E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Auditoria de red basada en captura pasiva de tráfico

Grado en Ingeniería
en Tecnologías de Telecomunicación

Trabajo Fin de Grado

Jesús María Saso Fernández

Tutor: Eduardo Magaña Lizarrondo

Pamplona, 25 de junio de 2014

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Índice

Summary 4
KeyWords 4
1. Introduction 5
2. State of the art 6
2.1. Network Miner 6
2.2. Data echo 10
2.3. ChaosReader 11
2.4. Argus 12
2.5. Xplico 15
2.6. Justniffer 20
2.7. CapAnalysis 21
2.8. Ntopng 25
2.9. NetworkTimeout 32
2.10. Aol/Moloch 38
2.11. Summary of applications 40
3. Useful tools for traffic analysis 42
3.1. CapInfo 42
3.2. Editcap 42
3.3. Awk 42
3.4. Sort 43
3.5. MacConversations 43
3.6. Mapvalues 44
3.7. IpConversations 44
3.8. ProcesaConexiones 45
3.9. InfoDups 46
3.10. Analisis_dns.py 47
3.11. ProcesaTrazasDNS 47
3.12. IdentificaRedes 48
3.13. TCPTrace 50
3.14. Xplot 52
3.15. Xming 53
3.16. Script-informes 53
3.17. Wireshark 53

3.17.1. Creación del perfil de configuración 53
3.17.2. Colorear Paquetes 54
3.17.3. Creación de columnas 56
3.17.4. Creación de filtros predefinidos 57
3.17.5. Exportación e importación de perfiles 58
4. Procedimientos generales de análisis de tráfico 59
4.1. Análisis general 59
4.2. Problemas habituales en la tecnología de enlace 59
4.3. Problemática de IP 61
4.3.1. IP Spoofing e IP Duplicada 62
4.4. Problemas habituales en TCP 63
4.4.1. Problemática web 65
4.4.2. Problemática Citrix 66
5. Mejora en la visualización de TCPTrace 67
5.1. Verificación 67
5.2. Obtención de gráficas 68
5.3. Conversión xpl a gpl y edición 68
5.4. Dibujado y función de zoom 69
5.5. Ventajas 69
6. Conclusions 71
Bibliografía 72
Anexos 73
Anexo 1 74
Anexo 2 76
Anexo 3 78
Anexo 4 80
Anexo 5 82
Anexo 6 84
Anexo 7 86
Anexo 8 88
Anexo 9 90
Anexo 10 92

Summary

This project aims to achieve various objectives. The first of them is to guide the user into finding and solving problems that the network administrator might have encountered in the network and also into analyzing a trace. This project will teach the reader how to use available state of the art network monitoring tools. And finally, it will show a tool to review the TCP windows sequences of one or several TCP connections.

In the case of trace analysis, this will be illustrated with some of the typical cases that can happen as, for example: slow web browsing, IP spoofing, duplicated IP packets, problems with the Citrix service, etc. This guide explains the steps that you should check in order to detect what the problem is, using public/private currently available tools.

In relation to the design of a graph of the TCP connection, we will show the available tools and we will see that they do not have the required functionalities. We will therefore develop a new tool, step by step in order to obtain a basic tool which will give us the result we need.

KeyWords

Traffic, analysis, TCPTrace, tcpdump, wireshark, network monitoring, duplicated packets, Xplico, trace, http, problems, flows and pcap.

1. Introduction

Nowadays, several business and domestic user have several problems of connectivity when they try to browse the internet, receive/send emails, watch videos on YouTube, use services of Citrix virtualization, etc. In addition, there are a lot of tools that try helping to find what problems there are on the network. A basic disadvantage of this tools is that protocols, services and applications are being updating every day and the major part of them are obsolete.

Then, they need a person with knowledge in this area in order to find and solve the problems of the network. Depending on the size of network and functions that are used, the manager is likely not to know all about protocols and it can take long time to find the problems.

This project aims to achieve various objectives. The first is to show, test and review tools of the state of art that are available. They are reviewed about performing, functions, deep of analysis, graphs of the traffic trace, etc. In addition, to complete the analysis, we are going to analyze the basic tools that allow to do exhaustive analysis of specifics parameters of the traffic trace.

The second objective of this project aims to be a step by step guide about how to analyze the traffic trace in order to discard the task that are not problematic and then finding the cause of bad working point of the network.

And finally, we will deduce that there are not tools with the functions that we need, so we develop a tool (exactly a script for Linux) in order to improve the existing tool TCPTrace to make graphs in PNG format with legends where they show the different series that are drawn.

In summary, this project allows to learn several guidelines about the available tools and how they should be used to find the problems that can be present in a network. And in the case of tools that have not some functionalities, some clues about how they should be developed are presented.

2. State of the art

In this chapter, we are going to provide a comparison of the main free/trial tools that we could use to analyze easily traffic traces using visual tools. The requisite is that the tools can read from a pcap file and provide automatically the analysis information. In this comparison, the testing is performed with a trace that have a size of 420MB in order to check which information data is obtained and the time used to complete the analysis. This trace contains the typical traffic of a business, as for example: HTTP, SSL, FTP, STP, RTMP (YouTube), DHCP, etc. More information are in the Anexo 6. This trace was uploaded to the Naudit Portal [26] by an enterprise. The more well-known program for analyzing traffic traces is Wireshark (x64) that took 120 seconds to analyze the trace on an Intel dual core host of 2.66GHz and it needed 470MB of RAM memory. Wireshark did not need any available free space on hard disk unless you want to export some packet data.

To search for related tools, the key words that have been used have been “analyze”, “traces”, “pcap”, [14]. The following tools will be analyzed:

- Network Miner
- Data Echo
- ChaosReader
- Argus
- Xplico
- Justniffer
- CapAnalysis
- Ntopng
- NetworkTimeout

2.1. Network Miner

This tool, Network Miner [15], is a basic tool to perform traffic traces analysis from capture files or live through network devices (it needs the pcap library). This tool was developed to be used on Windows system by means of graphical mode or line command mode. Also, it is possible to use it in other operative systems using emulators. It can understand the following protocols: HTTP, SMB, FTP, TFTP, SSL, TLS, TOR, etc. One advantage of this tool is that it can identify for each host in the network what operating system has, the number of connections keep opened, etc. On other tab, there are related information of traffic that there are in the trace, for example: credentials, pictures, files that have been transferred, messages, parameters exchanged in the get/post requests, etc.

On the performance aspect, it is slow (150-180 seconds), and it does not need to be installed (excepting for the pcap library required for analysis in real time). The slowness is due to export the content of trace to real files in the hard disk drive. In the experiment case, it needed 170MB of hard disk. The RAM memory stores all the information that it exports and all statistics, so the tool needed 800MB of RAM.

Main output analysis data:

- Hosts (a screenshot is shown in Figure 1)
  - IP
  - MAC
  - Hostname
  - Operative System
  - TTL
  - TCP server
  - In/out coming connections
  - In/out coming sessions
- Frames (a screenshot is shown in Figure 2)
  - Source/destination MAC
  - Source/destination IP address, TTL, total length
  - Source/destination TCP port, sequence number and Flags
- Files
  - Possibility of viewing the exchanged files over http/ftp/smtp
- Images
  - History of exchanged images through html
  - Thumbnail/real images
- Messages
  - Instant messages that use certain protocols
- Credentials
  - Cookies, form fields that could be credentials
- Sessions
  - The different established connections to server
- DNS
  - Resolutions made with request and reply
  - Date of request
  - Duration between request and reply
- Parameters
  - Exchanged data through GET/POST http
- Anomalies (a screenshot is shown in Figure 3)
  - Detected faults in the trace for example duplicated packets, incomplete files, etc.
- Read from pcap file or network adapter
- Export content of connections to files in order to view the interchanged data

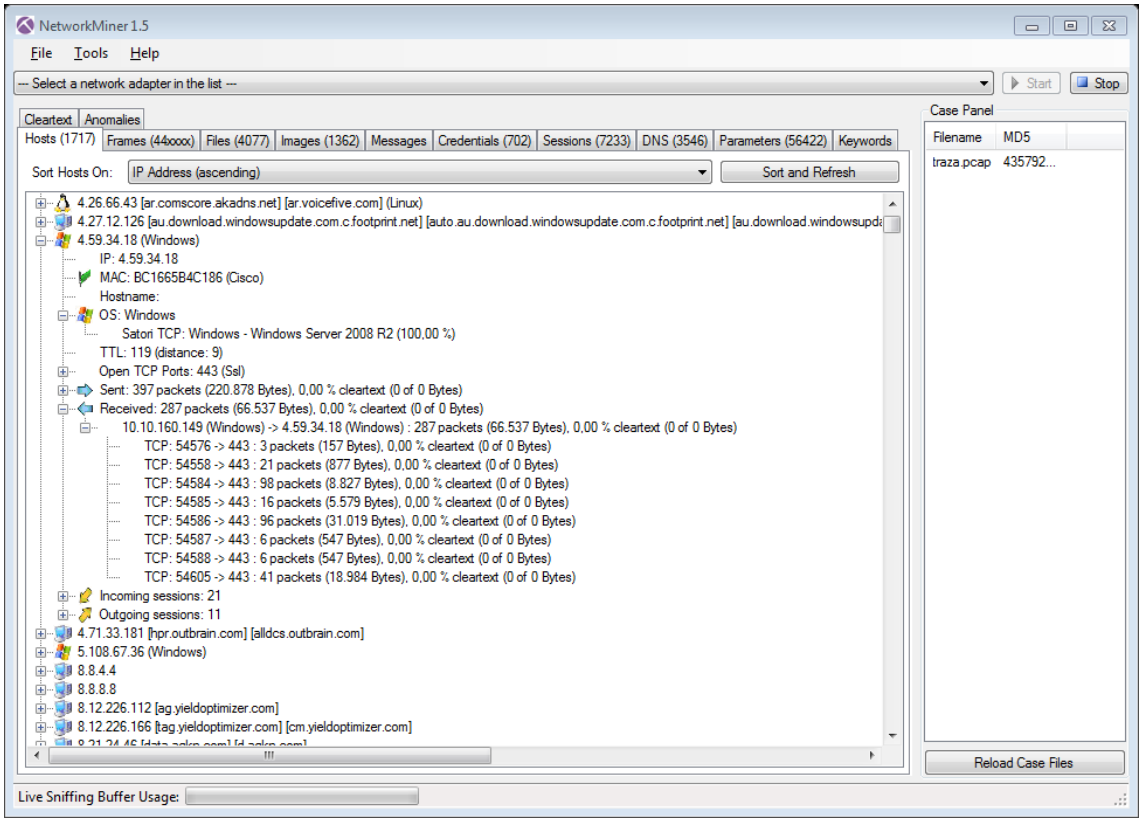

Figure 1: Network Miner, hosts

In the Figure 1, there is information for each host, as for example: IP address, the MAC and name of the host, the operating system that it is running, etc. On the side of TCP, it shows statistics about the number of opened connections, the number of packets and data that it has exchanged. In addition, the connections of each server are showed with statistics indicating the port number of source and destination.

On the next Figure 2, it shows the tab of frames. The objective of this tab is to show the information of all layer of the packets (link, network, transport). This is useful to find problems about certain fields of header as for example: a bad established TCP connection, high decrement of TTL, etc.

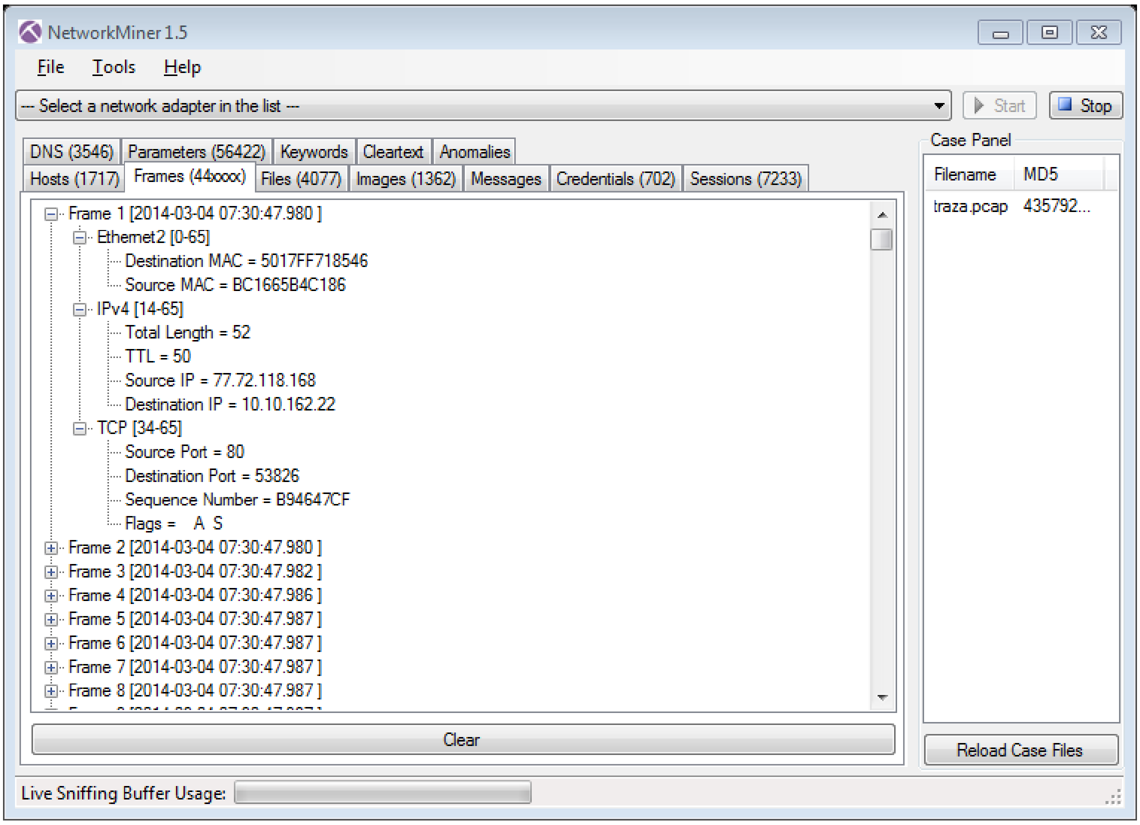

Figure 2: Network Mine, Frames

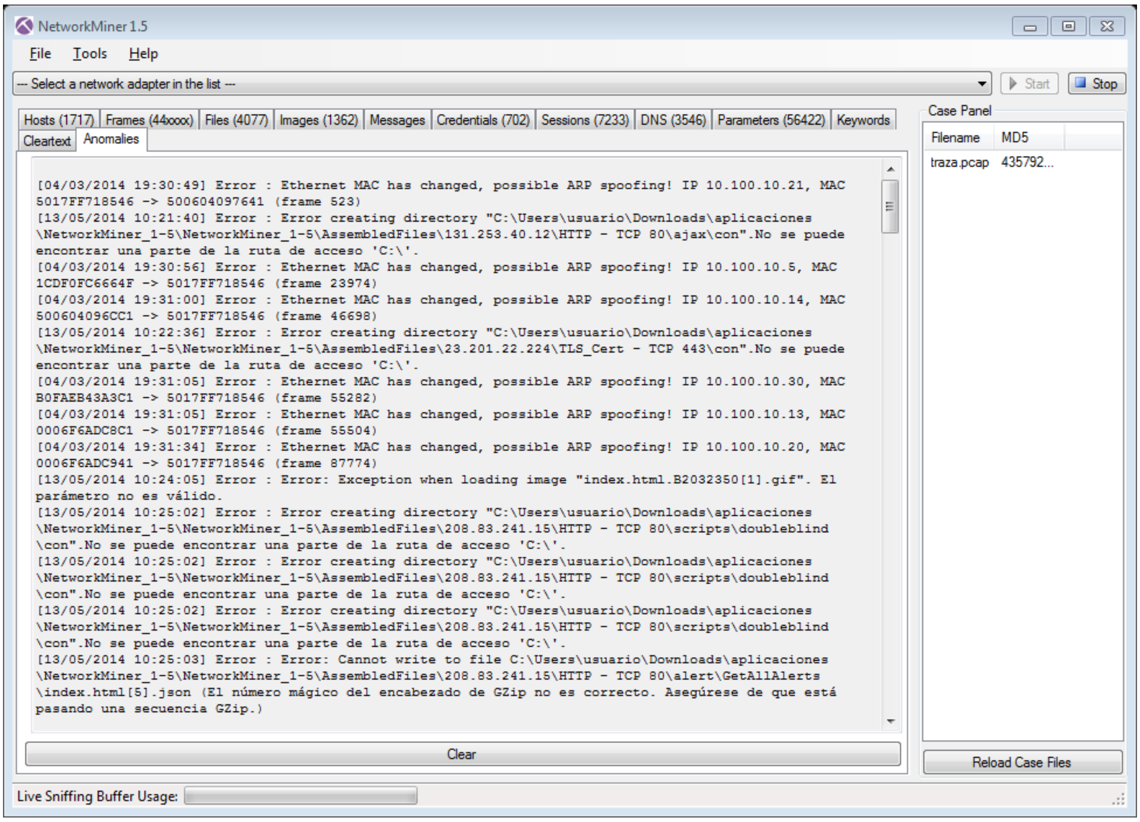

Figure 3: Network Miner, deficiencies

A useful function of this tool is to show the problems that the tool has had reading the traffic packets from the file or network adapter. This tab is called as “anomalies” that we can see in the Figure 3. It notifies problems as for example ARP poisoning because of exceeding the threshold of gratuitous ARP packets.

2.2. Data echo

This next tool, Data echo [16], needs to be installed in the computer with pcap library in order to be able to perform on live analysis.

This tool only exports the connection content to html files. It groups all connections by hosts. As it is so simple, it processes very fast the content (faster than wireshark, it needed 60 seconds), with a low consumption of RAM (100MB) because the connections content is saved in hard disk. As disadvantage, it shows some information about connection from the traffic trace or real time capture.

Main output analysis data:

- Client IP address
- Server IP address
- Timestamp of the connection
- Used ports
- Name of hosts
- Reply (connection level)
- Viewfinder of web content (if the content had a website)
- Export connections to files (<100MB)

In the Figure 4, we can see 3 different parts that are “capture host”, “capture web data” and “capture text data”. In the first (the left vertical side), it is a list of all hosts that are found on the traffic trace. It is a dropdown tree sorting the information by server, date and the typical information of the TCP connection. In the “web data” it shows the information as a web browser understands. In this case, if the request is a web page, we will see the webpage in this section. And finally, the other section is the translated data to ASCII in order to be interpreted the protocol line by line.

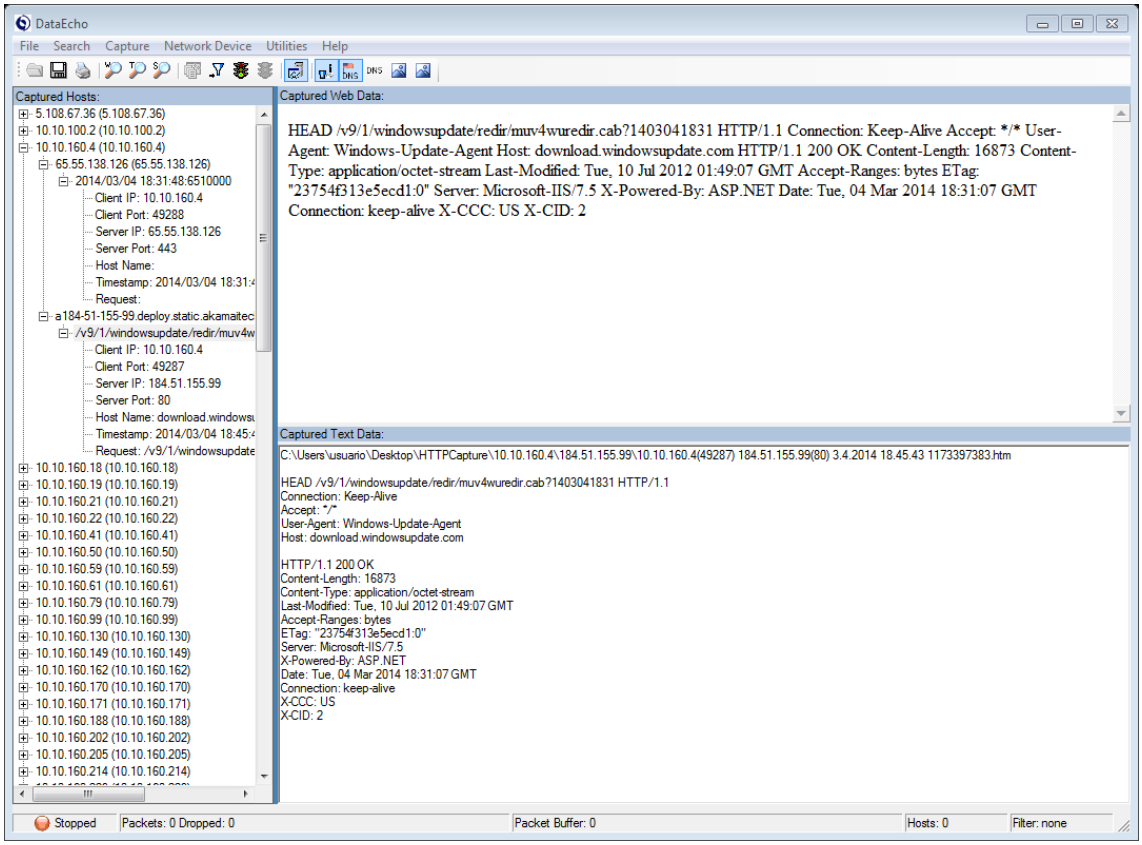

Figure 4: Data Echo

2.3. ChaosReader

This ChaosReader [17] is an analyzer that creates reviews in html format where you can view the content of connections, indicating the raw data or parsing to correct format if it is some known format as JPG, HTML, SNMP, etc. It will resolve the name protocols if this name has been implemented on the application. It processes the traffic trace very fast (it is a bash script in Linux) and it finished 10 seconds later from starting. The information is saved in the hard disk drive (it needed 500MB available space) and it needed less than 100MB of RAM. The GUI is very simple, but it is effective.

Main output analysis data:

- Viewfinder of images and html content
- Details about date, IP address and port, method of the request GET/POST with exchanged parameters
- Summary of flows, size of exchanged data and connection content exported to files if it is possible (for example http, images, etc.)
- Possibility of view exchanged information for each connection.
- Log of each http request with date, size, file request, error code, method request and URL
- It can read from pcap file or network adapter in real time.

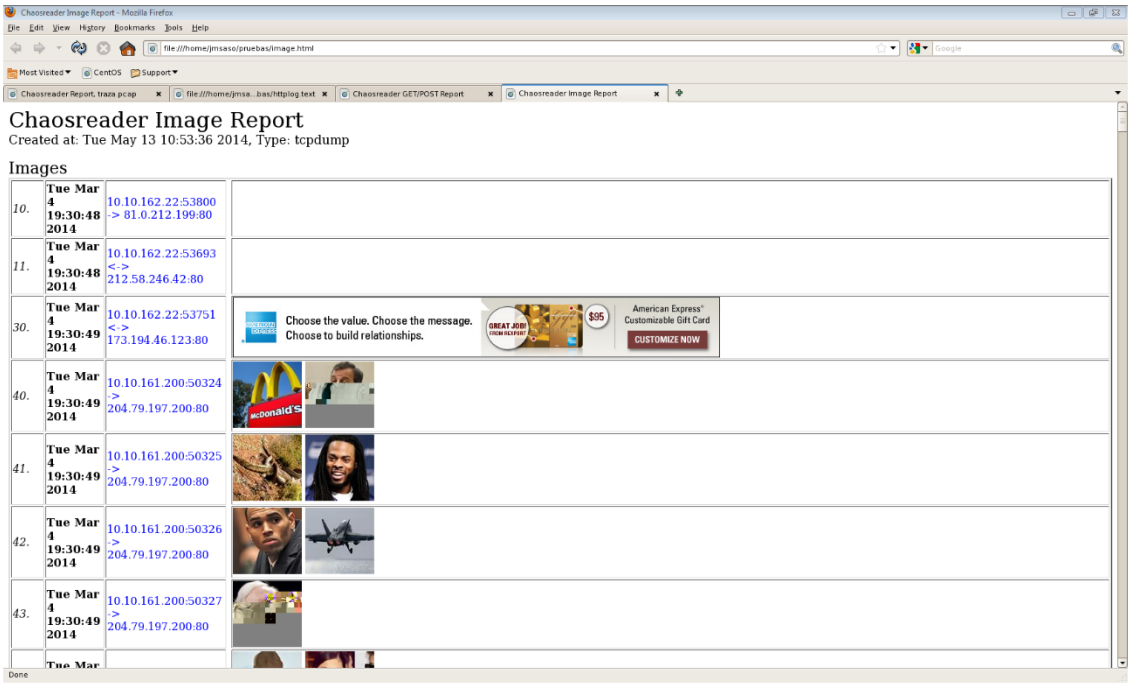

Figure 5: Example of one kind of statistical reviews of ChaosReader

The Figure 5 shows one of the 4 reviews that ChaosReader create when analyze the traffic trace. In this screenshot, we can view the date when the connections were established and in the cases that it transferred images are showed.

