Universitat Oberta de Catalunya

uoc.edu

Eficiència dels sistemes de xifrat en protocols de serveis de xarxa

Nom Estudiant : Victor Martin Avellana
Pla d'estudis de l'estudiant : Grau d'enginyeria informàtica
Treball final de grau : Seguretat Informàtica

Nom Consultor/a : Gerard Farràs Ballabriga
Nom Professor/a responsable de l'assignatura : Helena Rifà Pous

Data Lliurament : 10/01/2023

FITXA DEL TREBALL FINAL

Títol del treball:	*Eficiència dels sistemes de xifrat en protocols de serveis de xarxa*
Nom de l'autor:	*Victor Martin Avellana*
Nom del consultor/a:	*Gerard Farràs Ballabriga*
Nom del PRA:	*Helena Rifà Pous*
Data de lliurament (mm/aaaa):	*01/2023*
Titulació o programa:	*Grau d'enginyeria informàtica*
Àrea del Treball Final:	*Seguretat informàtica*
Idioma del treball:	*Català*
Paraules clau	*Xifrat, RSA/ECDSA, TLS*

Resum del Treball:

Avui en dia, la comunicació per mig d'Internet ha esdevingut imprescindible en la nostra vida diària a tots els nivells; social, professional, amb els bancs, amb l'estat o amb el comerç electrònic. Internet proporciona l'accés, però necessitem poder garantir la seguretat d'aquestes comunicacions. El protocol SSL/TLS va ser implementat arrel d'aquest fet.

El protocol TLS fa ús de criptografia de clau publica. Aquest, es val de les *cipher suites* en el procés d'establir una connexió xifrada. Aquests protocols son els encarregats d'oferir els serveis d'autenticació, xifrat, intercanvi de claus i integritat.

Assumida la necessitat de seguretat, cal garantir que estem implementant unes claus que aporten eficiència al procés de xifrar la informació. Ens ha calgut un estudi de rendiment i una comparativa entre claus RSA i ECDSA, a fi de conèixer els tipus de claus més optimes a implementar.

Les *cipher suites* van ser creades per a ser una entitat en si mateixes. Els protocols de cada *cipher suite* son compatibles entre ells. En conjunt sumen, aporten un grau de seguretat que ha estat validat com segur per IANA i RFC.

Els certificats de clau pública de tipus ECDSA fan ús d'algorismes de corba el·líptica per a xifrar i signar informació. La seva implementació, junt amb l'ús de TLSv1.3 aporta la màxima eficiència en el rendiment de les aplicacions que han d'oferir una comunicació xifrada.

Abstract (in English, 250 words or less):

Nowadays, communications through Internet have become essential in our daily life at all levels; social, professional, financial affairs, to pay tax or with the *eCommerce*. At the same time, all business have to submit their documentation trough Internet. The network provides access, but we must have to assure the security of those communications because IP Protocol was not designed to be secure itself. The SSL/TLS protocol was made as result of this.

The TLS protocol uses public-key Cryptography. This, is using *Cipher Suites* to create a secure connection. These protocols are responsible for providing authentication, encryption, key exchange, and integrity services.

After we have understood we need security connections, we have to implement keys that provide efficiency to the proces of encrypting the data. In order to know the keys with the highest performance, we have carried out a performance study and a comparison between ECDSA and RSA keys.

The cipher suites were created to be an entity in themselves. The protocols of each cipher suite are compatible with each other. Together they provide a degree of security that has been validated as a security by IANA and RFC.

ECDSA public key certificates make use of elliptic curve algorithms to encrypt and sign information. This implementation, together with the use of TLSv1.3 brings the highest efficiency in the performance of applications that must offer an encrypted communication.

Per la meva dona, Montse. Sense tu
res no hagés estat possible

Índex

Llista de figures

Llista de taules

1. Introducció

1.1. Motivació del Treball

Aquest treball ha nascut de la necessitat de conèixer quin benefici pot aportar en la empresa on treballo actualment (Grupo Cementos Portland Valderrivas) la implementació d'un xifrat creat per claus de tipus ECDSA en els servidors de SAP.

Som una empresa filial del grup FCC, on son allotjats els servidors de SAP, i ens connectem a la central per mig d'un enllaç a 100 Mbps. Els usuaris experimenten una demora d'uns 2 minuts en realitzar una cerca mitjanament complexa en SAP. Es evident que el rendiment del programari afecta directament a la tasca que unes 2.000 persones realitzen diàriament de manera simultània, i ens vam proposar veure com es podia millorar.

Aquest treball en concret ha volgut estudiar els tipus de xifrat creats per claus RSA i ECDSA a fi d'esbrinar els beneficis que pot aportar l'ús de certificats de tipus ECDSA.

Una primera cerca d'informació provinent d'altres treballs similars ens va conduir a l'afirmació de que un xifrat creat per un certificat RSA de 3072 bits proporciona el mateix grau de seguretat que un xifrat creat per un certificat ECDSA de 384 bits. Aquesta conclusió va ser el punt de partida per que s'iniciés el projecte en la empresa de canviar el tipus de xifrat en els servidors de SAP.

Tanmateix, ha estat el punt de partida per desenvolupar aquest treball. Hem volgut conèixer les característiques dels xifrats RSA I ECDSA, i hem creat un entorn que simula un ample de banda similar per a realitzar proves de rendiment de certificats creats a tal efecte. Ens hem centrat en dos aspectes, conèixer el temps en establir una connexió xifrada, i en conèixer el nombre de connexions xifrades simultànies que pot suportar el servidor.

Aquest treball ha aportat coneixement de com opera el xifrat, i la certesa de que l'ús d'un certificat de tipus ECDSA redueix quasi a la meitat el temps de connexió.

1.2. Context i justificació del Treball

Accedim a internet per mig d'un munt de maquinari diferent i transmetem dades per mig d'un medi insegur (*Internet*). Tenim la necessitat que algunes de les dades siguin transmeses des d'un canal segur, sobretot en accedir a entitats bancaries, organismes oficials o amb operacions *comerç electrònic.*

El protocol TCP/IP ens assegura la connexió, però no ens ofereix seguretat per si mateix. Hem fet ús de la *criptografia* per a xifrar la informació a fi de poder crear canals segurs per on transmetre informació. L'ús de certificats digitals de clau publica es qui ens proporciona aquestes característiques a les connexions segures.

TCP / IP es val d'altres protocols, com per exemple SSL/TLS per a crear un canal per on transmetre les dades. SSH crea una connexió segura de punt a punt, tot fent ús d'un protocol de la capa d'aplicació.

En aquests moments *el xifrat* usat en les transmissions per internet és creat en la seva gran majoria per claus publiques de tipus RSA. Aquest tipus de *xifrat* ha esdevingut un estàndard en les connexions segures. Malgrat això, poca a poc, moltes aplicacions van adoptant de manera nativa el *xifrat* creat per claus de tipus ECDSA. Aplicacions forca conegudes, com per exemple; els navegadors web, servidor web *Apache*, *FileZilla, SAP*, i fins i tot *Amazon Web Services*, ofereixen suport i l'accepten en primer lloc claus de tipus ECDSA en qualsevol connexió que duen a terme.

Les connexions han d'oferir seguretat, però no per aquest fet ha de minvar el rendiment dels equips en xifrar i desxifrar les dades. Fruit d'aquest fet les claus ECDSA van ser introduïdes en SSL/TLS en l'any 2008, però la seva implementació ha estat lenta. El xifrat amb claus RSA ha esdevingut un estàndard, ha sofert diferents auditories, i podem afirmar que avui en dia ha assolit la maduresa. Per contra, ECDSA no ha estat objecte de tantes auditories, i ha estat víctima d'implementacions errònies que han fet possible vulnerar la seguretat que oferien. Aquest fet sembla que hagi marcat unes preferències per el xifrat RSA per part dels creadors dels webs.

Es conegut el fet que *"Una clau publica RSA de 2048 bits proporciona un nivell de seguretat de 112 bits. ECDSA realitza la mateixa feina amb una clau publica de 224 bits. Això ens mostra que ECDSA pot ser implementada en dispositius IoT sense haver d'assignar molts recursos"* [1].

Aquest estudi vol verificar aquest fet en algunes de les aplicacions més usades avui en dia, a fi de comprovar la eficiència que pot oferir implementar un sistema de clau publica amb xifrat ECDSA, amb la finalitat de cercar un model que ofereixi el màxim de seguretat amb un nivell òptim de rendiment del servei.

1.3. Objectius del Treball

L'objectiu d'aquest treball es realitzar un estudi dels diferents tipus de certificats existents, així com les seves longituds de claus existents, i poder avaluar com afecta al rendiment de les aplicacions.

Objectius:

- Conèixer els diferents tipus de certificats, així com les longituds de claus més usades.
- Estudiar i analitzar com interactuen els certificats amb les aplicacions, especialment en el procés d'establiment de connexió (Handshake).
- Estudi i anàlisi del comportament de les aplicacions en el procés de xifrar la informació.
- Estudi de com afecta la versió de TLS i el certificat respecte al rendiment que proporciona l'aplicació.
- Anàlisi de quin efecte té l'elecció d'un determinat certificat respecte a la cipher suite seleccionada per l'aplicació en el procés de Handshake.
- Examinar i avaluar el rendiment que proporciona cadascun dels certificats. Crear uns manuals per la creació de certificats SSL i SSH.
- Verificar els avantatges que pot proporcionar l´ús de certificats creats amb claus publiques de tipus ECDSA en aplicacions Web, aplicacions FTP i SSH.
- Dur a terme un estudi comparatiu de rendiment amb un servidor web Apache, un servidor FTP: FileZilla i un equip que faci us de SSH.

1.4. Enfocament i mètode seguit

Per a dur a terme aquest treball volem analitzar com es produeix l'establiment de les connexions segures a fi de conèixer com es realitza el procés amb connexions SSL/TLS i SSH, a fi de poder mesurar el temps ocupat en realitzar la connexió.

Ens interessa conèixer tant el temps que triga en establir una connexió segura com el nombre de connexions que pot suportar el equip que fa de servidor en un moment donat. Volem conèixer com afecta al rendiment de l'aplicació que s'està executant com a servidor. En cap cas plantegem reduir el nivell de seguretat proporcionada, sinó que volem establir fins a quin punt podem augmentar el grau de seguretat de la connexió, donada per la longitud de clau, sense que afecti negativament al rendiment de l'aplicació feta servir com a servidor.

Per a realitzar aquesta tasca adoptar el següent mètode de treball anomenat Kanban:

- Aquesta tecnologia ens permet visualitzar i gestionar els fluxos de treball.
- Permet limitar el treball en curs, definir polítiques de processos,
- Visualització global del estat del treball a realitzar. Veiem tasques fetes, tasca realitzant-se i tasques pendents.
- Identificació de les fases de treball. Establir límits de principi i fi de tasca.
- Establir com interactuen les diferents tasques de cada fase. Definir dependències. Definir ordre d'actuació.
- Veure com evoluciona cada fase.

Aquest mètode de treball queda establert de la següent manera:

- Definició de l'àmbit de treball
- Instal·lació del programari a avaluar: Servidor Apache, FileZilla, OpenSSL i OpenSSH.
- Creació de certificats SSL/TLS i SSH.
- Investigació sobre criptografia de clau pública
- Investigació sobre el protocol SSL/TLS.

- Investigació i diferencies en certificats RSA / ECDSA.
- Investigació del protocol SSH.
- Estudi del Handshake. SSL/TLS i SSH.
- Investigació recomanacions seguretat FIPS 140-2 i NSA.
- Establiment condicions de xarxa on realitzar els proves.
- Definició tasques a mesurar: tipus i longitud de clau.
- Avaluació de resultats obtinguts.
- Obtenció de conclusions.

Volem aportar coneixement en base a primer; presentar el que volem avaluar. En segon lloc, preparar un marc on obtenir dades del rendiment ofert per els diferents certificats. Totes aquestes dades han d'anar parelles a oferir un mínim de seguretat exigida, fins a conèixer fins quin punt podem augmentar la longitud de la clau sense que aquest fet afecti al rendiment del servidor. En tercer lloc procedirem a la captura de dades. En obtenir dades, establirem comparatives i podrem extreure conclusions.

1.5. Planificació del Treball

Per a realitzar el projecte crearem un escenari on durem a terme les diferents captures de dades.

- Es tracta de definir un escenari que simuli un entorn real de connexió a internet. Per això definirem diferents velocitats en una xarxa d'àrea local, tot fent ús d'una configuració de programari de tipus client - servidor.
- Durem a terme la instal·lació del programari OpenSSL, i crearem els certificats.
- Durem a terme la instal·lació del programari Apache i FileZilla.
- Durem a terme la instal·lació del programari SSH. Creació de certificats.

Establirem un seguit de tasques a realitzar

- Mesura de rendiment Web en connexions SSL/TLS. Aplicacions web: Google Crome, Mozilla Firefox i Microsoft Edge.
- Mesura rendiment connexions SSL/TLS.
- Mesura rendiment FTPS en connexions SSL/TLS. Aplicació: FileZilla.
- Mesura rendiment SFTP en connexions SSH. OpenSSH.

Id	Nombre de tarea	Duración	Comienzo	Fin
1	PAC 1	9 días	vie 30/09/22	mar 11/10/22
2	Definició projecte i amplitud	2 días	vie 30/09/22	dom 02/10/22
3	Confecció document	6 días	lun 03/10/22	lun 10/10/22
4	Entrega Pac 1	0 días	mar 11/10/22	mar 11/10/22
5	PAC 2	24 días?	mié 12/10/22	mar 08/11/22
6	Definicio terminilogia	22 días	mié 12/10/22	dom 06/11/22
7	Creacio claus	4 días	lun 24/10/22	jue 27/10/22
8	Instalacio OpenSSL	1 día	lun 24/10/22	lun 24/10/22
9	Creació certificats	3 días	mar 25/10/22	jue 27/10/22
10	Instalacio programari	12 días?	vie 28/10/22	mar 08/11/22
11	Instalacio servidor Web Apache	2 días?	vie 28/10/22	sáb 29/10/22
12	Instalacio servidor FileZIlla	2 días?	vie 28/10/22	sáb 29/10/22
13	Instalació client FIleZilla	2 días?	vie 28/10/22	sáb 29/10/22
14	Config. Segura amb certificats	4 días	dom 30/10/22	mié 02/11/22
15	Config. conexio segura Servidor Web Apache	2 días	dom 30/10/22	lun 31/10/22
16	Config. Conexio segura FileZIlla	2 días	mar 01/11/22	mié 02/11/22
17	Configuracio SSH	5 días?	jue 03/11/22	lun 07/11/22
18	Entrega Pac 2	0 días	mar 08/11/22	mar 08/11/22
19	PAC 3	22 días?	mié 09/11/22	mar 06/12/22
20	Adquisicio dades	15 días?	mié 09/11/22	dom 27/11/22
21	Proves rendiment web	9 días?	mié 09/11/22	dom 20/11/22
22	Proves rendiment ftp	6 días?	lun 21/11/22	dom 27/11/22
23	Analisi i investigacio	3 días?	lun 28/11/22	mié 30/11/22
24	Validació premises	3 días?	jue 01/12/22	lun 05/12/22
25	Entrega Pac 3	0 días	mar 06/12/22	mar 06/12/22
26	PAC 4	27 días?	mié 07/12/22	mar 10/01/23
27	Memoria final	27 días?	mié 07/12/22	mar 10/01/23
28	Entrega memoria	0 días	mar 10/01/23	mar 10/01/23
29	PAC 5	5 días?	mié 11/01/23	mar 17/01/23
30	Presentació video	5 días?	mié 11/01/23	mar 17/01/23
31	Entrga Pac 5	0 días	mar 17/01/23	mar 17/01/23
32	PAC 6	5 días?	lun 23/01/23	vie 27/01/23
33	Defensa asincrona	5 días	lun 23/01/23	vie 27/01/23
34	Entrega Pac 6	0 días	vie 27/01/23	vie 27/01/23

Figura 1.1. Cronologia planificació tasques

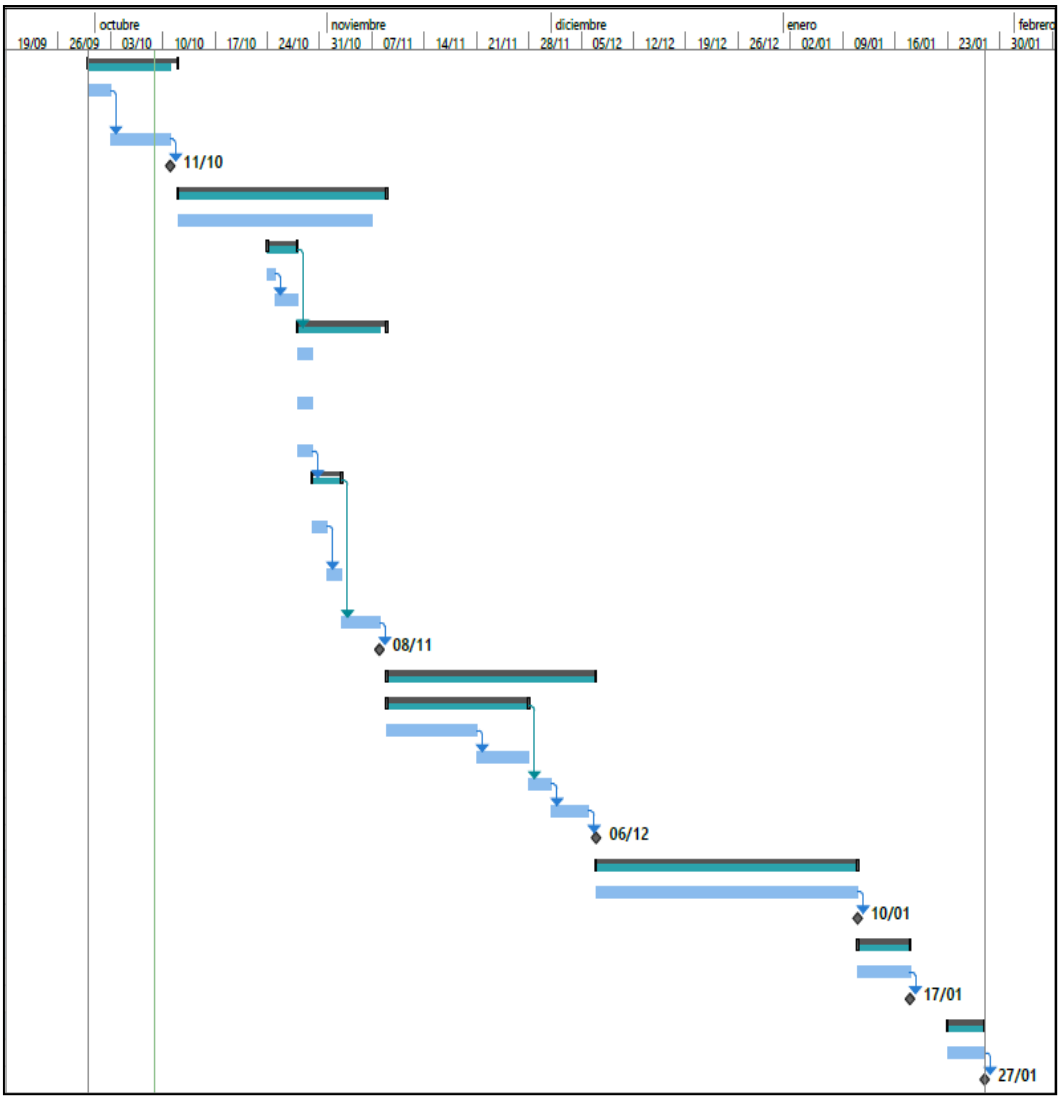


Figura 1.2. Cronologia tasques simultànies

Podem observar com la gran majoria de tasques s'han executat seguint un ordre establert, inici execució i finalització. En acabar ha estat possible realitzar la següent fase.

La fase d'instal·lació de programari s'ha realitzat de manera simultània junt amb la redacció del estat del Art. La fase de creació i prova de certificats ha estar subjecte a diverses revisions a fi d'aconseguir un correcte funcionament amb els navegadors.

1.6. Breu sumari de productes obtinguts

Aquest treball es centra en l'estudi dels tipus de xifrat generat per els certificats ECDSA i RSA. Inclou característiques, modus de xifrar la informació, tipus d'algorismes usats i comparatives de rendiment.

El producte final ens aporta coneixement amb la informació obtinguda de les característiques dels certificats i de com operen. Els resultats de les comparatives de rendiment ens verifica la hipòtesi amb la que treballàvem; millora de rendiment en fer ús de certificats de tipus ECDSA.

Aquest projecte ha generat dos documents on queda documentada les principals línies d'actuació, el desenvolupament i el coneixement que s'ha adquirit. Aquests documents son la Memòria i un vídeo de presentació.

1.7. Breu descripció dels altres capítols de la memòria

La obra es centra en descriure protocols de transport, seguretat i xifrat. Comencem descrivint el medi per on es transmeten dades, i la necessitat d'establir una connexió segura degut a la naturalesa de les dades. Tot seguit descrivim que aporta la criptografia, analitzem els diferents tipus de xifrat i concloem la obra amb una comparativa de rendiments dels diferents tipus de certificat

Capítol 2

Descripció del que se entén per comunicació i necessitat d'establir una connexió segura. Inclou una breu descripció de la xarxa i de la capa de protocols OSI. Introduïm el concepte de Criptografia i descrivim les característiques que demanem que tingui tota connexió segura.

Capítol 3

Introducció de la Criptografia de clau pública, introducció del concepte de certificat i descripció de les característiques dels certificats RSA i ECDSA.

Capítol 4

Coneixement de SSL/TLS. Veure que ens aporta, com opera, i descripció del concepte encaixada de mans.

Capítol 5

Ens aporta coneixement sobre el xifrat SSH, de les seves característiques i de com opera.

La resta de capítols

Aporten informació per a establir un marc on realitzat proves de rendiment amb aplicacions que facin ús de HTTPS, FTPS i SFTP a fi d'obtenir resultats sobre el comportament dels certificats RSA I ECDSA. Arrel del coneixement obtingut hem pogut arribar a conclusions de convertien la hipòtesi del projecte en informació veraç. Aquesta informació verifica igualment les conclusions d'altres projectes realitzats per altres fonts consultades.

2. Seguretat en el transport de dades

En aquest capítol passarem a introduir el concepte de seguretat, entès com a seguretat de les dades transmeses per un medi no segur (la xarxa). Passarem a explicar breument el funcionament d'una xarxa. Enunciarem les principals característiques de tota connexió segura, i coneixerem els avantatges que ens proporciona la criptografia.

2.1. La comunicació

Comencem entenent el fet de comunicar-se. El *paradigma de la comunicació* estableix que un emissor A transmet dades cap a un o molts receptors B per mig d'un canal, i es val d'un codi comú.

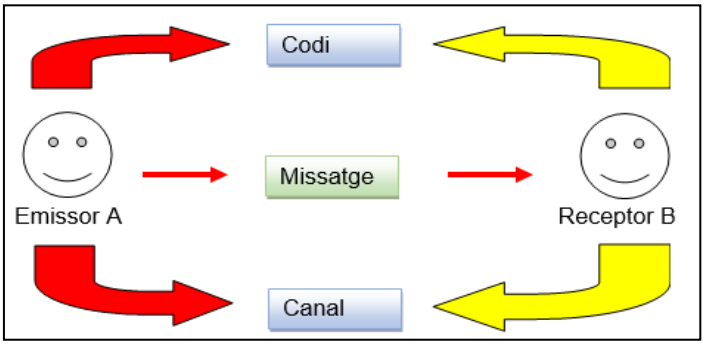


Figura 2.1 Paradigma de la comunicació

El canal per on viatja el missatge ha anat canviant en el temps segons la època. El missatge sempre ha estat codificat segons l'idioma usat, però l'enviament de missatges en temps de guerra va fer que el paradigma agafés un altre camí; *la Criptografia*. Aquesta, va permetre codificar els missatges de manera que el contingut quedés xifrat i tant sols pogués llegir el missatge qui disposés de la clau de xifrat. Llavors es van trobar amb el problema dels *Generals Bizantins*, van tenir la necessitat de trobat un algorisme que asseguri la veracitat del missatge rebut.

Problema dels Generals Bizantins

Suposem que el cap d'un exercit dona l'ordre d'atacar l'enemic a 3 generals (per mig de missatgers), i la ordre que arriba als generals pot ser no la mateixa que va donar. Es pot donar el cas que un general canvies de bàndol i donés la ordre d'atacar. Tanmateix es pot donar el cas que el missatge que arribi hagi

estat modificat i fos diferent a la ordre donada. En resum, com es podia conèixer qui es el traïdor i quin era el missatge autèntic?

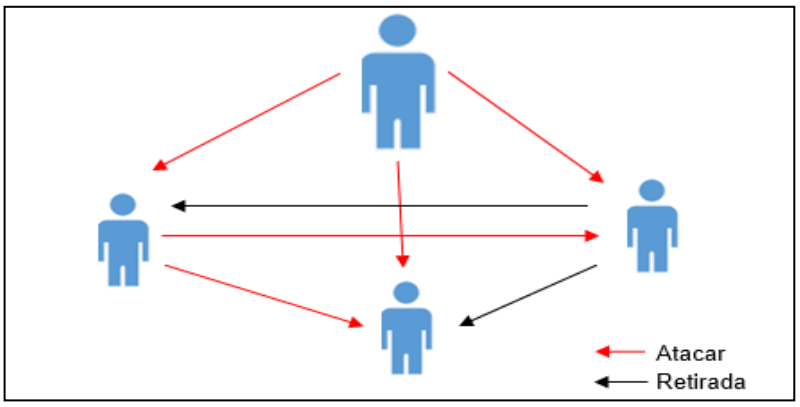


Figura 2.2. Problema generals bizantins

La resposta al problema dels *Generals Bizantins* passa per proporcionar un sistema que signi el missatge amb una signatura coneguda per tothom, i en augmentar el nombre de destinatari dels missatges. Si el missatge es oral i no signat, no hi ha manera de saber si el missatge es cert, de manera que tot depèn de la lleialtat del general que ha emès la ordre i de qui transportava la ordre. Si el missatge esta signat, augmentar el nombre de destinataris de la ordre i permetre l'enviament de missatges a tots els participants, afavoreix trobar els traïdors ja que havien signat ordres contradictòries.
Com a resposta al problema neix el concepte anomenat "*Proof of Work"* com a manera de verificar que el missatge rebut no ha estat modificat.

2.2. La xarxa

Una xarxa es formada per dispositius físics interconnectats per mig de cablejat. Entenem com a dispositius físics: Routers, switxos, cablejat i computadors. Podem veure que les xarxes es connecten amb altres per a formar xarxes més grans. El terme es conegut com a *Xarxes de Computadors*. *"Internet és el exemple més conegut en aquesta interconnexió de totes les xarxes*" [2].

Fem servir les xarxes per compartir informació, recursos, enviament de correu electrònic, xat, telefonia IP, xarxes socials, comerç electrònic, comunicacions amb organismes estatals, etc. Totes questes connexions son realitzades per mig d'aplicacions que gestionen la transmissió de dades. Tenim la necessitat de que algunes d'aquestes dades siguin transmeses de manera segura.

2.2.1. Model OSI

El model OSI vol explicar com s'estableix una comunicació entre 2 computadors. Tot comença quan un usuari envia dades a un altre usuari per mig de la xarxa, el missatge es descompost en paquets que s'envien per la xarxa. La mida més petita es denomina "paquet de dades". Els paquets de dades viatgen fins a destí per mig de l'encaminament.

El model OSI presenta un model en capes on cada capa té una funció molt especifica. La informació del programa es dividida en trossos de la mateixa mida i es encapsulada per cada capa. La encapsulació pren la funció de donar informació al paquet de dades de com arribar a destí.

El model OSI és teòric i es descomposta en 7 capes. El model TCP / IP pren les funcions del model OSI en les xarxes.

Nª	Nom
7	Aplicació
6	Presentació
5	Sessió
4	Transport
3	Xarxa
2	Enllaç
1	Físic

Figura 2.3. Model OSI

Capes del model OSI

1. *Física:* els seus protocols defineixen com "navegar" per el medi físic.
2. *Enllaç:* administra la transferència de les trames en un mitja comú.
3. *Xarxa:* encaminament. Defineix com arribar a destí.
4. *Transport*: defineix com segmentar, transferir i recompondre dades.
5. *Sessió*: administra les connexions entre les aplicacions.
6. *Presentació*: presenta una informació comprensible a la capa d'aplicació.
7. *Aplicació:* serveis i aplicacions usats per l'usuari.

2.2.2. Model TCP/IP

El model *TCP / IP* pren com a referencia el model teòric *OSI*, però les capes no es corresponen. Aquest model defineix 4 categories de funcions per a du a terme una comunicació entre dos participants. Es tracta d'un estàndard obert, és definit en els RFC i han estat documentats per part de IETF.

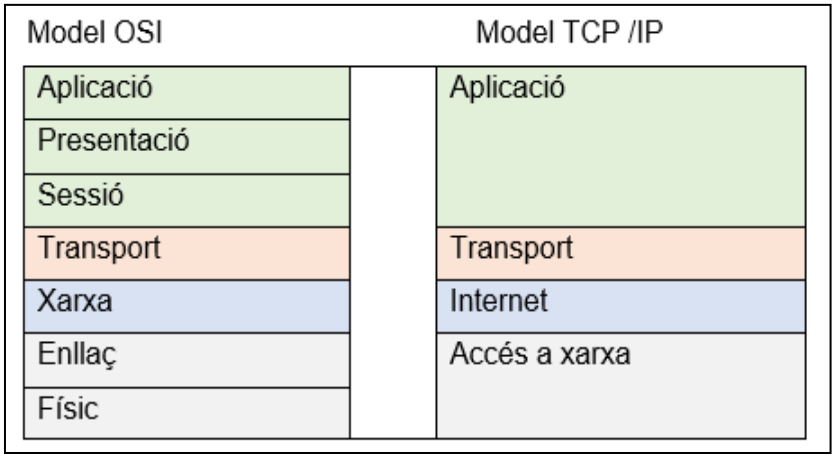


Figura 2.4. Comparativa models de capes

En una transmissió de dades, cada capa es "comunica" amb ella mateixa en el destí.

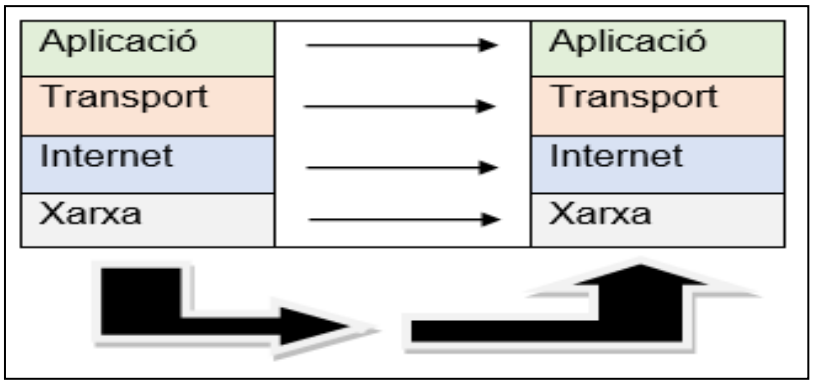


Figura 2.5. Comunicació entre capes. model TCP/IP

Capes del model TCP / IP

1. *Accés a xarxa*: característiques físiques del medi de comunicacions. Dreces MAC.
2. *Internet:* encaminament. Cerca la millor ruta a usar per arribar a destí. Dreces IP, encaminament.
3. *Transport:* Control d'errors, segmentació, control de flux, direccionalment d'aplicacions.
4. *Aplicació:* Aplicacions d'usuari i serveis de xarxa

2.2.3. Procés d'encapsulament

Tot comença quan un usuari selecciona una aplicació i envia dades cap un altre usuari (capa d'aplicació). La capa de transport s'encarrega de segmentar les dades i les encapsula amb les seves instruccions. El segment passa a la capa d'Internet on es encapsulat amb una capçalera de la capa de transport (paquets). Els paquets son encapsulats per la trama i llavors les trames ja poden accedir al medi [3].

En rebre el destinatari les trames, es dur a terme el camí invers. Cada capa recull la seva informació i entrega les dades a la capa superior. Al fi, les dades son recompostes al seu estat original i entregades al usuari.

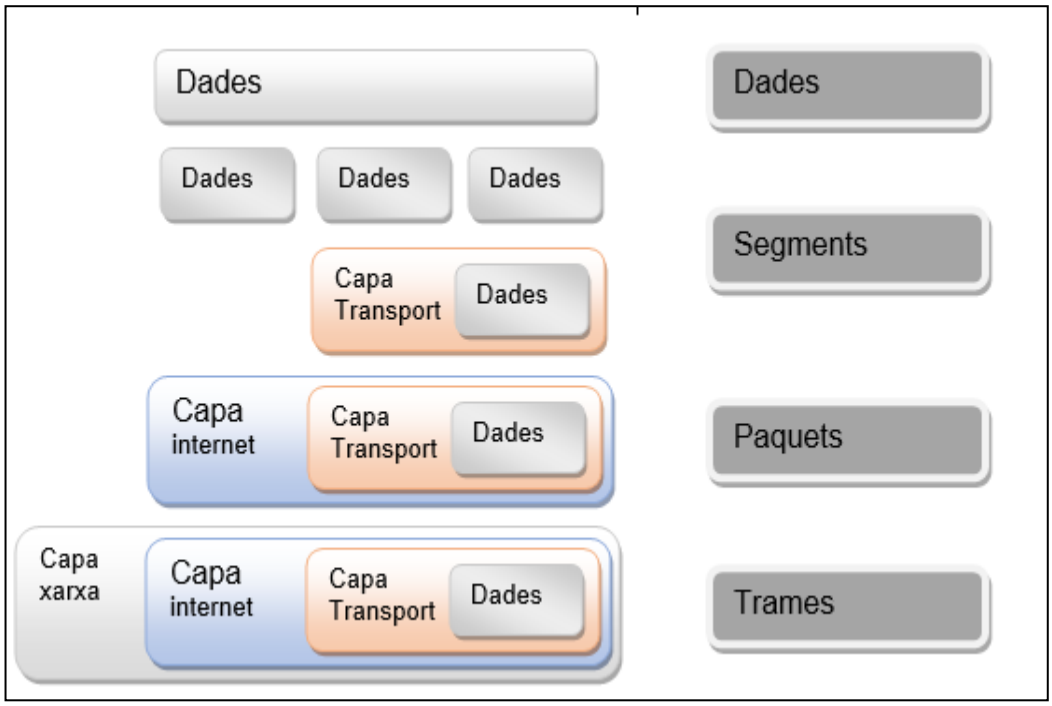


Figura 2.6. Procés d'encapsulament

2.2.4. El paquet de dades

Hem pogut veure com es forma una trama per mig del procés d'encapsulament. Cada capa és regida per un protocol específic.

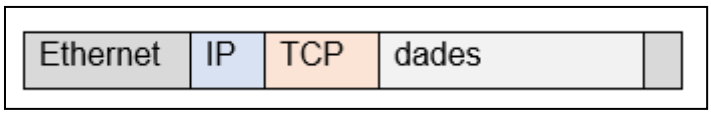


Figura 2.7. Paquet TCP/IP

2.2.5. Protocols de la pila TCP/IP

Aplicació: HTTP, HTTPS, FTP, FTPS, DNS, SMTP, POP3, IMAP, DHCP, SMB, P2P,VoIP, Telnet, TFTP, SSH.

Transport: TCP, UDP, SSH

Internet: IP, ARP, ICM

Xarxa: Ethernet, Token Ring, RS-232, FDDI

2.2.6. Seguretat

Veiem com el missatge es enviat per mig d'un medi no segur, i la pila de protocols TCP / IP per si sola no protegeix la informació. És precís encapsular específicament els paquets amb protocols que permetin una connexió segura tot fent ús del xifrat. Tanmateix, cal assegurar la veracitat de la informació enviada amb l'ús de certificats.

Les transmissions de dades segures es recolzen en altres protocols per garantir la seguretat en les mateixes. Els certificats son els encarregats de xifrar les connexions fetes mitjan SSL/TLS, SFTP, FTPS HTTPS, DNSSec entre altres.

2.3. La Criptografia

Neix com a necessitat de enviar informació secreta. *"Els missatges enviats per els canals de comunicació han estat xifrats a fi que evitar que algú pugui interceptar, manipular o falsificar les dades transmeses*."

"*La criptografia es la ciència que estudia la escriptura de secrets, amb l'objectiu de que el missatge quedi ocult*". Aquesta definició, extreta de la obra Principis de la Criptografia, deixa coberta en tota la seva extensió que és i que fa la Criptografia [4].

Aquesta, es val d'algorismes matemàtics per a xifrar la informació. *"La Criptografia estudia, des d'un punt de vista matemàtic, els mètodes per protegir la informació"* [5]. Els diferents mètodes criptogràfics es basen en problemes matemàtics coneguts, i son seleccionats aquells que ofereixen un tipus de complexitat intractable; es a dir, els computadors normals no disposen de suficient potencia de càlcul per a resoldre'ls.

