



第8單元

網路鑑識

單元學習目標

- 建立學員對網路數位跡證(Artifacts) 類型的基本知識與網路鑑識的基本認知
- 掌握MITRE D3FEND的網路流量分析內容
- 了解網路鑑識核心技術(包含Wireshark 與Network Miner等)的運用
- 掌握網路鑑識核心工作內容(包含擷取流量中的資料、解密加密流量等)

網路鑑識(Network Forensics)

- 網路鑑識(Network Forensics)著重於網路跡證(Artifacts)的鑑識分析
- 網路跡證(Artifacts)主要有
 - Pcap：完整且龐大的網路封包
 - Flow：大流量情境的分析常使用
 - Log(包含各種類型的日誌)

MITRE D3FEND網路流量分析(1/5)

- MITRE D3FEND網路流量分析(Network Traffic Analysis)主題

1. 管理網路活動分析(Administrative Network Activity Analysis)
2. 位元組序列檢視(Byte Sequence Emulation)
3. 憑證分析(Certificate Analysis)
 - 主動式憑證分析(Active Certificate Analysis)
 - 被動式憑證分析(Passive Certificate Analysis)
4. 客戶端-伺服器負載分析(Client-server Payload Profiling)
5. 連線嘗試分析(Connection Attempt Analysis)
6. DNS流量分析(DNS Traffic Analysis)
7. 網路檔案雕刻(File Carving)
8. 流入會話量體分析(Inbound Session Volume Analysis)

MITRE D3FEND網路流量分析(2/5)

- MITRE D3FEND網路流量分析(Network Traffic Analysis)主題

9. IPC流量分析(IPC Traffic Analysis)
10. 網路流量社群偏差(Network Traffic Community Deviation)
11. 每主機下載上傳比率分析(Per Host Download-Upload Ratio Analysis)
12. 協定元資料異常偵測(Protocol Metadata Anomaly Detection)
13. 中繼模式分析(Relay Pattern Analysis)
14. 遠端終端會話偵測(Remote Terminal Session Detection)
15. RPC 流量分析(RPC Traffic Analysis)

MITRE D3FEND網路流量分析(3/5)

- D3-FC 網路檔案雕刻(File Carving)

- 定義：透過使用網路串流重組(reassembly)軟體從網路應用協定中識別與提取在流量中的檔案
- 運作原理
 - 協定串流重組軟體(Protocol stream reassembly software)透過分析擷取的網路資料封包重新建立定向位元組流(directional byte stream)
 - 一旦串流被重組，就會應用模式匹配(pattern matching)來確定它是否包含感興趣的檔案
 - 感興趣的文件包含可執行檔、 archive文件或文件檔案
 - 擷取文件後，將使用標準文件分析技術進行處理
 - 範例網路協定包含 HTTP、SMTP、FTP、HTTP/2 及 TLS/HTTP/Dropbox
- 範例工具：NetworkMiner可自動化進行File Carving

MITRE D3FEND網路流量分析(4/5)

● D3-RPA:Relay Pattern Analysis

– 定義：偵測內部主機在內部網路與外部網路之間的中繼流量

– 運作原理

- 中繼(Relay)可能使用各種代理、轉送或路由技術來橋接受保護網路與外部網路
- 用來偵測中繼網路的防禦分析可以透過比較多個主機之間的網路會話
- 幾乎相似的網路統計資料的主機可能是中繼網路的一部分
- 統計資料可以包含發送與接收的位元組數、會話發起時間、封包大小或封包到達時間資料

– 挑戰與注意事項

- 複雜的內部網路 VPN 或路由封裝可能會影響偵測分析
- 此外，不需要的封包可能不會被轉發，並且可能會在中繼處添加額外的封包，從而使檢測更加複雜

MITRE D3FEND網路流量分析(5/5)

● D3-BSE: Byte Sequence Emulation

- 定義：分析位元組序列(Byte Sequence)並確定它們是否可能是某種惡意 shellcode。(同義詞：Shellcode 傳輸偵測)
- 運作原理
 - 對位元組(視為機器碼指令或組合語言)進行分析，並比對已知 shellcode 的常見元件的指令，例如堆疊樞軸(stack pivots)、讀取記憶體位址表(Memory Address Table)以及禁用保護或執行程式碼的函數之系統呼叫(system calls)
 - 如果位元組序列包含與惡意 shellcode 中使用的序列類似的序列，則整個位元組序列將被標記，並且可以呼叫後續技術來處理
 - 原理類似IDS/IPS使用signature(特徵碼)進行偵測
 - 同樣會有False Negatives與False Positives問題