2.4. Argus

Argus [18] is a system for the analysis of traffic data at flow level. The processing load is very low (it ended in less than 10 seconds) because it is modulated in order to be efficient in processing. The basic results are returned in data files without a graphic user interface:

It is possible to do a deeper analysis of the traffic trace with graphical tools in order to view, for example, the data speed, RTT, more used ports, etc. It is possible to do it easily with examples that are documented in the website of the tool or the user manual. To draw with those tools, you need to execute the tools of the client version. There are two kind of tools packages, the client and server version. The server version allows to get the key information of the trace. After that, the client allows to do graphs (for example: histogram, bars or others kind of graphs that are available in the website), filtered information by flows, etc.

Main output analysis data:

- Output data is saved in a private file format
- Statistics are got through auxiliary tools
- It is possible to do a lot statistics
    - It can create graphs of connection between hosts with information of IP address, size, duration, etc.
    - It can group traffic by:
        - Protocol
        - Source IP address
        - Destination IP Address

- Source port
- Destination port
- Source TOS
- Destination TOS
- Source TTL
- Destination TTL
- Full geo-positioning (country, city, street, etc. If it is possible)
- Etc.

- It exports the content to text, or shows it using graphs
- Graphs
  - Bytes
  - Packets
  - Average speed
  - Duration
  - Losses
  - Jitter
  - Load
  - Export in PNG format
  - You can set titles, size, etc.

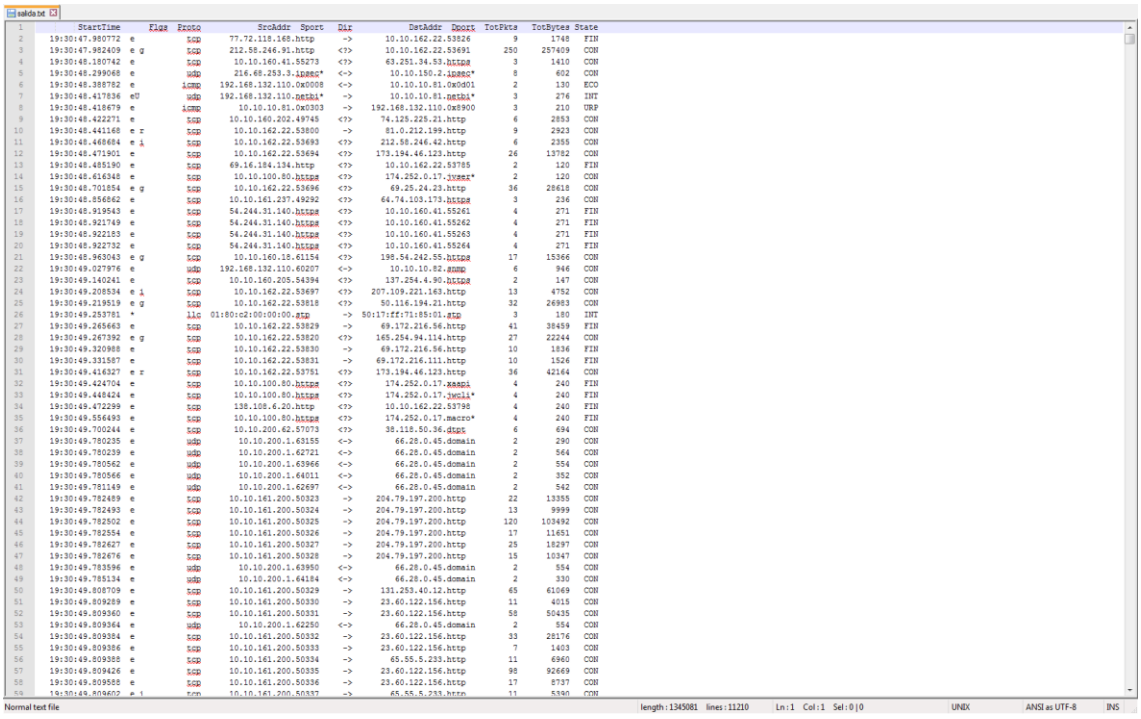

Figure 6: Basic example of use of Argus application

The example that we can view in the Figure 6 is the offered information by Argus in the basic analysis. It is classified in columns as for example: the timestamp of connections, all flags of the connection, the source and destination of IP addresses and ports, etc.

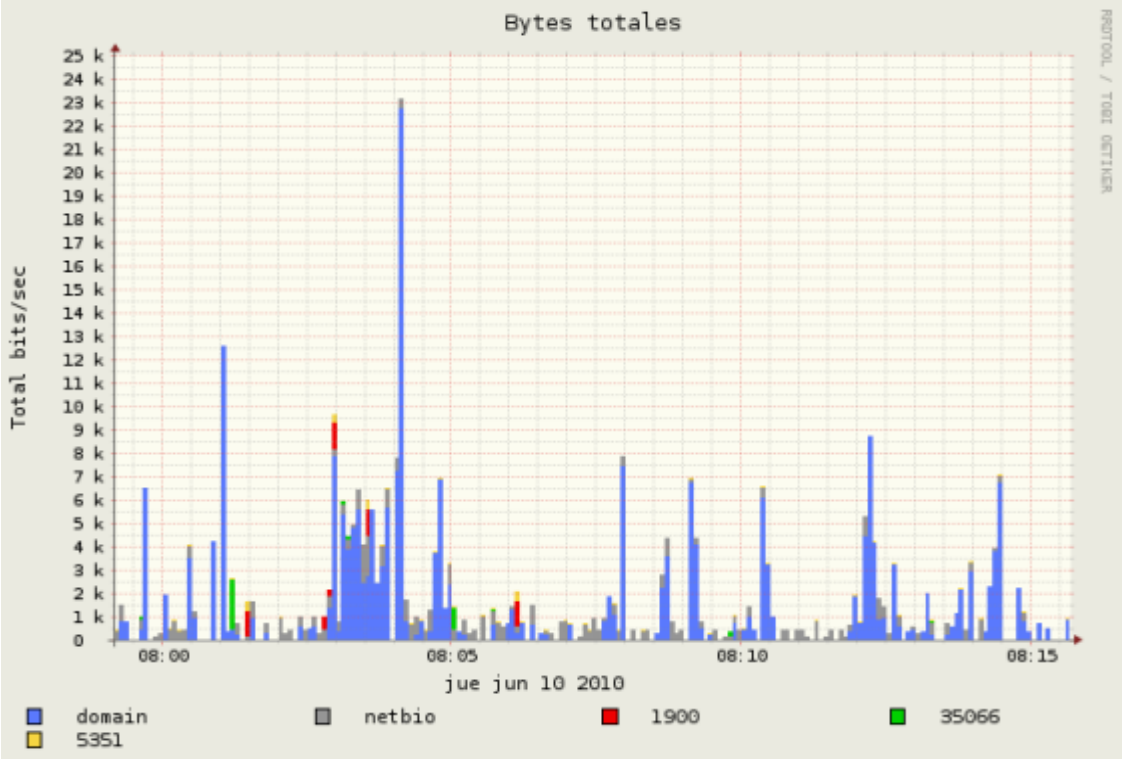

Figure 7: Graph that was created by Argus. It shows the different UDP services [6]

In Figure 7, we can see the average bitrate of the data trace by different services over UDP. To draw these graphs, the tool needs the RRDTool [11] library between others.

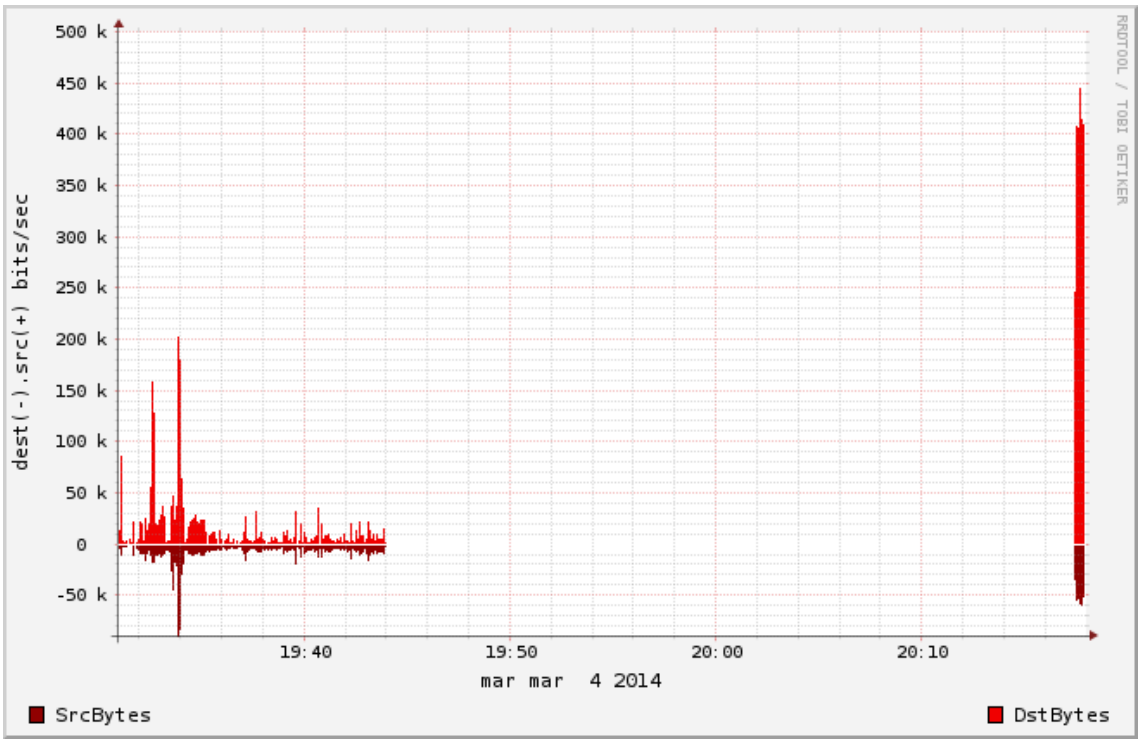

Figure 8: Argus, this is an example of data rate

In the Figure 8, we can view a graph of the inbound and outbound traffic. The throughput is measured in bits/sec and the bottom of graph, there is a legend of draw series.

2.5. Xplico

This web application Xplico [19] supports showing general statistics of connections for a specific host. It permits analyzing a pcap file (the limit of size of file is defined in the settings of apache server or the server that you use) or doing network capture in real time. In the case of our experiment, it needed between 3 and 4 minutes to analyze the trace. It has a lot of different statistics that you can get.

Figure 9 presents the summary of the traffic packets. This information is about all protocols that Xplico supports. To show this, it organizes in boxes that contains the basic summary for each item of the main classification of protocols.

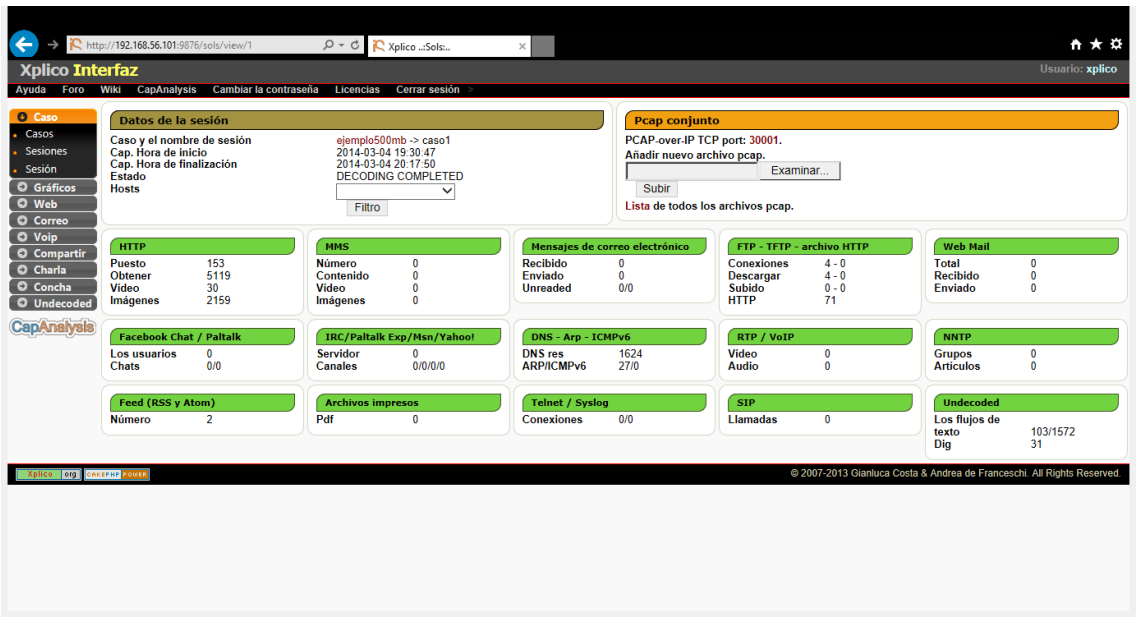

Figure 9: Xplico and the summary GUI

Main output analysis data:

- Graphs
  - DNS
    - Host Popularity (a screenshot is shown in the Figure 11)
    - Reply time by hours (a screenshot is shown in the Figure 10)
  - ARP
    - Information of who IP address ask for who MAC
  - ICMP
    - Support of IPv6 and IPv4 address
    - Source
    - Destination
    - Time between request and reply
  - GeoMap
    - Drawing of flows over Google Earth maps, in kml format
- Web
  - Site (a screenshot is shown in the Figure 12)
    - Visited Webs
    - Web size

        - Method of request
        - You could download the pcap file of connections that you want
        - You could view the content if it is possible (websites, images, videos, flash, audio, Json, etc.)
    - You could read the Json files
    - It shows exchanged images in full, thumbnail format and her context.
- E-mail
    - Analyze of email protocol: smtp , imap, pop3
        - Date
        - From
        - To
        - Message
        - Subject
        - Size
        - Priority
    - WebMail Analyze
        - Date
        - From
        - Subject
        - To
        - Size
        - Provider
- VoIP
    - Support of Sorbo and RTP protocol (Sorbo is a type of protocol and network equipment of VoIP owner)
        - Date
        - From
        - To
        - Duration
        - Possibility of listening/watching the call/video
- Sharing
    - HTTP File
        - Files transfers (any extension that were transferred via http)
    - FTP (a screenshot is shown in the Figure 13)
        - Connection lists
        - Used account (user and password)
        - Action that the user did
        - File that the user transferred
    - TFTP
    - Network printers
        - IP address of printer and the content that it printed
    - MMS
        - Date
        - From
        - To
        - Content
- Talkers
    - NNTP

        - Messages
    - Facebook
        - Messages of Facebook chat
    - Messenger MSN
        - Date of Started and ended
        - Messages
        - Duration
    - Yahoo! Messenger
    - IRCs
        - Date
        - URL
        - Channels
    - Paltalk
        - Date
        - Room name
        - Duration
    - Exp Paltalk
        - Identification of user nick
- Concha
    - Telnet
        - Date
        - User
        - Host
    - Syslog
        - Date
        - Host
        - Size
- Not decoded connections
    - TCP-UDP Flows (a screenshot is shown in the Figure 14)
        - Date
        - To
        - Ports
        - Protocols
        - Duration
        - Size
        - Content
    - Dig
        - Date
        - Record (name of file)
        - File Extension or the kind of file
        - Size

In Figure 10 we can view a time graph of the average DNS response. This is very useful to detect problems of server saturation. On the other hand, we can know what server is the most demanded with other kind of graph that is showed in the Figure 11. The name of server is resolved so we can view their names and the number of requests.

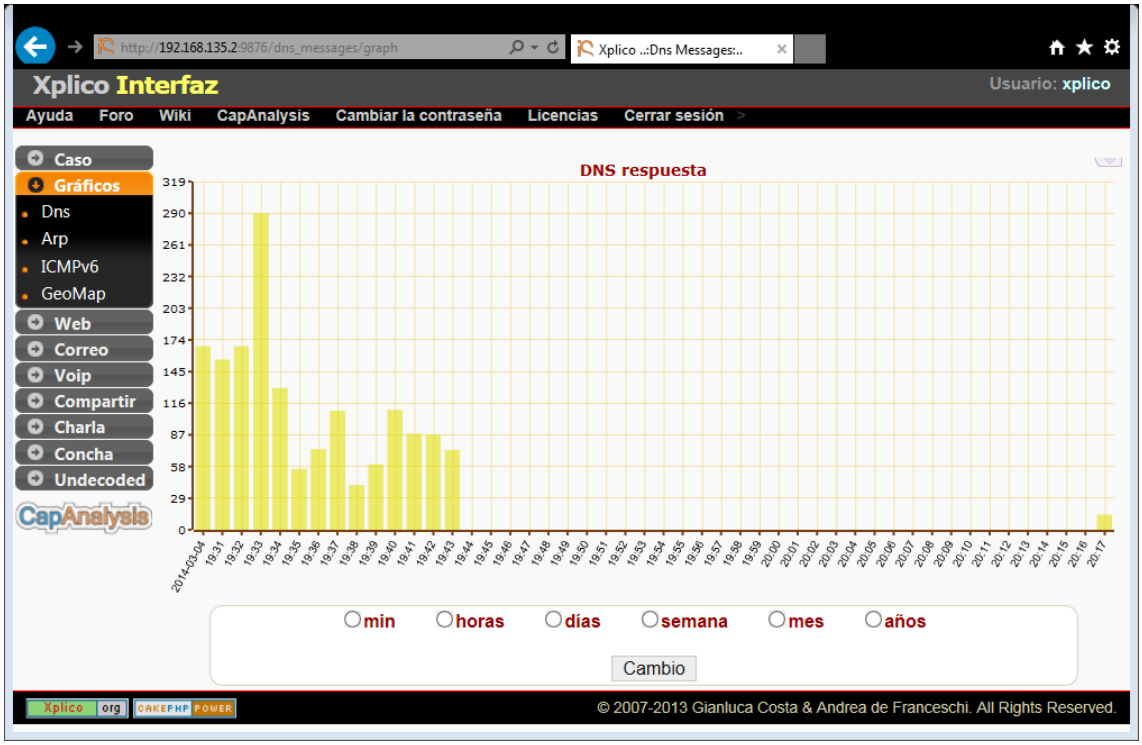

Figure 10: Xplico, time graphic of DNS response

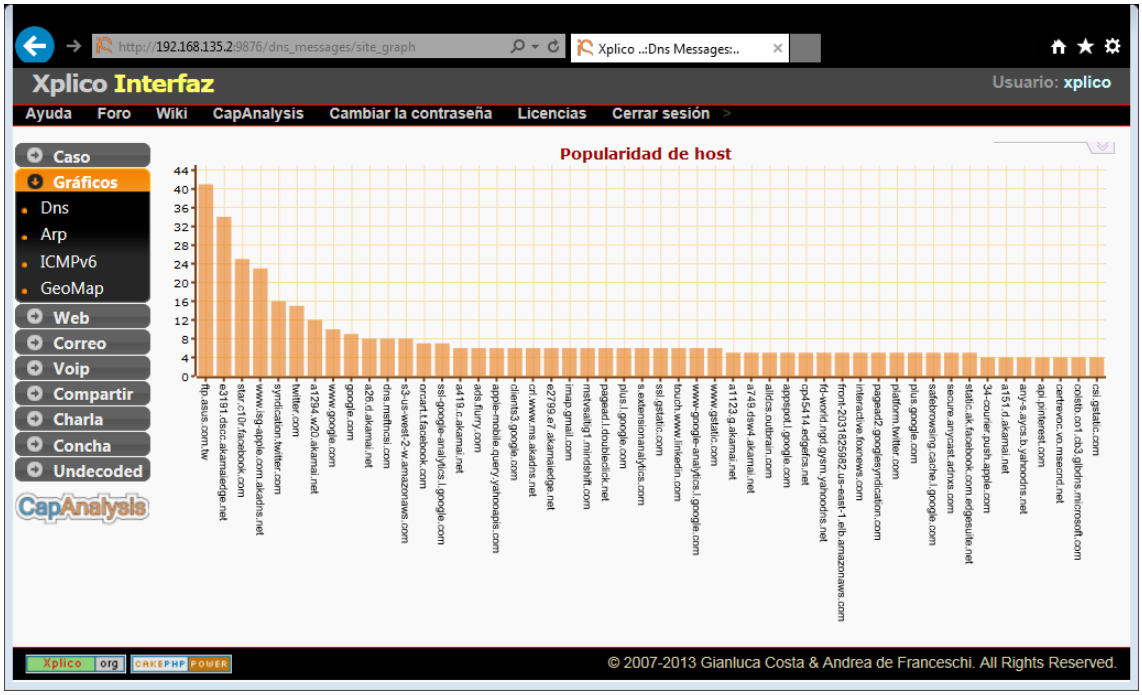

Figure 11: Xplico, amount of host resolutions

In the Figure 12, we can view the sites that are in the traffic trace, their size and the method used to realize the request. If we need more information, we can get though the xml file of the connection. In this case, it is shown the html view, but there are more options of views as for example: images, videos, flash, audio and Json files. In the next figure (Figure 13), we can view the transferred made over FTP. In addition, there are information of who made the upload or download and what file was transferred.

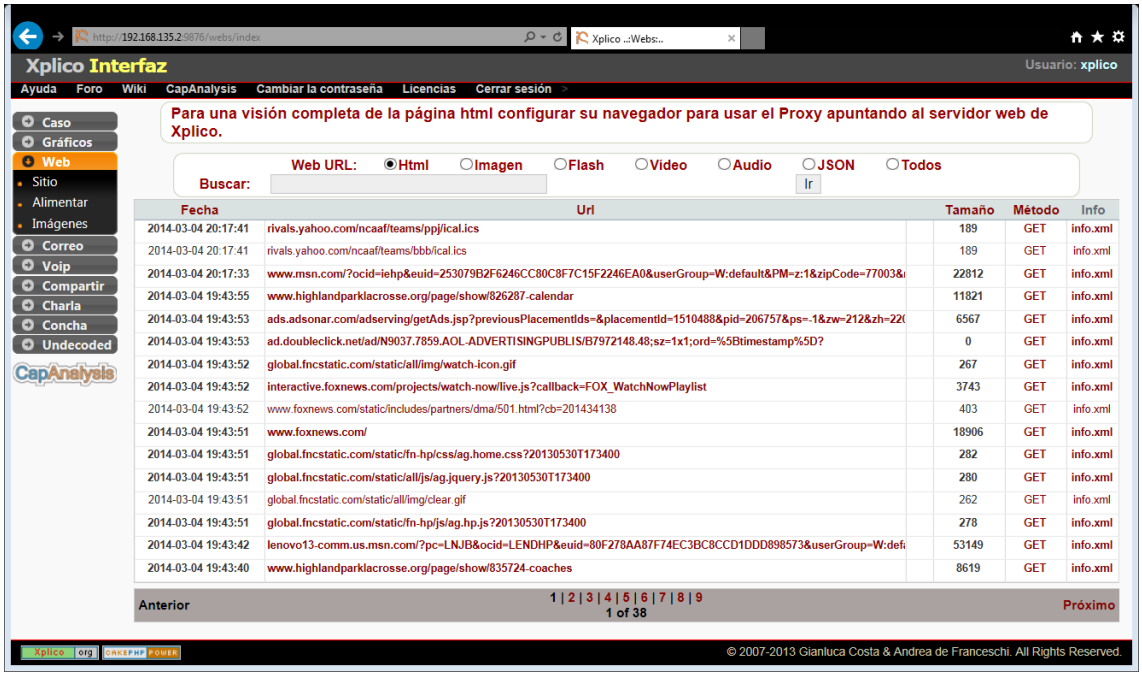

Figure 12: Xplico, web section

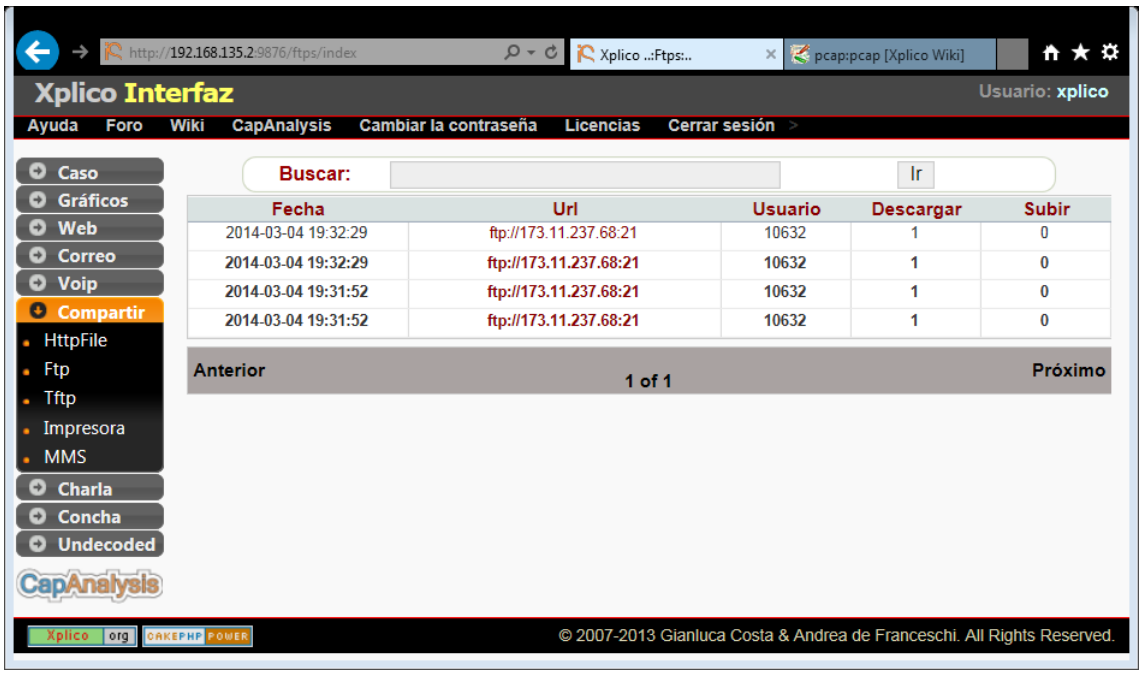

Figure 13: Xplico and the FTP service

Finally, the last screenshot of Xplico (Figure 14) where we can view how it classifies the connections of other protocols that it does not know. It shows essential information that we can use to deduce what protocol is (in the case that Xplico can not recognize).

