Projecte de recerca bàsica o aplicada
PAC3 — Tercera Prova d'avaluació continuada

Cognoms: *Sanchez Cañadas*
Nom: *Oscar*

- Per a dubtes i aclariments sobre l'enunciat, adreceu-vos al consultor responsable de la vostra aula.
- Cal lliurar la solució **en un fitxer OpenDocument o PDF fent servir una de les plantilles lliurades conjuntament amb aquest enunciat. Adjunteu el fitxer a un missatge adreçat a l'espai d'avaluació de l'aula virtual**.
- El fitxer ha de tenir l'extensió *.odt* (OpenDocument) o *.pdf* (PDF) segons el format en què feu el lliurament.
- La data límit de lliurament és el **19 de gener** (a les 24 hores).

Respostes

Protocol d'Encaminament Anònim en xarxes descentralitzades

Universitat Oberta de Catalunya (UOC)
Autor: Sanchez Cañadas, Oscar. Estudiant de la UOC e Enginyer tècnic de telecomunicacions
Tutors: Rifa, Helena & Marquès, Joan Manuel. Professors d'informàtica i multimèdia
Gener de 2012, Barcelona, Catalunya, Espanya

RESUM

Proposem un protocol per realitzar l'encaminament anònim dins de les xarxes peer-to-peer, les xarxes descentralitzades.

Hem creat una petita aplicació per l'enviament d'un arxiu des d'un usuari que es connecta a l'aplicació fins un servidor que s'hi assignarà a aquest, el qual es donat per un Directory, que conté totes les direccions d'aquestes Entities.

També hem fet unes proves per comprovar el comportament de l'algorisme dins d'una aplicació, tal com el grau d'anonimat que aporta, o la latència d'aquest segons la mida de l'arxiu o el rendiment de cada Entity per paquet.

I finalment extraiem conclusions i proposem millores per l'algorisme per treballs futurs.

1 INTRODUCCIÓ

En les ultimes dècades l'ús generalitzat d'Internet ha incrementat l'interès sobre els mètodes per a la protecció de la privacitat i les comunicacions anònimes, tant de la comunitat acadèmica com del públic en general.

S'han proposat diversos dissenys del sistema en l'àmbit acadèmic, dels quals s'han implementat alguns i són utilitzats per diversos grups per tal de protegir la seva identitat a Internet. Encara que la encriptació pot ocultar el contingut de la informació en la xarxa, no és suficient per ocultar la direcció IP als altres usuaris.

La arquitectura dels sistemes descentralitzats no necessiten un servidor centralitzat. Aquest disseny pot evitar molts atacs que es fan a un servidor centralitzat, ja que cap atac pot trencar tots els nodes. A més aquest tipus d'arquitectura és capaç fer un balanç de la carrega per si mateix.

El nostre propòsit és dissenyar i implementar un sistema robust, d'alta disponibilitat amb contingut d'emmagatzematge lliure, i a més, ha de garantir la fiabilitat, la privacitat i l'anonimat.

2 JUSTIFICACIÓ

Actualment, un gran nombre de plataformes P2P, sistemes distribuïts i d'emmagatzematge compartit han adoptat mecanismes de recerca i enrutament basats en taules hash distribuïdes(DHT). DHT proporciona un sistema d'auto-organització amb un alt rendiment d'enrutament, precisió de recerca, alta escalabilitat i balanç de carrega automàtic.

DHT pot ser construït sobre una xarxa de voluntaris composta per equips d'usuaris finals oferta per els seus propietaris. La facilitat i la flexibilitat per convertir-se en membre d'una xarxa de voluntaris dona a aquestes xarxes un gran potencial computacional i d'emmagatzematge. No obstant això, aquest tipus de xarxes també s'enfronten a una gran quantitat de problemes de seguretat. Tots els nodes d'aquestes xarxes poden transmetre, rebre i redirigir dades, per tant podem arribar a ser molt vulnerables, ja que qualsevol node pot rebre la informació de qualsevol altre i poden haver nodes maliciosos, contingut infectat, espionatge de comunicacions i molts problemes relacionats amb la seguretat. Actualment existeixen molts problemes d'aquests tipus en aquestes xarxes, que podria ser pal·liat amb un protocol que proporcionés anonimat.

Per tant, en aquest projecte ens centrem en qüestions de fiabilitat i privacitat relacionades amb l'emmagatzematge de les dades. La investigació es basa en el disseny i la creació d'un protocol que ens permet garantir l'anonimat dels voluntaris dins d'una xarxa de distribució basada en DHT.

3 OBJECTIUS DE LA RECERCA

L'objectiu d'aquest projecte és el disseny i la implementació d'un sistema robust, d'alta escalabilitat i contingut lliure que resisteixi als intents de bloqueig d'adversaris poderosos. Tot el contingut emmagatzemat pot ser buscat públicament i disponible de la manera mes convenient possible.

Qualsevol pot publicar qualsevol contingut emmagatzemat a la xarxa, mentre que s'evita l'eliminació del contingut o la determinació de la verdadera identitat de les persones involucrades per part dels adversaris interns o externs. A més, els nodes membres que contribueixen a l'emmagatzematge no deurien tenir una manera fàcil de determinar el que estan emmagatzemant, garantint una denegació versemblant.

En resum, l'objecte principal d'aquest disseny és la creació d'una proposta d'un protocol d'anonimat que proporcioni privacitat al contingut emmagatzemat per els voluntaris, assegurant el seu anonimat, i confidencialitat, evitant qualsevol suplantació d'identitats.

4 VISIÓ GENERAL DELS SISTEMES D'ANONIMAT ACTUALS

Analitzem els algorismes basats en baixa latència perquè per la nostre recerca es basa en aquests tipus de sistemes.

4.1 Onion Routing

Onion routing és el disseny amb major prevalença per a les comunicacions anònimes.

Cada onion router manté una clau privada i una pública. La clau pública s'ha de donar a conèixer els clients. Per establir les connexions anònimes el iniciador construeix un túnel d'encriptació múltiple, o circuit, a través de la xarxa. Tots els missatges processats han d'estar encriptats asimètricament.

Els remitents en una xarxa onion routing només utilitzen la clau pública per establir un circuit de xifrat i després utilitzar la criptografia a través de la clau simètrica més ràpida per a transferir les dades reals.

El iniciador genera dos claus secretes simètriques, una per re-enviar i una altre per a contestar. El iniciador envia la onion al primer router en la ruta, que elimina la capa més externa del xifrat amb la seva clau privada. Cada servidor al llarg de la ruta repetirà la mateixa operació fins que la onion arribi al final de la seva ruta. Una vegada construït al circuit, el iniciador pot retransmetre tràfic utilitzant la clau simètrica generada en cada salt. [1]

4.2 PipeNet

L'algorisme PipeNet selecciona una seqüencia aleatòria dels servidors de la xarxa, similar a la selecció de ruta de la onion routing. Els clients doncs, creen un túnel xifrat múltiple mitjançant l'establiment d'una clau simètrica amb el primer salt, un túnel a través de la connexió xifrada i l'establiment d'un segon salt, i així successivament. [1]

Encara que el disseny de PipeNet és poc pràctic per a les xarxes reals com Internet.

4.3 Tarzan

Tarzan, Crowd i AP3 són sistemes d'anonimat molt similars.[3,5]

Els iniciadors en Tarzan construeixen una xarxa de circuits a través d'aquesta mitjançant la generació de claus simètriques per cada salt i encriptant-les amb les claus públiques de els servidors en el circuit.

Com el sistema Crowds, tots els participants en la xarxa retransmeten per altres usuaris.

Tarzan és un sistema flexible, transparent, descentralitzat i altament escalable. Utilitza túnels entre els nodes i és transparent tant per el client com per el servidor. Aquest sistema és capaç de proporcionar anonimat a les aplicacions existents sense necessitat de fer cap modificació. [6]

Tarzan utilitza un protocol peer-to-peer gossip per tal de compartir la informació sobre tots els servidors. Descobreix els servidors mitjançant la selecció d'un node al atzar de veïns que ja coneix, com AP3 i Crowd. [5]

El iniciador d'un túnel anònim crea un circuit a través de la xarxa Tarzan agafant aquest primer al atzar. El segon salt es selecciona mitjançant el conjunt escollit per el primer salt, i així successivament. El tràfic de aplicacions s'envia a través del circuit amb un xifrat per capes, molt similar al que ja hem comentat amb la onion routing. [1]

4.4 MorphMix

MorphMix és també similar a Tarzan, és un sistema d'anonimat peer-to-peer de baixa latència on tots els nodes de la xarxa fan de retransmissors. Aquest també utilitza múltiples capes per tal de fer el xifrat simètric abans de transferir les dades al llarg del circuit.

Un node *a* estableix una clau simètrica mitjançant l'enviament a *b* de la meitat de l'algorisme d'encriptació Diffie-Hellman amb la clau pública del

testimoni *w.* El node *b* transmet el xifrat cap a *w* que elimina la capa de encriptació i envia el resultat a *a.* Aquest genera la meitat de l'algorisme de Diffie-Hellman i l'envia a través de *b.* Les dues parts són capaços de generar la seva clau simètrica compartida. [1]

4.5 Salsa

Salsa és un sistema basat en DHT anònim per capes. Cada node disposa d'un ID determinat per realitzar la funció criptogràfica hash per obtenir la direcció IP. [3]

Les recerques recursives per rebre una clau pública i les claus de cada retransmissor es realitzen mitjançant la petició de l'iniciador d'una petició a *r* nodes, incloent-s'hi a ell mateix. El circuit s'anirà estenen aleatòriament.

