Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the bachelor thesis:

Comet Monte, Sergi; Megias Jimenez, David, dir. Disseny i Implementació d'un Sistema de Comunicacions Basat en Esteganografia de Xarxa Sobre els Protocols TCP i IP. 2023. (Enginyeria Informàtica)

This version is available at https://ddd.uab.cat/record/280713

under the terms of the BY-NC-ND license

Disseny i Implementació d'un Sistema de Comunicacions Basat en Esteganografia de Xarxa Sobre els Protocols TCP i IP

Sergi Comet Monte

Resum– Aquest treball presenta el disseny i implementació d'un sistema estenogràfic per enviar un fitxer secret aprofitant els datagrames IP i segments TCP que es generen en una transmissió de *streaming* amb Plex. Per a cada paquet que s'envia des de el port del servidor de Plex al client i que conté dades, es modifica el camp ID del datagrama IP per enviar el secret. Per identificar cada un dels paquets modificats pel sistema esteganogràfic, es modifica l'ordre en què apareixen les opcions del segment TCP, així com el valor del *Timestamps Value* quan existeix una opció de tipus=8 (*Timestamps*) en el camp opcions del segment TCP. Per realitzar la reordenació dels paquets en el client, es fa servir el número de seqüència dels segments TCP dels paquets que va rebent el client. Per realitzar el control, en el servidor, dels paquets que han sigut correctament rebuts pel client, es fa servir el camp ACK del segment TCP dels paquets que fa rebent el servidor. Amb l'esquema proposat, s'ha aconseguit aprofitar més del 95% dels paquets de la transmissió de *streaming* per enviar el secret, i també la indetectabilitat amb l'IDS Snort.

Paraules clau– Esteganografia TCP, Esteganografia IP, Streaming, número seqüència TCP, número ACK TCP, opció marca de temps TCP

Abstract– This article presents the design and implementation of a steganographic system to send a secret file by exploiting IP datagrams and TCP segments generated in a streaming transmission with Plex. For each packet sent from the Plex server port to the client that contains data, the ID field of the IP datagram is modified to send the secret. To identify each packet modified by the steganographic system, the order in which TCP segment options appear is modified, as well as the value of the Timestamps Value when there is an option of type=8 (Timestamps) in the options field of the TCP segment. To perform packet reordering on the client side, the sequence number of the TCP segments of the packets received by the client is used. In order to control the packets that have been successfully received by the client on the server side, the ACK field of the TCP segment of the packets received by the server is used. With the proposed scheme, more than 95% of the streaming transmission packets have been successfully utilized to send the secret, ensuring undetectability with Snort IDS.

Keywords– TCP Steganography, IP Steganography, Streaming, TCP Sequence Number, TCP ACK Number, TCP Timestamp Option.

1 INTRODUCCIÓ

El terme esteganografia prové del grec *steganos* (que significa ocult), i *graphos* (que significa escriptura). És un tècnica que permet amagar missatges dins uns altres anomenats portadors, de manera que no es detectin els missatges que es volen amagar. Els inicis de la esteganografia se situen al 440 a.C. a l'antiga Grècia, quan ja s'amagaven missatges escrits en unes tauletes de fusta que es cobrien amb una cera; a sobre d'aquesta cera s'escrivia un altre missatge que amagava el primer, i així el receptor només havia de treure la cera per accedir al missatge amagat. Amb el pas del temps i l'evolució de les tecnologies, aquesta tècnica ha evolucionat fins al que s'anomena Esteganografia Digital, on es fan servir com a portadors del missatge a amagar vídeos, imatges, àudios, fitxers diversos i també els protocols que s'utilitzen actualment en Internet

• E-mail de contacte: sergicomet25@gmail.com
• Menció realitzada: Tecnologies de la Informació
• Treball tutoritzat per: David Megías Jiménez (DEIC)
• Curs 2022/23

(TCP/IP, HTTP, FTP, etc.) [1] [2]. En aquest darrer cas anomenat Esteganografia de Xarxa.

L'objectiu del treball és desenvolupar un sistema esteganogràfic per a la transmissió d'informació secreta entre un emissor i un receptor, aprofitant el camp d'identificació del datagrama IP i el camp d'opcions del segment TCP dels paquets que genera el protocol TCP/IP mentre s'està realitzant una transmissió de *Streaming*. Per millorar l'eficiència del sistema esteganogràfic es tindran en compte les següents característiques:

- Quantitat de paquets utilitzats per enviar el secret i quantitat d'informació secreta que pot transmetre cada datagrama i/o segment.
- La fiabilitat de l'enviament de la informació secreta davant d'errors de transmissió.
- La capacitat que la informació secreta transmesa no sigui detectada.

La resta del document està organitzat de la manera següent. A l'apartat 2, s'exposen les especificacions funcionals de la interfície gràfica del client i servidor. A l'apartat 3, es mostra el disseny del sistema estenogràfic realitzat.A l'apartat 4, s'explica en detall la implementació del sistema. A l'apartat 5, es presenten els resultats obtinguts i finalment, a l'apartat 6, es conclou el treball amb les conclusions.

2 ESPECIFICACIONS FUNCIONALS

El sistema consta d'una interfície gràfica única per al client i per al servidor (Apèndix A.1). Permet escollir entre treballar en mode servidor per enviar el secret o en mode client per rebre el secret.