網路鑑識常用工具(1/2)

- Windows作業系統上常用網路鑑識工具
 - Wireshark/Tshark(詳見後續分析)
 - NetworkMiner(詳見後續分析)
 - WinDump(Windows版Tcpdump，已不在支援開發)
 - Npcap (<https://npcap.com/>)
- 線上PCAP分析工具
 - packettotal.com
 - apackets.com
- 更多工具請參閱
 - <https://github.com/caesar0301/awesome-pcaptools>



網路鑑識常用工具(2/2)

- Linux作業系統上常用網路鑑識工具
 - Wireshark/Tshark(詳見後續分析)
 - NetworkMiner(詳見後續分析)
 - Tcpdump
 - PcapXray
 - Bro/ Zeek
 - Arkime/Moloch - IPv4 traffic capturing, indexing and database system.
 - ngrep - Search through network traffic like grep.
 - PcapViz - Network topology and traffic visualizer.
 - <https://github.com/1ultimat3/PcapViz>

網路鑑識工具：ngrep(1/2)

- ngrep (network grep)
- 是一個簡單易用且功能強大的網路封包分析工具

ngrep類似 Linux 的 grep 指令，可使用正規表示法來匹配網路封包中的資料(payloads)，抓取出實際含有指定資料的封包

```
(root@Kali-2023)-[~]
# ngrep -h
usage: ngrep <-hNXViqpevxLDtTRM> <-IO pcap_dump> <-n num> <-d dev> <-A num>
      <-s snaplen> <-S limitlen> <-W normal|byline|single|none> <-c co
ls>
      <-P char> <-F file>
      <match expression> <bpf filter>
      <-K count>

-h is help/usage
-V is version information
-q is be quiet (don't print packet reception hash marks)
-e is show empty packets
-i is ignore case
-v is invert match
-R is don't do privilege revocation logic
-x is print in alternate hexdump format
-X is interpret match expression as hexadecimal
-w is word-regex (expression must match as a word)
-p is don't go into promiscuous mode
-l is make stdout line buffered
-D is replay pcap_dumps with their recorded time intervals
-t is print timestamp every time a packet is matched
-T is print delta timestamp every time a packet is matched
  specify twice for delta from first match
-M is don't do multi-line match (do single-line match instead)
-I is read packet stream from pcap format file pcap_dump
-O is dump matched packets in pcap format to pcap_dump
-n is look at only num packets
-A is dump num packets after a match
-s is set the bpf caplen
-S is set the limitlen on matched packets
-W is set the dump format (normal, byline, single, none)
-c is force the column width to the specified size
-P is set the non-printable display char to what is specified
-F is read the bpf filter from the specified file
-N is show sub protocol number
-d is use specified device instead of the pcap default
-K is send N packets to kill observed connections
```

資料來源：<https://linuxhint.com/how-to-use-ngrep/>
<https://blog.gtwang.org/linux/linux-ngrep-network-packet-analyzer/>

網路鑑識工具：ngrep(2/2)

- ngrep [參數] [patterns匹配規則] [BPF 規則]

- [patterns匹配規則]可使用正規表示法(regular expression)或是十六進位的資料
- [BPF 規則]是Berkeley packet filter (BPF)用來透過關鍵詞(keywords)例如協定種類(protocol)或連接埠等)來指定封包選擇規則

範例	範例指令
監看所有 SMTP 郵件傳送 (port 25) 的封包	ngrep -d any port 25
監看含有 error 字樣的 syslog 封包	ngrep -d any 'error' port syslog
篩選來源或目的 IP 位址為 192.168 開頭的封包	sudo ngrep -q 'HTTP' 'host 192.168'
將符合條件的封包資料儲存下來，可以使用 -O 參數指定儲存的 pcap 檔案	sudo ngrep -O output.pcap -q 'HTTP'

網路鑑識工具：Arkime/Moloch(1/2)

- Arkime(前身為 Moloch)是一個大規模、開源、索引封包擷取與搜尋工具
 - Arkime 以標準 PCAP 格式儲存與匯出所有封包，可以在分析工作流程中使用喜歡的 PCAP 擷取工具
 - Arkime具有可擴充性(Scalability)
 - Arkime 設計為跨多個叢集系統部署，以便提供擴充能力來處理每秒數gigabits的流量。
 - PCAP 保留在感測器硬碟，而元資料則會保留在 OpenSearch/Elasticsearch 叢集。兩者都可以隨時增加。