Si un resultat d'una recerca és major que un llindar de distancia cap al ID destí, aquesta es descartarà.

El iniciador utilitza una funció hash criptogràfica pública per calcular el ID de cada direcció IP tornada. El ID del més proper serà la clau destí seleccionada, i els altres resultats seran descartats. [3]

La comprovació dels límits no serà eficaç en el cas de que l'atacant es faci amb el control de la clau del node més proper. [3]

4.6 Cashmere

Aquest sistema utilitza retransmissors virtuals composats per un conjunt de nodes per a la resistència. [3]

Cada grup de retransmissors ha de tenir un par de claus pública/privada, i a més cada membre del grup té que tenir una clau pública/privada del grup.

S'utilitzarà l'ID del grup com a clau per enviar els missatges, utilitzant un prefix. El root del grup és el responsable de processar els missatges en nombre del grup retransmissor. Per a detectar fallades i nodes maliciosos, Cashmere utilitza ACKs extrem-a-extrem. [3]

4.7 Tor

Tor és la segona generació del sistema Onion Routing creat per las limitacions del disseny original, afegint confidencialitat directa perfecta, control de congestió, servidors de directory, comprovació de la integritat, polítiques de sortida configura-ble, i un disseny pràctic per als serveis de localització a través de punts de trobada. [4]

Tor aporta confidencialitat, ja que ara utilitza un disseny incremental per a la construcció del camí, on el iniciador negocia les claus de sessió en cada salt successivament el circuit.

4.7.1 Millores respecte a Onion Routing

Es més fiable, ja que el iniciador sap quan un salt falla i pot tractar de estendre un nou node.

Millora la eficiència i l'anonimat al multiplexar fluxos TCP al llarg de cada circuit.

El iniciador pot enviar el tràfic als nodes fins a la meitat del circuit, evitant d'aquesta manera els atacs basats en l'observació del final del circuit.

Incorpora un control de congestió. Mitjançant ACK extrem-a-extrem pot mantenir l'anonimat mentre aquests nodes permeten detectar la congestió i poder controlar-la.

Tor proporciona servidors Directory signats per descobrir els nodes coneguts i els seus estats actuals.

Incorpora una verificació de integritat extrem-a-extrem, per a que cap node pugui canviar el contingut de la informació de les cel·les.

Proporciona uns punts de rendezvous i serveis ocults. Els clients negocien el punt rendezvous per connectar-se els serveis ocults. [4]

4.7.2 Servidors Directory

Els servidors de Directory són els responsables de la agregació i distribució de la informació registrada dels routers coneguts en la xarxa. La informació del Directory signat també pot ser reflexat per els altres routers de la xarxa amb la finalitat de reduir la carrega en els servidors Directory. [1]

La utilització d'aquests servidors és més simple i més flexible que utilitzar flooding. [4]

4.7.3 Establiment del circuit

El iniciador del circuit negocia les claus de sessió amb cada router en el camí del circuit mitjançant la negociació de DHT i la autenticació per RSA. [1]

Cara onion router manté una clau d'identitat que és utilitzada per signar els certificats TLS. La clau onion és utilitzada per desencriptar sol·licituds dels usuaris per la realització d'un circuit i la negociació de claus. [4]

4.7.4 Localització de serveis ocults

La localització de serveis ocults permet oferir serveis TCP, com un servidor web, sense la necessitat de revelar la direcció IP. Aquest tipus d'anonimat protegeix contra atacs DoS, ja que no és coneix la IP. [4]

4.7.5 Solucions actuals als problemes de Tor

Per tal de resoldre els problemes de escalabilitat en la xarxa Tor, es van proposar dos sistemes, NISAN i Torsk. Els dissenys incloent mecanismes per tal de mitigar els atacs de fugues d'informació. NISAN proposa incorporar l'anonimat en la pròpia recerca, i en canvi Torsk utilitza nodes buddy. [5]

4.8 NISSAN

Nisan proposa una construcció amb 3 retransmissors aleatoris des de una llista que ens proporciona el router per construir el túnel. Aquesta configuració podria permetre a l'atacant enllaçar la sortida de l'ultim node cap a l'iniciador, ja que l'ultim node està en contacte directament amb l'iniciador, per tant si aquest node es compromès, l'atacant pot trencar el túnel. [5]

Per a evitar això podem establir un circuit parcial del primer node i després amb un proxy buscar el segon node i després el tercer, i així successivament. Malauradament aquesta construcció per si sola és vulnerable als atacs publick- key modification i route capture attack. [5]

Una manera de mitigar els atacs de la primera construcció és utilitzar una ruta mes llarga. Encara que aquesta construcció podria incrementar la latència d'arrancada i fa el sistema mes vulnerable als atacs DoS. [5]

4.9 Torsk

Torsk requereix la selecció d'un nombre de nodes aleatori de tota la xarxa per a seleccionar la privacitat. Un atacant no pot associar l'objectiu de la recerca amb la consulta. Doncs, l'atac de seguiment hop-by-hop no pot ser aplicat i els buddies es mantenen en secret. [5]

Torsk proposa utilitzar un camí aleatori per la selecció de buddies. Després el proces buddy és realitzat off-line. A continuació Torsk sol·licita que buddies deuran ser de un us.

Finalment, Torsk aplica la veracitat del certificat de cada hop del camí aleatori, prevenint els atacs des de que disposa de camins aleatoris per incrementar la oportunitat de que els nodes maliciosos siguin seleccionats com a buddies.

5 ANÀLISI DELS SISTEMES ANÒNIMS P2P

5.1 Estructura del sistema

La arquitectura descentralitzada dels sistemes P2P pot mantenir més nodes usuaris i serveis dels que pot mantenir una arquitectura centralitzada. Per tant, la arquitectura descentralitzada és molt mes escalable, i la escalabilitat de la xarxa és un element molt important per a qualsevol sistema.

Si el radio dels nodes maliciosos és invariable, el grau d'anonimat del sistema incrementa amb l'augment de l'escala de la xarxa, ja que, si el nombre de nodes maliciosos és invariable, i la xarxa creix amb nodes honestos, la proporció de nodes maliciosos cada vegada serà més petita i per tant el grau d'anonimat serà més gran. De tal manera, si els nodes maliciosos augmenten, el grau d'anonimat sofrirà un decrement. La xarxa P2P és ideal per a construir una gran xarxa.

El grau d'anonimat sembla relativament alt inclús si l'entorn de la xarxa és molt negatiu. Això vol dir que si tots els nodes maliciosos poden només observar la comunicació però no poden formar part d'aquesta i totes les característiques de la comunicació de nodes honestos sembla igual, el sistema pot també mantenir un alt grau d'anonimat.
Quan un node necessita un servei anònim, primer aquest buscarà els nodes per DHT. Aquest proces pot ser víctima la fuga d'informació en les comunicacions anònimes. Si un usuari aconsegueix el interferir en aquest proces, aquest pot trobar a l'emissor, però no pot identificar el túnel que el emissor va crear. A més si l'ultim node és també maliciós, el destí real pot ser identificat, pel que l'atacant pot rastrejar a l'emissor real de aquest túnel utilitzant l'atac de sincronització o atac bridge.

La diferencia amb els sistemes P2P estructurats és que aquests utilitzen el protocol gossip per descobrir els nodes actius i seleccionar els retransmissors. Aquests sistemes són menys afectats per els nodes maliciosos que els sistemes no estructurats.

5.2 Mida del túnel

En el proces de routing, cada node retransmissor només coneix el seu predecessor i el seu successor, així que l'atacant necessita al menys controlar més de la meitat dels nodes en el túnel per a la seva reconstrucció.

Al principi, el grau d'anonimat incrementa ràpidament amb l'augment de la mida del túnel, però quan la mida excedeix a més de 10 salts, el grau d'anonimat comença a disminuir molt lentament. Això és perquè encara que un túnel sigui molt ampli i sigui molt difícil de controlar per l'atacant, contra

més salts hi sumem, mes possibilitats donem a l'atacant per unir-se al nostre túnel i esbrinar l'emissor. [2]

La mida variable del túnel pot proporcionar un grau d'anonimat més alt que una mida fixa.

5.3 Selecció dels nodes retransmissors mitjançant mergin of cluster

Amb el mètode mergin of cluster, el grau d'anonimat incrementa quan un clúster creix amb la escala de la xarxa, però aquest és encara molt relativament petit. La principal raó d'això és perquè el clúster és molt més petit que el total dels nodes de la xarxa i els clústers no cooperen entre ells, així que l'anonimat és mes petit. Si combinem els clústers, i si l'adversari vol trencar el túnel, aquest deurà unir-se a tots ells. Així que l'estratègia és que l'emissor únicament selecciona el següent node que és preferible en el disseny del sistema.

5.4 Estratègia de retransmissió

L'estratègia de retransmissió en la majoria dels sistemes és l'anomenada Onion routing. Quan el missatge es retransmès d'un node cap al node següent, aquest es encriptat amb la clau simètrica del següent node, i el següent node desencripta aquest missatge i ho torna a encriptar amb la clau del següent node i ho torna a enviar al següent node, i així fins arribar al node receptor. Per tant, si hi ha un node maliciós, aquest haurà de controlar el primer i l'ultim node si vol analitzar els missatges dins del túnel.

