Disseny i implementació d'un sistema de comunicacions secretes basat en esteganografia de xarxa sobre el protocol IP

Enric Aynès Riba

Resum —Aquest article presenta el disseny, la implementació i resultats experimentals d'un sistema de comunicació esteganogràfic que està inspirat en el mètode de Ping Tunnel, el qual consisteix en la transferència secreta d'informació utilitzant les peticions ICMP Echo Request i les respectives respostes ICMP Echo Reply. La idea principal del disseny està basada en utilitzar certs camps de les capçaleres IP i ICMP per tal de transferir informació amb l'objectiu que un sistema de detecció d'intrusos no pugui captar que s'està efectuant una comunicació secreta a la xarxa. El sistema està pensat per a una via de comunicació emissor-receptor mitjançant un control de recepció que compta amb una finestra variable per tal de millorar la velocitat de transmissió. Per poder avaluar la implementació, s'ha realitzat una transferència d'informació en una xarxa que disposa d'un sistema de detecció d'intrusos que permet veure els límits de la implementació fins a ser detectada la comunicació. El sistema dissenyat té una capacitat esteganogràfica de 0.14 bytes per byte enviat i arriba a una velocitat de transmissió estable de 400bytes/segon.

Paraules clau — esteganografia de xarxa, capçalera IP, capçalera ICMP, snort, patrons ocults

Abstract —This paper presents the design, the implementation and the experimental results of an steganographic communication system that is inspired by the Ping Tunnel method, which consists in transferring secret information by using ICMP Echo Request and the correponding ICMP Echo Reply. The design's main idea consists in using some IP and ICMP heads to transfer information with the purpose that an intrusion detection system can not find that a secret communication is being carried out on the network. The system is designed for a sender-receiver way throughout a reception control that uses a variable window in order to improve the transmission rate. To evaluate the implementation, an information transfer on a network that has an intrusion detection System has been carried out, which allows finding the implementation limits until the communication is detected. The designed system has a steganographic capacity of 0.14 bytes per byte sent and reaches a stable transmission speed of 400 bytes/second.

Index Terms — network steganography, IP header, ICMP header, snort, hidden paterns

1 Introducció

L'esteganografia és la pràctica d'ocultar un missatge secret dins d'un portador que no és secret [1], el qual pot ser gairebé qualsevol cosa, des de documents fins a vídeos.

L'objectiu principal d'aquesta via de comunicació consisteix a ocultar i enganyar, ja que és una forma de comunicació encoberta i pot implicar l'ús de qualsevol mitjà per tal d'ocultar missatges. Cal destacar que no és una forma de criptografia, perquè no implica codificar dades o alterar l'estructura del missatge, sinó que és una manera d'ocultar dades que es pot executar de manera intel·ligent. A més, la criptografia és una ciència que permet en gran mesura la privacitat i, en canvi, l'esteganografia és una pràctica que es realitza mitjançant el secret i l'engany.

- *E-mail de contacte: 1456455@uab.cat*
- *Menció realitzada: Enginyeria de Tecnologies de la Informació*
- *Treball tutoritzat per: David Megías Jiménez*
- *Curs 2022/23*

Actualment, alguns exemples esteganogràfics es basen a incrustar textos secrets dins d'una imatge, amagar un script dins d'un document de Word o Excel o, fins i tot, modificar els protocols de xarxa per dur a terme una comunicació secreta.

Hi ha tres elements que són essencials en l'ocultació d'informació [2]: la capacitat, la seguretat i la solidesa. La primera, consisteix en la quantitat d'informació que es pot amagar; la segona, es refereix a la incapacitat que té un detector d'intrusos per detectar informació oculta i, la tercera, representa la quantitat de modificació que el mitjà de cobertura pot resistir abans que la informació oculta es corrompi. Més concretament, l'esteganografia de xarxa utilitza el tràfic visible de la xarxa com a portador de les dades secretes.

Les tècniques d'esteganografia en xarxa es poden classificar en diversos mètodes d'emmagatzematge i de temporització en funció de com es codifiquin les dades secretes al portador.

D'una banda, els mètodes d'emmagatzematge són aquells que amaguen informació i utilitzen dades de l'usuari mitjançant la modificació de dades en camps específics de protocol, com ara els bits no utilitzats d'una capçalera. Es distingeixen dos tipus de mètodes d'emmagatzematge: els que mantenen l'estructura del portador i els que la modifiquen. De l'altra, els mètodes de temporització amaguen informació a través de modificar els temps dels missatges o paquets del protocol. Existeixen certes tècniques de temporització que són agnòstiques al protocol de la xarxa portadora, fet que implica que no depenen de les característiques d'aquest. En canvi, n'hi ha d'altres que són conscients del protocol i depenen de determinats camps de capçalera i de la semàntica del protocol portador.
Paral·lelament, existeixen solucions híbrides, les quals combinen les característiques d'ambdós mètodes. Aquestes són resultat de l'evolució de la complexitat de les xarxes i els protocols que les regeixen.