En el cas que es treballi en mode servidor s'escollirà el port pel que treballarà el servidor de *streaming*, el fitxer secret que es desitjarà enviar, si es realitzarà compressió del secret a enviar, i també si es voldrà xifrar la informació secreta. El xifrat realitzat serà amb una clau simètrica de 32 bytes i fixa per aquest treball.

En el cas que es treballi en mode client, s'escollirà, igual que en el mode servidor, el port pel qual treballarà el servidor de *streaming*. A més, es podrà escollir si es desitja simular una pèrdua de paquets i amb quina freqüència es produirà.

En ambdós casos, es pot seleccionar si es vol obtenir registres detallats, tant a la pantalla com en un fitxer de disc que es guardarà com a "./log.txt". Durant la transmissió, es mostrarà un resum en temps real de totes les dades transmeses, la durada de la transmissió i també es mostrarà el codi de redundància cíclica (CRC) del secret que s'envia en el cas del servidor, i el CRC del secret rebut en el cas del client. Això permetrà confirmar si la transmissió del secret ha estat correcta.

3 DISSENY

El sistema s'ha dissenyat tenint en compte que un client web en una màquina amb Ubuntu realitza la visualització d'un vídeo emmagatzemat en una altra màquina amb Ubuntu i amb un servidor de Plex en funcionament. A més, s'ha tingut en compte tant si la visualització es realitza a través d'una xarxa local com si es realitza a través d'Internet.

Un cop s'ha iniciat el sistema en mode client i mode servidor, respectivament, en cadascuna de les màquines, el servidor, aprofitant el tràfic de paquets de sortida cap al client envia senyals de sincronització (SYNC), fins que el client, retorna al servidor una altre senyal de SYNC. Un cop el servidor ha rebut el senyal de SYNC del client, s'estableix la connexió i es comencen a enviar les dades secretes marcant cada paquet amb la senyal de dades (DADES), tenint en compte que s'aprofita tot el tràfic de paquets amb mida més gran de 0.

Per a cada paquet de DADES enviat, el servidor va guardant en un *buffer* la informació del secret enviat, el número de seqüència del segment TCP (SEQ) (*Número de seqüència* Fig. 1) i el número de reconeixement esperat (ACK) (*Número de confirmació de la recepció* Fig. 1).

Tots aquests paquets marcats amb el senyal de DADES van arribant al client, que els anirà guardant en un *buffer*. En aquest *buffer* es guardarà el secret, el número de seqüència (SEQ) del paquet rebut i el número d'ordre que es veurà més endavant.

A mesura que el servidor va rebent els ACK, que el protocol TCP del sistema operatiu del client ha anat enviant al servidor corresponent a cada paquet marcat com DADES, es van marcant com rebuts en el *buffer* del servidor. En el moment que s'arriba al número màxim de paquets enviats i ja s'han rebut tots els ACK, el servidor envia una senyal de final de bloc (FIBLOC) al client i aquest contesta amb una altre senyal de FIBLOC. Un cop el servidor rep la senyal de FIBLOC del client, se segueixen enviant més dades secretes marcant de nou cada paquet com DADES.

Quan el servidor arriba al final del secret a enviar i ja s'han rebut tots els ACK, aquest envia una senyal de final de secret (STOP) al client. Quan el client rep el senyal STOP del servidor, envia un senyal de STOP al servidor i finalitza la recepció. Quan el servidor rep el senyal de STOP del client finalitzarà la transmissió (Apèndix A.2).

4 IMPLEMENTACIÓ

4.1 Eines utilitzades

Tot el sistema s'ha implementat amb 2 màquines virtuals, una per al client i una altra pel servidor, amb 4GB RAM, 20GB de disc dur i sistema operatiu Ubuntu v. 20.04.5 en cadascuna.

S'ha fet servir Python 3.8.10 per realitzar els diferents programes que componen el sistema, la llibreria Scapy per a realitzar la modificació del paquets TCP/IP [3], la llibreria NetfilterQueue per interceptar la pila TPC/IP del *kernel* de l'Ubuntu [4], la llibreria Pycryptodome per realitzar el xifratge del secret en AES-256, la llibreria Zipfile per la compressió del secret i finalment la llibreria Tkinter [5] per a realitzar la interfície gràfica.

A més, s'ha fet servir la llibreria Socket per obtenir l'adreça IP d'una màquina, la llibreria Threading per treballar amb fils i la llibreria Datetime per calcular el temps de durada de la transmissió.

Per a realitzar l'anàlisi dels paquets s'ha utilitzat el Wireshark i per realitzar les diferents comprovacions que el

Bit offset	Bits 0–3	4–7	8–15	16–31
0	Port origen			Port destinació
32	Número de seqüència			
64	Número d'Acknowledgment			
96	Offset dades	Reservat	CWR ECE URG ACK PSH RST SYN FIN	Finestra
128	Checksum			Urgent pointer
160	Opcions (opcional)			

Fig. 1: Camp opcions del segment TCP

sistema no és detectable per un sistema de detecció d'intrusos (IDS), s'ha fet servir Snort v.2.9.7.0 GRE (*build 149*) configurat amb les regles per defecte i instal.lat en el client [6].