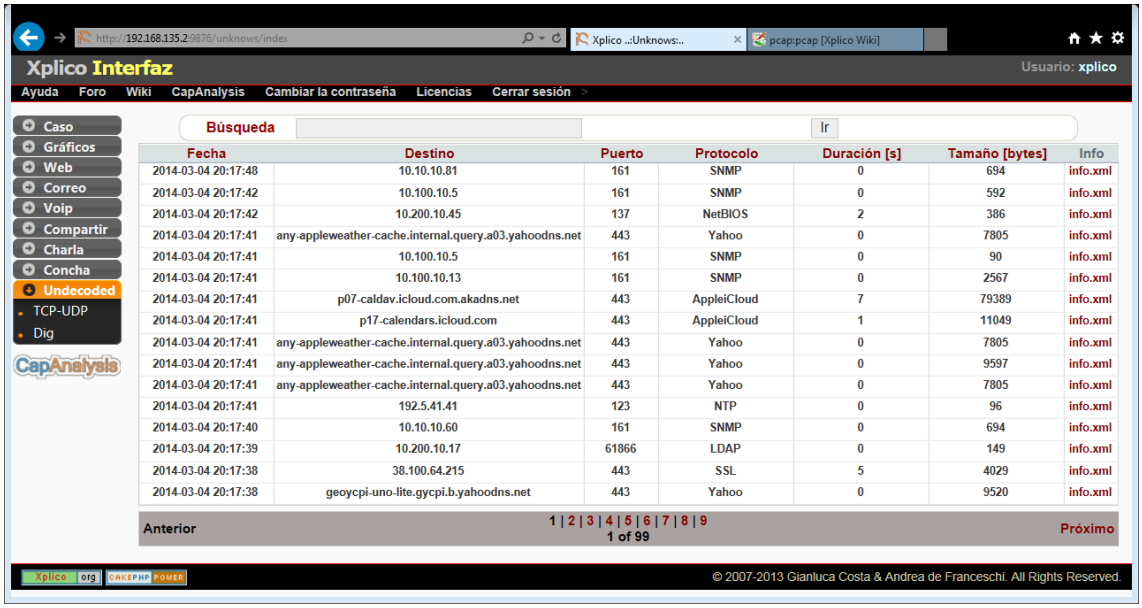

Figure 14: Xplico and generic TCP/UDP connections

2.6. Justniffer

The next tool Justniffer [20] works on command line in Linux. This utility reassembles, sorts and shows the TCP flows. It looks like a tcpdump, showing all information in the Linux terminal. The advantage of this tool is that you can get information for each connection, with header data of HTTP, SMTP, etc. that can be specified in the execution. It supports analyzing from pcap files or the network adapter in real time.

Main output analysis data:

- Output data formatted as apache logs
- It supports showing detailed information of
  - HTTP
  - JDBC
  - RTSP
  - SIP
  - SMTP
  - IMAP
  - POP
  - LDAP
- It supports getting the file content in binary format
- Keywords to get values
  - %close.originator
  - %close.time
  - %close.timestamp
  - %close.timestamp2
  - %connection
  - %connection.time
  - %connection.timestamp
  - %connection.timestamp2

- %dest.ip
- %dest.port
- %idle.time.0
- %idle.time.1
- %newline
- %request
- %request.grep
- %request.header

The performance of this tool has not been evaluated because certain libraries could not be installed, for example libboost or the installer could not recognize the Justniffer application.

2.7. CapAnalysis

CapAnalysis [21] allows doing high-level analysis very fast. It supports reading from files or network adapters through a NetCat connection (to transfer a big file that apache server does not allow because of the upload settings or other reasons). In other case, you could upload the pcap file through a website of this tool. In this case, the size of file is limited by the apache settings or the server that you use.

The analyzed functions that are supported are varied allowing to do filters based on hosts, times, ports, geo-position of host, etc. The GUI is very interactive. It recognizes several well-known protocols. In addition, it supports recognizing the services of some applications.

The time that it needs to process our trace, is 60 seconds approximately with a low memory impact because it exports the output statistics to files. It needs less than 100MB of available space on RAM memory. So, it needs more space on the hard disk drive.

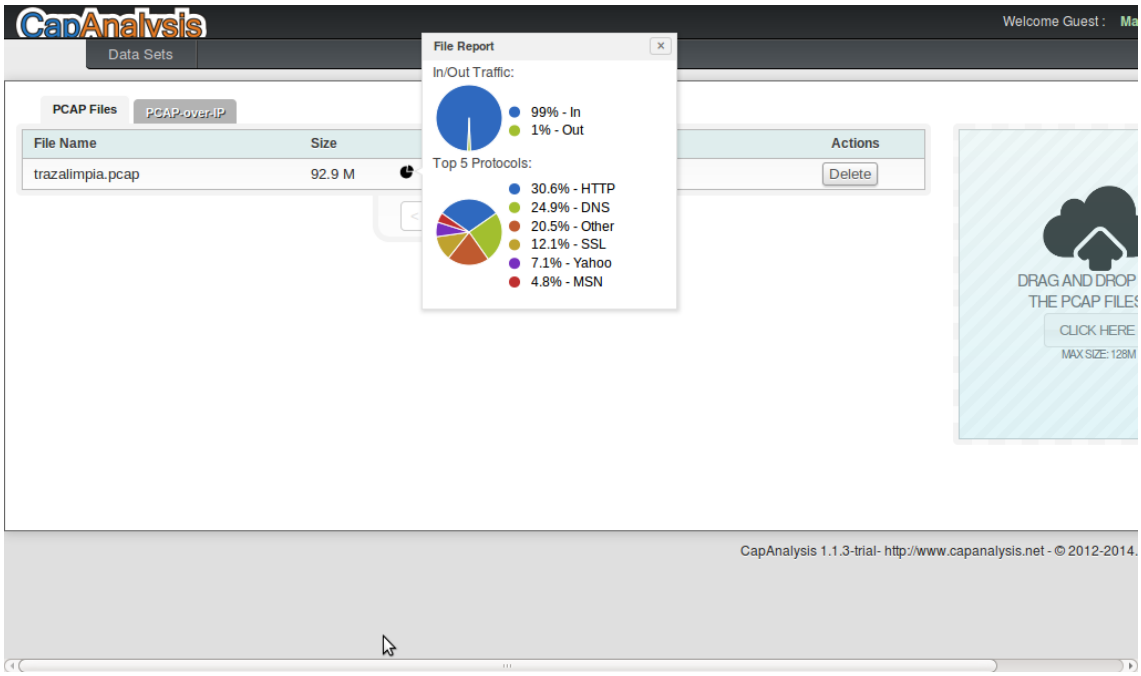

Figure 15: CapAnalysis, analyzed example of files

In the screenshot of the Figure 15, we can view the portal where we can upload the trace file and information about protocols that are present in it and the relation between input and output data.

Main output analysis data:

- General statistics: traffic, ports
- By flows (a screenshot is shown in Figure 16)
    - Wireshark style information
        - Date, time, source and destination IP Address, resolution of destination (DNS), source and destination ports, transport used layer, upper protocol and identified known websites and applications, destination, amount of data in units and percentage, amount of lost data, number of packets, percentage of packets sent and received, duration of the flow
        - You could select what columns you see
        - You could filter by IP Addresses and ports in source and destination
        - Show/hide flows of known services and applications
        - Filter by amount of data sent and/or receive
        - Filter by temporal range of capture
- Overview (a screenshot is shown in Figure 17)
    - Graphs of destination port
        - By flow
        - By data amount
        - By duration
    - Graphs in time
        - Of flow numbers
        - Amount of data sent
        - Amount of data received
        - Amount of all data (received and sent)
        - Duration of connections
    - Graphs protocol vs Country
        - By flows
        - By Data
    - Graphs Protocol vs Date
        - By flows
        - By data
- Statistics (a screenshot is shown in Figure 18)
    - Graph of IP Address vs. flows
    - Graph of destination IP address vs. flows
    - Graph of source IP address vs. total data
    - Graph of protocols vs. flows
    - Graph of countries vs. flows
    - Graph of duration vs. flows
- By hours (a screenshot is shown in Figure 19)
    - Time series Graph with hourly resolution
        - By flow numbers
        - By amount of data

        - By amount of sent data
        - By amount of received data
        - By duration
    - General statistics
        - Duration of capture with temporal identification
        - Number of flows
        - Duration
        - Sent bytes
        - Received bytes
    - Host counter
    - Destination graphs
        - By IP address
        - By flows
        - By data
        - By duration
    - Graph of protocols that predominate
        - By flows
        - By data
        - By duration
    - Graph of protocols vs. Countries
        - By flows
        - By data
- GeoMap
    - It supports identifying the position and data direction in the map
    - This does not work in the trial version
- Source IP address
    - Information about
        - IP address
        - Flows
        - Sent bytes
        - Received bytes
        - Percentage
    - Options of personalization
        - Shown/hidden columns
        - It filters by IP Addresses and ports of source and destination
        - It filters by known services
        - It filters by amount data that have sent or received
        - It filters by time slot
- Protocols (a screenshot is shown in Figure 19)
    - Graph of number of source IP Address vs. number of destination IP Address
    - Size of circles in function of number of exchanged data volume for each protocol
    - Detailed Information for each protocol of host numbers that there are in source and destination, total and by source and destination exchanged data
    - Number of flows
- Timeline
    - Graph of connection number that there are for each day
    - Graph of amount of exchanged data for each day
    - Graph of amount of received data for each day

- Graph of amount of sent data for each day

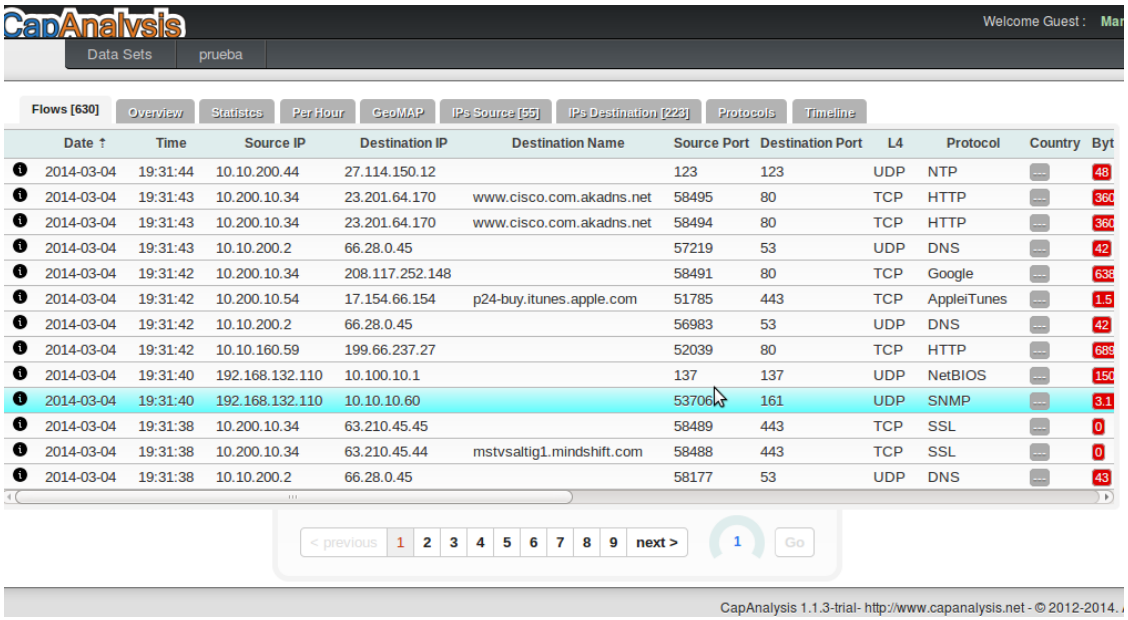

Figure 16: example of flow information on CapAnalysis

This screenshot (Figure 16) shows elemental information for each flow as for example: the IP addresses, ports, hostname of destination, transport layer that it uses, protocol or application of this connections, etc.

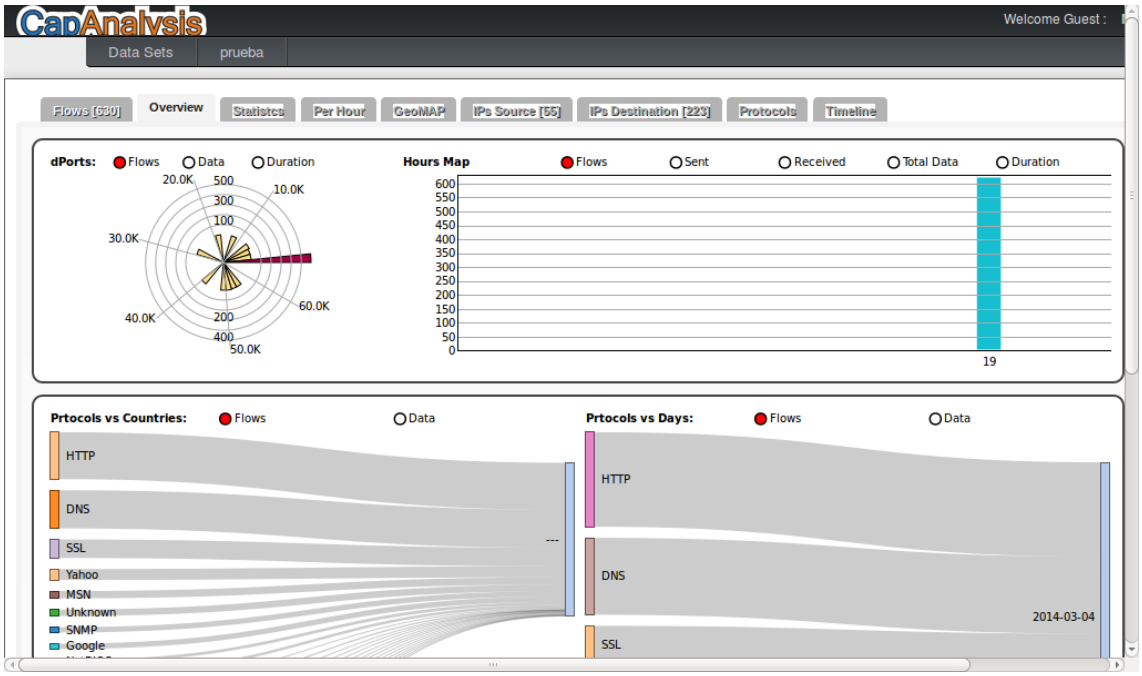

Figure 17: Example of overview graphs on CapAnalysis

The Figure 17 shows graphs of the protocols, countries, number of flows or bytes, duration of connection and other forms of drawing statistics of the traffic trace. If we need more graphs of statistics, we can view other section of the CapAnalysis as the Figure 18 shows.

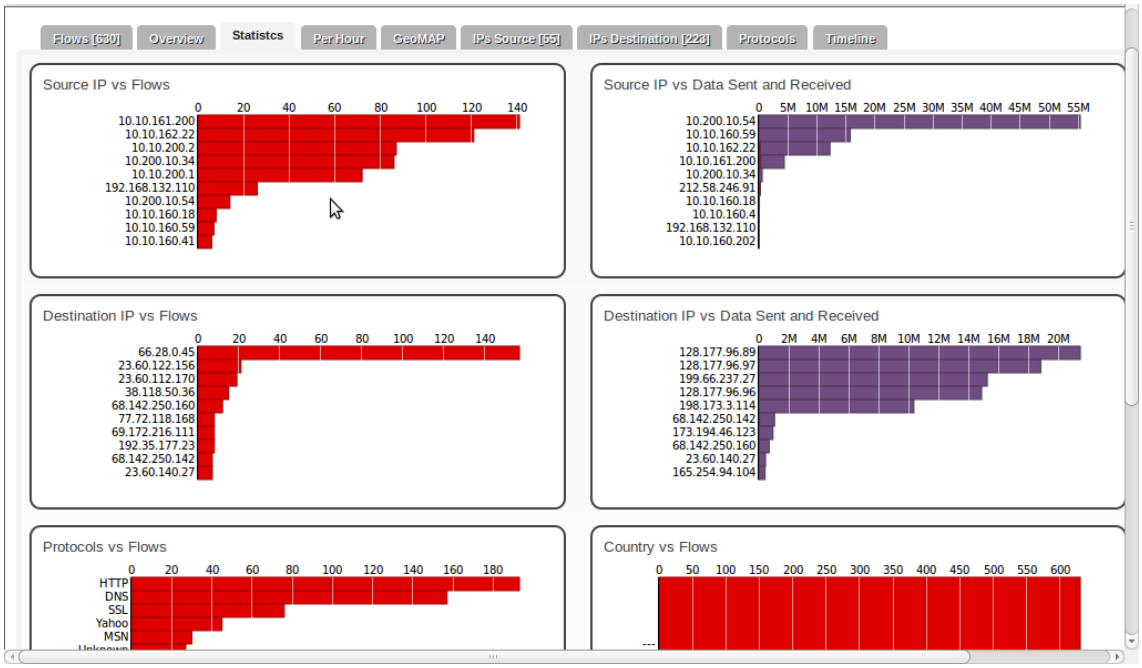

Figure 18: Graphs of statistics on CapAnalysis

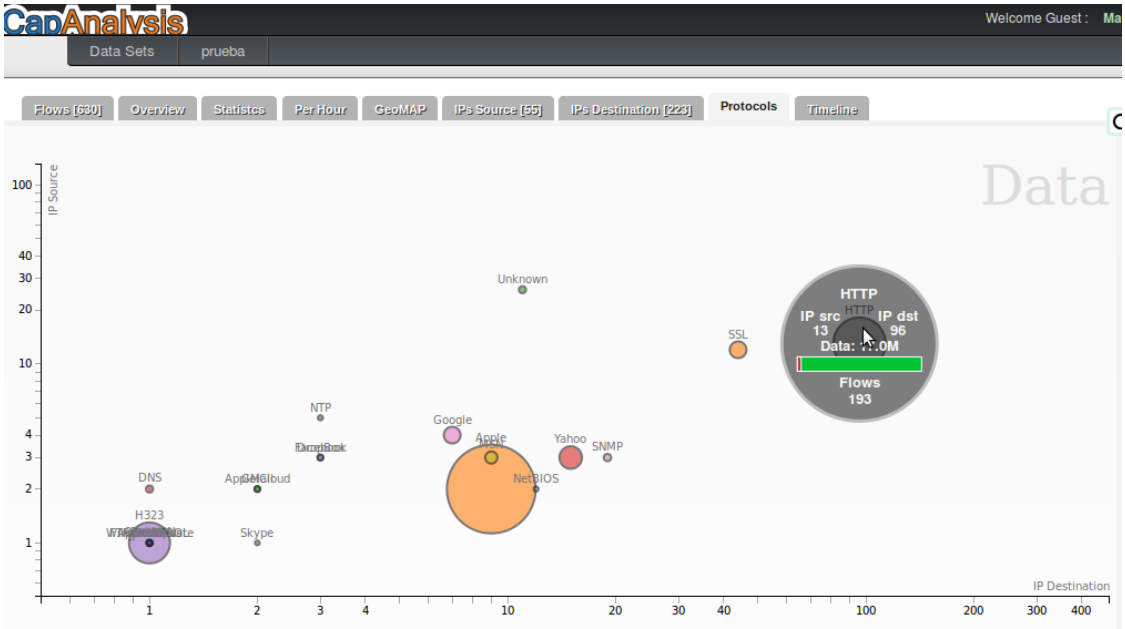

Figure 19: Graph of protocols on CapAnalysis

One form of knowing the relevance of the protocols in the traffic trace is the graph of protocols of CapAnalysis (Figure 19) where it draws circles for each protocol with a size that depends of presence of protocol in comparison with the rest protocols of trace.

2.8. Ntopng

The next tool that we are going to analyze is Ntopng [22] that is a tool of the Ntop collection. This tool has been developed over time for example, the GUI, the functions are supported on independent modules are added in order to improve the analysis.

The performance of the application is very good, it only needs around 10 seconds for analyzing the trace and it needs few RAM memory. The reason is because it processes the trace and saves the generated statistics so Ntopng only needs to load results in order to send them to web browser of the user who is asking for it. The web interface is very nice and dynamic using html5 in order to be compatibility with all web browser that support html5. The statistics are in real time and it can read from a pcap file. That means that if you close the process, you lose all generated data.

The main functions are:

- Home
  - Talkers
    - Information of big recent flows
    - Indication of throughput of channel
  - Hosts (a screenshot is shown in Figure 20)
    - Graph of top host (IP address) with more traffic (inbound and outbound)
  - Applications
    - Graph of top protocols of analyzed traffic
  - ASNs
    - Graph of top ASNs
  - Senders
    - Graph of top instant flows
- Flows
  - Active flows (a screenshot is shown in Figure 21)
    - Information
      - Date of connection started or first time that it appeared
      - Last packet that have been seen
      - Packets/sent bytes
      - Packets/received bytes
      - Flags of L4 that have been seen
    - Application (adobe, http, flash, etc.)
    - Transport protocol
    - VLAN
    - IP addresses and ports of both extremes of flow
    - Duration
    - Break Down (percentage between inbound and outbound)
    - Instant Throughput
    - Total Bytes for each flow
    - Options
      - Application profile
        - All
        - Adobe Flash
        - Apple
        - Apple iTunes
        - Citrix Online
        - DHCP
        - DNS

        - Dropbox
        - FTP
        - Facebook
        - Google
        - Google Gmail
        - H323
        - HTTP
        - ICMP
        - IMAPS
        - IPSecs
        - LDAP
        - MDNS
        - NTP
        - NetBIOS
        - OpenVPN
        - SMTP
        - SNMP
        - SSL
        - SSL_no_Cert
        - Skype
        - Twitter
        - Unencrypted_Jabber
        - Unknown
        - Yahoo!
    - columns (The same than before)
- Host
    - List (a screenshot is shown in Figure 22)
        - IP Address
        - VLAN
        - Geo-position
        - Name
        - The date when It was seen
        - ASN
        - Break down (percentage between inbound and outbound)
        - Instant throughput
        - All traffic
    - Top hosts (local)
    - Tree
        - Graph of as flows are interrelated
    - Hosts GeoMap
        - Host are draw on the google maps
    - Hosts TreeMap
        - Relational graph of exchanged traffic
    - An analyzed host
        - Summary
            - MAC (Router). The software deduce if MAC is real (the IP address is in the local range) or it is MAC of router (Gateway, it is public IP Address)

        - IP Address
        - Operating System
        - Name, IP address, local/remote, public/private IP address
        - The first time that it was seen
        - The last time that it was seen
        - Percentage of inbound/outbound traffic
        - Inbound traffic, packets/bytes
        - Outbound traffic, packets/bytes
        - Json with information of the host
        - Graph of last activity of host
    - Traffic
        - Protocol graph
        - Protocol statistics
            - Protocol of L4
            - Sent bytes
            - Received bytes
            - Break down (percentage between inbound and outbound)
            - All bytes
            - Percentage between this protocol and all protocols
    - Packets
        - Distribution chart of sent packets
        - Distribution chart of received packets
    - Application Protocols (a screenshot is shown in Figure 23)
        - Summary Graph of application protocols
        - Statistics
            - Application name
            - Sent Bytes
            - Received Bytes
            - Break down (percentage between inbound and outbound)
            - All Bytes
            - Percentage between this and the rest of applications
    - Flows (a screenshot is shown in Figure 24)
        - Information of flows
        - Application
        - L4 protocol
        - IP address and port of client
        - IP address and port of server
        - Duration
        - Instant throughput
        - All bytes
    - Talkers
        - Graph of flows between hosts
    - Contacts
        - Graph of hosts that exchanged data
        - Statistics

        - Clients (start the connection)
            - Destination servers
            - Number of times
        - Servers (listen the calls)
            - Clients that call to it
            - Number of times
    - Local hosts matrix
        - Direction
        - Data that sends or receives
- Interfaces
    - Summary
        - Name (interface or pcap trace)
        - Kind of file: pcap...
        - Bytes (without counting the Ethernet layer)
        - Received packets
        - Sent packets
    - Packets
        - Distribution Graph of packets by size
    - Protocols
        - Application
        - All data that are had seen
        - Percentage
- Accounts
    - User
    - Name
    - Group
- Data exportation
    - Json
    - All / the IP address of a host

In the screenshot of the Figure 20, the host that have transferred more traffic are draw in proportion of all traffic trace. The form of identifying the host is the IP address unless it can resolve the DNS of host. On the bottom of the screenshot, there is a small analog count that indicates the throughput of data. The next Figure 21 shows last active flows of the traffic trace. In this case, it show the instant speed for each active flow.

