usuari 1 | Packet Loss Usuari 2 | Packet Loss Usuari 3 | Packet Loss Usuari 4 | Average |
|---|---|---|---|---|---|---|---|---|---|
| Test 1 UDP | 15385 | 14804 | 15040 | 2036 | 2,96% | 3,90% | 3,40% | 6,68% | 4,23% |
| Test 3 UDP | 12584 | 12928 | 12736 | 11773 | 3,65% | 6,98% | 10,14% | 19,91% | 10,17% |

Taula amb la pèrdua de paquets i enviats en total de cada usuari a les proves 1 i 3 d'UDP. Elaborat per David Didenco.

Prova 1 TCP wireshark Conversation

| Bits/s A → B | Bits/s B → A | Bytes | Bytes A → B | Bytes B → A | Dirección A | Dirección B | Duración | Inicio rel | Packets A → B | Packets B → A | Paquetes | Puerto A | Puerto B | Stream ID | Bytes/Paquet | Mitjana |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 42737 | 81611 | 3519658 | 1209671 | 2309987 | | | 226.438430156 | 24.705725164 | 18659 | 24625 | 43284 | 52461 | 7777 | 9 | 81,315 | 80,399 |
| 47632 | 80558 | 3919459 | 1456365 | 2463094 | | | 244.601131418999 | 6.55695164 | 22620 | 26494 | 49114 | 50036 | 7777 | 4 | 79,803 | |
| 47105 | 80230 | 3908865 | 1446014 | 2462851 | | | 245.578672807 | 5.56826280 | 22488 | 26535 | 49023 | 59625 | 7777 | 3 | 79,735 | |
| 45250 | 78934 | 3822438 | 1392818 | 2429620 | | | 246.240416237 | 4.91608101 | 21490 | 25851 | 47341 | 58402 | 7777 | 2 | 80,743 | |

Taula completa de les interaccions del servidor i els usuaris a la prova 1 utilitzant TCP. Elaborat per David Didenco.

Prova 1 UDP wireshark Conversation

| Bits/s A → B | Bits/s B → A | Bytes | Bytes A → B | Bytes B → A | Dirección A | Dirección B | Duración | Inicio rel | Packets A → B | Packets B → A | Paquetes | Puerto A | Puerto B | Stream ID | Bytes/Paquet | Mitjana |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 19513 | 45200 | 4177808 | 7764608 | 3401348 | | | 27.247.434.258.000.000 | 22.093.610.678 | 11177 | 52388 | 63565 | 33626 | 7777 | 1 | 65,725 | 62,261 |
| 25162 | 10516 | 4237486 | 8181386 | 3419348 | | | 260.113.618.782 | 235.640.461 | 14731 | 54348 | 69079 | 64824 | 7777 | 2 | 61,343 | |
| 26511 | 10746 | 4289546 | 8488215 | 3440725 | | | 25.614.049.756.300.0 | 27.522.217.251 | 15332 | 54872 | 70204 | 62650 | 7777 | 3 | 61,101 | |

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | 00 | | | | | | | | |
| | | | | | | | 276. | 7.59 | | | | | | | |
| | | 451 | | 355 | | | 065. | 3.58 | | | | | | | |
| | 103 | 456 | 957 | 715 | | | 491. | 6.05 | 1740 | 5675 | 741 | 637 | 777 | | 60,87 |
| 27744 | 081 | 7 | 408 | 9 | | | 422 | 9 | 7 | 5 | 62 | 28 | 7 | 0 | 4 |

Taula completa de les interaccions del servidor i els usuaris a la prova 1 utilitzant UDP. Elaborat per David Didenco.

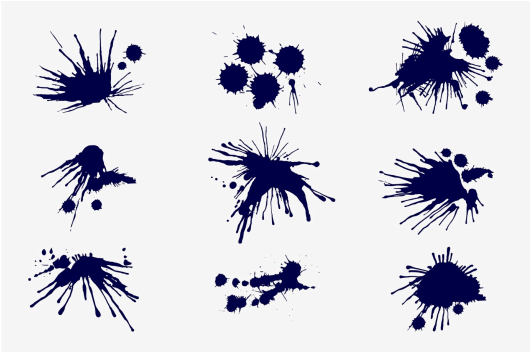
### Arxiu amb tota la informació recollida

En aquest arxiu comprimit es troben tots els resultats de les proves, així com els arxius de gravació de Wireshark. Aquests arxius són els que s'han utilitzat per a realitzar els gràfics, taules, etc. per a la posterior anàlisi i formulació de conclusions.

Enllaç:
https://drive.google.com/file/d/1044nLrIVpBsAtJD6euvNUrqVKhQLO8vv/view?usp=sharing

## Atribució a recursos utilitzats en el videojoc

En aquesta secció donaré atribucions a uns recursos visuals que s'han utilitzat per a la creació del videojoc.



Il·lustració d'impactes de bales de paintball. Utilitzat al videojoc. Creat per Freepik: freepik

"Paintball Marker Spyder MR 100" (https://skfb.ly/o9RCY) by olgasardykova is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).

## Altres

Aquest arxiu comprimit conté tot el procés gravat de la creació d'aquest treball de recerca (té una mida de 20 GB):
https://drive.google.com/file/d/1knAlNPbBlzgw4AMmk17igQDQLJcTS_Jq/view?usp=sharing
**Atenció:** La carpeta descomprimida té una mida de 100 GB aproximadament. Ja que com els arxius són tots vídeos, ocupen molt d'espai.

Tot el procés de creació del videojoc, així com els arxius d'aquest estan pujats a la meva compta de GitHub, un servei de núvol que et permet pujar projectes relacionats amb la programació. La pàgina està en angles, però simplement és per un tema de format. L'enllaç al projecte és el següent: https://github.com/Estikno/PaintballOnline