En xifrar un missatge *M* el que fem es aplicar-li un algorisme de xifrat f que transforma el missatge en text xifrat C.

$$C = f(M)$$

Per desxifrar el missatge M ens cal conèixer la funció inversa f^{-1} a fi de poder recuperar el missatge original a partir del text xifrat.

$$M = f^{-1}(C)$$

Al llarg de la historia hem conegut de molts tipus de xifrats, i la clau radica en l'algorisme de xifratge. Alguns dels xifrats "famosos" son els següents:

- La xifra de Cèsar: basat en la substituir de lletres
- Xifrat de transposició: basat en canviar l'ordre dels caràcters de text.
- Xifrat de substitució: basat en substituir lletres altres lletres, o símbols.

Avui en dia ha canviat aquest paradigma, es treballa amb algorismes coneguts e amb una clau de xifratge k per a xifrar un missatge.

$$C = e(k, M)$$

Es a dir que es necessita aquesta clau k tant per xifrar el missatge com per desxifrar el missatge.

$$M = d(x, C) = d(x, e(k, M))$$

Aquest paradigma l'anomenem ***suposició de Kerckhoffs,*** i afirma que els algorismes han de ser coneguts per tothom i la seva seguretat sols ha de dependre de la clau.

Alguns d'aquests algorismes han creat diferents sistemes de xifrat.

- *RSA*. Algorisme que usa el problema matemàtic conegut com la factorització de nombres primers. No s'ha trobat la manera de trobar com factoritzar un nombre gran en els seus nombres primers, tot fent ús de computadors que tenim a l'abast.
- *ECC*. Basat en l'ús d'algorismes basats en corbes el·líptiques.
- *El gamal*. Basat en l'algorisme Diffie-Hellman. Basa la seva seguretat en la dificultat de calcular logaritmes discrets.

2.3.1 Tipus de criptografia

Criptografia de clau simètrica

Aquest sistema fa ús de una sola clau per xifrar i desxifrar el missatge. Es fa necessari que emissor i receptor comparteix la clau de xifrat.

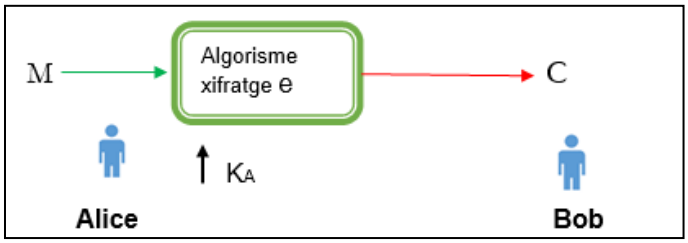


Figura 2.8. Procés de xifrat

Alice escriu un missatge M i el xifra amb una clau K_A per mig d'un algorisme de xifrat conegut e, i envia a Bob el missatge C xifrat.

$$C = e(KA, M)$$

En *Bob* té una clau K_B que és la mateixa clau que la de *Alice*, i tots dos fan ús del mateix algorisme de xifrat.

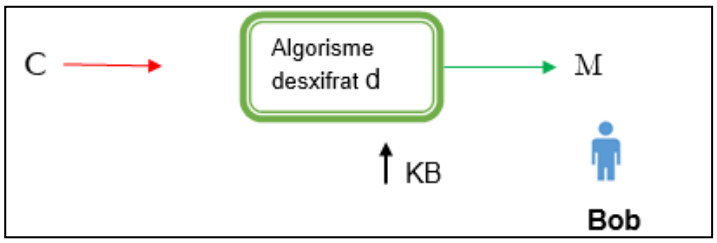


Figura 2.9. Procés de desxifrat

Bob rep el missatge xifrat C, i aplica l'algorisme de desxifrat d i la clau Kb per a obtenir el missatge original, ara ja llegible.

$$M = d(KB, C) = d(KB, e(KA, M))$$

Fa ús de dos tipus de tècniques per al xifrat de missatges; xifrat de flux i xifrat en bloc. La senzillesa del sistema fa que guanyi en velocitat, per contra, ens presenta el problema de com compartir la clau de xifrat de manera segura. És per això que es dona pas al sistema criptogràfic de clau publica, ja que resol aquest problema.

Criptografia de clau asimètrica o publica

El sistema es basa en la generació d'un parell de claus, una clau privada i una clau pública per a xifrar la informació. La clau pública es al abast de tothom, mentre que la clau privada sols és al abast del propietari de la clau.

Tipus d'algorismes de clau publica

- **"*Diffie y Hellman*** *proposen en 1976 un intercanvi de claus entre dos usuaris sense la intervenció d'aquests y sense que se hagi establert prèviament un canal segur*" [6]. Aquest protocol permet que es posin en contacte dos usuaris per mig d'un canal insegur, i es basa en la propietat de que qui accedeix pot obtenir la clau publica del destí. El emissor farà servir aquesta clau publica per enviar un missatge que sols pot ser llegit amb la clau privada del receptor.
    - Aquest protocol proporciona la creació d'un canal segur, però no verifica la autenticitat del emissor.
    - Algorisme basat en la dificultat de calcular logaritmes discrets.
    - Aquest algorisme du a terme el conegut com a encaixada de mans en rebre una sol·licitud d'accés per part d'un altre.

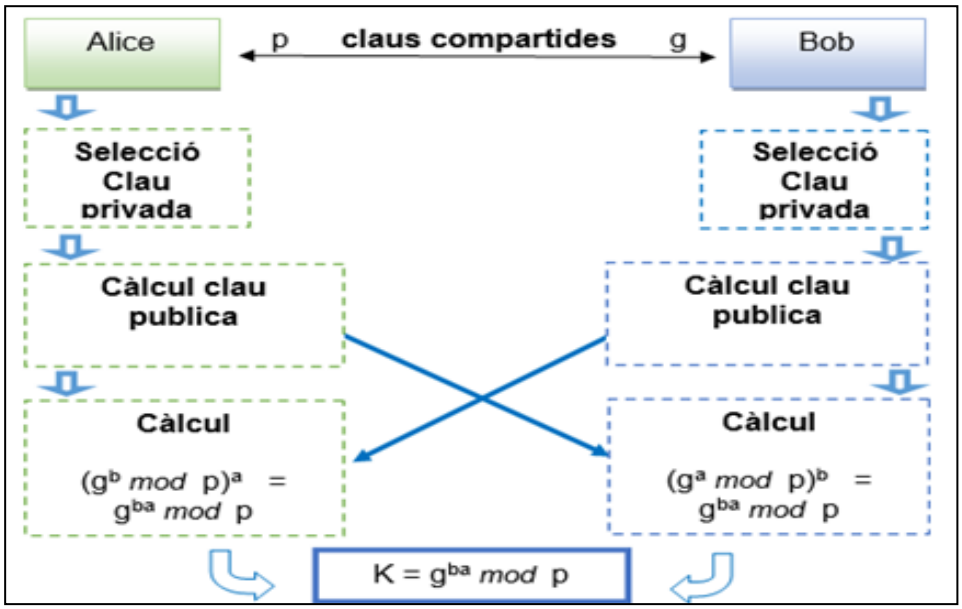


Figura 2.10. Algorisme Diffie-Hellman

- **RSA** es l'algorisme més usat i es fa servir com estàndard en tot tipus d'aplicacions per signar i xifrar informació. L'algorisme es basa en el problema de la factorització de nombres grans. Computacional ment

esdevé un problema intractable de resoldre per els computadors que usem de manera habitual.

L'algorisme pren el funcionament dels algorismes de clau publica. Per xifrar un missatge cal disposar com a mínim de la clau publica del receptor del missatge. Llavors se envia el missatge al propietari, i com és l'únic que disposa de la clau privada, serà l'únic que podrà llegir el missatge enviat.

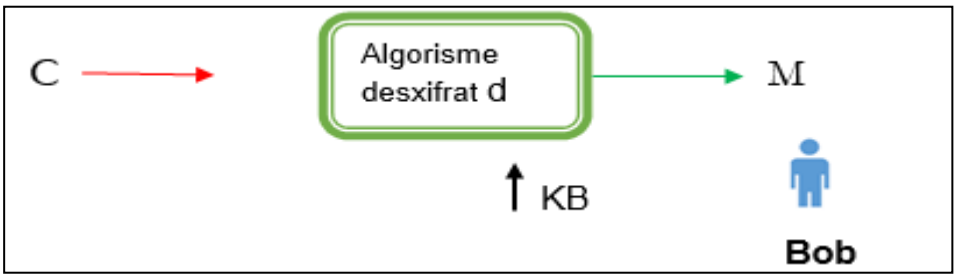


Figura 2.11. Procés xifrat / desxifrat

1. Alice envia a Bob la seva clau publica Ka
2. Bob envia a Alice un missatge xifrat c signat amb la clau publica de Alice. m es el missatge original. e es la clau publica de Alice.

$$c = me \pmod n$$

3. Alice rep el missatge c i el desxifra fent ús de la clau inversa

$$m = c^{Kca} \pmod n$$

Aquest sistema es basa en diversos factors, per exemple: que no es pot resoldre sense conèixer la clau, en guardar bé la clau privada i en la longitud de la clau d xifrat. La clau privada pot ser exposada, llavors deixarà de tenir validesa. Actualment es considera segur una clau de 2048 bits de longitud. A claus més grans, i missatges més grans disminueix la velocitat en xifrar / desxifrar; els computadors actuals realitzen aquesta tasca amb bastanta celeritat.

- **Corbes el·líptiques**. La criptografia de corba el·líptica (ECC) és una variant de la criptografia de clau publica basada en les matemàtiques de corbes el·líptiques [7]. L'intercanvi de claus es realitza igual que en el algorisme RSA, el que canvia es l'algorisme aplicat per xifrar i desxifrar el missatge.

Una corba el·líptica fa referencia a la equació $Y2 = X3 + a \cdot X + b$

Cal exigir que la corba no sigui de tipus singular, és a dir que les seves 2 parcials son iguals a 0. És condició suficient i necessària que el discriminant sigui diferent a 0. Per exemple: $4 \cdot a3 + 27 \cdot b2 \neq 0$. Una corba que tingui una singularitat, quan corba es creua amb ella mateixa no es una corba el·líptica.

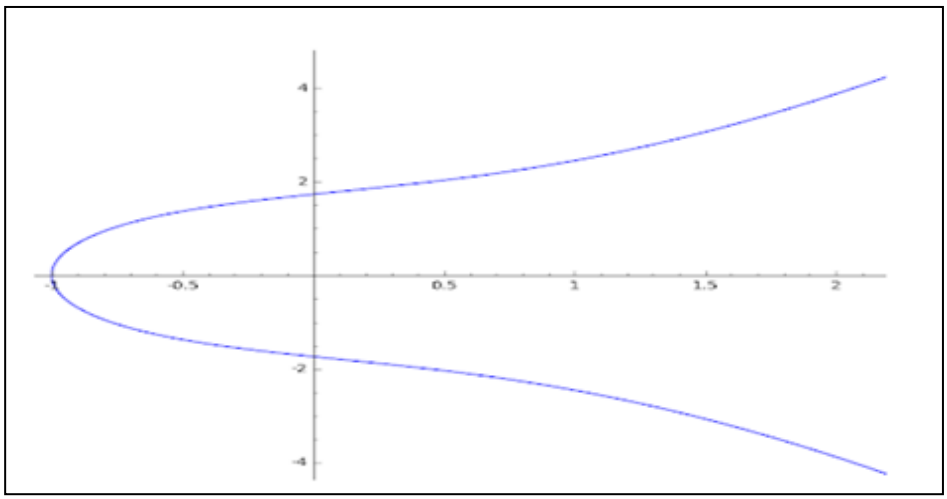


Figura 2.12. Corba el·líptica

Històricament

- Sorgeixen de manera implícita en el treball del matemàtic Grec *"Diofanto".*
- La aritmètica del segle XVIII posa en relleu el treball del matemàtic citat anteriorment.
- En els anys 50 es descobreixen connexions amb l'aritmètica l'anàlisi complex. Permetent la demostració del últim *teorema de Fermat*.
- Actualment han esdevingut indispensables en la teoria de nombres.

Font:

Introducción a las curvas elípticas - Bing video [8]

La corba el·líptica

Partim de la corba el·líptica $Y^2 = X^3 - 4 \cdot X + 1$

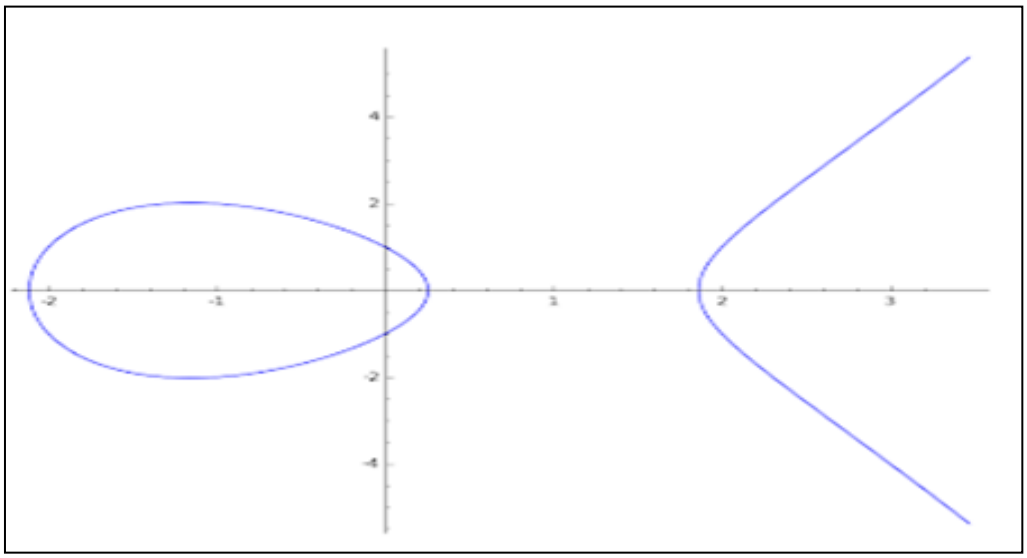


Figura 2.13. Exemple corba el·líptica

La geometria li dona una estructura de grup Abelià, cosa que ens permet triar 2 punts i operar. “SI tracem una recta que talli els punts P i Q, la recta tallarà exactament en un punt R.”

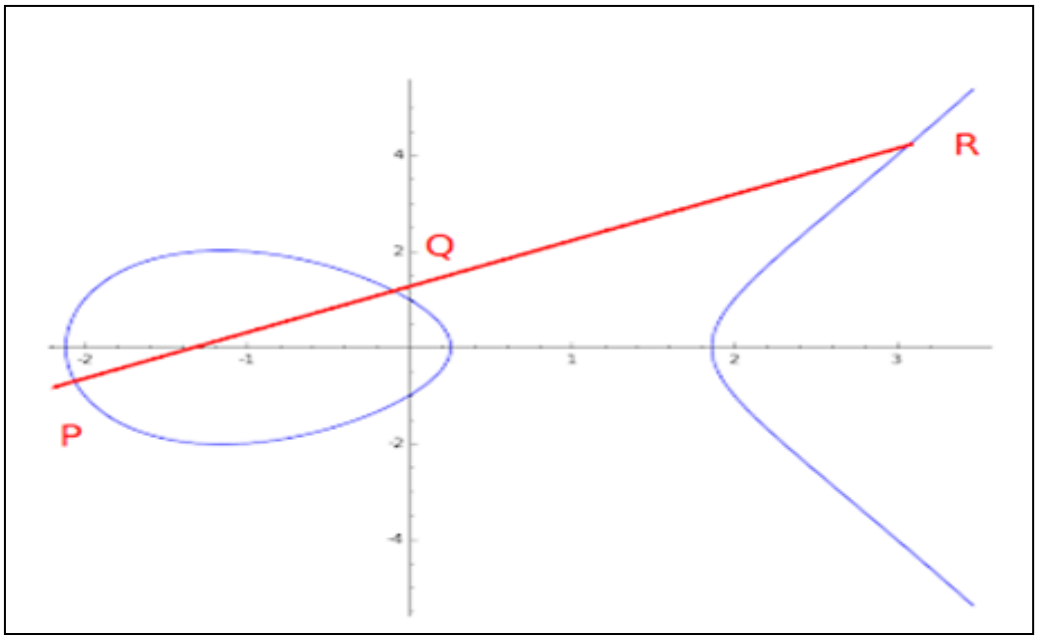


Figura 2.14. generació d’un punt

“Per definició, ens valdrem del valor simètric d’aquest punt R respecte al eix d’abscisses” [9] . Per convenció, pren el valor - R.

Ara ja podem definir la operació suma com l'invers del tercer punt de tall de la recta formada per els punts P i Q. Es a dir, P + Q = - R

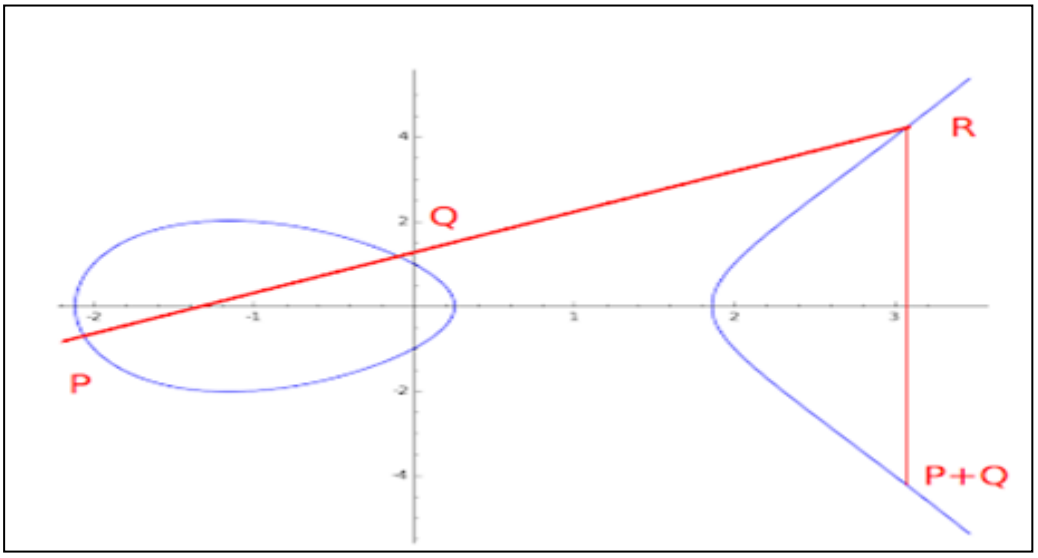


Figura 2.15. P + Q = -R

L'objectiu final de totes aquestes operacions es poder realitzar operacions matemàtiques a fi de poder construir sobre aquestes corbes els algorismes criptogràfics que ens interessin. El que hem fet es definir un grup algebraic sobre nombres R. Els algorismes triats volen buscar punts sobre nombres reals.

Grup abelià sobre nombres enters.

Aquest grup complirà les següents propietats:

- Suma. P + Q = - R || si -R = O => P = Q
- Element O. P + O = P
- 2P. Definim el producte 2 · P com la recta tangent a la corba que passa per el punt P. 2 P = - R

Partint d'aquesta premissa, la Criptografia de clau publica vol trobar un mecanisme que ens permeti trobar un punt del tipus kP, sent k un escalar (a, b).

2.4. Principis bàsics de seguretat en les xarxes

Una transmissió segura ha de garantir que les dades arribaran al destinatari sense que ningú hagi estat capaç de veure el missatge transmès, sense que ningú hagi pogut modificar les dades i sense que ningú pugui suplantar la identitat del emissor. Tota comunicació segura es basa en les propietats de *confidencialitat, autenticació, integritat i no repudi* [10].

Propietats

Confidencialitat. Propietat aplicada a un missatge que garanteix que ningú podrà veure les dades transmeses, a part del nostre receptor. Per tal de no poder accedir als missatges, aquests son xifrats amb un codi que sols coneix l'emissor i el receptor.

Autenticació. Propietat aplicada a un missatge que atorga la autoria del missatge al seu autor. En definitiva, vol garantir que ningú pugui suplantar la identitat del autor del missatge. Entren en joc autoritats de CA per a garantir la autenticitat del autor.

Integritat. Propietat que assegura que les dades no es puguin modificar sense que l'emissor o el receptor ho detectin. Les funcions de hash asseguren que el missatge no ha estat modificat. Qualsevol modificació feta farà canviar el hash del missatge.

No-repudi. Propietat que neix de les anteriors. Ningú es pot desdir d'un missatge rebut que compleix amb les 3 propietats anteriors, ja que aquestes, validen com a autèntic el missatge.

Altres objectius

Son objectius derivats de les propietats citades i son fets servir en aconseguir entregar un missatge segur: control d'accés, autenticació d'entitat, signatura, autorització, validació certificació, confirmació i revocació. Tots ells son necessaris per establir un entorn on els missatges enviats siguin transmesos de manera segura.

SSL estableix una comunicació xifrada entre 2 interlocutors que es comuniquin via https. Per si sol no és capaç de proporcionar autenticitat a un missatge, de manera que qualsevol transmissió xifrada podria "usurpar" la autoria del missatge.

El protocol TLS és l'hereu del protocol SSL. La capa TLS fa ús de certificats x.509. Fa us de criptografia asimètrica i xifra els missatges, aportant confidencialitat, integritat i autenticitat als missatges.
"El protocol TLS 1.2 ha estat definit en el RFC 6176 per la seva compatibilitat amb SSL i TLS, i per descartar l'ús de SSL 2.0". El web està començant a acceptar de manera progressiva TLS 1.3 per la seguretat que ofereix.

En la vida real les aplicacions fan us dels protocols segurs per establir connexions segures.

- **HTTPS.** Fa ús del protocol SSL / TLS. Versió 1.2 o 1.3. Port 443
- **FTPS**. FTP sobre SSL / TLS. Port 21, 990
- **SFTP**. Fa ús del protocol SSH. Port 22
- **DNS.** Varies alternatives. Es dona més importància a protegir els servidors DNS (integritat i seguretat) que no en oferir una connexió segura. Port 53

  Opcions
  - **DNSSEC.** Criptografia clau publica.
  - **DoT.** Dns Sobre TLS. Port 853. Molt usat per Google
  - **DoH.** DNS sobre HTTPS
  - **DNSCRypt.** OpenSoure. TCP 443
- **SMTPS.** SMTP sobre TLS. Port 587
- **IMAPS.** IMAPS sobre TLS. Port 993
- **POP3S.** POP3 sobre TLS. Port 995
- **VPN.** ÚS DE IPSec, port 500. PPTP, port 1723. L2TP, port 1701

3. Criptografia de clau publica

El sistema criptogràfic de clau pública es basa en la generació de dos claus; una privada i una pública. La clau privada es manté secreta, mentre que la pública es pot dir a tothom. No és possible obtenir la clau privada a partir de la clau pública. Un missatge xifrat amb la clau pública sols pot ser desxifrat amb la clau privada corresponent, i a l'inrevés.

El xifrat amb clau publica assegura la confidencialitat. Mentre que el xifrat amb clau privada (signatura digital) assegura confidencialitat i autenticitat.

3.1. Infraestructura

Ens cal una infraestructura que permeti que aquests actors puguin actuar en les xarxes obertes assegurant la identitat de la clau privada.

3.1.1 Autoritat de certificació

"Es tracta d'una entitat pública, reconeguda i acceptada a nivell mundial, que se encarrega de certificar el vincle entre un parell de claus i una identitat" [11].

En cas de que el *parell de claus* sigui creat per una d'aquestes entitats, el *certificat digital* vincula una *clau publica* a una identitat. Aquesta vinculació li atorga confiança al *certificat públic*, tant mateix, la *clau privada* és signada per la C.A., i la valida davant de tothom.

3.1.2 Autoritat de registre

Es tracta de la entitat encarregada de verificar l identitat.

3.1.3 Autoritat de validació / revocació

Es crea un llistat amb els certificats vàlids, i aquells que han estat revocats. Poden ser revocats per estar caducats, o per que se han vist compromesos.

3.1.4 Autoritat segellat de temps

Es la encarregada de crear marques de temps. Permeten demostrar que una dada existia en un moment concret.

3.1.5. Entitat final

Sol els titulars dels certificats. Apareixen al final de la cadena de certificació.

3.2. El certificat digital

La creació del parell de claus pot ser duta a terme de diferents maneres segons la necessitat i l'ús que se li vulgui donar.

3.2.1 Tipus de certificats

- Auto emesos. Permet xifrar, signar documents i crear connexions segures. Planteja el problema de guardar les claus, i de que tota connexió feta serà mostrada com que no es confia. El certificat sol ser creat per suites com *OpenSSL, OpenPGP*.

- El parell de claus es generat per la CA. Els certificats validen al usuari arreu del mon. Algunes del les entitats: *CAcert, VeriSign, Thawte*.
    - Certificat de clau publica de autoritat de certificació. Certificat emès a una entitat que és capaç de signar altres certificats.
    - Certificat de clau publica d'entitats finals. Certificat emès a una entitat final.
- Els certificats CA també poden ser
    - Auto emesos: Titular i emissor son el mateix.
    - Auto signats: La clau privada signa el certificat públic.
- El parell de claus es creat per un dispositiu de hardware especialitzat. Les claus privades queden emmagatzemades en el dispositiu. *DNI 3.0*

3.2.2. Ús de certificats

"Un certificat digital creat per una autoritat de certificació reconeguda vincula al subscriptor amb unes dades de verificació de signatura i en confirma la seva identitat" [12]. Aquesta afirmació esdevé la base del Comerç electrònic i de la presentació de documentació a entitats governamentals. De fet, a nivell global podem afirmar que l'enviament de dades xifrades el sustenten els certificats, sense ells no seria possible la vida tal com la concebem actualment.

Un certificat és usat tant per enviar missatges xifrats com per crear connexions xifrades. Aquest, ens valida de manera inequívoca cap als altres amb la garantia de que ningú pot dubtar de la *autenticitat* del nostre missatge. Així mateix, qui el rep no es pot desdir de haver rebut aquest missatge de nosaltres.

Tot missatge s'envia xifrat, el fet que sols qui disposi d'alguna de les claus pugui llegir el missatge en garanteix la *integritat* d'aquest.

3.3. El estàndard x.509

El estàndard defineix un entorn de treball per als certificats de clau pública, així com la validació del certificat. Estàndard creat per el Sector de Normalització de les Comunicacions (UIT-T) [13].

3.3.1. Certificats

Una CA emet un certificat, associant una clau publica a un nom distingit (DN) o un alternatiu (email).

3.3.2. Estructura del certificat

Certificat

- Versió
- Num. de sèrie del certificat
- ID del algorisme usat. (RSA o DSA)
- Emissor
- Validesa
- Subjecte: DN(Nom distingit), CN(Nom comú), OU(Unitat organitzacional), O(Organització, C(País)
- Informació de clau publica
- Algorisme
- Clau publica
- Id. Emissor
- Id Subjecte

3.3.3. Classes de certificat

- Classe 1. Adjunta nom i dreça
- Classe 2. La CA comprova el DNI o el permís de conduir
- Classe 3. Verificació del crèdit disposat
- Classe 4. Verificació el càrrec que ocupa en una organització

3.3.4. Finalitat

- Certificats SSL de tipus client. Dirigit a una persona física
- Certificat SSL de tipus servidor. Identifica un servidor
- Certificat S/MIME. Es fa servir en serveis de correu electrònic.
- Certificat per a signar codi.
- Certificat AC. Verifica si un certificat es de confiança

3.4. Certificats RSA

Aquest sistema de criptografia de clau pública és basat en la factorització dels nombres primers. Actualment és el sistema mes usat a nivell mundial. Diferents

organitzacions de seguretat han definit unes longitud de clau considerades segures, i han esdevingut el estàndard actual en les connexions xifrades actuals. Els xifrats amb claus de 2048 i 3072 bits.

L'algorisme RSA

Aquest algorisme es dissenyat per escollir dos nombres primers, p i q de manera que sigui inviable de trobar-los per mig de computadors convencionals. La fortalesa de l'algorisme es basa en dos factors, en primer lloc la dificultat que tenim d'obtenir el missatge "M" sense conèixer l'exponent "e". En segon lloc tenim la dificultat d'obtenir "d" a partir de "n" [14].

1. Obtenció de n
   n es resultat de $p \cdot q$
2. Es selecciona e
   $1 < e < (p - 1) \cdot (q – 1)$ MCD $(e, (p - 1) \cdot (q – 1)) = 1$
3. Es calcula l'invers de e i l'anomenem d
   $e \cdot d = 1 \pmod{(p - 1) \cdot (q – 1)}$

Generació de claus

La clau publica està formada per un nombre n, calculat com a producte de dos nombres primers molt grans ($n = p \cdot q$), i un exponent e. Formada per (e, n)
La *clau privada* es un altre exponent que anomenarem "d", calculat a partir de p, q i e. Formada per (d, n) [15].

Xifrat $C = M^e \bmod n$

Desxifrat $M = C^d \bmod n$

3.5. Certificats ECDSA

Aquest sistema de criptografia de clau publica es basa en l'ús d'operacions amb corbes el·líptiques. Actualment està sent adoptat de manera nativa per moltes aplicacions; navegadors web, aplicacions FTP, SAP PI, SSL/TLS, Amazon Web Services, etc deguts als avantatges que ofereix en fer servir claus de una longitud inferior per a proporcionar el mateix grau de seguretat que altres sistemes de xifrat [16].

Avantatges

Eficiència i seguretat en son els paràmetres amb els que aquest tipus de certificat posa de relleu els avantatges que presenta sobre el xifrat RSA.

Seguretat

- RSA necessita com a mínim claus de 2048 bits, mentre que ECDSA en te prou amb 112 bits.
- Capacitat d'usar claus amb mes seguretat sense afectar al rendiment.

Claus RSA	Claus ECDSA	Seguretat
1024 bits	160 - 223 bits	80 bits
2048 bits	224 - 255 bits	112 bits
3072 bits	256 - 383 bits	128 bits
7680 bits	384 - 511 bits	192 bits
15360 bits	512 bits	256 bits

Taula 3.1. Comparativa longitud claus

Eficiència.

- RSA necessita de més capacitat de computació per a xifrar i desxifrar en comparativa amb ECDSA.
- Ús de claus mes petites per oferir la mateixa seguretat.
- Ús de claus més petites necessita de menys memòria.
- Mesurada com complexitat de temps en xifrar, signar.
- En funció de la mida d'entrada.
- Carrega més rapida dels llocs web.

Compatibilitat

- Acceptat de manera nativa per la majoria de navegadors web.
- Acceptat de manera nativa per Amazon Web Services.
- Útil en entorns on la potencia de procés es reduïda.

Implantació

- Definit en el document ANS X9.62.2005, Public Key Cryptography for the financial Services Industry (ECDSA).
- Definit en el document FIPS 186 – 5.

- Especificat com *algorisme de signatura digital* en estàndard de suite B de la NSA.

Generació de claus

1. Tria de la corba el·líptica E.
2. Selecció d'un punt P d'ordre n.
3. Selecció aleatòria d'un numero d en l'interval (1, n – 1).
4. Càlcul Q = dP.
5. d serà la clau privada.
6. Q serà la clau publica.

Procés de signatura

1. Seleccionar un nombre k de manera aleatòria.
2. Calcular kP = (x_1, y_1).
3. Calcular r = x_1 mod n. Si r = 0 torna al pas 1.
4. Calcular (k-1) mod n.
5. Calcular s = k -1 (H (m) +dr) mod n. H (m) es el Hash del missatge a signar. Us de SHA -1.
6. La signatura es composta per (r , s).

Procés de verificació

1. Verificar que r y s pertanyen al rang [1, n - 1].
2. Calcular w = (s – 1) mod n.
3. Calcular u_1 = H(m) w mod n.
4. Calcular u_2 = r · w mod n.
5. Calcular u_1P + u2Q = (x0,y0).
6. Calcula v = x_0 mod n.
7. La firma verifica si y solo si v = r.

4. SSL/TLS

4.1. El protocol SSL

SSL es un protocol criptogràfic de la capa de transport. Aquest protocol crea una *connexió* per a transmetre dades. Per proporcionar aquesta seguretat es val del us de certificats x.509. Proporciona a la connexió *autenticitat i privacitat.*

SSL implica les següents fases

- Negociació de l'algorisme a usar
- Intercanvi de claus públiques
- Xifrat del tràfic

4.2. El protocol TLS

Aquest protocol neix com una actualització de SSL 3.0. Pren la seva funcionalitat i la millora aportant les qualitats d'autenticació, confidencialitat i integritat [17]. Estàndard definit per la IETF i la RFC

Avui en dia l'estàndard es la versió 1.2, però la versió 1.3 s'està adoptant de manera nativa el molts llocs per oferir més seguretat. Les organitzacions de seguretat recomanen desactivar versions inferiors.

TLS proporciona

- *Confidencialitat.* Algorismes de xifrat simètric. Claus generades a partir d'un secret compartit durant l'intercanvi de claus.
- *Integritat.* Ús de funcions de *Hash.*
- *Autenticació. Certificats de clau publica x.509.*
- Fordward Secrecy. Si la clau és exposada deixarà de ser valida .
- Control de sessió.

Versions

Versió	RFC	Data	Actualitzat
TLS 1.0	RFC 2246	01/1999	02/2013
TLS 1.1	RFC 4346	04/2006	01/2020
TLS 1.2	RFC 5246	08/2008	01/2020
TLS 1.3	RFC 8446	08/2018	03/2020

Taula 4.1. Versions TLS

4.3. Versions del protocol TLS

TLS 1.0 i 1.1. Versió obsoleta.

TLS 1.2. Ús del protocol SHA-2 per proporcionar autenticitat. Incorpora mètode de xifrat AEAD.

TLS 1.3. Millores criptogràfiques. Redisseny del procés de Handshake.

Les organitzacions de seguretat recomanen desactivar versions inferiors a TLSv1.2.

4.4. Aplicacions que fan ús de TLS

HTTPS. Conegut com a HTTP sobre TLS. Usat per navegadors i servidors web. El servidor disposa un parell de claus i un certificat que associa la seva identitat amb la seva clau publica. Per establir una connexió se estableix un guió de com fer la connexió: acord, creació de la clau i connexió. Aquesta protecció ha de permetre al client assegurar-se que s'havia connectat a un servidor autèntic, i poder enviar-li dades per mig d'un canal segur. Port 443.

FTPS. Parlem de FTP sobre TLS. Es un protocol d'intercanvi de fitxers. TLS proporciona seguretat, xifrat i autenticació. Port 21 o 990.

Tipus

Implícit. Un client inicia la connexió sobre el port TCP 990 i negocia com establir una connexió segura.

Explícit. El client es connecta al port TCP 21 i el servidor no negocia sinó que requereix establir una connexió segura.

NNTPS. Protocol segur usat en la transferència de noticies de xarxa. "News". Fa ús del port 563 quan NTPS es connecta al servidor de noticies mitjan SSL.

SMTPS. Es una manera de protegir SMTP aplicant la seguretat de la capa de transport. Usat per l'enviament de correu electrònic. Ús del port 587 per a connexions xifrades.

POP3, IMAP sobre SSL/TLS. Els protocols de correu son encapsulats per la capa de transport TLS. Proporcionen poder connectar-se als servidors de correu d'una manera segura. Ports 995 i 993.

4.5. Handshake

La encaixada de mans s'estableix quan un client vol iniciar una conversa SS/TLS amb un altre participant. "*El protocol Handshake s'encarrega de proporcionar autenticació entre els participants, negocia claus de xifrat, claus compartides i modus de proporcionar integritat a les dades."* [18].

- Proporciona autenticació per mig de claus RSA, ECDSA o PSK.
- Proporciona confidencialitat. Les dades sols son visibles en ser desxifrades.
- Proporciona integritat. Les dades no poden ser modificades sens que sigui detectat el canvi.

4.5.1 Diferencies versions TLSv1.2 / TLSv1.3

- TLSv1.3 prescindeix d'algorismes de xifrat simètric.
- Canvia el concepte de cipher suite. Separació dels mecanismes d'autenticació i intercanvi de claus del de xifrat i del de HASH.
- TLSv1.3 elimina xifrat RSA estàtic i Diffie-Hellman.
- Tots els missatges de rebut son xifrats.
- Redisseny del procés, l'enviament del certificat es du a terme en la fase de Server Hello.
- Els algorismes de corba el·líptica passen a ser natius.

4.5.2. Comparativa Handshake versions TLS

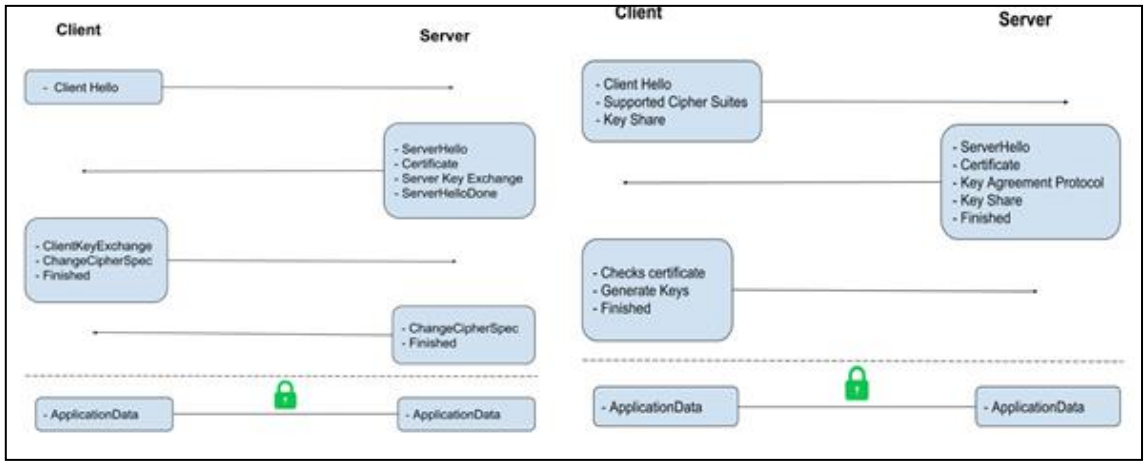


Figura 4.1. Comparativa Versions Handshake TLS

4.5.3 Procés Handshake

El client vol establir una connexió TCP amb el servidor (a) y envia una petició d'accés. El servidor respon (b); ofereix la verificació de la seva identitat amb l'enviament de la clau publica del servidor. En una tercers fase (c), el client servidor una clau secreta que tots dos participants farà servir cada cop que necessitin establir una sessió SSL.