D'altra banda, per tal de detectar la comunicació esteganogràfica, s'utilitzen eines d'estegoanàlisi, tant per a fitxers com per a xarxes senceres. Per aquest darrer aspecte, s'utilitzen els sistemes de detecció d'intrusos (IDS), ja que els sistemes d'esteganografia poden representar una amenaça de seguretat. Un IDS és un dispositiu o una aplicació que controla una xarxa per detectar activitats malicioses o infraccions a les polítiques d'ús [3]. Concretament, per a poder detectar la comunicació esteganogràfica de xarxa s'utilitzen els IDS de xarxa (NIDS), que s'encarreguen d'analitzar el trànsit de la xarxa.

En aquest projecte es desenvolupa un sistema de comunicació secreta entre dos punts finals a través d'aprofitar les característiques del protocol IP. Més concretament, s'envia informació secreta en datagrames IP modificats, utilitzant una tècnica esteganogràfica d'emmagatzematge, la qual manté l'estructura del portador a través d'incrustar informació secreta en un conjunt de bits que no són necessaris per a la comunicació , de manera que la informació secreta pugui passar desapercebuda per les eines d'anàlisi de xarxes, com ara els sistemes de detecció d'intrusos de xarxa.

2 Objectius

El plantejament del projecte s'ha centrat en seguir l'objectiu principal de l'esteganografia: amagar el procés de la comunicació [4], és a dir, amagar l'existència de l'intercanvi de dades.

Els objectius que han guiat l'elaboració del projecte es detallen a continuació:

- Dissenyar un sistema de comunicacions secretes basat en esteganografia de xarxa sobre el protocol IP
- Implementar el disseny proposat mitjançant la creació d'una aplicació.
- Comprovar la indetectabilitat del sistema esteganogràfic dissenyat a través d'un IDS.
- Mostrar els resultats obtinguts i detectar les mancances del sistema.

3 Aquitectura de la xarxa

Per tal de plantejar el disseny de la comunicació esteganogràfica, s'ha de tenir en compte principalment la xarxa on es pretén realitzar la transferència d'informació. En aquest cas, es tracta d'una xarxa bàsica, que podria ser una xarxa domèstica o d'oficina. Aquesta està formada per quatre elements: el host transmissor, el host receptor, un router per a comunicar els 2 hosts i un altre host amb l'IDS instal·lat per analitzar la xarxa (veure Fig. 1).

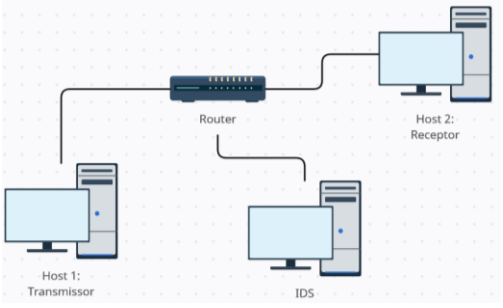

Fig. 1: Esquema de la xarxa

4 Disseny

El disseny proposat està inspirat en el Ping Tunnel [7], una aplicació que permet encapsular de manera fiable connexions TCP a un host remot mitjançant paquets de sol·licitud i resposta d'Echo ICMP. Aparentment, pot semblar una recurs amb poques aplicacions pràctiques però, en certs casos, pot resultar molt útil i perillós a la vegada, ja que pot arribar a ser el canal d'un atac. En aquest cas, s'utilitzarà l'enviament ICMP Echo Requests i d'ICMP Echo Replies per tal de dur a terme una comunicació secreta.

Aquesta proposta consisteix en utilitzar diferents camps de les capçaleres per a incrustar-hi el missatge. D'aquests camps, no se n'ha d'alterar ni l'aparença ni el funcionament de protocol ICMP ja que, si fos així, podria resultar més fàcil detectar-ne l'ús fraudulent.

Així doncs, per tal que la informació passi desapercebuda, s'aprofitaran els següents bytes: 2 bytes de la capçalera dels paquets IP i 4 bytes de la capçalera dels paquets ICMP, els quals seran substituïts pel missatge secret.
Aquests bytes seleccionats corresponen als camps següents:
De la capçalera IP (veure Fig. 2):

- Identificador: 16 bits que contenen el valor identificador del paquet.

De la capçalera ICMP (veure Fig. 3):

- Identificador: 16 bits que contenen el valor identificador del paquet.
- Número de seqüència: 16 que estableixen el número de seqüència de cada host.

Aquests camps seran utilitzats tenint en compte les següents premisses:

- Poden prendre qualsevol valor a través d'utilitzar els 16 bits que els hi pertanyen.
- No és necessari que els camps segueixin un format concret tal com ho faria el protocol Ping, ja que es monitoritzarà la comunicació.
- Quan la comunicació hagi de passar a través del router, aquest no modificarà el valor dels camps tal com passaria, per exemple, amb el camp TTL.
- En cas de fer-ne una comprovació mitjançant el Checksum, no en detectaria cap error.