Pel que fa al sistema de *streaming*, s'ha instal·lat Plex v.4.87.2 en el servidor i s'ha configurat per a compartir un vídeo de prova. En el client, simplement introduint en un navegador la IP del servidor i el port 32400 (http://ip_servidor:32400) s'accedeix al vídeo del servidor en *streaming* [7].

4.2 Captura de paquets

Si el sistema funciona en mode servidor, el programa afegirà un regla al *firewall* de l'Ubuntu perquè tots els paquets TCP que surtin des de la IP local del servidor i amb el port origen el port del servidor es desviïn a una funció que gestionarà aquests paquets. També afegirà una altra regla al *firewall* perquè tots els paquets TCP que entrin amb IP de destinació del servidor i port de destinació el del servidor, es desviïn a la funció que gestionarà aquests paquets. Per a fer això s'ha utilitzat la llibreria NetfilterQueue.

Un cop els paquets ja passen per la funció, s'utilitza la llibreria Scapy per a analitzar el contingut de cada camp del datagrama IP i/o segment TCP. A més, aquesta llibreria permet modificar qualsevol camp i tornar-lo a la pila TCP/IP perquè segueixi el seu camí. Quan es modifica un camp el *cheksum* canvia, però amb Scapy es recalcula automàticament si l'hem esborrat primer.

Per la part del client, es fa el mateix, però les regles del *firewall* del Ubuntu desvien el tràfic de sortida on la IP origen sigui la IP local del client i port de destinació el port del servidor. Per als paquets d'entrada, s'afegeix una regla on es desvien tots els paquets d'entrada amb IP de destinació del client i port origen del servidor. En l'apèndix A.3 es pot observar un fragment de codi amb les instruccions utilitzades.

4.3 Control i modificació de paquets

Per poder identificar els paquets que el sistema esteganogràfic fa servir per intercanviar informació entre servidor i client, s'ha fet servir el camp opcions del segment TCP. A mes, també s'ha fet servir el camp *TS Value* per l'opció de tipus *timestamps* per a identificar el tipus de paquet que s'està rebent o enviant (SYNC, DADES, FIBLOCK, STOP i REENVIAMENT). Si ens fixem en la Fig. 1, podem observar l'estructura d'un segment TCP i en concret, passarem a analitzar el contingut del camp opcions.

Aquest camp d'opcions sempre serà múltiple de 4 bytes (32 bits) i, depenent de la informació que emmagatzemi, tindrà una estructura o una altra. En concret, ens fixarem en la Fig. 2, on es mostra una captura del Wireshark d'un paquet enviat pel servidor Plex al client, per indicar com s'ha fet servir aquest camp.

En aquesta captura s'observen 12 bytes que corresponen al *timestamps* del segment TCP corresponent al paquet enviat pel servidor. El *timestamps* indica una marca de temps que permet, tant al servidor com al client, comprovar quant triga a viatjar un paquet entre la font i la destinació (anada i tornada).

Aquests 12 bytes s'estructuren, per una banda, en 2 bytes que contenen el valor 01 indicant que no realitzen cap operació [8]; per altra banda, 10 bytes que contenen la informació de *timestamps*. Aquests 10 bytes estan configurats de la següent manera: el primer byte té un valor de 08, el qual indica que la informació és de tipus *timestamp*. El següent byte té un valor de 0a, indicant que la longitud d'aquest camp és de 10. Després, hi ha 8 bytes més que contenen el valor del *timestamp* del servidor, utilitzant 4 d'aquests bytes, i el valor del *timestamp* del client, utilitzant els altres 4 bytes [9]. Cal destacar que els 2 bytes amb valor 01 serveixen únicament per emplenar el camp d'opcions perquè tingui una mida múltiple de 4 bytes (Fig. 3).

Després de fer moltes proves i d'analitzar tots els paquets transmesos del servidor al client i viceversa, s'ha pogut observar que, en tots els casos, aquests 10 bytes corresponents al *timestamps* mai apareixen al principi de les opcions. Per tant, per tal de marcar un paquet com a paquet que conté informació estenogràfica, s'ha modificat el camp d'opcions del segment TCP perquè el camp *timestamps* estigui al principi de les opcions, tal i com es mostra a la Fig. 4. Així, tot paquet que tingui el camp *timestamps* al principi serà un paquet que ha estat manipulat pel sistema esteganogràfic.

A més, també s'ha modificat el camp de *timestamp value*, modificant-lo el mínim possible, perquè doni un número mòdul 6. Així podrem codificar 6 tipus de paquets diferents, encara que només ens en calen 5 (SYNC, DADES, FIBLOCK, STOP i REENVIAMENT) per aquest disseny. Es restaran fins a 5 unitats en funció del número mòdul 6 que es desitgi.