In the next page, Figure 22 have a list of all host that are present in the traffic trace with information of the instant throughput, last time seen, local or remote host, VLAN ID, etc. This is useful to know the activity for each hosts and if we need to know more information, we can know all flows that manage a host with a simple click in the name or IP address of host.

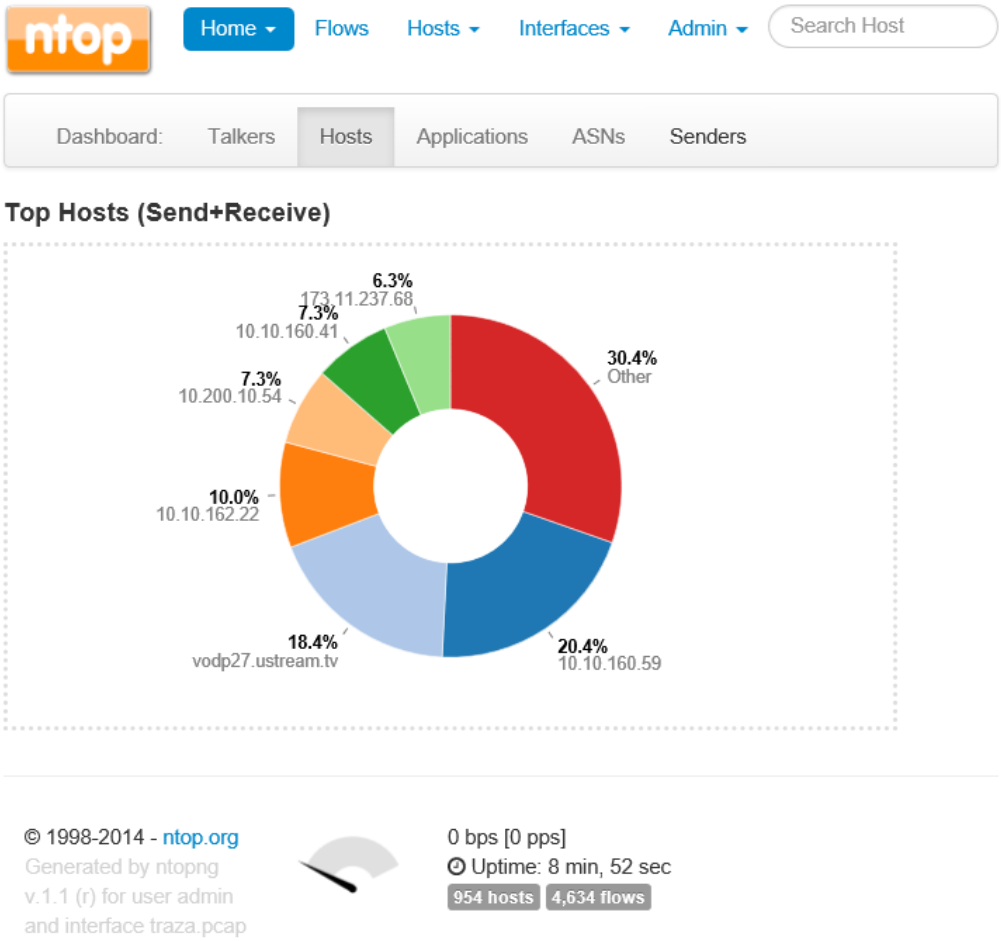

Figure 20: Ntopng, example of hosts

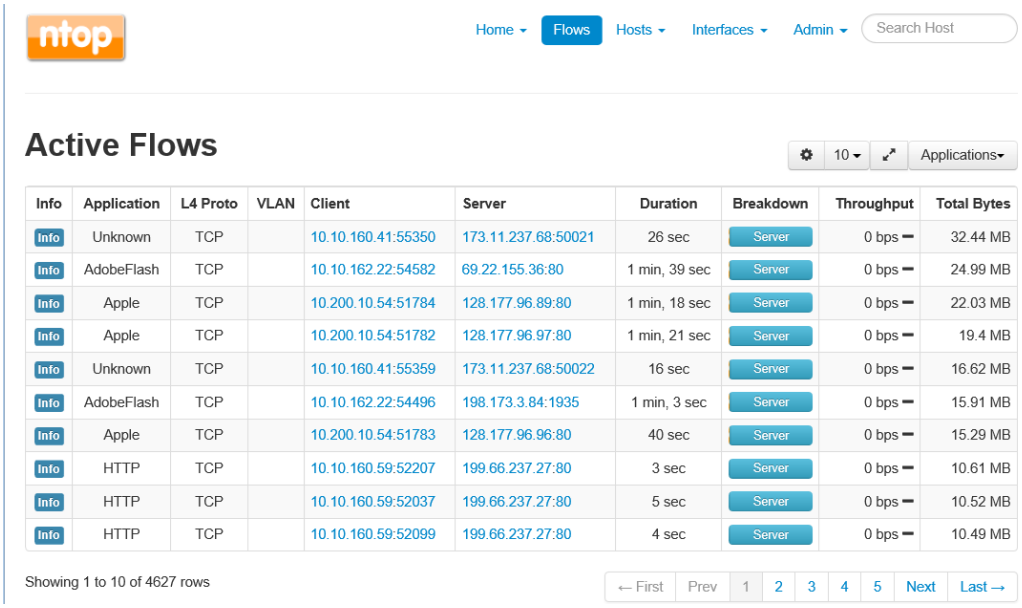

Figure 21: Ntopng, flows that are seen

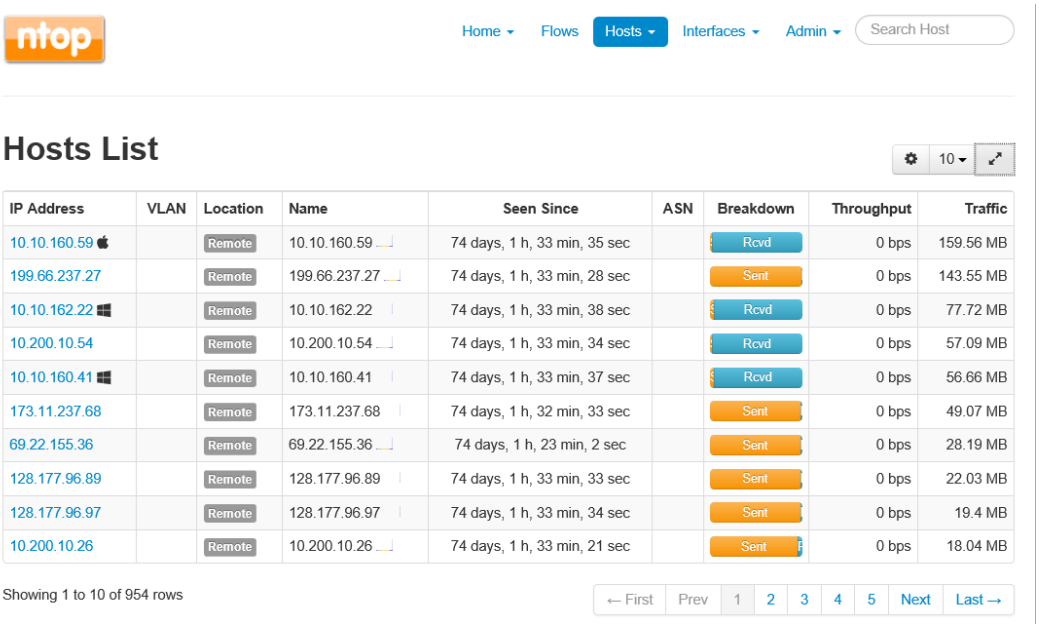

Figure 22: host information on Ntopng

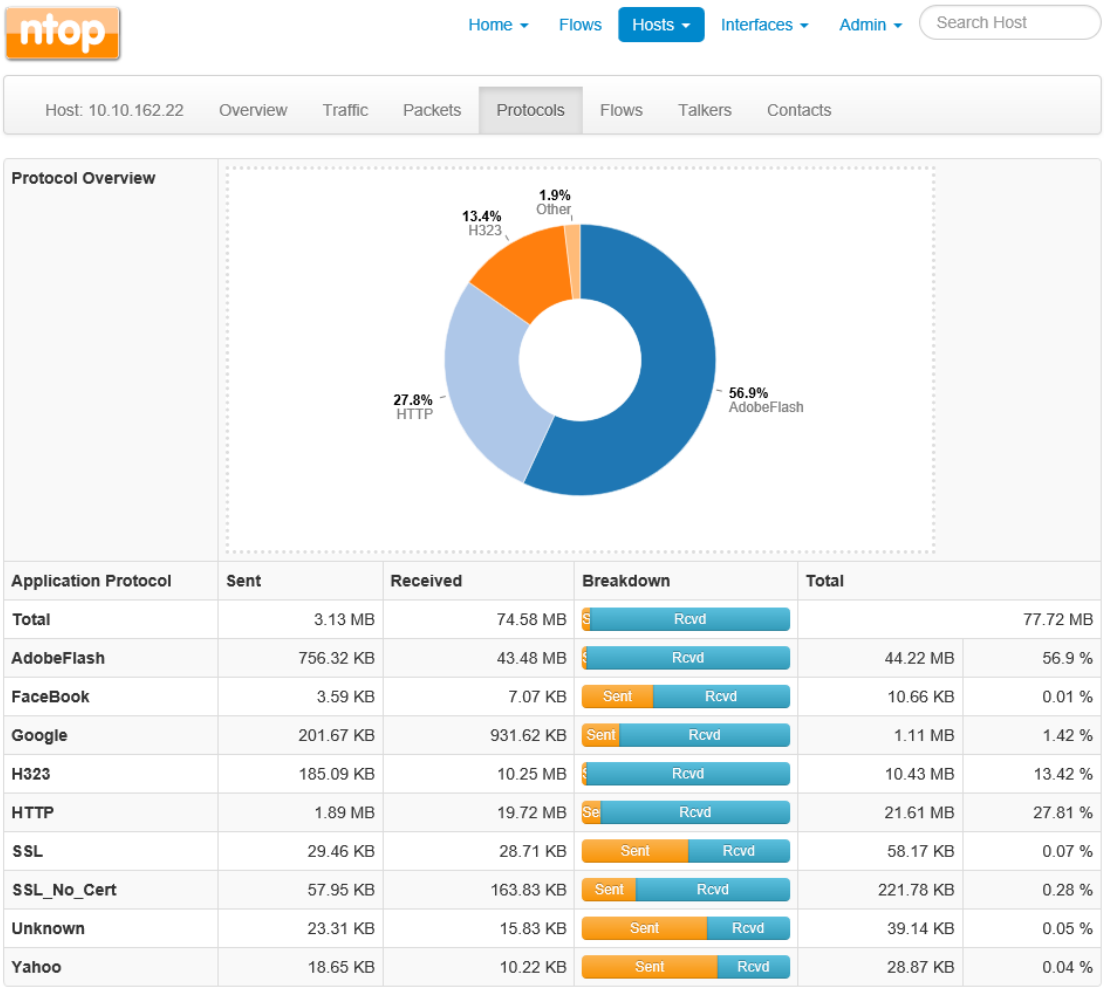

Figure 23: Ntopng, example of protocols that are used by a host

In the Figure 23, we can analyze the protocols that a host uses and the relation of inbound and outbound traffic for each protocol in bytes and percentage.

We can view a summary of a determinate flow as the following is showed in the Figure 24, where there are information of the first and last time seen, traffic breakdown, actual throughput, flags seen in the connection, etc.

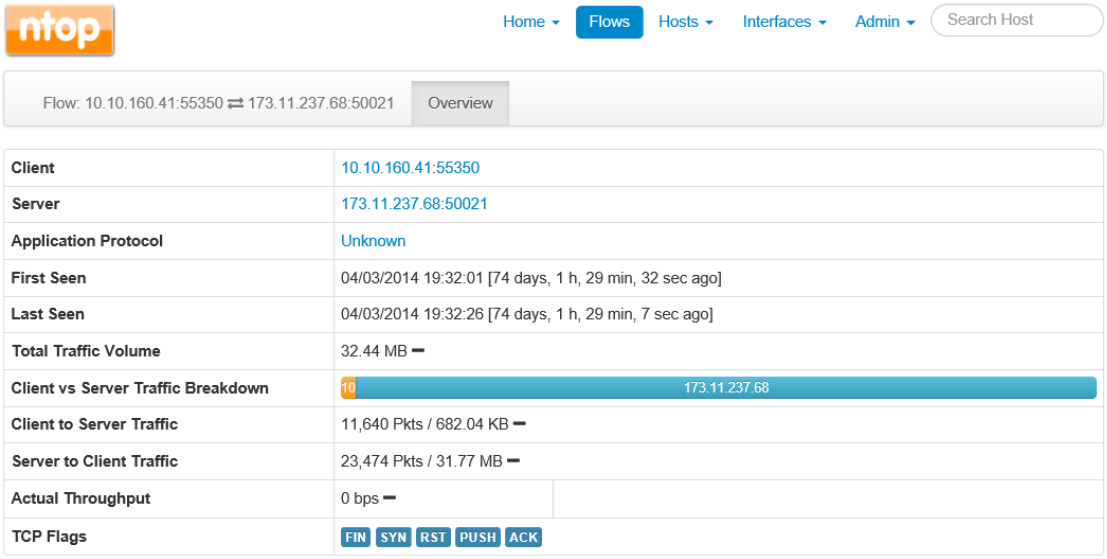

Figure 24: Ntopng, detailed information of a flow

On aspect that we have not forgotten is the modules of Ntop. One of them is Nprobe that collects the information of sFlow or NetFlow that are sent by switches and routers. This module is free. Now it supports more than 170 protocols but you can add more protocols and statistics using our scripts in LUA language. It has a lot of plugins in order to support more protocols as HTTP, DNS, BGP, MySQL, etc. but those plugins and full version of Ntopng, Nprobe, etc. are not free. We have to pay in order to get access all functions and protocols that Ntop support.

2.9. NetworkTimeout

In this occasion, the tool that we are going to analyze is a WebApp [23], although there are version for installing in virtual machines or web servers as AWS (Amazon, etc.), so those version are some free and other are not free. In addition, each version have different functions as for example only analyze the traffic of one host, from file, etc. In our case, we are going to analyze the web App version with a free account. The first disadvantage of this tool is that the traces that are uploaded are shared to everybody, so you lost the privacy. In addition, we can upload traces of 30MB if we do not pay for it. The website offers detailed analysis about different aspects between the layer 2 and layer 7 of the traffic trace. It shows errors, events, counts of flows, protocols, applications, etc.

Other aspect to analyze is the GUI, it is dynamic using the flash player in order to be compatible with all web browser that support this plugin. It has living graphs because it shows more information when the mouse is on the bars, points, etc. One interesting advantage is that it supports to export a PDF format because it saves that we do not have to take a screenshot or printing it in order to save the statistics and graphs. The performance is not be able to rate

because we do not know information of host where it is processed, but a good connection to Internet is necessary because the server has to send us the flash application and the information for statistics.

The main functions are:

- Summary
  - Graphs of bytes and packets by application (L7)
  - Graph of throughput
  - Graphs of HTTP, DNS and DB use
  - GeoMaps
  - Graphs of HTTP, DB, DNS, SMTP, FTP, LDAP, SNNP, FIX, AAA, MemCache, HTTP-AMF, CIFS, NFS, SCSI, ICA and IBMMQ Overview
- Application by default
  - Map
  - Web
    - URI
    - Hits
    - Graph of HTTP performance
    - Number of Http errors with the same URI
    - Time of response
    - Total time
    - Processing time
    - Number of requests
    - Number of responses
    - Number of response Errors
    - Status code
    - Transactions per second
    - RTT
    - Goodput and RTOs
  - SSL
    - Number of SSL version
    - Handshake alerts
    - Cipher suites
    - RTT
  - DB
    - Graphs of DB performance
    - Request bytes and packets
    - Response bytes and packets
    - Request and response RTOs
    - Number of requests
    - Number of responses
    - Number of response errors
    - Graphs of transaction metrics
    - Graphs of transaction per second
    - Graphs of RTT

        - Graphs of goodput and RTOs
    - DNS
        - Graphs of DB performance
        - Request bytes and packets
        - Response bytes and packets
        - Request and response RTOs
        - Number of requests
        - Number of responses
        - Number of response errors
        - Graphs of transaction per second
        - Graphs of RTT
        - Graphs of processing time
    - ICA
        - Graphs of DB performance
        - Client messages
        - Server CGP messages
        - Encrypted
        - Server messages
        - Client CGP messages
        - Aborts
        - Launches
        - Graphs of screen update per second
        - Load time
        - Client latency
        - RTT
        - Graph of APP client bytes
    - NAS
        - Graphs of DB performance
        - Number of responses
        - Number of requests
        - Number of FS info
        - Number of response errors
        - Number of locks
        - Number of reads
        - Number of writes
        - Graph of access time
        - Graph of RTT
        - Graph of transactions per second
    - AAA
        - Goodput and RTOs of congestion requests
        - Goodput and RTOs of congestion responses
        - RTT
        - Transactions per second
        - Requests
        - Aborts

        - Responses
        - Radius requests
        - Errors
    - IBNMQ (system for sharing messages between different systems)
        - Requests
        - Responses
        - Errors
        - Sever messages
        - Server to server
        - Client to server
        - Client messages
        - Warnings
        - Message Methods
        - Graph of transactions per second
        - Graph of RTT
        - Congestion of requests and responses
- Network
    - Summary
        - Graph of IP fragments
        - Graph of Icmpv6 packets
        - Graph of Arp Frames
        - Graph of Unicast, broadcast, Multicast packets and bytes
    - Devices
        - Name host
        - MAC address
        - VLAN
        - IP address
        - Discovery time
        - Description
    - Multicast
        - Graph of packet count by group
        - Graph of byte count by group
        - Table
            - Device
            - Packets and bytes of multicast
            - Packets and bytes of broadcast
    - L2
        - Graph of frame count by distribution
        - Graph of count by type
        - Graph of frame count by size
        - Graph of throughput
        - Graph of packets
        - Number of VLAN tagged
    - L3
        - Graph of packet count by protocol

- Graph of bytes count by protocol
- Table
  - Device
  - Packets in and out
  - Bytes in and out
  - IP fragments in and out
- Graph of top groups (packets and bytes)

○ L7

- Name of device
- Packets in and out
- Bytes in and out
- Graphs of bytes by protocol
- Graphs of packets by protocol

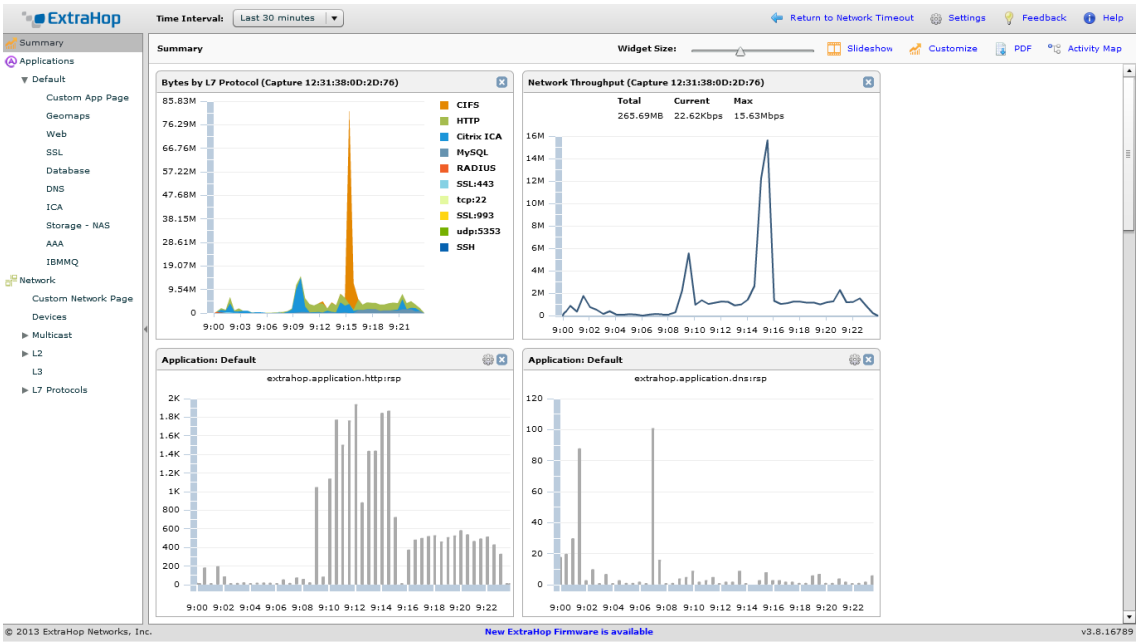

Figure 25: NetworkTimeout, the summary of the traffic trace

On the Figure 25, we can view some graphs of the traffic trace. In this screenshot, there are graphs of the different applications, the throughput of capture, the number of transactions of DNS, http, etc.

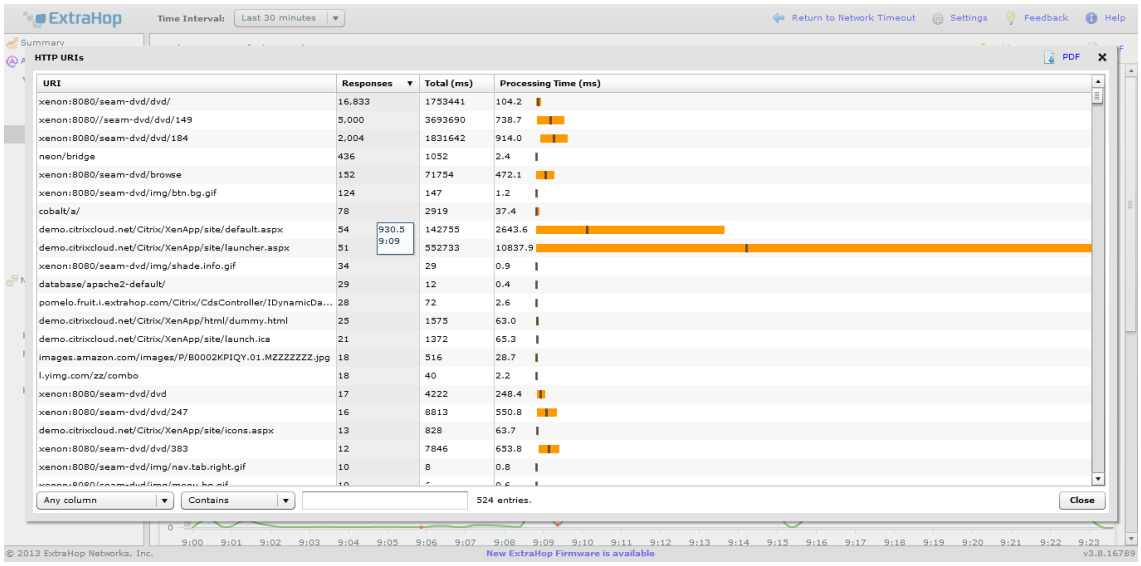

Figure 26: NetworkTimeout, statistics of the http protocol

On the Figure 26, we can view the URIs that have been requested, the time that server needs to process and reply, etc. It is useful to detect problems as for example: busy servers, a bad configuration of TCP/IP stack, bad URIs, etc.

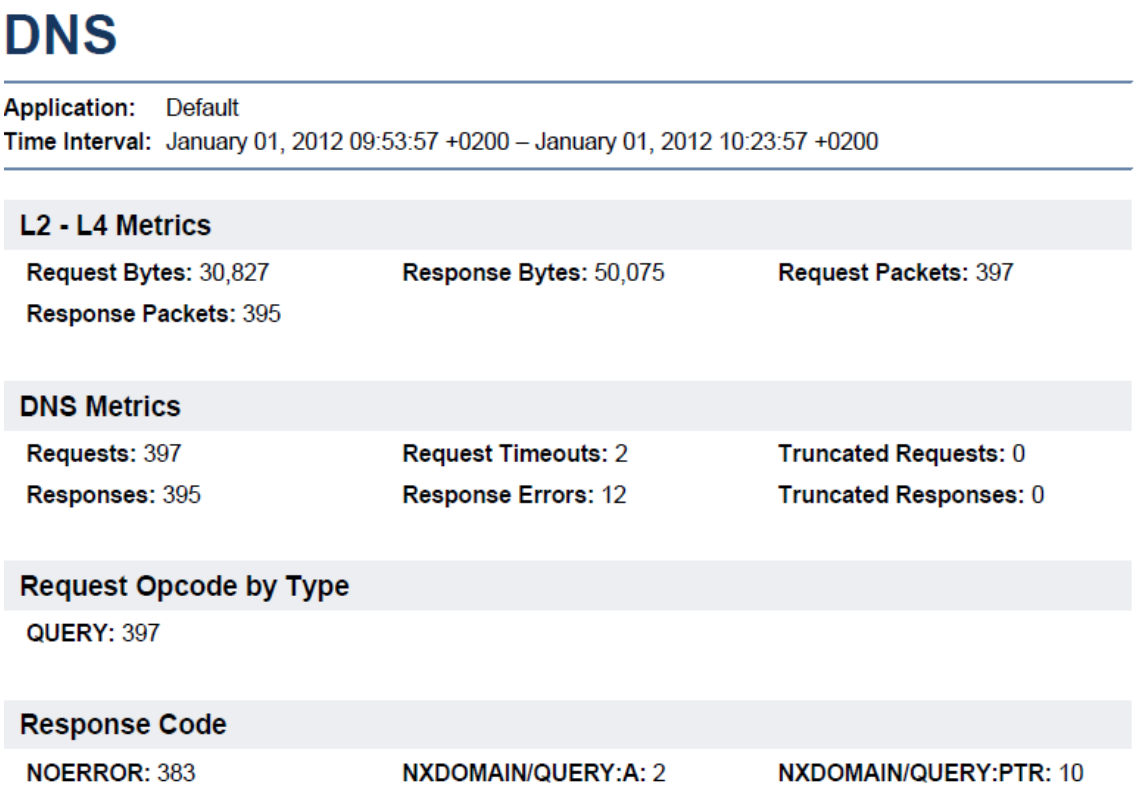

Figure 27: NetworkTimeout, fragment of DNS review.