0-3	4-7	8-15	16-18	19-31
Versió	Mida Capçalera	Tipus de Servei	Longitud Total	
Identificador			Flags	P. de Fragment
Time To Live		Protocol	Checksum	
Adreça IP d'Origen				
Adreça IP de Destí				
Opcions				

Fig. 2: Capçalera del protocol IP

0-7	8-15	16-18	19-31
Tipus	Codi	Checksum	
Identificador		Número de seqüència	

Fig. 3: Capçalera del protocol ICMP

4.1 Estructura del missatge secret

Aquest apartat comprèn explícitament com s'introdueix la informació a enviar i l'estructura que segueix.

4.1.1 Bytes disponibles

Un cop escollits els camps que s'utilitzaran, s'obté un bloc de bits on es pot incrustar informació (veure Fig. 4)

0-15	16-31	32-47
Identificador (IP)	Identificador (ICMP)	Número de seqüència (ICMP)
(2 bytes)	(2 bytes)	(2 bytes)

Fig. 4: Bits utilitzables

A través d'aquesta selecció de camps, s'obté un ample de banda de 6 bytes de dades per cada paquet enviat.

Tenint en compte que cada paquet disposa de 6 bytes, serà necessari enviar diversos paquets per a poder transmetre la informació secreta. Per aquesta raó, s'ha dissenyat un mecanisme per identificar cada paquet individualment, fet que permet reconstruir el missatge secret ordenadament.

El mecanisme dissenyat compta amb una capçalera que s'adjunta a cada paquet, la qual permet que el receptor pugui confirmar la rebuda d'informació.
Així doncs, caldrà reservar un conjunt de bits per al disseny d'una capçalera que permeti solventar aquesta casuística.

4.1.2 Capçalera

El disseny de la capçalera ha passat per un procés de desenvolupament, el qual ha implicat diferents canvis. A continuació, es mostren els passos que s'han seguit i els motius pels quals s'ha anat modificant.

Es parteix de la base que la capçalera del sistema ha de complir la funció d'identificar la informació rebuda i permetre informar al transmissor que s'han rebut els paquets correctament.

Tal com es pot veure a la Fig. 5, el primer disseny utilitzava només 1 byte. L'emisor, disposava de 3 bits per indicar el número de seqüència, 3 bits per indicar la confirmació esperada, 1 bit per indicar l'inici i 1 bit per indicar la fi de la comunicació, i el receptor, disposava de 3 bits per indicar el número de paquet que havia rebut, 3 bits per indicar el paquet esperat, 1 bit per indicar l'inici i 1 bit per indicar la fi de la comunicació.

	0	3	6	7	8
Enviament	Número de SEQ		Número de #ACK	Inici	Fi
Recepció	Número de ACK		Número de #SEQ	Inici	Fi

Fig. 5: Primer disseny de capçalera

Després de fer proves, es va veure que aquesta capçalera permetia poques iteracions fins reiniciar el número de seqüència a 0. A més, es va arribar a la conclusió que era innecesari enviar la confirmació que s'esperava rebre (#ACK), ja que no aportava informació necessària i tampoc hagués permès enviar grups de ICMP Echo requests més grans de 8.

El següent disseny de capçalera també disposava d'1 byte, però aquesta vegada estava distribuïda de la següent manera: l'emisor, utilitzava 6 bits per enviar el número de seqüència i mantenia els bits d'inici i fi de la transmissió; el receptor, utilitzava 6 bits per enviar el número de paquet que havia rebut i mantenia també els bits d'inici i fi de la transmissió. (Veure Fig. 6)

	0	3	6	7	8
Enviament	Número de SEQ			Inici	Fi
Recepció	Número de ACK			Inici	Fi

Fig. 6: Progrés en el disseny de la capçalera

Aquesta segona capçalera ja permetia més iteracions fins a reiniciar el número de seqüència, concretament 64, però encara tenia alguna mancança. Per exemple, pel que fa al receptor, el camp d'inici del les respostes no s'usaria mai, ja que no és qui iniciaria la comunicació.

Després d'haver provat els darrers dissenys, els quals implicaven tenir una capçalera pròpia per al control de recepció, es va veure que si s'observaven els diferents missatges ICMP que s'enviaven, resultava sospitós, esteganogràficament parlant, anar veient com en diferents paquets IP el camp Identificador era totalment aleatori i no seguia cap ordre. A més, tots els identificadors eren números parells.

Així doncs, per tal que la comunicació pogués passar encara més desapercebuda, es va optar per utilitzar el camp Identificador de la capçalera IP com a capçalera del sistema, però introduint-hi una nova modificació.