En la captura de la Fig. 4 veiem un exemple, de com el servidor envia un paquet al client marcat amb el senyal de SYNC. Observem com el camp de *timestamp value* és 150737106, aquest número mòdul 6 és 0, el qual correspon al senyal de SYNC. El valor original era 150737108 i només s'ha restat 2 per aconseguir un número que mòdul 6 doni 0.

```
▾ Transmission Control Protocol, Src Port: 32400, Dst Port: 44466, Seq: 226, Ack: 262, Len: 25
    Source Port: 32400
    Destination Port: 44466
    [Stream index: 18]
    [Conversation completeness: Incomplete (12)]
    [TCP Segment Len: 25]
    Sequence Number: 226    (relative sequence number)
    Sequence Number (raw): 3998275540
    [Next Sequence Number: 251    (relative sequence number)]
    Acknowledgment Number: 262    (relative ack number)
    Acknowledgment number (raw): 1739967944
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
    Window: 501
    [Calculated window size: 501]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x8446 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  ▾ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
    ▸ TCP Option - No-Operation (NOP)
    ▸ TCP Option - No-Operation (NOP)
    ▸ TCP Option - Timestamps: TSval 145958441, TSecr 3679851101
  ▸ [Timestamps]
  ▸ [SEQ/ACK analysis]
    TCP payload (25 bytes)
```

Fig. 2: Camp opcions del segment TCP

```
0000  00 0c 29 97 07 36 00 0c  29 81 19 7c 08 00 45 00
0010  00 4d e3 02 40 00 40 06  d3 a1 c0 a8 01 0a c0 a8
0020  01 ac 7e 90 ad b2 ee 50  d7 d4 67 b5 cd c8 80 18
0030  01 f5 84 46 00 00 01 01  08 0a 08 b3 26 29 db 56
0040  12 5d 17 03 03 00 14 55  90 9b c0 d3 5c ef 6d 36
0050  d9 da 58 47 38 83 c5 39  89 a6 9e
```

Fig. 3: Camp Timestamps en hexadecimal

Per tant, la variació màxima que el *timestamp value* arribarà a tenir serà de -5 unitats, per tal d'aconseguir els 6 tipus de paquets possibles.

Cal destacar, que una variació excessiva en el *timestamps* podria interferir negativament en la transmissió, ja que distorsionaria els *timestamps* del protocol TCP. En aquest cas, després de fer proves no s'han detectat anomalies ni des del Wireshark, ni des del Snort.

Així, tots els paquets que han estat modificats pel sistema seran correctament identificats. A més, es podrà saber de quin tipus de paquet estem tractant (SYNC, DADES, FI-BLOC, STOP i REENVIAMENT) i sense que es produeixin falses identificacions.

Pel que fa als paquets que el servidor retransmet degut a problemes de transmissió, el sistema fa servir el *buffer* on ha anat guardant els número de SEQ, secret i número d'ordre per detectar si és un paquet que ja ha estat transmès. En aquest cas, el sistema sabrà que és una retransmissió perquè el número de SEQ ja existirà en el *buffer* i el que farà és enviar de nou el mateix secret amb el mateix número d'ordre. En la següent secció s'explica en detall el número d'ordre.

4.4 Confirmació i redordenació de paquets

Amb la modificació del camp *timestamps* s'aconsegueix disposar dels 2 bytes del camp identificació del datagrama IP (Fig. 5), fent servir 1 byte pel secret (byte menys significatiu) i un altre byte pel número d'ordre (byte més significatiu). Cal destacar, que aquest número d'ordre no es farà servir exactament per ordenar els paquets, ja que es farà servir el propi SEQ del segment TCP per fer-ho.

És important assenyalar que tots els paquets que continguin secret estaran marcats amb la senyal de DADES i faran servir el byte menys significatiu del camp identificació del datagrama IP. Per això, en una transmissió de *streaming* entre xarxes amb diferent *Maximum Transmission Unit* (MTU), el protocol IP farà servir aquest camp identificació per poder ajuntar de nou els datagrames IP. Per tant, no es podrà fer servir aquest camp per emmagatzemar ni el número d'ordre, ni el secret. Amb això, és important tenir en compte que aquest sistema esteganogràfic es veurà limitat en transmissions on el MTU de cada xarxa per on circulin els paquets no sigui el mateix [10].

Aquest número d'ordre s'utilitza per solucionar el problema que el servidor de Plex de vegades envia dades a ports diferents del client. Quan s'envia un paquet del servidor al client per un altre port del client, el comptador de SEQ és diferent i no podríem reordenar els paquets correctament. Es fa servir aquest byte com un número que indica l'ordre del port que s'està utilitzant. Si, per exemple, el primer paquet que conté secret s'envia al port del client número 46826, aquest byte serà 0, si els següents paquets també s'envien al mateix port del client el número es mantindrà a 0, però si el port del client canvia, el número passarà a 1. Per tant, aquest byte guardarà com a màxim 256 canvis de port i per a cadascun d'ells els paquets s'ordenaran pel SEQ.

Quan s'arribi al número d'ordre 256, el servidor haurà d'esperar a haver rebut tots els ACK del segment TCP, que el sistema operatiu del client hagi anat enviant automàticament segons el funcionament del protocol TCP/IP [11] [12], per començar a enviar un nou conjunt de 256 números d'ordre. Per saber si s'han rebut tots els ACK que el client hagi enviat, el servidor va guardant en un *buffer* tots els ACK que espera rebre per a cada paquet que hagi enviat. Tenint en compte que només s'utilitzen paquets amb longitud de dades més gran que 0 per enviar el secret, podem determinar el corresponent ACK que el servidor rebrà sumant la longitud del paquet de dades al número de seqüència corresponent del segment TCP.