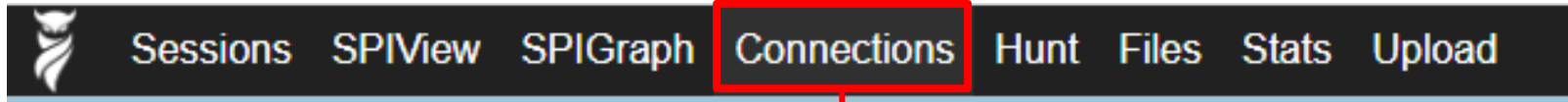


Arkime Architecture請參閱
<https://arkime.com/architecture>

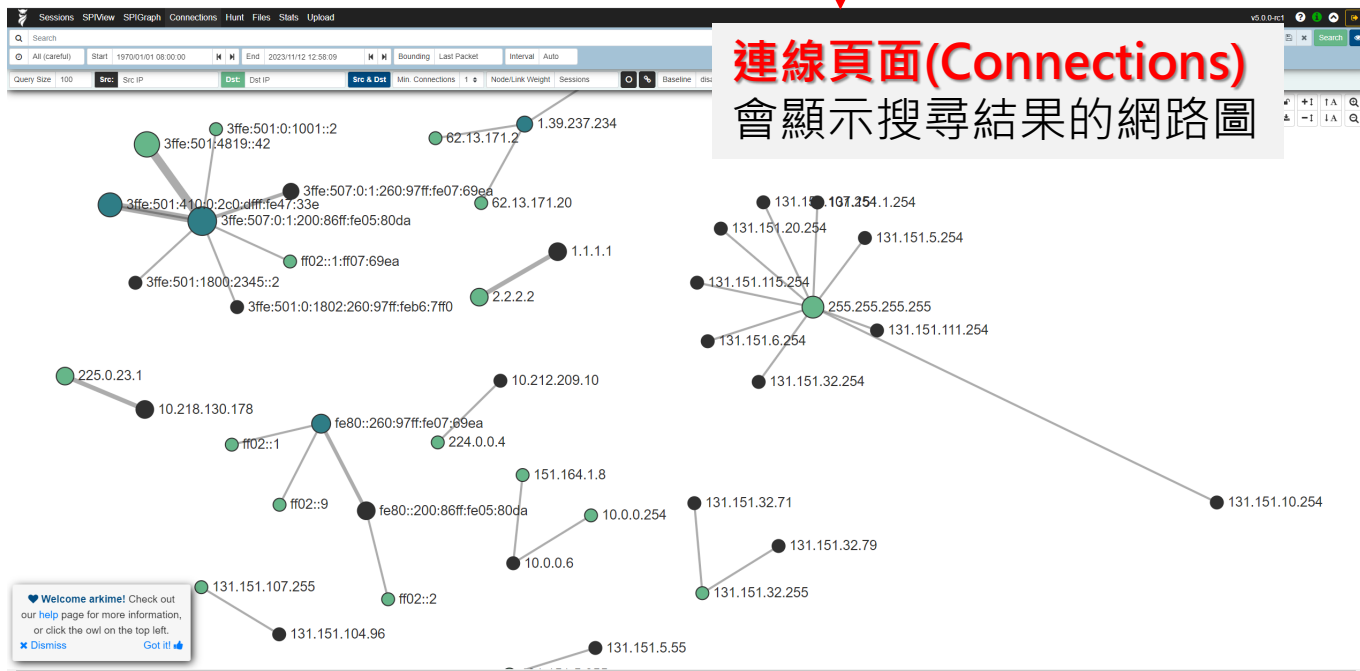
官方網址: <https://arkime.com/>

網路鑑識工具：Arkime/Moloch(2/2)

- Arkime提供 Web 介面：
 - 提供了一個 Web 應用程式，用於 PCAP 瀏覽、搜尋、分析及 PCAP carving



- ✓ Sessions 會顯示依所選時間區段和搜尋表達式(search expression)的索引化會話(indexed sessions)清單。
- ✓ SPIVIEW 可查看每個擷取欄位的唯一值以及會話數目。
- ✓ SPIGraph則會顯示欄位頂部唯一值(top unique values)的時間視圖(temporal view)。