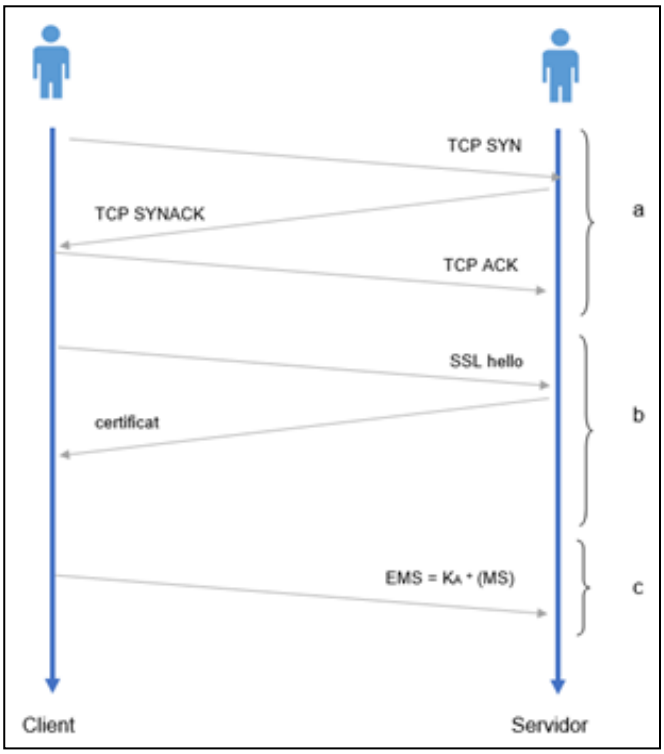


Figura 4.2. Procés Handshake TLS

El client vol establir una connexió TCP amb el servidor y envia una petició d'accés.

Fase a

1. El client envia una petició d'accés, junt amb un llistat d'algorismes criptogràfica suportats i la Id del client.
2. El servidor rep la petició i selecciona un algorisme simètric (AES, 3DES), un de clau publica (RSA, ECDSA) i un algorisme MAC. Respon amb missatge de confirmació informat al client que fer servir per establir la connexió.
3. El client confirma la recepció, verifica la clau i confirma la autenticitat del servidor enviant un missatge ACK de recepció de la informació

El client estableix una connexió segura per acordar la clau mestre a fer servir.

Fase b

1. El client inicia una connexió segura creada a partir de la clau publica del servidor.
2. El servidor rep la petició i envia el certificat.

El client estableix una connexió segura

Fase c

1. El client envia de manera segura una pre-clau, creada a partir de la clau publica del servidor.
2. El servidor rep aquesta pre-clau i en calcula una clau mestre. Segons l'algorisme usat, tots dos participants hauran creat la mateixa clau mestre. A partir d'ara tots dos faran us d'aquesta clau compartida per enviar-se informació per mig d'un canal segur.

Transferència de dades

En un primer moment, es farà ús de la clau compartida per crear les comunicacions entre els dos participants.

Primer es crea el canal amb la clau mestre. Sessió.

En segon lloc s'envien les dades xifrades.

Tancament del canal

El canal es tanca en tancar l'aplicació. Aquest protocol fa ús de criptografia de clau publica per a xifrar les connexions. En aquest cas hi tenim dos actors; el servidor web i el client podem apreciar com el protocol permet crear una connexió segura amb.

5. SSH

SSH és una aplicació que permet establir sessions segures amb un servidor remot. L'aplicació defineix el seu protocol, és a dir, es aplicació i protocol la vegada [19].

El protocol proporciona mecanismes de *confidencialitat, integritat i autenticitat* davant del usuari. L'aplicació xifra totes les connexions. Tanmateix, el protocol pot ser utilitzat per part d'altres aplicacions, ja que permet la encapsulació de connexions TCP a qualsevol port.

5.1. Característiques

- SSH proporciona un servei equivalent al del protocol SSL/TLS en quant a xifrat., amb la diferencia que la connexió segura permet l'administració remota del equipo on ens connectem.

Xifrat. El protocol xifra totes les connexions. Durant la connexió, el protocol negocia la clau a fer servir. Un cop s'ha establert la connexió el paquet SSL és xifrat amb aquesta clau.

Paquet de dades

4	1	L_m	L_p	L_{mac}
Longitud	Lp	missatge	padding	MAC

Llegenda

Longitud. Defineix Longitud del paquet. Bytes. No inclou la Lmac.

Lp. Defineix bytes de padding.

Lm. Defineix la longitud del missatge

Padding. Bytes aleatoris de padding

MAC. Missatge d'autenticació. Negociat per l'algorisme.

Algorismes de xifrat

3des-cbc	REQUIRED	three-key 3DES in CBC mode
blowfish-cbc	OPTIONAL	Blowfish in CBC mode
twofish256-cbc	OPTIONAL	Twofish in CBC mode, with a 256-bit key
twofish-cbc	OPTIONAL	alias for "twofish256-cbc" (this is being retained for historical reasons)
twofish192-cbc	OPTIONAL	Twofish with a 192-bit key
twofish128-cbc	OPTIONAL	Twofish with a 128-bit key
aes256-cbc	OPTIONAL	AES in CBC mode, with a 256-bit key
aes192-cbc	OPTIONAL	AES with a 192-bit key
aes128-cbc	RECOMMENDED	AES with a 128-bit key
serpent256-cbc	OPTIONAL	Serpent in CBC mode, with a 256-bit key
serpent192-cbc	OPTIONAL	Serpent with a 192-bit key
serpent128-cbc	OPTIONAL	Serpent with a 128-bit key
arcfour	OPTIONAL	the ARCFOUR stream cipher with a 128-bit key
idea-cbc	OPTIONAL	IDEA in CBC mode
cast128-cbc	OPTIONAL	CAST-128 in CBC mode
none	OPTIONAL	no encryption; NOT RECOMMENDED

Figura 5.1. Algorismes xifrat

Compressió

En cas de haver estat definida , el camp *payload* serà objecte de ser comprimit.

Integritat

La integritat de les dades es aconseguida inserint en cada paquet la MAC. La MAC es formada per un secret compartit creat en l'intercanvi de claus, el numero de paquet i el contingut del paquet. Aquest camp no es presenta xifrat, però si que ha estat signat per algun dels algorismes de autenticació usats.

The following MAC algorithms are currently defined:

hmac-sha1	REQUIRED	HMAC-SHA1 (digest length = key length = 20)
hmac-sha1-96	RECOMMENDED	first 96 bits of HMAC-SHA1 (digest length = 12, key length = 20)
hmac-md5	OPTIONAL	HMAC-MD5 (digest length = key length = 16)
hmac-md5-96	OPTIONAL	first 96 bits of HMAC-MD5 (digest length = 12, key length = 16)
none	OPTIONAL	no MAC; NOT RECOMMENDED

The "hmac-*" algorithms are described in [RFC2104]. The "*-n" MACs use only the first n bits of the resulting value.

Algorismes de Clau pública

Aquest protocol opera amb la majoria d'algorismes de clau publica coneguts. Defineix el format de la clau, algorismes de xifrat i codificació de les signatures.

Format de la clau

```
ssh-dss      REQUIRED    sign Raw DSS Key
ssh-rsa      RECOMMENDED sign Raw RSA Key
pgp-sign-rsa OPTIONAL    sign OpenPGP
certificates(RSA key)
pgp-sign-dss OPTIONAL    sign OpenPGP
certificates(DSS key)
```

Algorisme de xifrar

Defineix si les Claus seran fetes servir per signar, xifrar o les dos coses.

Codificació

La signatura és codificada pe el certificat o la clau publica i el nombre de bytes de la clau.

Intercanvi de Claus

Aquest mètode especifica com son creades les claus per a obtenir xifrat i autenticació, i com es du a terme la autenticació del servidor

L'intercanvi de Claus s'inicia quan cada participant presenta ell llistat de algorismes suportades. Es farà servir els algorismes preferits que siguin suportats per les dos parts.

Es fa ús de l'algorisme Diffie-Hellman.

Autenticació d'usuari

Un cop els dos participants de la connexió disposen de la clau compartida, l'usuari que vol accedir ha d'autenticar les seves credencials d'accés. Es necessita el nom d'usuari, el port i el servidor.

Connexió

Un cop validat les credencials del usuari se estableix una connexió segura, de tipus túnel, amb el servidor. Finalitzarà quan el client decideixi tancar la connexió.

- Fa ús del port 22. Posteriorment la connexió es redirigeix al port usat per l'aplicació. El client ha d'iniciar una sessió TCP apuntant al port 22. Un cop establerta la sessió, el protocol s'encarrega d'entregar els paquets a la aplicació de destí que li pertoqui.

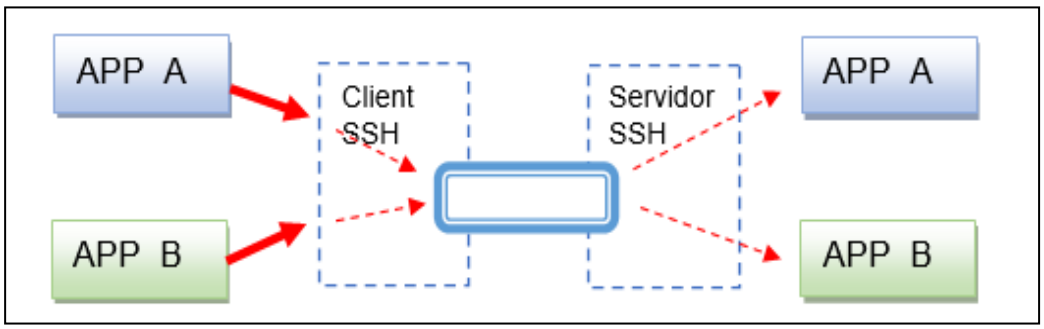
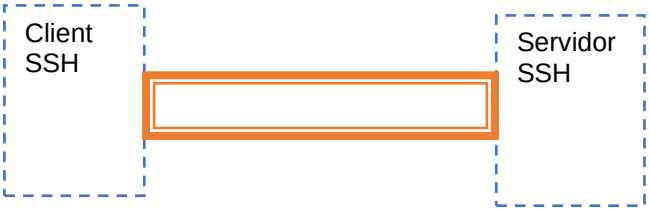


Figura 5.2. Connexió túnel SSH

- Estableix una connexió de tipus client – servidor per on passen totes les connexions. Cal establir una sessió punt a punt abans d'enviar dades.

- Estableix una connexió de tipus túnel



5.2. La capa de transport SSH

En SSH queden ben diferenciades dos capes; la capa d'aplicació i la capa SSH. Aquesta realitza diferents funcions, algunes depenen d'altres de manera que podem presentar la capa SSH subdividida en tres capes, que cadascuna depèn del seu nivell superior.

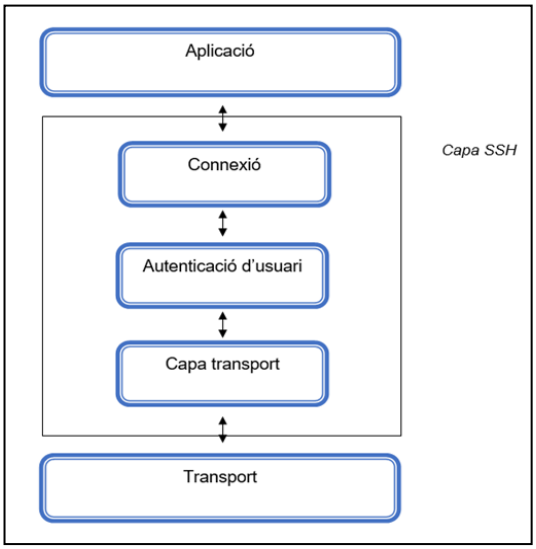


Figura 5.3. Capa transport SSH

5.3. El protocol de transport SSH

Aquest protocol es disposat en tres capes; capa de connexió, capa d'autenticació d'usuari i capa de transport. També pren la funció de crear els paquets SSH i gestionar el transport [19].

El protocol de connexió

- Gestió de les sessions entre client / servidor
- Funciona en dos sentits.
- Redirecció de ports TCP

El protocol d'autenticació d'usuari

- Anònim. Sense cap comprovació
- Amb contrasenya

Amb llistes d'accés

- Llistes d'accés amb autenticació de client
- Amb clau publica

El protocol de la capa de transport SSH

- Estableix la connexió
- Autentica al servidor
- Intercanvi de claus
- Petició de serveis a altres protocols

5.4. Aplicacions SSH

SSL pot ser instal·lat en equips servidor o client. El podem fer servir com a protocol d'una aplicació (sftp) o si fem ús de l'aplicació podrem accedir de manera remota a un equip servidor SSH i controlar-lo.

Algunes de pes aplicacions que fan us de SSH son:

- PuTTY
- KITTY
- WinSCP
- SmartFTP
- Security
- FileZilla
- Secure Shell (W10)

6. Rendiment HTTPS

En aquesta part del estudi volem comprovar el rendiment dels navegadors web respecte al tipus de clau triada per establir una connexió segura. Tanmateix, posar de rellevància la importància que presenta la compatibilitat dels navegadors vers la suite de xifrat oferta en la fase de Handshake, i per acabar, presentar i comparar les dades obtingudes del estudi.

6.1. Descripció

En primer lloc volem posar en relleu els navegadors i les cipher suites suportades. Seguidament passarem a analitzar el procés d'encaixada de mans segons la versió de TLS seleccionada en la fase de Handshake; podrem analitzar quina cipher suite queda seleccionada i , per acabar, coneixerem el temps que ha trigat cada navegador en carregar el web, en base d'això n'establirem una comparativa i presentarem les dades obtingudes.

6.1.1. Especificacions navegadors web

En la fase de prova dels certificats creats es va posar de relleu que alguns dels certificats creats amb *OpenSSL* no eren compatibles amb els navegadors web, anem a ser específics; el tipus de certificat i la longitud de la clau no era suportada per la *suite de xifrat* oferta per el navegador durant la fase de

Handshake. A conseqüència d'això no és possible l'accés al web per fer ús d'un protocol no admès.

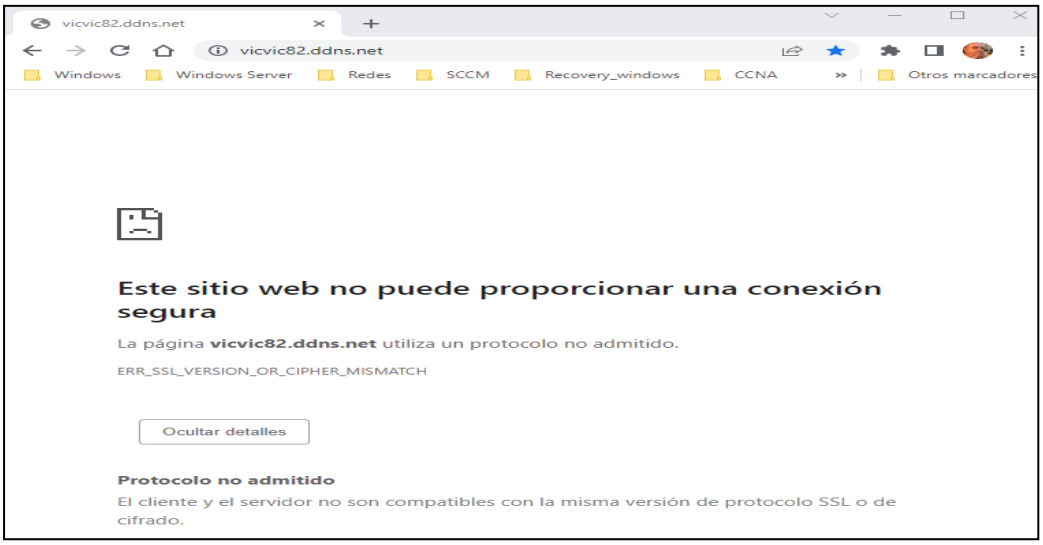


Figura 6.1. Clau incompatible amb suite de xifrat.

L'administrador web, en la fase de definir el grau de seguretat del web, ha de realitzar una tasca de verificació; tant per els tipus de certificats que accepten els navegadors com per la longitud de la clau acceptada. En aquesta fase els interlocutors son les suites de xifrat ofertes per els navegadors web i el tipus de claus usades per la creació del certificat de servidor. La premissa a validar es crear un certificat amb un tipus i una longitud de clau compatible per els navegadors web que ens aporti seguretat i un rendiment acceptable.

En termes generals els certificats RSA son compatibles amb tots els navegadors, la problemàtica la presenten els certificats de tipus ECDSA. Hi ha corbes que no son compatibles amb les suites de xifrat que presenta el navegador, arrel d'això, si el certificat instal·lat en el servidor web no és compatible amb les cipher suites.

D'altra banda també entra en joc per la elecció de la cipher suite, la versió de TLS seleccionada i les cipher suites ofertes OpenSSL. A continuació presentem les cipher suites compatibles amb els navegadors. Informació obtinguda de la fase de Handshake.

Suite de xifrat	Chrome	Firefox	Edge
TLS_AES_256_GCM_SHA384	SI	SI	SI
TLS_CHACHA20_POLY1305_SHA256	SI	SI	SI
TLS_AES_128_GCM_SHA256	SI	SI	SI
ECDHE-ECDSA-AES256-GCM-SHA384	SI	SI	SI
ECDHE-RSA-AES256-GCM-SHA384	SI	SI	SI
DHE-RSA-AES256-GCM-SHA384	X	X	X
ECDHE-ECDSA-CHACHA20-POLY1305-SHA256	SI	SI	SI
ECDHE-RSA-CHACHA20-POLY1305-SHA256	SI	SI	SI
ECDHE-ECDSA-AES128-GCM-SHA256	SI	SI	SI
ECDHE-RSA-AES128-GCM-SHA256	SI	SI	SI
DHE-RSA-AES128-GCM-SHA256	X	X	X
ECDHE-ECDSA-AES256-SHA384	SI	X	X
ECDHE-RSA-AES256-SHA384	SI	X	X
DHE-RSA-AES256-SHA256	X	X	X
ECDHE-ECDSA-AES128-SHA256	X	X	X
ECDHE-RSA-AES128-SHA256	X	X	X
DHE-RSA-AES128-SHA256	X	X	X
ECDHE-RSA-AES256-SHA	X	X	X
DHE-RSA-AES256-SHA	SI	SI	SI
ECDHE-ECDSA-AES128-SHA	SI	SI	SI
ECDHE-RSA-AES128-SHA	SI	SI	SI
DHE-RSA-AES128-SHA	SI	SI	SI
ECDHE-RSA-AES128-SHA	X	X	X
AES256-SHA256	X	X	X
AES128-SHA256	X	X	X
AES256-SHA	X	X	X
AES128-SHA	X	X	X

Taula 6.1. Cipher suites compatibles per els navegadors web

Google Chrome

Mostrem a continuació les Cipher Suites ofertes per el navegador *Chrome* en establir una connexió *Https* amb una url. En la fase de *Handshake,* el client envia un missatge de *HELLO* i ofereix 16 *Cipher Suites [20].*

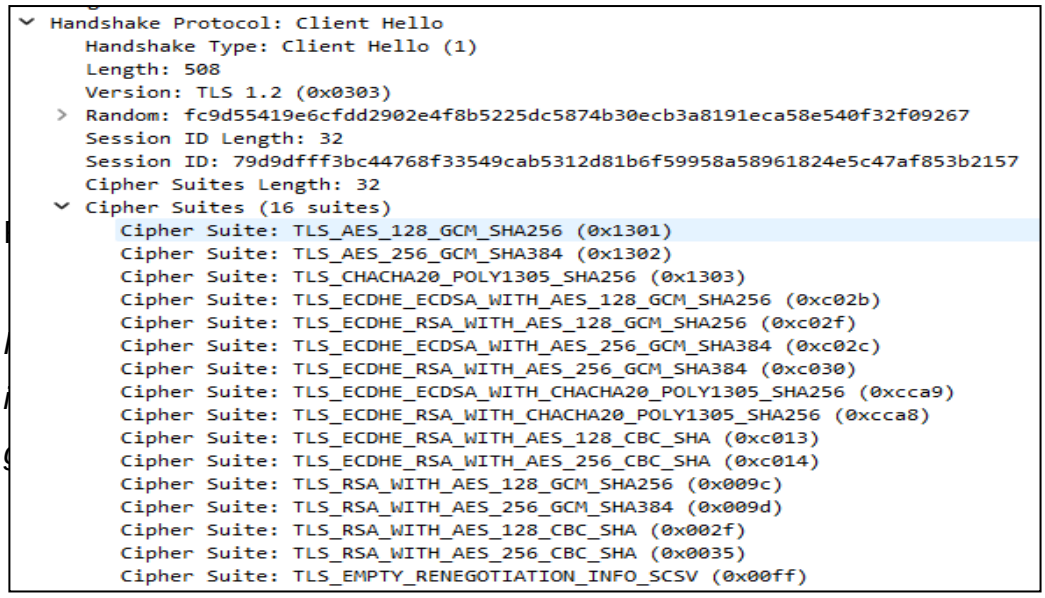


Figura 6.3. Cipher suites ofertes per el navegador Google Chrome

Microsoft Edge

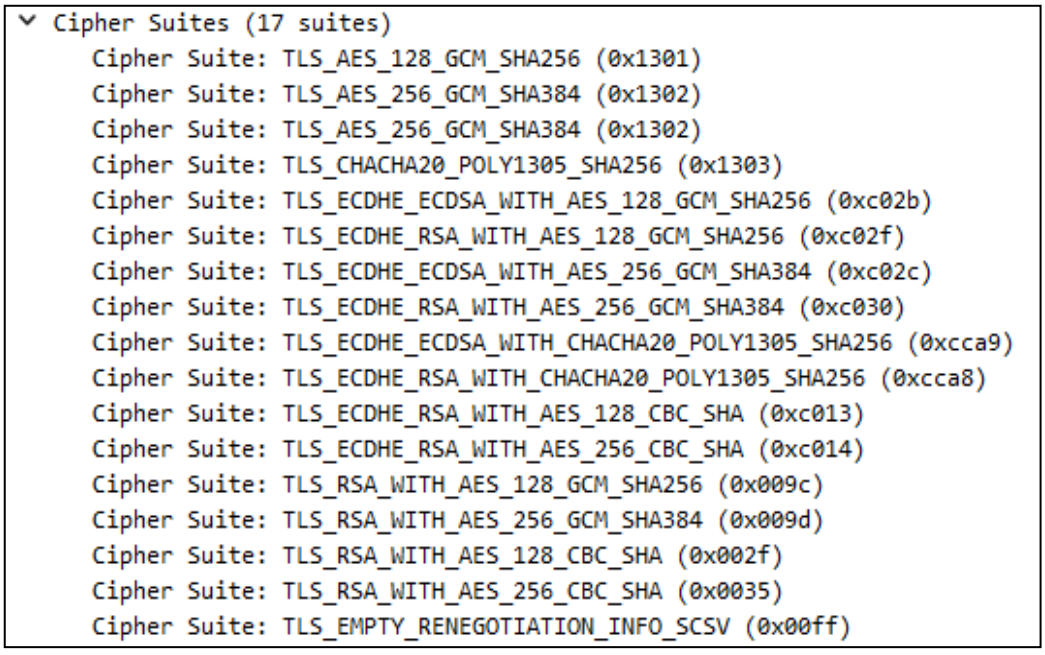


Figura 6.4. Cipher suites ofertes per el navegador Microsoft Edge

Mozilla Firefox

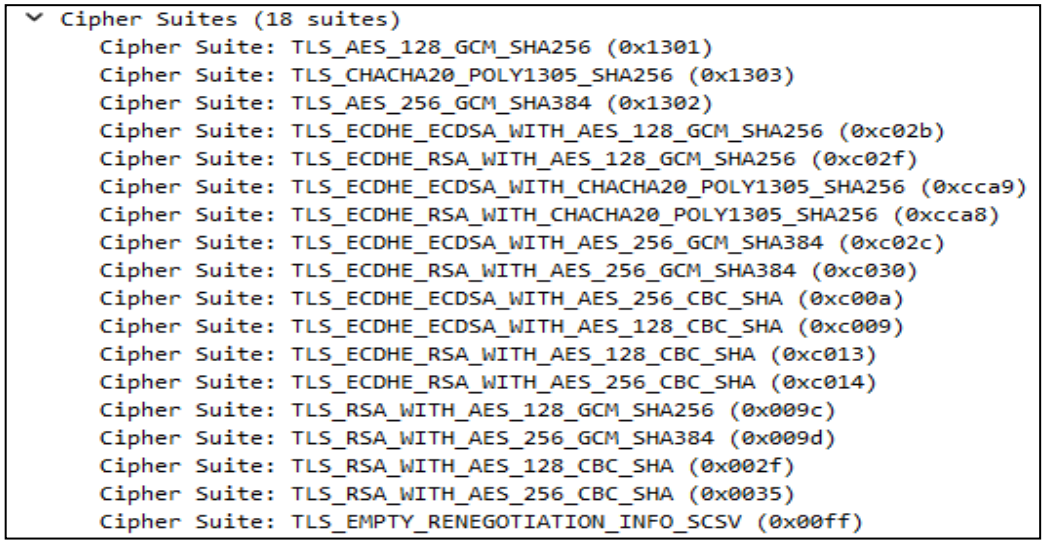


Figura 6.5. Cipher Suites ofertes per el navegador Mozilla Firefox

6.1.2 Descripció d'una cipher suite

Una cipher suite es en si una entitat única, cadascuna ha estat confeccionada segons els algorismes que li son compatibles. La cipher suite és la responsable de proporcionar mètodes d'intercanvi de clau, autenticació i funcions de Hash. [22].

Components d'una cipher suite

TLS_ClauIntercanvi_ClauAut_WITH_Longtud_Clau_Mode_FuncioHash

Llegenda:

Protocol Intercanvi claus. Versió de TLS. TLSv1.2 o TLSv1.3

Clau Intercanvi. Negociació del algorisme d'intercanvi de claus

Autenticació. Negociació del algorisme de clau publica. Signatura i autenticació

Clau_Mode. Negociació claus compartides simètriques per xifrar el Handshake.

FuncioHash. Negociació de la funció d'autenticació a fer servir

Les Cipher Suites recomanades per TLS 1.2 segons el CNN son:

- TLS-ECDHE-ECDSA-AES128-GCM-SHA256
- TLS-ECDHE-ECDSA-AES256-GCM-SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384

TLSv1.3 Incorpora els següents canvis

- Eliminació algorismes simètrics. Predeterminat AEAD.
- Eliminació ús Hash SHA1.
- En els mecanismes d'intercanvi de clau sols fan us dels que proporcionen Fordward Secrecy.
- Incorporació dels algorismes de corba el·líptica.
- Noves claus. Ús de primitives HKDF.
- Suites suportades
    - TLS-AES_128-GCM_SHA256
    - TLS-AES_256_GCM_SHA384
    - TLS-AES_CHACHA20_POLY1305_SHA256
    - TLS-AES_128_CCM_SHA256
    - TLS-AES_128_CCM_8_AES256

Protocols de la cipher suite	
ECDHE	Eliptic curve Diffie-Hellman. Protocol d'establiment de claus. Ús de corbes el·líptiques. Establiment d'un secret compartit (efímer)
ECDH	Eliptic curve Diffie-Hellman. Protocol d'establiment de claus. Ús de corbes el·líptiques. Establiment d'un secret compartit per usar-se com a clau
DHE	Diffie-Hellman. Protocol d'establiment de claus. Sense autenticació
AES	Advanced Encryption Standar. Esquema de xifrat per blocs.
AES-GCM	Advanced Encryption Standar. Esquema de xifrat per blocs. Proporciona autenticació. Diferents longituds de clau
3DES	Triple-DES. Algorisme de xifrat. Us de l'algorisme DES i realitza el xifrat tres cops
DES	Estàndard de xifrat de dades. Considerat insegur
SHA	Secure Hash Algoritm. Incorpora funcions de hash. Proporciona autenticitat. SHA-1,SHA-2
HMAC-SHAx	Proporciona autenticitat i integritat al missatge

Taula 6.2. Protocols inclosos en la cipher suite

Algorisme d'intercanvi de claus	Tipus certificat servidor
ECDH_ECDSA	El certificat ha de contenir una clau publica ECDH. Ha de ser signat amb ECDSA
ECDHE_ECDSA	El certificat ha de contenir una clau publica ECDSA. Ha de ser signat amb ECDSA
ECDHE_RSA	El certificat ha de contenir una clau publica ECDH. Ha de ser signat amb RSA
ECDHE_RSA	El certificat ha de contenir una clau publica RSA autoritzada per ser usada en signatures digitals. Ha de ser signada amb RSA

Taula 6.3. Tipus de certificat Servidor

Mètode d'autenticació	Tipus certificat servidor
ECDSA_sign	El certificat ha de contenir una clau publica ECDH. Ha de ser signat amb ECDSA
ECDSA_fixed_ECDH	El certificat ha de contenir una clau publica ECDH amb la mateixa corba que la clau ECDH del servidor. Ha de ser signat amb ECDSA
RSA_fixed_ECDH	El certificat ha de contenir una clau publica ECDH amb la mateixa corba que la clau ECDH del servidor. Ha de ser signat amb RSA

Taula 6.4 Tipus de certificat client

Font taules:

https://www.secg.org/sec2-v2.pdf [21]

6.1.3. Corbes compatibles

La elecció del tipus de corba el·líptica determina la longitud de la clau del certificat ECDSA. Hem pogut comprovar que els navegadors web ja no accepten corbes de menys de 256 bits (2048 bits. RSA), i que alguns tipus no son compatibles amb les cipher suites. Anem a presenta corbes compatibles i la seva correspondència amb certificats RSA. Per aquest estudi han resultat compatibles les corbes secp256r1, secp384r1 i secp521r1 [21].

Corbes recomanades		
Corba	Tamany clau	Tamany Clau RSA
secp192k1	192	1536
secp192r1	192	1536
secp224k1	224	2048
secp224k1	224	2048
secp256k1	256	3072
secp256r1	256	3072
secp384r1	384	7680
secp521r1	521	15360

Taula 6.5. Corbes recomanades

A continuació passem a presentar un llistat de corbes i la seva compatibilitat amb corbes d'altres organitzacions.

SECG	AINSI X9.62	NIST
sect163k1		NIST K-163
sect163r1		
sect163r2		NIST B-163
sect193r1		
sect193r2		
sect233k1		NIST K-233
sect233r1		NIST B-233
sect239k1		
sect283k1		NIST K-283
sect283r1		NIST B-283
sect409k1		NIST K-409
sect409r1		NIST B-409
sect571k1		NIST K-571
sect571r1		NIST B-571
secp160k1		
secp160r1		
secp160r2		
secp192k1		
secp192r1	prime192v1	NIST P-192
secp224k1		
secp224r1		NIST P-224
secp256k1		
secp256r1	prime265v1	NIST P-256
secp384r1		NIST P-384
secp521r1		NIST P-521

Taula 6.6. Corbes equivalents definides per SECG, ANSI I NIST

6.1.4. Handshake

En el moment que un client web vol iniciar sessió en accedir a un servidor web configurat per establir una connexió HTTPS, el primer pas per accedir-hi es la negociació del xifrat a implementar. *"En la fase de negociació de TLS (Handshake) s'acorden els paràmetres a fer servir"* [20]. Aquests paràmetres son implementats de manera conjunta, i s'han establert diferents combinacions a fi d'obtenir una connexió segura que sigui compatible per a tothom.

Per part de diferents organismes internacionals -NIST, CCN, RFC- han establert les recomanacions per establir connexions segures. Aquestes, son dinàmiques ja que amb el temps s'ha aconseguit trencar la seguretat d'algunes d'elles, i aquest fet ha promogut que es marqués un límit mínim de seguretat; la RFC de SSL/TLS impedeix l'ús d'alguns d'ells en els navegadors per ser considerats no segurs [21].

L'algorisme queda dividit es dos segments ben diferenciats; la primera part proporciona mètodes d'intercanvi de la clau, autenticació i mode de com operar, la segona es una funció de Hash.

El procés de Handshake ha estat sotmès a millores que fan que la connexió sigui més segura i mes eficient, ja que s'han escurçat els passos. Aquest, depèn de la versió de TLS negociada.

6.2. Abast

Per a realitzar la captura de dades realitzarem les proves a les següents velocitats: 10, 100 Mbps amb una doble finalitat; primer veure el temps necessari per establir una connexió HTTPS en condicions "normals" d'internet, i una altre, la velocitat de xarxa real que ens podem trobar en qualsevol LAN.

Paràmetres

- Ús de certificats auto signat
- Certificats RSA / ECDSA
- Longituds de 1024, 2048, 3076, 7680 i 15360 bits
- Url: https://vicvic82.ddns.net
- Mida del web. 10 Kb

- Adreçament DNS. Es configura l'arxiu hosts
- TLSv1.2 / TLSv1.3
- Xarxa LAN. 10 / 100 Mbps
- Cipher suite predeterminada per el Handshake

Navegadors

- Microsoft Edge
- Google Chrome
- Mozilla Firefox

Des d'un client web configurarem una sessió HTTPS. Des del programa WireShark capturarem el temps que triga en establir sessió i en descarregar un fitxer de dimensió coneguda, en un entorn de xarxa de 10 / 100 Mbps. Obtindrem la dada del timestamp del darrer paquet.

6.3. Comparativa

En aquesta part del estudi volem comprovar el rendiment dels navegadors web respecte al tipus de clau triada per establir una connexió segura.

Certificats RSA	Certificats ECDSA
rsa_1024.pem	ecdsa_160.pem
rsa_2048.pem	ecdsa_224.pem
rsa_3072.pem	ecdsa_256.pem
rsa_7680.pem	ecdsa_384.pem
rsa_15360.pem	ecdsa_521.pem

Taula 6.7. Comparativa seguretat certificats

Les corbes ecdsa 160 i 224 han resultat no ser compatibles amb l'estàndard TLS. A conseqüència d'aquest fet no ha estat possible capturar dades no establir cap comparativa amb aquestes longituds de claus.
Les dades de la comparativa han estat obtingudes per mig de capturar el procés d'establir una sessió HTTPS per mig de WireShark.