En la Fig. 6 es mostra una captura realitzada amb el Wireshark d'un segment TCP corresponent a un paquet que va del servidor al client, on veiem com el SEQ és 3537727697 i té una longitud de 25 bytes. No obstant això, el ACK esperat pel servidor serà 3537727697+25=3537727722. Tal i com es mostra a la figura 7, el següent paquet que es rep és un paquet del client al servidor i amb ACK 3537727722. Ara, el servidor té constància que el client ha rebut el paquet que se li va enviar amb SEQ 3537727697.

```
▾ Transmission Control Protocol, Src Port: 32400, Dst Port: 46826, Seq: 1, Ack: 1, Len: 25
    Source Port: 32400
    Destination Port: 46826
    [Stream index: 14]
    [Conversation completeness: Incomplete (12)]
    [TCP Segment Len: 25]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 1402232201
    [Next Sequence Number: 26    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 3578982752
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
    Window: 501
    [Calculated window size: 501]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x4dde [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  ▾ Options: (12 bytes), Timestamps, No-Operation (NOP), No-Operation (NOP)
    ▾ TCP Option - Timestamps: TSval 150737106, TSecr 3684629759
        Kind: Time Stamp Option (8)
        Length: 10
        Timestamp value: 150737106
        Timestamp echo reply: 3684629759
    ▸ TCP Option - No-Operation (NOP)
    ▸ TCP Option - No-Operation (NOP)
  ▸ [Timestamps]
  ▸ [SEQ/ACK analysis]
    TCP payload (25 bytes)
```

Fig. 4: Camp opcions modificat del segment TCP

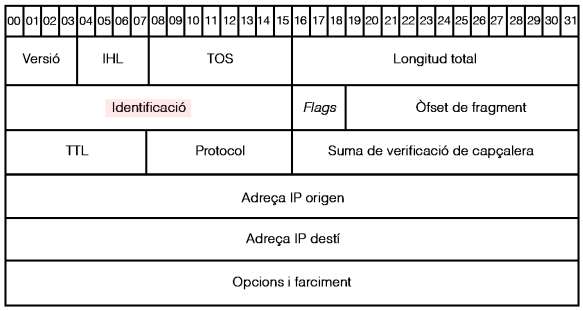


Fig. 5: Camp identificació datagrama IP

Quan el client rep un paquet del servidor, el primer que fa és analitzar el tipus de paquet que és (SYNC, DADES, FIBLOC, STOP i REENVIAMENT) i en el cas que sigui de dades, guarda en un *buffer* el número d'ordre, el SEQ i el secret.

Quan el servidor rep un paquet del client, comprova el ACK rebut amb els ACK guardats en el *buffer* i marca com a correctament enviats tots paquets que tinguin els ACK iguals o inferiors l'ACK rebut, ja que el protocol TCP pot realitzar confirmació múltiple de paquets. A més, un cop el servidor ha arribat al número màxim d'ordre i ja no existeixen paquets pendents de confirmació, envia al client un senyal de FIBLOC perquè el client ordeni els paquets segons número d'ordre i número de seqüència. Un cop el client ha ordenat els paquets els guardarà en un *buffer* definitiu.

Finalment, quan el servidor ha arribat al final del secret a transmetre, es realitzarà el mateix procés que amb FIBLOC, però amb el senyal de STOP. En aquest cas, el client realitzarà una descompressió i/o un desxifratge en cas que el secret hagi arribat comprimit i/o xifrat. Finalment, es guardarà en disc el secret rebut.

Per a que el client sàpiga si el secret arribarà comprimit i/o xifrat, el servidor envia juntament amb el senyal de SYNC 4 possibles valors codificats dins el camp identificació del datagrama IP. El procediment seguit per modificar el camp identificació ha estat el mateix que s'ha utilitzat per modificar el camp TSValue del *Timestamps*. Com que només ens calen 4 tipus de valors (NOCOMPRIMIT-NOXIFRAT, COMPRIMIT-NOXIFRAT, NOCOMPRIMIT-XIFRAT, COMPRIMIT-XIFRAT), s'ha augmentat el mínim possible el camp identificació del datagrama IP per a que doni un número en concret mòdul 4, i així poder codificar aquests 4 valors. A part d'això, s'aconsegueix modificar molt poc el valor inicial del camp identificació del datagrama IP.

5 Resultats

En la Fig. 8 es mostren 3 taules comparatives. Cada taula correspon a un mida diferent del secret i en cada taula es mostren els diferents paquets que es transmeten en funció del número de paquets que es perden en la transmissió.

La columna "freqüències de pèrdua de paquets" ens indica cada quants paquets rebuts pel client es perdrà un paquet. Els "bytes enviats" indiquen el total de bytes transmesos durant la reproducció de *streaming*, els "paquets enviats" són tots els paquets transmesos, amb o sense dades, durant la reproducció. Els "paquets enviats + secret" són tots els paquets enviats que contenen secret. Finalment, la columna "% paquets + secret" indica la proporció de paquets amb secret enviats respecte el total de paquets enviats i el "% secret enviat" la proporció de bytes enviats de secret respecte els bytes totals enviats per la reproducció de *streaming*, tenint en compte que, per a cada paquet amb secret enviat, s'està enviant 1 byte de secret.