[Si el primer node del túnel és maliciós, aquest pot identificar la relació entre l'usuari i el servidor real. Si el primer node és honest, altre node en el túnel no pot identificar l'emissor real directament i només existeix una possibilitat 1/N (N nombre total de nodes) per esbrinar la identitat correcta de l'emissor. [2]

6 PROTOCOL D'ENCAMINAMENT ANÒNIM

Hem construït una aplicació descentralitzada per comunicar-nos amb un servidor, de manera que aquesta comunicació sigui anònima i que no hi hagi cap servidor central que gestioni la informació.

6.1 Aplicació descentralitzada per comunicacions anònimes

Dins d'aquesta aplicació hem creat una interfície per l'usuari (User Application) per tal de que aquest pugui enviar documents cap el servidor. Aquesta interfície fa la transmissió a través del protocol d'encaminament anònim que hem creat.

6.2 Serveis Entity

Hem creat Entities, que són les encarregades de retransmetre els paquets i gestionar els documents que els usuaris li envien. Cada Entity està replicada en varies instàncies per tal de garantir la disponibilitat de la informació en cas de fallades.

6.3 Servidors Directory

Hem creat un servidor Directory, que és l'encarregat de posar en marxa les Entities i d'assignar aquestes als usuaris.

6.4 Autoritat Certificadora

La creació d'una autoritat certificadora (CA) era indispensable per certificar la identitat dels usuaris i serveis que participen en la publicació del treball.

Es crea un certificat auto-signat tant per la CA com per el Directory i es fiquen dins d'un repositori de claus PKCS#12, on es guardaran també els certificats dels usuaris i les Entities.

6.5 Funcionalitat

L'usuari introdueix el seu treball a l'User Application. Aquest contacta amb el Directory a través d'una connexió SSL amb autenticació mútua per obtenir l'identificador de l'Entity on s'enviarà el treball tal i com es mostra a la figura 1.

El Directory assignarà una Entity que tingui lliure a aquest usuari, i marcarà aquesta Entity com “en curs”, i li retornarà una llista d'Entities i la seva Entity destí.

A continuació, l'User Application executa el Protocol d'Encaminament Anònim posant com a destí

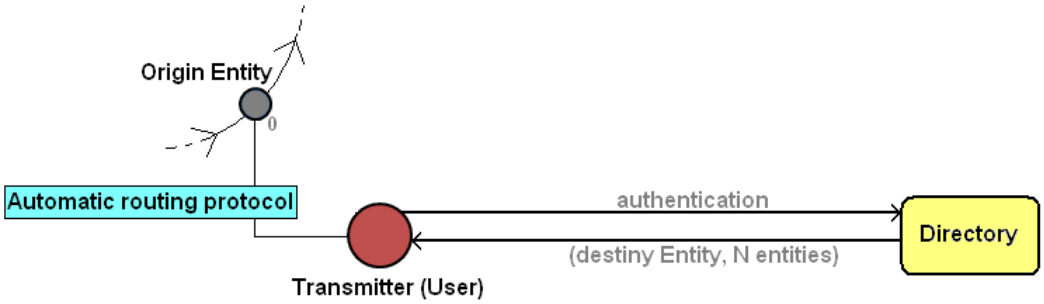

Figura 1. Autenticació i assignació Entity destí

l'identificador de l'Entity origen, una Entity que s'agafa aleatòriament entre tota la llista de les Entities.

L'Entity destí agafa el treball i l'emmagatzema, en canvi, encara que aquest no envia cap tipus de confirmació a l'usuari, serà la Entity origen, quan aparegui després de la Entity destí, l'encarregada de comunicar a l'usuari com a anat l'operació. Doncs, l'User Applications mostrarà el resultat a l'usuari per pantalla.

Finalment, després de realitzar la operació i l'User Application rebi la confirmació, el Directory marcarà la Entity com "lliure".

6.6 Algorisme del Protocol d'Encaminament Anònim

El protocol d'encaminament anònim segueix un format d'Onion routing en el qual un missatge es xifrat en múltiples capes. Cada capa conté la identitat i un paquet xifrat pel pròxim encaminador de la xarxa.

6.6.1 Mida del túnel

Al posar en marxa l'aplicació per tal de pujar un document al servidor, el Directory crea *N* identificadors aleatoris, amb els quals, l'User Application crea una cadena de M Entities, que serà la mida del túnel a utilitzar en el protocol.

Com els autors, Jia Zhang, Haixin Duan, Wu Liu i Jianping Wu, senyalen al seu article [2] *Anonymity analysis of P2P anonymous communication system*:

> "At the beginning, the anonymity degree rises rapidly with the tunnel length increasing, but when the length exceeds 10, the anonymity degree will not rise, but instead declines slowly. This is because although a long tunnel is hard for adversary to control, it also gives adversary more chances to join in and guess the sender. This phenomenon means frequent weak attacks can also make the system fragile. So, how to make tradeoffs to avoid frequent weak at- tacks should be considered by developers."

Van analitzar diferents mides i van veure que amb una mida molt petita el sistema donava molta facilitat a l'atacant de saber qui era el transmissor, i que encara que el túnel fos massa gran, el grau d'anonimat arribava a un punt on no podia créixer més ja que al incrementar el túnel, s'incrementava la possibilitat de que un atacant formés part del túnel.

Per tant, després d'analitzar la situació, l'autor ens proposa que la mida ideal del túnel és 10, encara que també ens comenten que una mida variable del túnel pot aportar un major anonimat al sistema que una mida fixa.

Per tant, en el nostre protocol, hem creat una mida del túnel variable, on es determina per el nombre d'alumnes.

$$N = (A^{1} * 2) + 10$$

D'aquesta manera millorem l'anonimat pel fet de garantir que hi ha un nombre mínim de 10 Entities desplegades, i què no s'acabaran els identificadors degut a fallades durant el procés de publicació dels documents, ja que com a mínim hi ha 2 Entities més per cada usuari.

6.6.2 Selecció dels nodes del túnel

El protocol d'encaminament anònim crea una cadena de M Entities amb les N Entities que ha creat el Directory.

$$M \geq (N + 2)$$

El protocol crea la cadena de M Entities ordenades aleatòriament de manera que la Entity destí estigui enmig d'aquesta cadena, i assegurant, que l'Entity origen, l'Entity situada a la primera posició, es torna a repetir al menys un cop més.

La Entity destí precedeix a la segona aparició de la Entity origen, per tal de ser que sigui la Entity origen qui respongui a l'usuari sobre la resolució de la transmissió. La posició de la segona Entity origen és aleatòria, només compleix que aparegui entre la Entity destí i l'ultima Entity.

Per tal d'evitar que la Entity origen sigui l'ultima Entity, creem una repetició d'una Entity quan l'Entity origen estigui a l'ultima posició.

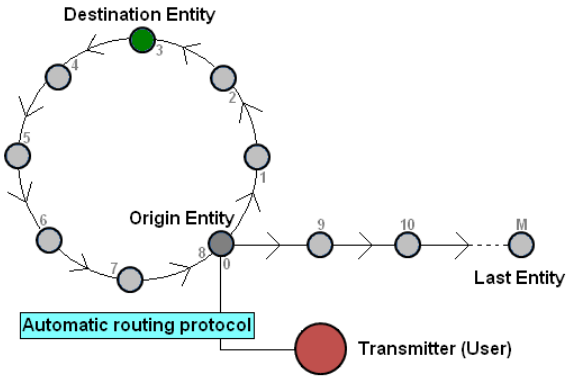

Figura 2. Exemple d'una cadena

Com veiem a la figura 2, l'Entity origen es troba a la posició 0 el primer cop, i a la posició 8 el segon cop, on envia la resolució de la operació a l'usuari.

1 A: Nombre d'usuaris al sistema

6.6.3 Creació paquet a enviar

Una vegada l'usuari ja té autorització per part del Directory, aquest construeix el paquet per enviar les dades a l'Entity destí.

1. L'User Application genera M claus simètriques associades a cadascuna de les Entities de la cadena (figura 2).

2. L'User Application construeix el paquet a enviar començant per el missatge que s'enviarà a la última Entity de la cadena:

$$Paquet_M = Id\_Entity_M,\ E_{pbk\_M}(K_M,0),\ padding$$

Com es pot observar, el primer paquet està composat pel ID de l'última Entity de la cadena (Id_Entity_M), i la clau simètrica (K_M) generada per l'usuari xifrada amb la clau pública de l'Entity, i finalment el padding opcional.

3. L'User Application comença a niar els paquets successivament depenent de l'Entity a la que correspongui:

a) Si l'Entity següent és una Entity retransmissora, l'algorisme crearà el següent paquet:

$$Paquet_X = Id\_Entity_X,\ E_{pbk\_X}(K_X,len_1),\ E_X(role,Id\_Entity_{X+1},\ Paquet_{X+1}),\ padding$$

Com observem, el paquet que crea es crea per aquesta Entity, té el paquet ($Paquet_{X+1}$) i el ID (Id_Entity_{X+1}) de la Entity que anirà després. A més en aquest cas tenim el paràmetre *role* que ens indicarà l'acció d'aquesta Entity, en aquest cas “router”, ja que només és un retransmissor.