Velocitat 100 Mbps

Protocol: TLSv1.2

Cipher suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

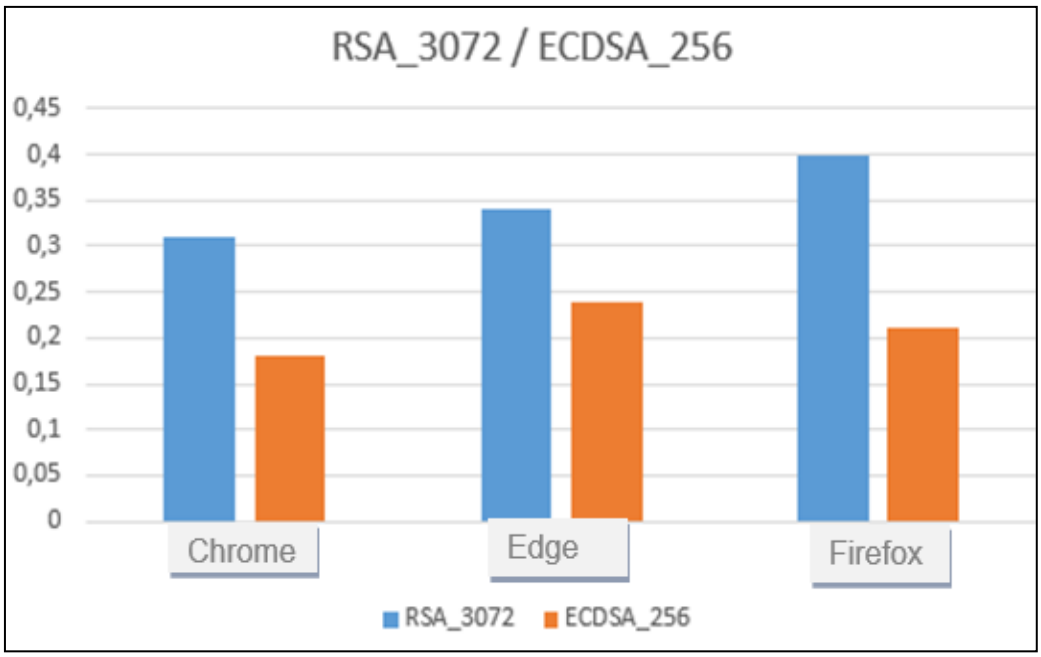


Figura 6.7. Comparativa rsa_3072 / ECDSA_ 256 / TLS_v1.2 / 100 Mbps

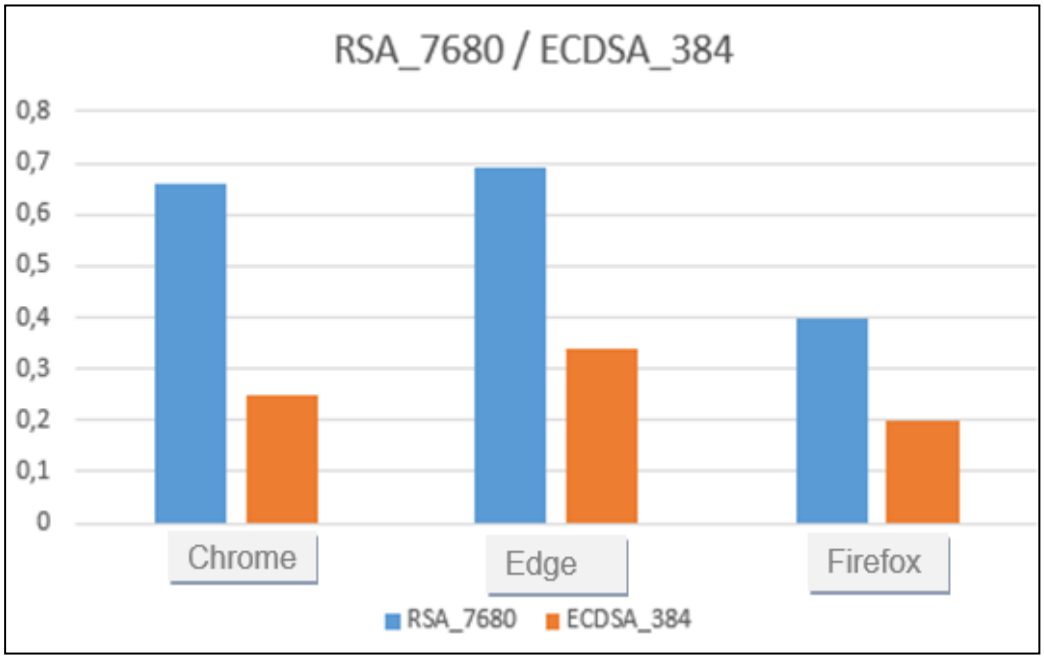


Figura 6.8. Comparativa rsa_7680 / ECDSA_ 384 / TLS_v1.2 / 100 Mbps

Velocitat 100 Mbps

Protocol: TLSv1.3

Cipher suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

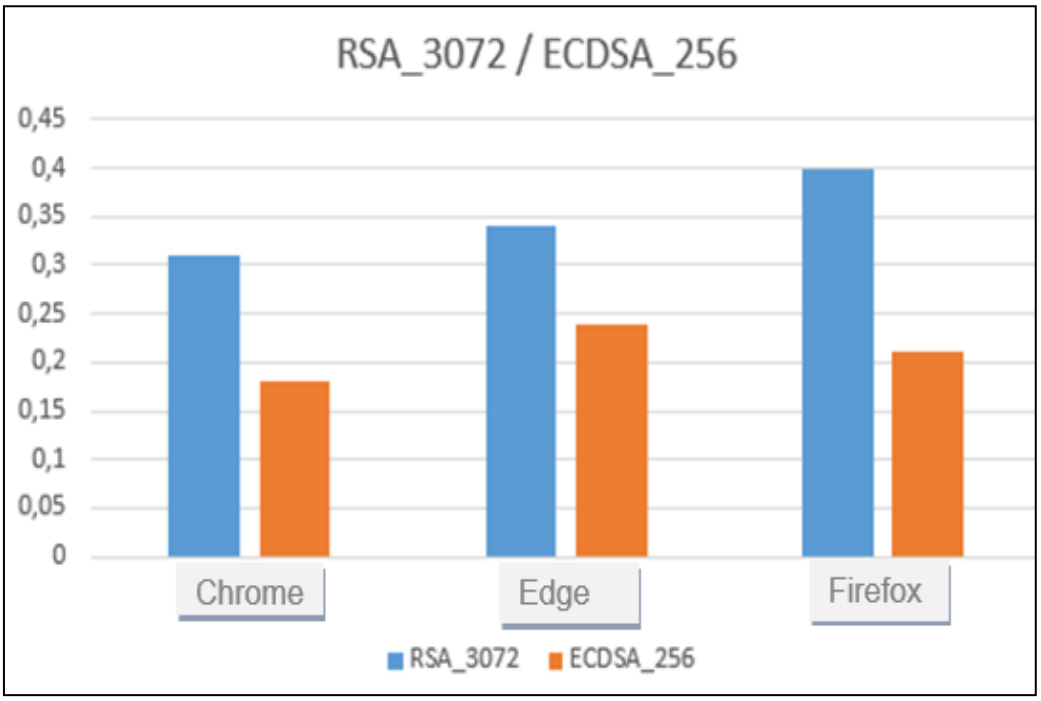


Figura 6.9. Comparativa rsa_3072 / ECDSA_ 256 / TLS_v1.3 / 100 Mbps

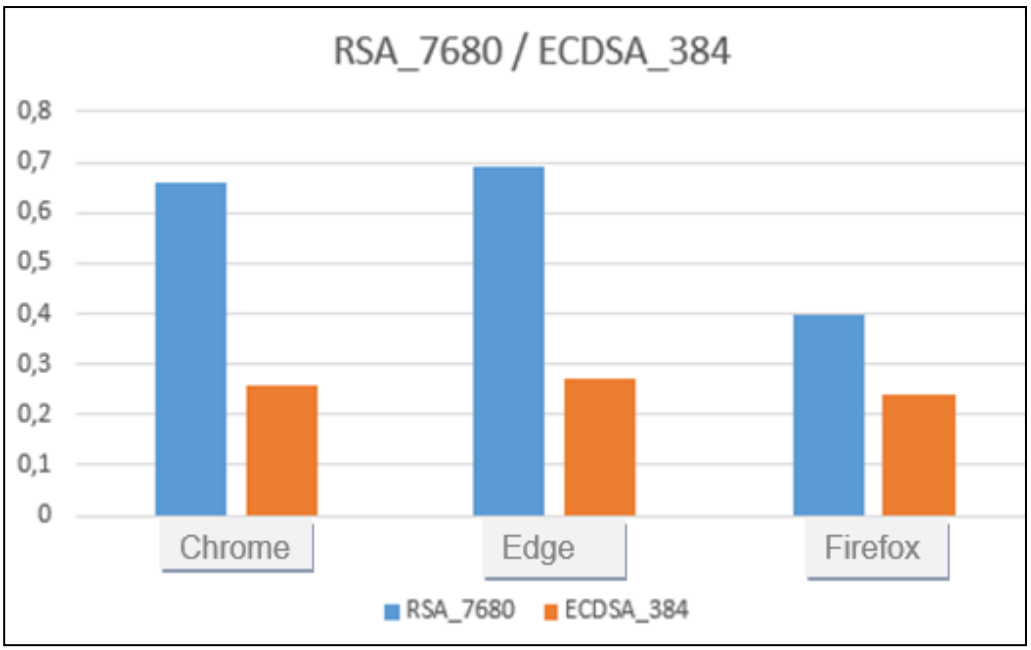


Figura 6.10. Comparativa rsa_7680 / ECDSA_ 384 / TLS_v1.3 / 100 Mb

Velocitat 10 Mbps

Protocol: TLSv1.2

Cipher suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

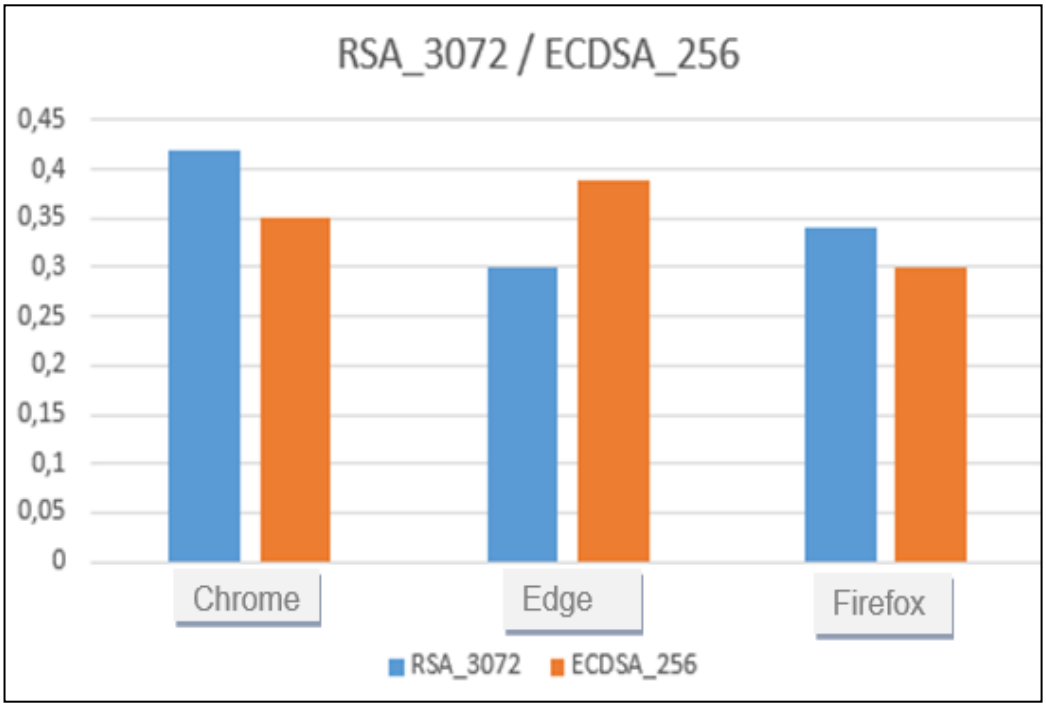


Figura 6.11. Comparativa rsa_3072 / ECDSA_ 256 / TLS_v1.2 / 10 Mbps

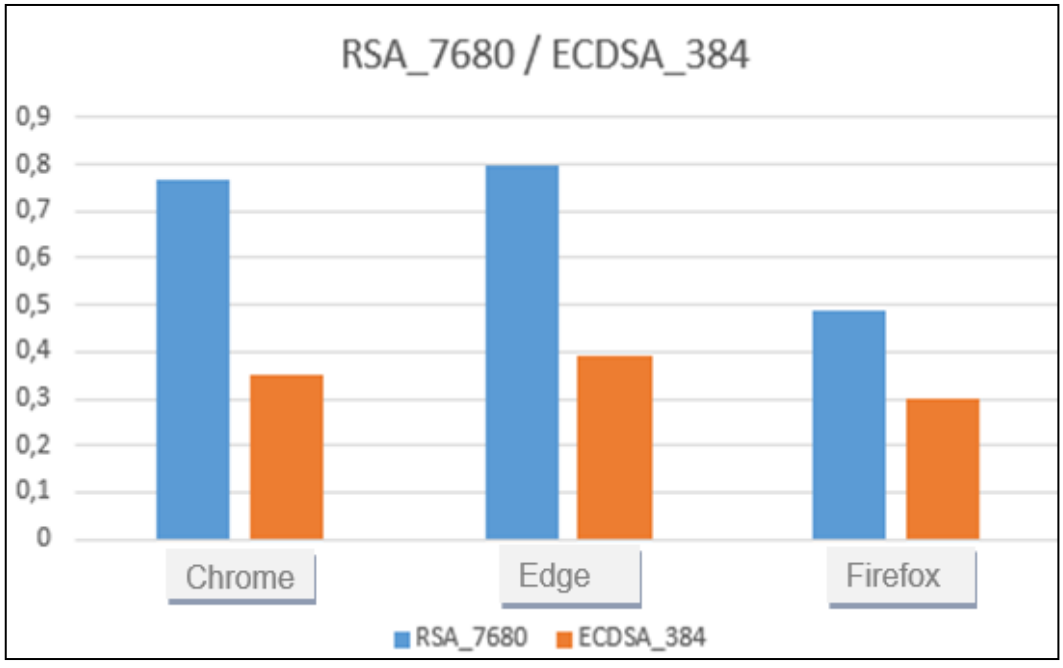


Figura 6.12. Comparativa rsa_7680 / ECDSA_ 384 / TLS_v1.2 / 10 Mbps

Velocitat 10 Mbps

Protocol: TLSv1.3

Cipher suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

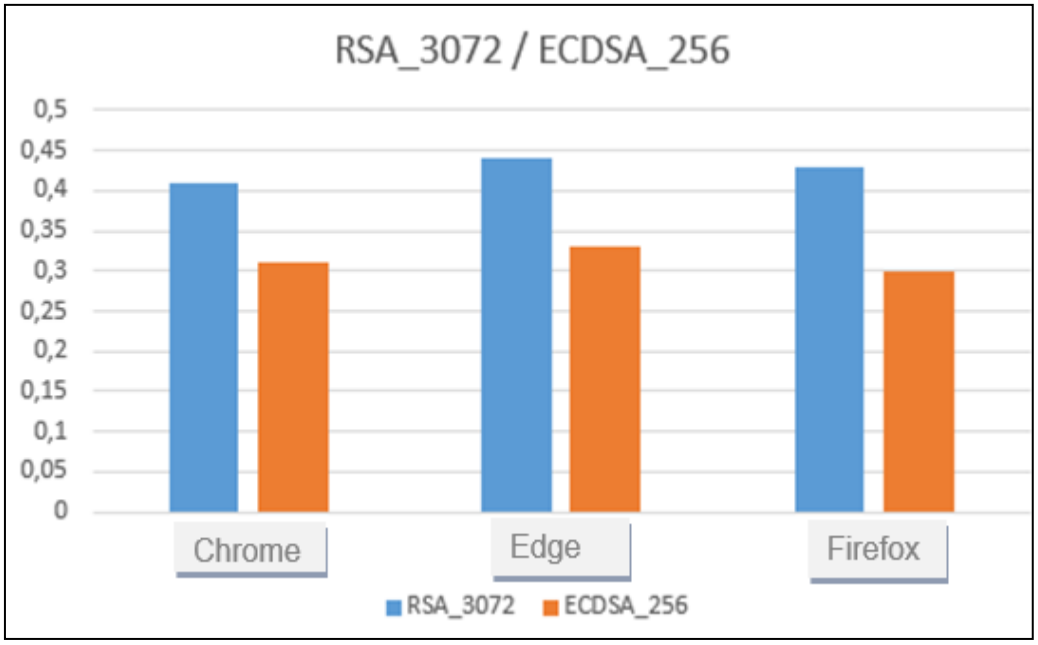


Figura 6.13. Comparativa rsa_3072 / ECDSA_ 256 / TLS_v1.3 / 10 Mbps

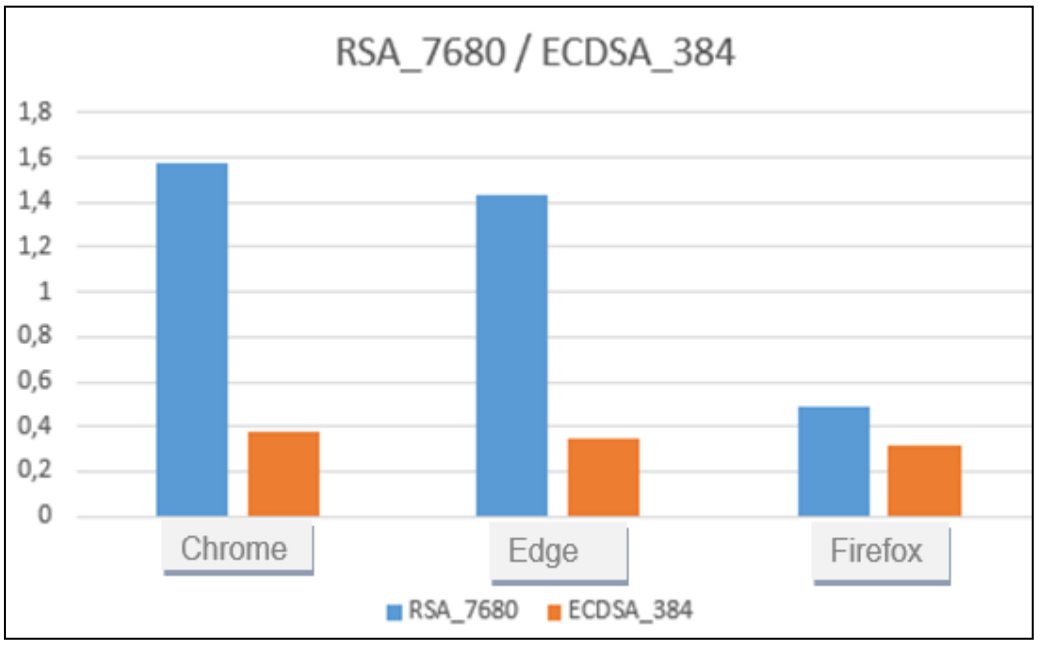


Figura 6.14. Comparativa rsa_7680 / ECDSA_ 384 / TLS_v1.3 / 10 Mbps

6.4. Discussió dades

En aquesta part del estudi volem comprovar el rendiment dels navegadors web respecte al tipus de clau triada per establir una connexió segura. En termes genèrics, la comparativa mostra les diferencies de temps en carregat el web. Mesurat en segons.

- En tota la comparativa s'aprecia com amb una longitud de clau similar, *ECDSA* presenta un rendiment millor en tots els navegadors. El més rellevant es que l'ús d'una clau *ECDSA_384*, combinat amb l´ús de *TLSv1.2* redueix quasi a la meitat el temps necessari en carregar un web. Aquest temps es veu reduït més d'una tercera part amb l'ús de *TLSv1.3.*
- Podem apreciar una lleugera diferencia de rendiment respecte als navegadors, però quasi no és apreciable. Firefox resulta oferir una lleugera avantatge.
- Haver de xifrar la informació amb menys bits fa que *ECDSA* mostri un rendiment superior, que és més apreciable en webs amb més dades.
- Les diferencies entre *RSA i ECDSA* son mes apreciables en entorns amb menys ample de banda de descarrega.

Hem comprovat com

- RSA és compatible amb tots els navegadors, mentre que ECDSA sols ofereix compatibilitat amb unes longituds de clau de 256 i 324 Bits
- ECDSA guanya força en fer us de TLSv1.3
- El temps de xifrat augmenta desproporcionadament en fer ús d'una clau de 15360 Bits. Es per això que els navegadors no presten suport a claus tant grans, sols Firefox en presta suport.
- De lluny, l'ús d'una clau ECDSA_384 es la que presenta més avantatges en quant a seguretat i rendiment. El fet que la clau ECDSA_521 hagi estat vulnerada degut a problemes de compatibilitat amb Java ha fet que els desenvolupadors no la presentin com una clau compatible en alguns navegadors web.

7. Rendiment FTPS

7.1. Descripció

Per a conèixer el rendiment que ens ofereix un servidor FTPS estudiarem una connexió FTPS. Ens interessa conèixer quant a quan triga el temps de connexió i el temps de descàrrega. Obtindrem aquesta dada per mig del analitzador de xarxa anomenat *WireShark.*

Per la realització de les proves s'ha dut a terme la instal·lació del programa FileZilla Server i s'ha configurat per oferir connexions xifrades per mig dels certificats SSL creats. Tanmateix, hem configurat *FileZilla Server* per a oferir una connexió FTP explícit sobre TLS generada a partir dels certificats auto signats.

"En computadors que treballin sobre entorn Windows s'ha de tenir en compte la utilització de la CPU durant l'enllaç de la connexió degut als càlculs que haurà d'executar en xifrar i desxifrar les dades" [24].

Per a dur a terme la captura de dades d'una connexió FTPS hem efectuat la descarrega d'un fitxer de 55,840 Mbytes, a la vegada que capturàvem amb WireShark la durada de la descarrega.

7.2. Abast

Totes les proves son realitzades en un entorn de xarxa LAN a 10 Mbps, volem simular una situació habitual en molts dispositius connectats a internet, D'altra banda, així es posen de manifest les diferencies de rendiment.

FileZilla ha acceptat totes les longituds de clau dels certificats RSA creats, mentre que sols ha acceptat els certificats ECDSA amb claus de 256, 384 i 512 bits, degut a les especificacions TLS.

En la prova hem anat alternat els certificats fins a obtenir dades fiables de totes les longituds de clau. Hem seleccionat diferents velocitats de xarxa a fi de poder comparar una descàrrega des d'internet a diferents velocitats. Tanmateix hem volgut estudiar el temps de descarrega total, i el temps que durava el procés de *Handshake. Dades obtingudes amb WireShark.*

7.3. Comparativa

10 Mbps

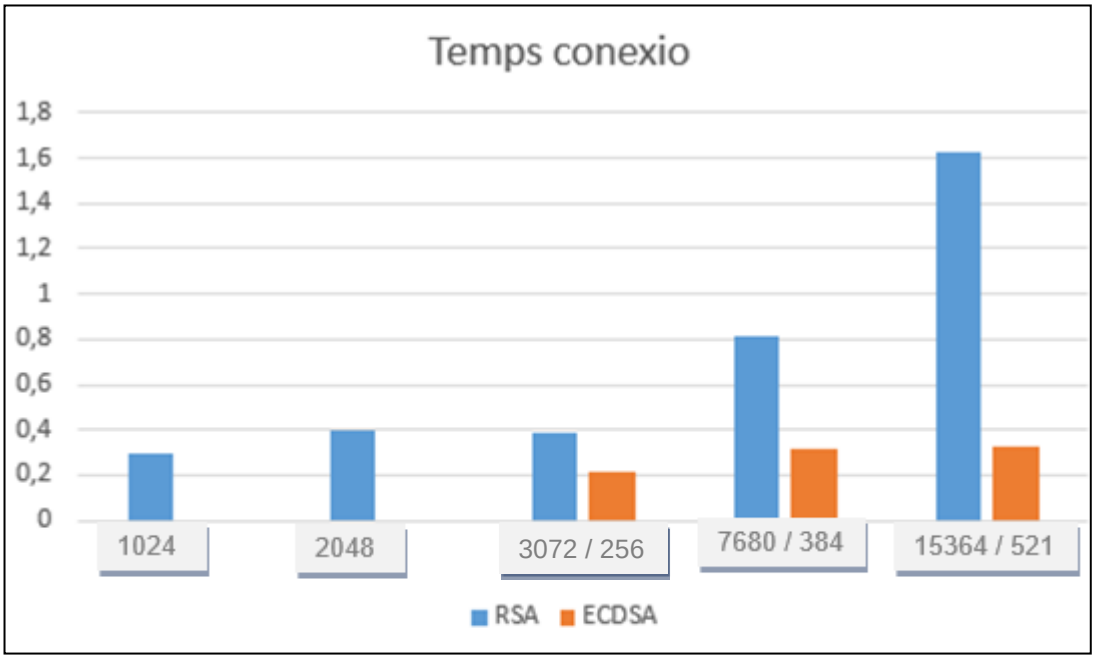


Figura 7.1. Comparativa rendiment TLS. Connexió / 10 Mbps

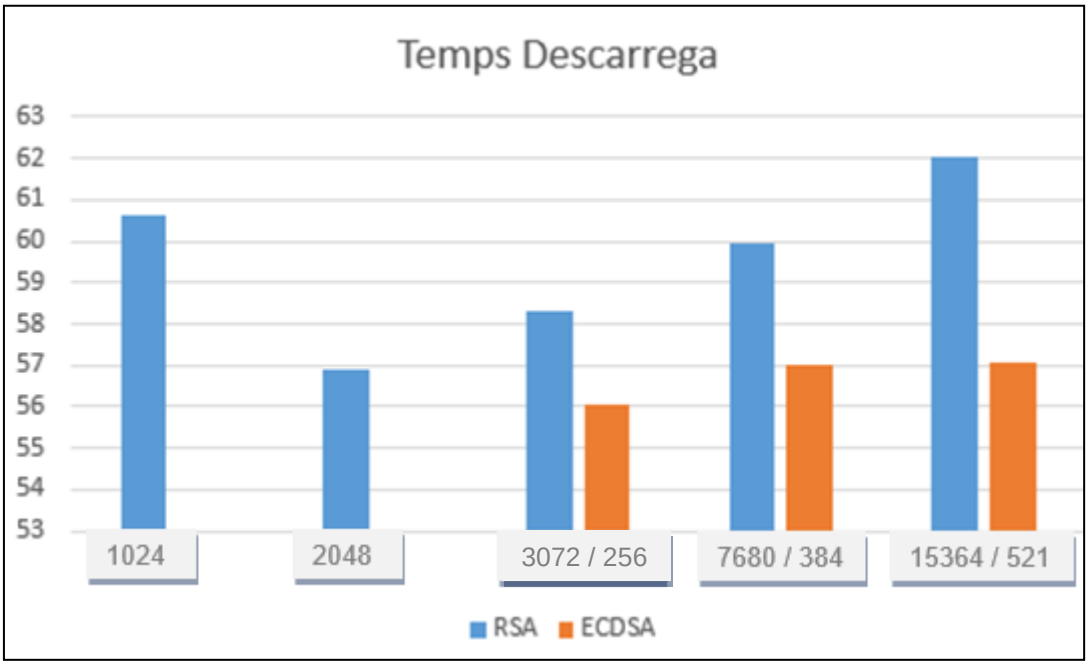


Figura 7.2.. Comparativa rendiment TLS. Descarrega / 10 Mbps

100 Mbps

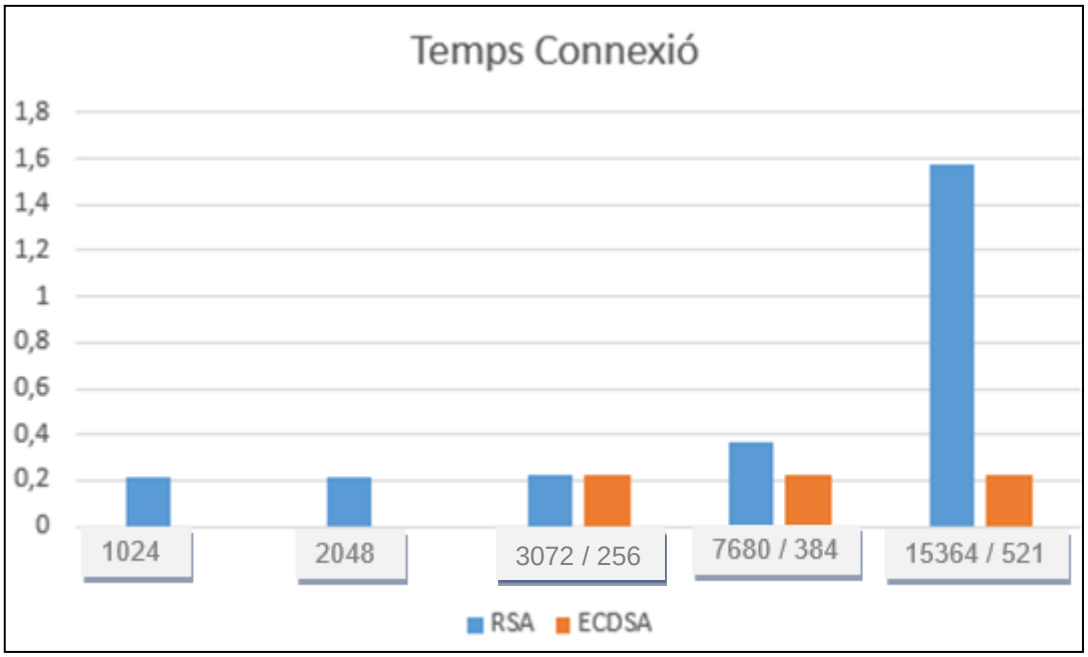


Figura 7.3. Comparativa rendiment TLS. Connexió / 100 Mbps

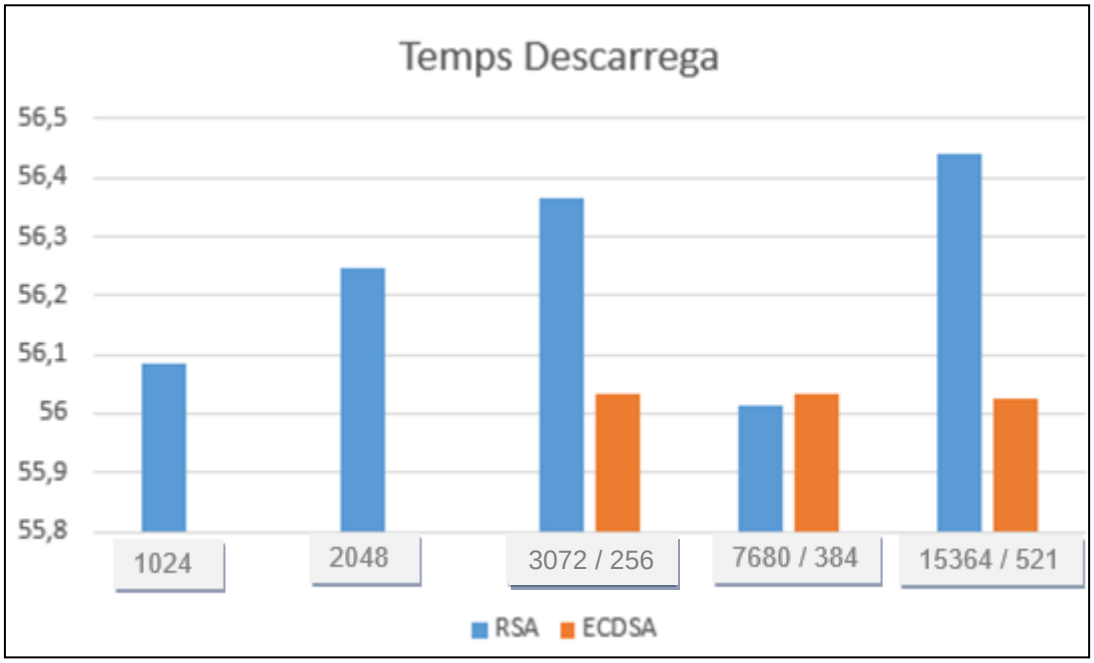


Figura 7.4. Comparativa rendiment TLS. Descàrrega / 100 Mbps

7.4. Discussió dades

De la comparativa d'una descarrega a 10 Mbps podem obtenir unes dades semblants a una connexió a internet habitual. En termes generals, en les connexions dutes a terme amb certificats RSA s'observa com augmenta el temps de connexió conforme fem ús d'una longitud de clau més gran. Els certificats RSA_3072 i RSA_7680 ofereixen un rendiment més que raonable, i proporcionen una seguretat validada per FIPS i RFC. Una longitud més gran de clau ja presenta uns temps no assumibles.

Hem observat com la elecció de la cipher suite responsable del Handshake determina en gran part el rendiment, així com la versió de TLS feta servir.

Podem observar com ECDSA, amb TLSv1.3 redueix quasi una tercera part els temps de connexió al servidor (Handshake + descarrega de la interfície). On més diferencia podem percebre es en la descarrega del fitxer. Tot i sent acceptables els temps de descàrrega, trobem una diferencia de mitja d'uns 5 segons amb certificats RSA.

Am una connexió de 100 Mbps, tot es raonablement acceptable en quan a usabilitat, tots dos certificats han presentat uns temps similars. La diferencia s'ha fet notòria en fer ús de les claus més grans, però diferencies de dins a un segon son perfectament acceptables per computadors amb recursos.

De nou en surt com a guanyador un certificat ECDSA creat amb claus de 384 BITS, seguit de ECDSA_256 I RSA_3072 i RSA_7680.

8. Rendiment SSL/TLS

8.1. Descripció

Hem volgut oferir una altre perspectiva respecte al rendiment TLS. Ara ens ha interessat conèixer quantes connexions en un cert període de temps accepta el servidor. Aquest es un tema del que principalment s'entén que es responsabilitat del hardware i unes bones connexions, però hem pogut comprovat que l'elecció correcte del certificat i de l'algorisme de cipher suite fet servir poden optimitzar les connexions realitzades.

Per a monitoritzar el rendiment SSL/TLS que ens ofereix el servidor farem ús del programa OpenSSL. Per mig de l'ordre *s_time* implementa un client genèric que es connecta a un altre equip. Nosaltres ens connectarem al servidor i sol·licitarem la pàgina per defecte establerta en aquest. Per mig d'aquesta comanda podem obtenir el nombre de connexions en un període determinat i la quantitat de dades transferides [25].

8.2. Abast

- Ús de certificats auto signat
- Certificats RSA / ECDSA
- Longituds de 1024, 2048, 3072, 7680 i 15360 bits
- Url: https://vicvic82.ddns.net
- Mida del web. 10 Kb
- Adreçament DNS. Es configura l'arxiu hosts
- TLSv1.2 / TLSv1.3
- Xarxa LAN. 10 / 100 Mbps
- Cipher suite predeterminada per el Handshake

8.3. Comparativa

La comparativa aporta dades del nombre de connexions establertes en un segon per usuari, i del nombre de Handshakes establerts per segon. En l'annex podem trobar una mostra de com s'han obtingut les dades de la comparativa, així com una taula amb totes les dades obtingudes.