Es pot observar com el percentatge de paquets que contenen secret respecte al número de paquets totals enviats és superior al 90%, per tant, s'aconsegueix aprofitar la major part dels paquets que es generen durant la reproducció de *streaming* per enviar el secret.

Pel que fa a la quantitat d'informació transmesa, sempre serà de 1 byte per a cada paquet amb secret, amb això, el percentatge de bytes enviats del secret respecte els bytes totals enviats es manté en el 0.07%.

Respecte a les proves realitzades amb el IDS Snort, configurat amb les regles predeterminades, cal indicar que en cap cas s'han observat alarmes. Tampoc s'han detectat anomalies amb el Wireshark, que indiquin mal funcionament

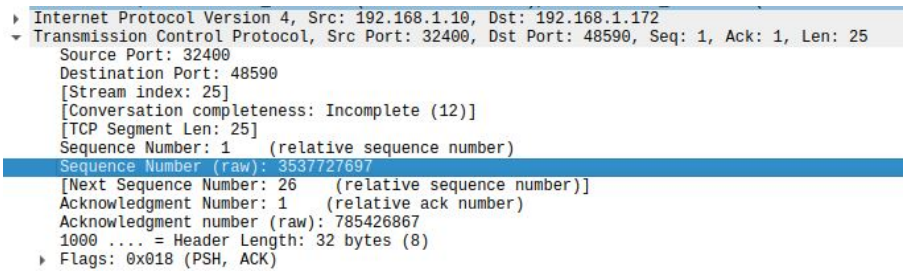


Fig. 6: Captura Wireshark del paquet enviat pel servidor

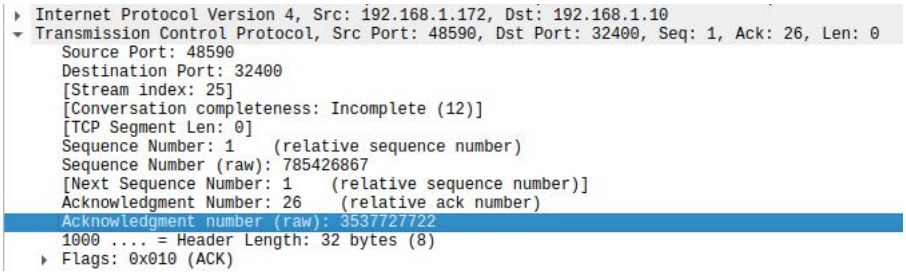


Fig. 7: Captura Wireshark del paquet enviat pel client

Secret (bytes)	Freq. pèrdua paquets	Bytes enviats	Paquets enviats	Paquets enviats + secret	% paquets + secret	% secret enviat
291	mai	437.971	328	300	91,46%	0,07%
291	1/20	448.497	326	313	96,01%	0,07%
291	1/10	463.885	354	328	92,66%	0,07%
291	1/5	515.090	371	365	98,38%	0,07%
291	1/2	795.064	605	582	96,20%	0,07%

Secret (bytes)	Freq. pèrdua paquets	Bytes enviats	Paquets enviats	Paquets enviats + secret	% paquets + secret	% secret enviat
581	mai	886.676	636	592	93,08%	0,07%
581	1/20	936.185	675	623	92,30%	0,07%
581	1/10	959.317	695	657	94,53%	0,07%
581	1/5	1.062.542	766	734	95,82%	0,07%
581	1/2	1.676.751	1.195	1.175	98,33%	0,07%

Secret (bytes)	Freq. pèrdua paquets	Bytes enviats	Paquets enviats	Paquets enviats + secret	% paquets + secret	% secret enviat
1.171	mai	1.767.750	1.252	1.181	94,33%	0,07%
1.171	1/20	1.851.142	1.305	1.239	94,94%	0,07%
1.171	1/10	1.936.803	1.362	1.306	95,89%	0,07%
1.171	1/5	2.155.825	1.507	1.466	97,28%	0,07%
1.171	1/2	3.411.805	2.446	2.407	98,41%	0,07%

Fig. 8: Estadístiques enviament

del protocol TCP/IP provocades per la manipulació dels paquets per part del sistema esteganogràfic realitzat en aquest treball.

6 Conclusions

Amb el sistema esteganogràfic dissenyat en aquest treball s'ha aconseguit aprofitar la major part dels paquets transmesos en una reproducció de *streaming* amb Plex. A més, gràcies al sistema implementat, s'ha assegurat l'enviament del secret enviat davant errors de transmissió, aprofitant el propi sistema de reenviament que disposa el protocol TCP. Així mateix, s'ha aconseguit reordenar els paquets en el client utilitzant el camp SEQ del protocol TCP i el número d'ordre del sistema dissenyat. Això permet garantir la integritat i l'ordre correcte dels paquets que contenen la informació secreta.

A més a més, per augmentar la quantitat d'informació secreta a enviar, s'ha inclòs un sistema de compressió que redueix la informació inicial a enviar.