On the Figure 27, it is a screenshot of a fragment of DNS review. In this case we can view the DNS metrics as for example: the number of requests, responses and response errors. In addition, there are counts of total of response codes that DNS servers have sent.

2.10. Aol/Moloch

This is an open source project in GitHub [24] that allows to analyze on live and from files with the style of Wireshark. The main differences is that it can analyze big files (a lot of Gigabytes) of traffic traces without delay compared with wireshark. It is due to Moloch exports the header information to the BBDD of Elasticsearch [27] (other tool) in order to access data faster.

Other difference is that it allows to apply other filters that wireshark has not got as, for example: the ASN number, countries, etc. But it is not as complete as Wireshark. It is useful to analyze big traffic traces in order to search the flows that you need to analyze and exports them to pcap file and process it with other applications. Wireshark would need more time to export the same flows.

But all are not advantages, because this application need a lot of power. The installation of the application need more than 3-4 GB of space in the HDD. In addition, all captures are copied to a folder the application for future exportations. This space is used by the application and the other tools that it need to work correctly as, for example: java 7 oracle or higher, Elasticsearch, etc. In theory, it should work with the OpenJDK of java, but we could not get it working with this version of java.

Other requisite is the RAM memory, the execution of application need more than 1.2GB. If we view the requirement specification of this application, it should be installed in a server as in the demo version. It is:

- 10x HP server ProLiant DL380 with quad or six cores
- 64GB of RAM
- 2TB of HDD

It is true unless it is used for small traces but then it is not useful. In the case of our traffic trace, it needed less than 30 seconds to copy it to the captures folder and add the information to the DB. It was fast comparing with wireshark.

On the other hand, we have a lot of problems when we want to install it on the computer. For installation, it needs the oracle java with version 7 or higher. In the documentation said that it is for performance and it needs “vm hotspot”. Other problems is the “automatic installation” because it checks the availability of the some libraries and some of them are installed by the soft (the script of the installation) and other are warned and stop the installation. The duration of installation is long, between 1-2 hours in an Intel Dual core 2.66GHz. And we have got it working one time.

Then, we are going to show screenshots of other user [25] in order to explain a few the GUI of the application. To access to the tool, we have to surf to https://localhost:8005/ and enter the login account. In Figure 28, we can view the analysis by flows with the typical header of wireshark but with more information as the country, a bar for indicating the amount of data in this flow, duration, etc. In addition, on the top right, there is map where the intensity of color indicates the number of inbound and outbound flows. The other graph is the number of sessions by time. The form of top is where we have to enter the filter that we want to apply.

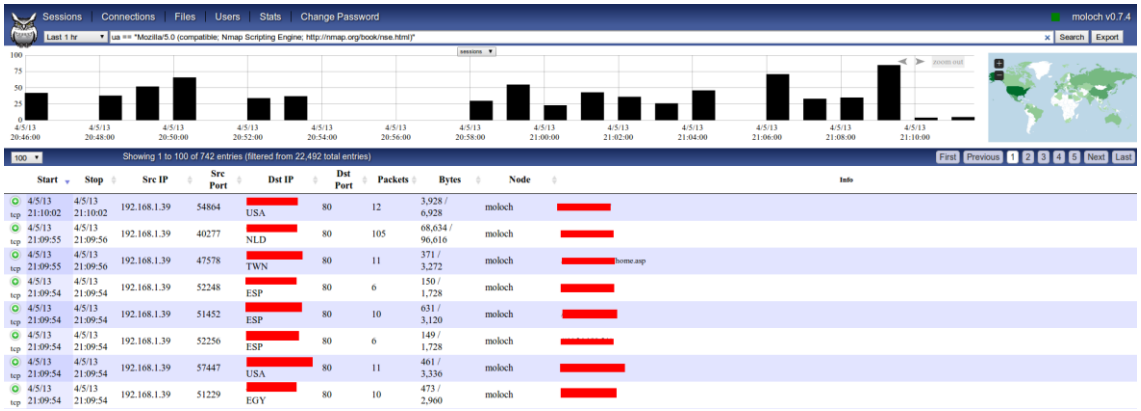

Figure 28: Moloch [25], the basic analysis

In the Figure 29, we have applied the DDOS filter in order to view the number of packets that we have received and sent. A DDOS filter is when a group of hosts (botnet) send the same packet of data without stopping. If the server are not protect of this type of attacks, it could not support all request and then it could not work. This useful to detect attacks of hackers and bot networks.

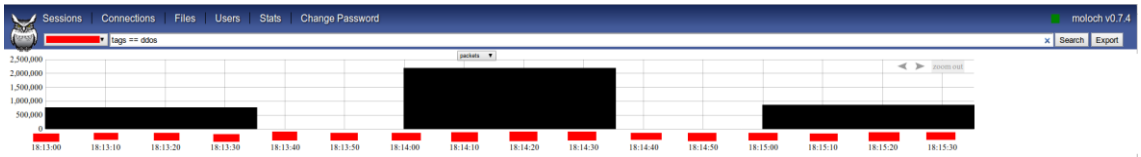

Figure 29: Moloch [25], example of DDOS

And the next screenshot (Figure 30), we can view the drawing of hosts that are received DDOS packets with source port 53. It is useful to find the server that creates more problems.

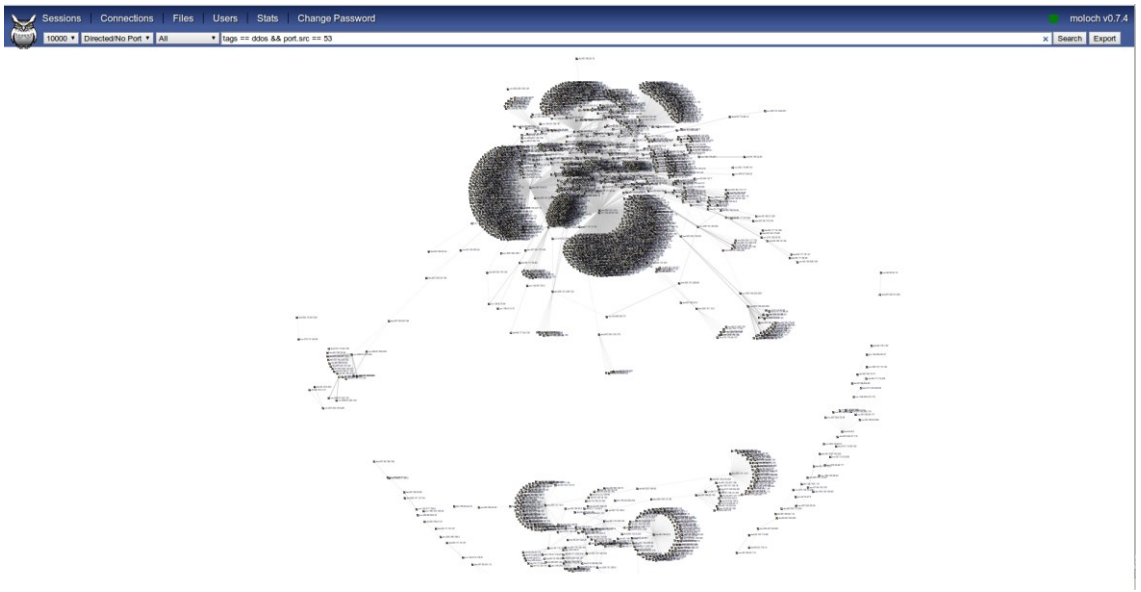

Figure 30: Moloch [25], drawing of flows

2.11. Summary of applications

After we have tested all present applications, our opinion is the following: the first group of applications (Network miner, data echo and Chaos Reader) is very simple because they are more oriented to analyze the http protocol and some derivations as for example they can extract images (Data echo can not extract them) and show the essential information about them. Argus and Justniffer can analyze more protocols but they work mainly in the Linux terminal with an exception of Argus that can draw graphs but you have to execute more lines manually.

Xplico is a derivate of CapAnalysis (it is showed in the website), but CapAnalysis is not free. Between Ntopng and Xplico, is a difficult answers because if we don't need to know or experiment the same experience than user, we use Ntopng but in other case, Xplico is very useful to know that the user saw or listen when was captured in the packet trace. The main characteristics of tools are summarized in the Table 1.

The tool that is called "NetworkTimeout" supports more statistics than Xplico and Ntopng (without modules) but it has not privacy and traces contain public and private information so we prefer using other applications instead of the NetworkTimeout. And the case of Moloch, is a good tool for analyze big traces because it can apply filters and export it very easy but you need a powerful computer or server in order to get it working correctly.

In addition, Table 2 shows information about the tools performance where Ntopng is selected as the best of all again. There are 2 tools that are not rated because of problems of installation that have been explained previously. The characteristics that have been rated are the processor architecture, RAM memory and storage needed and finally the duration for analyzing our trace.

Table 1: comparison of the main characteristics between applications

App	SO	File	Live	Portable	Graphs	View of content	Protocols	Performance	Costs
Network Miner		✓	✓	✓	✗	✓	HTTP, FTP, email, SSL, TLS, TOR	Slow due to write data	Free ($700 to get full characteristics)
Data Echo		✓	✓	✗	✗	✓	HTTP	Fast	Free
ChaosReader		✓	✓	✓	✗	✗	HTTP, FTP, SMTP	Fast	Free
Argus		✓	✓	✗	✓	✗	Ethernet, IP, TCP/UDP	Very Fast	Free
Xplico		✓	✓	✗	✓	✓	Main	Slow	Free

Wireshark		✓	✓	✓	✓	✓	+Main	Normal	Free
Justniffer		✓	✓	✗	✗	✓	TCP	Not rated	Free
CapAnalysis		✓	✓	✗	✓	✗	Main	Fast	178€
Ntopng		✓	✓	✗	✓	✗	Main, depend of active modules	Fast	Free modules are not free
NetworkTimeout (web version)		✓	✗	✓	✓	✗	Main	Fast	Free. More characteristics and modules are not free
Moloch		✓	✓	✗	✓	✓	Main	Fast	Free

Table 2: performance Comparison with the example trace

App	CPU	Duration (s)	RAM (MB)	Storage (MB)
Network Miner	x32	165	800	170
Data Echo	x32	60	100	<100
ChaosReader	x32	10	<100	500
Argus	x32	<10	<50	<20
Xplico	x32 y x64	210	100*	550*
Wireshark	x32 y x64	120	470	-
Justniffer	x32 y x64	**	**	**
CapAnalysis	x32 y x64	60	<50*	<100*
Ntopng	x32	10	<50*	<100*
NetworkTimeout	X32, x64, ARM	***	***	***
Moloch	x32	<30	1.5GB	>500

*estimated values
**it could not be rated
*** The performance on web apps can not be rated

3. Useful tools for traffic analysis

In any analysis, we need to have certain group of tools in order to find the problems that the network has. To get fast and accurate results, we are going to explain several tools that we will use in the section called “general methods of traffic analysis”.

3.1. CapInfo

This tool [1] gives us a summary of main characteristics of a packet trace that is useful to know. The command line that we have to use is the following:

CapInfo [options] <infile>…

This application is easy to use as we have to execute it in a terminal with the path of packet trace as argument. This tool gives us the following information:

- File format, type of capture
- Encapsulation of packets in the trace
- Number of packets
- File size
- Size of captured data
- Duration of captured trace
- Date of the start and the end of the capture
- Average speed in bps and Bps
- Average size of packets

3.2. Editcap

The next tool that we are going to explain is editcap [2] that is used to convert the trace to other formats and versions in order to reduce problems of compatibility. To use it, we have to give as arguments the following data:

- Name of input trace file
- Name of output file
- -F <kind> to set the format of exportation, for example libpcap.
- -T <kind> to set the input encapsulation from which to export, for example Ether.

There are other options that you can set like the range of time that you wish to export (-A -B), changing the timestamp of the trace (-t), cutting each packet to delete private data (-s) between others options that can result very attractive in some situations. Those options can be consulted in the user manual.

3.3. Awk

As we are working with raw data to be more efficient and fast on high speed networks, we usually can not execute applications as wireshark due to high resources consumption and necessity of extra hardware. So we can use command line applications that are very useful to process the data by columns and registers as we wish. This tool awk [3] helps us to do these functions. The arguments that we have to give are the following:

- IN.txt: this is the input text file

- -f Out.txt: this is the output file
- -F ",": we set the character that delimits the columns

This tool supports processing line by line the content of a text file or the content that other application creates dynamically to standard output. Then, we can process the content of each column accessing each value. The value of each column is saved in $1, $2, $3, etc. variables. Awk can process with classical operators as +-*/%^!++--<><=>= ==!=…. Those are similar operators to those of c/c++. Then, it can reorganize, filter or process the content of each register.

There is a derived application from this that is called gawk. Both tools have the same basic functions, but gawk has some extra functions.

3.4. Sort

Other tool that is very advisable to use, is Sort [4] because it orders the content of file by generated text columns. The options that it provides are the following:

- -r: to set that it sorts in reverse order
- -u: it deletes the duplicated lines
- -o <file.txt>: to export the results to a text file
- -n: to compare with numbers (strings are converted to numbers)
- -k <6>: to set what columns it have to sort

3.5. MacConversations

The objective of this tool [5] is to get a list of talkers at link level that are present on the LAN. The command line is:

MacConversations –f input_file [-p formato] [-u] [-d pkts_a_descartar] [-o output_file_instead_of_stdout] [-t pcap_filter] [-v] [-s] [-g] [-h]

To use this tool, you can set the followings options:

- -f <file>: you have to set the path of the packet trace
- -u: this means that the last file is an array of traces
- -p <pcap>: you indicate the format of the input packet trace: raw or pcap format.
- -o <file>: file where it will export the content instead of showing on terminal
- -g: give global statistics through stderr

We get as output as information is formed by lines, that each line is a MAC flow and in addition we get global statistics. We get the following information:

- Source MAC
- Destination MAC
- Number of packets
- Number of bytes
- Timestamp of first packet
- Timestamp of last packet
- Number of trace packet of first packet
- Number of trace packet of last packet

3.6. Mapvalues

When you export data from an Ethernet conversation, it is very useful to know the mapping between the number present in the some packet field and its meaning. This script [5] is used to resolve the mapping using as an input a data file with all the possible mappings. The command line:

Mapvalues.sh <file to translate > <column> <map file>

You should use it the following format:

- <file.txt>: this is the input file
- <1>: you should indicate the number of column to translate
- <filemap.txt>: this is the file that contain the translation table

3.7. IpConversations

This tool [5] provides an IP flow summary that there is between pair hosts, or multicast/broadcast traffic. The command line:

IpConversations –f input_file [-p formato] [-u] [-e pkts_a_descartar] [-1 dirRed/prefixLengh] [-2 dirRed/prefixLengh] [-o output_file_instead_of_stdout] [-tpcap_filter] [-m outputfile_timeseries|-] [-n msecs_point_in_timeseries] [-z] [-g] [-v] [-s] [-d] [-a] [-h]

The parameters that we have to set are the following:

- -f <file>: here is where we set the trace file or trace list file
- -u: this indicates that the last file is an array of trace files.
- -p <pcap>: trace format, for example: raw, pcap, etc.
- -o <file>: file contains the output of application
- -g: it gives global statistics through stderr
- -a: it deactivates calculating the number of ports

The output that it generates has the following information:

- Source IP address
- Destination IP address
- Source MAC
- Destination MAC
- Boolean to notify the variation of route
- Number of packets seen from source to destination
- Number of packets seen from destination to source
- Number of fragment packets
- Number of bytes of fragment packets
- First packet seen
- Last packet seen
- Number of bytes of packets seen from source to destination
- Number of bytes of packets seen from destination to source
- List of TCP ports seen in source
- List of TCP ports seen in destination

- Additional MAC seen
- Additional MAC seen
- Indicator of more Macs

In addition, we can get the global statistics that the application give through error output. They are the following:

- Number of Ethernet frames
- Accumulated number of Ethernet Bytes without counting the CRC
- Time of first Ethernet frame
- Time of last Ethernet frame
- Number of 802.1Q packets seen
- Number of IPv6 packets seen
- Accumulated number of Ethernet bytes of IPv6 packets without counting the CRC
- Number of IPv4 packets seen
- Accumulated number of Ethernet bytes of IPv4 packets without counting the CRC
- Number of IPv4 fragment packets seen
- Number of IPv4 fragment packets seen with incomplete IP header
- Number of IPv4 fragment packets seen that have as destination Multicast IP address
- List of encapsulated protocols over IP layer
- Number of IPv4 fragment packets of each protocol
- Accumulated number of Ethernet bytes of IPv4 packets of each protocol without counting the CRC
- Duration of analysis

3.8. ProcesaConexiones

Ya disponemos de los flujos IP que hay en la traza, pero nos resultaría de mayor interés conocer en más detalle las conexiones TCP que haya en la traza. Esta aplicación [5] nos soluciona esta necesidad, exportando una gran cantidad de estadísticas que resultan muy útiles para la detección de problemas en las conexiones TCP. La línea tipo de ejecución es:

procesaConexiones –f input_file [-p formato] [-u] [-d pkts_a_descartar] [-o ouput_file_instead_of_stdout] [-t pcap_filter] [-i] [-v] [-s] [-g] [-a maxTimepoInactividad] [-b tipoDeHash] [-c] [-e maxPacketsBeforeConnectionsClean] [-j maxSecondsBeforeConnectionsClean] [-l modoLimpieza] [-h]

Hay mucha más opciones como sucede en la mayoría de las aplicaciones pero estas son las principales que nos resultaran bastante útiles. Tendremos que pasarles los siguientes argumentos:

- -f <file.txt>: El fichero con la traza a nivel Ethernet
- -o <Out.txt>: El fichero donde exportará el contenido a nivel TCP
- -g: Nos da estadísticas globales por la salida stderr
- -p <pcap>: Especificamos el formato de exportación, típico pcap, raw,…
- -a <núm.>: El tiempo máximo de inactividad de conexión TCP (libera RAM)
- -e <núm.>: Nº paquetes procesados para liberar RAM
- -j <núm.>: Tiempo que debe transcurrir de traza para liberar RAM

Respecto a la salida, disponemos de dos formas por la terminal o volcado a fichero. Además, disponemos de la opción –g como en la aplicación anterior que nos permite obtener las estadísticas globales por la salida de error. Por la salida principal, obtenemos 101 columnas diferentes de datos que se obtienen por cada conexión TCP. Las principales columnas son:

- Dirección IP:Puerto en origen y destino
- Duración de la conexión vista
- Duración del handsake inicial de la conexión
- Duración de la transmisión de datos en cada sentido
- Numero de paquetes total y por cada flag SYN, RST, FIN, Push
- Numero de ACK duplicados
- Bytes transmitidos
- Bytes desordenados
- MAC de origen y destino de cada sentido
- Mínimo, máximo tamaño de ventana anunciada (ignora la opción de escalado)
- Número de veces que la ventana se llena y duración de la misma

3.9. InfoDups

En muchas conexiones, por el tipo de transferencia, el estado de la red, etc. incluso por el modo de captura (SPAN en switch), puede provocar en algunas ocasiones que vuelvan a aparecer paquetes duplicados. La solución para ver la traza sin paquetes duplicados para no pensar que se tratan de retransmisiones y ver las estadísticas de cuantas ha habido, es esta aplicación InfoDups[5], la cual tiene la siguiente línea de ejecución:

InfoDups [options] –i <file>

Para usarla deberemos pasarle tan fácilmente los siguientes argumentos cuando la ejecutemos:

- -D: indicamos que la traza está en formato RAW
- -i <file.txt>: El fichero a analizar
- -x: devuelve información detallada
- -v: muestra el progreso del análisis
- -T <núm.>: número de hilos para procesar el fichero

Con estas opciones ya podemos sacar rendimiento a esta aplicación y facilitarnos el análisis de la traza. Dispone de más opciones pero se han mencionado aquellas que resultan más interesantes.

La salida de la aplicación nuevamente son dos, una por terminal y otra por la salida de error. Si redirigimos ambas obtendremos los datos almacenados en ficheros. La salida de error, es para la notificación de problemas en el parseo, no se obtienen estadísticas como en las otras ocasiones. En la salida principal obtendremos la siguiente información:

- Posición del duplicado
- Diferencia en la posición respecto al duplicado
- Tipo de duplicado
- Flag de payload vacío

- Si ha cambiado el VLAN id
- Si ha cambiado el DSCP id
- Diferencia temporal entre copias
- Diferencia del TTL entre copias

Se pueden obtener más información de la salida si se activa el flag correspondiente alcanzando hasta 23 parámetros diferentes.

3.10. Analisis_dns.py

Esta aplicación [5] escrita en Python y en bash nos generará estadísticas sobre la resolución de nombre (DNS) de aquella traza que le hagamos analizar. Esta aplicación está formada por dos herramientas, una encargada de extraer la información de la traza y la segunda en generar estadísticas sobre los datos extraídos. La forma de ejecutarlas es la siguiente:

Python analisis_dns.py <traza. Pcap> >salidaSDUP_dns.txt

analisis_dns.sh salidaSDUP_dns.txt

La primera nos extraerá la siguiente información entre otra más:

- Dirección IP del solicitante
- Dirección IP del servidor DNS
- Hora de inicio petición
- Duración de la resolución
- Dirección resuelta

En el caso de la segunda herramienta nos devolverá gráficas y estadísticas sobre el comportamiento del servicio DNS. En el caso de que no sean las que necesitamos, con otras herramientas podemos realizar un postprocesado del fichero de salida y generar nuestras propias estadísticas. Resulta muy útil para detectar problemas de saturación entre otros. Para más información se puede consultar su ayuda en el propio programa.

3.11. ProcesaTrazasDNS

Existe otra aplicación del mismo estilo que la anterior que analiza también el protocolo DNS que se llama procesaTrazasDNS [5]. Esta aplicación se diferencia de la anterior en varios aspectos. El primero de todos ellos es el tiempo que necesita para procesar la traza. Como ejemplo, se ha realizado la prueba con una traza de solo 22.000 paquetes de DNS en los que esta ha tardado 78 veces menos que la anterior herramienta. Esto se debe a que una está programada en Python y ésta está escrita en c/c++. Otro aspecto que tiene como ventaja esta herramienta a la anterior es el nivel de estadísticas que puedes obtener con ella. La anterior solo devuelve información de las respuestas del servicio de DNS, en cambio esta nos devolverá información de las peticiones y respuestas e incluso procesa más campos del protocolo DNS.

Para poder utilizar esta herramienta basta con que utilicemos la siguiente línea de ejecución:

parserDNS [-h] [-v] [-p format] –i pcap [-a] [-t filter] [-d filePacketDiscard] [-f] > salida.txt

Los argumentos que nos encontramos son los siguientes:

- -h: obtención de ayuda de la herramienta
- -v: obtención de estadísticas de procesamiento
- -p: formato de la traza la cual puede ser raw or pcap
- -i: ruta de la traza que se va analizar
- -a: si la ruta anterior se trata de un conjunto de rutas de trazas
- -t: si se desea aplicar un filtro previo al análisis
- -d: no tener en cuenta aquellos paquetes indicados
- -f: para verificar si se trata de un paquete DNS o es otro protocolo que utiliza el puerto 53.

La salida de esta herramienta es más extensa obteniendo los siguientes campos en la salida:

- Numero de preguntas vistas para esta Query
- Numero de respuestas vistas para esta Query
- Código de error en la respuesta
- Flag para indicar que hay truncamiento en la respuesta (petición vía TCP)
- Flag para indicar que el servidor puede hacer peticiones recursivas
- Timestamp de la pregunta
- Timestamp de la respuesta
- Dirección IP del cliente y servidor
- Puerto origen del cliente
- ID de la petición DNS
- Cadena con las preguntas
- Cadena con las respuestas

3.12. IdentificaRedes

Con esta herramienta [5] lo que se logra es a través de otras micro-herramientas ya comentadas, convertir la traza en grafos a través de 4 sencillos pasos para así obtener un contexto de la topología lógica de la red que se va analizar. Además, a través de esta herramienta se obtienen diversos resúmenes que nos servirán para posteriores análisis ya que se basa en parte de ellos para realizar la topología.

Para hacer uso de esta herramienta se necesitan instalar varios programas como es java e yEd graphic entre otros. Para ello, disponen de un micro instalador para hacer la instalación más llevadera. Se puede consultar los pasos detallados en el manual que proviene la herramienta de cómo se usa.

Para hacer uso de dicha herramienta, al tratarse de un conjunto de micro-herramientas, deberemos de realizar 4 pasos para llevarlo a cabo. Todos ellos se deberán ejecutar uno a uno de forma manual.