8.3.1. Comparativa claus RSA / ECDSA

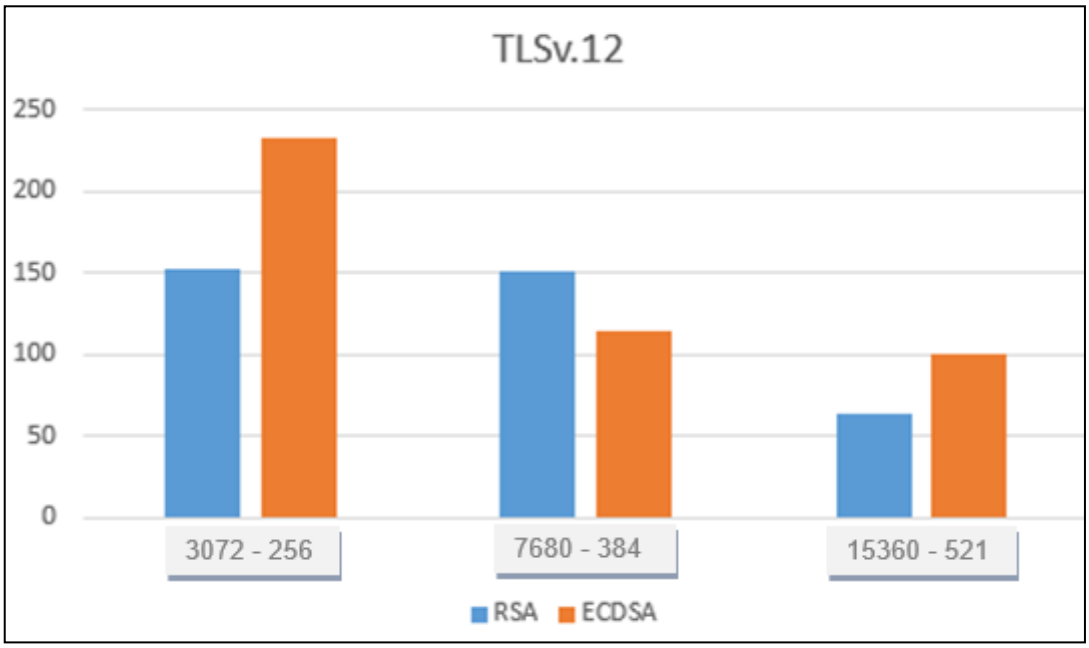


Figura 8.1. Comparativa nombre connexions x segon. TLSv1.2

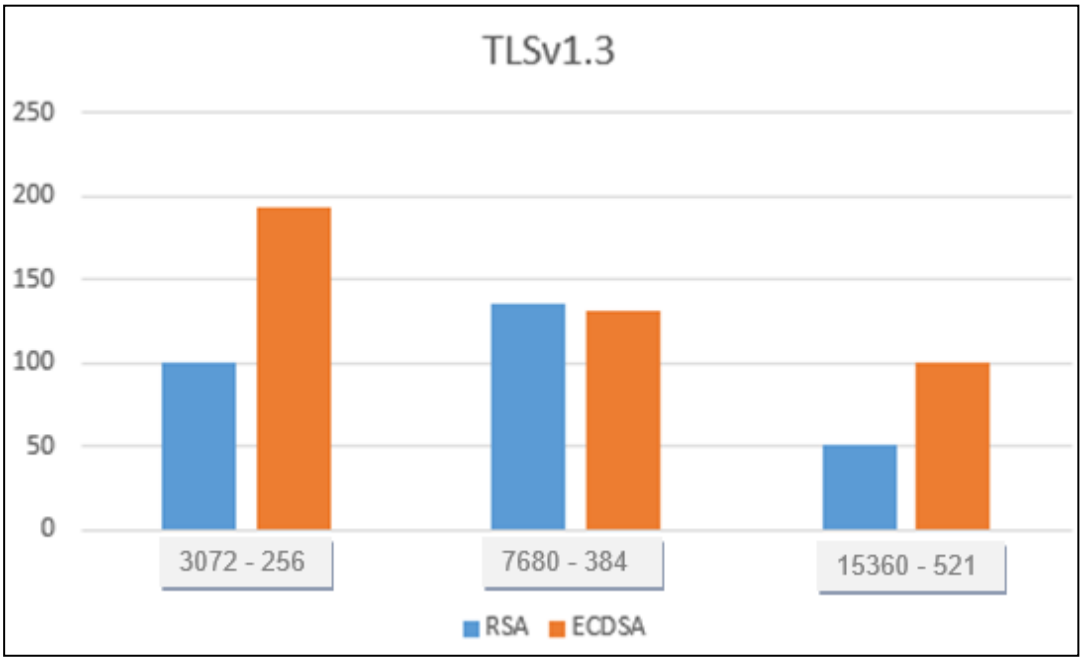


Figura 8.2. Comparativa nombre connexions x segon. TLSv1.3

8.3.2. Comparativa claus RSA / ECDSA. Handshake

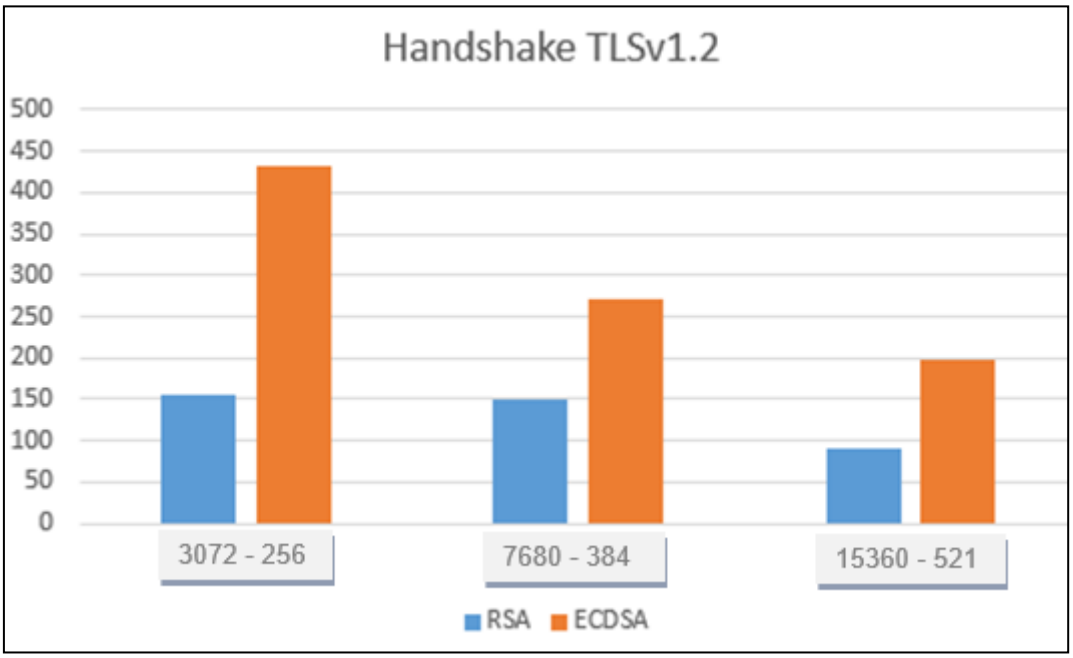
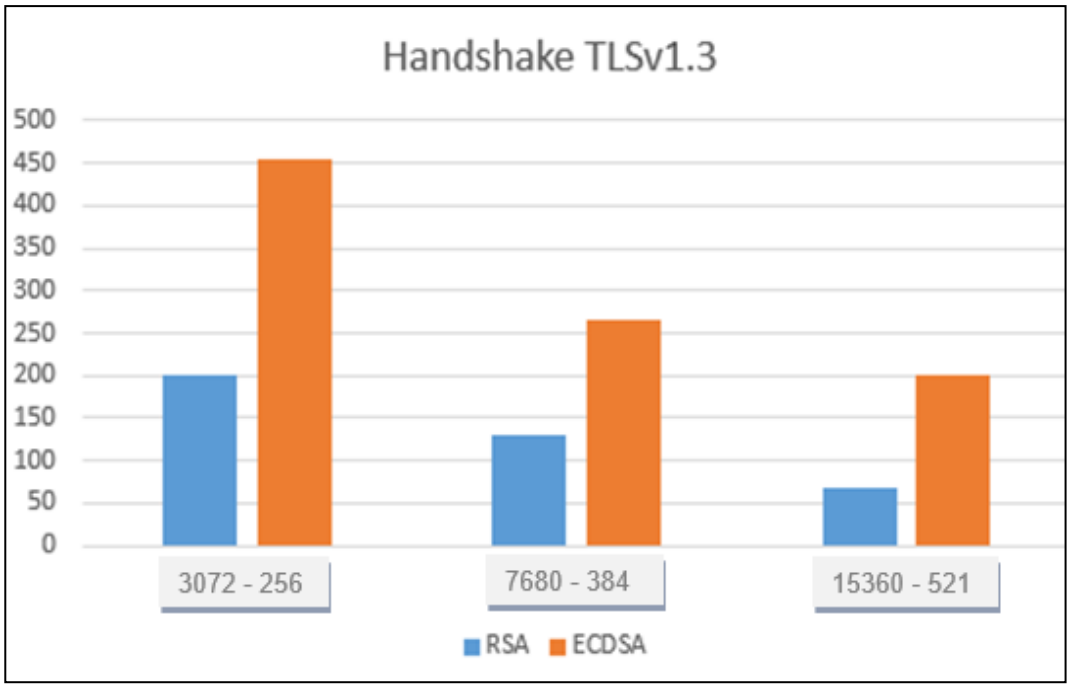


Figura 8.3. Comparativa nombre Handshake x segon. TLSv1.2



omparativa nombre Handshake x segon. TLSv1.3

8.3.3 Comparativa claus RSA / TLSv1.2 -TLSv1.3

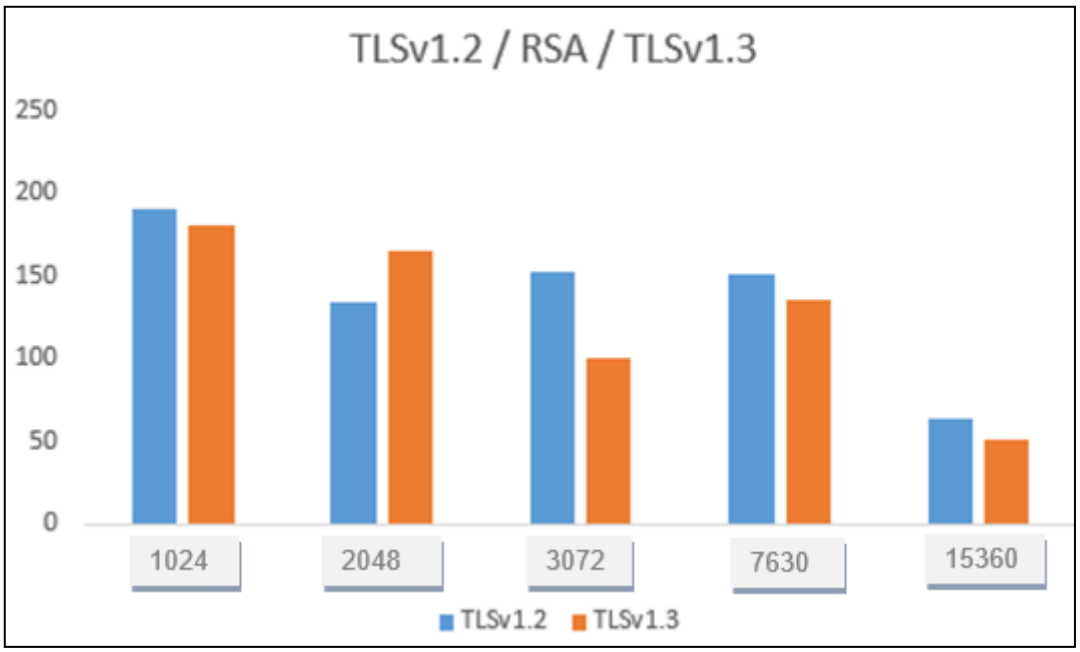


Figura 8.5. Comparativa RSA. TLSv1.2 – TLSv1.3

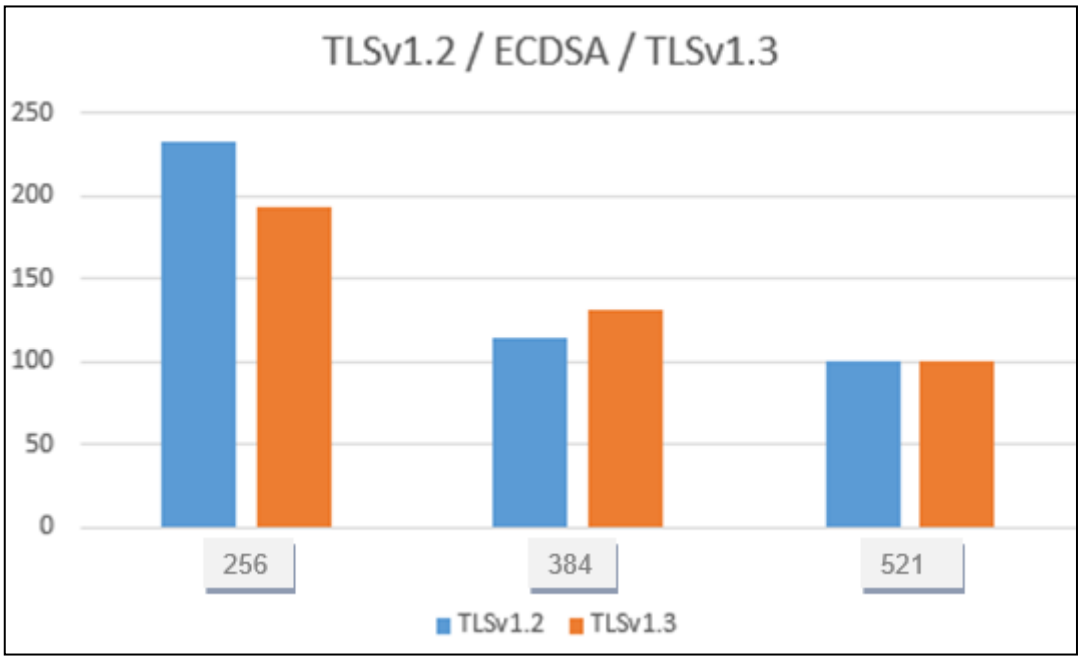


Figura 8.6. Comparativa ECDSA. TLSv1.2 – TLSv1.3

8.4. Discussió dades

Hem volgut oferir una altre perspectiva respecte al rendiment TLS. Ara ja no es tan lligada al rendiment de l'aplicació i als segons que pot trigar en realitzar una acció. Ara ens ha interessat conèixer quantes connexions en un cert període de temps accepta el servidor. Aquest es un tema del que principalment s'entén que es responsabilitat del hardware i unes bones connexions, però hem pogut comprovat que l'elecció correcte del certificat i de l'algorisme de cipher suite fet servir poden optimitzar les connexions realitzades.

De nou observem una certa millora amb connexions TLSv1.3, propiciada sobre tot per que ha reduït el nombre de passos per concloure el Handshake. L'ús de corbes el·líptiques per a xifrar el Handshake és el responsable d'aquest fet. Aquesta millora es veu potenciada en cas d'estar configurat en el servidor un tipus de certificat ECDSA, ja que per defecte, quedarà seleccionat en el Handshake una *cipher suite* que xifri tota la connexió amb claus ECDSA.

El nombre de connexions establertes amb un certificat *ECDSA_256* / *RSA_3072* dobla les connexions establertes respecte a certificats amb claus mes grans; percebem que claus més llargues requereixen més temps en xifrar i desxifrar la informació. En visualitzar sols el nombre de Handshake aquest fet queda encara més en evidencia.

En definitiva, el certificat *ECDSA_256* es presenta com a guanyador indiscutible, doblant el nombre de connexions respecte als seu homònim RSA. Aquest comportament es millorat en usar connexions TLSv1.3.

Hem volgut mostrar les diferencies entre xarxes de diferents rendiments, de fet 10 Mbps ha estat un estàndard per connexions a internet, mentre que 100 Mbps encara és la velocitat real en moltes LAN. Les diferencies s'han fet notables amb velocitats de xarxa més lentes, encara que perfectament assolibles per els computadors actuals.

9. Rendiment SSH

Per a poder realitzar la comparativa hem instal·lat el sistema operatiu *Ubuntu* 20.03 en un entorn virtualitzat. Tanmateix hem tingut que tornar a crear les claus ja que SSH no reconeix els certificats x.509. En aquesta practica hem creat diferents tipus de certificats de Host que proporcionin autenticitat al servidor, a la vegada que li proporciona una connexió xifrada.

9.1. Descripció

SSH estableix una connexió xifrada de punt a punt, creant un efecte túnel per on viatgen les dades. A diferencia d'una connexió SSL/TLS, SSH permet l'administració remota d'equips. Hem configurat un entorn on sols ens valem de SSH per a establir una connexió xifrada per mig del programa *FileZilla.*

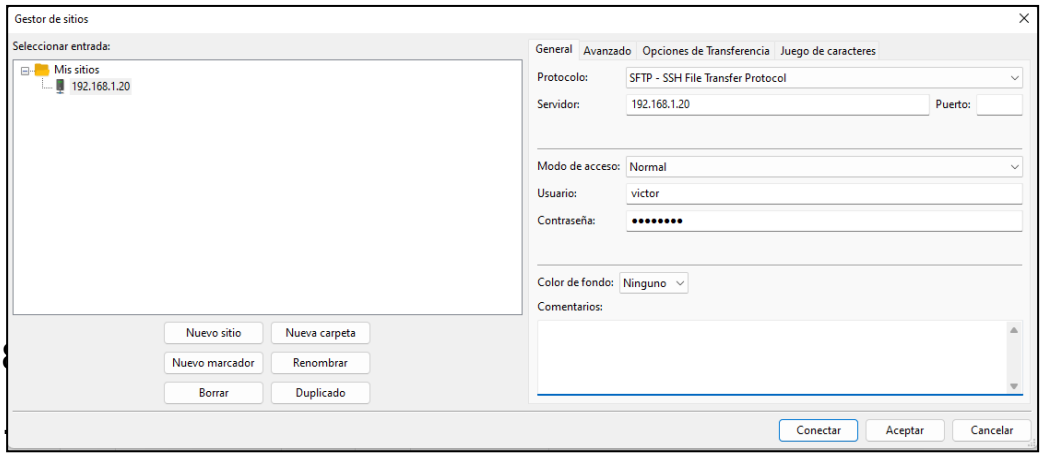


Figura 9.1. Connexió SFTP amb FileZilla

9.2. Abast

- Velocitats xarxa 10 / 100
- Algorisme Handshake: predeterminat
- Hem tingut que tornar a crear les claus ja que SSH no reconeix els certificats x.509. Se han creat claus RSA i ECDSA d'unes longituds de clau a fi de poder establir una comparativa. RSA ha permès crear claus de 2048, 3072, 7680 i 15364 bits, ECDSA de 256, 384 i 521
- Des d'un client FileZilla configurarem una sessió SFTP. Des del programa WireShark capturarem el temps que triga en establir sessió i en descarregar un fitxer de dimensió coneguda, en un entorn de xarxa de 10 / 100 Mbps. Obtindrem la dada del timestamp del darrer paquet.

9.3. Comparativa

10 Mbps

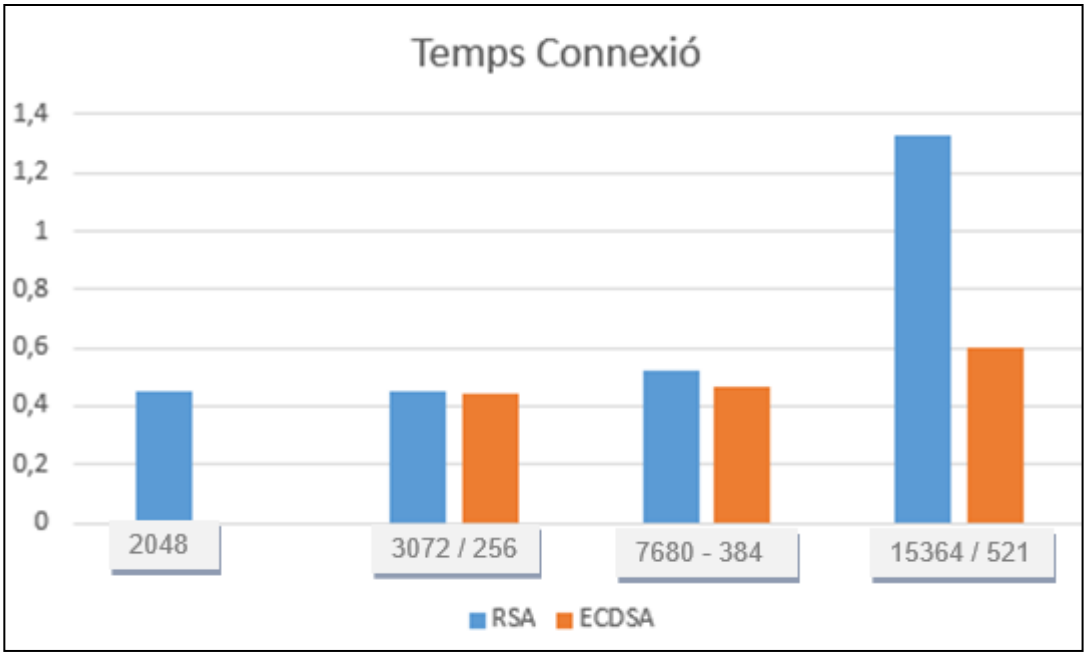


Figura 9.2. Comparativa rendiment SSH. Connexió / 10 Mbps

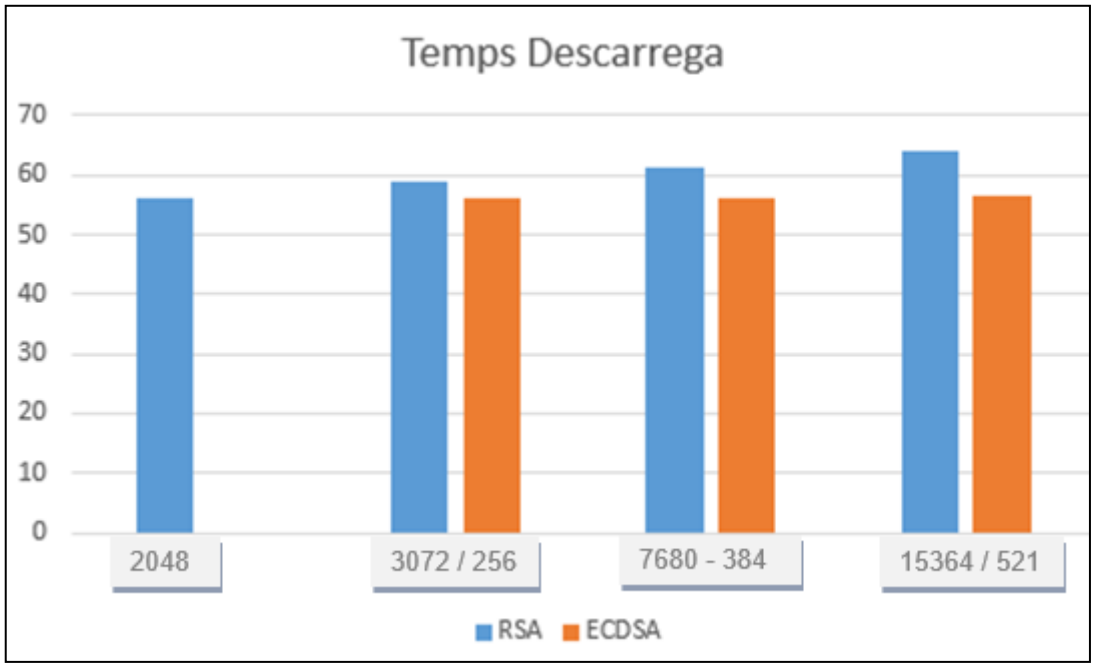


Figura 9.3. Comparativa descàrrega SSH. Connexió / 10 Mbps

100 Mbps

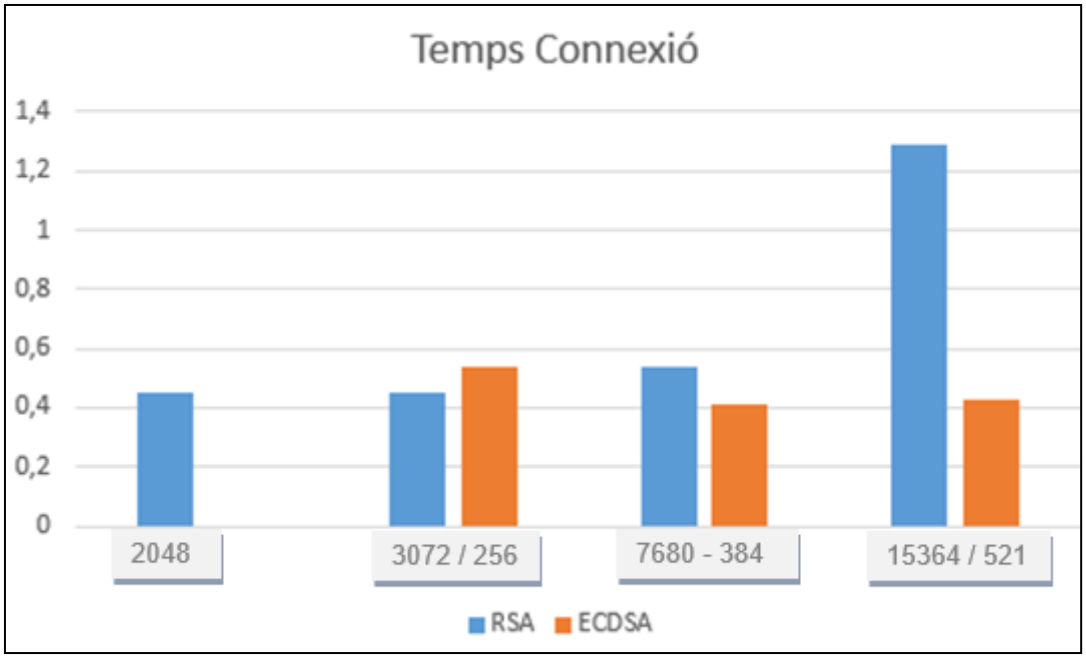


Figura 9.4. Comparativa rendiment SSH. Connexió / 100 Mbps

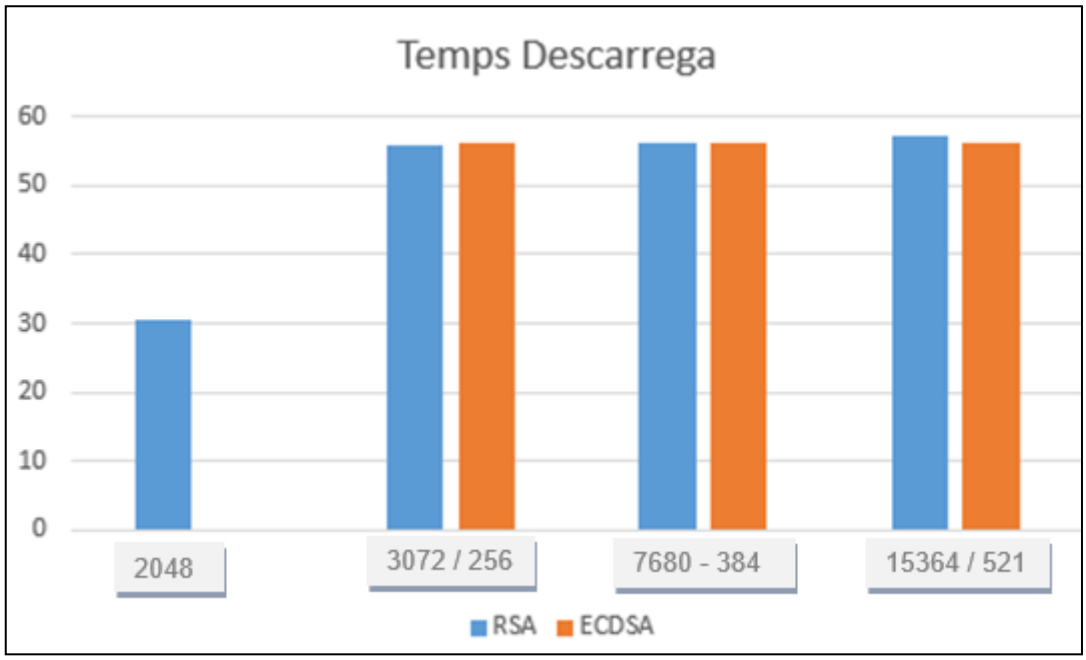


Figura 9.5. Comparativa rendiment TLS. Connexió / 100 Mbps

9.4. Discussió dades

En aquest cas la tònica es que menys en algunes excepcions, el rendiment és molt similar i quasi no es nota la diferencia de fer servir un parell de claus RSA o ECDSA. Unes diferencies de dècimes de segon no son motiu per poder afirmar que un d'ells és més eficient que l'altre. En aquest cas SSH no sols xifra la connexió sinó que la gestiona per complert, ja que es tracta d'un protocol de la capa d'aplicació; podem suposar que aquest fet es el que li dona la eficiència observada en les proves.

Els temps de connexió son òptims per igual i quasi no es nota diferencia a causa de fer servir corbes el·líptiques per xifrar el Handshake i reduir molt aquest procés. Ja sabem que les corbes el·líptiques fan ús de menys bits per oferir el mateix graus de seguretat. En quant al procés de Handshake, en contra de TLS, es veu molt reduït i optimitzat en quant a nombre de passos a realitzar.

Hem volgut comprovar per igual com afecta el xifrat de les dades respecte al temps que dura la descarrega. La resposta es que no sols no s'aprecia diferencia entre fer servir un certificat RSA o ECDSA sinó que inclòs en fer servir diferents longituds de clau el temps és quasi el mateix.

10. Conclusions

- Conclusions del treball

En primer lloc hem volgut posar de relleu la importància que presenta la compatibilitat dels navegadors vers la suite de xifrat oferta en la fase de Handshake. Fixar una cipher suite pot provocar que el navegador client no pugui carregar el web ja que el certificat instal·lat en el servidor web no suporta cap dels protocols oferts per el navegador web..

Així mateix hem pogut validar la idea de que cada cipher suite es en si una entitat construïda a fi d'oferir compatibilitat entre els algorismes que la composen i proporciona un grau de seguretat acceptable validat per les diferents entitats que defineixen Internet. IANA, RFC.

En connexions HTTPS o FTPS ha quedat evident que l'elecció de la cipher suite estava condicionada per el tipus de certificat instal·lat en el servidor.

S'ha posat de relleu que la seguretat proporcionada per les claus no sols depèn de la longitud de la clau o l'elecció de la corba el·líptica, sinó que depèn d'una encertada combinació entre el protocol fet servir per l'intercanvi de claus, les longituds de la clau i la seva combinació amb els protocols d'autenticació.

S'ha pogut comprovar que els certificats creats amb claus ECDSA, en comparació amb els seus equivalents RSA proporcionen un millor rendiment en quant a temps de xifrat de dades. El Handshake amb claus ECDSA es realitza de manera més eficient, així mateix permet la càrrega més ràpida del web.

La versió TLSv1.3, junt amb certificats de tipus ECDSA és la combinació que ha ofert un millor rendiment, tant en l'establiment de la connexió com el xifrat del es dades. El temps total s'ha vist reduït quasi a la meitat, la longitud de clau 384 en certificats ECDSA ha esdevingut la més eficient en totes les proves.

- Assoliment d’objectius inicials

Creiem haver assolit els objectius plantejats de manera parcial. Certament, el resultat de les proves ha estat objectiu, a la vegada que el coneixement adquirit ha estat veraç. Al principi vàrem formular unes hipòtesis, i un mètode per a validar aquesta hipòtesi. En iniciar la fase de verificació de seguida va quedar palès la lentitud d’efectuar la captura de dades; s’establien uns temps de varies setmanes en aconseguir tota la informació desitjada. La idea inicial era que l’abast inclogués sis de les cipher suites més usades. L’error d’aquest treball ha estat no haver trobat un algorisme més eficient que hagués obtingut totes les dades.

- Anàlisi seguiment planificació

El mètode aplicat ha esdevingut essencial en el seguiment de la planificació del projecte. Aplicar la metodologia *Kanban* ha forçat a definir etapes, fluxos de treball i com associar cada etapa a la següent. Cada etapa no podia continuar fins a ser assolida per complert, llavors es passava a l’altre. Cada etapa va estar emmarcada en un període de temps; assolir cada etapa en el seu marc ha estat el principal objectiu marcat en la confecció d’aquest projecte. Vam haver de modificar l’àmbit de captura de dades en tant esdevenia impossible d’acomplir amb el projecte, i es va emmarcar en oferir dades en un marc concret: en les dades proporcionades per la cipher suite seleccionada per defecte.

- Línies de treball futures

Aquest projecte ens ha permès adquirir coneixement sobre protocols de seguretat, intercanvi de claus, *Handshake*, xifrat, etc. La realització de les proves ha posat de rellevància com de important es la cipher suite triada.

Com a línies de treball futur ha quedat fixada la necessitat d’obtenir coneixement sobre els beneficis en l’ús de TLSv1.3 en quant a seguretat. Així com es necessari un estudi més profund sobre com el *Handshake* tria una *cipher suite* en concret.

11. Glossari

Autenticació. Uneix sense que hi haguí cap possible dubte la informació continguda en un certificat i l'emissor del certificat.

Cipher suite. Conjunt d'algorismes que asseguren les connexions per xarxa. Inclouen algorismes d'intercanvi, xifrat i autenticació.

Confidencialitat. Propietat que assegura que sols qui estigui autoritzat tindrà accés a la informació.

Criptografia. Ciència que estudia tècniques matemàtiques relacionades amb assegurar la informació.

ECDSA. Tipus de xifrat de clau publica que fa ús de corbes el·líptiques com a algorisme per xifrar la informació.

IETF. Organisme de normes d'internet. Desenvolupa estàndards oberts.

Kanban. Mètode de gestió de projectes basat en fluxos de treball.

Handshake. Protocol d'encaixada de mans. Negociació dels paràmetres per establir una connexió xifrada.

Hash. Funció. Crea un missatge de mida variable. Un cop aplicada la funció de Hash, si es modifica el document el missatge varia.

Integritat. Propietat que assegura que una informació no ha estat alterada.

No repudi. Una informació signada per un certificat vàlid ha de ser acceptada per tothom, sense que ningú se'n pugui desdir.

RSA. Tipus de xifrat de clau publica que fa ús de la factorització de nombres enters per a xifrar la informació. Xifrat i Signatura digital.

SSH. Protocol i programa. Crea un canal segur d'accés. Proporciona accés remot.

SSL/TLS. Secure Socket Layer / Transport Layer Security. Protocol de transport de la pila de protocols, que encapsula els paquets IP i li proporciona seguretat.

RFC. Monogràfic creat per la IETF. Defineix els protocols usats en el funcionament d'Internet.

Xifrat. Acció. Un text, missatge o document xifrat no son llegibles. Proporciona confidencialitat.

12. Bibliografia

Comparando ECDAS vs RSA. Nick Nazaridis. 27/06/2018. [1]
https://www.ssl.com/es/articulo/comparando-ecdsa-vs-rsa/

Redes de Computadoras. 5ta Edition. Tanebaum | Wetherall. [2]
Ed. Pearson. Cap 1. Pag 1

Cisco CCNA 1. Wendel Odom [3]
Ed. Ciscopress.com Cap 1. Pag 29

Introducció a la criptografia. C. Pérez Solà, J. Herrera. Pag 7 [4]
©FUOC • PID 00235165

Mecanismes de protecció. Jordi Herrera Joancomartí. Pag 5. [5]
©FUOC • PID 00187027

Diffie-Hellman. Protocol Diffie-Hellman [6]
https://es.wikipedia.org/wiki/Diffie-Hellman

Criptografía con curvas elípticas. El laberinto de Falken. A. Serrano [7]
https://www.ellaberintodefalken.com/2014/06/criptografia-con-curvas-elipticas.html

Introducción a las curvas elípticas. Encuentro virtual de física y matemàtica. 2016 Gabriel Chica Reyes. [8]
Introducción a las curvas elípticas - Bing video

Criptografia de clau publica. Viquipedia. [9]
https://ca.wikipedia.org/wiki/Criptografia_de_clau_pública

Seguretat en el Comerç Electrònic. J. Joancomartí. Pag 7 - 9 [10]
© FUOC • PID_00160564

Infraestructura de clau publica. C. Pérez Solà, J. Herrera. Pag 7. [11]
©FUOC • PID_00255020

Sede electrònica. Obtención de certificados Electrónicos. [12]
https://www.sede.fnmt.gob.es/certificados/persona-fisica

X.509. Wikipedia. [13]
https://es.wikipedia.org/wiki/X.509

Problema RSA. Wikipedia [14]
https://es.wikipedia.org/wiki/Problema_RSA

RSA. Wikipedia [15]
https://es.wikipedia.org/wiki/RSA

ECDSA. Wikipedia [16]
https://es.wikipedia.org/wiki/ECDSA

Requisitos de Seguridad para TLS. Informe CCN-PYTEC nº8 . CCN [17]
https://www.ccn.cni.es/index.php/es/docman/documentos-publicos/boletines-pytec/378-pildorapytec-nov2020-seguridad-tls/file

Transport Layer Protocol. RFC 4253. January 2006 [18]
https://www.rfc-editor.org/rfc/rfc4253

Secure Shell. Wikipedia [19]
https://es.wikipedia.org/wiki/Secure_Shell

ECC Cipher Suites for TLS. RFC 4492. Mayo 2006 [20]
https://www.rfc-editor.org/rfc/rfc4492

Recommended Eliptic Curve Domain Parameters. Daniel Brown. Enero 2010 [21]
https://www.secg.org/sec2-v2.pdf

OpenSSL. Certificat #4282. NIST. Computer Security Resource Center [22]
https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4282

FileZilla. The free FTP Solution [23]
https://filezilla-project.org/

S_TIME. OpenSSL. Kit d'eines de criptografia i SSL/TLS [24]
https://www.openssl.org/docs/man1.1.1/man1/openssl-s_time.html

Handshake SSH. Russell Jones. Març – 2022. [25]
https://goteleport.com/blog/ssh-handshake-explained/

TLSv1.3. Rfc-editor.org [5]. Agost 2018 [26]
https://www.rfc-editor.org/rfc/rfc8446

TLSv1.2. Rfc-editor.org. Agost 2006 [27]
https://www.rfc-editor.org/rfc/rfc5246

13. Annexos

13.1. Instal·lació OpenSSL

Tots els certificats fets servir el aquest projecte han estat creats amb OpenSSL v3.0.5. Es denomina a si mateix com “un kit d’eines de criptografia i SSL/TLS”.

- El programari es de lliure distribució. El programa te una llicencia d’estil Apache, tothom pot aconseguir-lo i fer-lo servir amb finalitats comercials i no c comercials subjectes a condicions de llicencia senzilla
- Proporciona funcions criptogràfiques a altres paquets com OpenSSH i navegadors web per establir connexions segures.
- Permet establir als navegadors connexions SSL/TLS, i la creació de certificats digitals.
- Fa ús dels següents algorismes criptogràfics

  Per a xifrar

  `AES, DES, 3DES, Camelia, SEEC, IDEA, CAST – 128, RC2, RC4, RC5`

  Funcions HASH

  `MD2, MD4, MD5, SHA – 1, SHA – 2, MDC – 2`

  De clau publica

  RSA, DSA, Diffie-Hellman, Corbes el·líptiques
- EL programari fet servir conté un mòdul FIPS validat, i que te per numero de seguretat el 4842. “*OpenSSL FIPS Provider”* és una biblioteca de programari que proporciona una interfície de programació (API) en llenguatge C per a les aplicacions que requereixen una funcionalitat criptogràfica. [22].

Es procedeix a la descarrega des del enllaç https://www.openssl.org/source i es descarrega la darrera versió del programa per a Windows. En sol·licitat ajut ens mostra funcions usades.

13.2. Creació certificats SSL/TLS

Per aquest treball s'han creat certificats RSA i ECDSA de tipus auto signats. Se ha procedir a la creació de certificats a fi de poder comprovar-ne el rendiment.