Per aquesta raó, en el disseny actual s'utilitzen els 16 bits del camp identificador com a capçalera, els quals estan distribuïts de la següent manera: l'emisor, disposa de 15 bits per enviar l'identificador i 1 bit per per indicar la fi de la comunicació; el receptor, també disposa de 15 bits per notificar el paquet que s'espera rebre i 1 bit per confirmar la fi de la comunicació. (Veure Fig. 7)

	0	16
Enviament	Fi	Número de SEQ
Recepció	Fi	Número de ACK

Fig. 7: Disseny de la capçalera

Aquesta capçalera permet mantenir una aparença de normalitat, on l'identificador IP de cada paquet va augmentant progressivament i no és fins a enviar l'últim paquet que es marca el bit de "Fi". Aquesta distribució permet també realitzar 32768 iteracions fins a reiniciar el número de seqüència.
Per contra, augmenta la mida de la capçalera i, conseqüentment, es redueix l'ample de banda per a informació secreta que disposa el disseny.

4.1.3 Missatge transportat

Els camps "Identificador" i "Número de seqüència" de la capçalera ICMP estan pensats per a transportar la informació desitjada. Per tant, els bits utilitzables queden estructurats seguint la Fig. 8.

0-15	16-31	32-47
Identificador (IP)	Identificador (ICMP)	Número de seqüència (ICMP)
Capçalera (2 bytes)	Informació (2 bytes)	Informació (2 bytes)

Fig. 8: Estructura dels bits utilitzables

Aquesta configuració permet un ample de banda de 4 bytes/paquet. Per posar en context, per enviar un arxiu d'1 kilobyte es necessiten 250 paquets o per enviar un arxiu d'un megabyte en 250000 paquets aproximadament. Així i tot, per a un missatge de text concís només es necessiten uns 10 paquets d'informació.

Per definir la velocitat de transmissió, cal veure l'apartat d' "Implementació", en el qual es detalla la quantitat de paquets que es poden enviar cada segon. Cal tenir en compte que omplir una xarxa amb molt trànsit pot ser vist com un atac de denegació de servei i aquest fet alertaria l'IDS.

4.2 Protocol d'enviament

L'enviament de la informació passa per un llarg procés que compta amb diferents fases.

1. Obtenir el contingut a enviar, ja sigui un missatge de text, una imatge o un document.
2. Convertir el contingut en algun element enviable, concretament en bytes.3
3. Xifrar els bytes a enviar amb la finalitat que no hi hagi els bytes en clar incrustats en els paquets ICMP.
4. Procedir a l'enviament dels paquets respectant la finestra lliscant establerta.
5. Escoltar la xarxa a l'espera de confirmació del receptor dels paquets enviats.
6. Una vegada enviat el darrer paquet, esperar la confirmació i acabar l'execució.

4.3 Protocol de recepció

Per tal de rebre els missatges i extreure la informació incrustada, cal seguir els passos següents:

1. Observar el tràfic que circula per la xarxa.
2. Un cop rebuts els missatges ICMP, comprovar que estiguin dirigits al nostre host i que provinguin del host transmissor.
3. Extreure la informació dels camps adequats de cada paquet i encadenar-la de manera ordenada.
4. Enviar al transmissor la confirmació dels paquets rebuts.
5. Repetir els punts 2, 3 i 4 les vegades necessàries per rebre tota la informació fins a aconseguir un paquet amb el bit de "Fi" activat.
6. Un cop s'ha rebut l'ultim paquet, enviar els bytes cap al desencriptador.
7. Desxifrar els bytes.
8. Convertir els bytes desxifrats en el format d'informació correcte.
9. En aquest punt ja es pot veure la informació secreta.

4.3.1 Recepció fora de línia

D'altra banda, també s'ha dissenyat un algorisme per extreure un missatge secret d'un arxiu que contingui paquets de dades d'una xarxa, com pot ser un arxiu .pcap obtingut amb algun tipus d'analitzador de dades, com el Wireshark [9]. Aquest algorisme permetria enviar, en qualsevol moment, tot el missatge secret a través de la xarxa. En cas que des del receptor s'hagi anat guardant el contingut que ha passat per aquesta, es podria recuperar el missatge secret. En aquest punt, però, no hi ha un control de recepció ni tampoc es contempla la pèrdua de paquets, una

hipòtesi a contemplar es enviar diverses vegades el missatge per contemplar que un paquet de cada varis pot ser extraviat o eliminat pel router.

5. IMPLEMENTACIÓ

Per tal de posar a prova el disseny plantejat, s'ha programat una aplicació per a consola en llenguatge python. Per a realitzarla-la, s'ha utilitzat la llibreria Scapy [8], la qual està dissenyada per a la manipulació de paquets. També, permet crear, manipular i falsificar paquets d'un ampli nombre de protocols, permetent enviar-los, capturar-los i emparellar peticions i respostes, a més de realitzar escaneigos i/o atacs a la xarxa. Scapy pot ser executat tant en Linux, Windows com en Mac OS..