1. Ejecutar la herramienta IdentificaRedes con el que obtendremos información relevante de la traza a nivel de Ethernet, IP, y protocolos de enrutamiento, STP, LLDP, etc.
2. ObtenerInfoTraza: con este obtendremos más información detallada de a través de análisis recursivos de los archivos obtenidos del paso anterior. Logrando un total de 25 archivos de información de diferentes análisis

3. Obtención de la topología de la red a nivel MAC. Para ello utilizaremos el programa yEd con el que a través de la selección del layout que más se adecue al análisis (recomendable de tipo radial) tendremos la topología de una forma sencilla y además de ver a simple vista que equipos son los que más tráfico realizan.
4. D3js. A través de este JavaScript de ejecución vía navegador web el cual requiere que sea de una versión reciente por el tipo de API que necesita. Se obtiene un grafo de que servidores/equipos son los que más tráfico intercambian. Con una herramienta llamada PhantomJS se puede exportar las gráficas sin necesidad de navegador al formato deseado como puede ser PNG, PDF, BMP,…

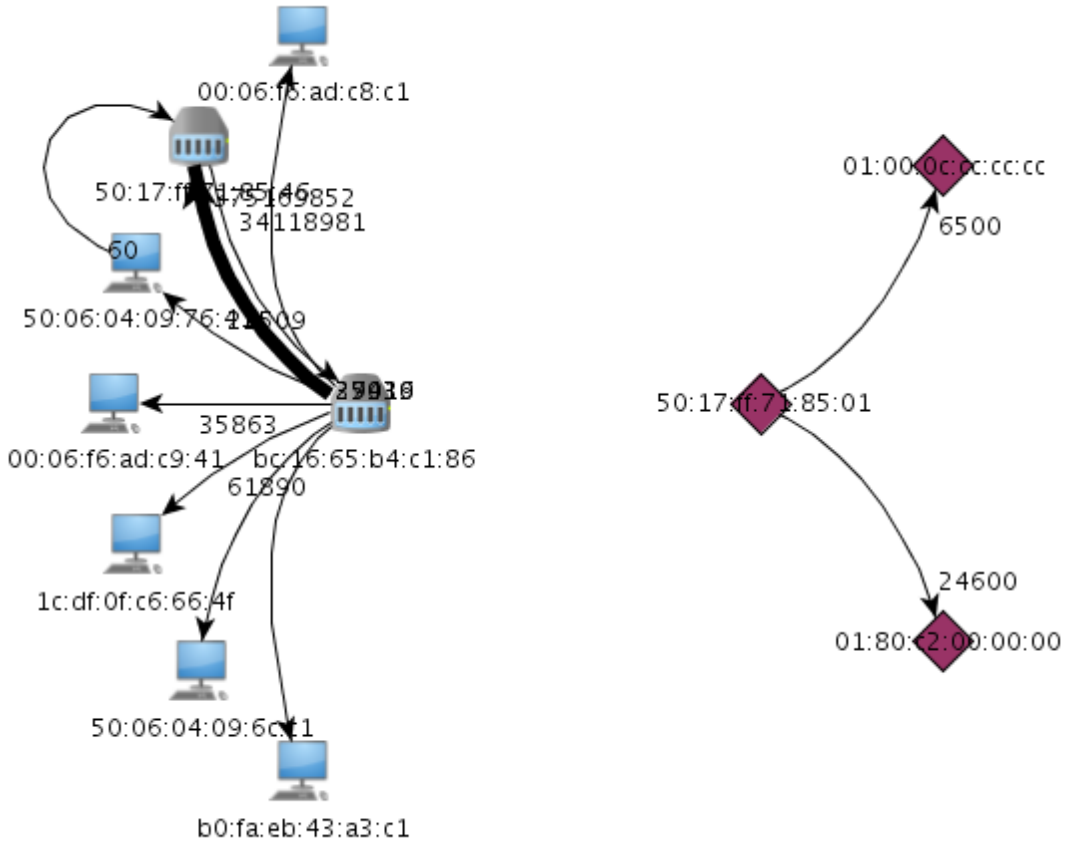

Figura 31: ejemplo de grafo de la topología

En la Figura 31, podemos apreciar el grafo lógico de los flujos que se han establecido entre aquellos equipos que se han detectado en la traza. En el caso de los routers que se han dibujado se debe a la realización en función del número de IPs detectadas con misma MAC. En caso contrario se dibuja con el icono de un equipo. Para poder dibujar el tráfico generado por un tráfico Multicast, se emplea el icono de rombo morado con la MAC correspondiente al grupo Multicast que representa.

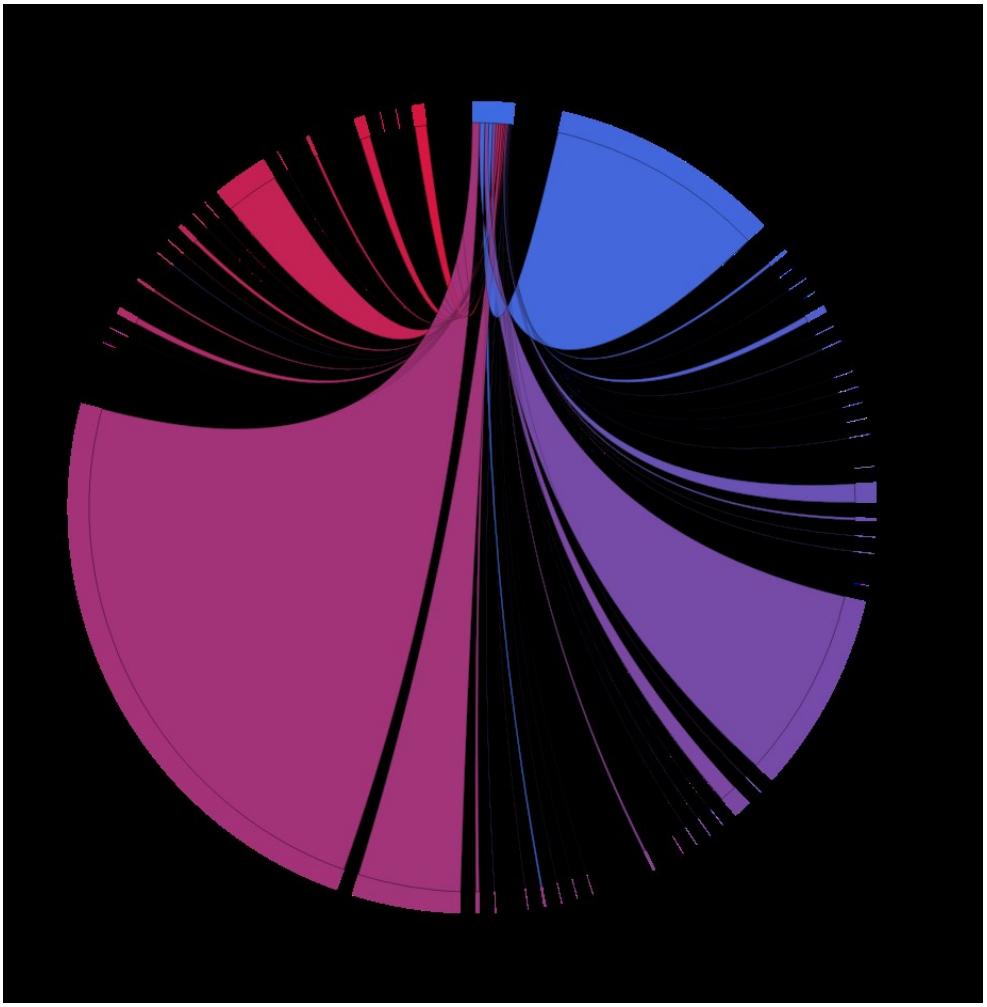

Figura 32: Ejemplo grafo de nivel de red IP

La Figura 32 se corresponde con la representación de la gráfica Chord de redes IP que se han accedido en este caso con un mascara de red /12 (se define en el paso 2º). La forma de interpretar esta grafica es la siguiente: la anchura del flujo de la red correspondiente es la cantidad de bytes que envía en comparación al resto del tráfico de la traza. La anchura de la red es la cantidad de bytes que ha enviado en total en comparación con los del resto de la traza.

Finalmente obtenemos una variedad de ficheros con estadísticas a nivel de Ethernet, protocolos de enrutamiento, balanceo de carga, de nivel de red para obtener la topología lógica y grafos circulares donde apreciamos los porcentajes de quien envía más y a quien se lo envía.

3.13. TCPTrace

Esta es un herramienta [6] muy útil para obtener información de la traza. Tiene diferentes usos, como el más básico que te ofrece información de entre qué equipos intercambian información avisándote de advertencias como puede ser los paquetes duplicados ya sean por retransmisiones, o debido a la forma de captura que provoque duplicados íntegros. Esta herramienta está disponible para todos los sistemas operativos. Y en el caso de exportar gráficas, requiere de otro programa como puede ser Jplot o Xplot para su visualización.

Para obtener la información corta y básica basta con que ejecutar el programa pasándole como argumento el path de la traza. La información que obtienes en las columnas

son el número de paquetes por segundo y el número de paquetes vistos. También te dice si la conexión está completa, si se ha reseteado, completada con Reset, si es un sentido,… todo ello para que conozcas de un simple vistazo el estado de las conexiones de los equipos.

Tiene diferentes opciones con argumentos adicionales como son:

- -b: sirve para forzar a tener una salida en formato corto
- -l: obtienes información detallada por cada conexión o flujo
- -oN: fuerzas a que solo muestre información sobre la conexión/flujo número N que aparece en la traza. Esto es muy habitual utilizarlo en conjunto con la función de –l para tener información detallada sobre cierta conexión que has detectado que podría tener problemas.
- -p: obtienes información resumida de las cabeceras de los paquetes de las conexiones/flujos que hay en la traza. Si deseas obtener información detallada sobre una conexión/flujo en particular, se utiliza –P y –oN para obtenerlo.
- -r –l: con esta combinación obtendremos información detallada sobre la conexión e información sobre estadísticas del RTT, pérdidas, acks,… que ha tenido la conexión/flujo que hayamos seleccionado con –oN o de todas las que haya en la traza.
- -e: con este argumento conseguiremos información detallada sobre el contenido del payload de las conexiones/flujos de la traza o del que hayas elegido.
- -n: Permite desactivar el realizar la resolución de nombres (DNS) a las direcciones IP que hay en la traza.

En el caso de que se desee obtener gráficas a través de esta herramienta, lo que se debe utilizar son los siguientes comandos:

- -T: gráfica del Throughput
- -R: gráfica del RTT
- -S: gráfica del crecimiento del número de secuencia
- -N: gráfica de la evolución de la ventana cwin
- -F: gráfica del tamaño de cada segmento
- -G: crear todas las gráficas

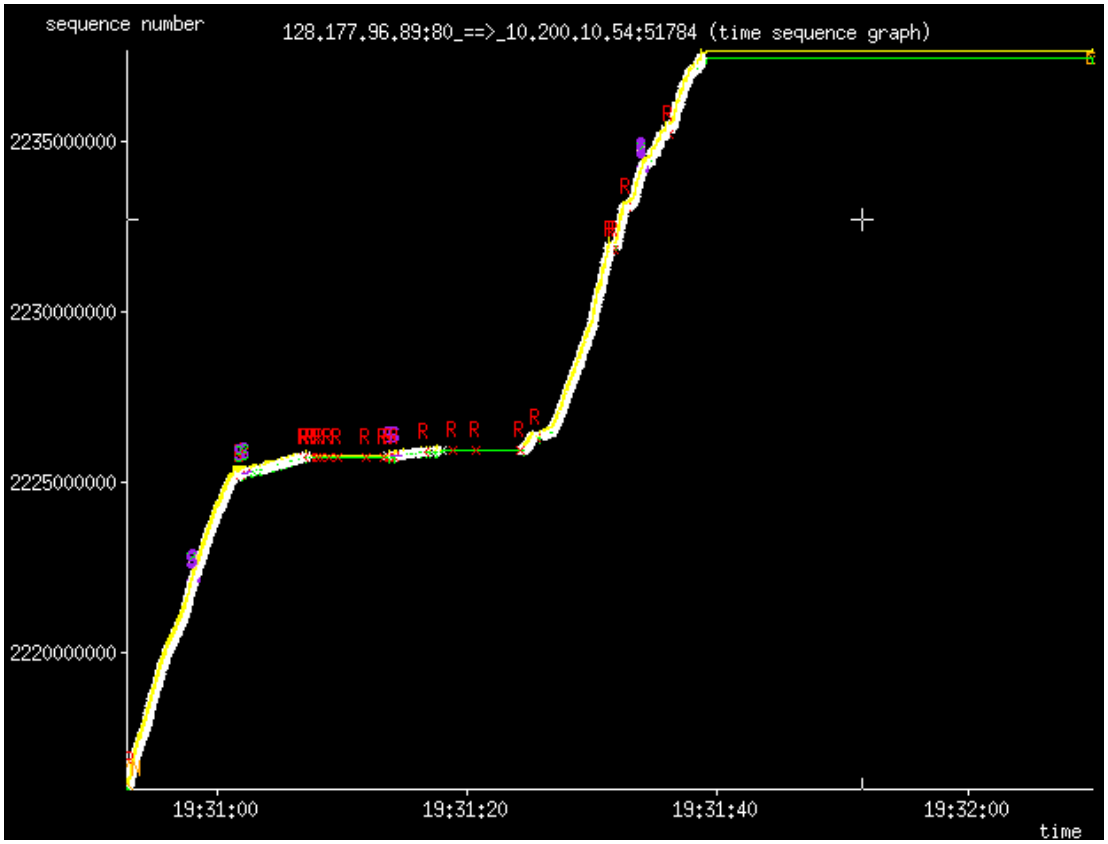

Figura 33: Ejemplo gráfica número de secuencia con TCPTrace

Es fácil identificar cada una de ellas dado que el nombre de los ficheros que crea tiene la finalización del nombre con su correspondiente identificación del tipo de gráfica. La más interesante de todas es la gráfica de crecimiento del número de secuencia (Figura 33), ya que te permite ver cuando envía información y cuánto tarda en recibir la confirmación, como cuanto máximo puede enviar en cada momento (deduces que ventana es la limitante), cuando se producen eventos de retransmisión, Flags de inicio y fin de conexión (si están presentes en la captura), etc. Se tiene toda la información en el manual del programa y en guías disponibles en la red.

3.14. Xplot

Ya tenemos exportado las gráficas con el anterior programa pero para representarlas tenemos que utilizar otros programas como pueden ser Jplot o Xplot [6]. Dado que Xplot está disponible para todas las plataformas, vamos a utilizar este, además de que es más eficiente en términos de consumo que Jplot que está montado sobre Java y da algunos problemas según la versión de java que emplees. Para usar Xplot en Windows debemos instalar Xming para poder usar el sistema de ventanas de Linux. Una vez instalado, debemos correr en una terminal de Windows (cmd.exe) el comando “set DISPLAY=localhost:0” para que cuando vaya hacer uso Xplot, sepa dónde debe representarlo. Finalmente debemos ejecutar xplot.exe <Nombre fichero.xpl> y ya dispondremos de las correspondientes gráficas. Si se desea abrir varias de una sola vez, tan solo debemos hacer uso del comodín *.xpl para que abra todos los ficheros correspondientes. El tipo de grafica que nos genera es el que hemos visto en la Figura 33.

Para conocer cómo se debe manejar tenemos la ayuda con el argumento –h, guías sobre su uso, pero es tan sencillo como crear rectángulo para hacer zoom en dicha zona, y clic para deshacer el zoom realizado, y clic con el secundario para cerrar dicho gráfico.

3.15. Xming

Este pequeño programa Xming [7] es muy utilizado por herramientas típicas de Linux que se han exportado a Windows que requieren de un servidor de ventanas al estilo Linux (servidor de X) para poder exportar las ventanas gráficas de los equipos remotos. Es tan sencillo de usar como instalar, ejecutar, y en los correspondientes programas apuntar a localhost:0 para que se redireccione las ventanas a dicho servidor.

3.16. Script-informes

Dada la elevada cantidad de herramientas que se dispone y lo que debemos realizar en común para cualquier tipo de análisis, tenemos este programa [5] con el que podemos ser más eficientes, ya que nos ejecuta múltiples herramientas automatizadas de forma que tengamos disponibles los informes más básicos, conversión de la traza a un formato común de las herramientas disponibles, etc. Además, para que podamos ver en un momento cual puede ser el problema, nos ofrece un fichero HTML en el que nos muestra las principales estadísticas sobre:

- RTT
- Retransmisiones
- Pérdidas
- Duplicados
- Equipos con más tráfico
- Estadísticas sobre ICMP, DHCP, DNS

Para más información, nos la muestra en tiempo de ejecución.

3.17. Wireshark

Una herramienta muy común y extendida entre los analistas de trazas es Wireshark [13], con la que podremos analizar la mayor parte de las trazas que nos encontremos con un buen abanico de opciones. Se han realizado múltiples manuales de uso de esta herramienta y personalización. A priori, se desconocía algunas de sus funciones como la posibilidad de exportar e importar los perfiles que generas al personalizar la interfaz. Por lo que vamos a explicar cómo adaptarlo para analizar el DNS y como exportar el perfil generado.

3.17.1. Creación del perfil de configuración

Una buena forma de compartir personalizaciones y tener la interfaz de wireshark ordenada y limpia es la creación de perfiles. Para ellos nos situaremos con el ratón en la parte inferior donde pone “Profile”, desplegaremos el menú con el botón secundario. Ahí seleccionaremos crear uno nuevo como podemos ver en la Figura 34.

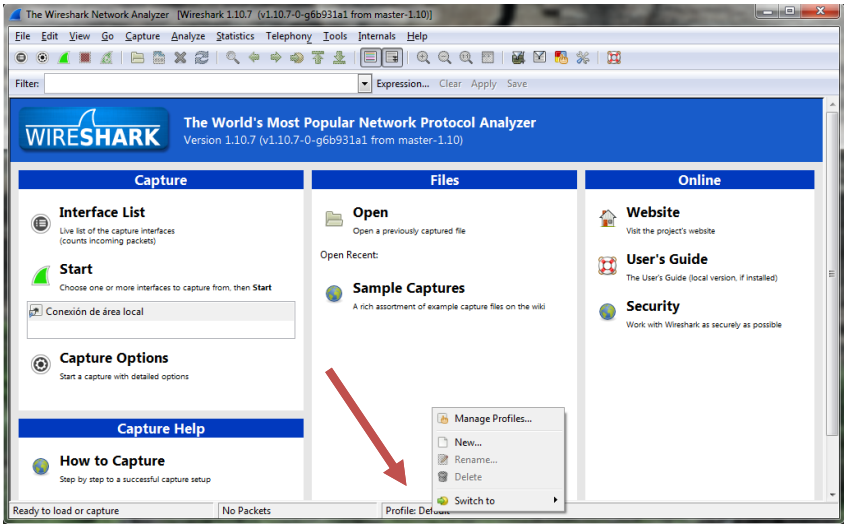

Figura 34: Wireshark, menú desplegable de perfiles

Se nos abrirá una nueva ventana donde deberemos seleccionar si es una configuración local o global y el nombre de la nueva configuración. Para poder realizar una exportación más sencilla deberemos elegir dentro del grupo global la plantilla clásica. Seguidamente deberemos darle el nombre que deseemos tal y como puede ver en la Figura 35. Ahora todas las configuraciones y personalizaciones que realizásemos sobre esta nueva plantilla se podrán exportar a otros equipos.

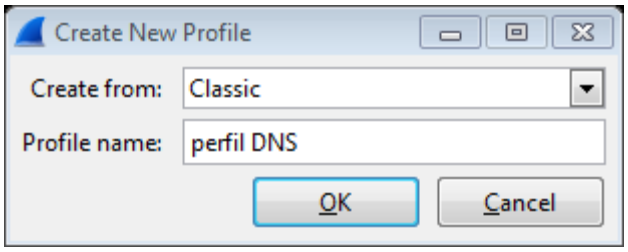

Figura 35: Wireshark, creación del perfil

3.17.2. Colorear Paquetes

Una buena utilidad es el que cambie el color a aquellos paquetes que cumplen cierta condición por ejemplo que sean de cierto protocolo o que cumplan ciertas condiciones como por ejemplo temporales, de error, etc. Vamos a realizar lo primero para que nos identifique aquellos que sean de tipo DNS con un color verde claro. Para realizar esta personalización, deberemos abrir el menú de reglas de colores a través del botón que hay disponible en la barra de herramientas (Figura 36).

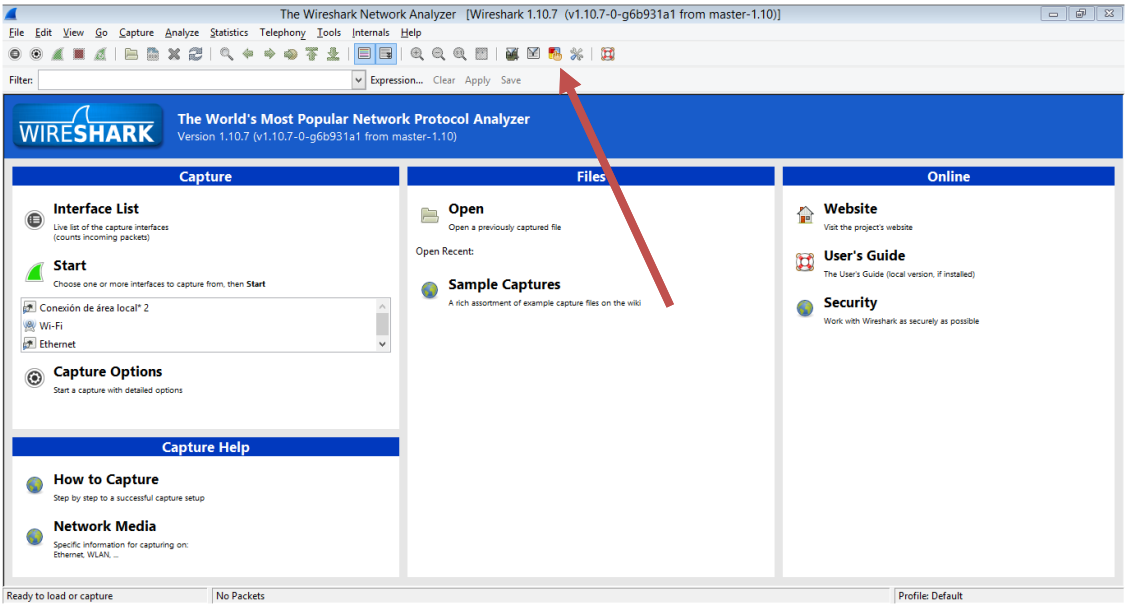

Figura 36: Wireshark, Botón del menú de reglas de colores

Una vez que se nos haya abierto la nueva ventana (Figura 37), podremos observar las reglas que hay definidas en el perfil actual. Entonces, pulsaremos el botón de crear un nuevo perfil.

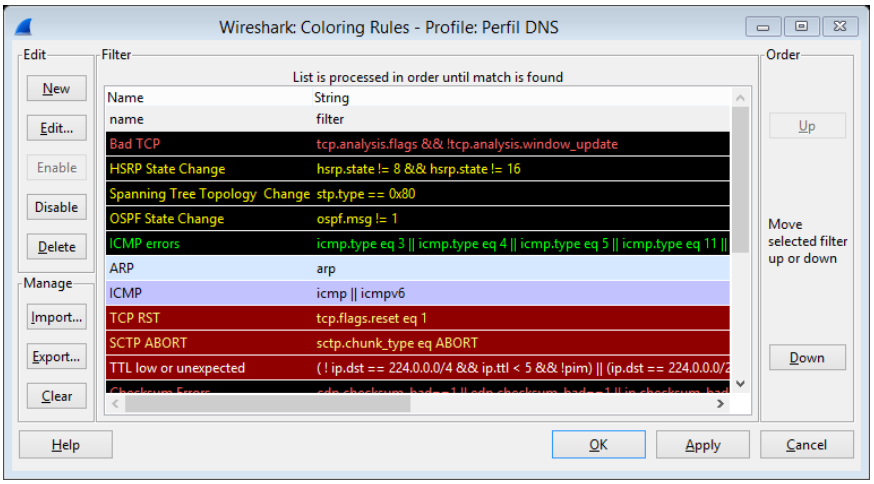

Figura 37: Wireshark, Gestion de reglas de colores

Una vez que se nos haya abierto la ventana (Figura 38) de creación/edición de reglas, definiremos el nombre de la regla, el filtro que deberá de cumplir y los colores para el texto y fondo de dicho paquete que mostrará en el listado de paquetes de la traza. En nuestro caso de ejemplo, definiremos la regla para que nos muestre de otro color aquellos paquetes que sean de tipo DNS con color verde claro.

