Eina d'anàlisi de rendiment d'una xarxa

Alan Almagro Guerras
ETIS

Maria Isabel March Hermo

19/11/2008

RESUM

Les eines d'anàlisis de rendiments de xarxes consisteixen en un control dels protocols que fan que les xarxes funcionin correctament. Actualment el protocol més estès és el TCP/IP.

El projecte intentar detectar el tipus de protocols que tenim en la nostra xarxa per així poder fer un anàlisis i estudiar les possibles anomalies que pugui patir la xarxa.

El resultat és una sèrie de paquets que hem capturat amb la nostra targeta de xarxa amb mode promiscu, els quals mostrem per pantalla quins ens sigut, tot seguit de unes estadístiques i uns gràfics per ajudar-nos a analitzar els resultats. Tanmateix, tindrem una utilitat per detectar totes les IP's que han passat per la nostra interfície de xarxa, per tenir constància de que les màquines que intercanvien paquets són màquines conegudes per nosaltres i no hi ha cap altra equip desconegut per nosaltres que estigui intercanvien paquets per la xarxa o fent atacs per detectar vulnerabilitats a la nostra xarxa.

En definitiva, farem una aplicació que ens permet controla els protocols que estan treballant en la nostra xarxa i poder entendre i aprendre més sobre ella.

INDEX

Portada 1
Resum 2
Índex 3
1 Introducció 4
1.1 Justificació treball fi de carrera 6
1.2 Objectius 7
1.3 Enfocament i mètode seguit 8
1.4 Planificació projecte 9
1.5 Productes obtinguts 12
1.6 Breu descripció capítols de la memòria 13
2 Estudis previs 18
3 Implementació 29
3.1 Primer prototip 29
3.2 Definició del nostra IF 30
3.3 Integració del primer prototip amb el format IF 30
3.4 Classes creades 31
3.5 Creació interfícies de la aplicació 34
3.6 Programació de les interfícies 39
3.7 Finestra “Selecció d’interfícies” 40
3.8 Thread encarregat d’agafar paquets 40
3.9 Modificació de les taules 41
3.10 Començar a agafar paquets 42
3.11 Parar el Thread de agafar paquets 43
3.12 La instrucció swich del mètode receivePacket() ... 43
3.13 Mètode de detecció IP’s 44
3.14 Interfície d’equips detectats 44
3.15 Estadístiques 45
3.16 Problemes trobats en la implementació 50
4 Manual d’instal·lació 51
4.1 Com funciona 51
5 Conclusions 52
6 Glossari 53
7 Bibliografia 54

1 INTRODUCCIÓ

El món de les xarxes és un món bastant desconegut per la majoria de les persones que navegant per internet, de fet ni es poden imaginar tota la informació i tot els protocols que hi ha per a que ells puguin enviar un e-mail o llegir la pàgina Web del seu diari preferit. Per això hem decidit que aquest projecte vagi encaminat per saber què passa realment quan naveguem per internet, per la nostra xarxa particular o a la pròpia empresa. D'aquesta manera, poder extraure la informació que ens ajudi a esbrinar si hi ha alguna cosa que no està funcionant con hauría de ser i així comprovar que el rendiment és òptim.

Nosaltres com administradors de xarxa, quantes vegades ens hem preguntat ¿Aquesta xarxa va molt lenta? o ¿Li passa alguna cosa a la xarxa? O moltes qüestions per l'estil. Tot això ens ha dut a necessitar unes aplicacions que ens ajudin a esbrinar què passa per la nostra xarxa. Gràcies a elles podem saber quins protocols treballen, quants paquets passen per ella i de quin tipus són. Aquestes aplicacions ens ajuden a fer estadístiques per després poder analitzar si esta fallant alguna cosa, si tenim algun tipus d'analitzador a la nostra xarxa per desprès rebre atacs, etc. Aquest tipus de programari reben el nom de sniffer.

La nostra aplicació és una eina per monitoritzar que succeeix a la nostra xarxa, és a dir, anirà escoltant el que passa per la targeta de xarxa i mostrant-nos els resultats del tipus de paquets que ha anat capturant.
Es tracta d'anar agafant cada paquet que passi per la nostra interfície de xarxa i anar analitzant el que vulguem d'ell. En el nostra cas, anirem mirant quin tipus de protocols es tracta i quines són les IP que s'envien aquests paquets.

Aquest tipus d'aplicacions són molt útils tant en el món empresarial com en un àmbit més particular. Encara que segur que és més interessant per una empresa, ja que és més vulnerable a rebre atacs d'algun intrús que vulgui accedir a les nostres bases de dades o qualsevol part de la LAN.
Una part també important d'aquesta aplicació és que gràcies a ella podem aprendre molt sobre les xarxes, ja que podem veure que és el que passa quan nosaltres ens connectem a un ftp, fem un ping o visitem una pàgina web. Ja que penso que no som conscients de la gran quantitat d'informació i de paquets que s'envien per poder fer unes d'aquestes accions.

L'aplicació la farem amb el llenguatge de programació Java, ja que ens permetrà executar el nostra aplicatiu tant en sistemes Linux com en sistemes Windows, encara que nosaltres farem totes les proves sota el sistema operatiu Windows, ja que és el sistema operatiu més estès i més familiaritzat estic en la configuració del java. L'avantatge que té en Java és que si volem fer que l'aplicació funcioni sota Linux, no faran falta grans canvis a nivell de programació.

Les aplicacions que utilitzarem per programar seran l'eclipse i el Netbeans. El primer perquè és amb la eina de programació que més còmode estic i més he treballat. La segona perquè ens ajudarà molt alhora de fer les interfícies en Java, que és una feina bastant complicada i que no he fet mai.

La llibreria que farem servir per capturar la informació que passa per la nostra targeta de xarxa és la JPCAP, més endavant explicarem amb detall com funciona aquestes llibreries.

1.1 JUSTIFICACIÓ TREBALL FI DE CARRERA

He decidit fer el treball fi de carrera sobre la investigació del que passa en la nostres xarxes ja que penso que és un tema bastant important en aquests dies, els quals internet s'ha convertit en un fenomen indispensable, tant per un ús particular com per el món empresarial.
Penso que en el món actual aquests tipus d'aplicacions són necessàries perquè la seguretat a internet i en les xarxes locals és bastant vulnerable i sempre es descobreixen nous forats de seguretat. És indispensable estudiar el què passa per a les xarxes per poder crear connexions més segures.

Personalment vaig veure molt interessant aquesta àrea de la informàtica perquè és un dels seus punts més importants i d'on es pot aprendre més coses. Sobre tot, pots veure a la pràctica totes les teories del funcionaments del protocols estudiats a les assignatures de xarxes.
Vaig veure que aquesta àrea era bastant completa, ja que em permetria aprendre més sobre xarxes però també sobre programació, ja que gran part del projecte estàs programant, una programació bastant orientada a les xarxes.

Per altra banda, fer aquest tipus d'aplicació amb el llenguatge de programació Java em permet practica i conèixer amb més profunditat aquest llenguatge que és dels més utilitzats per fer aplicacions.

Per últim, aquest projecte em dona la oportunitat de realitzat tota la gestió que es fa en un projecte, es a dir, seguir tots els passos necessaris per a la realització d'un projecte, des de la planificació inicial, fins a les proves finals.

1.2 OBJECTIUS

El objectiu principal de aquest treball de fi de carrera és investigar què és el què passa sobre la nostra xarxa, en aquest cas ens centrarem en las xarxes Ethernet i sobre els protocols a nivell de aplicació, transport i xarxa.

Els protocols inicials i les dades que extraurem inicialment són:

Protocol IP: Captura les direccions IP tant del emissor com el del receptor i mostrar-les per pantalla, així com saber quin tipus de protocol es tracta i saber la versió IP.

Protocol ARP: Captura la direcció MAC tant del destí com la del origen. Així com la direcció IP del origen i del destí i per últim comprovar quin tipus d'operació és. Si es una petició o una resposta.

Protocol ICMP: Captura la IP del destí i del origen, així com saber el tipus de petició ICMP .

Protocol TCP: Captura la IP del destí i del origen, i esbrinar a quin tipus de protocol de aplicació pertany el paquet.

Protocol UDP: Captura la direcció IP tant del destí com del origen i busca el tipus de protocol de aplicació que per tant el paquet UDP.

Totes aquestes dades les anirem mostrant en una taula que tindrà la aplicació en la seva interfície principal.

També serà un dels objectius del projecte mostrar les estadístiques en un gràfic dels protocols capturats, així com una gràfica dintre de cada protocol. És a dir, dintre del protocol TCP mostrar si han sigut del tipus http, ftp, smtp, etc. I així successivament.

Detectar Equips: Es vol detectar els equips que hi ha al nostre voltant, per això anirem guardant les noves IP's detectades i les guardarem en un vector. Aquesta informació la mostrarem a la taula de Equips detectats.

1.3 ENFOCAMENT I MÈTODE SEGUIT

La metodologia a seguir per a realitzar el projecte serà cerca documentació tècnica sobre el funcionament de les xarxes i centrar-nos en el tipus de protocol que fa servir. També investigarem les diferents aplicacions que ja hi ha en el mercat per fer-nos una idea del que podem arribar a fer. Una vegada tenim tota informació necessària i tinguem les idees clares del que volem fer i com ho podem fer, ens œntrarem en l'anàlisi i disseny de la aplicació. Els últims passos per a fer la aplicació serà centrar-nos en la seva implementació, solucionar els problemes que no havíem comptat en el seu anàlisis i fer el jocs de proves.

1.4 PLANIFICACIÓ DEL PROJECTE

Eina d'anàlisi de rendiment d'una xarxa.

Objectiu del TFC

Volem crear una aplicació que ens indiqui si el rendiment de la xarxa es correcta o n'hi ha alguna anomalia que fa que el rendiment de la xarxa no sigui el que hauria de ser.

Recerca d'informació: 29 setembre – 12 Octubre.

Cerca informació sobre tipus de aplicacions que analitzen les xarxes i les seves diferents funcions, així podrem veure que volen que faci la nostra aplicació.
Cercarem també informació sobre el protocol necessari per poder fer la aplicació, en principi el TCP/IP.

Anàlisis i disseny de la aplicació 13 Octubre – 9 Novembre

Una vegada tenim clar que ha de fer la aplicació y quins sons els protocols necessaris per poder arribar el nostra objectiu, podem començar el anàlisis i disseny de la nostra aplicació.

- Identificar les classes necessàries per poder tractar el protocol TCP/IP.
- Anàlisis dels diferents apartats de la aplicació
- Diagrama de les classes de la aplicació i com estaran connectades entre elles.
- Organigrames o pseudocodi per les classes més importants de la aplicació.
- Identificar possibles problemes que podem tenir alhora de la implementació
- Analitzar el que seria el funcionament correcta de una xarxa, depenen de la seva velocitat i la carrega que tingui
- Analitzar el cost per fer que la aplicació també funcioni sobre xarxes wifi, si no es molt elevat, fer també el diagrama i les modificacions necessàries en els organigrames per adaptar l'aplicació a aquest tipus de xarxes.

Implementació de la aplicació i joc de proves. 10 Novembre – 21 Desembre

Implementa la aplicació amb el disseny obtingut i amb les classes identificades en el anàlisis. Farem servir el pseudocodi per implementar les diferents classes.

Joc de proves que ens ajudi a comprovar que la aplicació esta funcionant correctament. Comprovarem si la aplicació ens funció ajudant-nos del anàlisis previ que em fet, sapiguem que ens hauria de donar si la xarxa funciona correctament.

Documentació de la aplicació. 22 desembre – 28 desembre

Fer el manual de la aplicació amb els següents apartats:

- Instal·lació de la aplicació.
- Tot el que pot fer la aplicació.
- Explicar pas a pas com fer les diferents operacions.
- Indicar per quines xarxes i sistemes operatius es pot fer servir la aplicació.

Preparació de la presentació i acabar la memòria 29 desembre – 11 gener

En aquest punt ja ha de estar acabada la aplicació i els manuals fets. Tocarà acabar de preparar la memòria i preparar la presentació de la aplicació.
Crear un presentació que faci referència a la memòria i a la aplicació.

Comentari final de la planificació del projecte

Al acabar el projecte he vist que les coses no han anat com em principi volia. Les dates anàlisis es van desviar massa ja que vaig trigar bastant en descobrir com agafar paquets de la xarxa amb el JPCAP, a partir d'aquí les dates es van endarrerir en tots els seus passos. En l'apart del anàlisis, quant vaig començar a investigar el funcionament del JPCAP, vaig veure altres formes i possibilitats que tenia per fer el projecte que es desviaven del plantejament inicial, com per exemple fer investigacions de més protocols i no només centrar-me en el protocol TCP/IP.
A més, vaig començar la part de implementació quant encara estava en l'anàlisi fent uns petits prototips per poder veure les possibilitats que tenia i aprendre amb la practica.
El apartat d'implementació també vaig trigar més del plantejament inicial perquè vaig tenir que aprendre com funcionava la classe swing de java per poder fer l'entorn gràfic.
Amb tot això va fer que el projecte es desvies bastant de les dates establertes inicialment però tot i així he pogut acabar dintre de les dates i aprendre bastant sobre el món de les xarxes lligat amb la programació.

1.5 PRODUCTES OBTINGUTS

Els producte obtingut és una aplicació per captura els paquets que passen per la targeta de xarxa.
Aquest producte el primer que fa és demanar-te que indiquis per quina interfície de xarxa vols agafar els paquets(per si tens més de una) i no et deixarà agafar paquets en el cas que no tinguis targeta de xarxa.
Tot seguit començarà a agafar els paquets que haguem indicat en el filtrat, si no hem posat cap filtre agafarà i analitzarà tots els paquets que passin per la xarxa. Aquest paquets els anirà mostrant en una taula amb la informació concreta de cada protocol que hem definit.
Quant acabem de agafar paquets ens mostrarà el resum dels totals de paquets agafats i ens deixarà accedir a les gràfiques de cada protocol. Una última utilitat que té és indicar quantes IP's ha capturat en tots els paquets capturats, és una manera de saber si tenim algun intrús a la nostra xarxa.