13.2.1. Creació de certificat auto signat RSA amb la suite OpenSSL

1. Creació de la clau privada per la CA

```
openssl genrsa -des 3 -out ca.key 2048
```

2. Creació certificat per la CA

```
openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

3. Claus i certificat per el servidor

```
openssl genrsa  -out server.key 2048
openssl req -new -key server.key -out server.csr
cp server.key server.key.pass
openssl rsa -in server.key.pass -out server.key
```

4. Certificat autosignat per el per el servidor

```
openssl x509 -req -sha256 -days 365 -in server.csr -CA ca.crt -
CAkey ca.key -set_serial 01 -out server.crt
```

Llegenda

req	Permet la creació i gestió de peticions d'emissió de certificats x.509
x509	Permet la gestió de certificats x.509
genrsa	Permet la generació d'una clau privada RSA
-new	Especifica nova sol·licitud
-nodes	Especifica no xifrar la clau privada amb un password
-in	Especifica la petició de certificat a signar
-out	Especifica el fitxer de sortida. Nom i format
-signkey	Es signa la petició amb la CA existent

Tipus d'arxiu

*.csr	Sol·licitud de signatura d'un certificat
*.key	Clau privada
*.crt	Certificat
*.pem	Certificat usat per Apache. Format Base64
*.der	Arxiu de certificats en format binari. Usat per Java
*.P7B	Arxiu ce certificar + cadena. Sense la clau privada
*.p12	Arxiu de certificat en format binari. Conte certificat + clau privada

13.2.2. Creació de certificat auto signat ECDSA amb la suite OpenSSL

1. Creació de la clau privada per la CA

```
openssl ecparam -genkey -name secp224k1 -out pkeyecc224.pem
```

2. Creació de la sol·licitud de la signatura

```
openssl req – new -sha256 -key pkeyecc224.pem  -out 224.csr
```

3. Creació del certificat auto signat

```
openssl req -x509 -days 360 -key  pkeyecc224.pem -in 224csr.pem
-out certECC224.pem
```

Llegenda

ecparam	Manipulació generació de claus de corba el·líptica
list_curves	Mostra els noms de les corbes implementades
genkey	Generació de claus privades EC
x509	Generació d'un certificat autosignats
-key	Especifica arxiu d'on llegir la clau privada
-new	Genera nova petició de certificat
-name	Nom de la corba a fer servir
-in	Fitxer per llegir
-genparam	generació d'un arxiu de paràmetres en lloc de la clau privada
-pkeyopt	Especifica longitud de la clau
req	Per processar i generar un arxiu CSR
-signkey	Es signa la petició amb la CA existent
-noout	No demana password
-out	Especifica el nom de sortida

13.2.3 Per la comparativa es crearan els següents certificats

RSA	ECDSA	Corba
rsa2048.pem	---	---
rsa3072.pem	ecc256.pem	secp256r1
rsa7680.pem	ecc384.pem	secp384r1
rsa15360.pem	ecc521.pem	secp521r1

13.3. OpenSSL s_time

L'ordre *s_time* sol·licita una pàgina al servidor i mesura el temps de realitzar aquesta petició. D'aquesta ordre en podem obtenir el nombre de connexions en un temps determinat, la quantitat de dades transferida i un càlcul del temps mitjà dedicat a cada connexió [25].

Comanda captura descarrega web

```
OpenSSL s_time -connect vicvic82.ddns.net -www / > temps.txt
```

Comanda captura Handshake

```
OpenSSL s_time -connect vicvic82.ddns.net > tempsahand.txt
```

Opcions openssl

`-connect host:port`	Especifica host i port on connectar
`-www /`	Crida GET. Obté la pàgina index.html del servidor
`> temps.txt`	Redirigeix sortida cap un fitxer.txt

```
Openssl s_time -connect vicvic82.ddns.net:443 -www / > 10/10_rsa_1024.txt
Collecting connection statistics for 30 seconds
**********************************************************************
287 connections in 1.50s; 191.33 connections/user sec, bytes read 29077979
287 connections in 31 real seconds, 101317 bytes read per connection

Now timing with session id reuse.
starting
rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr
306 connections in 0.97s; 315.87 connections/user sec, bytes read 31003002
306 connections in 31 real seconds, 101317 bytes read per connection
```

```
Openssl s_time -connect vicvic82.ddns.net:443 > 10/10_rsa_1024_hand.txt
Collecting connection statistics for 30 seconds
**********************************************************************
2614 connections in 7.25s; 360.55 connections/user sec, bytes read 0
2614 connections in 31 real seconds, 0 bytes read per connection

Now timing with session id reuse.
starting
**********************************************************************2536
connections in 5.88s; 431.66 connections/user sec, bytes read 0
2536 connections in 31 real seconds, 0 bytes read per connection
```

13.4. Creació certificats SSH

Un certificat SSH no té res a veure amb un certificat x.509. SSH ha creat un Standard propi de certificat per a proporcionar autenticació i xifrat. Aquests certificats contenen una clau púbica, informació d'identitat i validesa. Existeixen dos tipus de certificats, els d'usuari i els de host. Per aquest test hem creat certificats de host. Ha calgut instal·lar OpenSSH.

Configuració de certificats de Host

1. Creació d'un certificat

```
sudo ssh-keygen -t tipus -b mida -C "comentari" -f cert
```

2. Configuració de l'arxiu *sshd_config. Conté tota la imformació per configurar el servidor ssh*

```
Sudo nano /etc/ssh/sshd_config
```

```
Port 22
   ListenAddress 0.0.0.0
   #certificats per el test
   HostKey /etc/ssh/rsa_2048
   #HostKey /etc/ssh/rsa_3072
   #HostKey /etc/ssh/rsa_7680
   #HostKey /etc/ssh/rsa_15360
   #HostKey /etc/ssh/ecdsa_256
   #HostKey /etc/ssh/ecdsa_384
   #HostKey /etc/ssh/ecdsa_521
```

3. Reinici servei ssh

```
sudo service sshd restart
```

13.5. Recomanacions seguretat FIPS 140 – 2

FIPS 140 -2 és un estàndard de seguretat del govern dels Estats Units

Estableix 4 nivells de seguretat segons quin tipus de producte es tracti.

- Nivell 1. Ús de xifrat software. Requisits basics de seguretat
- Nivell 2. Requeriment d'autenticació basat en càrrec del usuari
- Nivell 3. Resistència a la intrusió física.
- Nivell 4. Protecció avançada. Destinat a entorns no segurs

Fa ús dels següents protocols

- Ús del algorisme de xifrat AES, 3DES
- Ús de l'algorisme de signatura digital DSA, RSA i ECDSA
- Ús d'algorismes d'autenticació AES, 3DES i HMAC
- Ús d'algorismes de Hash. SHA-1, SHA-256 o SHA-384

13.6. Recomanacions seguretat NSA

La NSA recomana l'ús del seu estàndard de xifrat anomenat Suite B per establir comunicacions segures.

- Recomanació d'us de l'algorisme de xifrat AES
- Ús de l'algorisme d'intercanvi de claus ECDH (Eliptic Curve Diffie-Hellman)
- Ús de l'algorisme de signatura digital ECDSA
- Ús d'algorismes de Hash. SHA-256 o SHA-384

13.7. Recomanacions Centre Criptològic Nacional. CCN-PYTEC

- Ús de TLS versió 1.2 o superior
- Ús de VPN admeses en catàleg de productes de seguretat TIC
- No es recomana l'ús de claus pre compartides (PSK)
- Recomanat mètodes de Key Exchange DHE o ECDHE, ja que proporcionen Forward Secrecy (FS)
- Recomanació de que la implementació TLS 1.2 accepti l'ús de corbes el·líptiques.
- En cas de fer servir DHE cal que TLS 1.2 faci ús de la extensió Supported_Groups

- Es recomana fer servir autenticació per mig de certificats de clau pública x.509v3, RSA o ECDSA
- Es recomana l'ús de la extensió signature_algoritms per indicar els algorismes de signatura suportats. Les funcions de Hash han de ser iguals o superiors a SHA-256
- Recomanació ús de claus RSA de 3072 bits y claus ECC de 256 bits
- TLS recomana l'algorisme de xifrat AES
- TLS recomana ´'us de SHA-2 o superior
- Cipher-suites recomanades en TLS 1.2
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS 1.3 recomana fer ús de qualsevol de les Cipher-suites ofertes per aquest
- Recomanació d'us de certificats emesos per una CA
- Cal que el certificat sigui validat. Estat de revocació i cadena de certificats. Ús d'una implementació que compleixi la norma RFC 5280
- Es rebutjarà una connexió TLS si el certificat no superi la validació, o no pugui ser verificat l'estat de revocació del certificat

Font:
https://www.ccn.cni.es/index.php/es/docman/documentos-publicos/boletines-pytec/378-pildorapytec-nov2020-seguridad-tls/file

13.8. Manual instal·lació servidor web Apache

Apache es un servidor web de codi obert i gratuït desenvolupat i mantingut per una comunitat d'usuaris anomenada "Apache Software Foundation". Es fa us de la distribució Apache HTTP Server 2.4 proporcionada per part de Haus Distribution. Es descarrega des de la web i s'instal·la Apache com a servei en Windows.

Instal·lació Apache 2.4

- Executem una consola d'administració amb drets d'administrador
- Ens situem en el directori c:\Apache24\bin
- Instal·lem el servei amb la instrucció: `httpd.exe -k install`
- Iniciem Apache amb la instrucció: `httpd.exe -k start`

Directoris Apache

La instal·lació crea diferents directoris, cadascun d'ells realitza una feina concreta.

- *El directori bin*
  Conté els arxius executables del programa
- *El directori cgi-bin*
  Permet la execució d'scripts basats en Perl, cgi,...
- Directori conf
  Conté els arxius de configuració primaris
  - Arxiu httpd.conf
    Configuració principal d'Apache
  - Carpeta SSl
    Emmagatzema els certificats fet servir per el servidor Apache
  - Carpeta extra:
    Altres config
  - Carpeta original
  - Httpd-ssl:
    Informació accés SSL
- *Directori error*
  Missatges configurats d'error
- *Directori htdocs*
  Directori principal. Inici del web
- *Directori icons*

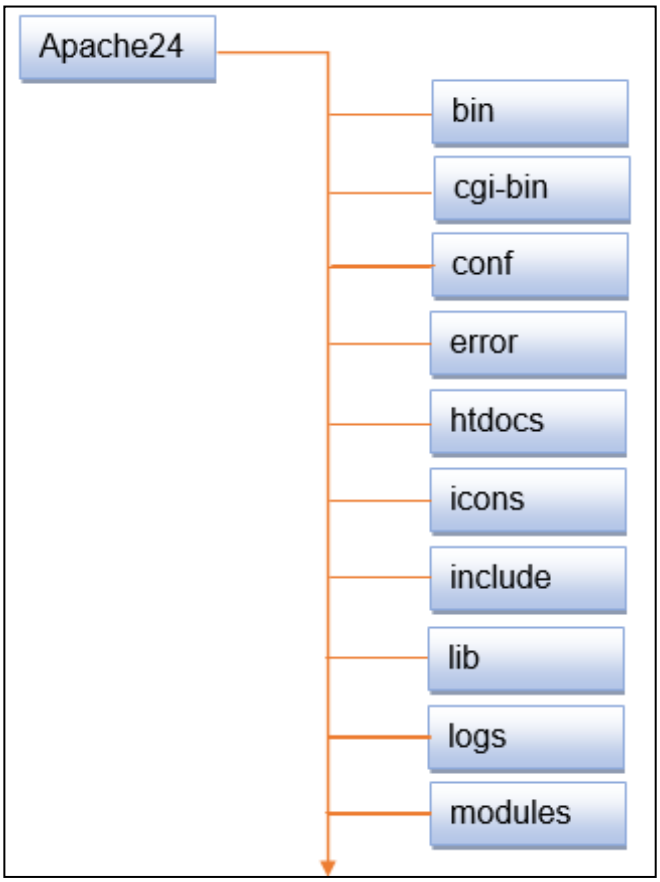
Conté icons del programa

- Directori include
- Directori lib

  Conté llibreries del programa
- Directori logs

  Mostra logs creats
- Directori modules

  Conté mòduls del programa



Arxius de configuració principals

- Httpd.conf

Define SRVROOT "/Apache24"	defineix directori arrel
Define ENABLE_TLS1.3 "Yes"	Habilita suport TLS
Listen 80	Escolta per el port 80
ServerName localhost:80	Conf Hostname + port
include conf/extra/httpd-ssl.conf	Per habilitar configuració ssl

- Httpd-ssl

Listen 443	Defineix port

ServerName www.example.com:443
SSLEngine on Habilita SSL
SSLCipherSuite Defineix mòduls acceptats
Certificats
SSLCertificateFile "${SRVROOT}/conf/public.crt"
SSLCertificateKeyFile "${SRVROOT}/conf/private.key"
#SSLCACertificatePath "${SRVROOT}/conf/ssl.crt"

13.8.1. Configuració per acceptar connexions SSL

Es duen a terme les següents configuracions per acceptar connexions segures. Per aquesta prova s'ha comprat el domini vicvic82.ddns.net i creat un certificat signat per.

- Copiar els certificats al directori
- Obrir l'arxiu https.conf i des comentar les següents línies. #

  #LoadModule ssl_module modules/mod_ssl.so

  #Include conf/extra/httpd-ssl.conf
- Obrir l'arxiu conf/extra/httpd-ssl.conf i configurar les següents línies

  ServerName vicvic82.ddns.net:443

  ServerAdmin vicvic82@gmail.com
- Configurar el fitxer per que vagi a buscar els certificats

  # Server Public Certificate:

  SSLCertificateFile "${SRVROOT}/conf/vicvic82_ddns_net.crt"

  # Server Private Key:

  SSLCertificateKeyFile "${SRVROOT}/conf/vicvic82_ddns_net.key"

  # Certificate Authority (CA):

  SSLCACertificatePath "${SRVROOT}/conf/vicvic82_ddns_net.pem"
- Reiniciar el servei Apache per que agafi la configuració

  apachectl stop

  apachectl startssl

Font:

https://httpd.apache.org/docs/2.4/es/

13.9. Manual instal·lació servidor FTP FileZilla

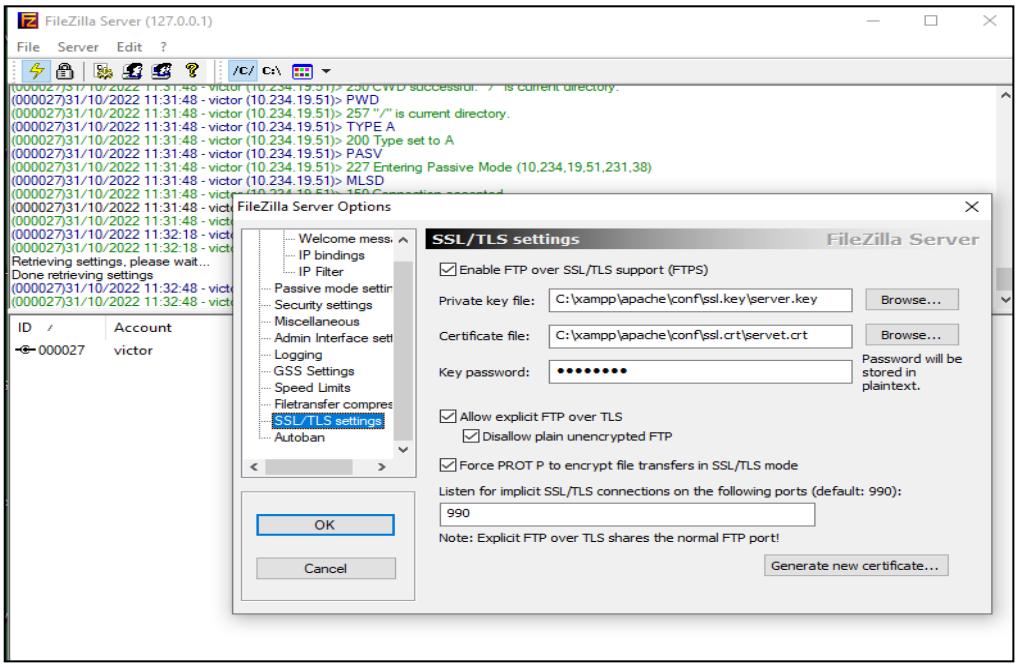
FileZilla es un programari de codi obert que s'encarrega de fer transferències d'arxius a través d'internet. Es fa us de la distribució FileZilla Server 1.5.1 proporcionada per part de filezilla-project.org.

Característiques

- Xifrat FTPS, SFTP
- Compatible amb IPv6
- Interfície gràfica
- Versió de Servidor i Client
- Suport HTTP/1.1
- Suport fitxers > 4gb

La seva instal·lació permet

- La gestió d'usuaris
- Gestiona directoris compartits
- Limitar velocitat de descarrega segons usuari



Configuració FileZilla Server per establir connexió segura

13.10. Manual instal·lació servidor SSH

Per a dur a terme la instal·lació d'un servidor SSH s'ha instal·lat *Linux Ubuntu 22.04* sota un entorn virtual. Un cop instal·lada l màquina, assignada una ip fixa i configurat el Firewall per acceptar connexions SSH, s'ha dut a terme la instal·lació del servei OpenSSH com a servidor. Hem configurat l'accés amb usuari i pwd.

Instal·lació SSH

Podem instal·lar SSH com a servei servidor per mig de la comanda

```
sudo apt install openssh-server
```

La següent comanda inicia el servei

```
sudo /etc/init.d/ssh start
```

La següent comanda para el servei

```
sudo /etc/init.d/ssh stop
```

La següent comanda reinicia el servei

```
sudo /etc/init.d/ssh restart
```

Configuració servidor

Guardarem les claus creades al següent directori

```
HostKey /etc/ssh/nom_clau_key
```

La següent comanda permet la configuració de diferents paràmetres

```
sudo nano /etc/ssh/sshd_config
```

Algunes de les configuracions aplicades en l'arxiu sshd_config

Port 22

Protocol 2

PermitRootLogin no

ListenAddress 192.168.1.0/24

AllowUsers: victor@vicvic82.ddns.net

13.11. Captura de dades WEB. Web 10 Kb

Protocol: TLSv1.2 / Cipher suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

	10 Mbps			100 Mbps		
	Chrome	Edge	Firefox	Chrome	Edge	Firefox
Sense xifrat	0,28	0,22	0,24	0.22	0,23	0,20
RSA_1024	0,40	0,36	0,25	0,33	0,29	0,25
RSA_2048	0,33	0,34	0,30	0,30	0,25	0,24
RSA_3072	0,42	0,30	0,34	0,28	0,59	0,30
RSA_7680	0,77	0,80	0,49	0,66	0,69	0,40
RSA_15360	3,55	4,19	3,68	3,78	3,52	3,53

Taula 3.2. Web 10 Kb / TLSv1.2 / RSA

	10 Mbps			100 Mbps		
	Chrome	Edge	Firefox	Chrome	Edge	Firefox
ECDSA_160	---	---	---	---	---	---
ECDSA_224	---	---	---	---	---	---
ECDSA_256	0,33	0,35	0,30	0,23	0,28	0,16
ECDSA_384	0,35	0,39	0,30	0,25	0,34	0,20
ECDSA_521	---	---	1,47	---	---	1,26

Taula 3.3. Web 10 Kb / TLSv1.2 / ECDSA

Protocol: TLSv1.3

	10 Mbps			100 Mbps		
	Chrome	Edge	Firefox	Chrome	Edge	Firefox
Sense xifrat	0,28	0,22	0,24	0,22	0,23	0,20
RSA_1024	0,39	0,35	0,21	0,26	0,27	0,24
RSA_2048	0,35	0,32	0,20	0,25	0,26	0,22
RSA_3072	0,41	0,44	0,43	0,31	0,34	0,40
RSA_7680	1,58	1,43	0,49	0,68	0,64	0,61
RSA_15360	3,54	3,45	3,42	3,26	3,39	3,24

Taula 3.4. Web 10 Kb / TLSv1.3 / RSA

	10 Mbps			100 Mbps		
	Chrome	Edge	Firefox	Chrome	Edge	Firefox
ECDSA_160	---	---	---	---	---	---
ECDSA_224	---	---	---	---	---	---
ECDSA_256	0,31	0,33	0,30	0,18	0,24	0,21
ECDSA_384	0,38	0,35	0,32	0,26	0,27	0,24
ECDSA_521	---	---	1,63	---	---	1,46

Taula 3.5. Web 10 Kb / TLSv1.3 / ECDSA

13.12. Dades Obtingudes amb OpenSSL s_time

Velocitat: xarxa 10Mbps

Web: 10 Kb

Cipher suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)

	Connections / user sec		Handshake / user sec	
	TLSv1.2	**TLSv1.3**	**TLSv1.2**	**TLSv1.3**
RSA_1024	191.33	181.33	360.55	347.73
RSA_2048	134.07	165.73	314.11	330.37
RSA_3076	152.57	101.05	156.36	202.44
RSA_7680	151.77	136.42	149.76	131.49
RSA_15360	64.00	51.20	90.67	68.00

Dades RSA. SSL/TLS. 10Mbps/10Kb

	Connections / user sec		Handshake / user sec	
	TLSv1.2	**TLSv1.3**	**TLSv1.2**	**TLSv1.3**
ECDSA_160	---	---	---	---
ECDSA_224	---	---	---	---
ECDSA_256	233.60	193.35	431.06	455.65
ECDSA_384	115.76	132.23	271.30	266.49
ECDSA_521	100.16	100.63	197.04	201.96

Dades ECDSA. SSL/TLS. 10Mbps/10Kb

13.13. Captura de dades FTPS

Per a dur a terme la comparativa, hem efectuat la descarrega d'un fitxer de 55,840 Mbytes, a la vegada que capturàvem amb WireShark la durada de la descarrega. Hem anat alternat els certificats fins a obtenir dades fiables de totes les longituds de clau. Hem seleccionat diferents velocitats de xarxa a fi de poder comparar una descàrrega des d'internet a diferents velocitats. Tanmateix hem volgut estudiar el temps de descarrega total, i el temps que durava el procés de *Handshake.*

	10 Mbps		100 Mbps	
	Connexió	Descarrega	Connexió	Descarrega
Sense xifrar	0,111	56,029	0,106	55,924
RSA_1024	0,303	60,605	0,219	56,083
RSA_2048	0,397	56,902	0,216	56,248
RSA_3072	0,384	58,326	0,221	56,367
RSA_7680	0,814	59,962	0,366	56,014
RSA_15364	1,623	62,016	1,580	56,439

Captura de dades FTPS amb certificats RSA

	10 Mbps		100 Mbps	
	Connexió	Descarrega	Connexió	Descarrega
Sense xifrar	0,111	56,029	0,106	55,924
ECDSA_160	---	---	---	---
ECDSA_224	---	---	---	---
ECDSA_256	0,212	56,051	0,226	56,032
ECDSA_384	0,318	57,032	0,211	56,034
ECDSA_521	0,324	57,042	0,217	56,026

Captura de dades FTPS amb certificats ECDSA

13.14. Captura de dades SFTP

	10 Mbps		100 Mbps	
	Connexió	Descarrega	Connexió	Descarrega
RSA_1024	---	---	---	---
RSA_2048	0.454	56.201	0.454	30.545
RSA_3072	0.454	58.824	0.454	56.042
RSA_7680	0.524	61.236	0.537	56.351
RSA_15364	1.325	64.217	1.286	57.131

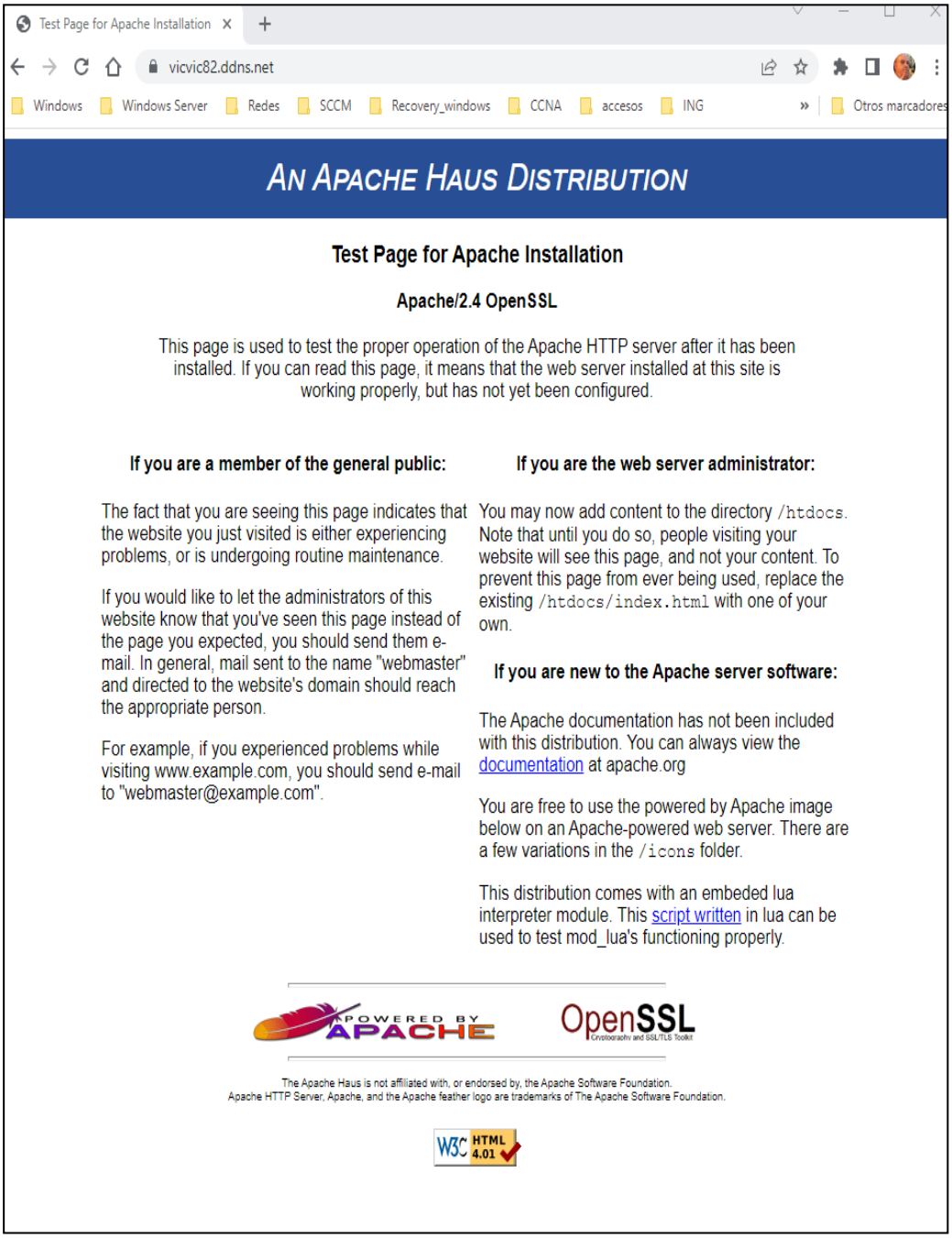
Captura de dades SFTP amb certificats RSA

	10 Mbps		100 Mbps	
	Connexió	Descarrega	Connexió	Descarrega
ECDSA_160	---	---	---	---
ECDSA_224	---	---	---	---
ECDSA_256	0.466	56.291	0.538	56.258
ECDSA_384	0.469	56.327	0.412	56.261
ECDSA_521	0.598	56.361	0.424	56.395

Captura de dades SFTP amb certificats ECDSA

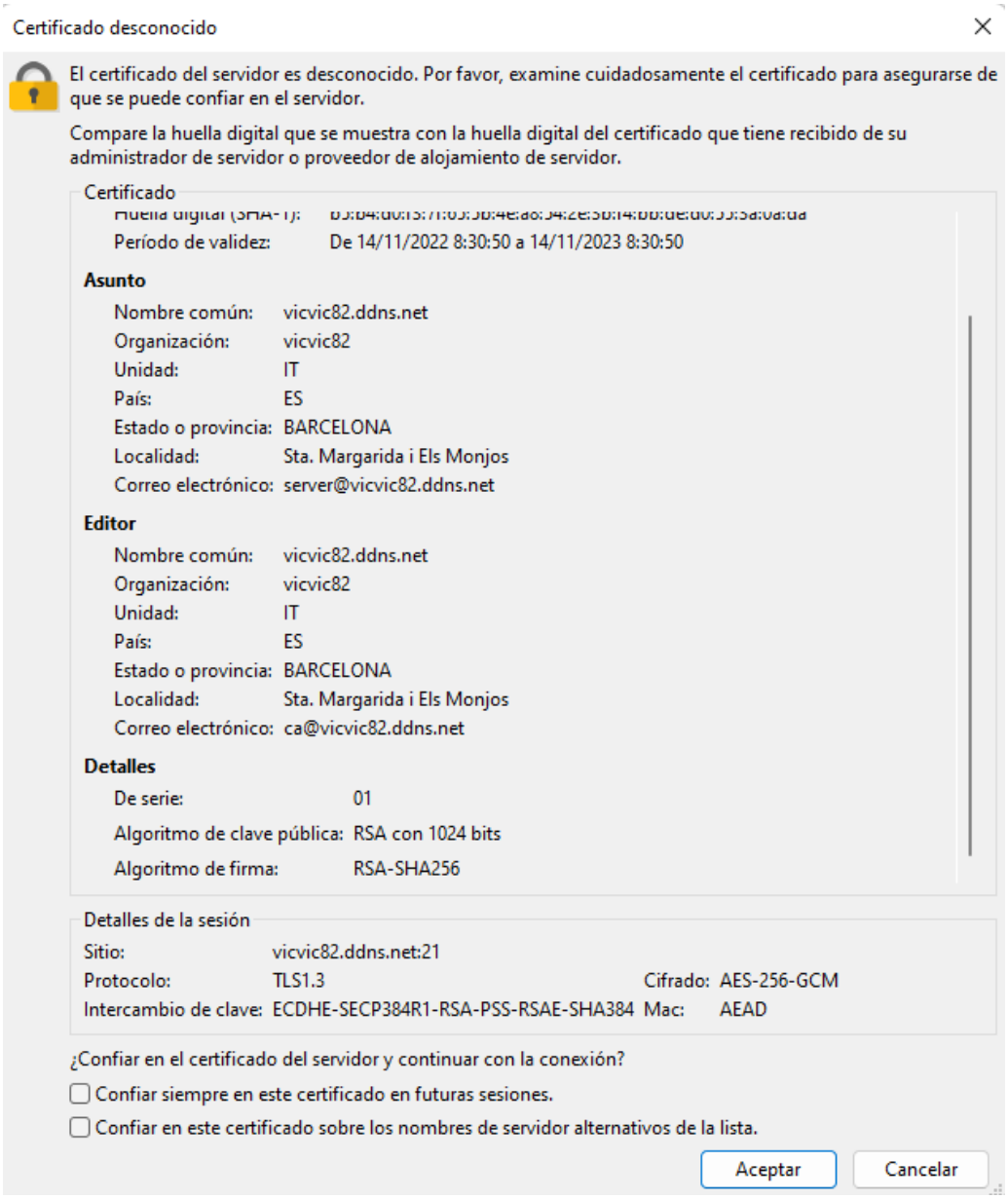
13.15. Accés HTTPS

Exercici 1: se ha comprat el domini vicvic82.ddns.net i creat un certificat validat per TrustCor. Configurat DNS dinàmic apuntant al equip que fa de servidor web



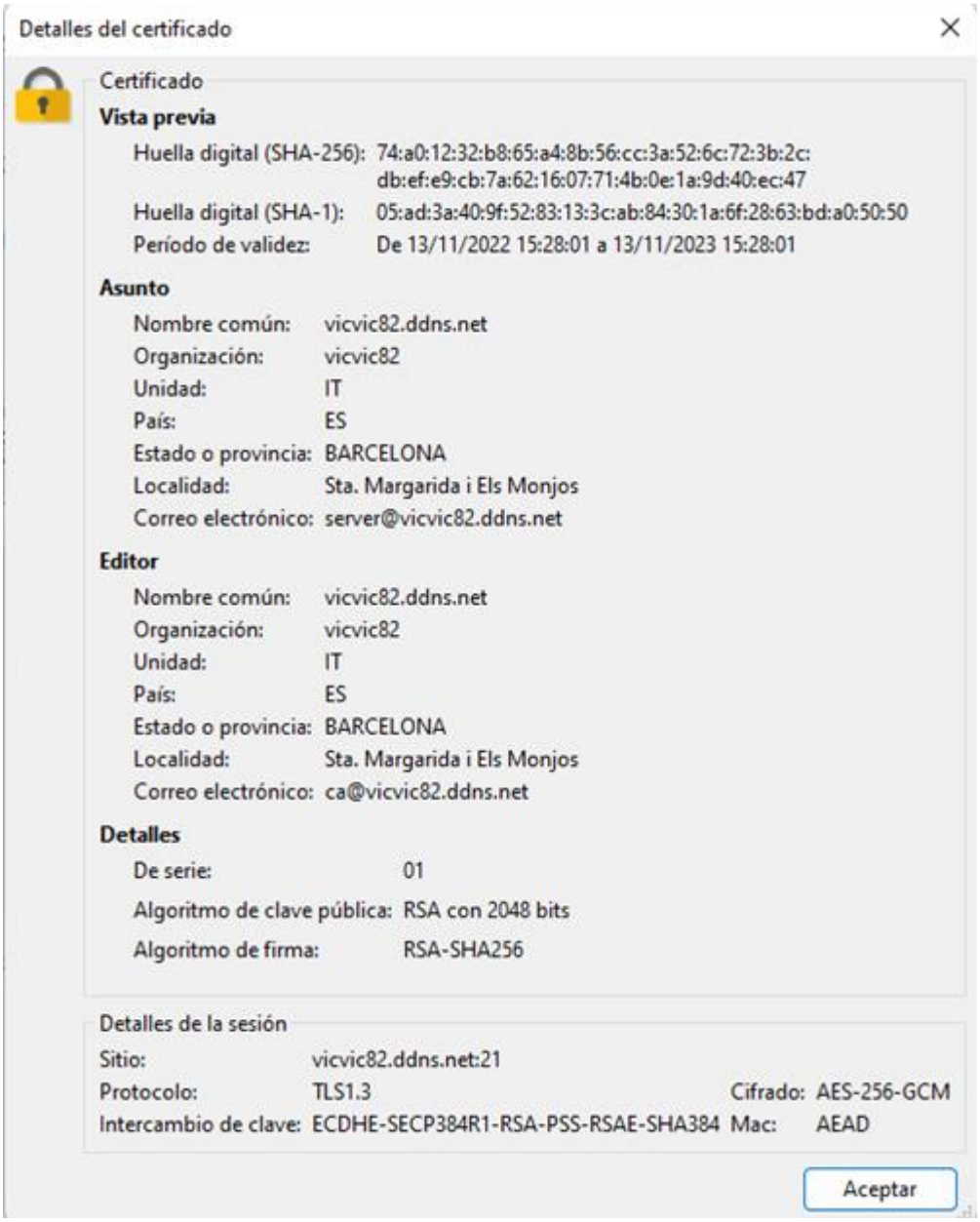

Visor de certificados: vicvic82.ddns.net
General
Detalles
Enviado a
Nombre común (CN) vicvic82.ddns.net
Organización (O) <No incluido en el certificado>
Unidad organizativa (OU) <No incluido en el certificado>
Emitido por
Nombre común (CN) TrustCor DV SSL CA - G2 - RSA
Organización (O) TrustCor Systems S. de R.L.
Unidad organizativa (OU) <No incluido en el certificado>
Período de validez
Emitido el lunes, 24 de octubre de 2022, 23:18:17
Vencimiento el martes, 24 de octubre de 2023, 23:18:16
Huellas digitales
Huella digital SHA-256 F8 6A C7 77 18 C6 2A DB 94 B1 28 7E 02 BB FC DD 20 FA 57 40 97 71 01 D6 94 2C 8E 50 39 51 B0 D0
Huella digital SHA-1 D6 20 8A 21 55 4E 23 82 C0 5C 52 CE 85 73 BC B3 7C 8F 7F 47

Exercici 2: Es procedeix a la creació dels diferents certificats auto signats SSL/TLS creats amb OpenSSL. En mostrem el certificat obtingut.

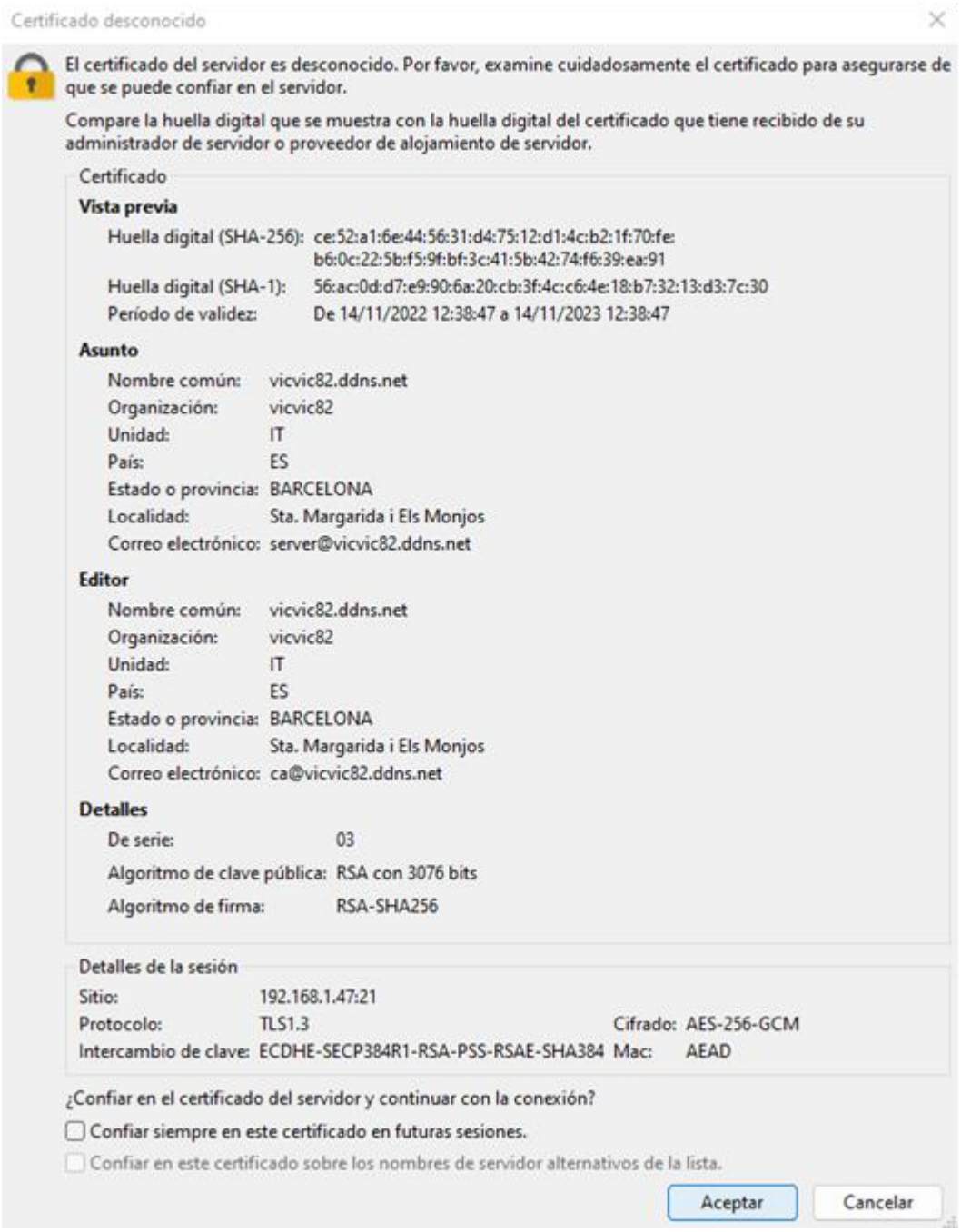


Mostra certificat del servidor FTPS

Podem observar com ens proporciona informació de la versió de protocol TLS establerta, així com la cipher suite triada per establir l'intercanvi de claus, algorisme de xifrat i el protocol usat per a proporcionar autenticació.

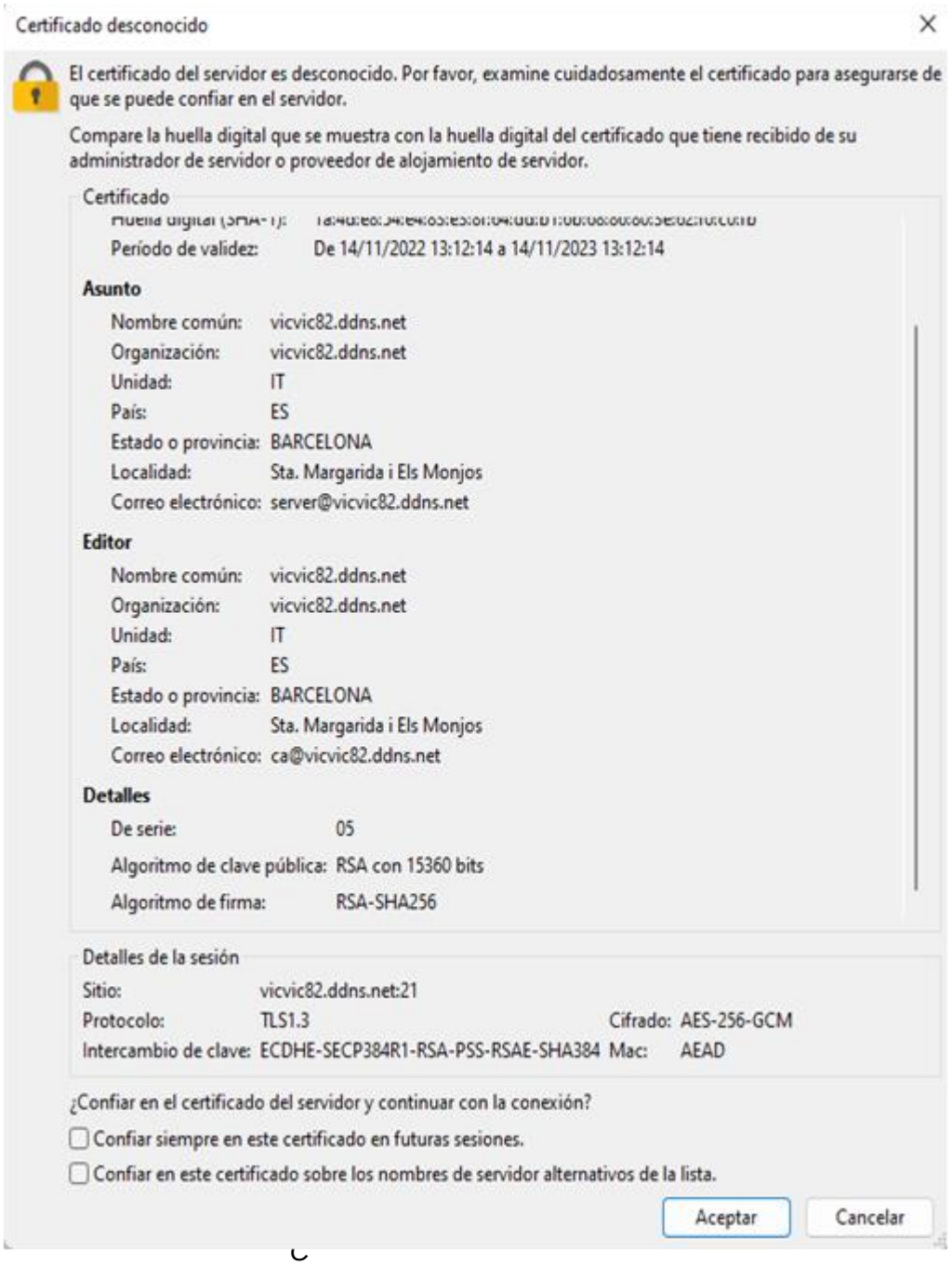


Certificat RSA_2048 bits



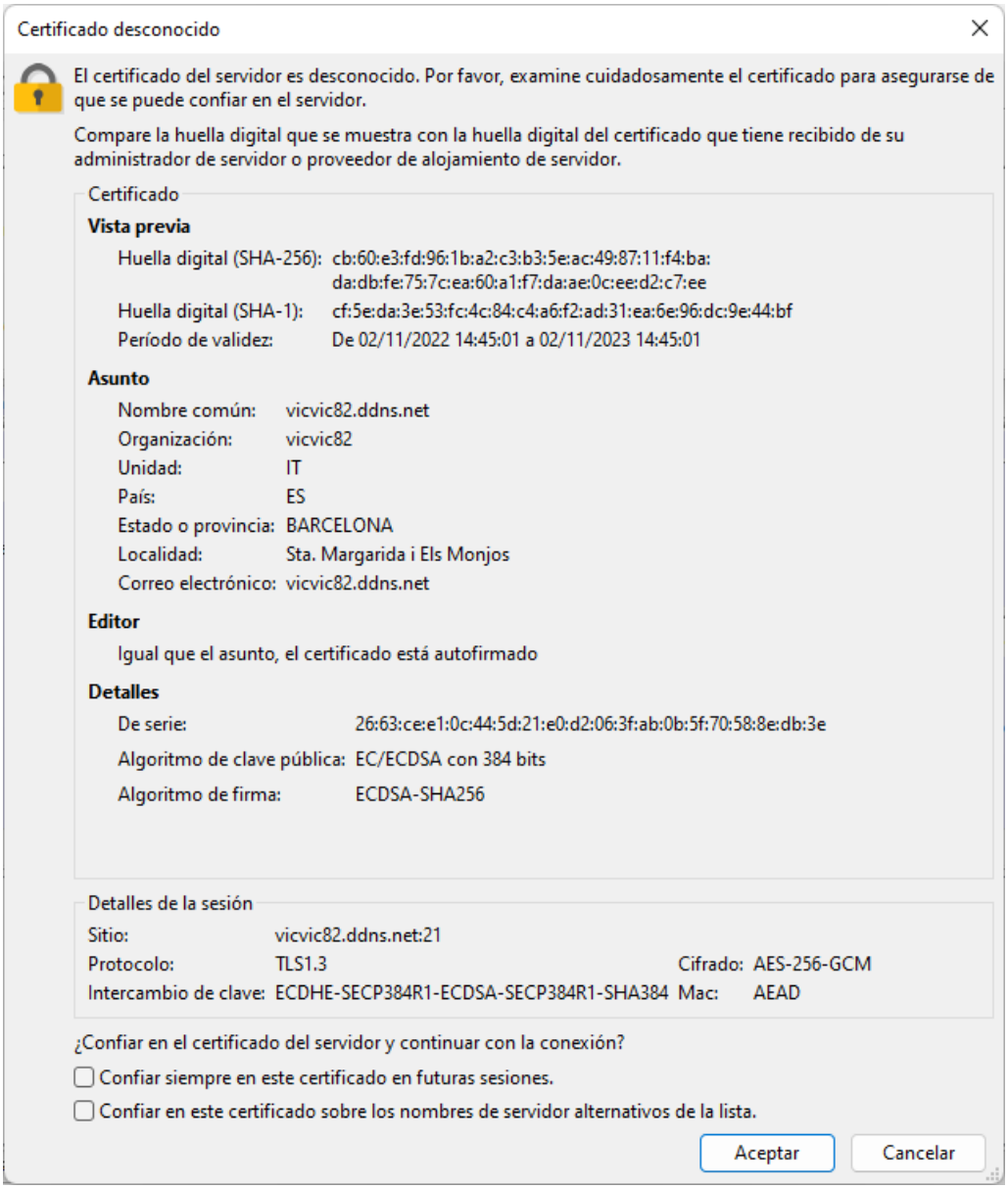
Certificat RSA_3076 bits

Certificat RSA_7680 bits



Certificat RSA_15360 bits

Certificat ECDSA_256 bits



Certificat ECDSA_384 bits

Certificat ECDSA_528 bits

13.16. Captura dades HTTPS

Des d'un client web hem establer connexió amb el servidor. Hem executat el programa WireShark i hem capturat el procés d'inici de sessió; això ens ha proporcionat dos informacions molt valuoses en aquest estudi, i que de fet s'han convertit en part clau. Estem parlant d'esbrinar com es produeix el inici de sessió, Handshake, i després hem obtingut el temps total del establiment de la sessió capturant el timestamp del darrer paquet.

Mostra captura WireShark. TLSv1.2

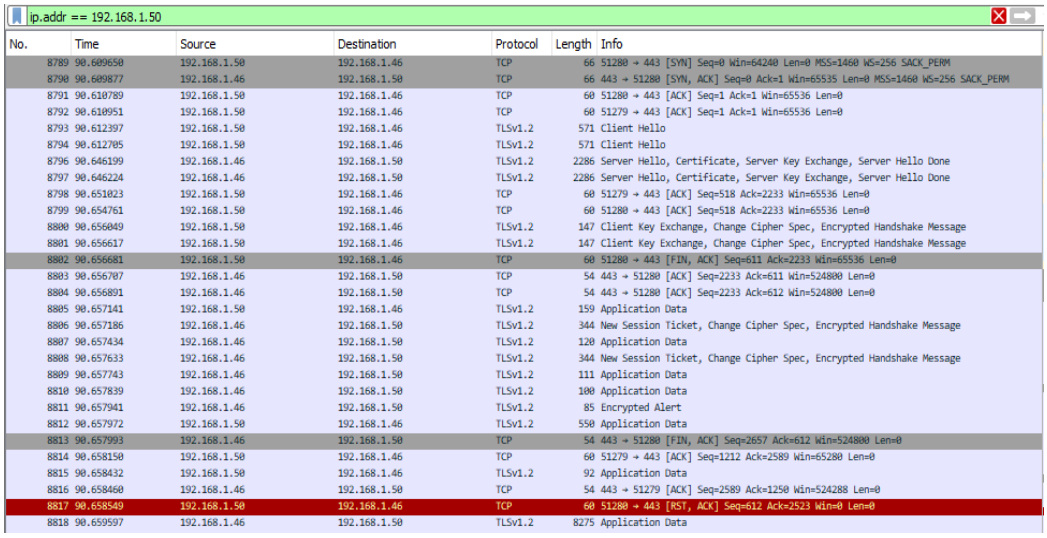
Velocitat: xarxa 10Mbps

Web: 10 Kb

Protocol: TLSv1.2

Claus: RSA_2048

Cipher suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)



Captura connexió TLSv1.2

A continuació mostrem el timestamp del darrer paquet, ens informa de que la pàgina web ha necessitat 6,076 segons en ser carregada.

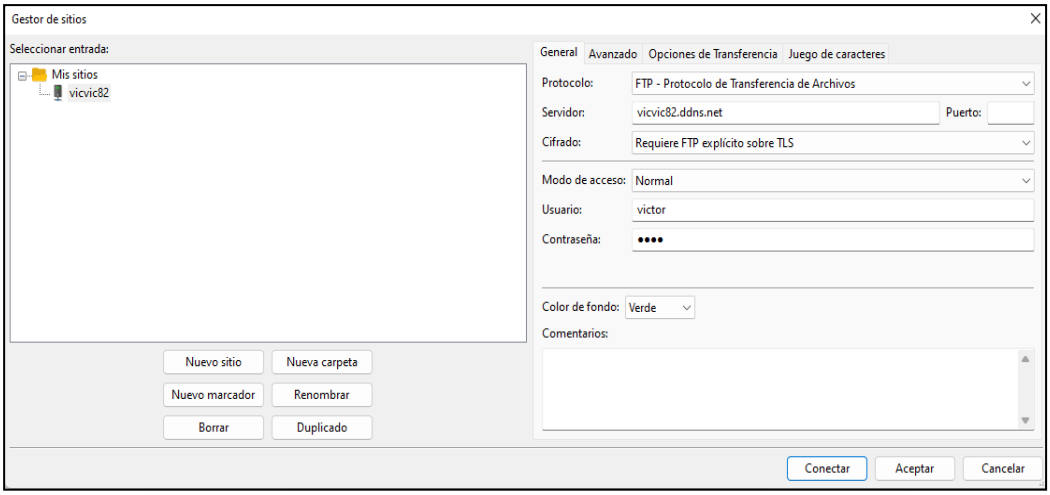
```
[Timestamps]
    [Time since first frame in this TCP stream: 6.076372000 seconds]
    [Time since previous frame in this TCP stream: 0.000153000 seconds]
```

Mentre que el procés de Handshake s'ha realitzat amb 0,048 segons

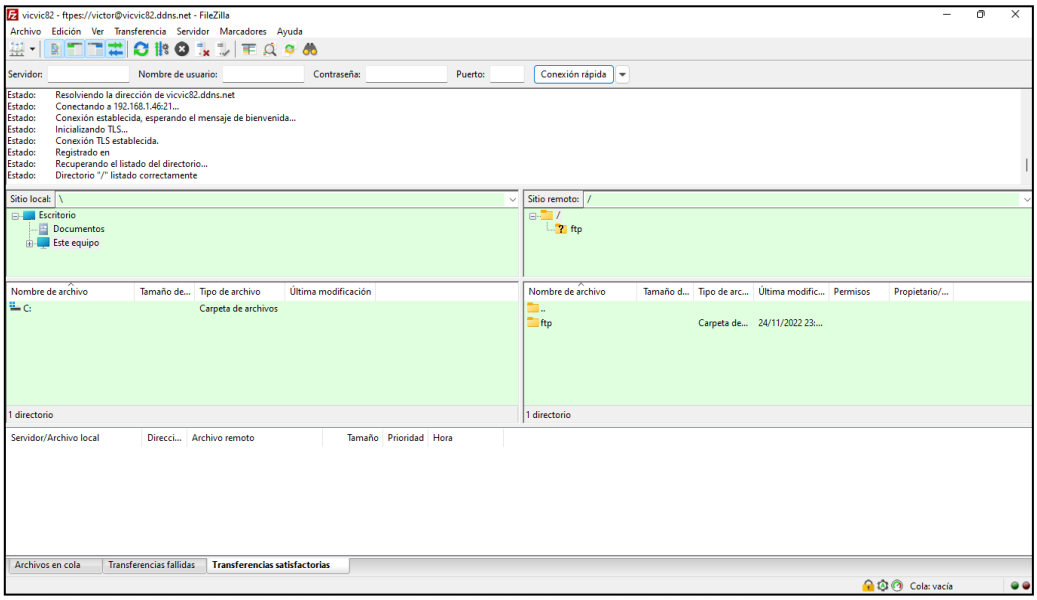
```
[Timestamps]
    [Time since first frame in this TCP stream: 0.047983000 seconds]
    [Time since previous frame in this TCP stream: 0.000742000 seconds]
```

13.17. Captura dades FTPS

Des d'un client FTP hem establer connexió amb el servidor. Hem executat el programa WireShark i hem capturat el procés d'inici de sessió; aquest fet ens ha permès de conèixer el procés de Handshake, a la vegada que obteníem el temps que dura la descarrega d'un fitxer de 55,840 Mbytes .

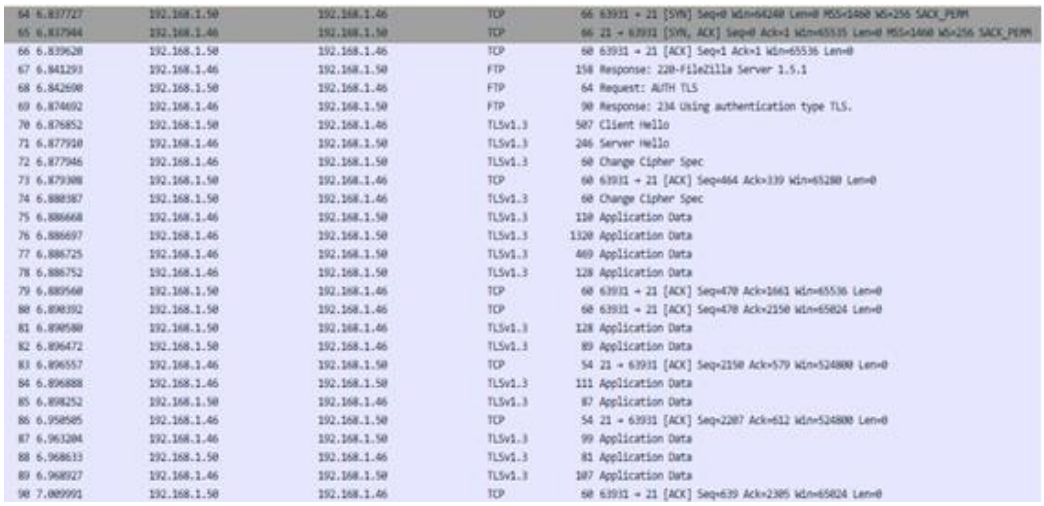


Configuració accés FTPS explícit



Establiment connexió FTPS explicita

Abans d'executar la connexió, executem l'analitzador de xarxa WireShark per a obtenir el temps de connexió al servidor.



Captura inici de sessió FTPS

El servidor envia una resposta de Benvingut al servidor establint un inici de connexió FTP.