També es pot veure com ara dins del xifrat amb la clau pública de la clau simètrica, també es xifra amb len_1 que serveix per determinar on acaba exactament el xifrat, ja que durant la retransmissió del missatge, les Entities entremig del camí poden afegir dades de farciment (padding) als paquets per tal d'alterar la seva aparença i augmentar l'anonimat.

b) Si l'Entity següent és una Entity origen, l'algorisme crearà el següent paquet:

$$Paquet_X = Id\_Entity_X,\ E_{pbk\_X}(K_X,len_1),\ E_X(role,messageId,\ Id\_Entity_{X+1},\ Paquet_{X+1}),\ padding$$

Aquest paquet per la Entity origen porta el *role* com a “origen”, paràmetre que indica a l'Entity que ho rep que aquest missatge l'ha originat ell mateix, i que té l'identificador intern *messageId*.

c) Si l'Entity següent és la Entity destí, l'algorisme crearà el següent paquet:

$$Paquet_X = Id\_Entity_X,\ E_{pbk\_X}(K_X,len_1),\ E_X(role,authz,\ op,\ len_2,\ <data>,\ Id\_Entity_{X+1},\ Paquet_{X+1})$$

Quan es crea el paquet de l'Entity destí posem el *role* com a “destiny”, i a més porta un codi d'operació *op* per determinar la operació a realitzar com enviar o rebre un document. També es composa del paràmetre len_2 que és la mida de les dades, i les dades. A més també s'ha incorporat el terme *authz,* que és el codi d'autorització de l'operació.

4. Finalment, enviem el missatge:

$$Paquet = Paquet_1,\ EC,\ padding$$

On veiem que el paquet final a enviar incorpora el paquet de la Entity origen, EC, que és un codi d'error d'1 byte, i el padding.

6.6.4 Estratègia de retransmissió

Una vegada l'Entity origen rep el paquet enviat per l'User Application, aquest torna a enviar el paquet a la següent Entity i estableix un timeout per si succeeix algun error.

En el cas de que algun error aparegui durant la retransmissió i l'Entity origen no rebés el paquet a retransmetre altre cop, o el *messageId* d'aquest sigui desconegut, l'Entity origen tancaria la connexió amb l'usuari comunicant-li el codi d'error corresponent.

Quan una Entity rep un nou paquet, desxifra el text xifrat del missatge utilitzant la clau privada i comprova:

a) Si $len_1 = 0$, és l'Entity del node final de la transmissió i no ha de retransmetre cap paquet.

b) Si *role*=”origen”, és l'Entity origen, per tant retransmet el paquet i engega el timeout. Si el timeout es compleix, s'enviarà a l'usuari un paquet amb *EC*=2. Si en canvi, és el segon cop que apareix, s'enviarà la confirmació a l'usuari amb l'*EC* posat per l'Entity destí.

c) Si *role*=”router”, és una Entity retransmissora i per tant retransmetrà el següent paquet.

d) Si *role*=”destiny”, és l'Entity destí, per tant aquest comprova que l'usuari tingui autorització per fer l'operació, i en cas de que no estigui autoritzat, s'enviarà el codi d'error *EC*=7 a l'usuari. A continuació, l'Entity emmagatzemarà el camp <data> si $len_2 \neq 0$. Si l'operació ha anat bé, es posa *EC*=1.

Quan l'User Application rep el missatge de confirmació de transmissió, es tanca la connexió i es mostra a l'usuari el missatge.

7 ANÀLISI DEL PROTOCOL D'ENCAMINAMENT ANÒNIM

Per fer l'anàlisi anem variant el padding dels paquets de les Entities. Una vegada definit el padding variarem el timeout de la connexió entre l'Entity origen i l'usuari.

Totes les proves que es mostraran a continuació s'han fet mitjançant l'escriptura de logs dins del protocol, per tal de poder analitzar-lo. I també totes les proves estan realitzades en mode *localhost*.

7.1 Test del padding

Per fer una estimació del padding més eficient per el nostre protocol vam fer diversos test variant la mida del padding. Es fixa un timeout a la connexió entre l'Entity origen del segon cop i l'usuari.

Per començar fixarem un padding a totes les Entities de 5KB per tal de veure els recursos que consumeix cada Entity i comprovem la mida dels paquets que arriba a cadascuna de elles.

7.1.1 Padding fix a 5KB

Fem les proves del padding amb un document de 1MB i comprovem el temps de processament total de la transferència, i el temps de processament per cada Entity segons la variació de usuaris, i per tant la variació del numero de Entities al túnel.

La primera prova es fa simulant que només hi ha un usuari al sistema, per tant hi ha un numero mínim de 12 Entities.

Amb aquest sistema que utilitza un túnel de 12 nodes com a mínim, un atacant que trenqui un node, si no és el primer, no podrà conèixer qui és l'origen del paquet, i tampoc podrà saber si l'emissor ha estat el node anterior, ja que el paquet està encriptat amb la clau simètrica, i aquesta està encriptada amb una clau pública RSA.

Si el node maliciós és el primer, podria arribar a descobrir el node emissor encara que si el primer node és honest, cap altre node del túnel podrà identificar el node emissor directament.

El mateix passa amb el role de les Entities, un atacant que trenqui un node no podrà saber qui és el node font, origen, destí o retransmissor.

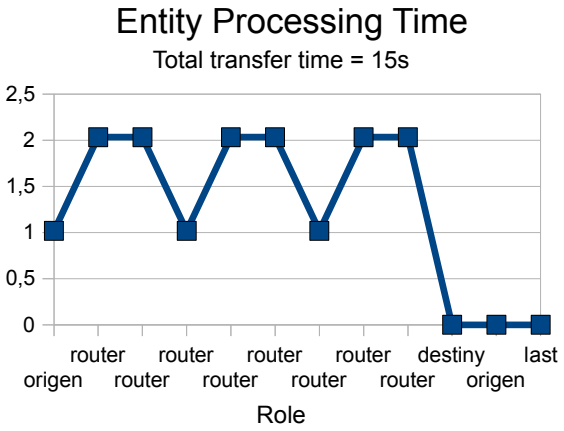

Figura 3. Temps de processament per Entity

Després de llançar la prova, veiem que n'han sortit 12 Entities al sistema, i que el temps de transferència total del document amb aquestes ha estat de 15 segons, un temps acceptable per una transmissió d'un arxiu d'1MB.

A la figura 4 es mostren diverses proves analitzant el temps de transmissió d'1MB variant el nombre d'usuaris al sistema, de 0 a 5, i per tant el nombre d'Entities.

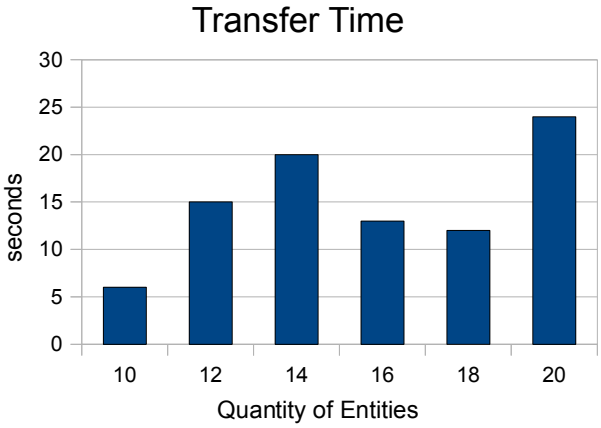

Figura 4. Temps de transferència

Com podem veure, contra més Entities al sistema, més triga la transferència del document, encara que depèn de la posició en la que aparegui l'Entity destí, ja que si apareix al principi, la transmissió serà més ràpida, al no totes les Entities tenir que transmetre l'arxiu, només aquelles que estan davant de l'Entity destí. I com observem a la figura 4, en els tests on la quantitat d'Entities al sistema eren 16 i 18, la transmissió ha estat mes ràpida, perquè l'Entity destí ha aparegut abans i menys nodes han hagut de transmetre l'arxiu.

En la nostre prova l'Entity destí apareix en l'antepenúltima posició, el que vol dir que les 9 Entities anteriors han hagut de transmetre un paquet més gran de 1MB, i per tant han necessitat més temps de transmissió.

Si observem la figura 5, veurem la mida dels paquets en cadascuna de les Entities.

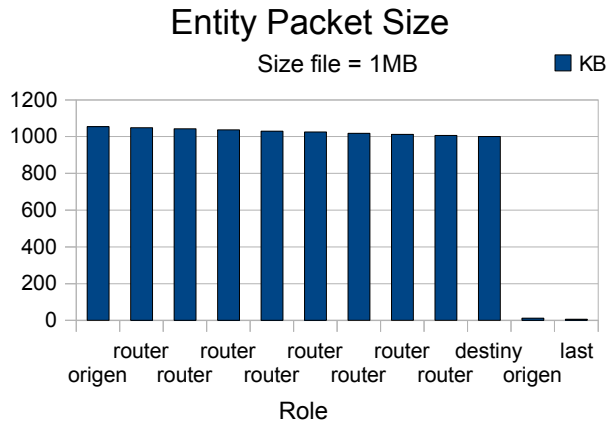

Figura 5. Mida del Paquet per Entity

Com s'observa a la gràfica de la mida dels paquets que arriben a cada Entity, veiem un clar descens de la mida d'aquests després de passar per l'Entity destí i aquest descarregui el document.