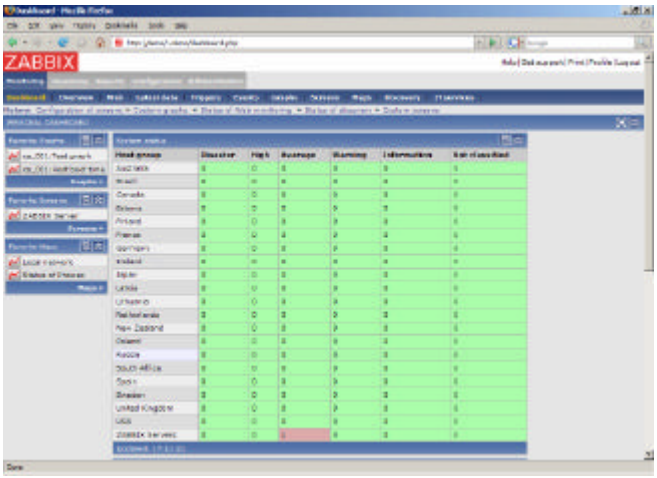
1.6 BREU DESCRIPCIÓ DELS CAPÍTOLS DE LA MEMÒRIA

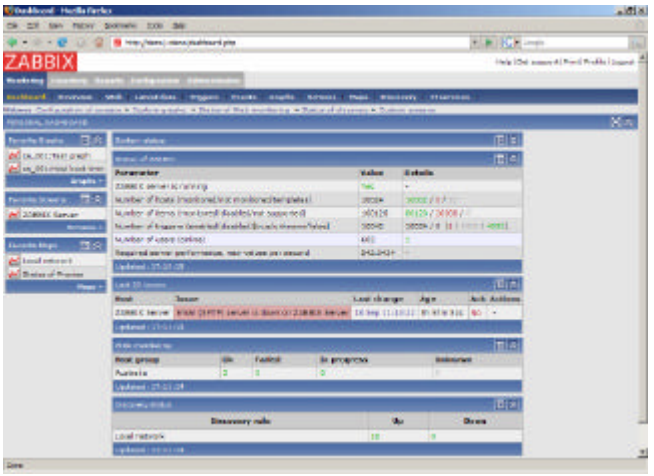
Primer prototip: Explicació del funcionament del primer prototip proposat.
Definició del nostra IF: Mostra el if que fem servir per detectar el tipus de protocol que és cada paquet.
Integració del primer prototip amb el format IF: Com integrem el prototip inicial amb la manera de classificar el diferents protocols.
Classes creades: Enumerem les classes del nostra projecte.
Creació interfícies de la aplicació: Mostrem com són les interfícies de la aplicació.
Programació de les interfícies: Explicació de la programació feta en les interfícies creades anteriorment.
Finestra "Selecció d'interfícies": Explicació del funcionament d'aquesta interfície.
Thread encarregat d'agafar paquets: Explicació de com funciona el thread que agafar els paquets.
Modificació de les taules: Part on expliquem com mostrem les dades.
Començar a agafar paquets: Accions que succeeixen quant prement el botó "Començar".
Parar el Thread de agafar paquets Accions que succeeixen quant prement el botó "Parar".
La instrucció swich del mètode receivePacket(): Part on s'explica que fem dintre del mètode receivePacket().
Mètode de detecció IP's : Procediment per saber les IP's detectades per el nostra aplicació.
Interfície d'equips detectats: Funcionament de la interfície "Equips Detectats".
Estadístiques: Aquí expliquem quines són les estadístiques que farem i com mostrarem aquestes dades dintre d'uns gràfics.
Problemes trobats en la implementació: Comentari sobre els problemes més importants sorgits en el procés d'implementació.

Recerca d'aplicacions similars

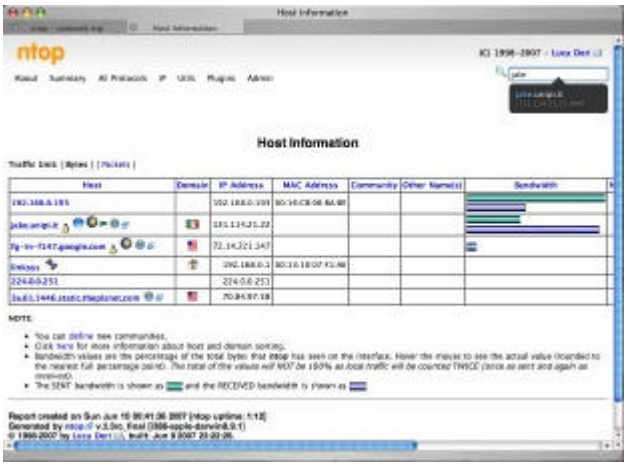
Tot seguit farem una petita explicació dels diferents tipus de aplicacions que hi ha actualment en el mercat, amb diferents possibilitats, totes elles estan encaminades el control de la xarxa.

ZABBIX: Es tracta d'un sistema de monitorització de la xarxa i els seus components. Està basat en un entorn Web, on es podem configurar i visualitzar el estat de tots els serveis que suporta SNMP i el sistemes de xarxa.

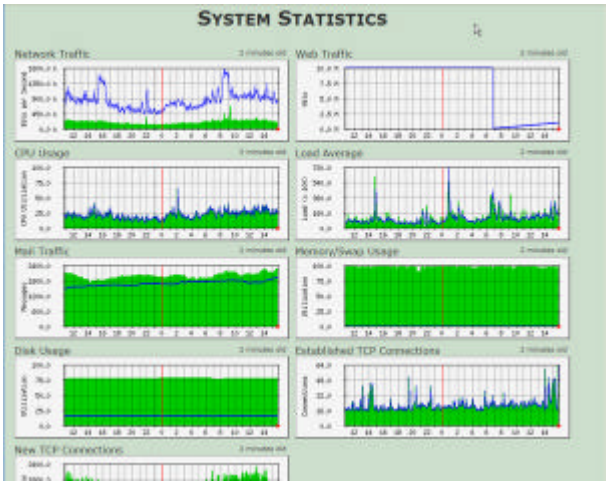


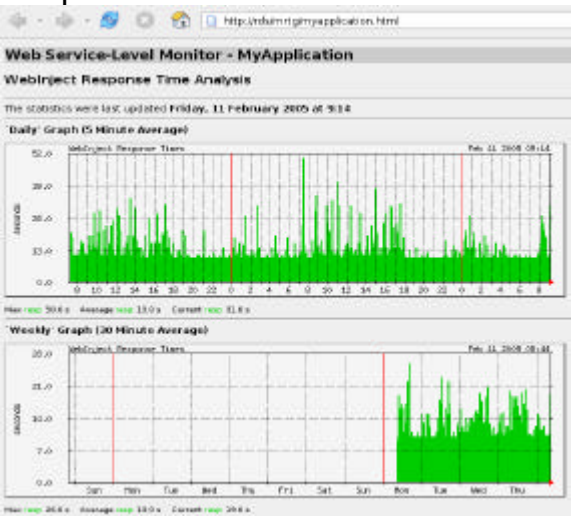


NTOP: És una eina que permet monitoritzar en temps real els usuaris i aplicacions que estan consumint els recursos de la xarxa en un instant en concret, i a més, a ser capaços de ajudar-nos a l'hora de detectar males configuracions de algun equip o a nivell de servei. Té una petita part web que permet a qualsevol usuari que tingui la contrasenya, pugui veure la sortida del NTOP i d'aquesta manera podem veure els seus resultats remotament des de qualsevol màquina.

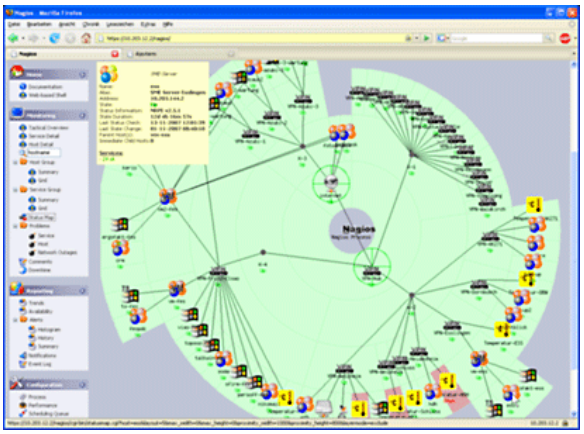


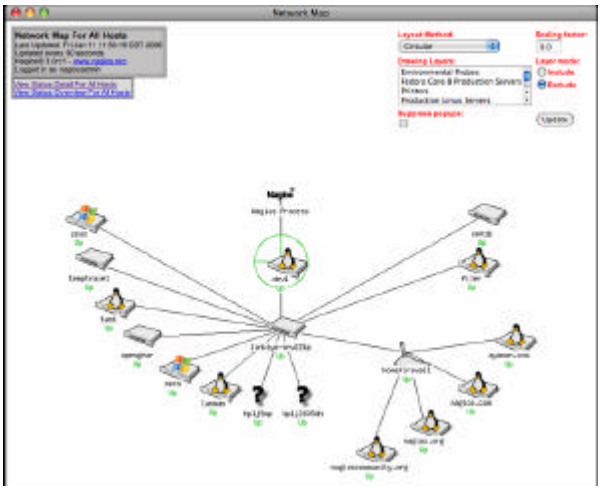
MRTG: eina per monitoritzar la carrega del tràfic o el estat de la nostra xarxa, existeixen multitud de serveis com squid, apache, snort... que tenen plantilles predefinides per monitoritzar els paràmetres específics de cada servei amb MRTG. MTRG genera pàgines HTML que contenen imatges PNG que ens proporcionen una informació visual del tràfic dels nostres dispositius.



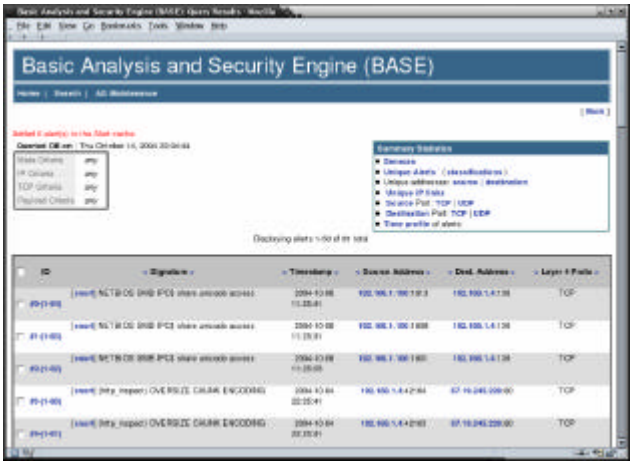


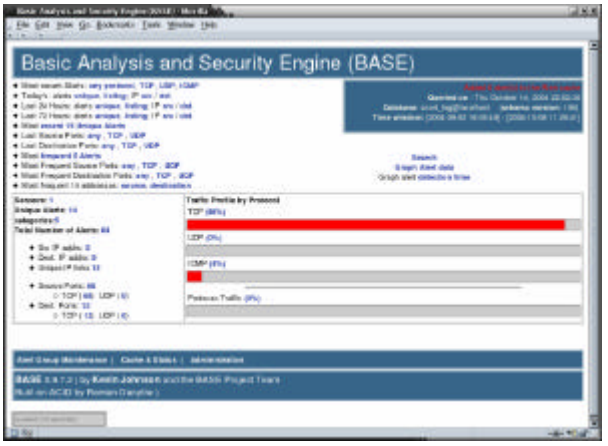
Nagios: Sistema de monitorització de equips i serveis, dissenyat per informar dels problemes de la nostra xarxa. Dissenyat per sistemes Linux. El dimoni de monitorització realitza consultes intermitents en els equips i en els serveis específics fent servir “plugins” externs els quals ens retornen informacions a Nagios.



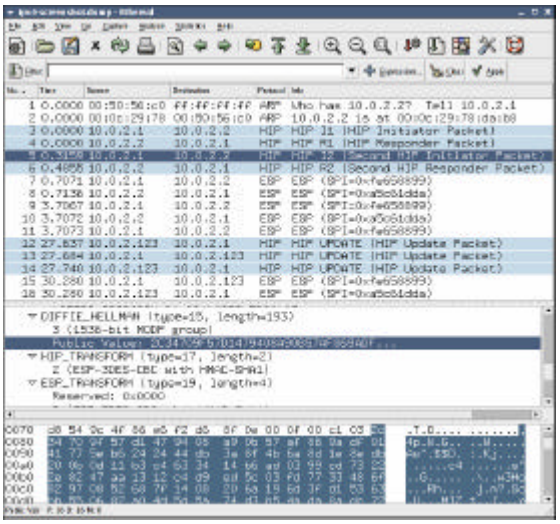


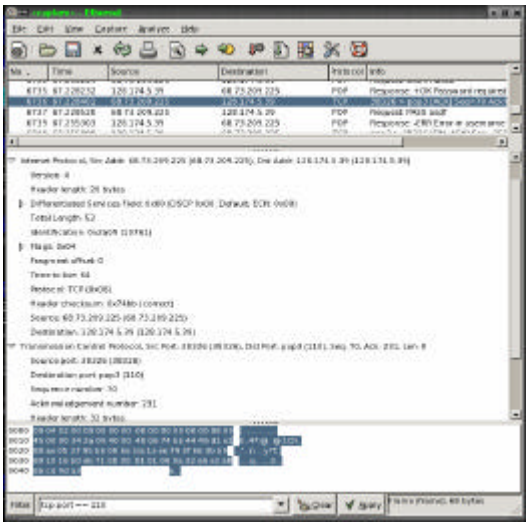
Snort: Sistema de detecció de intrusos, capaç de generar anàlisis del tràfic en temps real i generar logs de paquets IP. Pot realitzar anàlisis de protocols, cerques de patrons establerts i es pot fer servir per a detectar gran varietat de atacs e intents, com per exemple, buffer overflows, escaneixos de ports silenciosos, atacs CGI, escaneixos SMB, fingerprinting, etc.



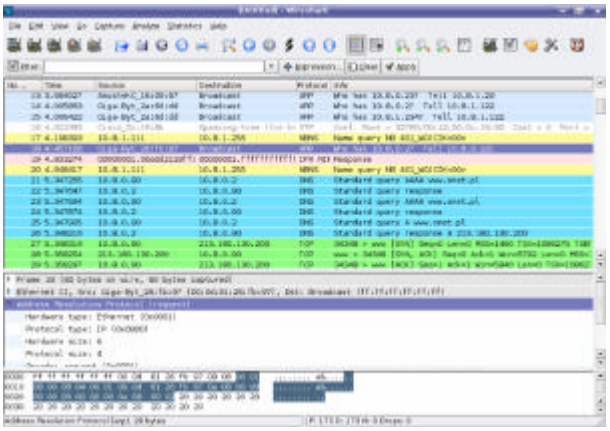


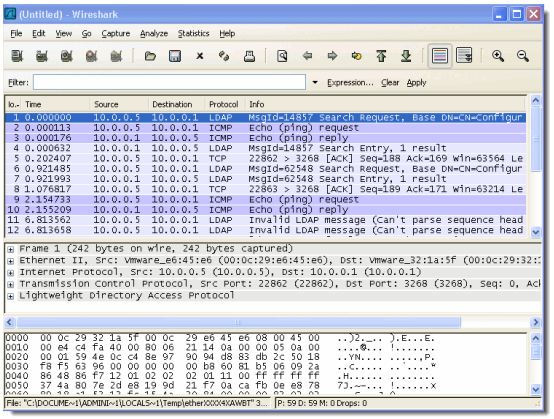
Ethereal: És un analitzador de protocols de xarxa lliure per a linux i windows. Permet examinar dades de la xarxa en viu, o capturar-los en disc. És poden cercar interactivament entre les dades capturades, veure resums i detalls de informació de cada paquet. Conté moltes característiques poderoses que inclouen un complet llenguatge de filtres i la habilitat de veure el fluix reconstruït en una sessió TCP. Aquesta eina va ser clau per el desenvolupament de Samba.