Un cop comprovat que el IDS Snort i el Wireshark no detecten anomalies, podem afirmar que la informació secreta transmesa no serà detectada fàcilment. Així també, l'ús d'un sistema de criptografia de clau simètrica com l'AES-

256 ens permet assegurar que, fins i tot si la informació secreta és detectada, no serà possible desxifrar el contingut sense la clau adequada. Això proporciona una capa addicional de seguretat i protecció per a la informació confidencial transmesa.

Per millorar el nombre de paquets, que s'aprofiten per enviar el secret durant la reproducció de *streaming*, es podrien fer servir els paquets que el servidor segueix enviant al client mentre el servidor està esperant els ACK del client. Així, tots aquests paquets que el servidor no està aprofitant, abans d'enviar al client el senyal de FIBLOC, podrien incloure secret. Aquests paquets es podrien gestionar en un segon *buffer*, tant en el client com en el servidor, i així aprofitaríem més paquets de la transmissió.

Per altre costat, la quantitat d'informació secreta que s'envia no és del tot elevada. Per millorar-la, es podria fer servir part de les opcions del datagrama IP per enviar més informació, modificant l'ordre de certes opcions o modificant estratègicament certs camps.

També es podrien realitzar altres modificacions en les opcions del segment TCP, diferents de les que s'han fet en aquest treball. Per exemple, en comptes de posar únicament el camp *timestamp* al començament, es podria combinar l'ordre d'aquest camp amb el dels 2 bytes de farciment.

Una altra millora que es podria realitzar, vigilant si la transmissió de *streaming* travessa algun *router*, seria utilitzar el camp TTL (Fig. 5) del datagrama IP per transmetre més quantitat de secret. Si la transmissió fos en xarxa local podríem fer servir el byte sencer, però en cas de travessar un *router*, el valor d'aquest camp aniria disminuint en una unitat i aleshores, hauríem de fer servir les parts altes del byte.

AGRAÏMENTS

Voldria donar el meu sincer agraïment a l'Eva, la meva dona, per la paciència que ha tingut mentre he estat submergit en aquest projecte, així com, als meus germans Ricard i Violeta pel suport que m'han donat per a dur-lo a terme.

Tanmateix, no voldria deixar de donar les gràcies al David Megías, director del TFG, per les seves indicacions i correccions que han fet molt més senzilla l'elaboració d'aquest treball.

REFERÈNCIES

[1] Wikipedia. "Esteganografía". [En línia]. Disponible a: https://es.wikipedia.org/wiki/Esteganografía Consulta: 1 de Març de 2023.

[2] FdIwiki ELP. "Esteganografía". [En línia]. Disponible: https://wikis.fdi.ucm.es/ELP/Esteganografía Consulta: 1 de Març de 2023.

[3] Omer Rosenbaum. "How to Use Scapy – Python Networking Tool Explained". [En línia]. Disponible: https://www.freecodecamp.org/news/how-to-use-scapy-python-networking/ Consulta: 2 de Març de 2023.

[4] Chema Alonso. "Ataques en redes IPv4&IPv6: Network Packet Manipulation con Scapy y NetfilterQueue". [En línia]. Disponible: https://www.elladodelmal.com/2021/02/ataques-en-redes-ipv4-network-packet.html Consulta: 1 de Març de 2023.

[5] "Interfaces gràficas de usuario con Tk.". [En línia]. Disponible: https://docs.python.org/es/3/library/tk.html Consulta: 21 de Maig de 2023.

[6] "Manual snort". [En línia]. Disponible: http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node9.html Consulta: 20 de Maig de 2023.

[7] "Plex Media Server, instala este servidor de medios en Ubuntu 20.04". [En línia]. Disponible: https://ubunlog.com/plex-media-server-instalacion-ubuntu-20-04/ Consulta: 20 de Maig de 2023.

[8] Iana. "Transmission Control Protocol (TCP) Parameters". [En línia]. Disponible: https://www.iana.org/assignments/tcp-parameters/tcp-parameters.xhtml Consulta: 19 de Maig de 2023.

[9] RFC 7323. "Transmission Control Protocol (TCP) Parameters". [En línia]. Disponible: https://www.rfc-editor.org/rfc/rfc7323.html Consulta: 19 de Maig de 2023.

[10] Wojciech Mazurczyk and Krzysztof Szczypiorski. "Steganography in Handling Oversized IP Packets". [En línia]. Disponible: https://arxiv.org/ftp/arxiv/papers/0907/0907.0313.pdf Consulta: 2 de Març de 2023.

[11] Youtube. "Curso de Redes10.7Protocolo TCP:número de secuencia". [En línia]. Disponible: https://www.youtube.com/watch?v=drBwqN038vM Consulta: 1 d'Abril de 2023.

[12] Youtube. "#7. Números de secuencia, reconocimiento y números de secuencia inciales". [En línia]. Disponible: https://www.youtube.com/watch?v=4L7kKKE2sMc Consulta: 5 d'Abril de 2023.

[13] Punam Bedia, Arti Duab. "Network Steganography using the Overflow Field of Timestamp Option in an IPv4 Packet". (2019).Third International Conference on Computing and Network Communications (CoCoNet'19).