La programació d'aquesta aplicació ha estat ideada per tal de complir els següents requisits.

- Proporcionar una comunicació senzilla per a l'usuari mitjançant una interfície gràfica.
- L'usuari trasnmissor ha de ser capaç d'escollir el missatge o informació a enviar.
- El receptor ha de ser capaç d'extreure i mostrar la informació secreta.
- El receptor ha de ser capaç de confirmar a la font que el missatge s'ha rebut correctament.
- L'aplicació és la mateixa tant per a transmissor com per a receptor. És l'usuari qui decideix si vol enviar o rebre informació.

Per tal que l'aplicació funcioni correctament, el transmissor i el receptor necessiten conèixer l'IP de la màquina amb la qual es vol realitzar la comunicació.

5.1 Menú principal

El menú principal s'ha creat per tal que l'usuari disposi d'una idea generalitada del contingut de l'aplicació i pugui seleccionar l'acció que vulgui realitzar. Veure Fig. 9

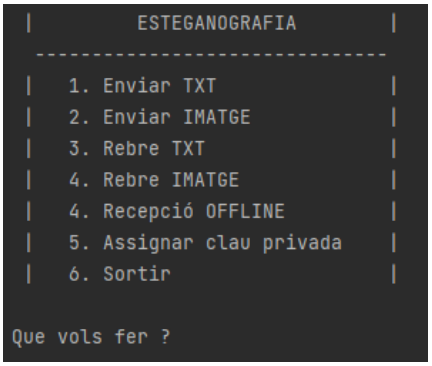

Fig. 9: Menú principal de l'aplicació

En aquest espai, l'usuari pot escollir si vol enviar o rebre un missatge de text, una imatge, assignar la clau privada del xifrador (veure més endavant l'apartat "Xifrador"), seleccionar l'algorisme de recepció fora de línia o simplement acabar amb l'execució del programa.

A través de seleccionar numèricament l'opció a realitzar, comença l'execució del programa. En cas que l'usuari no seleccioni correctament una opció, es torna a mostrar el menú principal.

5.2 Algorisme d'enviament

De la mateixa manera que el disseny de la capçalera, la transmissió de dades també ha passat per diferents processos d'elaboració a mesura que el projecte ha anat avançant.

Primerament, es bolcaven tots els paquets a la xarxa esperant que el receptor no tingués cap problema ni que es perdés cap paquet en el transcurs de la transmissió. Per tal que fos així, es va implementar un bucle que construïa paquets i els enviava a la xarxa. Aquest mètode, però, no es realista, ja que en una transmissió a vegades es poden perdre paquets o inclús ser descartats pel router.

Seguidament, es va entendre la necessitat que el receptor disposés d'algun mecanisme de confirmació que permetés comunicar al transmissor que estava rebent correctament la informació. Es va implementar un sistema de control que, cada vegada que enviés un paquet, el transmissor esperava la confirmació. Cal tenir en compte que el temps de transmissió d'un paquet d'extrem a extrem pot variar depenent de la xarxa on es troben el transmissor i el receptor i aquest pot ser elevat. Així doncs, aquest algorisme és poc eficient, ja que el temps de transmissió d'un paquet d'extrem a extrem provocava un temps de transmissió elevat.

Finalment, l'algorisme actual compta amb un control de transmissió amb una finestra adaptable. Aquest fet, implica que es pugui modificar la quantitat de paquets que s'envien abans de rebre la confirmació. El fet que sigui adaptable permet posar a prova el sistema envers al IDS i veure on troba el límit abans de ser descoberts. Modificant la finestra, també es disminueix els temps de transmissió, ja que requereix menys temps d'espera per a transmetre tota la informació. A continuació, es mostra l'algorisme d'enviament final.

Algorisme 1 Enviament

```
1:  function EnviarMissatgeControlCapçalera
2:      while not finalTransmissió
3:          while finestra > 0
4:              capçalera = setCapçalera(offset)
5:              p1 = missatgeSecret [offset:offset+2]
6:              p2 = missatgeSecret [offset+2:offset+4]
7:              send(paquet(capçalera, p1, p2))
8:              actualitzaFinestra()
9:          if finestra == 0
10:             sniff(xarxa)
11:                 actualitzaOffset()
```

5.3 Algorisme de recepció

Pel que fa a la recepció, convé que destacar que també ha passat per diferents etapes a mesura que avançava la programació del disseny.

En un inici, simplement escoltava la xarxa i encadenava un rere l'altre els paquets rebuts fins a obtenir una notificació del darrer paquet a rebre, però no tenia cap mena de control sobre els paquets perduts.