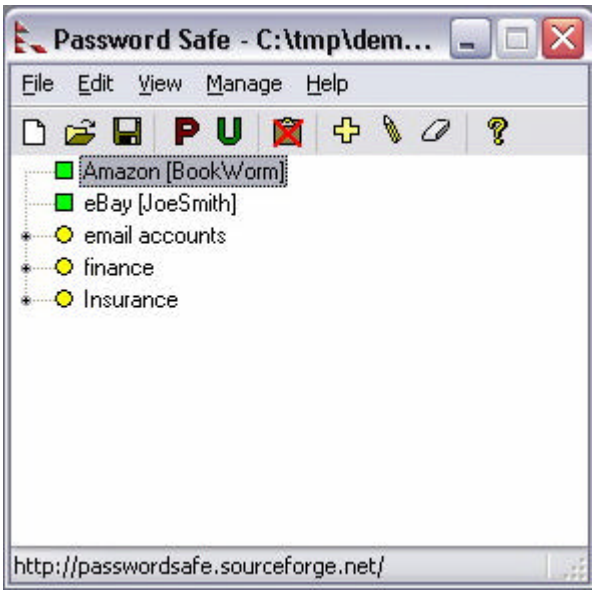


WireShark: És la versió nova del Ethereal, és un analitzador de protocols utilitzat per realitzar anàlisis i solucionar problemes de les xarxes de comunicacions, i com eina didàctica per ensenya. És programari lliure i corre sobre Linux, Solaris, FreeBSD, NetBSD, OpenBSD i Mac OS X, així com en Microsoft Windows.





Password Safe: Gestiona totes les contrasenyes dels dispositius. Ens permet tenir totes les contrasenyes de tots els programes, equips i webs que fem servir normalment, sense tenir que recordar-les.



2 ESTUDIS PREVIS

Petita explicació del funcionament del protocols que volem analitzar

En aquest punt explicarem el funcionament que té cada protocol que volem estudiar, no és un estudi amb profunditat de cada protocol però hem de estudiar una mica cada un d'ells per tenir una idea del que podem analitzar.

Aquí tenim una petita imatge de la pila de protocols TCP/IP on podem localitzar els protocols que volem analitzar:

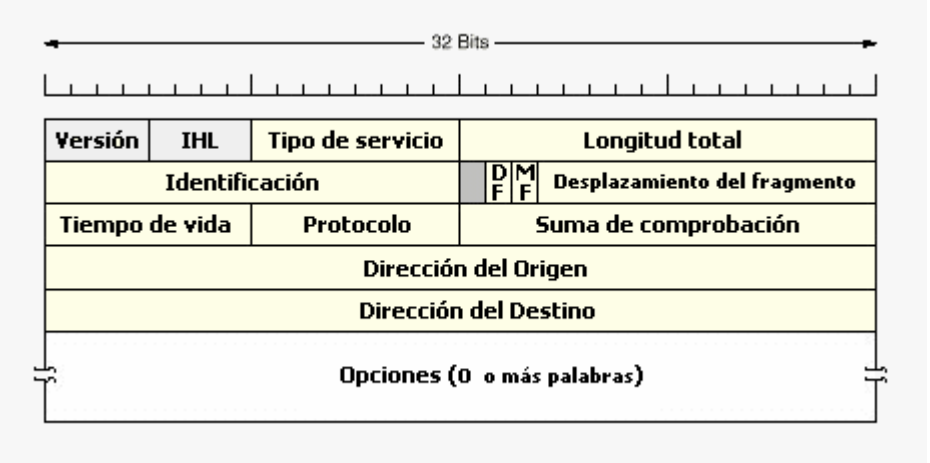
Si ens fixem en la imatge anterior de la pila TCP/IP, podem observar que els protocols IP, ICMP i ARP estan a la capa de nivell de xarxa, mentre que els protocols TCP i UDP els trobem en la capa de nivell de transport.

Protocol IP

El protocol IP(Internet Protocol) és un protocol no orientat a connexió fet servir tant per l'origen com per el destí de la comunicació de dades a traves de una xarxa de paquets commutats.

Les dades en una xarxa basada en IP són enviades en blocs coneguts com a paquets o datagrames(en el protocol IP aquestes dos denominacions es fan servir indistintament). En IP no es necessita cap configuració abans de que un equip intenti enviar paquets a un altre equip que no s'hagin comunicat abans.
IP no té cap mecanisme per saber si un paquet arriba o no el seu destí i únicament proporciona seguretat de les seves capçaleres i no de les dades transmeses(això ho fa mitjan checksums o sumes de comprovació).Si es necessita fiabilitat, aquesta es proporcionada per la capa de transport.
Si la informació a enviar supera la grandària màxima (MTU) en el tram de xarxa per que el circula, podrà ser dividida en paquets més petits i adjuntada quant sigui necessari. A més, aquest fragments poden anar per camins diferents depenen de com de congestionada estigui la xarxa.
Les capçaleres IP contenen les direccions de les màquines de origen i destí, direccions que es faran servir per els commutadors i els encaminadors per decidir per quins trams de xarxa són transmesos aquest paquets.

Imatge capçalera IP:

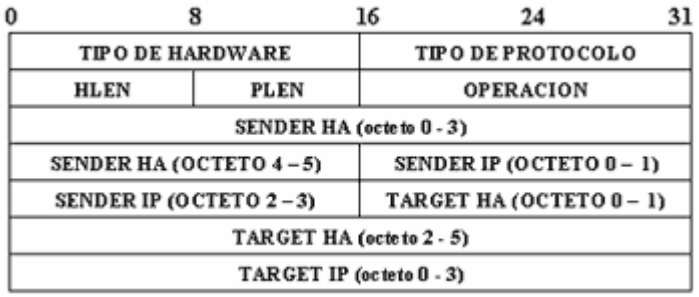


Protocol ARP

El protocol ARP (Address Resolution Protocol) és un protocol de nivell de xarxa responsable de trobar la direcció de Hardware (Ethernet MAC) que correspon a una determinada direcció IP. Per això s'envia un paquet ARP (ARP request) a la direcció de difusió de la xarxa (broadcast(MAC==ff ff ff ff ff ff)) que conté la direcció IP per la que es pregunta, i espera que una màquina respongui (ARP reply) amb la direcció Ethernet corresponen. Cada màquina manté una caché amb les direccions traduïdes per a reduir el retard i la carrega. ARP permet a la direcció de Internet ser independent de la direcció Ethernet, però això només funciona si totes les màquines ho suporten.
ARP es fa servir en els següents caos de comunicació:

- Quant 2 equips estan a la mateixa xarxa i un vol enviar un paquet a un altre.
- Quant 2 equips estan en xarxes diferents i han de fer servir un encaminador per arriba el altre equip.
- Quant un encaminador necessita enviar un paquet a un equip a traves d'un altre encaminador.
- Quant un encaminador necessita enviar un paquet a un equip de la mateixa xarxa.

Imatge capçalera ARP:

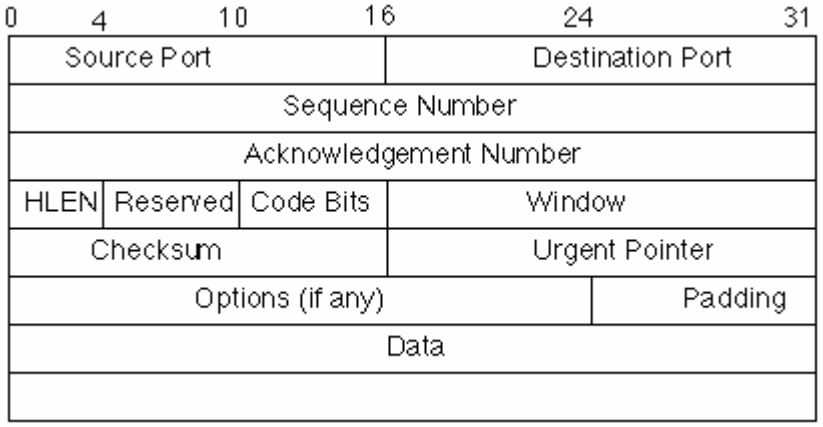


Protocol TCP

El protocol TCP (Transmission Control Protocol) és un dels protocols fonamentals en Internet. Va ser creat entre els any 1973 – 1974. El protocol garanteix que les dades seran lliurades en el seu destí sense errors i en el mateix ordre que van ser transmeses. També proporciona un mecanisme per distingir distintes aplicacions dintre d'una mateixa màquina, a traves del concepte de port. Per exemple, el port 80 per HTTP, el 25 per SMTP, el 21 per FTP, etc.

TCP és un protocol de comunicació orientat a connexió i fiable del nivell de transport.
En la pila de protocols TCP/IP, TCP és la capa intermèdia entre el protocol de internet(IP) i la aplicació. Habitualment, les aplicacions necessiten que la comunicació sigui fiable i, ja que la capa IP aporta un servei de datagrames no fiable (sense confirmació), TCP afegeix les funcions necessàries per a prestar un servei que permeti que la comunicació entre els dos sistemes sigui: lliure d'errors, sense pèrdues i amb seguretat.

Imatge de la capçalera TCP:



Protocol UDP

El protocol UDP (User Datagram Protocol) és un protocol de nivell de transport basat en el intercanvi de datagrames. Permet l'envio de datagrames a traves de la xarxa sense que s'hagi establert prèviament una connexió, ja que el propi datagrama incorpora suficient informació de la direcció on ha de anar a la seva capçalera. Tampoc té confirmació, ni control de flux, per lo tant, els paquets poden avançar-se els uns als altres. Tampoc es pot saber si ha arribat correctament, ja que no hi ha confirmació de lliurament o de recepció. El seu ús principal és per els protocols com el DHCP, BOOTP, DNS i demés protocols en els que el intercanvi de paquets de la connexió/desconnexió son majors, o no son rentables amb respecte a la informació transmesa, així com per a la transmissió de àudio i vídeo en temps real, on no es possible realitzar retransmissions per els estrictes requisits de retard que es tenen en aquest casos.

Imatge capçalera UDP:

0---3	4---7	8---11	12---15	16---19	20---23	24---27	28---31

Puerto de origen UDP	Puerto de destino UDP
Longitud del mensaje UDP	UDP Checksum
Datos	

Comparació protocols de transport TCP i UDP

UDP proporciona un nivell de transport no fiable de datagrames, ja que pràcticament no afegeix informació al paquet que envia al nivell inferior, només la necessària per a la comunicació extrem a extrem. El fan servir aplicacions NFS(Network File System) i RCP(Comandes per a copiar fitxers entre ordinadors remots), però sobre tot s'utilitza en feines de control i en la retransmissió de àudio i vídeo a traves de la xarxa. No introdueix cap retard per establir una connexió. No manté cap estat de connexió i no realitza cap seguiment de aquest paràmetres. Així, un servidor dedicat a una aplicació particular pot suportar més clients actius quant la aplicació funciona sobre UDP en lloc de TCP.

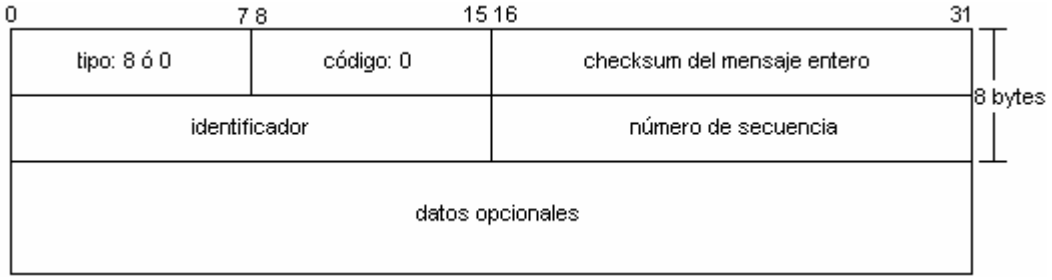
TCP proporciona un transport fiable de flux de bits entre aplicacions. Esta pensat per a poder enviar grans quantitats d'informació de forma fiable, alliberant al programador de aplicacions de la dificultat de gestionar la fiabilitat de la connexió(retransmissions, pèrdues de paquets, ordre en que arriben els paquets, duplicació de paquets, etc) que gestioni el propi protocol. Però la complexitat de la gestió de la fiabilitat té un cost en eficiència, ja que per a porta a terme les gestions anteriors es té que afegir molta informació als paquets a enviar. Degut a que els paquets a enviar tenen una grandària màxima, quanta més informació afegeix el protocol per a la seva gestió, menys informació que prové de la aplicació podrà contenir aquell paquet (el segment TCP té una sobrecàrrega de 20 bytes en cada segment, mentre que UDP només afegeix 8 bytes). Per això, quant es més important la velocitat que la fiabilitat, es fa servir UDP. En canvi, TCP assegura la recepció en destí de la informació a transmetre.

Protocol ICMP

El protocol ICMP(Internet Control Message Protocol) és un subprotocol de control i notificació d'errors del protocol de internet (IP). Tanmateix, es fa servir per enviar missatges d'error, indicant per exemple que un servei determinat no esta disponible o que un encaminador o host no pot ser localitzat.
ICMP difereix del propòsit de TCP i UDP ja que generalment no es poden utilitzar directament per les aplicacions de usuari en la xarxa. La única excepció és la eina de ping i traceroute, que envien missatges de petició Echo ICMP (i rep missatges de resposta Echo) per a determinar si un host esta disponible, el temps que es prenen als paquets en anar i tornar a aquest host i la quantitat de hosts per els que passa.

ICMP es part del conjunt de protocols IP. Els missatges ICMP son generats en resposta d'errors en els datagrames de IP o per el diagnòstic.
La versió de ICMP per a IPv4 és la ICMPv4, així com per IPv6 existeix la versió ICMPv6.
Els missatges ICMP són construïts en el nivell de capa de xarxa. IP encapsula el missatge ICMP apropiat amb una nova capçalera IP (per obtenir els missatges de resposta des de el host original que l'envia), i transmet el datagrama resultant de manera habitual.

Imatge paquet ICMP:



La llibreria JPCAP

La llibreria JPCAP es una llibreria de Java per captura i enviar paquets de xarxa, gràcies a ella, podem desenvolupar aplicacions que capturin paquets de la targeta de xarxa i visualitzar-les o analitzar-les amb Java. També podem desenvolupar aplicacions que enviïn paquets a traves de la nostra xarxa.

JPCAP ha sigut testejat sobre Microsoft Windows(98/200/XP/VISTA), Linux(Fedora, Mandriva, Ubuntu), Mac OS X(Darwin), FreeBSD i Solaris, i pot captura paquets Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP i ICMPv4.
També tenim la possibilitat de guardar paquets capturats en un arxiu i desprès analitzar aquest arxius que tenim guardats.
Una característica molt important és que automàticament identifica els tipus de paquets i genera els corresponents objectes Java (Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP i ICMPv4).
Un altre característica molt important d'aquesta llibreria és que ens permet aplicar uns filtres abans de capturar els paquets.