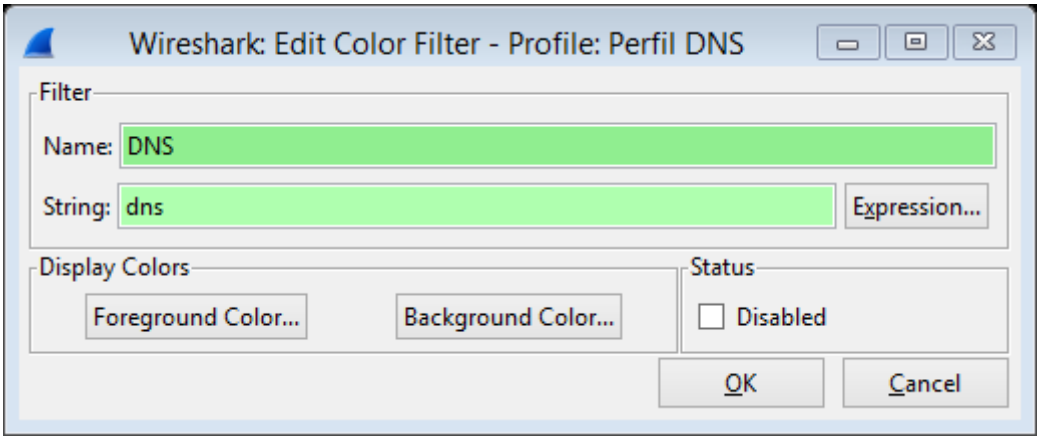

Figura 38: Wireshark, Creación de la regla de paquetes DNS

3.17.3. Creación de columnas

Muchas veces tenemos que analizar cierto campo de la cabecera de un protocolo y tenemos que estar mirando paquete a paquete para poder analizarlo. Este problema se puede resolver de una forma fácil además de que nos da la posibilidad de poder ordenar los paquetes por dicho campo.

Para conseguir esta ventaja se realiza en par de pasos muy sencillos, lo primero deberemos de seleccionar el paquete donde podemos ver dicho campo, y lo seleccionamos. Acto seguido desplegamos el menú secundario y pulsaremos la opción de añadir columna tal y como podemos ver en la Figura 39.

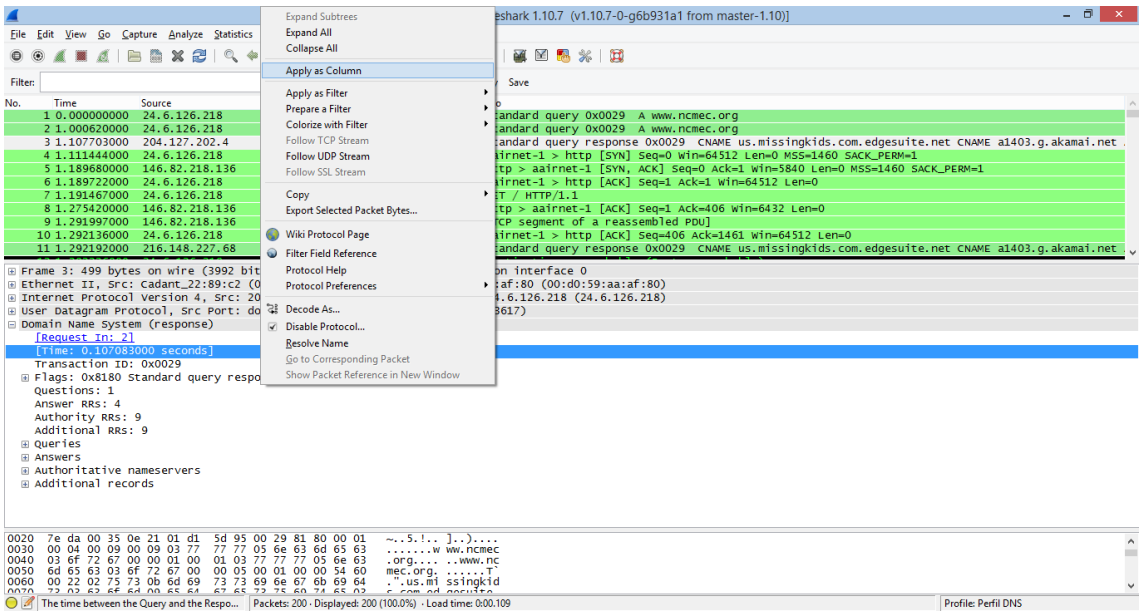

Figura 39: Wireshark, ejemplo de añadir columna

Como se observa en la Figura 40 se ha añadido una columna donde pone el tiempo entre la respuesta y la pregunta de los paquetes DNS. Pero el título de la columna no es muy orientativo, por tanto lo renombraremos a DNS Time como en nuestro ejemplo. Para lograrlo basta con que pulsemos con el botón derecho sobre la columna y seleccionemos editar detalles y reescribamos el nombre que deseamos. Como puedes ver, la columna es el filtro dns.time.

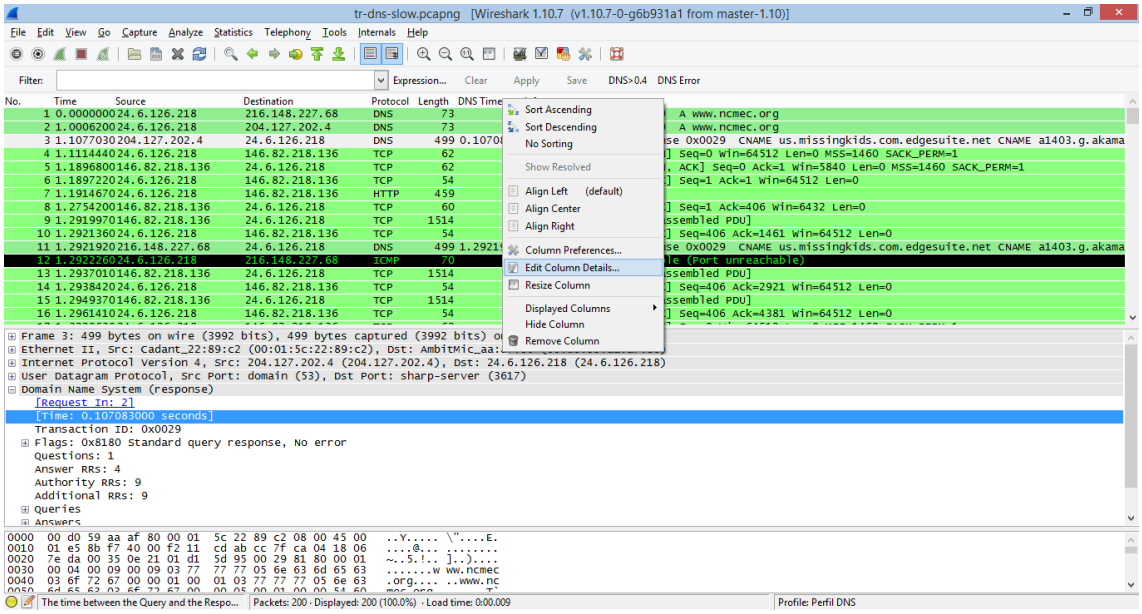

Figura 40: Wireshark, Editar el nombre de la columna

3.17.4. Creación de filtros predefinidos

En múltiples ocasiones nos ha sucedido que tenemos que escribir filtros varias veces para buscar fallos del mismo tipo en diferentes trazas. Para no tener que poner el mismo filtro una y otra vez, podemos crear accesos directos en la barra de herramientas para así con solo un clic, saber si hay paquetes que cumplen dicha condición.

Para lograrlo basta con que escribamos el filtro que deseamos como por ejemplo dns.time > 0.4 el cual estaríamos ante un problema de rendimiento con nuestro servicio de DNS y pulsamos en guardar en vez de aplicar. Esto nos abrirá una ventana (Figura 41) donde podemos ver nuestro filtro y deberemos escribir un nombre para recordar que es, como por ejemplos DNS>0.4.

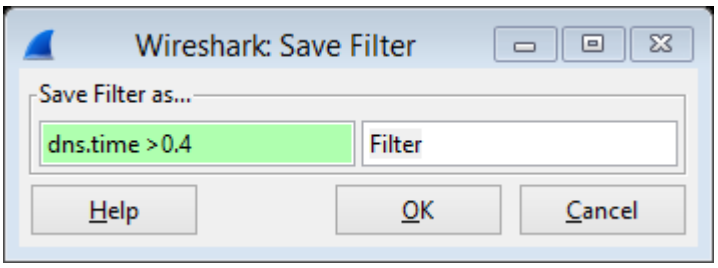

Figura 41: Wireshark, Creación filtros predefinidos

Este paso lo repetimos también para la detección de errores en la resolución de nombres con el filtro dns.flags.rcode >0, quedándonos la barra de herramientas (Figura 42) con los botones para aplicar los filtros que deseamos con un solo clic.

Figura 42: Wireshark, Barra de herramientas tras añadir los filtros deseados

3.17.5. Exportación e importación de perfiles

La realización de esta tarea es muy sencilla. En este caso se va a explicar sobre Windows pero es también realizable sobre otros sistemas operativos que pueden ejecutar wireshark. Para exportar basta con que este el wireshark cerrado, nos situamos sobre la carpeta C:\users\Tu nombre de usuario\AppData\Roaming\Wireshark\profiles\. En esta ruta encontraras una carpeta que se llama de igual forma que el perfil que has generado (en nuestro caso "Perfil DNS"), por lo que están sencillo de exportar como comprimir en Zip dicho directorio y descomprimirlo en la misma ubicación en otro pc. Después ejecutaremos wireshark y en el menú desplegable del botón secundario del ratón sobre profiles, seleccionaremos gestión de perfiles donde activaremos el perfil que deseamos o en el sub menú desplegable de "cambiar a" seleccionamos el que deseamos activar.

En el caso de importar el perfil explicado en este ejemplo basta con descomprimir el contenido de Zip en el directorio correspondiente siguiendo los pasos que se han comentado para activarlo.

4. Procedimientos generales de análisis de tráfico

Según el tipo de suceso se deben seguir unos pasos u otros para la detección rápida de la causa del mal funcionamiento reportado por el usuario y/o administrador de la red. Dada la gran variedad de casos vamos a dividirlo por análisis general y las variantes específicas de cada protocolo.

A lo largo de las siguientes secciones donde vamos a explicar unas pautas de cómo realizar los análisis. En algunos de los casos, existen ejemplos de problemas reales por lo que se referenciaran a ciertos análisis realizados con más detalle que se encuentran en los anexos de este documento.

4.1. Análisis general

Para cualquier tipo de análisis que debamos realizar, podemos seguir una serie de pasos con los que nos resultará confirmar los problemas y detectar otros más sin olvidarnos de la parte esencial, conocer un poco más el contexto al que vamos a analizar.

Los dos primeros pasos que deberemos realizar serán ejecutar el script-informes.sh para disponer de las estadísticas principales y, para complementar un poco más la información obtenida, ejecutaremos las herramientas de identifica redes que hemos comentado anteriormente para conocer un poco cómo es la topología de las comunicaciones que hay en el escenario.

Una vez realizado esos pasos, se revisarán las gráficas obtenidas por la herramienta de identifica redes para ver cómo de grande es la red en el punto donde estamos analizando, ya sea por número de hosts, equipos de red, cantidad de trafico cursado, número de flujos entre las diferentes redes vistas, etc. Una vez revisado las gráficas, revisaremos el informe principal devuelto por script-informes.sh para ver las estadísticas globales con las que nos permitirá identificar los equipos de interés complementándolo con revisiones en la traza original. Es muy útil para localizar servidores (por el puerto que emplean), puertas de enlace (MAC por la que se envía el tráfico hacia internet), routers (MACs por las que se envía a otras redes internas), switches (por la MACs de los grupos Multicast, tráfico de STP).

4.2. Problemas habituales en la tecnología de enlace

Muchos usuarios achacan sus problemas de red al ISP, al navegador, etc. Por lo que vamos a explicar cómo con una traza o medidas activas podemos deducir si es problema de algo explicado anteriormente o no. En el caso de que suceda que toda la navegación va lenta, comunicación de ficheros compartidos entre PCs dentro de la misma LAN y comunicación lenta con la WAN, puede ser por varios motivos:

- Una mala configuración de alguno de los protocolos de la pila TCP/IP
- Nodos saturados (equipos de red o extremos)
- Problema de la conectividad a dicha LAN.

Para determinarlo, al analizar la traza capturada, deberemos observar el RTT que tienen los paquetes hacia el exterior y entre equipos dentro de la misma LAN. En el caso de que se observen que en ambos se produzcan retardos muy elevados como de 100-200ms o más y de repente retardos de 1-4ms, deberemos prestar atención a si estamos conectados vía Wi-Fi u

otro medio compartido inalámbricamente o estamos conectados a una red cableada. Si se trata de una red inalámbrica se deberá verificar que otras redes, equipos,… están trabajando en la misma frecuencia, ya que se trata de contaminación inalámbrica provocando una navegación pésima. O que todos estén enviando/recibiendo datos produciendo una saturación en el punto de acceso (AP). En el caso de que esté conectado vía cable, deberemos buscar a través de los métodos explicados próximamente para detectar quien tiene un cuello de botella o una mala configuración que provoque una mala experiencia en la conectividad a la red.

- Protocolos de descubrimiento, ARP
- Bucles debidos a STP (trafico duplicado)
- Trafico broadcast
- Otros protocolos que se empleen en niveles superiores
- Control de congestión mediante mensajes Ethernet

Se realizará un análisis de los flujos entre MACs (realizado con MacConversations), para así conocer qué host son los que más datos intercambian, detectar los equipos que hay, los EtherTypes que emplea cada host, el número de paquetes y bytes transferidos por cada uno de ellos. Con ello realizaremos un procesado a nivel de enlace para comprobar si hay paquetes sospechosos como podrían ser un envenenamiento ARP (ejemplo en el informe de la Anexo 2, donde se produce un envío de un paquete ARP duplicado), sin resolución de la puerta de enlace.

Otro aspecto que deberemos de revisar es si hay paquetes de algún protocolo que haga algún bucle como podría ser una mala configuración de STP (Spanning Tree Protocol). Esto podría provocar que se saturasen ciertos caminos de la topología física provocando que los caminos lógicos se viesen gravemente afectados además de que podría no llegar a destino el paquete y quedarse en un bucle infinito. Esto lo identificaríamos fácilmente como tráfico duplicado en las trazas capturadas y que no fuese debido por una mala captura en el puerto SPAN del switch. Para detectarlo rápidamente, en el scrip-informes.sh (procesaConexiones e InfoDups) realizaran un análisis de la traza basándose en los campos que devuelve ambas herramientas. En el caso de InfoDups, nos avisará de la posición y la diferencia con el original en la traza del paquete duplicado, el tipo de duplicado (si es por sospecha, debido al switching, enrutado, NAT, Proxy, etc.), etc. Con el listado anterior nos permitirá poder pasárselo al procesaConexiones para que no tenga en cuenta dichos paquetes en las estadísticas que genera.

Otro aspecto que deberemos de revisar es el tráfico de broadcast, el cual si no se ha vigilado la configuración del STP y los multi-camino establecidos, podría producirse una inundación llegando incluso a bloquear la red con paquetes broadcast. Y por otro lado, deberemos de revisar la configuración que establecemos de reenvió de trafico de broadcast entre VLANs ya que podría provocar en todas inundaciones del paquete llegando a poder colapsar enlaces entre equipos al transportar varias VLANs.

Una vez conocido los hosts que intercambian datos en la red, deberemos revisar qué EtherType predomina en los flujos que intercambian para conocer qué tráfico de nivel de red emplean, y por tanto conocer qué debemos de revisar. Además podremos ver si es lógico que predomine o no ese tipo de tráfico. Un ejemplo de una red en la que haya una gran variedad de EtherTypes es el Anexo 2, en el que no provocan problemas de incompatibilidad entre ellos.

Además, revisaremos si se envían mensajes Ethernet (IEEE 802.3x Ethernet flow control) indicando pausa (pause frame) y el tiempo de espera para la reanudación de la transferencia, dado que algunos equipos de red envían este tipo de mensajes para que no se pierdan paquetes y realizar así un control de flujo. Este tipo de mensajes son enviados a la dirección Multicast 01:80:c2:00:00:01 que escuchan los equipos (salvo ciertas configuraciones). Este es un buen indicativo de saturación o casi saturación (dado que lo envían algunos antes de que se produzcan perdidas) de algún equipo de red.

Deberemos de revisar en el caso de que hay una configuración de VLANs (802.1Q) que podría provocar saturaciones en algún nodo en el caso de que no se haya extendido correctamente la VLAN por los equipos o se haya configurado de forma que hay un punto de riesgo en la red.

4.3. Problemática de IP

El siguiente análisis que se debe de hacer es a nivel de red, obteniendo estadísticas de qué tipos de paquetes se están transmitiendo como por ejemplo si son ipv4, si ya emplean ipv6, fragmentación, trafico Multicast, incompletos, etc. Todo lo que nos sea útil para detectar qué protocolos utilizan y si tienen problemas de compatibilidad entre ellos. Los aspectos que deberemos de revisaremos son los siguientes:

- Compatibilidad entre diferentes protocolos
- Configuración de sistemas de enrutamiento
- CRC de los paquetes IP
- Mensajes ICMP

Además de comprobar si hay alguna incompatibilidad entre ellos, ya que podría provocar una lentitud en servicios que pudiesen emplear diferentes niveles de red, como es el caso de IPv4 e IPv6 que podría provocar una lentitud o incluso quedarse sin acceso por una mala configuración de la doble pila de IPv4+IPv6.

Otro aspecto que deberemos de revisar son los protocolos de enrutamiento que haya disponibles como RIP, BGP, etc. Que con una configuración incorrecta podría provocar el envío de paquetes en forma de bucle hasta que se descartasen. Esto provocaría saturaciones en algunos nodos de la red además de provocar que los paquetes que se reenvían sin cesar nunca llegasen a destino. Una forma de detectarlo es observar los paquetes duplicados que varían únicamente el CRC y el TTL.

No nos podemos olvidar que debemos revisar en la medida de lo posible si el CRC de los paquetes está bien calculados ya que al producirse fragmentaciones y por otros motivos pueden provocar que el CRC no sea válido y según el firmware del nodo que lo detecte puede dejarlo pasar o desecharlo.

Otro aspecto interesante a revisar es el protocolo ICMP, que nos da información de tiempos entre equipos, problemas de acceso o restricción, redirecciones a un nuevo camino o incluso puede ser utilizado para realizar ataques MITM (Man In The Middle).

Una vez conocido el contexto actual mediante las revisiones de los informes de estas capas, deberemos revisar ya a nivel protocolo superior, a nivel de transporte y aplicación para averiguar cuál es el problema sino es por ninguno de los anteriores.

4.3.1. IP Spoofing e IP Duplicada

Para la detección de dicho problema, deberemos fijarnos en diferentes aspectos de la traza. Si intentásemos realizar este tipo de ataques conoceremos que se puede hacer mediante envenenamiento ARP, redireccionado mediante ICMP o suplantación DHCP. Estas son las técnicas más habituales.

Un indicio de que se está produciendo dicho suceso es por ejemplo, que en el tráfico de una conexión TCP se produzcan retransmisiones por exceder el TimeOut sin haber indicios previos como ACK duplicados. En el caso de disponer de acceso al equipo se observará alertas por parte del sistema avisando de disponer de una dirección IP que otro equipo la está usando.

El método más frecuente es mediante envenenamiento ARP por lo deberemos buscar si hay muchos ARPs gratuitos procedentes del mismo host para diferentes IPs. Este tipo de suceso lo puedes ver en la sección 3 “causas de la suplantación” del Anexo 2. Con herramientas tipo Wireshark, puedes realizar un filtrado de que haya uso de direcciones IP duplicadas en los anuncios de los paquetes ARPs. En el caso de que haya suplantación, este nos aparecerá. Una vez localizados, deberemos observar el comportamiento de dichos hosts, para ver si han hecho alguno de ellos el envenenamiento ARP o es debido por la asignación automática/manual de la IP.

En el caso de que no se deba a un envenenamiento ARP, deberemos observar los paquetes DHCP. Recordamos que en los filtros predefinidos de Wireshark, se corresponde con Bootp.x donde x lo podemos restringir a todos los DHCP (bootp.dhcp), a un tipo de DHCP en concreto, etc. El motivo de que DHCP sea un subfiltro de BOOTP es porque DHCP está basado en el protocolo de Bootp. Por eso se interpreta y se encuentra bajo dicho filtro en wireshark, tshark. Aquí observaremos cuantos servidores detectamos que hay en la red, y comprobaremos si han asignado las direcciones IPs a los hosts que tienen la misma para deducir si se trata de un problema de configuración o ataque de varios servidores DHCP o el problema radica en que se trata de un equipo que tiene dirección IP estática y la tiene incluida en el pool de los servidores. Además, en este caso es bueno revisar cómo actúan los servidores como el tiempo que asignan las direcciones IPs, tiempos de renovación para ver si los servidores están saturados o están sincronizados, Para comprobar si es debido por una mala configuración que provoque este tipo de problemas.

Si éste tampoco es el problema, entonces hay que revisar más a fondo la traza, y buscar si hay una elevada cantidad de ICMP de redirección que sería para desviar las comunicaciones y poder espiar el tráfico. Este tipo de comportamiento lo podemos averiguar fácilmente mediante filtrado del tipo de paquetes icmp.type eq 5. De esta forma si hay muchos paquetes de este tipo, deduciremos que dicho host es víctima de un MITM mediante esta técnica.

Si aun así, tampoco se detecta ninguna anomalía pero sabemos que somos víctimas de un MITM como por ejemplo en la navegación cifrada por recibir certificados no válidos, entonces el problema viene por realizar DNS Spoofing o cualquier de las anteriores técnicas pero desde

otra red (atravesando algún Router) ya que se requiere hacer uso de algunas de las técnicas comentadas para realizar Spoofing. Para el problema de detectar dirección IP duplicada se debe de poder detectar con algún método anterior, pero analizando más redes por donde viajan nuestras comunicaciones interceptadas.

4.4. Problemas habituales en TCP

Uno de los típicos problemas que pueden tener muchos usuarios es la lentitud con ciertos servicios que emplean TCP por debajo. Sea cual sea el problema, deberemos de revisar una serie de aspectos generales antes de buscar problemas en el comportamiento de ciertos tipos de servicios frecuentes. Los aspectos a revisar son los siguientes:

- Retransmisiones
- Tamaño ventana de congestión y recepción
- Micro pausas en el crecimiento del número de secuencia
- Velocidad de crecimiento del número de secuencia
- RTT
- Flujos vecinos que puedan interferir
- Protocolos inferiores y el MSS disponible

Lo primero que deberemos revisar es el número de retransmisiones con el script informes.sh (se realiza con procesaConexiones con la columna de número de segmentos y bytes retransmitidos en cada sentido) para detectar si hay pérdidas ocasionadas por una elevada cantidad de tráfico en algún nodo o se deben al comportamiento nativo de TCP para el crecimiento de la ventana de congestión. Una elevada tasa de pérdidas provoca una lentitud en la navegación por el control de recuperación de las pérdidas. Además, podría provocar una pérdida de conectividad según la frecuencia de las pérdidas que tuviese la red.

Si no son por pérdidas de paquetes, debemos revisar si se produce limitación por ventanas de recepción en alguno de los sentidos para saber si ambos pueden procesar la cantidad de datos que reciben del otro extremo. En dicho caso habrá que redimensionar los correspondientes equipos para que tengan la capacidad necesaria.

En el caso de que no se trate de ninguno de los casos anteriores, debemos revisar el crecimiento del número de secuencia y ventana ya que podría provocar que estuviese limitada la velocidad por una mala configuración de la conexión TCP y no creciese la ventana de congestión de red. Que por la aplicación en cuestión no disponga de más datos para enviar y por tanto actúe de dicha forma (ejemplo, una aplicación interactiva como el telnet). O que el servidor por configuración de reparto de recursos del SO no tenga la capacidad suficiente para enviar una elevada cantidad de datos a todos los clientes simultáneamente provocando que a todos se les fuera enviando lentamente y por tanto teniendo a todos muy insatisfechos con el servicio ofrecido.

Si el servidor está saturado lo que podríamos detectar en el crecimiento del número de secuencia son unas micropausas que podrían convertirse hasta en pausas prolongadas provocando retransmisiones sin ser necesarias provocando reducciones en el tamaño de la ventana de control de congestión. Este suceso lo podemos ver en la Figura 45**¡Error! No se encuentra el origen de la referencia.** del Anexo 1 en el que se puede apreciar como el

crecimiento del número de secuencia no tiene un crecimiento a tasa constante sino que se producen paradas provocando que la velocidad de transferencia en media sea constante y baja. Para analizar este problema basta con ejecutar el script informes que se encarga de utilizar la herramienta de procesaConexiones y un postprocesado de forma que obtiene las estadísticas que necesitamos para analizar la problemática de las micro-pausas. El postprocesado que hace consiste en coger los datos devueltos de los campos (75–78) de tiempo máximo que ha estado bloqueado en cada sentido y cuando ha sucedido que nos devuelve el procesaConexiones y de esta forma, realizando un ordenación por estas columnas podemos conocer que conexión TCP han estado más tiempo bloqueadas. Para la obtención de las gráficas podemos emplear la herramienta de TCPTrace y Xplot o la nTCPTrace que diseñaremos más adelante. Deberemos de tener especial atención que tipo de aplicación soporta debido a que puede variar su comportamiento.

Otro aspecto frecuente que deberemos de analizar es el crecimiento del número de secuencia respecto al tiempo. Este nos resultará útil para deducir si existe crecimiento de la ventana o está bloqueada a un valor máximo por no disponer de más buffer el receptor (se puede ver ejemplos en los informes de la Anexo 1 (Figura 45) y Anexo 6 (en la sección 3: análisis de navegación lenta)). Este problema es muy frecuente por malas configuraciones y/o bajos recursos de hardware en el equipo que provoque una transmisión lenta y por tanto un usuario insatisfecho con el servicio que le proporciona.