Degut a la necessitat de tenir un control de recepció, l'evolució de l'algorisme va comportar el desenvolupament d'un sistema de confirmació dels paquets rebuts, que els analitzava i confirmava al transmissor replicant un ICMP Echo Reply introduïnt l'informació secreta als mateixos llocs que el transmissor. L'aspecte negatiu d'aquest algorisme es troba en l'enviament d'un paquet com a resposta per cada paquet rebut.

Finalment, l'algorisme de recepció implementat en aquest projecte compta amb un control de recepció que té la capacitat de rebre un nombre variable de paquets fins a enviar la confirmació de rebuda. Això permet al transmissor enviar tants paquets com estigui estipulat sense esperar la resposta del receptor i, posteriorment, confirmar la rebuda del bloc de paquets inspeccionats. La imatge següent mostra la idea d'algorisme de recepció implementat.:

Algorisme 2 Recepció

```
1:  function RebreMissatgeControlCapçalera
2:    while not finalTransmissió
3:      sniff(xarxa)
4:        if paquet.src == "IP TRX" &&
                paquet.dst == "IP RC"
5:          capçalera = paquet.IP.id
6:          p1 = paquet.ICMP.id
7:          p2 = paquet.ICMP.seq
8:          if capçaleraOkey()
9:            missatgeSecret += p1 + p2
10:           actualitzaFinestra()
11:     if finestra == 0
12:       send(paquet(capçaleraESP))
```

5.3.1 *Reordenament de datagrames*

Per tal d'evitar retransmissions en casos que els paquets d'un bloc arribin desordenats al receptor, l'algorisme de recepció disposa d'una eina per no descartar directament aquests paquets.

En una primera instància, es guarden en memòria els paquets que no compleixen la condició de ser el paquet esperat. Una vegada s'acaben d'analitzar els que queden, el programa torna a revisar la llista de paquets desordenats per si algun paquet és el següent paquet esperat. Gràcies a la implementació d'aquest algorisme, s'augmenta la velocitat de transmissió i s'alleugereix la càrrega a la xarxa, ja que s'evita la possible retransmissió de paquets.

5.4 Xifratge

Per tal d'acomplir el punt 3 de l'apartat 4.2: "Amb la finalitat que no hi hagi els bytes en clar incrustats en els paquets ICMP, xifrar els bytes.", s'utilitzarà un sistema de xifratge de flux Salsa20 [5], que facilita la feina, ja que no es necessita una mida concreta de bloc per xifrar ni una clau molt extensa per tal que el missatge no es vegi a simple vista. La clau privada del sistema de xifratge se l'han d'haver compartit l'emissor i el receptor prèviament, d'altra manera no podrien realitzar la comunicació.

Un desavantatge que planteja el Salsa20 és que afegeix 8 bytes extres al missatge a enviar per després realitzar la descodificació, fet que comporta enviar dos paquets ICMP més al total de la transmissió. Si es vol enviar un missatge breu de text, pot implicar un canvi notable, però és innocu si es parla de transmetre un document o una imatge.

5.5 Entorn real

Pel que fa a la posada en pràctica del disseny present, les proves s'han dut a terme en una xarxa domèstica on els dos hosts estan connectats via ethernet al router. A més, un d'aquests dos hosts compta amb un IDS instal·lat i operatiu. La xarxa Ethernet domèstica permet una velocitat de connexió de 100 Mb/s.

Durant l'execució del programa, els datagrames IP són enviats a través de la xarxa amb la informació secreta incrustada.

No.	Time	Source	Destination	Protocol	Length	Info
578	84.102280	192.168.1.43	192.168.1.49	ICMP	42	Echo (ping) request (no response found!)
579	84.104229	192.168.1.43	192.168.1.49	ICMP	42	Echo (ping) request (no response found!)
580	84.106464	192.168.1.43	192.168.1.49	ICMP	42	Echo (ping) request (reply in 583)
583	84.217325	192.168.1.49	192.168.1.43	ICMP	60	Echo (ping) reply (request in 580)
584	84.227971	192.168.1.43	192.168.1.49	ICMP	42	Echo (ping) request (no response found!)
585	84.230694	192.168.1.43	192.168.1.49	ICMP	42	Echo (ping) request (no response found!)
586	84.233786	192.168.1.43	192.168.1.49	ICMP	42	Echo (ping) request (reply in 587)
587	84.284953	192.168.1.49	192.168.1.43	ICMP	60	Echo (ping) reply (request in 586)

Fig.10 : Exemple comunicació esteganogràfica

La Fig. 10 mostra alguns datagrames enviats a través de la xarxa en el transcurs d'una comunicació esteganogràfica. Aquesta transmissió està configurada amb una finestra de 3 datagrames fins a esperar la confirmació. D'aquesta manera, es veu reflectit, ja que cada 3 Echo Requests es veu una Echo reply enviada des del receptor cap al transmissor. També, es pot observar en la figura com el propi Wireshark dóna per vàlides les respostes generades manualment, ja que identifica que l'últim Echo Request de cada bloc de paquets es contestat per un Echo Reply.