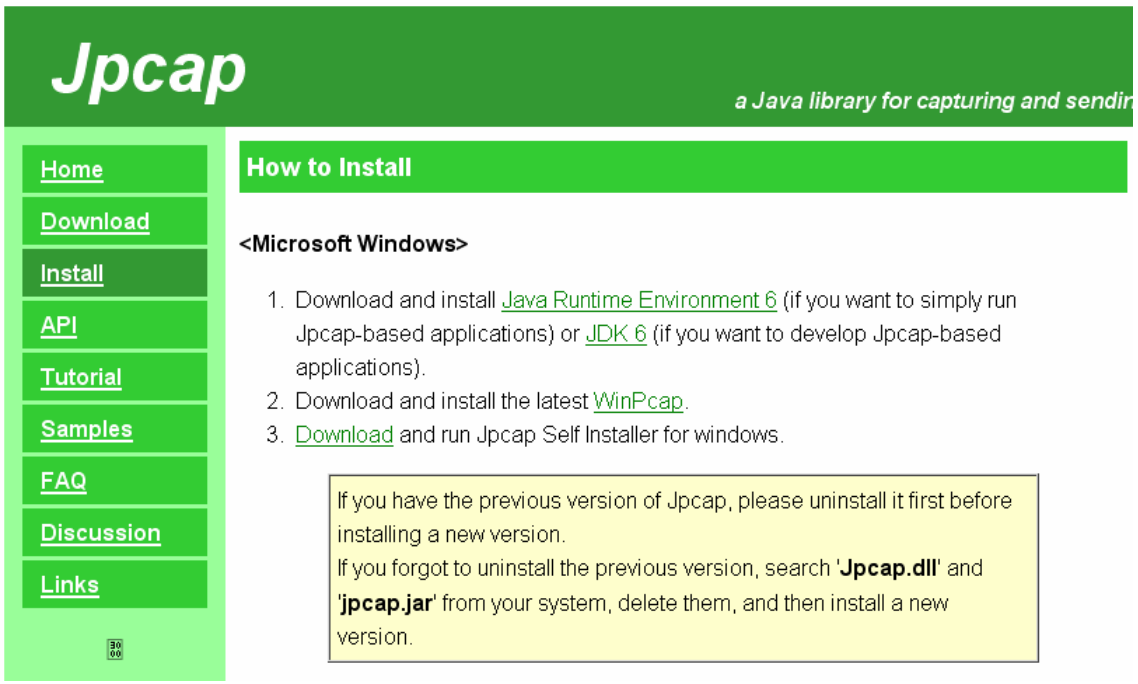
Per últim comentar que la llibreria JPCAP és de codi obert i la seva llicència esta sota GNU LGPL. A part, esta basat en libpcap/winpcap, i es pot implementar ,a part de en Java , en C.

Nota important: JPCAP captura paquets Wifi per algunes distribucions de Linux, Si fem servir Windows no capturarà cap paquet Wifi.

Instal·lació JPCAP

Per instal·lar la llibreria JPCAP hem de anar a la següent pàgina:

http://netresearch.ics.uci.edu/kfujii/jpcap/doc/install.html



Depenen del tipus de sistema operatiu que vulguis fer servir hauràs de seguir unes instruccions o unes altres. En el nostra cas hem fet servir Windows.

1- Descarreguem i instal·lem el JDK6.
2- Descarreguem i instal·lem la última versió del WinPcap.
3- Per últim descarreguem i instal·lem la JPCAP per a windows.

Funcionament JPCAP

Obtenció de interfícies de xarxa

Quant vulguem captura paquets de la xarxa, el primer que hem de especificar és per quina interfície volem captura els paquets.

JPCAP proporciona el mètode JpcapCaptor.getDeviceList() que retornar un vector de objectes de interfícies de xarxa.

El següent codi de exemple, obté una llista de les interfícies de xarxa i mostra la informació:

```
//Obtain the list of network interfaces
NetworkInterface[] devices = JpcapCaptor.getDeviceList();

//for each network interface
for (int i = 0; i < devices.length; i++) {
  //print out its name and description
  System.out.println(i+": "+devices[i].name + "(" + devices[i].description+")");

  //print out its datalink name and description
  System.out.println(" datalink: "+devices[i].datalink_name + "(" +
devices[i].datalink_description+")");

  //print out its MAC address
  System.out.print(" MAC address:");
  for (byte b : devices[i].mac_address)
    System.out.print(Integer.toHexString(b&0xff) + ":");
  System.out.println();

  //print out its IP address, subnet mask and broadcast address
  for (NetworkInterfaceAddress a : devices[i].addresses)
    System.out.println(" address:"+a.address + " " + a.subnet + " "+ a.broadcast);
}
```

Obrir una interfície de xarxa

Una vegada hem obtingut una llistat de les interfícies de xarxa de les que disposem i hem escollit per quina volem captura els paquets, podem obrir la interfície fem servir el mètode JpcapCaptor.openDevice().

Un petit exemple de com treballar aquest mètode:

```
NetworkInterface[] devices = JpcapCaptor.getDeviceList();
int index=...;  // set index of the interface that you want to open.

//Open an interface with openDevice(NetworkInterface intrface, int snaplen, boolean
promics, int to_ms)
JpcapCaptor captor=JpcapCaptor.openDevice(device[index], 65535, false, 20);
```

Quant cridem a aquest mètode podem especificar els següents paràmetres:

NetworkInterface intrface: Interfície de xarxa que volem obrir.

Int snaplen: Màxim número de bytes capturats per paquets.

Boolean promics: posarem aquest valor en True si volem que la nostra interfície estigui en mode promiscu. En cas contrari posem en False.

Int to_ms: Estableix el temps de espera per captura paquets.

JpcapCaptor.openDevice() retorna una instancia de JpcapCaptor. Llavors podem cridar varis mètodes de la classe JpcapCaptor que captura paquets des de la interfície de xarxa.

Captura paquets des de la interfície de xarxa

Una vegada ja tenim una instància del JpcapCaptor, ja serem capaços de captura paquets des de aquesta interfície.

Hi ha dues maneres de captura paquets en JPCAP. Fent servir un mètode callback o captura paquets un a un.

Mètode callback:

Primer, hem de implementar el mètode callback, definint una nova classe que implementi la interfície PacketReceiver. La interfície PacketReceiver defineix un mètode receivePacket(), per lo que necessitem implementar el mètode receivePacket() en la nostra classe.

Ara, nosaltres podem cridar els mètodes JpcapCaptor.processPacket() o JpcapCaptor.loopPacket() per començar a captura paquets fent servir el mètode callback.
Quant cridem aquest dos mètodes podem especificar el número de paquets que volem captura. També esta la opció de especificar-lo amb el -1, això farà que capturi paquets infinitament.

El següent codi es un exemple d'us d'aquest mètode:

La nostra classe que implementada de PacketReceiver().

```
class PacketPrinter implements PacketReceiver {
  //this method is called every time Jpcap captures a packet
  public void receivePacket(Packet packet) {
    //just print out a captured packet
    System.out.println(packet);
  }
}
```

En aquesta part del codi es on cridem al mètode processPacket(), que passem per paràmetre el número de paquets que ha de captura i li passem també per paràmetre la classe que hem implementat abans.

```
JpcapCaptor captor=JpcapCaptor.openDevice(device[index], 65535, false, 20);

//call processPacket() to let Jpcap call PacketPrinter.receivePacket() for every packet
capture.
captor.processPacket(10,new PacketPrinter());

captor.close();
```

3 IMPLEMENTACIÓ

3.1 Primer Prototip

El primer pas que farem es programa un petit prototip que ens capturi un paquet o infinits paquets i que els vagi imprimint per pantalla. Desprès ja podrem començar a saber coses del paquet que hem capturat.

El codi seria el següent:

Prototip.java

```
import jpcap.*;
class prototip implements PacketReceiver {

public static void main(String[] args) throws Exception {
        NetworkInterface[] devices = JpcapCaptor.getDeviceList();
        if(args.length<1){
        System.out.println("usage: java Tcpdump <select a number from the following>");
        for (int i = 0; i < devices.length; i++) {
        System.out.println(i+" :"+devices[i].name + "(" + devices[i].description+")");
        System.out.println("    data link:"+devices[i].datalink_name +
"("+devices[i].datalink_description+")");
        System.out.print("    MAC address:");
        for (byte b : devices[i].mac_address)
        System.out.print(Integer.toHexString(b&0xff) + ":");
        System.out.println();
        for (NetworkInterfaceAddress a : devices[i].addresses)
        System.out.println("    address:"+a.address + " " + a.subnet + " "+ a.broadcast);
        }
        }else{
JpcapCaptor jpcap = JpcapCaptor.openDevice(devices[Integer.parseInt(args[0])], 65535,
true, 20);
        jpcap.loopPacket(-1, new prototip());

public void receivePacket(Packet packet) {
        System.out.println(packet.toString());
}
```

En la crida a jpcap.loopPacket(-1,new prototip); en el primer paràmetre podem dir quants paquets volem agafar, o com en el nostra exemple posar -1 i fer una captura continua. El segon paràmetre és on es fa referència al callback perquè passem com a paràmetre un objecte. Desprès anirem a parar a la classe receivePacket, amb el paquet capturat, en el nostra petit prototip només el mostrem per pantalla, però a partir d'aquí es quant podem saber més coses del paquet capturat.

3.2 Definició del nostra IF

Una vegada ens arriba el paquet definirem un format de IF per saber de quin tipus de paquet es tracta, el nostra esquema serà com el següent:

```
si paquet()
    si ippaquet()
        si tcppaquet()

        si no
            si udppaquet

            si no
                si icmppaquet

                si no
                    altrespaquetsIP
                fi si
            fi si
        fi si
    si no
        si arppaquet

        si no
            altrespaquets
        fi si
    fi si
fi si
```

3.3 Integració del primer prototip amb el format del IF

Ara, el que farem es crear una classe per cada protocol que volem estudiar, en el nostra cas tenim, IP.java, TCP.java, UDP.java. ARP.java i ICMP.java. En cada un d'ells farem els mètodes per aconseguir les dades que volem d'aquests paquets.
Tot seguit, podem aplicar el format del if comentat en el apartat anterior, en el mètode receivePacket() on en el primer prototip només mostràvem les dades sense cap tipus de comprovació.
Ara sabrem en cada moment de quin tipus de paquet estem parlant, podrem col·locar els comptadors per més tard fer les estadístiques i mostrar el que vulguem depenen del tipus de paquet que sigui.

Fins aquí, tenim un codi que ens agafar el paquet, comprova de quin tipus de protocol és i crea el seu objecte.

3.4 Classes creades

En aquest apartat de la memòria explicarem que fan cadascuna de les nostres classes.

Projecte.java

Aquesta és la classe principal, és la classe que té el main que executa l'aplicació i crea l'objecte Jframe que és qui controla les interfícies.

Jframe.java

És la classe més important de l'aplicació, tenim des de el codi per carrega totes les interfícies fins a com gestionar els paquets capturats.

Aquesta classe té les següents funcions:

- Codi de totes les interfícies gràfiques.
- Codi que controla el treball entre totes les gràfiques, els seus events, etc.
- Codi amb el thread que controla els paquets que van arribant, on hem implantat el PacketReceiver.
- Les variables que fan de comptadors.
- La funció que compta les IP detectades.

IP.java

En aquesta classe tenim totes les operacions que volem fer amb aquest protocol i les seves dades.

Un array de String, títols: On posem els títols que tindran les capçaleres de la taula quant capturem paquets IP.
El seu propi constructor.
I els mètodes que retornen valors: getTitols(), getIPdst(), getIPsrc(), getIPVersio(), isType().

TCP.java

En aquesta classe deixarem tots els mètodes relacionats amb els paquets de tipus TCP.

Array de String títols: On posem els títols que tindran les capçaleres de la taula quant capturem paquets TCP, en aquest cas els títols són els següents: “IP origen", "IP destí", "Tipus de TCP”.
El seu propi constructor.
I els mètodes que retornen valors: getTitols(), getIPdst(), getIPsrc(), isType().

UDP.java

La classe UDP igualment que amb les anteriors, anirem deixant el codi per treballar amb aquest tipus de protocols.

Array de String títols: On posem els títols que tindran les capçaleres de la taula quant capturem paquets UDP, en aquest cas els títols són els següents: “IP origen", "IP destí", "Tipus de UDP”.
El seu propi constructor.
I els mètodes que retornen valors: getTitols(), getIPdst(), getIPsrc(), isType().

ICMP.java

La classe ICMP, igual que les anteriors però per treballar amb paquets del protocol ICMP.

Array de String títols: On posem els títols que tindran les capçaleres de la taula quant capturem paquets ICMP, en aquest cas els títols són els següents: “IP origen", "IP destí", "Tipus de ICMP”.
El seu propi constructor.
I els mètodes que retornen valors: getTitols(), getIPdst(), getIPsrc(), isType().

ARP.java

Per últim tenim la classe ARP, aquesta classe va una mica apart de les altres ja que no es tracta de paquets IP.

Array de String títols: On van els títols de les capçaleres dels paquets ARP. En aquest cas són: "MAC origen ","IP origen", "MAC destí", "IP destí", "Operació"
El seu constructor.
I els mètodes que retornen valors: getTitols(), getMACdst(),getIPdst(), getMACsrc(), getIPsrc(), getTipusOperacio().

Estadistica.java

Crearem una classe estadística per crear els gràfics amb la llibreria JFreeChart.
En aquesta classe tindrem el seu constructor, els mètodes setValue(), setChar(), createPanel() i show(), i els atributs serialVersionUID del tipus long, dataset del tipus DefaultPieDataset i jfc del tipus JFreeChart.

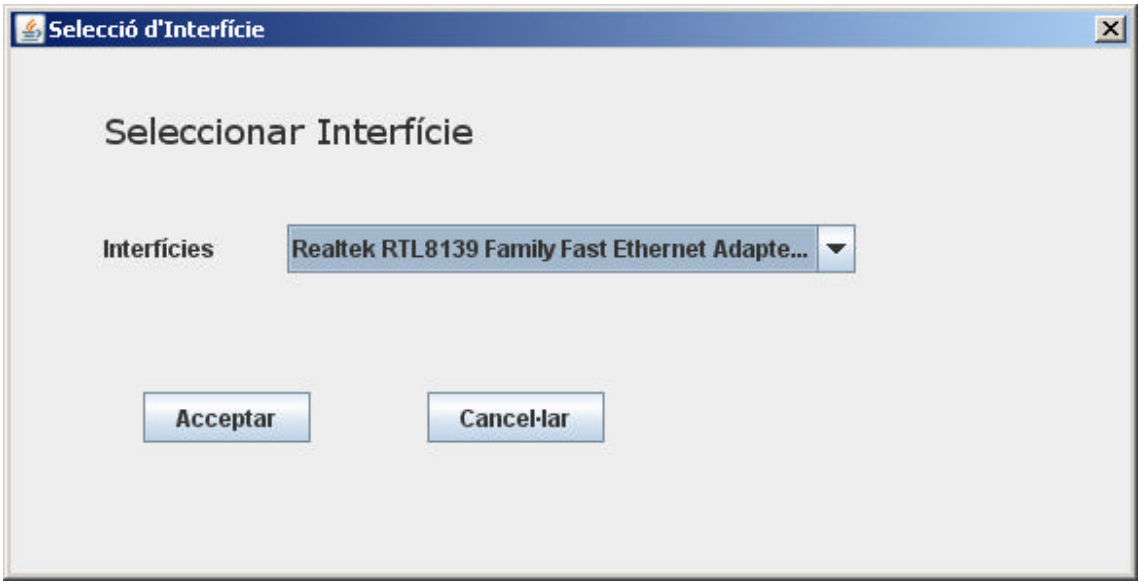
3.5 Creació interfícies de la aplicació

Per crear las interfícies en Java, hem fet servir la classe Swing, amb la qual hem aconseguit fer les interfícies de la aplicació. Per facilitar-nos la feina i aprendre més sobre aquesta classe hem fet servir el Netbeans, que és una aplicació que pots crear les interfícies i ell mateix et genera el codi Java. Una vegada obtenim el codi font l'hem estudiat per entendre'l i poder treballar sobre ell des de l'eclipse. Com hem comentat abans, totes aquestes interfícies estaran creades en la classe, Jframe.java.