En las conexiones de TCP también debemos tener en cuenta el RTT de las conexiones ya que cuanto más lejos esté el servidor de nosotros, más tiempo tarda en recibir las respuestas además de que el crecimiento de la ventana es más lento. Una vez tengamos una referencia de cuanto es el RTT, deberemos de observar como fluctúa el RTT de la conexión. Este es un factor importante ya que nos podrá ayudar a identificar si hay pérdidas en algún punto de la red y/o algunos de los extremos de la conexión TCP estén ocupados y por tanto tarde más en procesar los paquetes que recibe. Para analizar el RTT emplearemos las herramientas de procesaConexiones en conjunto con el script informes para obtener los resultados ordenados por el RTT mayor. Además podemos utilizar TCPTrace y Xplot para obtener graficas del RTT y ver como fluctúa a lo largo de la conexión. En el caso de que necesitemos concretar más información de cuando se produce los incrementos mayores de RTT y los datos que transmiten en dicho momento, emplearemos Wireshark para obtener más detalles. Puedes ver un ejemplo relacionado con el RTT en la Anexo 5, en la sección 3 de análisis de problemas con el proxy.

Deberemos revisar si hay algún otro tipo de servicio ya sea sobre TCP o en otro protocolo de nivel de transporte que pueda interferir introduciendo retardos o incluso perdidas por saturar el ancho de banda disponible o provocada por reglas de QoS. Un ejemplo típico es el uso de P2P que provoque saturaciones en uno o ambos sentidos y por tanto un mal funcionamiento del resto de servicios. Para revisar esta problemática, deberemos de exportar (tshark, editcap u otros disponibles) el fragmento temporal de la traza donde deseemos analizar con el script informes, o analizarlo mediante filtros en el wireshark.

Lo siguiente que deberemos de comprobar es vía que nivel de red está empleando para realizar las peticiones ya que puede estar empleando túneles de IPv6 sobre IPv4 (ejemplo sobre IPv6 y problemas de MTU en la tabla 8 del Anexo 10) que provoque un mal servicio según la

configuración empleada. En el caso de que no haya pérdidas, el RTT será normal salvo que haya algún problema de los comentados anteriormente. La forma más rápida de analizar las capas de protocolos que emplea es con el wireshark.

En el caso de que no se haya encontrado el problema causante, deberemos revisar el comportamiento ya a nivel de aplicación y su método de intercambio de datos para analizarlo de forma correcta.

4.4.1. Problemática web

Un problema frecuente que suelen decir bastantes usuarios es que navegan por la web de forma muy lenta. Este "problema" puede ser cierto o no, ya que tiene una parte subjetiva en el detectar dicho problema.

Lo primero que deberemos de revisar es la resolución de DNS para saber si el problema viene provocado por este otro servicio. Para revisar este servicio, deberemos fijarnos en el tiempo de respuesta a la petición generada. El tiempo que debería de tardar no debe ser muy alto, como mucho 2RTT a la dirección destino o en media inferior a 150ms (destino intercontinental). Ya que valores más altos provocan que tarde mucho tiempo en conectarse a los servidores de las páginas web.

Otro aspecto que debemos revisar en el servicio de DNS, y más esencial si cabe, es si encuentra el servidor destino. Así se puede deducir si es problema de que el servidor DNS no puede encontrar el servidor, esta censurado o la dirección consultada es incorrecta.

Se hará un análisis de cuanto ha durado, tráfico de datos, nº paquetes, retransmisiones o pérdidas por cada conexión para detectar si ha habido problemas de dicho tipo que ocasionaría una lentitud en la navegación. En resumen, realizar un análisis general de la traza tal y como hemos comentado anteriormente.

En la navegación web, se basa la conexión en petición-repuesta a la velocidad máxima que soporte la red, por lo que deberemos revisar si el tráfico que se cursa lo está haciendo a la velocidad máxima o está enviando paquetes de menor tamaño que el MSS que soporta. Este tipo de problemas los podemos ver en la tabla 3**¡Error! No se encuentra el origen de la referencia.** del Anexo 1 y en la tabla 8 del Anexo 10 donde podemos ver qué produce un retardo adicional por el envío de paquetes de menor tamaño de lo habitual. En el caso de que se envíen paquetes de un MSS inferior, se puede deber a que el servidor o cliente esté muy ocupado y necesita mejorar el hardware que dispone para que pueda asignar más recursos a cada conexión o que no disponga de más datos que enviar. Para la revisión de esta problemática lo podemos hacer mediante wireshark o el script informes.sh que emplea un postprocesado sobre los resultados arrojados por procesaConexiones.

En el caso de que no se haya detectado ningún problema con los anteriores deberemos realizar un seguimiento de las peticiones de las páginas web para revisar los siguientes comportamientos:

- conexiones persistentes
- emplean autentificación
- devuelve errores http

- emplean pipeline
- navega a través de un proxy
- emplean una capa SSL adicional.

Un ejemplo donde podemos apreciar diferentes problemas como la autentificación, errores en la navegación http o una mala configuración del keepalive es en la sección 3 de la Anexo 5.

Aquí es cuando deberemos revisar si aparecen tiempos de bloqueos ilógicos como puede ser entre handsake de autentificación (proceso en el que el navegador pide la página, el servidor responde con una pregunta de autentificación y el cliente tiene que responder con el usuario y contraseña de forma que le permita o no el paso el servidor) y las peticiones, o paradas entre peticiones seguidas de diferentes recursos excesivas. Y sin olvidarnos que haga paradas en la entrega de un mismo fichero. No deberemos de confundir con los tiempos de bloqueo del tipo del acceso a la BBDD y consulta a otros Web Services que introducen retardos y bloqueos adicionales los cuales serían lógicos. Este problema de introducir retardos por realizar consultas a otros servidores lo podemos apreciar en la sección 3 de la Anexo 3 en el que se explica que los retardos ocasionados son debidos a una parte del retardo introducido por la consulta de WebServices.

4.4.2. Problemática Citrix

Muchas empresas utilizan los servicios de citrix [12] para así poder dotar de movilidad a sus empleados para disponer siempre de su escritorio con su configuración. Este tipo de servicios está orientado más a nivel interno de la LAN pero también se pueden dar casos de usar vía internet empleando VPNs para su protección y acceso. Para ver un ejemplo de problemas con este tipo de servicios puede revisar el informe del Anexo 5.

Este tipo de servicios requiere de una alta interacción y transferencia de datos entre el servidor y el cliente, por lo que en una red LAN no debería de tener problemas dado que las velocidades suelen ser como mínimo de 100Mbps y RTT inferiores a 1ms. Pero puede darse el caso de que vaya lento; en esos casos, la metodología para encontrar el problema es la siguiente.

Lo primero que se debe verificar si los clientes están en la misma LAN o están enrutados por equipos dentro de la empresa o están conectados vía Internet empleando algún protocolo de VPNs. Una vez que conozcamos como se realiza la conexión, podremos descartar problemas de eficiencia de excesos de cabeceras, una baja velocidad de la comunicación entre ellos o un RTT elevado que puedan ocasionar los diferentes métodos de comunicación.

Lo siguiente a revisar son las retransmisiones y pérdidas que se produzcan en la comunicación. Debemos revisar antes si existen paquetes duplicados ya que provocaría confusiones en el análisis. En el caso de que haya retransmisiones deberíamos averiguar cuál es la causa para evitarlas, como equipos de red saturados, equipos (servidor o cliente) saturados y no puedan responder tan rápidos o por la metodología de priorización del sistema operativo para atender las conexiones y procesos.

En el caso de que no se haya detectado todavía la causa, se deberá hacer un análisis profundo de TCP para buscar cual es la causa que lo está provocando.

5. Mejora en la visualización de TCPTrace

Dada la gran cantidad de gráficas que debemos realizar manualmente ya que se carece de herramientas que las hagan de forma automática o carecen de un aspecto adecuado para la presentación en documentos, vamos a contar el desarrollo de una herramienta para la conversión de las gráficas de TCPTrace a Gnuplot para poder exportarlas después a otros formatos como PNG.

Para la realización de esta herramienta, deberemos establecer una serie de pasos para realizar una herramienta completa. Para ello, deberemos dividir el trabajo de conversión en diferentes etapas, las cuales serán:

- Verificación
- Obtención de la gráfica deseada
- Conversión de la gráfica al formato gnuplot para poder exportar
- Edición/personalización de la grafica
- Dibujado de la gráfica en formato PNG

Antes de nada, deberemos definir qué lenguaje vamos a emplear para programar ya que nos definirá como debemos trabajar para cada etapa. Dado que la mayoría de herramientas las debemos de ejecutar por líneas de comandos, lo realizaremos todo en bash y así ser lo más simples y eficientes posibles.

Antes de comenzar con el diseño de la aplicación, tenemos que definir los parámetros de entrada y salida de la aplicación. Para la entrada tendremos:

ntcptrace.sh PATH_TRAZA PATH_SALIDA [C R S<x> [Nº_stream_TCP [<rango inicial> <rango final>]]

- PATH_traza: ruta donde se encuentra alojada la traza.
- PATH_salida: ruta donde se generarán los ficheros temporales y de salida
- C/S: dibujar o no dibujar en la gráfica la ventana del buffer de recepción del host destino
- R/A: Especificar el eje temporal en relativo o en tiempo absoluto
- S<x> x ∈ [0 , 3]: Definir 4 estilos diferentes de representar él envió de datos
- Nº Stream: especificar el flujo que se desea analizar
- [x1, x2]: Especificar el rango temporal que se desea visualizar

Lo siguiente que tenemos que pensar es lo que deseamos que nos devuelve el programa. El objetivo de la aplicación es que nos cree las gráficas en formato PNG para así poder visualizarla o añadirla a un documento de forma fácil y sencilla. Por tanto, lo que generará serán tantas graficas en formato PNG como las 2 graficas de la conexión que hayamos dicho que analice o todas las gráficas de todas las conexiones que haya en la traza.

5.1. Verificación

En esta primera etapa, deberemos comprobar con una serie de instrucciones que los argumentos que se nos han pasado son los correctos o para indicar cómo se deben pasar. Este paso es muy importante para saber si existe el fichero de la traza que nos han pasado y si existe

la carpeta donde pretenden que escribamos los resultados. Esta comprobación la realizaremos con unos sencillos pasos gracias a las opciones del bash y del uso de condicionales para realizar estas pruebas.

Para la verificación de los argumentos pasados, deberemos tener en cuenta las posibles combinaciones de los argumentos que nos den para no devolver error al usuario ni realizar el procesado de los datos sin tener todos los datos completos. Para ello utilizaremos el método de contar el número de argumentos que se nos han pasado para comprobar con las combinaciones que vamos a soportar.

Para la comprobación de la existencia de los ficheros emplearemos la opción -f que nos devuelve true en el caso de que el fichero indicado existe o falso en el caso contrario. Para la comprobación de la existencia de carpetas emplearemos la opción –d. [9]

5.2. Obtención de gráficas

Lo siguiente que debemos programar, es la generación de la gráfica que deseamos, que en este caso lo que deseamos se trata de una herramienta de conversión de gráficas de número de secuencia que genera TCPTrace en formato Xplot a gnuplot que nos lo exportará al formato PNG útil para la presentación de informes. Por lo que deberemos de hacer es que genere con TCPTrace las gráficas que deseamos en formato Xplot.

Para ello aportaremos los argumentos necesarios a la aplicación y exportaremos la información que nos devuelve a un fichero con la opción –l de TCPTrace para así tener información extra de las conexiones que vamos a analizar. Deberemos de pasarle también en el caso de disponer de dicha información el número de conexión que debe de realizar el graficado.

5.3. Conversión xpl a gpl y edición

El siguiente paso que tenemos que realizar es la obtención de los nombres de los ficheros que tenemos que convertir de un formato al otro. Para lograrlo de una forma muy rápida es buscando en el directorio actual aquellos ficheros que tengan el patrón de terminar con extensión xpl. Este sistema de búsqueda de ficheros puede ser visto como muy peligroso dado que si encuentra ficheros con el nombre que está buscando intentará procesarlos además de que borrará aquellos que estén dentro de los nombres que emplee el programa. El motivo de usar este método es debido a que desconocemos los nombres de los ficheros que genera TCPTrace y el parsear el fichero de información llevaría bastante tiempo, por lo que resulta más fácil el trabajar en un directorio que defina el usuario donde trabaje sin problemas.

Una vez que ya conocemos los nombres de los ficheros, tenemos que llamar a una función que se encargue de realizar la conversión. Esta función lo que realizará es la exportación en series diferenciando color y tipo de serie adaptándolo a la estructura de espacios que empleará gnuplot para el reconocimiento de los datos. Además, implementaremos un sistema de recordatorio del valor mínimo para poder hacer ejes relativos o absolutos.

Finalmente, se realizará el montaje de las instrucciones finales en el fichero gpl para dejar listo la conversión, con su correspondiente leyenda, colores e iconos que habremos especificado en el código para que lo genere automáticamente. Con el mismo sistema que antes

de reconocimiento de existencia de directorio, lo utilizaremos para identificar qué series existen y cuáles no, para evitar errores y gráficas con leyendas extensas.

Finalmente una vez realizado este paso deberemos llamar a este programa tantas veces como ficheros dispongamos en el directorio generados por TCPTrace.

5.4. Dibujado y función de zoom

El último paso que nos queda de realizar es volver a realizar una indexación de los ficheros que disponemos con extensión gpl de forma que llamemos a gnuplot con cada fichero para que nos genere las gráficas que deseamos.

Una ventaja que tiene la interfaz de Xplot es la posibilidad de poder realizar un zoom en determinado zona. Esto a priori no se puede conseguir en imágenes en PNG, pero sí que podemos obtener un zoom conociendo el rango temporal donde deseamos realizarlo.

Para lograr este objetivo, existen varias formas de diseñarlo, como puede ser las opciones de xrange de gnuplot, guardando el contenido deseado, un hibrido de ambos o modificar la traza. Según el sistema que escojamos puede acarrear unos inconvenientes u otros en la representación de los ejes o su contenido, por lo que se opta por una mezcla de los dos primeros, cortando el contenido de forma que comience donde deseamos pero que lo que muestre de duración temporal lo especifiquemos con xrange, para así solventar los problemas estéticos que acarrea la representación de datos recortados pero en la menor medida posible.

5.5. Ventajas

El resultado de la herramienta es que obtenemos una gráfica similar a TCPTrace con la diferencia de que está adaptado al crecimiento del número de secuencia y generación de las gráficas en formato PNG del número de secuencia que podemos analizar con la herramienta TCPTrace. Además, se ha añadido la opción de poder delimitar el eje x para especificar en una zona temporal que deseas analizar y así poder hacer zoom que de forma genérica con gnuplot no soporta.

La mejor forma de apreciar las ventajas es realizar una comparación grafica como las que podemos ver en las Figura 43 la versión clásica y en la Figura 44 la versión que acabamos de programar. A simple vista, podemos apreciar como el fondo es diferente siendo blanco más agradable en la presentación de documentación oficial. Otra diferencia son los ejes que podemos configurar como en este caso que sea relativos, lo cual resulta más cómodo de interpretar. Otra diferencia apreciable es la leyenda, la cual en la versión clásica debemos deducirla que por el contrario, en la versión nueva ya nos viene y de forma dinámica.

En resumen, logramos crear graficas del número de secuencia de las conexiones de una forma más rápida, automatizada, en el formato deseado y con una estética más limpia, clara y sencilla.

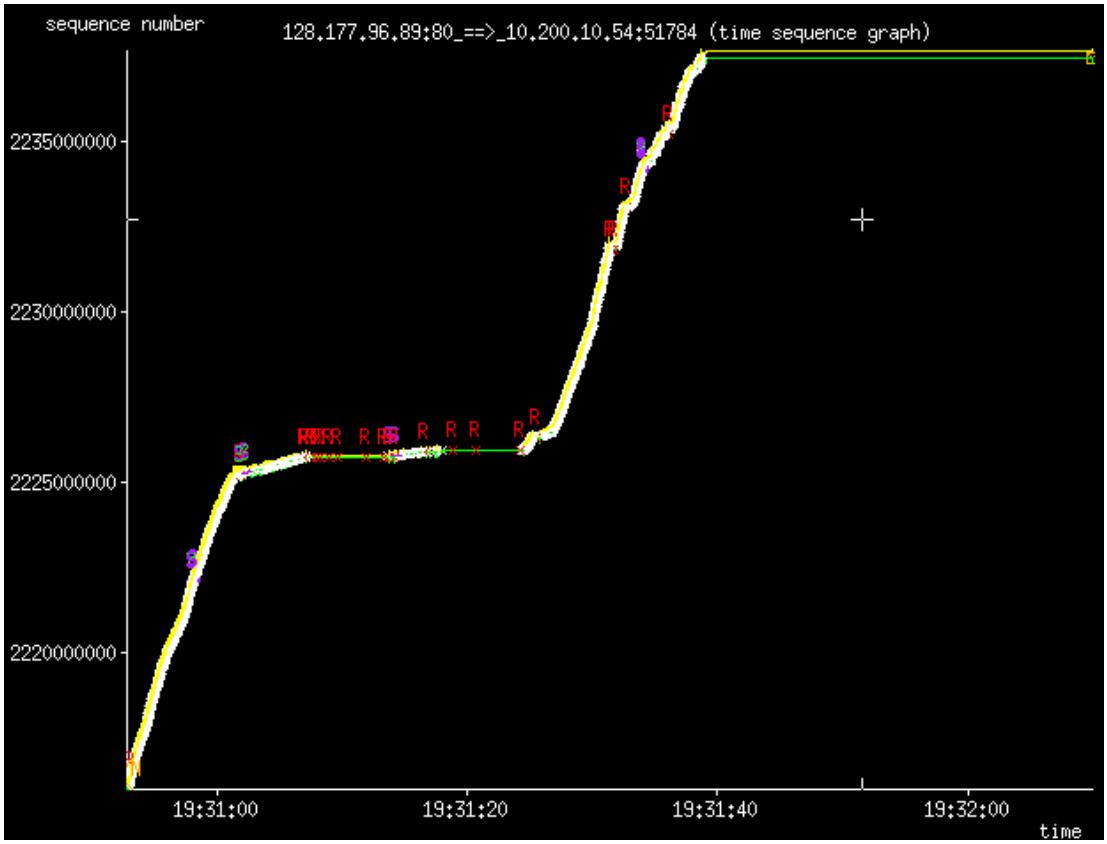

Figura 43: TCPTrace original

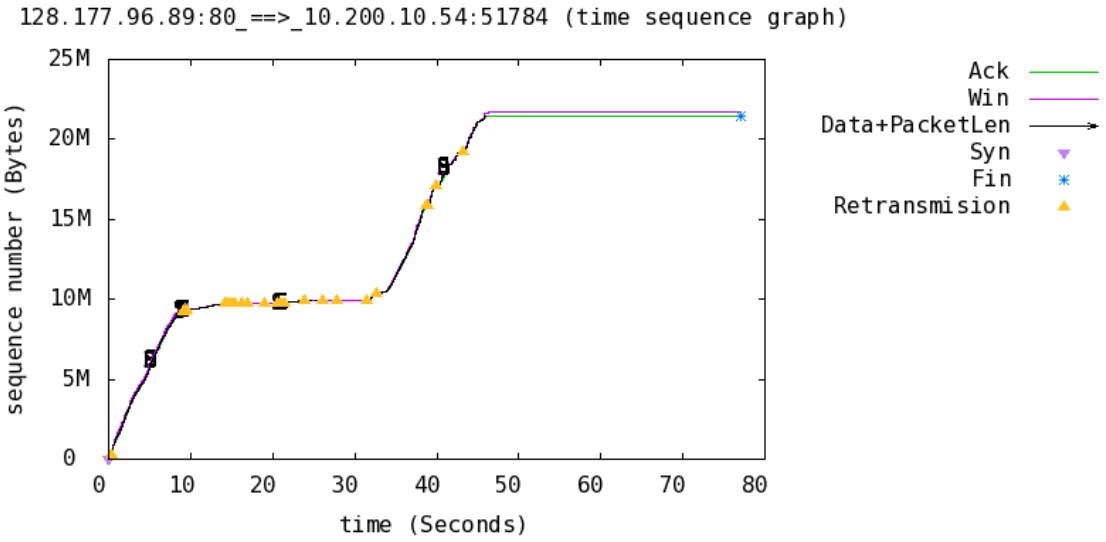

Figura 44: NTCPTrace rediseñado

6. Conclusions

Finally, it is time to review step by step all tasks that we have explained, analyzed and developed in this project. The traffic analysis is a complex task that needs not only a large set of tools but also, and even more important, a person with high level of knowledge about network protocols. As we have seen above, the state of the art in tools is very basic, there are a lot of applications that create general statistics about main protocols and some of tools deep a little more in protocols for Internet services. All tools have their limitations more or less. Therefore we have developed our own applications or scripts to automate some tasks. Mainly we have developed a script to graph TCP sequence numbers from the TCPTrace output and we have realized that the developing of tools require a lot of time even to make a very basic application. In addition, there is a simpler reason to develop applications, there are multitude of protocols, connectivity problems, etc. And all present tools do not support them, so we have to develop tools to be able to analyze the protocol problems that are not supported by the rest of current tools

Switching to the problematic of business and users that have day by day problems of network connectivity, for example: in private applications, connections between hosts in the same LAN, etc. The output generated by the tools requires an interpretation of conditions of the hosts trying to share information. In addition, we need to deduce if the hosts are busy and perform a deeper analysis of communication of the corresponding application.

We have explained the methodology of our traffic analysis. It allows to detect what the problems are in the network that we have to analyze. All of these without forgetting to develop tools that simplify the typical tasks of analysis. This world is always updating protocols, applications or services. So, it is very difficult to find tools that generate the solution of the problems of a traffic trace analysis. Always it would require a human analyst to get results with significate enough.

Finally, because there are not tools to analyze all network scenarios, this project does not finish here. There are multitude of branches through which we can continue as future work like for example developing new tools, improving other available tools or proposing new methods to analyze other network problematics.

Bibliografía

[1] Información obtenida de la ayuda de CapInfo, autor Wireshark, http://www.wireshark.org/docs/man-pages/capinfos.html

[2] Información más detallada a través de la ayuda de editcap, autor Wireshark, http://www.wireshark.org/docs/man-pages/editcap.html

[3] Articulo web donde explica cómo utilizar todas las funciones de Awk, autor Javier Peces, BULMA, en el portal http://www.linux-es.org/node/31

[4] Blog donde hay ejemplos de uso del programa Sort de Linux, autor Esteban M. Navas Martín, en el blog http://enavas.blogspot.com.es/2008/03/el-shell-de-linux-comando-sort.html

[5] Información y herramienta disponible en el repositorio SVN de Naudit, autor Naudit

[6] Guías de uso de herramientas en el blog de Seguridad y redes, disponible en: https://seguridadyredes.wordpress.com/category/tcptrace/

[7] Software Xming, configuración, autor,… en http://www.straightrunning.com/XmingNotes/

[8] Uso de condicionales en bash: http://www.linux-es.org/node/238

[9] Comprobación de existencia de ficheros en bash: http://itico.wordpress.com/2007/02/22/comprobar-si-existe-un-fichero/

[10] Guía de uso de wireshark y ejemplos de análisis de algunos protocolos: http://www.riverbed.com

[11] RRDTool, Tobias Oetiker: http://oss.oetiker.ch/rrdtool/

[12] Citrix http://www.citrix.es/

[13] Laura A. Chappeell, “Troubleshooting with Wireshark: Locate the Source of Performance Problems”, Ed. Laura Chappell University, 2014, ISBN-10: 1893939979.

[14] Wiki de análisis forense: http://www.forensicswiki.org/wiki/Network_forensics

[15] Network Miner, Netresec: http://www.netresec.com/?page=NetworkMiner

[16] Data Echo, soleranetworks: http://sourceforge.net/projects/data-echo/

[17] ChaosReader, Brendan Gregg: http://chaosreader.sourceforge.net/

[18] Argus, QoSient LLC: http://qosient.com/argus/

[19] Xplico, Gianluca Costa & Andrea De Franceschi: http://www.xplico.org/

[20] Justniffer, Plecno s.r.l.: http://justniffer.sourceforge.net/

[21] CapAnalysis, Evolka: http://www.capanalysis.net/site/

[22] Ntop, Luca Deri and others: http://www.ntop.org/

[23] NetworkTimeout, autor ExtraHop Network Inc. : http://www.networktimeout.com/

[24] Moloch, Aol: https://github.com/aol/moloch

[25] Other guide of Moloch, Alejandro Nolla: http://blog.alejandronolla.com/2013/04/06/moloch-capturing-and-indexing-network-traffic-in-realtime/

[26]Service of the traces analysis of Naudit HPCN: http://expertnetworkanalysis.naudit.es/

[27]Elasticsearch, Elasticsearch BV: http://www.elasticsearch.org/

Anexos

Anexo 1

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 2

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 3

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 4

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 5

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 6

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 7

Ref.: 1396362206

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 8

Ref.: 1396602865

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 9

Ref.: 1397202600

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.

Anexo 10

Ref.: 1392202395

Los anexos de este documento se han eliminado debido a que contienen información confidencial para satisfacer la ley de protección de datos (LPD). Si desea obtener más información, puede ponerse en contacto con el autor o el tutor de este documento.