```
File Transfer Protocol (FTP)
  220-FileZilla Server 1.5.1\r\n
        Response code: Service ready for new user (220)
        Response arg: FileZilla Server 1.5.1
    220-Please visit https://filezilla-project.org/\r\n
    220 Benvingut al servidor\r\n
```

Missatge Benvinguda al Servidor

El client realitza una consulta sobre com establir connexió,

```
File Transfer Protocol (FTP)
  AUTH TLS\r\n
      Request command: AUTH
      Request arg: TLS
```

Missatge requerint TLS

cosa que el servidor respon amb el missatge de "Using Autentication type TLS".

```
File Transfer Protocol (FTP)
  234 Using authentication type TLS.\r\n
        Response code: Security data exchange complete (234)
        Response arg: Using authentication type TLS.
```

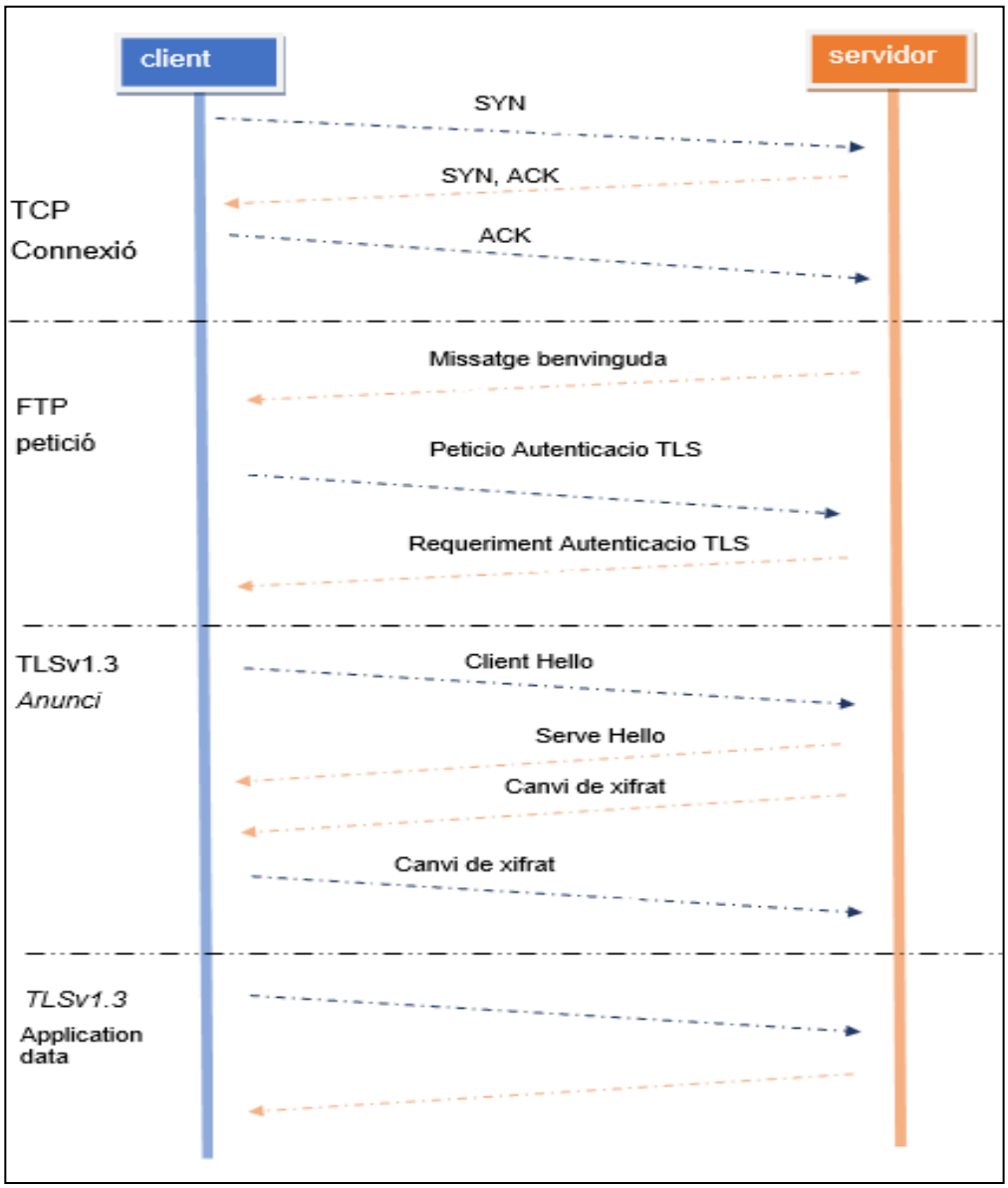
Missatge requeriment Autenticació TLS

A conseqüència de tot això s'inicia el procés de Handshake. El client envia un missatge de *Server Hello* oferint informació el tipus de petició, de les versions de TLS suportades, cipher suites suportades, algorismes

de signatura suportats, algorismes de Hash suportats, clau compartida i mètodes d'establiment de xifrat.

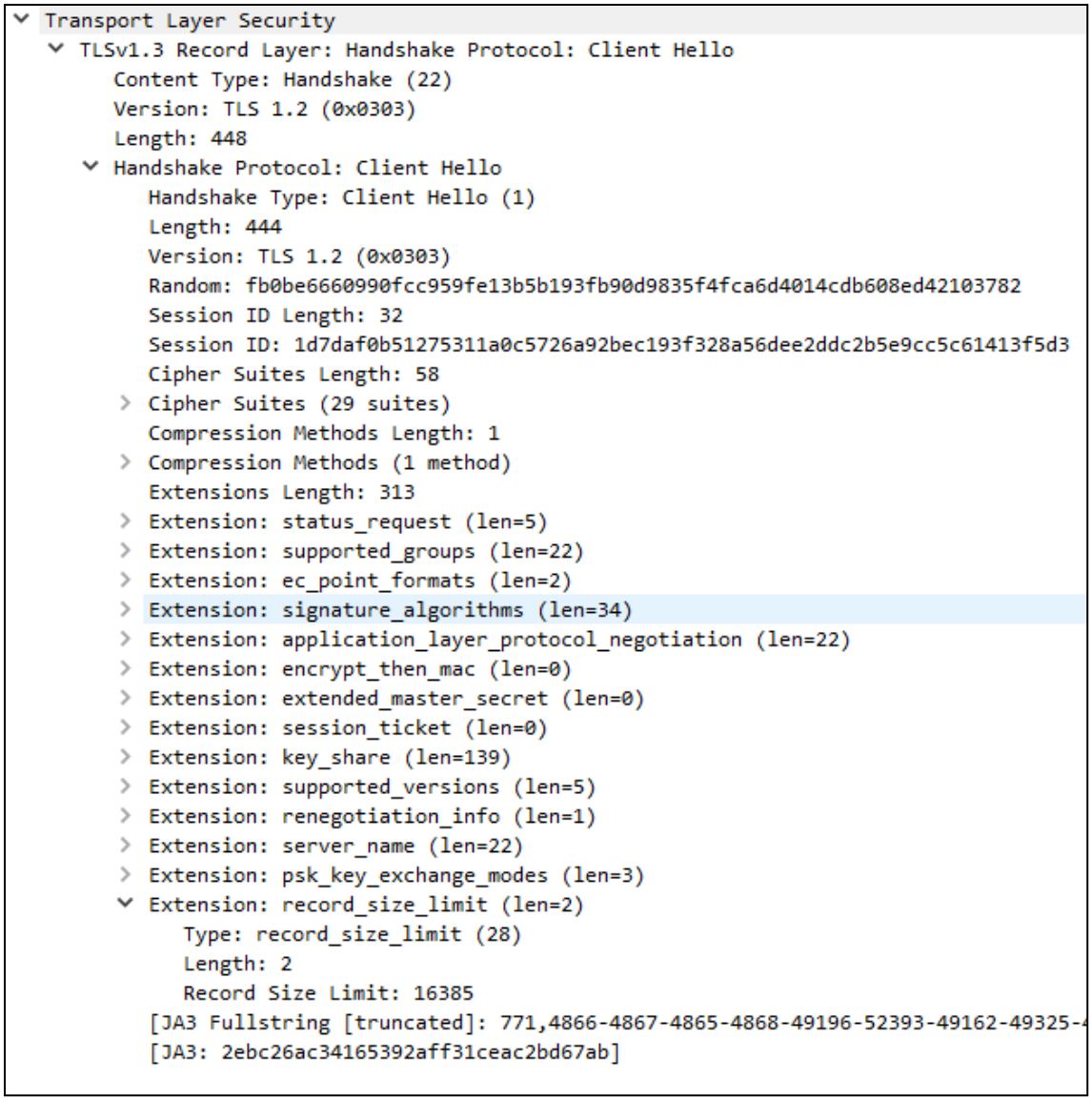
El servidor respon amb un missatge de Server Hello informant de versions TLS suportades, clau compartida i mètodes d'intercanvi de clau, mostra ja les versió de cipher suite a usar, algorismes de signatura, xifrat, autoritats de certificat. El missatge finalitza proporcionant autenticació amb l'enviament de la clau publica del servidor.

Tots dos s'envien missatges de canvi de xifrat, i el servidor genera un tiquet de sessió. A partir d'ara, tota la informació ja s'envia xifrada.



Procés Handshake TLSv1.3

El *Handshake* s'inicia amb l'enviament d'un Missatge *Client Hello.*



Missatge Client Hello TLSv1.3

El client envia un missatge de *Client Hello* i envia informació rellevant a com dur a terme el Handshake. Alguns del mes rellevants en son

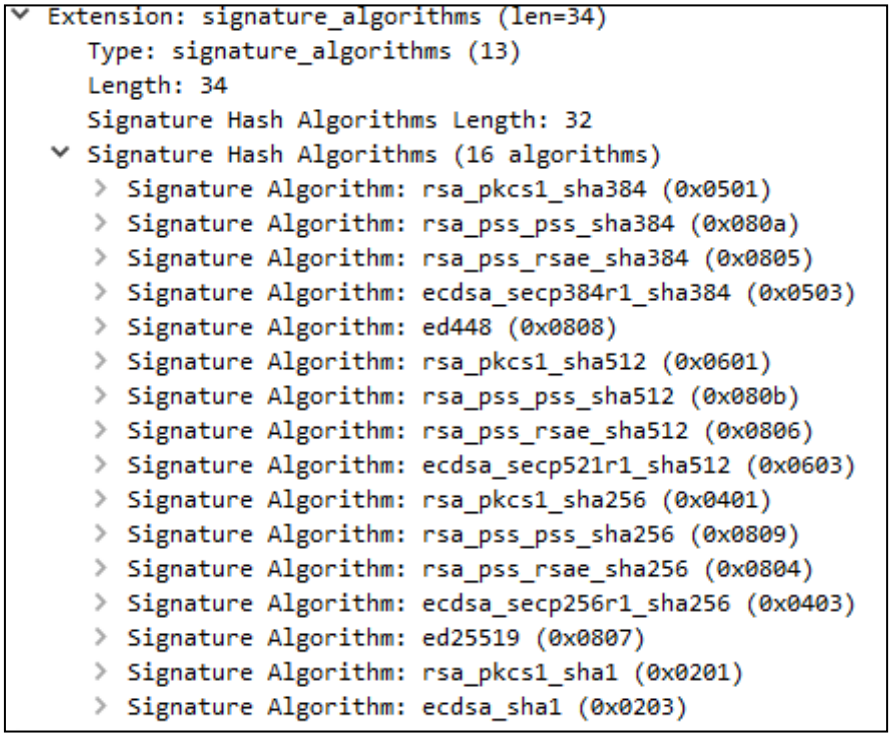
- Versió de TLS oferta
- Cipher suites suportades per el client
- Corbes suportades
- Clau compartida
- Mètodes d'intercanvi de la clau compartida
- Algorismes xifrat suportats
- Algorismes d'autenticació suportats
- Algorismes de signatura suportats

```
Cipher Suites (29 suites)
    Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
    Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
    Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
    Cipher Suite: TLS_AES_128_CCM_SHA256 (0x1304)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CCM (0xc0ad)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CCM (0xc0ac)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
    Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
    Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Cipher Suite: TLS_RSA_WITH_AES_256_CCM (0xc09d)
    Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
    Cipher Suite: TLS_RSA_WITH_AES_128_CCM (0xc09c)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x009f)
    Cipher Suite: TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xccaa)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CCM (0xc09f)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x009e)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CCM (0xc09e)
```

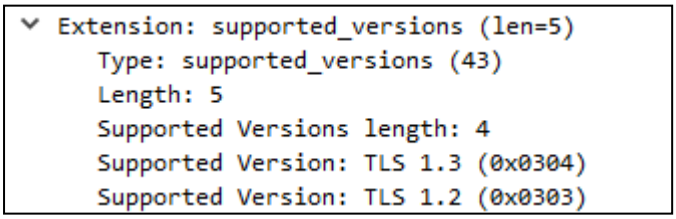
Cipher Suites ofertes per el client

```
▼ Extension: supported_groups (len=22)
     Type: supported_groups (10)
     Length: 22
     Supported Groups List Length: 20
   ▼ Supported Groups (10 groups)
        Supported Group: secp384r1 (0x0018)
        Supported Group: secp521r1 (0x0019)
        Supported Group: secp256r1 (0x0017)
        Supported Group: x25519 (0x001d)
        Supported Group: x448 (0x001e)
        Supported Group: ffdhe8192 (0x0104)
        Supported Group: ffdhe2048 (0x0100)
        Supported Group: ffdhe3072 (0x0101)
        Supported Group: ffdhe4096 (0x0102)
        Supported Group: ffdhe6144 (0x0103)
```

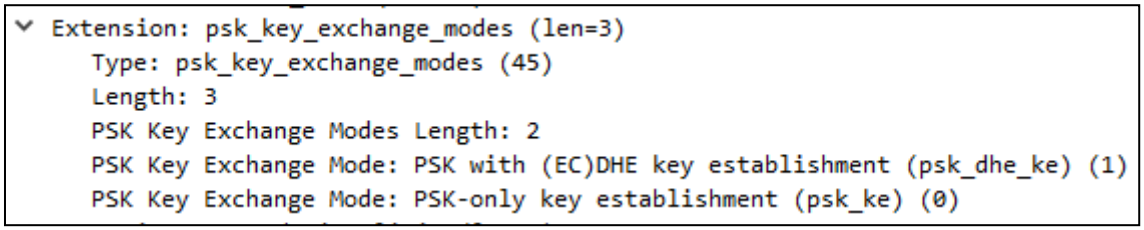
Corbes suportades per el client



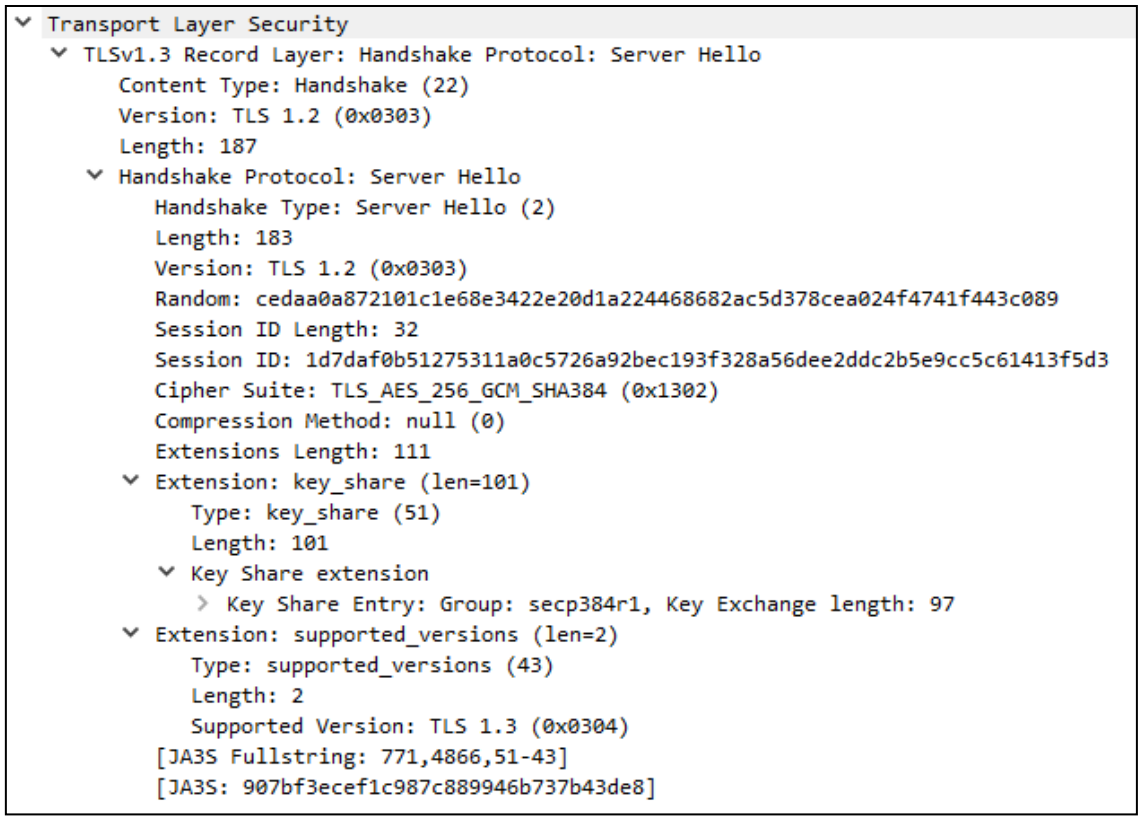
Algorismes signatura oferts client



. *Versió TLS suportada per el client*



Mètode intercanvi claus suportat per el client

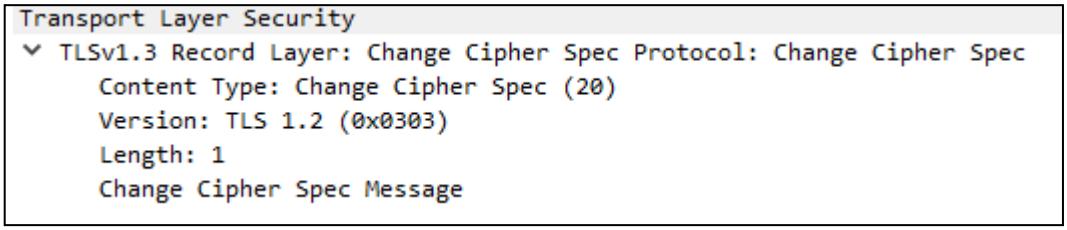


. *Missatge Server Hello*

A destacar

- La versió de TLS que es farà servir a partir d'ara
- La cipher suite usada en tota la sessió
- La clau compartida
- Com es xifra la clau compartida

Tant Servidor com client FTP s'envien un missatge d'avís de que van a canviar el xifrat segons els paràmetres acordats en el Handshake

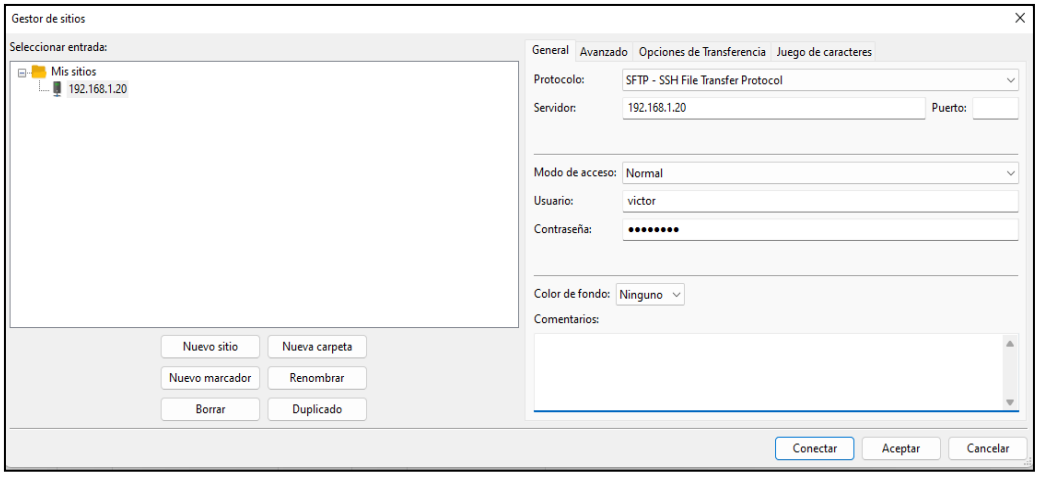


Missatge canvi de xifrat

Amb el tiquet de sessió, l'aplicació procedeix a mantenir un diàleg amb el client FTP a fi d'establir una sessió FTPS. Tant servidor com client s'envien dades.

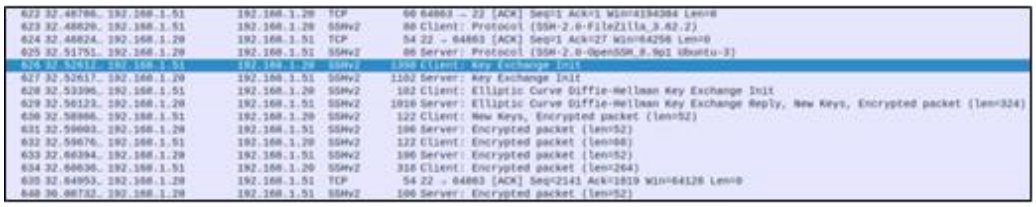
13.18. Captura dades SFTP

Des d'un client FileZilla configurarem una sessió SFTP.



Connexió SFTP amb FileZilla

Des del programa WireShark capturarem el temps que triga en establir sessió i en descarregar un fitxer de dimensió coneguda, en un entorn de xarxa de 10 / 100 Mbps.



Captura inici sessió ssh

Obtindrem el temps de connexió en capturar el timestamp del darrer paquet