A continuació mostrarem les interfícies de la nostra aplicació:

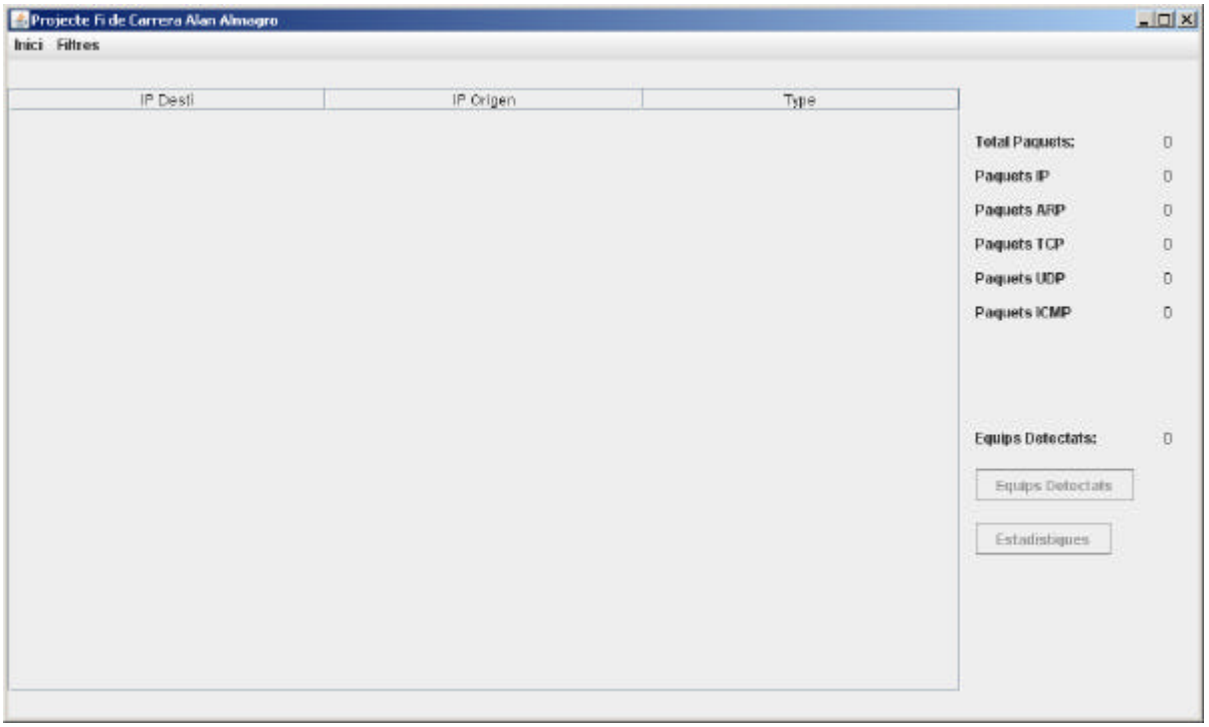
Interfície Selecció d'Interfície

En aquesta finestra seleccionarem per on volem escoltar el que passa per la xarxa, és a dir, si tenim més d'una targeta de xarxa hem de escollir per quina agafarem els paquets, d'aquesta manera també ens assegurem que tenim una targeta de xarxa en el nostra equip.

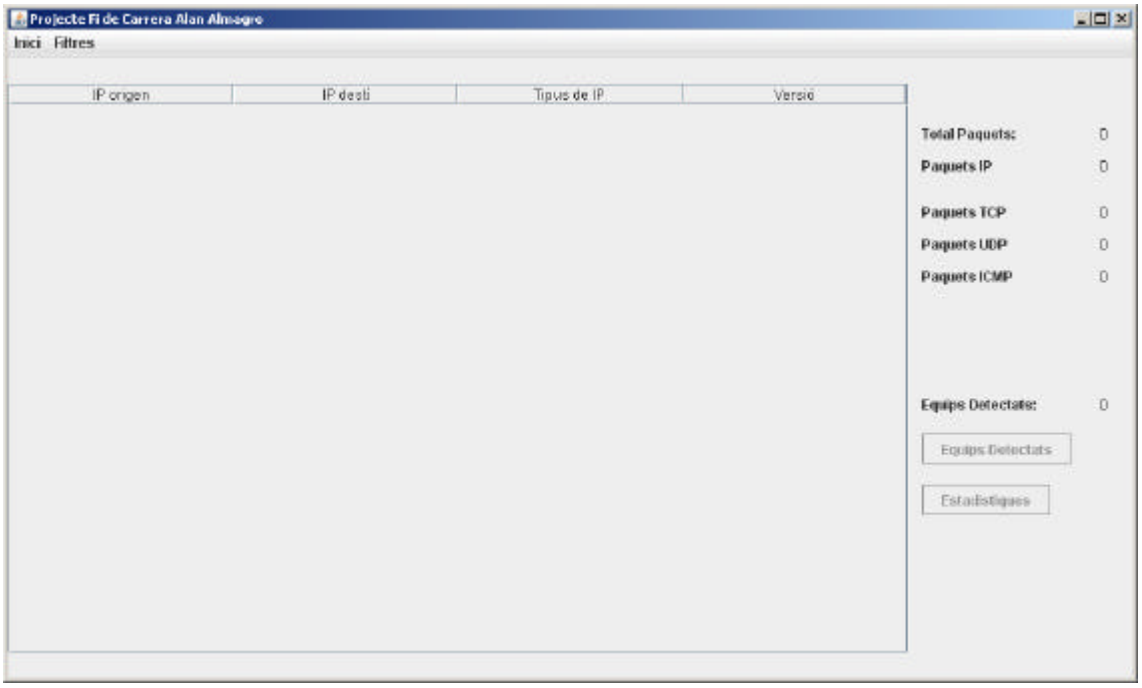


Interfície Inicial

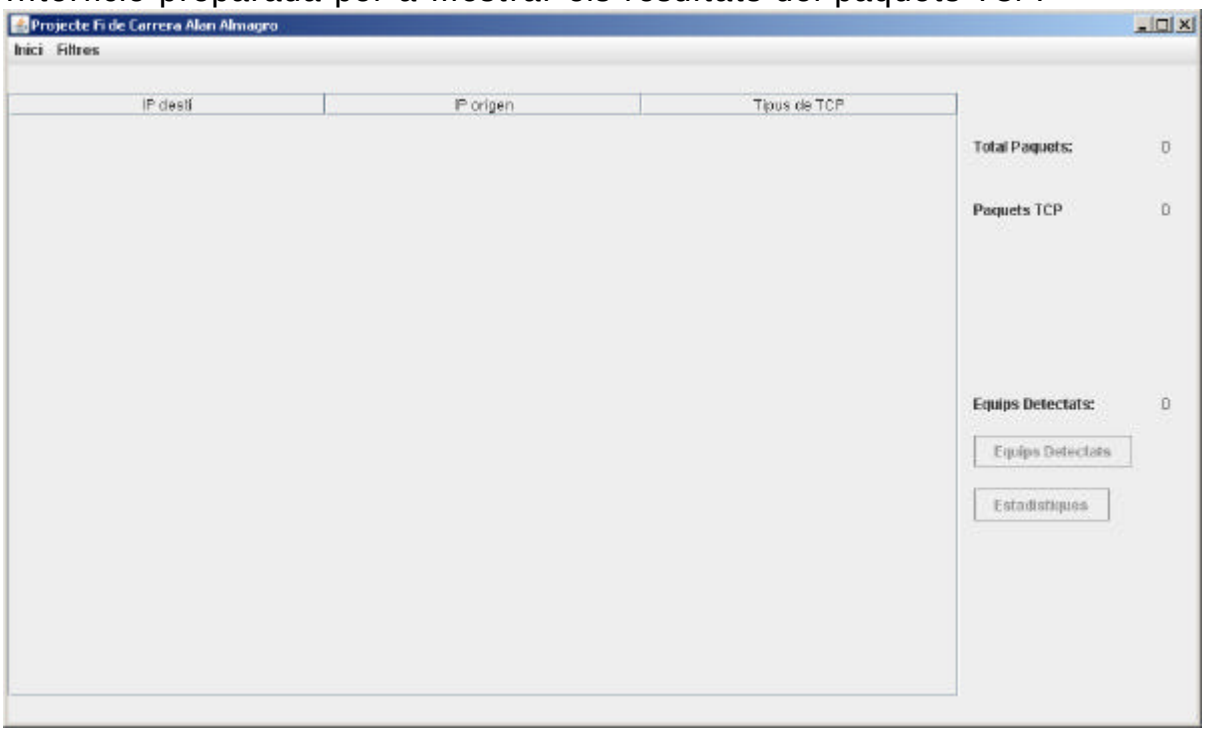
Pantalla principal, on podrem escollir el tipus de filtre que volem aplicar des de el menú filtres. El centre de la pantalla tenim la taula on anirem deixant les dades dels paquets. A la dreta de la finestra tenim les etiquetes amb els resums dels tipus de paquets capturats i els botons per veure estadístiques i els equips detectats.



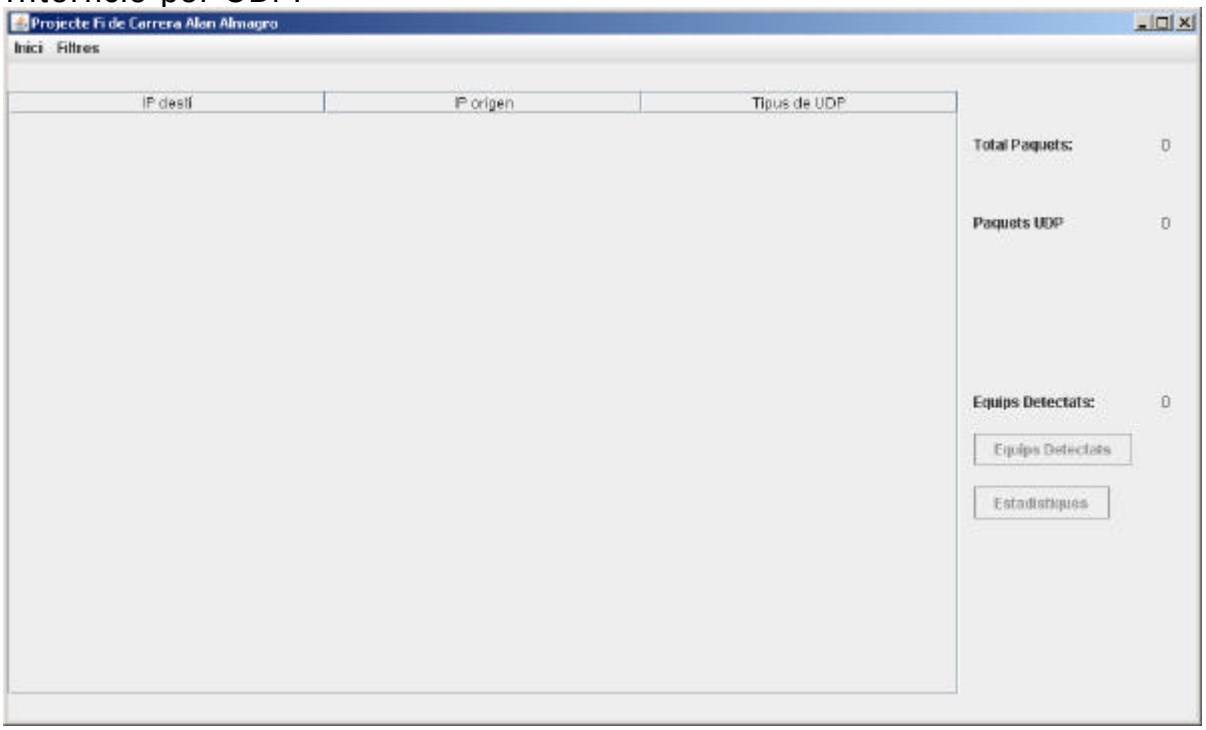
La mateixa interfície però ara adaptada per capturar només paquets IP. Com podem veure, les etiquetes que mostrem són la IP, TCP, UDP i ICMP.



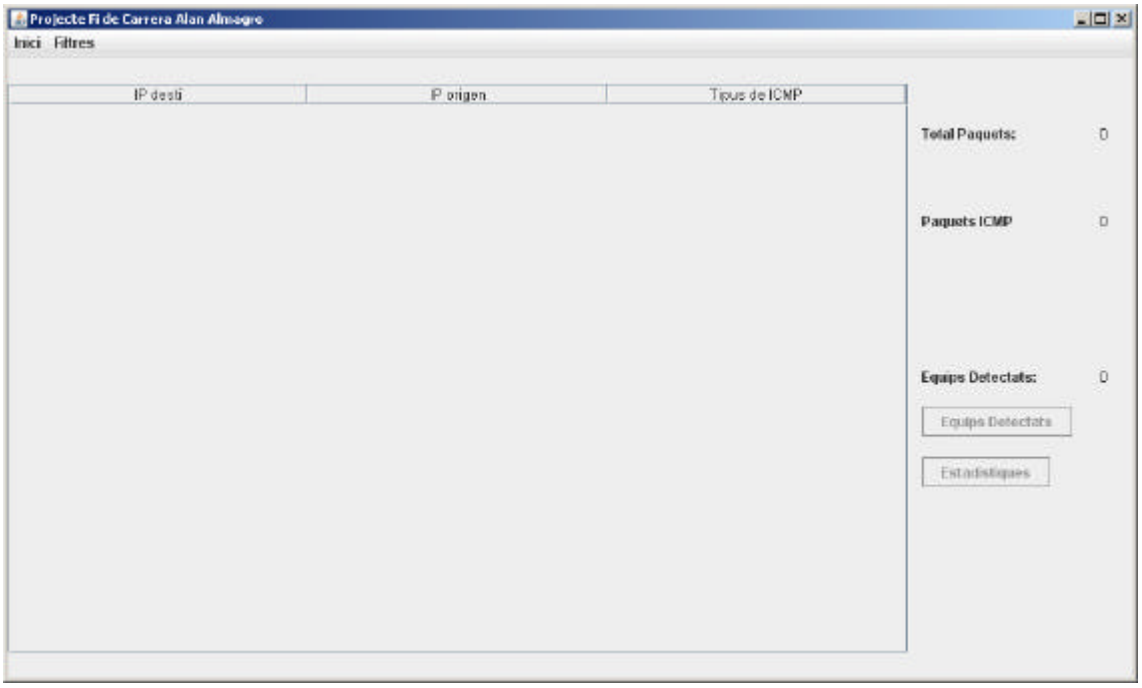
Interfície preparada per a mostrar els resultats del paquets TCP.