網路鑑識主題(1/4)

- 網路鑑識主題涵蓋許多主題
 - Deep Packet Inspection (DPI)
 - 統計流程分析(Statistical Flow Analysis)
 - 惡意流量分析：特別是擷取流量中的惡意程式
 - 根據檔案特徵(file signature: header與footer)從網路流量擷取檔案的工具，例如tcpxtract工具
 - 加密流量分析
 - 包含辨識駭客使用的各種tunneling的封包及VPN、HTTPS、SSH等
 - 無線網路流量分析(請參閱底下推薦書籍)



網路鑑識主題(2/4)

- Deep Packet Inspection (DPI)深度封包檢視
 - DPI 不僅分析TCP/IP header並涉及分析有效payload
 - 具有 DPI 功能的設備可以分析、評估及執行從Layer- 2 到Layer-7 本身的操作
 - DPI廣泛應用於底下領域與服務：
 - 流量控管(Traffic shapers)
 - 服務保證(Service assurance)：
 - 識別偽造的應用程式(fake applications)：
 - 惡意軟體偵測
 - 入侵偵測
 - 資料外洩防護 (DLP，Data Loss Prevention)



網路鑑識主題(3/4)

- 統計流程分析(Statistical Flow Analysis)
 - 透過收集網路封包的某些特定欄位(定義在不同解決方案)來展現網路流量的(統計)特徵
 - 可快速檢視與縮小大幅容量儲存(在大流量的機關更有吸引力)
 - 著名的Flow解決方案有 NetFlow(v5 、v9)、IPFIX 及 sFlow等
 - 工具
 - flow-tools
 - Nfdump：允許從網路流量記錄中過濾與計算簡單的摘要/top-N 統計資料
 - ◆ <https://github.com/phaag/nfdump>
 - 缺點
 - ◆ 缺乏對Flow特徵進行分析與建模的任何功能
 - ◆ 不提供任何工具來合併由於活動逾時而分裂的flow記錄

網路鑑識主題(4/4)

- 著名統計流量分析(statistical flow analysis)
 - Yet Another Flowmeter (YAF)
 - YAF 是一種工具，可將 pcap 檔案中的封包或網路介面的即時擷取資料處理為 IPFIX 導向的檔案格式的雙向流
 - SiLK - the System for Internet-Level Knowledge
 - SiLK 是由 CERT 網路態勢感知團隊 (NetSA，Network Situational Awareness Team) 開發的龐大流量分析工具集合，旨在促進大型網路的安全分析
 - SiLK 工具套件支援網路流量資料的高效收集、儲存及分析，使網路安全分析師能夠快速查詢大量歷史流量資料集
 - SiLK 非常適合分析大型分散式企業或中型 ISP 的主幹或邊界上的流量
 - SiLK 由兩大類應用程式組成：分析套件(Analysis Suite)與打包系統
 - <https://tools.netsa.cert.org/silk/silk.html>



Wireshark常用技術(1/3)

- 過濾器(filter)
 - 使用擷取過濾器(capture filter)
 - 使用顯示過濾器(display filter)
- Statistical Flow Analysis技術
- 擷取流量中的資料(檔案、程式等)
- 解密加密流量的技術(在特定條件下)

Wireshark常用技術(2/3)

● 擷取過濾器(capture filter)