Per tant, encara que un atacant que trenqui un node no pugui saber qui és l'origen del paquet, un atacant que aconsegueixi controlar tot el túnel podrà esbrinar la traça que ha seguit el paquet, ja que podem veure que la mida dels paquets cada vegada que passa una Entity va disminuint, i podrà suposar que l'Entity destí és la que agafa més dades del paquet, observant el descens de la mida dels paquets una vegada passada aquesta.

Si observem la diferencia de la mida dels paquets entre una Entity i la següent, com es mostra a la figura 6, podem observar clarament quina Entity està agafant més dades.

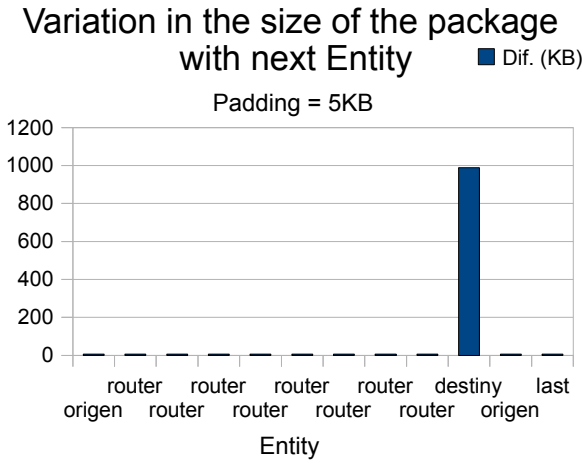

Figura 6. Variació de la mida dels paquets amb l'Entity següent

Per tal de resoldre aquestes deficiències, es proposa afegir un padding a totes les Entities de la mateixa mida que l'arxiu, d'aquesta forma un adversari no podrà esbrinar l'Entity destí.

7.1.2 Padding igual a la mida de l'arxiu a totes les Entities

En aquest cas tornarem a fer la prova amb un arxiu d'1MB i un usuari, és a dir, com a mínim 12 Entities, però el padding que utilitzarem serà igual a la mida de l'arxiu a enviar al destinatari.

D'aquesta manera volem evitar que un adversari que pugui controlar tot el túnel, pugui esbrinar qui és l'Entity destí.

Si observem la figura 7, podem veure el temps de processament que ha necessitat cada Entity del sistema.

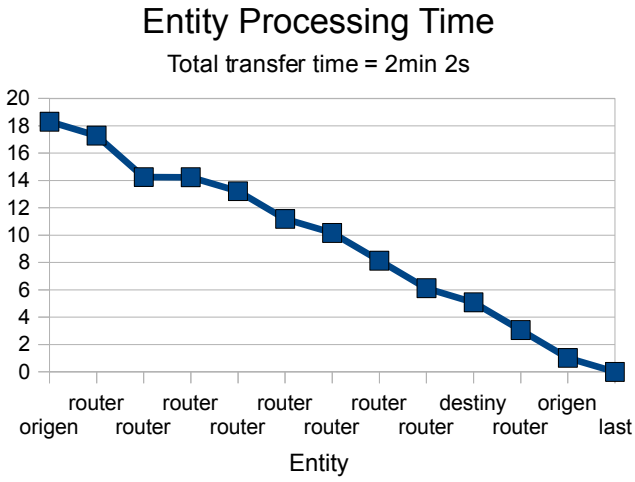

Figura 7. Temps de processament per Entity

Com podem veure, aquesta vegada han aparegut 13 Entities, i podem observar com el temps de processament de les Entities va disminuint cada cop que passa una d'elles, ja que el paquet que ha de processar cadascuna és cada vegada menor.

A causa d'això, el temps que ha necessitat el sistema per fer la transferència ha estat de 2 minuts i 2 segons, una latència més alta que en el cas anterior, encara que aquest sistema ens aporta més anonimat al sistema.

A la figura 8, podem observar els diferencies entre els paquets que arriben a les Entities en aquesta prova, amb el padding igual a la mida de l'arxiu.

Com veiem, aquesta vegada un adversari no podrà intuir qui és l'Entity destí, ja que totes les Entities agafen un paquet pràcticament igual.

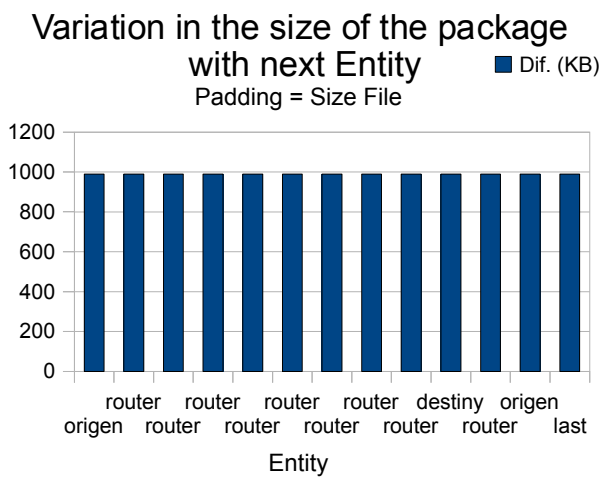

Figura 8. Variació de la mida dels paquets amb l'Entity següent

En canvi, si observem la figura 9, on es mostra la mida del paquet que arriba a cadascuna de les Entities, podem veure com la mida del paquet va disminuint.

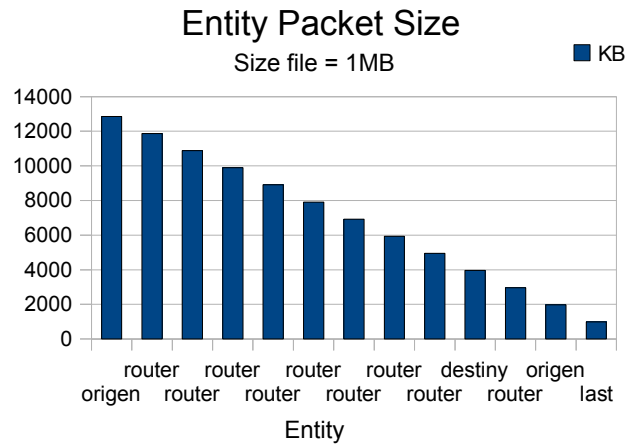

Figura 9. Mida del Paquet per Entity

Al observar això, un atacant podria esbrinar la traçada del paquet i a més podria saber qui és l'Entity origen, si aquest controla tot els nodes del túnel.

Veient que hem aconseguit reduir les probabilitats de que un adversari pugui arribar a conèixer l'Entity destí, encara que continuem tenim la fuga d'informació de la possibilitat de que un adversari aconsegueixi esbrinar la traçada dels paquets i per tant conèixer l'Entity origen.

Proposem un mètode per tal de resoldre aquesta fuga d'informació. El mètode es tracta de posar un padding petit a totes les Entities, i enviar la mida de l'arxiu, de forma que quan una Entity desxifri el paquet pugui obtenir la mida de l'arxiu i pugui afegir un padding igual a la mida de l'arxiu a partir de que l'Entity destí aparegui.

7.1.3 Padding petit i augmentat en el protocol de recepció a una mida igual a la mida de l'arxiu a les Entities posteriors a l'Entity destí

Per aquesta prova posem un padding igual a 1KB, que és relativament petit, i afegim la mida de l'arxiu dins del paquet encriptat.

A l'hora de rebre el paquet, l'Entity desxifra el paquet i si l'Entity destí ja ha aparegut, afegirà un padding per la següent Entity de una mida igual a la mida de l'arxiu.

Si observem la figura 10, comprovem com el temps de processament de cadascuna de les Entities, en aquesta prova amb un fitxer d'1MB, i un usuari al sistema, està ara molt més repartit, per tant el rendiment és millor.

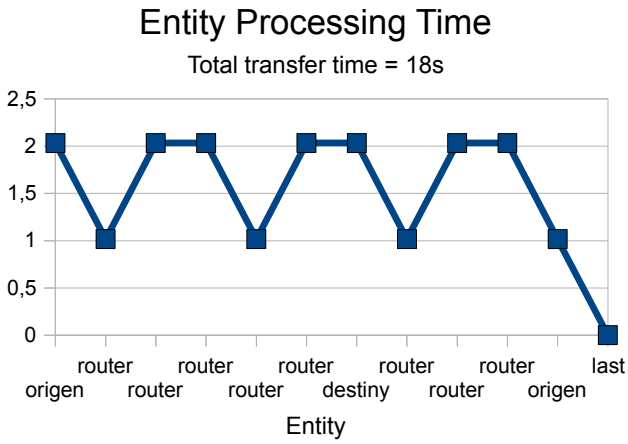

Figura 10. Temps de processament per Entity

A més, el temps total de la transmissió ha estat de 18 segons, per tant, també hem millorat la latència del sistema.

Si ara analitzem la mida dels paquets que arriben a cadascuna de les Entities, donaria un resultat com el que es mostra a la figura 11.

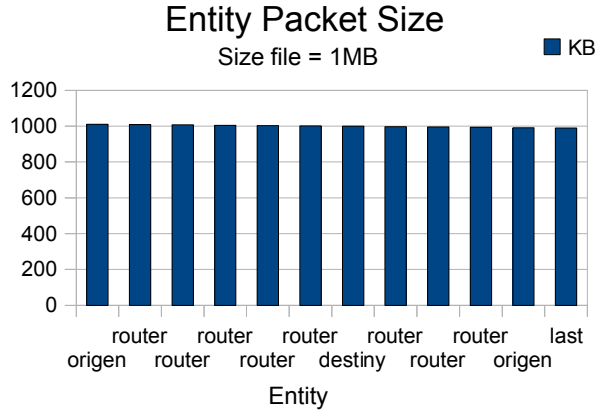

Figura 11. Mida del Paquet per Entity

Com podem observar, en aquesta prova un adversari que controlés tot el túnel del sistema, no podria esbrinar res per la mida dels paquets que arriben a les Entities perquè aquestes tenen una mida pràcticament idèntica.