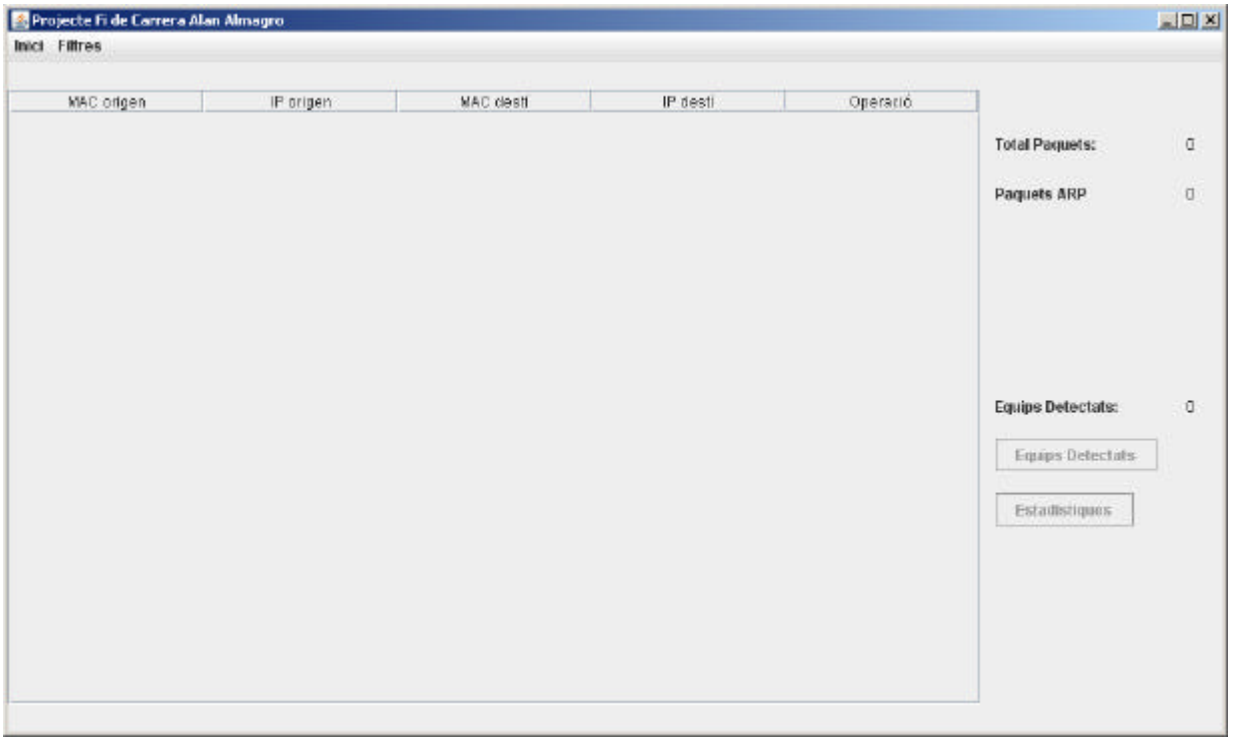
Interfície per UDP.



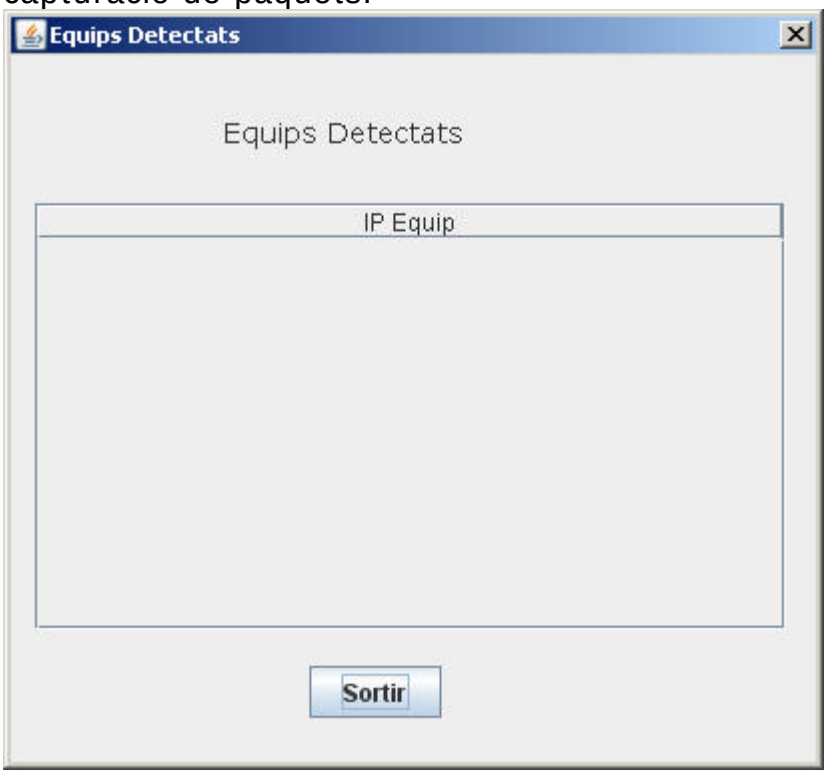
Interfície ICMP



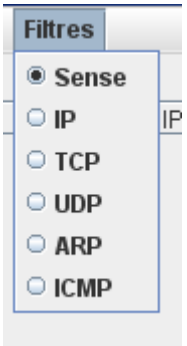
Interfície ARP



En aquesta interfície mostrarem les diferents IP's detectades en la capturació de paquets.



Els menús són els següents:



Com podem observar, tenim el menú INICI, on tenim la opció Començar, al picar aquesta opció s'obrirà la finestra de "Selecció de interfície". La opció Parar, per deixar de agafar paquets, i per últim la opció Exit, per sortir de la aplicació.
Per altre banda, tenim el menú de Filtres, per qui podem especificar el tipus de filtra que volem aplicar. He posat el tipus de filtres dintre d'un grup de botons de radio ja que només podem tenir seleccionada una opció de les sis que tenim.

3.6 Programació de les interfícies

En aquest apartat explicarem el que fa cada opció de les interfícies i el seu funcionament entre si.

Començarem per els menús, tenim el menú Inici i el menú Filtres. Tot seguit explicarem cada una de les funcions que tenen cada opció.
Menú INICI:

- Començar –> obrirà la finestra "Selecció de interfícies"
- Parar -> Para el fil que agafa els paquets i mostrarà els resultats en les etiquetes.
- Exit -> Surt de la aplicació

Menú Filtres:

- Sense ->Deixarà la taula per defecte, amb els camps IP origen, IP destí, Tipus paquet. Les etiquetes de resultats finals també seran les de per defecte, aquestes són: Total Paquets, Paquets IP, Paquets TCP, Paquets UDP, Paquets ARP i Paquets ICMP.
- IP -> A la taula mostrarà els camps: IP origen, IP destí, Tipus de IP i Versió IP. Les etiquetes que mostrarà seran Total Paquets, Paquets IP, Paquets TCP, Paquets UDP i Paquets ICMP. Es a dir, totes menys les dels paquets ARP.
- TCP -> A la taula mostrarà els camps: IP origen, IP destí, Tipus de TCP. Les etiquetes que mostrarà seran Total Paquets i Paquets TCP.
- UDP -> A la taula mostrarà els camps: IP origen, IP destí, Tipus de UDP. Les etiquetes que mostrarà seran Total Paquets i Paquets UDP.
- ARP -> A la taula mostrarà els camps: MAC origen, IP origen, MAC destí, IP destí i Operació. Les etiquetes que mostrarà seran Total Paquets i Paquets ARP.
- ICMP -> A la taula mostrarà els camps: IP origen, IP destí, Tipus de ICMP. Les etiquetes que mostrarà seran Total Paquets i Paquets ICMP.

També tenim dos botons a la finestra principal:

- Botó Equips Detectats -> S'obrirà la finestra de "Equips detectats" i farà que s'executi el mètode per indicar les ips detectades.
- Botó Estadístiques -> S'obrirà una pàgina Web amb la estadística dels resultats obtinguts.

Es important dir que aquest dos botons només estaran actius quant hagi alguna captura activa, es a dir, quant hagi dades a les taules.

3.7 Finestra “Selecció d’Interfície”

A continuació explicarem amb detall el funcionament d’aquesta finestra.

Al cridar aquesta finestra el primer que fem es executar el bucle que ens afegirà dintre del comboBox les interfícies que tenim en el nostra equip. Aquestes interfícies les detectem abans, quant hem creat el constructor del Jframe, i les guardem amb un vector de NetworkInterface.

En la interfície tenim dos botons:
El Botó “Cancel·lar” : Oculta aquesta finestra sense fer cap acció més.
El Botó “Acceptar”:

- Arranca el Thread encarregat d’agafar els paquets que passin per la interfície seleccionada el comboBox.
- Desactiva el menú dels filtres perquè no es puguin modifica mentre estiguem agafant paquets.
- Desactivar el menú Començar perquè no es produeixin errors i l’usuari no pugui tornar a obrir la finestra “Selecció d’Interfícies” fins que no es pari d’agafar paquets.
- Activar el menú Parar, per poder acabar d’agafar paquets.
- Oculta la finestra “Selecció d’Interfícies”.

3.8 Thread encarregat d’agafar paquets

Quant vaig començar l’aplicació no era conscients que necessitaria fer servir els threads (fils) per poder continuar amb l’aplicació.

El problema va sorgir quant vaig crear el mètode que anés agafant els paquets i mostrar-los a les taules, aquest mètode feia que no pugues obrir el menú per poder parar d’agafar paquets ja que tenia un bucle treballant alhora.
Per poder fer això vaig trobar l’opció dels threads, que et permeten executar varies accions alhora.
Llavors el que vaig fer es crear un thread per executar el mètode d’agafar els paquets de la interfície i la aplicació lliure per anar responent les peticions que fes l’usuari.

El mètode que té el Thread és el startGetPacket() i el seu funcionament és el següent:

- Obrim la interfície amb el mètode openDevice del nostra JpcapCaptor.
- Comprovem si hem d’aplicar algun filtre o no.

- Creem el Thread i en el seu mètode run posem un bucle que no pari d'agafar paquets fins que la variable captureThread sigui igual a null.
- Per últim executem el Thread.

La variable captureThread esta controlada per l'acció del botó parar, és en aquesta acció on posem a null aquesta variable i per lo tant sortirem del thread que finalitzarà.

3.9 Modificació taules

Arribats a aquest punt, teníem que saber com podriem canviar els toString del primer prototip per noves files a les taules creades amb la classe Swing.

Primer hem de saber que vam crear les taules de classe DefaultTableModel(), aquesta classe té el mètode add per afegir files i el mètode removeRow per eliminar-les.

Ara cada vegada que volem imprimir un resultat de un paquet capturat, el que farem es crida el mètode add de la taula corresponent, passant-li per paràmetres les dades que ha de imprimir a la nova fila.

Un altre punt important en les taules és que cada vegada que comencem a captura paquets de nou hem de esborrar les dades que hi ha a la taula de la antiga captura, per això farem servir els mètodes removeRow i el mètode getRowCount, el primer serveix per anar esborrant files i el segon ens va indicant quantes files quedant per esborrar a la taula. Aquesta acció la farem cada vegada que cridem el menú començar.

Un exemple de com quedaria la taula amb dades dels paquets ho tenim en la imatge següent:

IP origen	IP destí	Tipus paquet
/192.168.2.1	/224.0.0.1	IGMP
/192.168.2.100	/224.0.0.22	IGMP
/192.168.2.101	/224.0.0.22	IGMP
/192.168.2.101	/66.102.9.99	HTTP
/66.102.9.99	/192.168.2.101	HTTP
/192.168.2.101	/66.102.9.99	HTTP
/192.168.2.101	/66.102.9.99	HTTP
/66.102.9.99	/192.168.2.101	HTTP
/66.102.9.99	/192.168.2.101	HTTP
/192.168.2.101	/66.102.9.104	HTTP
/66.102.9.104	/192.168.2.101	HTTP
/192.168.2.101	/66.102.9.104	HTTP
/192.168.2.101	/66.102.9.104	HTTP
/192.168.2.101	/66.102.9.99	HTTP
/66.102.9.104	/192.168.2.101	HTTP
/66.102.9.104	/192.168.2.101	HTTP
/192.168.2.101	/66.102.9.147	HTTP
/66.102.9.147	/192.168.2.101	HTTP
/192.168.2.101	/66.102.9.147	HTTP
/192.168.2.101	/66.102.9.147	HTTP
/192.168.2.101	/66.102.9.104	HTTP
/66.102.9.147	/192.168.2.101	HTTP
/66.102.9.147	/192.168.2.101	HTTP
/66.102.9.147	/192.168.2.101	HTTP
/192.168.2.101	/66.102.9.147	HTTP

Total Paquets:	25
Paquets IP	25
Paquets ARP	0
Paquets TCP	22
Paquets UDP	0
Paquets ICMP	0
Equips Detectats:	8

Equips Detectats

Estadístiques

3.10 Acció Començar agafar paquets

Les accions que explicarem a continuació és el que succeeix desprès de prémer el botó començar.

Quant polsem el botó començar hem de cridar a la finestra “Selecció d’Interfícies”. Per fer això només hem de crida el mètode setVisible de la finestra passant-li per paràmetre el valor true.
També tenim que buidar la taula en cas que tingues dades de una escoltada anterior. Per això vaig tenir un problema, ja que no trobava cap mètode implantat a la taula que la buides automàticament, llavors vaig optar per fer el meu propi mètode que fes aquesta acció. El que faig és, primer comprovo si n’hi ha alguna fila, si no hi ha cap fila es que ja esta buit i que no hi havia dades, en cas contrari entro dintre d’un bucle que va eliminant files, no surto d’aquest bucle fins que no hi hagin més files, és a dir, fins que el mètode de la taula getRowCount és igual a zero. Hem de seguir el mateix procediment amb la taula de les ips detectades.
L’ultima acció que ha de fer és tornar a deixar desactivats els botons Estadística i Equips detectats.

3.11 Acció Parar el Thread de agafar els paquets

Ens preguntem que hem de fer quant l'usuari polsa aquest botó, primer hem de parar el thread que esta agafant paquets constantment.

Com hem explicat en apartats anteriors, la variable que controla que el Thread funcioni es la variable captureThread, aquesta mentre el seu valor sigui diferent de null farà que el Thread continuï treballant. Doncs, el que hem de fer es posar aquesta variable a null i el Thread acabarà la seva acció i deixarà d'agafar paquets.

Una vegada finalitzada la captació de paquets també farem les següents accions:

- Mostrarem els resultats finals en les etiquetes i tornarem a posar els comptadors a zero per una futura captura de paquets.
- Activarem el menú Començar, el menú dels filtres i els botons Estadístiques i Equips Detectats.
- Desactivarem el menú Parar.