5.6 Snort

Snort [6] és un sistema de detecció d'intrusos de xarxa desenvolupat per Cisco. És un programari gratuït de codi obert que es fa servir per observar els paquets entrants i detectar aquells que són perillosos per al sistema. Snort duu a terme un seguit de comprovacions a partir de regles fàcils de crear i implementar en qualsevol tipus de sistema operatiu i qualsevol xarxa.

Per tal de posar a prova el sistema d'esteganografia disesenyat, s'ha instal·lat l'Snort en un host de la xarxa i s'ha configurat perquè analitzi el tràfic de la xarxa que s'utilitza per a la comunicació. Pel que fa a les normes, no se n'ha creat cap d'específica per detectar el procés de comunicació esteganogràfic dissenyat, sinó que s'han descarregat l'última versió de les normes de la comunitat d'Snort i s'han activat totes aquelles que fan referència al protocol ICMP per detectar un ús fraudulent del protocol.

5.7 Resultats

La capacitat esteganogràfica d'ocultació es pot definir com la quantitat màxima d'informació que es pot incrustar al portador amb l'objectiu d'establir una comunicació secreta. Per avaluar el disseny, s'ha analitzat l'ample de banda esteganogràfic i la velocitat de transmissió.

D'una banda, els ICMP Echo Request enviats a la xarxa tenen una mida de 42 bytes, dels quals 6 bytes són informació secreta, contant la capçalera i el missatge secret. Aquests valors impliquen tenir una ràtio de bits incrustats per byte enviat de 0,14 bytes.

D'altra banda, els ICMP Echo Reply enviats tenen una mida de 60 bytes, dels quals 2 bytes són informació secreta, és a dir, la capçalera de control. A diferència del cas anterior, la ràtio redueix a 0.03 bytes d'informació incrustada per byte enviat. En aquest aspecte, és més important fixar-se en la petició d'ICMP Echo Request, ja que és el portador del missatge secret.

Per tal de valorar el rendiment del disseny, s'ha executat el programa amb diferents configuracions, així com diferents quantitats de bytes transmesos. S'ha buscat la relació de paràmetres que permet enviar més quantitat de bytes en menys temps de transmissió i comprovar a través de l'Snort si salta algun avís.

A continuació, es mostra la Taula 1, on es pot observar la relació que s'estableix entre la mida de la finestra i la quantitat de bytes enviats.

	Mida finestra [paquets]					
Bytes	**1**	**2**	**4**	**10**	**20**	**40**
10	0.627	0.561	0.523	0.510	0.506	0.483
20	0.722	0.601	0.481	0.508	0.488	0.505
40	0.886	0.624	0.362	0.512	0.412	0.486
100	1.815	1.248	0.810	0.600	0.530	0.522
200	3.23	1.919	1.240	0.730	0.645	0.587
400	6.428	3.088	2.108	1.114	0.833	0.742
1000	15.438	9.200	4.964	2.651	1.908	1.340
2000	31.707	15.641	10.429	4.467	2.870	2.090
4000	63.412	31.526	17.350	8.456	5.436	4.035
	Temps d'execució [segons]					

TAULA 1: Temps d'execució d'una comunicació real

El temps de transmissió per a grans quantitats de bytes, com podria ser un document o una imatge, es veu molt afectat per la mida de la finestra. En canvi, no implica una afectació considerable si s'envien pocs bytes, com podria ser un missatge concís de text.

Tal com mostra la taula, la configuració més lenta és, en tots els casos, utilitzar una finestra lliscant d'un sol paquet. Això és degut a l'espera constant del programa a rebre un paquet, ja sigui del transmissor o del receptor. Augmentant la finestra a 2 paquets, disminueix la quantitat de paquets que s'evoquen a la xarxa en un 25%, si es passa a una finestra de 10, es redueix en un 45%. Fins i tot, una finestra de 40 paquets redueix la càrrega a la xarxa un 51,25%.

Així doncs, amb una mida de finestra de 10 paquets, es pot obtenir uns temps de transmissió per sota d'un segon fins als 400 bytes d'informació a transferir. Aquesta finestra és adequada per enviar missatges de text escrits directament a l'aplicació.

També, amb una mida de finestra de 40 paquets s'obtenen temps d'execusió d'entre 1 i 4 segons per a transmissions de 1000 a 4000 bytes. Aquesta finestra és més adequada per a transmissions d'arxius petits. Per a arxius de dimensions superiors, caldria estudiar l'ampliació de la mida de la capçalera.

Per obtenir la velocitat de transmissió, s'han agafat les dades de la Taula 1 i s'han dividit entre els bytes enviats. Tenint en compte que per a quantitats de bytes petites s'ha seleccionat la finestra de 10 paquets i per a quantitats de bytes més elevades la de 40, a continuació es poden observar les velocitats de transmissió de les configuracions esmentades (Taula 2).