Per tant, aquesta prova no presenta la fuga d'informació respecte a la mida dels paquets que arriben a les Entities, i per tant, incrementa el grau d'anonimat al sistema.

La figura 12 mostra el resultat de la variació de la mida dels paquets amb l'Entity següent, és a dir, observant la mida dels paquets d'entrada i sortida, un adversari pot suposar que els bytes que falten són els que l'Entity ha utilitzat.

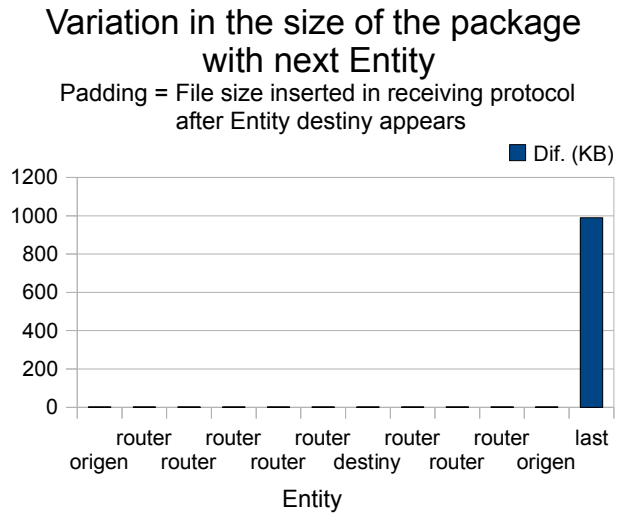

Figura 12. Variació de la mida dels paquets amb l'Entity següent

Com podem comprovar, la variació a totes les Entities, menys a l'última, és molt petita, és a dir, un adversari no podria esbrinar amb la variació de la mida dels paquets quina és l'Entity destí ni l'Entity origen, nomes pot veure quina és l'Entity que utilitza més bytes, que serà l'última Entity de la cadena i aquesta mai serà l'Entity destí, ni l'Entity origen.

Per tant, un adversari pot pensar que l'Entity destí és l'última, ja que les demés sembla que no agafin suficient informació com per ser aquesta.

Amb aquesta configuració del padding del protocol aportem un millor balanç de carrega, amb un temps de processament per Entity més repartit entre tota la cadena, un millor rendiment del sistema, aconseguint retallar el temps total de la transferència i a més aportant un alt grau d'anonimat, evitant que un adversari que controli tot el túnel, pugui intuir qui és l'Entity destí, origen o retransmissor per mitjà de la mida dels paquets o per la diferencia de mida de paquets que arriben a cadascuna de les Entities.

7.2 Test del timeout

Per fer el càlcul més òptim del timeout, fem diverses proves amb un Usuari al sistema, és a dir, 12 Entities com a mínim, i calculem el temps de transferència total i el temps de transferència per cada KB que enviem.

A la següent taula 1, podem observar el comportament del protocol per a una mida d'arxiu variant entre arxius molt petits, de 258 KB, fins a un arxiu de 8 MB.

Com podem observar a l'hora d'extraure els segons que utilitza el sistema per cada KB, la velocitat en tots els casos és exactament igual, per tant utilitzarem aquesta dada per tal de calcular el timeout.

Size File (KB)	Users	Tunnel Lenght	Transfer Time (s)	sec/Entity	Timeout (s)
258	1	12	**5**	0,02	**17,16**
516	1	12	**9**	0,02	**22,32**
778	1	12	**14**	0,02	**27,56**
1024	1	12	**18**	0,02	**32,48**
2048	1	12	**33**	0,02	**52,96**
3072	1	12	**51**	0,02	**73,44**
4096	1	12	**72**	0,02	**93,92**
5120	1	12	**84**	0,02	**114,4**
6144	1	12	**103**	0,02	**134,88**
7168	1	12	**122**	0,02	**155,36**
8192	1	12	**137**	0,02	**175,84**

Taula 1. Càlcul Timeout

Els segons per cada KB en tots els casos és de 0,02 seg/KB, per tant hem decidit afegir un timeout superior a:

$$Timeout > Size\ File \cdot 0{,}02\ seg/KB$$

Com el timeout ha de ser superior a aquesta multiplicació, es proposa el següent timeout:

$$Timeout = (Size\ File \cdot 0{,}02\ seg/KB) + Tunnel\ lenght$$

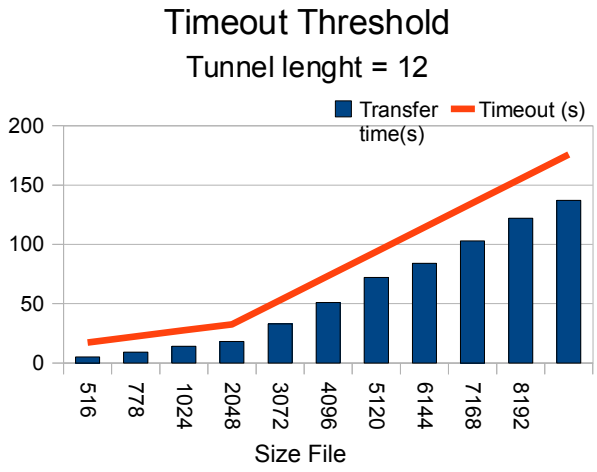

Figura 13. Llindar del Timeout

On afegim un segon més al timeout per cadascuna de les Entities que estan al sistema, i com podem observar a la figura 13, en tots els cassos el temps de transferència queda dins de la llindar del timeout proposat.

Ara fem les proves del timeout però aquest cop variant els usuaris del sistema i deixant fixe una mida d'arxiu a 1MB. A la taula 2, podem veure les proves variant entre 1 usuari i 10. Aquest cop hem afegit els segons que necessita cada Entity segons el temps de transferència total.

Size File (KB)	Users	Tunnel lenght	Transfer Time (s)	sec/Entity	Timeout (s)
1024	1	12	**18**	1,5	**32,48**
1024	2	15	**24**	1,6	**35,48**
1024	3	16	**25**	1,56	**36,48**
1024	4	18	**29**	1,61	**38,48**
1024	5	20	**32**	1,6	**40,48**
1024	10	30	**47**	1,57	**50,48**

Taula 2. Càlcul Timeout

Com veiem a la figura 14, el temps de transferència cada vegada s'aproxima més al llindar del Timeout, per tant si augmentem els usuaris pot ser que el temps que es necessita per la transferència sigui major que el timeout.

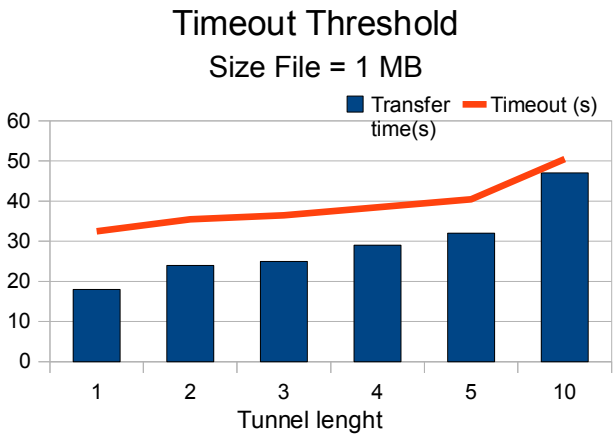

Figura 14. Llindar del Timeout

Ho comprovem fent proves variant l'usuari entre 1 i 50, i comparant el temps de transferència amb el timeout.

La taula 3, mostra els resultat de les proves, on efectivament quan els usuaris són més de 10, el timeout calculat es queda petit.

Veient la figura 15, podem observar la evolució del timeout respecte al creixement de les Entities al sistema. Per tant, el càlcul del timeout, no nomes depèn de la mida de l'arxiu, sinó també de les Entities en el sistema.

Size File (KB)	Users	Tunnel Lenght	Transfer Time (s)	sec/Entity	Timeout (s)
1024	1	12	**18**	1,5	**32,48**
1024	2	15	**24**	1,6	**35,48**
1024	3	16	**25**	1,56	**36,48**
1024	4	18	**29**	1,61	**38,48**
1024	5	20	**32**	1,6	**40,48**
1024	10	30	**47**	1,57	**50,48**
1024	15	40	**63**	1,58	**60,48**
1024	20	50	**88**	1,76	**70,48**
1024	25	60	**110**	1,83	**80,48**
1024	30	70	**130**	1,86	**90,48**
1024	35	81	**151**	1,86	**101,48**
1024	40	90	**172**	1,91	**110,48**
1024	45	100	**179**	1,79	**120,48**

Taula 3. Càlcul Timeout

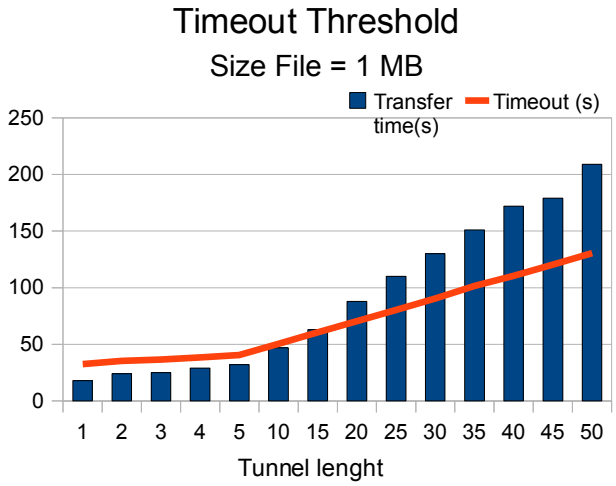

Figura 15. Llindar del Timeout

Per tant, modificarem la manera de calcular el timeout, per tal de que també depengui de les Entities al sistema. Si ens fixem a la taula 3, podem veure que cadascuna de les Entities necessiten entre 1,5 segons i 1,9 segons, per tant assegurarem que mai el timeout sigui menor que 2 segons per Entity al sistema.