```
[Timestamps]
   [Time since first frame in this TCP stream: 0.735531000 seconds]
   [Time since previous frame in this TCP stream: 0.055554000 seconds]
```

Timestamp darrer paquet

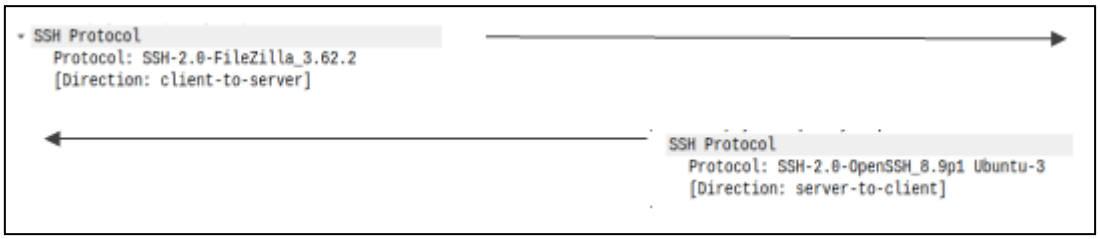
13.19. Handshake SSH

Podem observar com es du a terme la connexió. En primer lloc es presenten els actors; el client anuncia que sol·licita una connexió per el port 22, el servidor anuncia que està escoltant per aquest port per mig del protocol SSH.

Client Servidor

Protocol

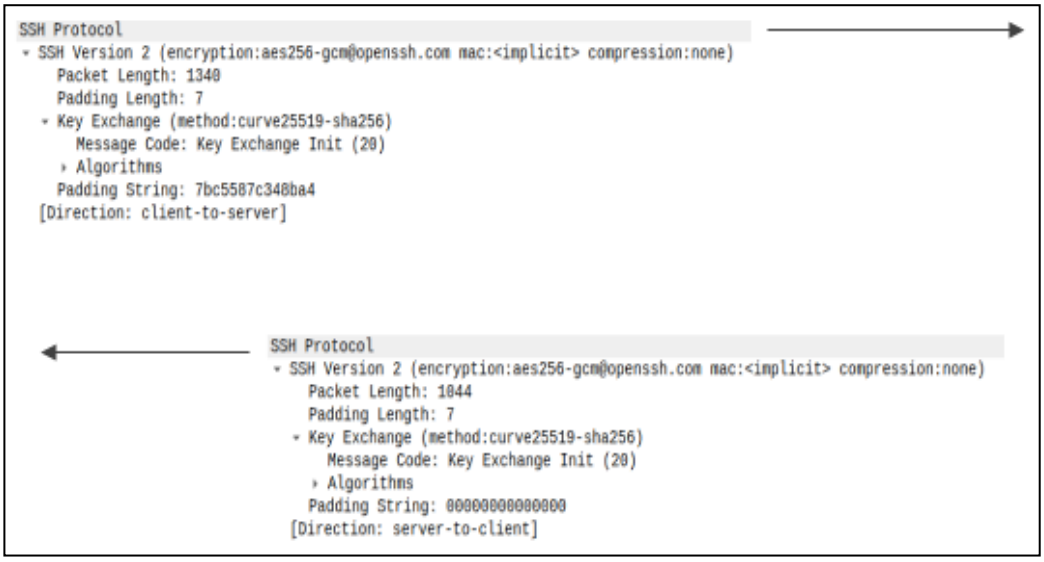
El client inicia el procés indicant la versió de protocol SSH. El servidor acorda la versió a fer servir.



Primer contacte aplicacions SSH

Key Exchange Init

Seguidament, els actors anuncien com dur a terme el Handshake. Client i servidor intercanvien un secret públic compartit, per establir un xifrat. Client ofereix protocols compatibles.



Key Exchange Init

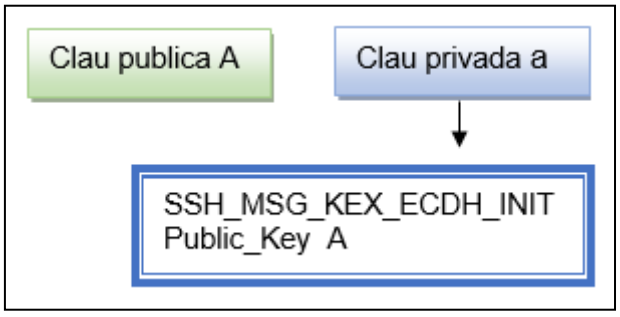
Protocols			
Intercanvi de claus	Xifrat simètric	Autenticació	Claus
curve25519-sha256@libssh.org	*chacha20-poly1305@openssh.com*		*RSA*
ecdh-sha2-nistp256	*aes128-gcm@openssh.com*	hmac-sha2-256	ECDSA
ecdh-sha2-nistp384	aes256-ctr		
ecdh-sha2-nistp521	*aes192-ctr*		
	aes128-ctr		

Protocols intercanvi claus SSH

Eliptic Curve Diffie-Hellman Key Exchange Init

El client inicia l'intercanvi de claus amb la generació d'un parell de claus efímer, i enviant al servidor la seva clau pública.

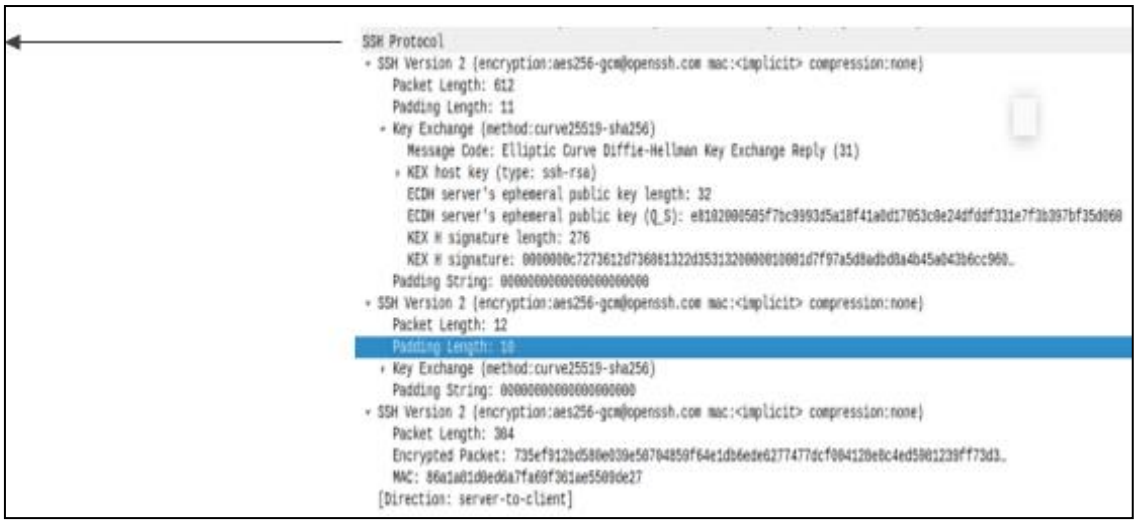
Eliptic Curve Diffie-Hellman Key Exchange Init



Enviament clau publica

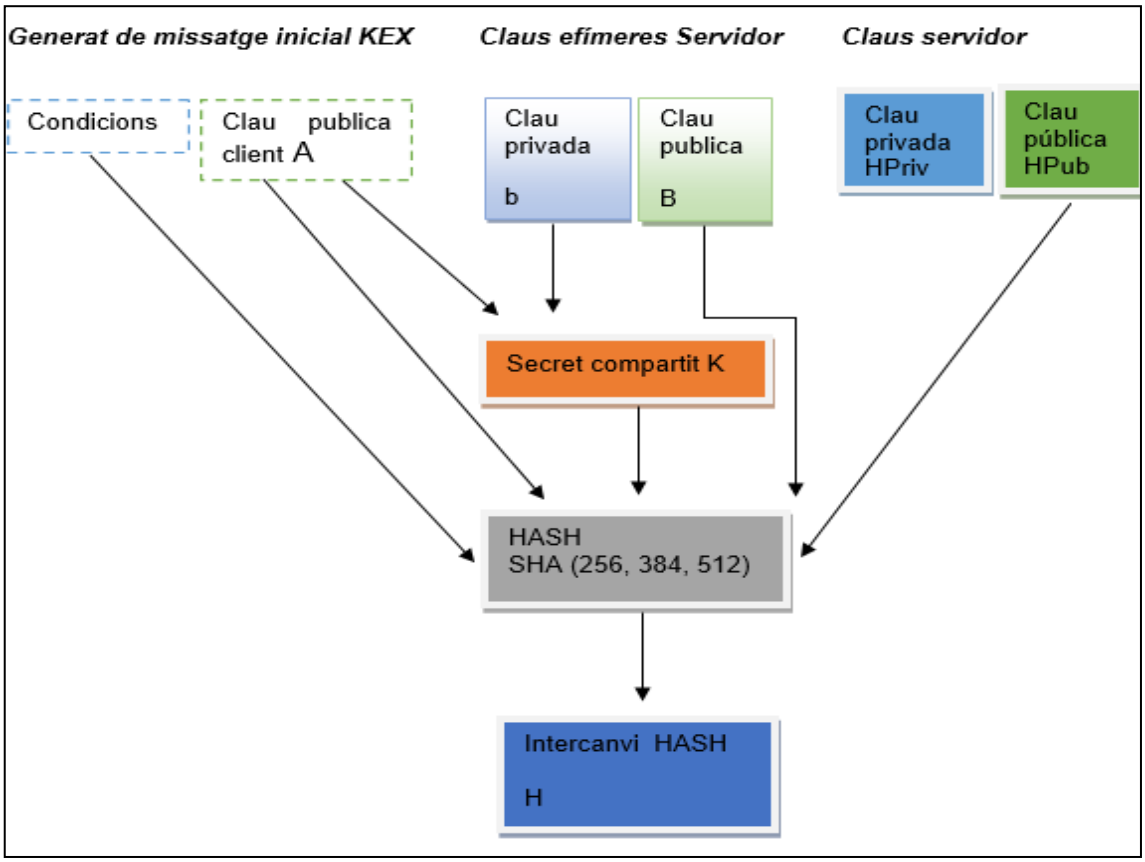
Eliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted Packet

El servidor reb el missatge i genera un parell de claus efímeres. Genera un secret compartit K amb la clau pública del client i les seves claus.



Resposta servidor. Missatge ECDH_KEY_REPLY

El servidor genera el Hash d'intercanvi i el signa, així l'altra part podrà reconèixer la seva clau en l'enviament del secret compartit signat i no tindrà cap dubte de l'autenticitat del servidor.



Generació HASH d'intercanvi H

Condicions generades per missatge inicial

Versió SSH client i servidor

Missatges client amb la seva clau publica A

Claus efímeres Servidor

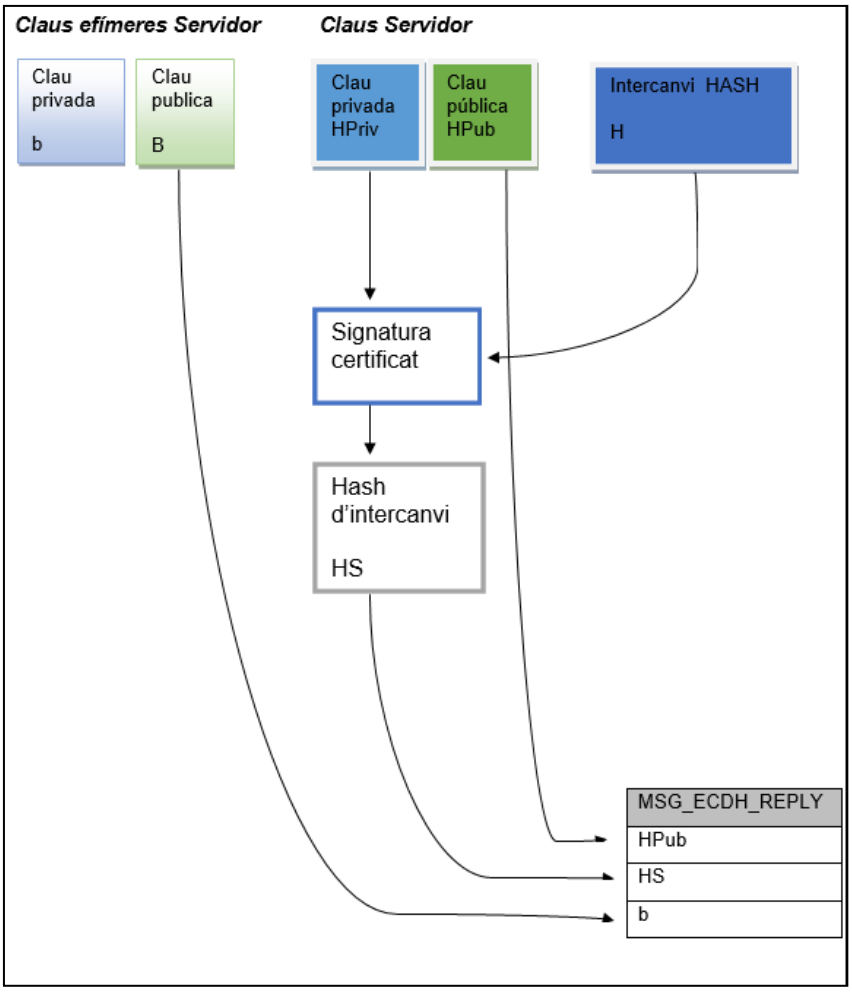
Clau privada b

Clau publica B

Claus Servidor

Clau publica i privada

Un cop és generat el Hash H d'intercanvi, signa un missatge d'intercanvi de HASH amb la clau privada del Servidor. A continuació s'envia un missatge de resposta de l'intercanvi de claus amb la clau pública del servidor (HPub), la signatura o Hash d'intercanvi signat (HS) i la clau pública efímera del servidor. [26]



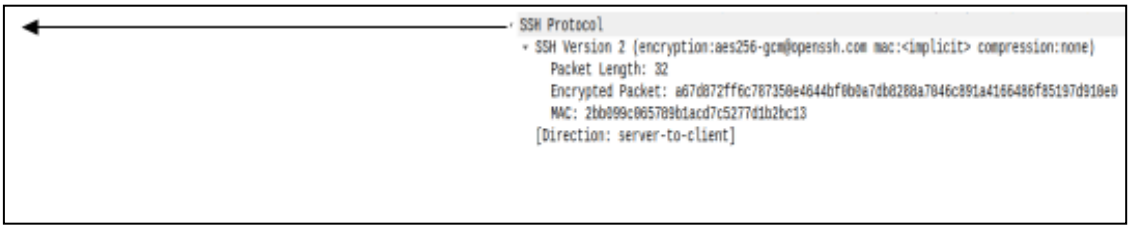
Missatge resposta servidor Diffie-Hellman

New Keys *Encrypted packet*

Amb tota la informació rebuda per el client, ja pot generar les claus de xifrat, d'inicialització i de HASH. Aquesta informació s'envia al servidor per informar de tota comunicació es xifrarà a partir d'aquestes claus. El client envia signat amb un algorisme de HASH les claus de xifrat, d'inici de connexió i de signatura de Hash al client.

Enviament noves claus

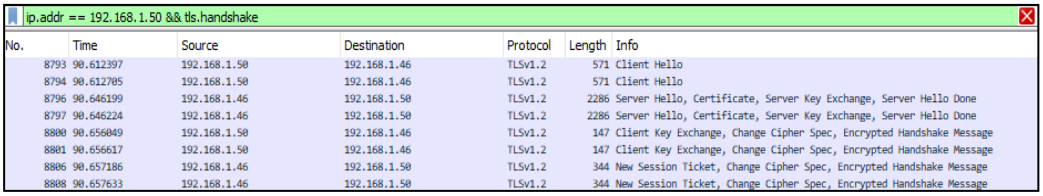
El servidor rep el missatge i envia un missatge que canvia les claus. A partir d'ara tota la comunicació s'establirà amb les claus acordades



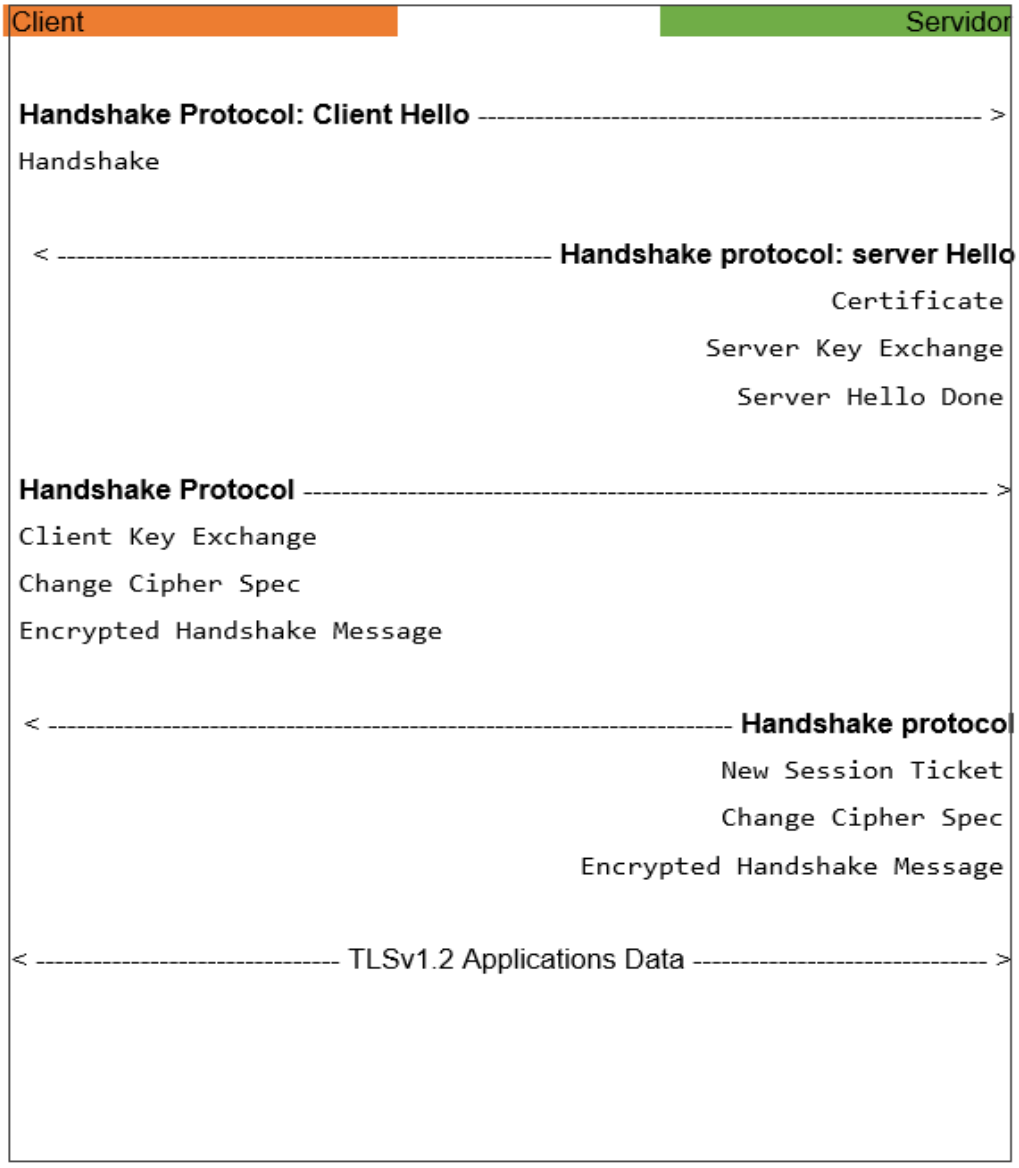
Avís canvi claus servidor

13.20. Procés Handshake TLSv1.2

La comunicació TLSv1.2 s'estableix quan un navegador demana accedir a un servei en un altre dispositiu amb un missatge de Client Hello. El servidor respon amb un missatge de Server Hello, Certificat, Server Key Exchange i Server Hello Done. El client envia missatge avisant que canvia de clau, de xifra. El servidor li dona un nou tiquet de sessió. Totes les dades s'enviaran ja xifrades.

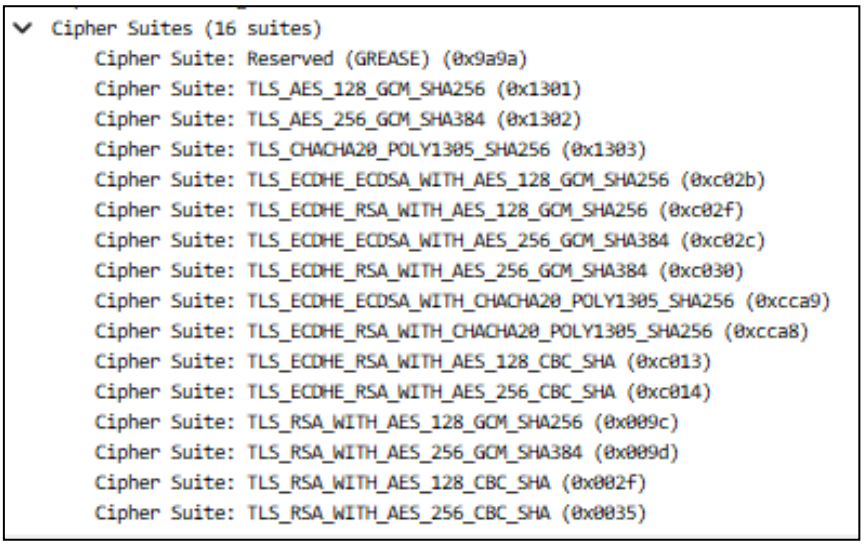


Captura Handshake TLSv1.2

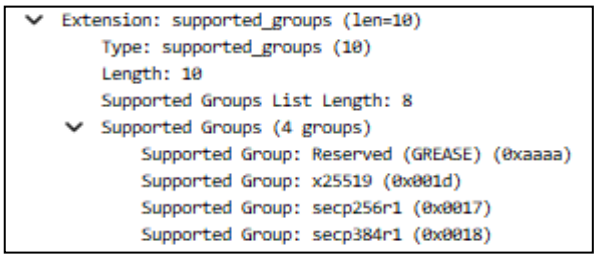


Procés Handshake TLSv.1.2

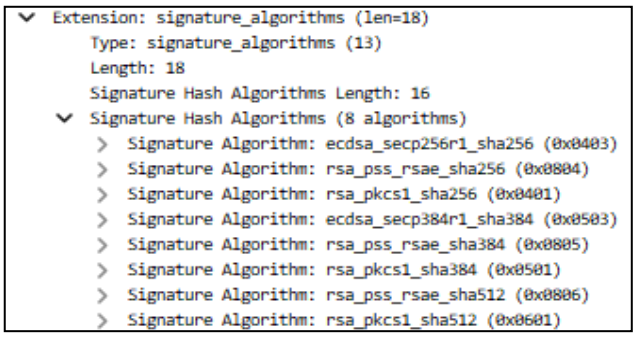
Missatge Hello TLSv1.2



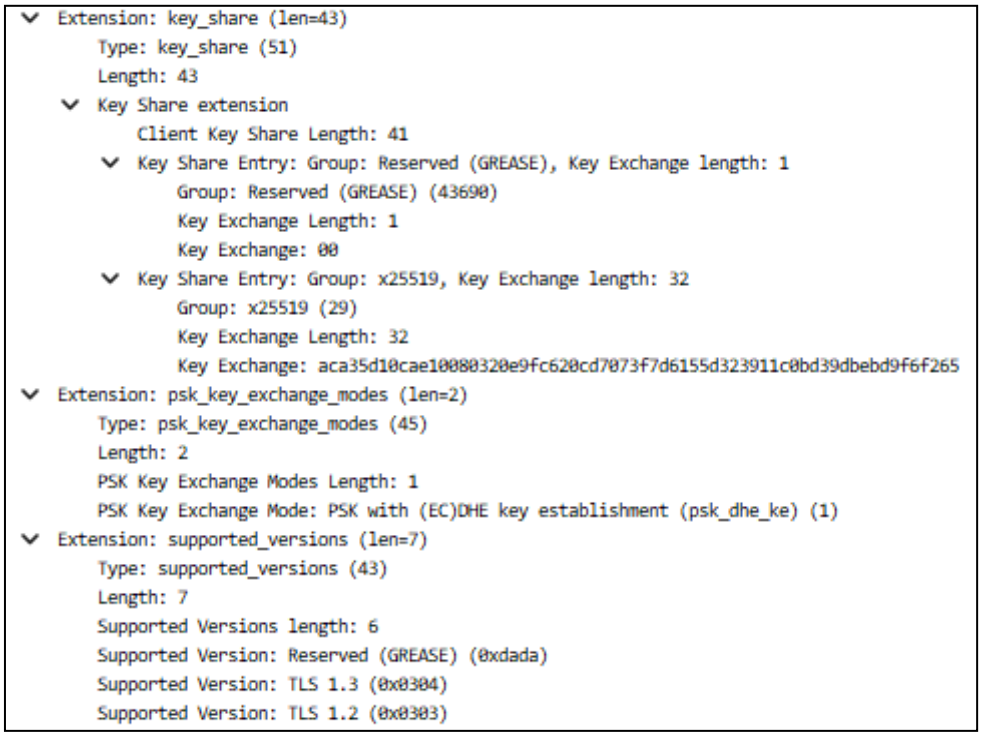
Cipher suites client. TLSv1.2



Corbes Handshake



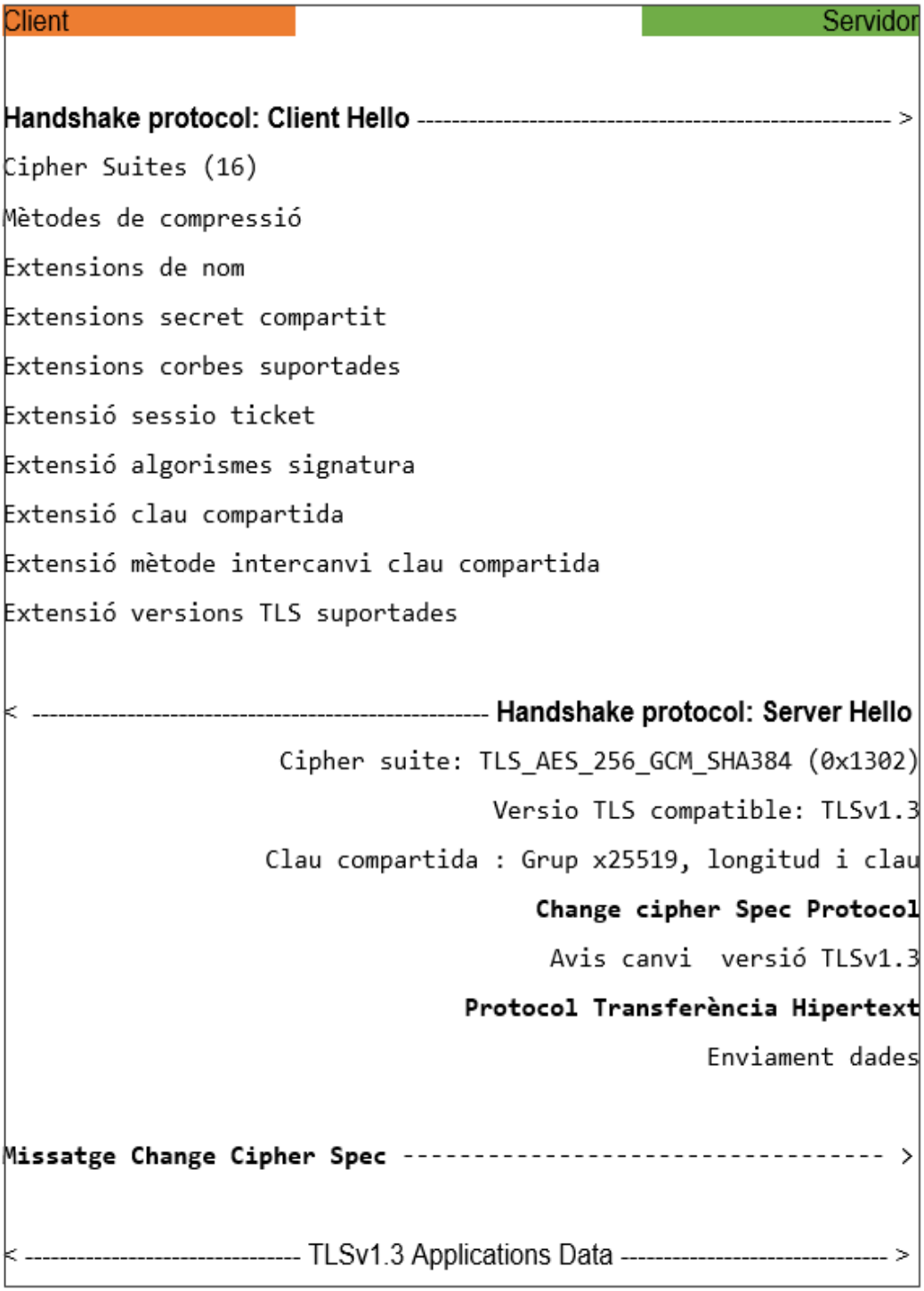
Algorismes signatura client



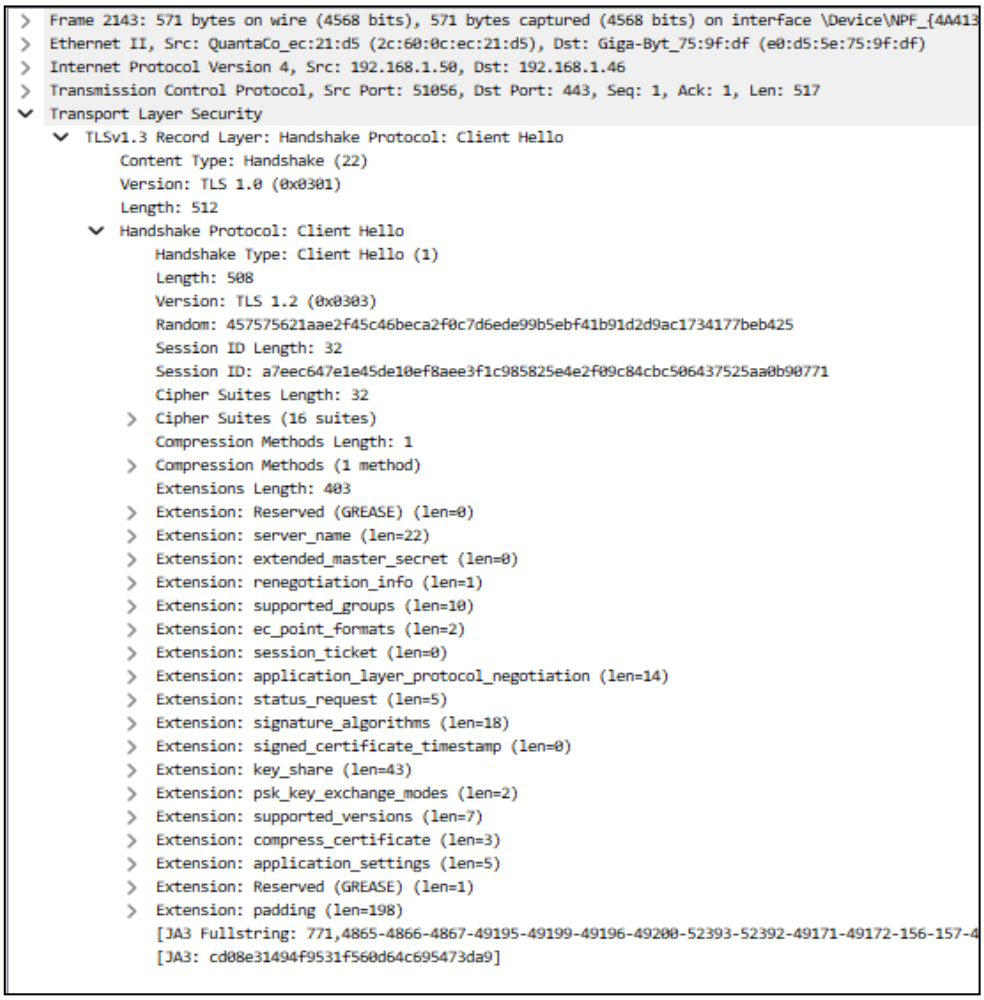
Clau compartida, mètodes d'intercanvi, versions TLS suportades

13.21. Procés Handshake TLSv1.3

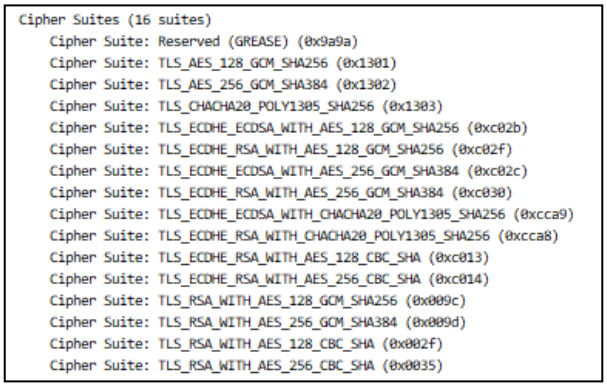
La comunicació TLSv1.3 s'estableix quan un navegador demana accedir a un servei en un altre dispositiu amb un missatge de Client Hello. El servidor respon amb un mossatge de Server Hello. Posteriorment tots dos anuncien que es va a canviar el xifrat de la connexió amb els paràmetres acordats. Totes les dades s'enviaran ja xifrades amb els paràmetres acordats



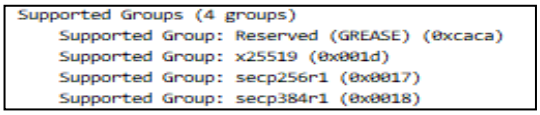
. Procés Handshake TLSv.1.3



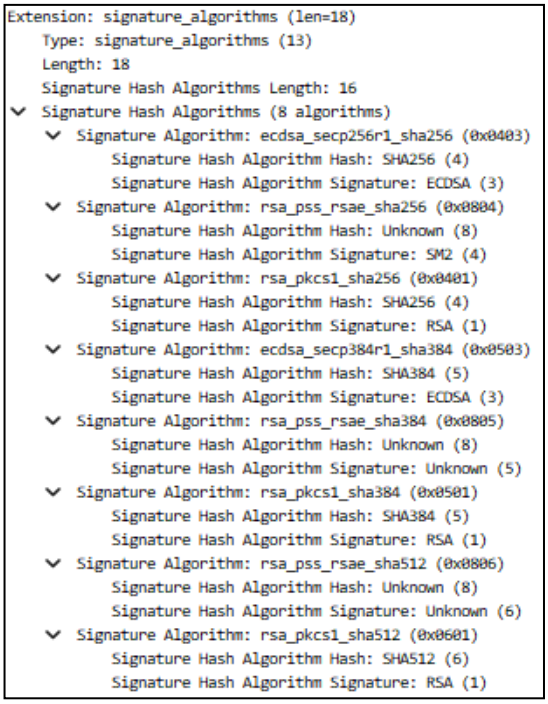
Missatge Hello TLSv1.3



. Cipher suites client. TLSv1.3



Corbes Handshake TLSv1.3



Algorismes signatura client TLSv1.3