	Mida finestra [paquets]	
Bytes	**10**	**40**
10	19,60	20,70
20	39,37	39,60
40	78,12	82,30
100	166,6	191,5
200	273,9	340,7
400	359,0	539,0
1000	377,2	746,2
2000	447,7	956,9
4000	473,0	991,3
	Velocitat [Bytes/segon]	

TAULA 2: Velocitat de transmissió d'una comunicació real

Tal com mostra la Taula 2, la configuració d'una finestra de 10 paquets, adient per a missatges de text, obté valors de fins a 359 bytes/segon.

D'altra banda, amb una finestra de 40 paquets, arriba a obtenir una velocitat de transmissió de gairebé 1000 bytes/segon.

Durant el transcurs de la posada a prova del sistema, l'Snort ha estat analitzant la xarxa seguint les normes de la comunitat i, en cap moment, ha saltat cap alerta relacionada amb la comunicació esteganogràfica.

Per comprovar el funcionament de l'Snort, s'ha implementat una norma que detecta quan circula per la xarxa un paquet ICMP i aquesta ha saltat correctament. (Veure Fig.11).

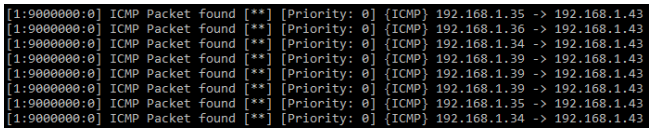

Fig. 11: Exemple alertes Snort

Amb aquesta comprovació, es pot evidenciar que l'IDS funciona correctament i no detecta la comunicació esteganogràfica.

6 Conclusió

En aquest treball, s'ha dissenyat i desenvolupat un sistema de comunicació esteganogràfica de xarxa sobre el protocol IP. Aquest sistema permet la comunicació entre dos usuaris sense que aquesta mateixa comunicació pugui ser detectada. Per tal de dur a terme aquesta transferència d'informació entre emissor i receptor, s'ha modificat la petició ICMP Echo Request i la seva resposta ICMP Echo Reply utilitzant una tècnica esteganogràfica d'emmagatzematge la qual manté l'estructura del portador.

També, s'ha dissenyat i progaramat una aplicació de consola amb Python, que permet posar a prova el sistema dissenyat en un entorn real. Aquesta aplicació compta amb un control de recepció que permet notificar el correcte desenvolupament de la transmissió, a més d'una finestra variable que permet ajustar l'aplicació a l'ús que se'n vol donar. Concretament, s'han escollit les finestres de 10 paquets i 40 paquets per a comunicacions de menys de 400 bytes i superiors respectivament, ja que són les que proporcionen un rendiment estable. obtenint velocitats de transmissió de fins a 1 KByte/s

Cal destacar que, gràcies a l'aplicació, s'ha pogut posar a prova el disseny i, tal com marcaven els objectius, la comunicació no ha estat detectada per l'IDS.

Finalment, es vol afegir que, aquest projecte, ha estat fruit de la investigació, d'un conjunt de proves assaig-error desenvolupades en diverses fases i de la tutorització pautada i guiada pel professorat ja que, prèviament a l'elaboració, no es disposava de coneixements relacionats amb l'esteganografia. Així i tot, s'han pogut aconseguir els objectius establerts a l'inici de l'article.

Agraïments

Els agraïments d'aquest article estan destinats principalment al tutor-mentor D.Megías, qui pacientment ha estat obert en tot moment a suggerir, proposar i supervisar la feina duta a terme per tal d'aconseguir els objectius establerts.

Bibliografia

[1] Kundur, D., & Texas (2003). "Practical Internet Steganography: Data Hiding in IP".

[2] Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander, Amir Houmansadr, and Krzysztof Szczypiorski. 2016. "Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures" (1st. ed.). Wiley-IEEE Press.

[3] (N.d.). Retrieved from https://www.snort.org/.

[4] W. Mazurczyk and K. Szczypiorski, "Steganography in Handling Oversized IP Packets," 2009 International Conference on Multimedia Information Networking and Security, 2009, pp. 559-564, doi: 10.1109/MINES.2009.246.

[5] D. (Ed.). (n.d.). Salsa20. Retrieved from https://www.pycryptodome.org/en/latest/src/cipher/salsa20.html.

[6] Beale, J., & Kohlenberg, T. (2007). Snort. Syngress Press.

[7] D. Stodle (2011, September). Ping tunnel: for those times when everithing else is blocked. Retrieved from https://www.mit.edu/afs.new/sipb/user/golem/tmp/ptunnel-0.61.orig/web/

[8] (N.d.). Retrived from https://scapy.net/

[9] Richard Sharpe, Ed Warnickle, Ulf Lamping. (2023, Genuary 30) Wiresherk User's Guide. Version 4.1.0.