*Timeout = (Size File · 0,02 seg/KB) + (2 * Tunnel lenght)*

Si ara fem el càlcul del timeout per les mateixes que a l'apartat anterior, comprovarem si aquest càlcul serà vàlid.

La taula 4, mostra els resultats de la prova, i com es pot observar cap transferència supera en temps al timeout calculat.

A més si ho observem a la figura 16, podem veure com no nomes el temps de transferència no sobrepassa la llindar del timeout, si no que a més és pot veure amb claredat que el timeout va creixent en proporció a les Entities i a més també creix en proporció a la mida de l'arxiu.

Size File (KB)	Users	Tunnel Lenght	Transfer time(s)	sec/Entity	Timeout (s)
1024	1	12	**18**	1,5	**44,48**
1024	2	15	**24**	1,6	**50,48**
1024	3	16	**25**	1,56	**52,48**
1024	4	18	**29**	1,61	**56,48**
1024	5	20	**32**	1,6	**60,48**
1024	10	30	**47**	1,57	**80,48**
1024	15	40	**63**	1,58	**100,48**
1024	20	50	**88**	1,76	**120,48**
1024	25	60	**110**	1,83	**140,48**
1024	30	70	**130**	1,86	**160,48**
1024	35	81	**151**	1,86	**182,48**
1024	40	90	**172**	1,91	**200,48**
1024	45	100	**179**	1,79	**220,48**
1024	50	110	**209**	1,9	**240,48**

Taula 4. Càlcul Timeout

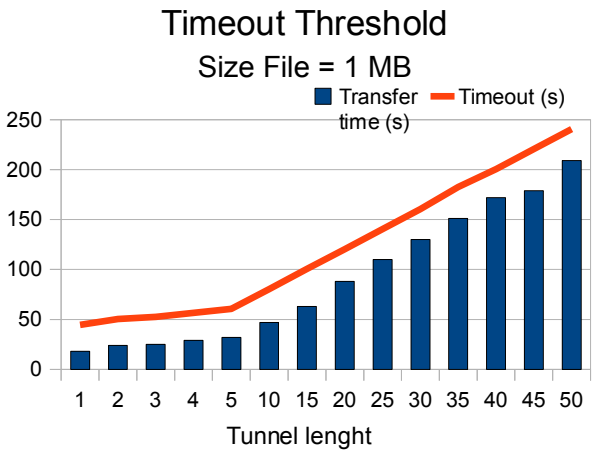

Figura 16. Llindar del Timeout

Amb això aconseguim obtenir el timeout més òptim per el nostre protocol.

7.3 Anàlisi de l'anonimat

Els certificats de Directory, Entities i usuaris signats per una autoritat certificadora, ens proporcionen autenticitat als participants del sistema, fet que farà molt difícil a un adversari, per tal de ficar-se dins del circuit suplantant la identitat d'alguns dels nodes del circuit.

A més també aportem anonimat amb l'estructura, la mida del túnel i l'estratègia de retransmissió del nostre sistema.

7.3.1 Estructura del sistema

Si suposem que el nostre sistema té 5 usuaris i per tant, una mida del túnel (N) de 20 salts al sistema i tants nodes maliciosos com M, i no considerem altres estratègies d'atac, l'entropia pot ser calculada com:

$$H^2(X^3) = \log_2 N$$

2 H: Entropia
3 X: Sistema de comunicacions anònima

I el grau d'anonimat del sistema amb M nodes maliciosos és:

$$D^4(X) = \log_2(N-M) / \log_2 N$$

A la figura 17, podem veure com el grau d'anonimat va disminuint a mesura que anem augmentant els nodes maliciosos, com era d'esperar, encara que per altre banda es pot observar com el grau d'anonimat va baixant molt lentament, i inclús, quan el 75% dels nodes del circuit són maliciosos, el nivell d'anonimat continua sent relativament alt. Això és perquè ho estem realitzant dins d'un sistema P2P.

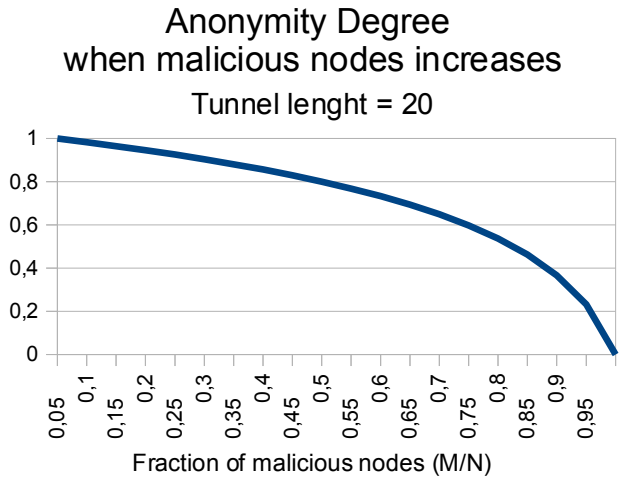

Figura 17. Grau d'anonimat amb l'increment dels nodes maliciosos

7.3.2 Estratègia d'enrutament

La figura 18 ens mostra la mida del túnel variable amb els usuaris d'una xarxa.

Com podem observar la mida mínima del nostre circuit és de deu, que és la mida quan no hi ha cap usuari i on el grau d'anonimat per mida del túnel comença a augmentar més lentament, ja que contra més gran és la mida del túnel més possibilitats té un adversari de formar part d'aquest.

Com hem vist anteriorment, a la figura 16, contra més gran és la mida del túnel, el temps de transferència del document serà major. El que vol dir, que contra més anonimat aportem a través de la mida del túnel, la latència del sistema augmentarà.

Un atacant necessita conèixer la direcció IP del node per tal de controlar a aquest [3]. En el protocol implementat, un atacant no pot conèixer la direcció IP dels nodes participants en el circuit, perquè utilitzem el protocol SSL, en canvi, si el node maliciós és el primer, aquest podria veure la IP del node transmissor.

4 D: Grau d'anonimat

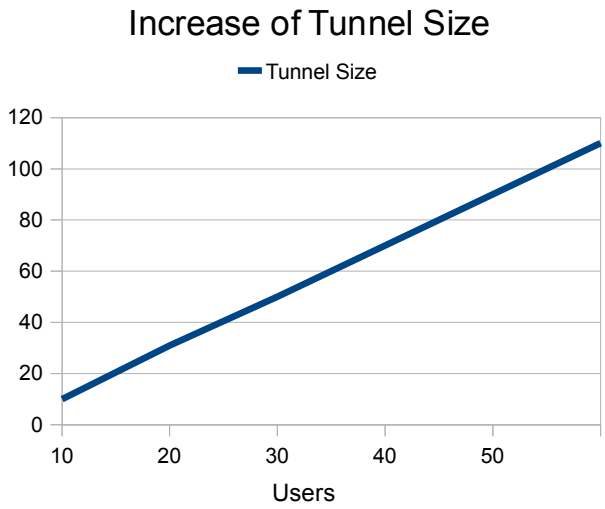

Figura 18. Increment de la mida del túnel

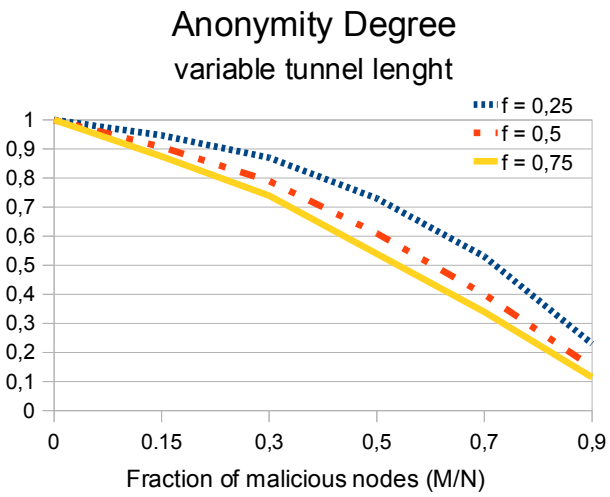

Figura 19. Grau d'anonimat amb un túnel variable

L'anàlisi de la mida del túnel de Andrew Tran, Nicholas Hopper i Yongdae Kim ens proposava una mida de túnel variable per augmentar el grau d'anonimat al sistema [3]. La figura 19, mostra el grau d'anonimat el nostre sistema, amb un túnel variable.