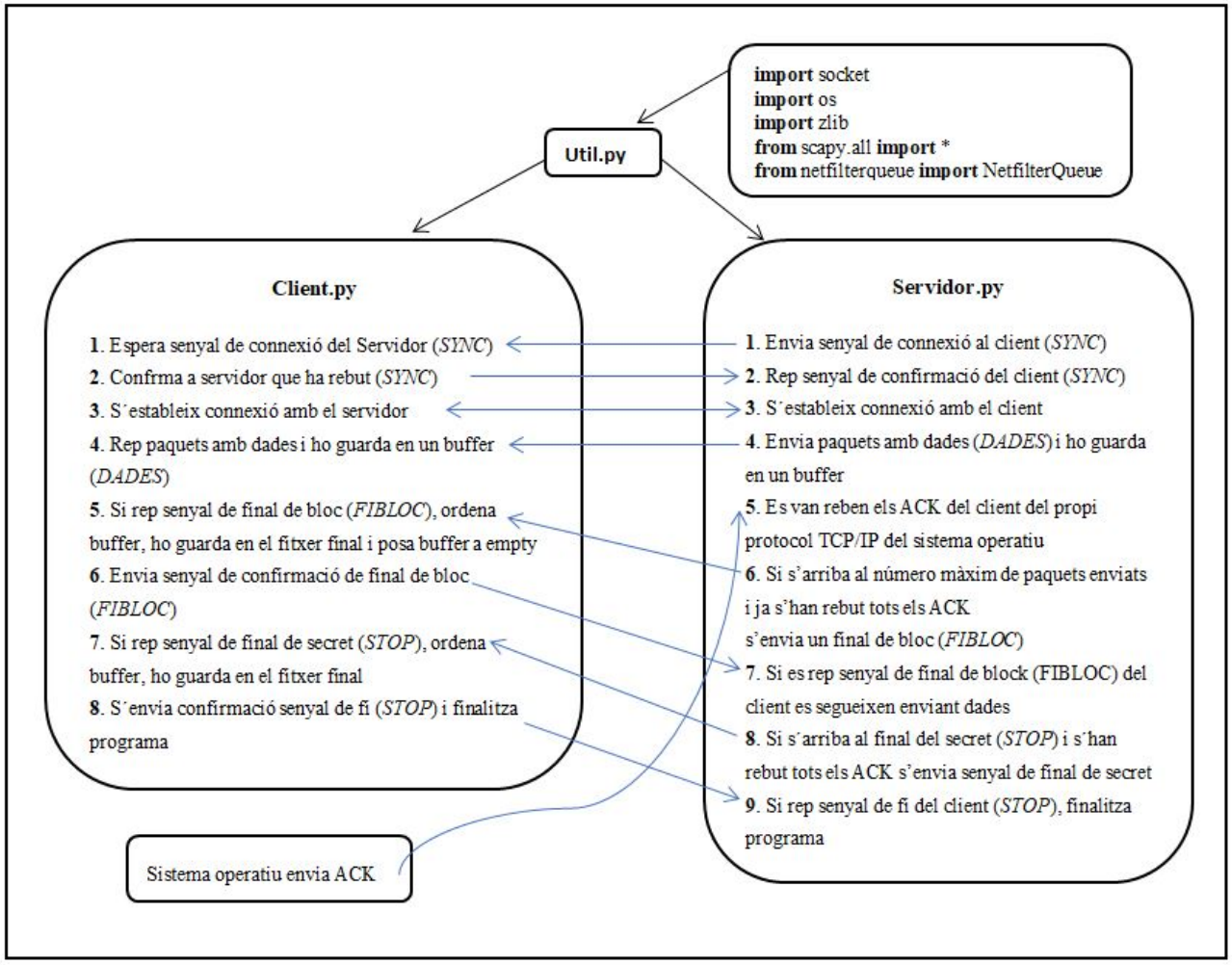
[14] UAB Servei de Biblioteques. "Citacions bibliogràfiques segons el model IEEE". [En línia]. Disponible a: https://ddd.uab.cat/pub/guibib/128946/modelIEEE_a2018.pdf Consulta: 15 de Juny de 2023.

A APÈNDIX

A.1 Interfície gràfica d'usuari



A.2 Disseny del sistema esteganogràfic



A.3 Codi per interceptar paquets

```
def iptables_on(esclient,iplocal,port):
    # Desviem paquets de sortida TCP amb ip origen la maquina local
    if esclient:
        order="iptables -A OUTPUT -p tcp -s " + iplocal + " --dport "+str(port)+" -j NFQUEUE --queue-num 1"
        os.system(order)
    # Desviem paquets d'entrada' TCP amb ip desti la maquina local
        ordre="iptables -A INPUT -p tcp -d " + iplocal + " --sport "+str(port)+" -j NFQUEUE --queue-num 1"
        os.system(ordre)
    else:
    # Desviem paquets de sortida TCP amb ip origen la maquina local
        order="iptables -A OUTPUT -p tcp -s " + iplocal + " --sport "+str(port)+" -j NFQUEUE --queue-num 1"
        os.system(order)
    # Desviem paquets d'entrada' TCP amb ip desti la maquina local
        ordre="iptables -A INPUT -p tcp -d " + iplocal + " --dport "+str(port)+" -j NFQUEUE --queue-num 1"
        os.system(ordre)
```