3.12 La instrucció swich del mètode receivePacket()

Com sabem, cada vegada que canviem la opció del filtre que volem aplicar, també canviem la taula perquè no són les mateixes dades a mostrar. Ara be, això vol dir que no podem aplicar la mateixa formula alhora d'omplir les dades de la taula.
Per això he creat un swich al mètode reveivePacket() amb les diferents opcions de mostrar les dades, aquest swich esta controlat per una variable que va canviem de valor cada vegada que seleccionem un tipus de protocol diferent.
La estrucutura del Swich serà la següent:

```
Swich (numfiltre)
        Cas 0:
                Format IF(especificat en el apartat anterior)
        Cas 1:
                Protocol IP
        Cas 2:
                Protocol TCP
        Cas 3:
                Protocol UDP
        Cas 4:
                Protocol ICMP
        Cas 5:
                Protocol ARP
```

3.13 Mètode detecció IP's

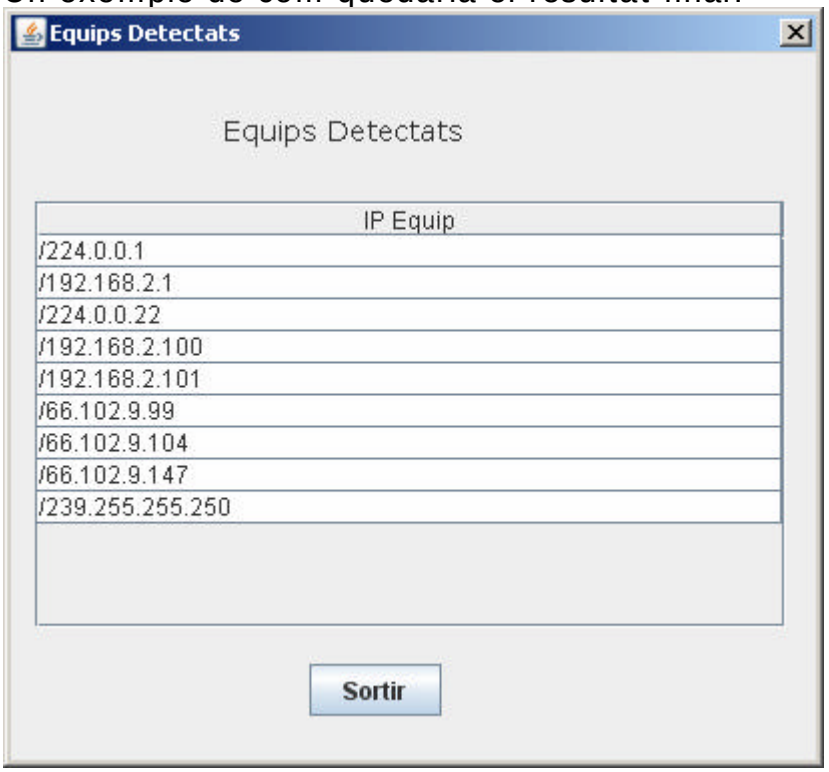
Una de les funcions que volíem donar a la aplicació és que fos capaç de saber quines són les ip's que hem detectat en el intercepció de paquets.
Per fer això hem creat un mètode que és cridat cada vegada que agafem un paquet, aquest mètode té la funció de comprovar, primer, si la ip d'origen del paquet esta en el vector de equipsDetectats, si és així no fa res però si no hi és l'afegeix, i segon, fer la mateixa operació però ara amb la ip de destí del paquet.

3.14 Interfície Equips Detectats

A continuació explicarem amb detall el funcionament d'aquesta finestra.

Com hem explicat anteriorment tenim un mètode que és l'encarregat d'anar controlant les ip's noves que es detectant i afegir-les el vector equipsDetectats. Aquesta Interfície es l'encarregada de mostrar els resultats en el seva taula.
Llavors el que farem quant l'usuari polsi el botó de "Equips detectats" és executar un bucle que recorri el vector de les ip's detectades i les vagi afegint a la taula.

Un exemple de com quedaria el resultat final:



Com podem observar, en aquesta finestra també tenim el botó "Sortir", aquest botó a de fer les següents accions:

- Esborrar les dades que té la taula.
- Oculta la finestra.

3.15 Estadístiques

Les estadístiques que volem mostrar en aquest projecte són:

- Quant ho fem sense filtres: *TCP, UDP, ICMP i ARP*
- Quant tinguem filtrat per IP: *TCP, UDP, ICMP, IGMP i IP encapsulation.*
- Quant tinguem per TCP: *HTTP, HTTPS, FTP, TELNET, SSH, SMTP, POP3, SMB i Netbios*.
- Quant tinguem filtrat per UDP: *Netbios, hsrp, ntp, snmp, dns i ssdp*.
- Quant tinguem filtrat per ICMP: *resposta ICMP, petició ICMP destí inaccessible, disminució del tràfic, temps excedit, problemes de paràmetres, petició de marca de temps, resposta de marca de temps, petició màscara subxarxa i resposta màscara subxarxa*.
- Quant tinguem filtrat per ARP: *Request ARP o Reply ARP*.

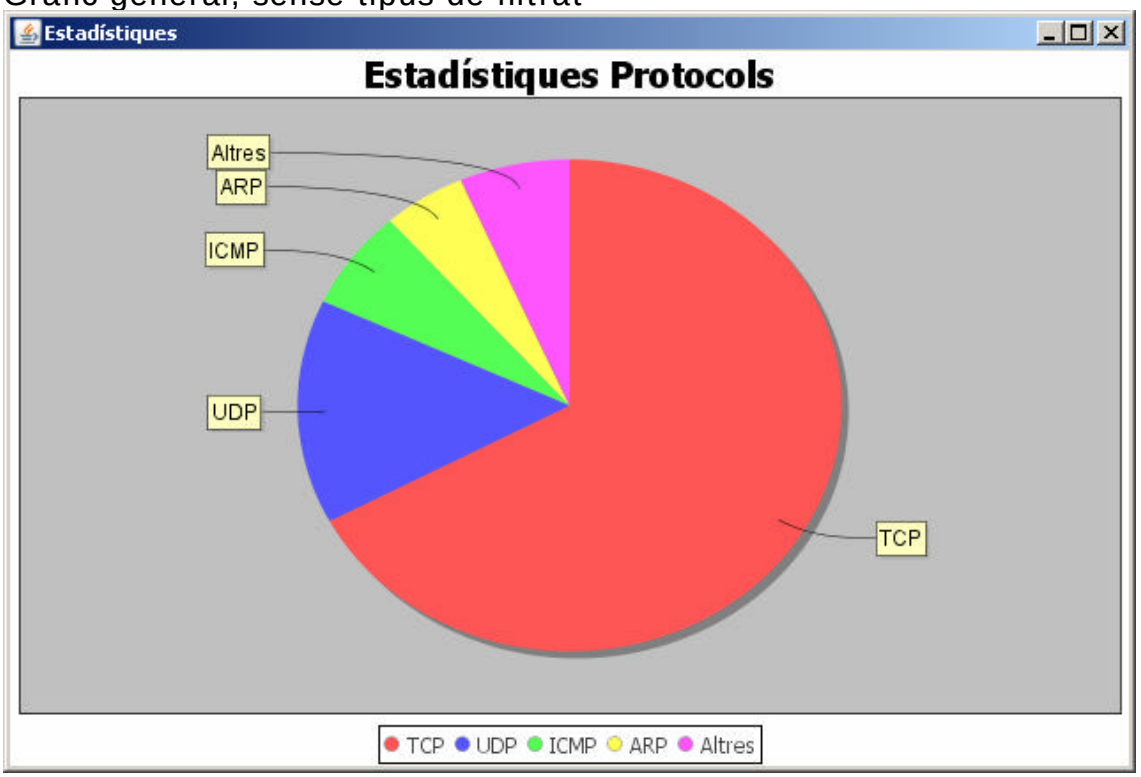
Com ho farem? Doncs l'idea inicial és fer les estadístiques a partir d'uns javascripts que generen estadístiques si els hi passem un vector amb les dades. Llavors, depenen del tipus de estadístiques que vulguem treure, l'enviarem uns comptadors o uns altres.

La idea era fer-ho amb els servlets de Java, ja que era la manera que podia crear una pàgina web amb java i passar-li els paràmetres. El primer inconvenient que m'he trobat és que els servlets necessiten d'un servidor web per poder-se executar, cosa que ja no m'agradava ja que això implicar tenir que instal·lar-li al usuari final un servidor web en la seva màquina. I un altre problema que m'ha fet desistir de fer servir servlets a sigut que no he sigut capaç de cridar a aquest servlet des de el botó Estadística. De fet tampoc he volgut perdre més temps amb aquesta idea ja que la idea de tenir que instal·lar un servidor web no m'agradava gens.

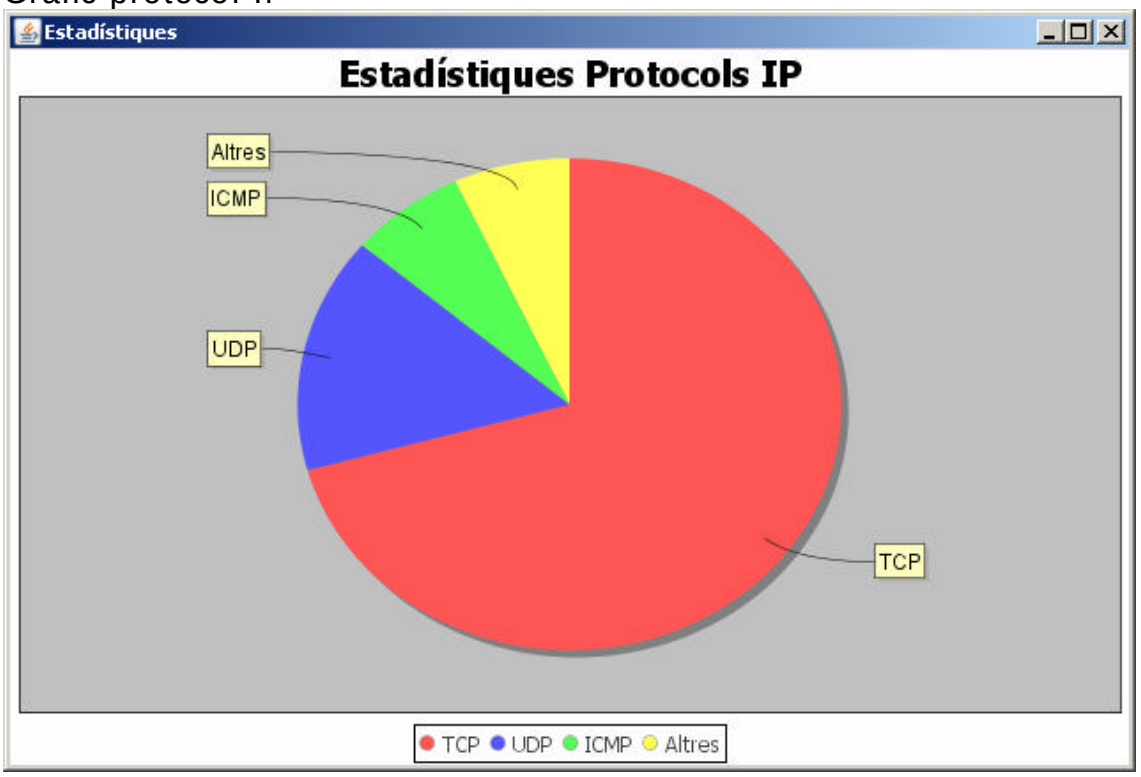
Desprès de investigar una mica més he descobert la llibreria JFREECHART que amb poques línies de codi i bastant senzill pots aconseguir uns gràfics molts bons.

He creat un switch del mateix nivell que el switch del filtres i fent servir el mateix menú filtres aniré canvien el tipus de gràfic que mostraré.
Una opció important que he afegit, es que de la mateixa captura de paquets, pots veure les diferents gràfiques de cadascun dels protocols, és a dir, si fas una captura sense filtra i desprès, d'aquesta captura, vols fixar-te en quins tipus de protocols TCP ha capturat, només has de canviar el tipus de filtra per TCP i tornar-li a donar al botó Estadístiques i et mostrarà la gràfica de TCP corresponent a aquella captura.
A continuació unes imatges de com quedant els gràfics:

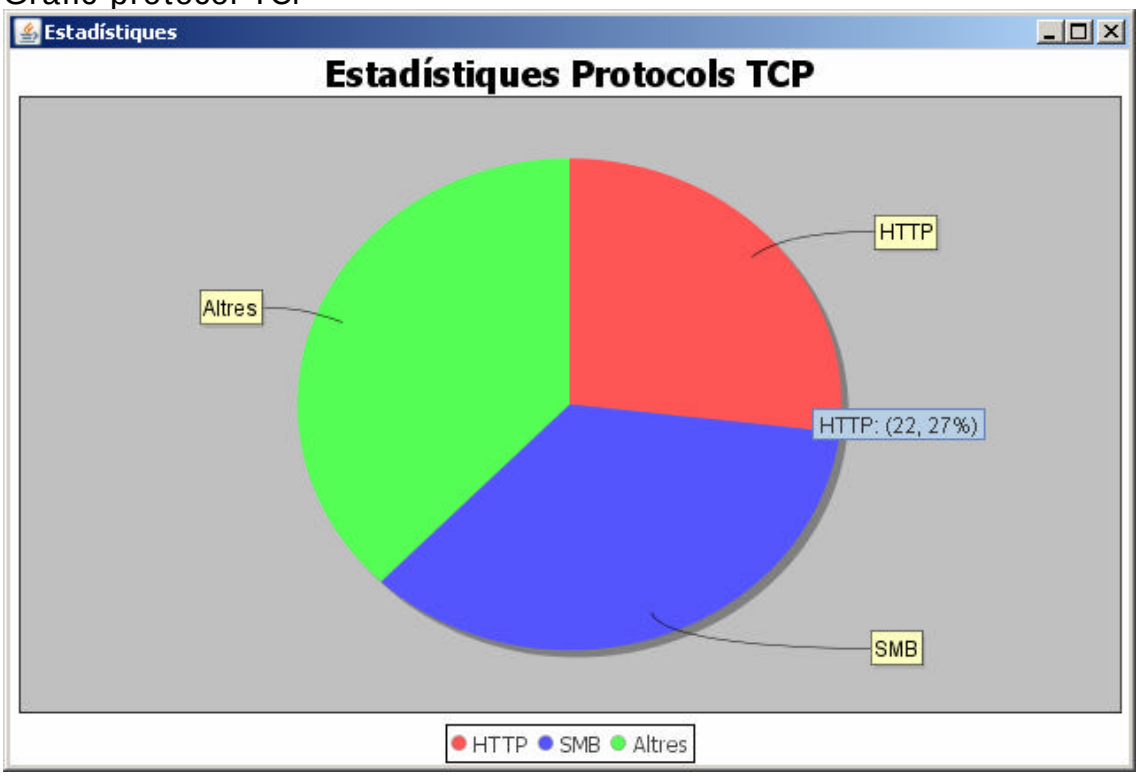
Gràfic general, sense tipus de filtrat



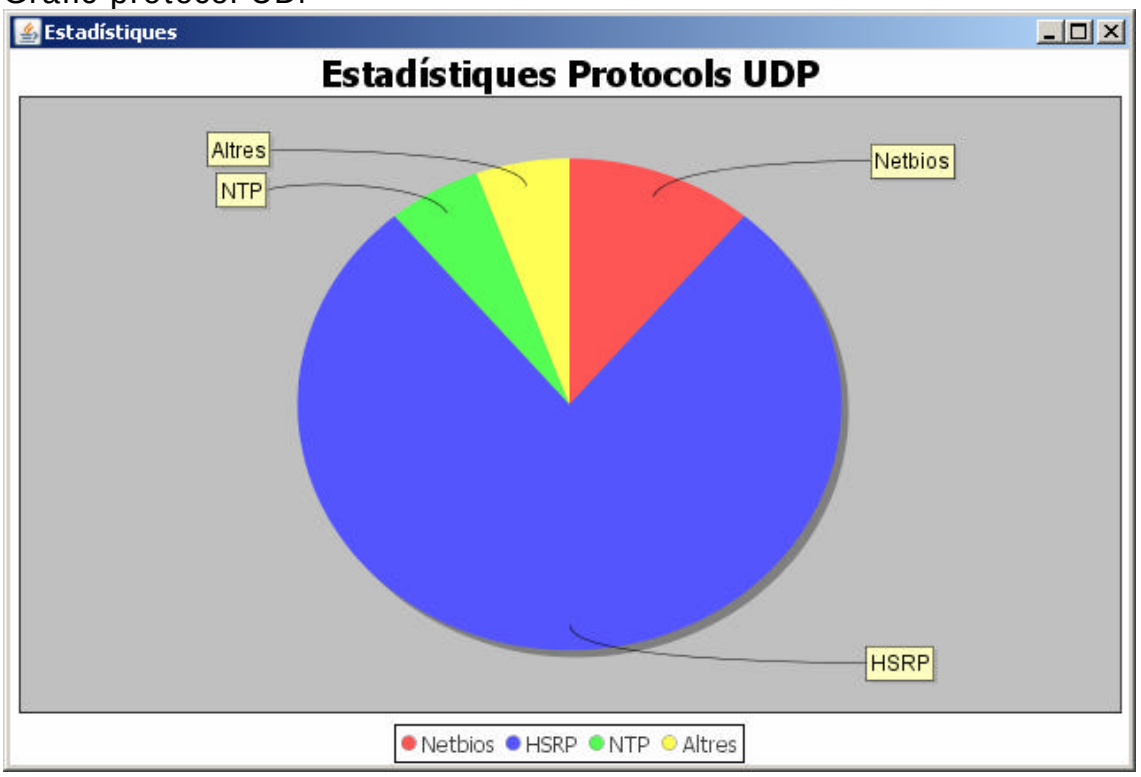
Gràfic protocol IP



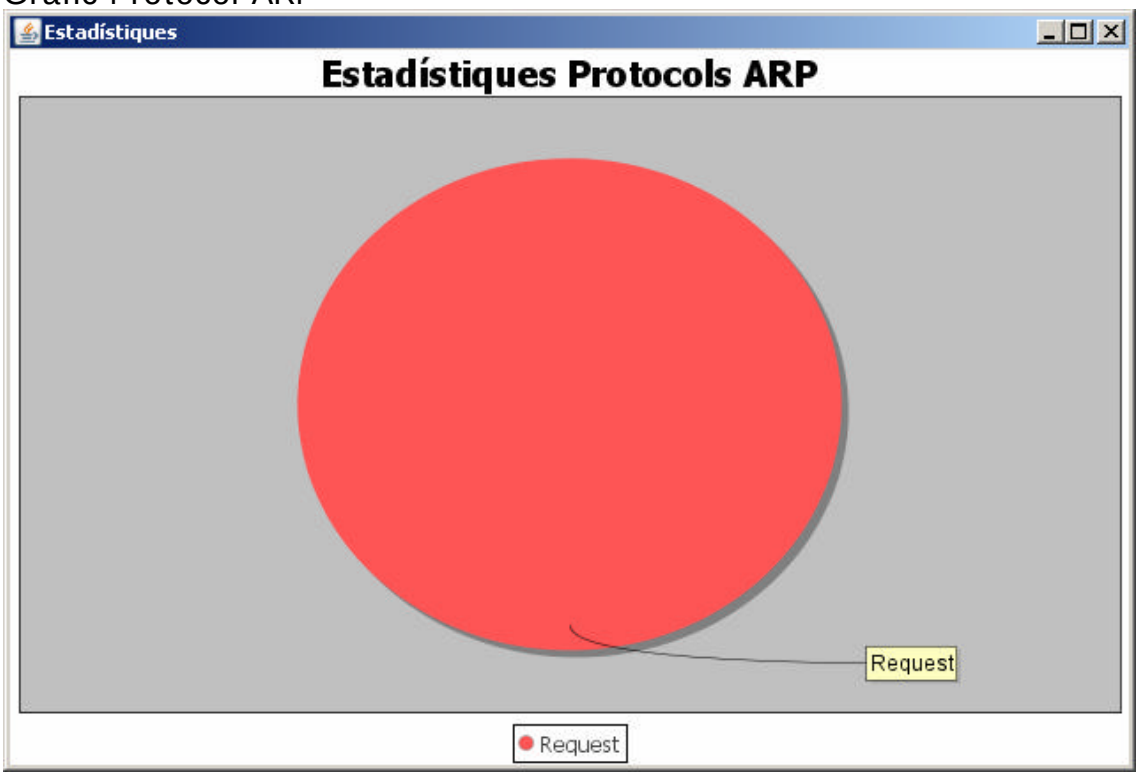
Gràfic protocol TCP



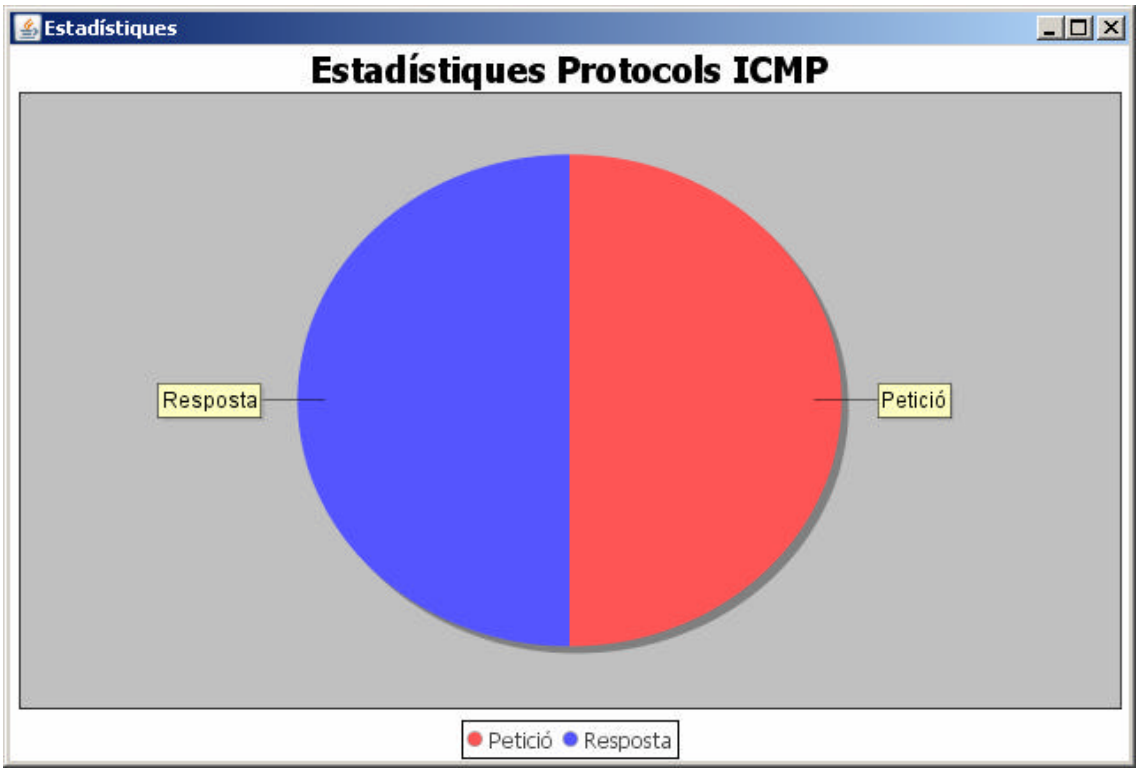
Gràfic protocol UDP



Gràfic Protocol ARP



Gràfic Protocol ICMP



3.16 Problemes trobats en la implementació

La veritat que en el moment de la implementació no he tingut grans problemes, ja que més be he tingut problemes en el anàlisis i en el disseny de com fer la aplicació i quines llibreries tenia que fer servir. Això si, una vegada solucionat les coses han anat bastant bé.

De totes maneres, el principal problema que m'he trobat en la implementació a sigut en el punt de fer els gràfics de les estadístiques. En el anàlisis vaig pensar que podia fer-ho amb servlets, llençant una pàgina web amb uns javascript que passant-li els resultats de les estadístiques et generessin unes gràfiques. Però quant vaig començar la implementació vaig veure que necessitava un servidor web instal·lat a la màquina per poder fer servir servlets, a més, no sabia com executar un servlet des de un botó de la classe swing. El vaig solucionar fent servir les llibreries JFreeChart i crec que m'ha quedat millor que si hagués fet servir servlets.

Un altre problema important va ser quant vaig descobrir que necessitava fer servir un thread per poder continuar amb la aplicació, ja que en començar el bucle que agafava paquets, aquest bucle agafava tots el recursos i no em deixava fer cap acció en el entorn gràfic de la aplicació, i doncs clar, no podia parar de agafar paquets. Això no ho vaig contemplar en el anàlisis, va sorgir quant em vaig trobar amb el problema i la solució com he dit abans, va ser executar el procés de agafar paquets en un thread i així em continuava funcionant el entorn gràfic de la aplicació.

Els demés problemes, més bé han sigut inconvenients, ja que han sigut problemes de no saber com funcionava exactament cada mètodes de les classes swing o com funcionava la llibreria JPCAP.

4 MANUAL D'INSTAL·LACIÓ

Per instal·lar la aplicació en el nostra equip hem de fer els següents passos:

- **Instal·lar JRE de java**: Ho podem descarregar des de la mateixa web de Java *http://www.java.com/es/download/*
- **Instal·lar WinpCap i JPCAP**: Executar els fitxers WinPcap_4_0_2.exe i JpcapSetup-0.7.exe, que estan a la carpeta "Producte TFC Alan Almagro".
- Copiar les llibreries jfreechart-1.0.12.jar i jcommon-1.0.15.jar, que estan a la carpeta "Producte TFC Alan Almagro\lib", en la carpeta del java "Java\jre6\lib\ext"

Per últim hem de executar el fitxer start.bat, ubicat en "Producte TFC Alan Almagro", per arrancar l'aplicació.

4.1 Com Funciona

El funcionament de l'aplicació és el següent:

Al obrir l'aplicació ens hem de anar a la opció Començar del menú Inici, s'obrirà un quadre de diàleg per poder escollir la interfície per la qual agafarem els paquets. Desprès d'això l'aplicació ja començarà a agafar paquets i mostrar el resultats per la taula principal.
Si volem algun tipus de protocol en concret, hem de anar el menú filtres i escollir el que volem, hem de fer això abans de començar a agafar paquets.
Per acabar de agafar paquets i poder veure les estadístiques totals i els seus gràfics, he de anar el menú inici i prémer la opció Parar.
Una vegada tenim una captura acabada, a la dreta podrem veure els resultats del paquets capturats.
Si prement el botó "Estadístiques" podem veure el gràfic de les estadístiques, i si volem veure el resultat d'algun protocol en concret, hem de anar el menú filtre, canviar al filtre desitjat i tornar a prémer el botó "Estadístiques".
Per últim, també podrem prémer el botó "Equips Detectats" on veurem una taula amb les IP que s'han intercanviat els paquets.

5 CONCLUSIONS

Per concloure amb el projecte, bàsicament direm que fer un projecte d'aquestes característiques m'ha ajudat a veure lo complicat que és fer una planificació d'un projecte, sigui quina sigui la seva dimensió, al seu inici. Calcular les dates de cada etapa i explicar que farem a cada etapa, pot arribar a ser una tasca molt complicada, ja que et vas trobant problemes i inconvenients que pot ser no es previs a la planificació. De totes maneres és molt necessari fer una bona planificació i intentar arribar a les dates compromeses perquè desprès la feina s'acumula.

A part d'això, el àrea del projecte ha sigut molt interessant per mi, perquè m'ha permès familiaritzar-me més amb el llenguatge Java fent coses que no havia fet mai, com per exemple treballar amb les finestres de java fent servir la classe swing.

En el apartat de les xarxes, he pogut veure en directe i amb el meu propi detector de paquets, tot allò que passa per la xarxa i que he estudiat en les assignatures de xarxes però d'una manera més teòrica. He vist que ara puc estudiar el funcionament de cada protocol per poder aprendre més sobre el seu funcionament i les moltes millores i possibles coses que podria arribar a fer si continues desenvolupant aquesta aplicació.

En definitiva, penso que ha sigut molt beneficiós per mi aquest projecte, encara que han sigut moltes hores dedicades, crec que ha valgut la pena, he après moltes coses sobre la programació amb Java i sobre els protocols de les xarxes.

6 GLOSSARI

Broadcast: Mode de transmissió d'informació on un node emissor envia informació a una multitud de nodes receptors de manera simultània.
Checksum: Forma de control de redundància, una mesura molt simple per a protegir la integritat de les dades, verificant que no hagin sigut corromputs.
Classe: Referent a programació d'objectes, és una col·lecció de dades que permet operar amb elles.
Encaminadors: Dispositiu per interconnectar les xarxes d'ordinadors que opera a la capa 3 (Nivell de xarxa).
Forats de seguretat: Referent a les xarxes, errada en la xarxa que permet a una persona no autoritzada a controlar o accedir a recursos de la xarxa.
Java: Llenguatge de programació orientat a objectes.
LAN: Interconnexió entre varis ordinadors o perifèrics.
Monitoritzar: Controlar alguna cosa de manera centralitzada.
MTU: (Maximum Transfer Uit) Expressa la grandària de la unitat de dades més gran que pot ser enviada fent servir el protocol IP.
Sniffer: Aplicació que captura trames de xarxa.
TCP/IP: Conjunt de protocols de xarxa en el què es basa Internet i que permet la transmissió de dades entre xarxes de computadors.
Thead: Fil d'execució, propietat que tenen el sistemes operatius per executar varis processos al mateix temps.

7 BIOGRAFIA

Pàgines web consultades:

http://www.iesgrancapitan.org/blog04/?p=92

Zabbix

http://vtroger.blogspot.com/2006/09/herramienta-para-monitorizacin-de.html

http://www.zabbix.com/

NTOP

http://tuxroot.wordpress.com/2008/01/15/ntop-otra-herramienta-para-monitorizar-la-red/

MRTG

http://oss.oetiker.ch/mrtg/

http://www.menalto.com/gallery/miscellaneous/screenshots/mrtg-pronto.png.html

Nagios

http://wiki.contribs.org/Nagios

http://www.nagios.org/

Snort

http://homepage.mac.com/duling/halfdozen/Snort-Howto.html

JPCAP

http://netresearch.ics.uci.edu/kfujii/jpcap/doc/

Winpcap

http://www.winpcap.org/

Ethereal

hipserver.mct.phantomworks.org/ethereal.html

WireShark

http://www.wireshark.org/

http://es.wikipedia.org/wiki/Wireshark

http://4sysops.com/archives/smartsniff-a-smarter-packet-sniffer-than-etherealwireshark/

password safe

http://passwordsafe.sourceforge.net/

Netbeans

http://www.netbeans.org/

Eclipse

http://www.eclipse.org/

Clase Swing de Java (Per programa finestres)

http://www.programacion.com/java/tutorial/swing/

Protocol IP

http://es.wikipedia.org/wiki/Protocolo_IP

http://personales.upv.es/rmartin/TcpIp/cap02s03.html

Protocol ARP

http://es.wikipedia.org/wiki/Address_Resolution_Protocol

http://neo.lcc.uma.es/evirtual/cdd/tutorial/red/arp.html

http://en.wikipedia.org/wiki/Address_Resolution_Protocol

Protocol UDP

http://es.wikipedia.org/wiki/UDP

Protocol ICMP

http://neo.lcc.uma.es/evirtual/cdd/tutorial/red/icmp.html

http://es.wikipedia.org/wiki/ICMP

JFREECHART

http://www.jfree.org/jfreechart/

http://snipplr.com/view/1581/java--jfreechart-example/

Llistat Ports

http://www.vermiip.es/puertos/

SSDP

http://es.wikipedia.org/wiki/SSDP