Com l'estratègia de retransmissió és basada en onion routing, un adversari ha de controlar el primer node i alguns altres nodes en el túnel, per tal d'identificar el iniciador, i per tant necessita al menys controlar més de la meitat dels nodes en un túnel per la seva reconstrucció.

La probabilitat de que un adversari identifiqui el node transmissor, dins del sistema amb el túnel variable és[2]:

$$P(S)=\sum((1-f)^{(l-1)})f(M/N)^{((l+1)/2)}$$

La variable *f*, és establerta per la probabilitat de que un node retransmissor, retransmeti el missatge de l'iniciador, que són tots nodes abans de l'Entity destí, ja que si un adversari controla els nodes que estan després de l'Entity destí, aquest no podrà obtenir rés. Per tant el grau d'anonimat al túnel variable és:

$$D(X) = (1\text{-}P(S)) \log_2(N\text{-}M) / \log_2 N$$

Com podem veure, el túnel de mida variable ens aporta al sistema un alt grau d'anonimat, ja que inclús quan la fracció de nodes compromesos al sistema és alt, com per exemple del 70%, aquest continua aportant un grau d'anonimat al sistema.

El valor de f serà 0.25, 0.5 o 0.75 quan l'Entity destí es troba en el primer quart, al mig, o a l'últim quart del túnel, respectivament.

Els autors Jia Zhang, Haixin Duan, Wu Liu i Jianping Wu ens mostra un anàlisi entre el túnel variable i el túnel fixe, on demostren que els túnels amb una mida variable aporten un grau d'anonimat més alt que no els túnels d'una mida fixa.

8 COMPARACIÓ AMB ALTRES SISTEMES ACTUALS

El nostre protocol realitza recerques de mode iterativa, tal i com es fa en altres sistemes com SALSA, i a més en aquest sistema també nomes pot ser trencat en cas de que un adversari sigui el primer node.

A SALSA, també són els clients el que escullen la construcció del circuit, encara que cada node nomes coneixerà el predecessor i el successor, exactament igual que el nostre protocol.

En canvi, aquest sistema nomes selecciona 3 nodes aleatoris per construir un circuit, i en el nostre cas el mínim de nodes en un circuit és de 12, i és variable, ja que augmenta amb el nombre d'usuaris al sistema.

El nostre algorisme encripta la clau simètrica dins d'un xifrat amb la clau pública del node, en canvi, a altres sistemes, com MorphMix, un node estableix una clau simètrica mitjançant l'enviament a un node *b* de la meitat de l'algorisme d'encriptació Diffie-Hellman amb la clau pública del testimoni *w.* El node *b* transmet el xifrat cap a *w* que elimina la capa d'encriptació i envia el resultat a *a.* Aquest genera la meitat de l'algorisme de Diffie-Hellman i la envia a través de *b*. D'aquesta manera, les dues parts poden generar la seva clau simètrica compartida.

El sistema Cashmere, encara que també utilitza retransmissors virtuals composats per un conjunts

de nodes per a la resistència, disposa de grups de transmissors on cadascun té una clau pública i privada per membre i altres claus publiques i privades per el grup.

De tal forma que el sistema utilitzarà l'ID del grup com a clau per enviar els missatges. El root del grup és el responsable de processar els missatges en nombre del grup retransmissora.

Com veiem hi ha altres sistemes que utilitzen característiques similars a la nostra, però amb el nostre algorisme hem aconseguit pujar el grau d'anonimat, ja que no nomes utilitzem un circuit amb un nombre de nodes variable i a partir de 10, que és la franja on el grau d'anonimat obtingut per un circuit de nodes comença a estancar-se, si no que a més hem aconseguit obtindre més anonimat a la possibilitat de que un adversari mesuri els paquets que s'envien entre els nodes, ja que li serà molt complicat esbrinar qui és el node destí o origen, observant la mida dels paquets, al tenir tots una mateixa mida i a més tots sembla que pràcticament no agafin informació al ficar padding després de llegir el paquet.

Si observem la taula podem veure altres sistemes que utilitzen les mateixes estratègies, tant per la estructura, la mida del túnel o l'estratègia de retransmissió.

Strategy	Anonymity	Performance	Other systems
Unstructured	High	-	MorphMix
Variable length	High	Low	AP3, Crowds
Onion routing	High	Mid	Tarzan, MorphMix

Taula 5. Estratègies implementades al protocol

Segons l'article [3], podem veure com es fa una comparació dels diferents sistemes, encara que veiem que cap dels sistemes implementats aporta un anonimat alt com el nostre protocol tant en l'estructura, com en la mida del túnel, com en l'estratègia de retransmissió.

I finalment, hem aconseguit obtindre un repartiment òptim de la carrega entre totes les Entities, el que dona un millor rendiment al protocol.

9 CONCLUSIONS

Hem aconseguit un protocol fiable, ja que l'User Application sap els errors que es produeixen durant el circuit i podria tractar de crear un de nou.

S'ha millorat la eficiència i l'anonimat al multiplexar els fluxos TCP al llarg de cada circuit.

L'User Application pot enviar tràfic a qualsevol node del circuit, és aleatori, i el destí mai pot ser l'últim node, d'aquesta manera disminuïm les possibilitats de que un adversari pugui realitzar els atacs basats en l'observació final del circuit.

El nostre protocol proporciona un servidor Directory signat per una CA i proporcionarà els certificats de cadascuna de les Entities i els usuaris.

Mitjançant l'encriptació de les dades amb una clau simètrica, i encriptant aquesta amb la clau pública del node, i enviant-ho al llarg d'un circuit de nodes encadenats i xifrats consecutius, aconseguim tenir privacitat al sistema.

Hem aconseguit pujar el grau d'anonimat amb el mètode en el proces de recepció de les Entities de fixar un padding de la mateixa mida que l'arxiu que estem enviant a totes les Entities que segueixen a la de destí.

Quan augmentem el grau d'anonimat al sistema per la mida dels circuits, la latència augmenta.

Per tant, el protocol ens aporta un alt grau d'anonimat, ja que a un adversari li serà molt complicat esbrinar qui és el transmissor o qui és el destí, gràcies al mètode d'utilització del padding i a la mida variable del túnel.

S'ha incorporat un timeout òptim, és a dir, un timeout proporcional a la mida del fitxer a enviar i al nombre d'Entities que disposa el sistema.

Repartim la carrega entre totes les Entities, ja que és millor tenir més serveis que necessitin menys recursos que tenir menys serveis però que aquests necessitin més recursos. D'aquesta manera s'ha millorat el rendiment del protocol.

Si el primer node del túnel és maliciós, aquest podria identificar a l'usuari real, encara que no sabrà si realment és l'emissor o només un retransmissor, però no podrà conèixer el servidor real. Si en canvi, el primer node és honest, altre node del túnel no pot identificar l'emissor real directament i només existeix una possibilitat 1/N (N nombre total de nodes al túnel) per esbrinar la identitat correcta de l'emissor.

El iniciador envia el tràfic al node destí del circuit, que mai serà l'últim, evitant d'aquesta manera els atacs basats en l'observació del final del circuit.

És molt difícil de determinar si el resultat de la petició és correcte [3]. En el protocol s'ha implementat un algorisme on l'usuari obtindrà una resposta del resultat de la transferència, tant si ha anat correcte com si aquest ha fallat. A l'apartat 7.2, hem vist el càlcul del timeout per donar una resposta tant si ha aparegut algun error en la transferència com si el paquet s'ha perdut durant el camí.

10 TREBALLS FUTURS

Algorisme crear en *localhost*, per tant es podrien fer proves implementant aquest protocol dins d'una xarxa descentralitzada com CoDeS, sistema del que disposa la UOC, on es permeten desplegar serveis en ordinadors contribuïts voluntàriament a una comunitat.

Es podria optimitzar una mica més les operacions del protocol, si quan creem el circuit posem un paràmetre aleatori entre 1 i 0, a totes les Entities que precedeixen a l'Entity destí, de manera que només algunes de elles apareguin en el circuit.

Podem donar-li sentit a l'algorisme si l'implementem en un context docent on hi ha un rol de professor i un d'alumne, on cadascun tinguin diferents permisos. I a més, podríem implementar el mètode d'obtenció o modificació d'un document del servidor. De manera que podríem aconseguir una aplicació completa descentralitzada per fer revisions peer-review de manera anònima.

11 BIBLIOGRAFIA

[1] Edman, Matthew and Yener, Buelent: On Anonymity in an Electronic Society: A Survey of Anonymous Communication Systems. Review. NY-USA, 2009.

[2] Jia Zhang, Haixin Duan, Wu Liu and Jianping Wu: Anonymity analysis of P2P anonymous communication systems. Article. Tsinghua University, 2011.

[3] Andrew Tran, Nicholas Hopper and Yongdae Kim: Hashing it Out in Public. Article. Chicago-USA, 2009

[4] Roger Dingledine, Nick Mathewson and Paul Syverson: The Second-Generation Onion Router. Article. San Diego-CA, 2004.

[5] Qiyan Wang, Prateek Mittal and Nikita Borisov: In Search of an Anonymous and Secure Lookup. Article. Chicago-USA, 2010

[6] Michael J. Freedman, Emil Sit, Josh Cates and Robert Morris: Introducing Tarzan, a Peer-to-Peer Anonymizing Network Layer. Berlin-Alemanya, 2002.