—在底下輸入擷取條件

➤ 擷取過濾器設定：host 172.18.5.4

◆意義：僅擷取進出 IP 位址 172.18.5.4 的流量

➤ 擷取過濾器設定：ip

◆意義：僅擷取 IPv4 流量

➤ 擷取過濾器設定：port not 53 and not arp

◆意義：擷取除所有 ARP 與 DNS 之外的流量

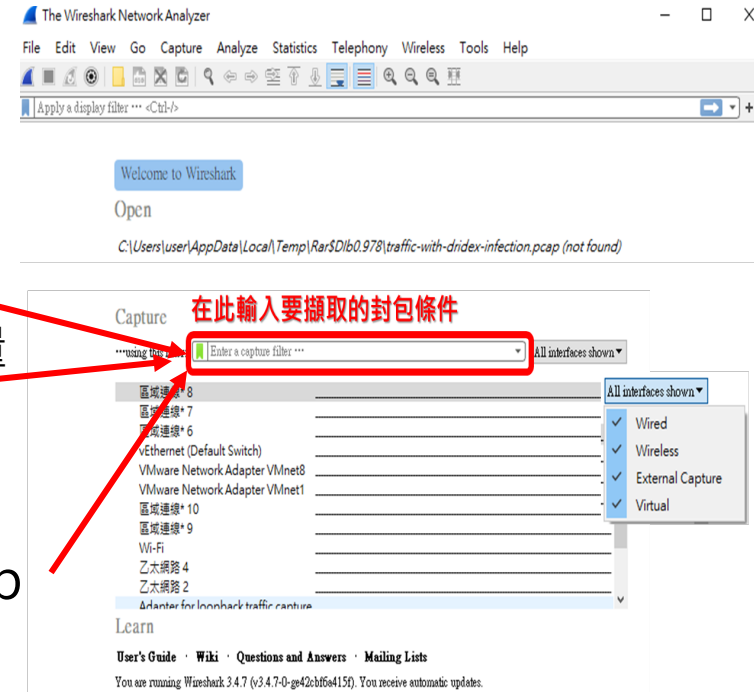
➤ 擷取過濾器設定：not broadcast and not multicast

◆意義：僅擷取單播流量(unicast)

➤ 更多擷取過濾器設定請參閱

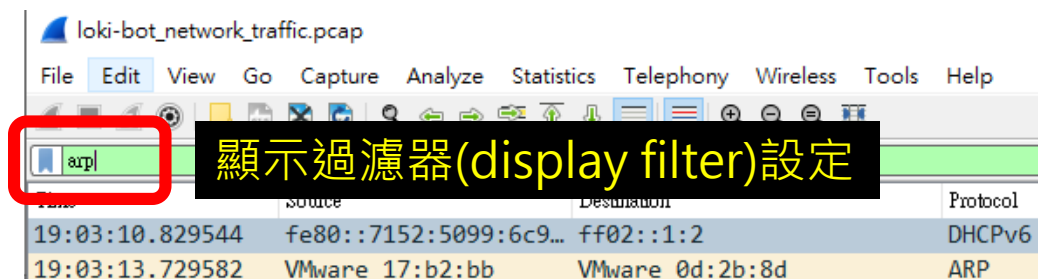
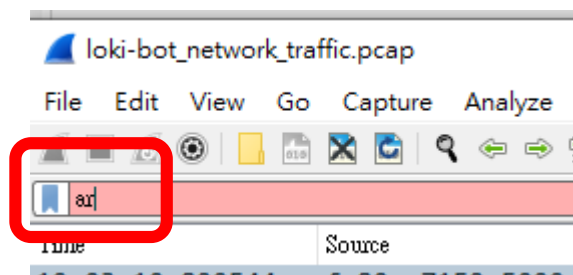
◆<https://wiki.wireshark.org/CaptureFilters>

◆https://www.wireshark.org/docs/wsug_html_chunked/ChCapCaptureFilterSection.html



Wireshark常用技術(3/3)

- 顯示過濾器(display filter)
 - 用來設定不同條件來挑選符合條件的封包
 - 設定**成功(綠色)**與**失敗(紅色)**的顏色不同



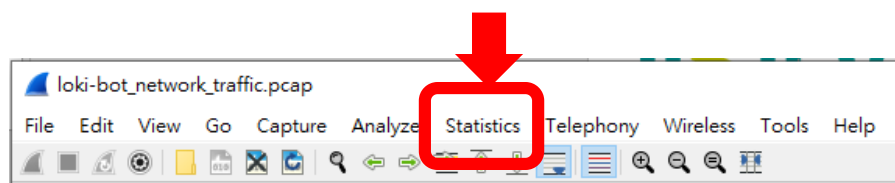
- 範例： 更多設定請參閱 <https://wiki.wireshark.org/DisplayFilters>

tcp.port eq 25 or icmp	僅顯示SMTP (連接埠 25) 與ICMP流量
ip. addr == 10.43.54.65	僅顯示與IP 來源 位址或IP 目標 位址是10.43.54.65封包
ip. addr != 10.43.54.65	顯示與IP 來源 位址或IP 目標 位址 非 10.43.54.65封包
ip. src == 10.43.54.65	僅顯示與IP 來源 位址是10.43.54.65封包
http.response.code == 200	僅顯示某些http回應(response)是成功(OK：代碼200)

Wireshark常用技術：統計技術(1/4)

- Wireshark 統計技術

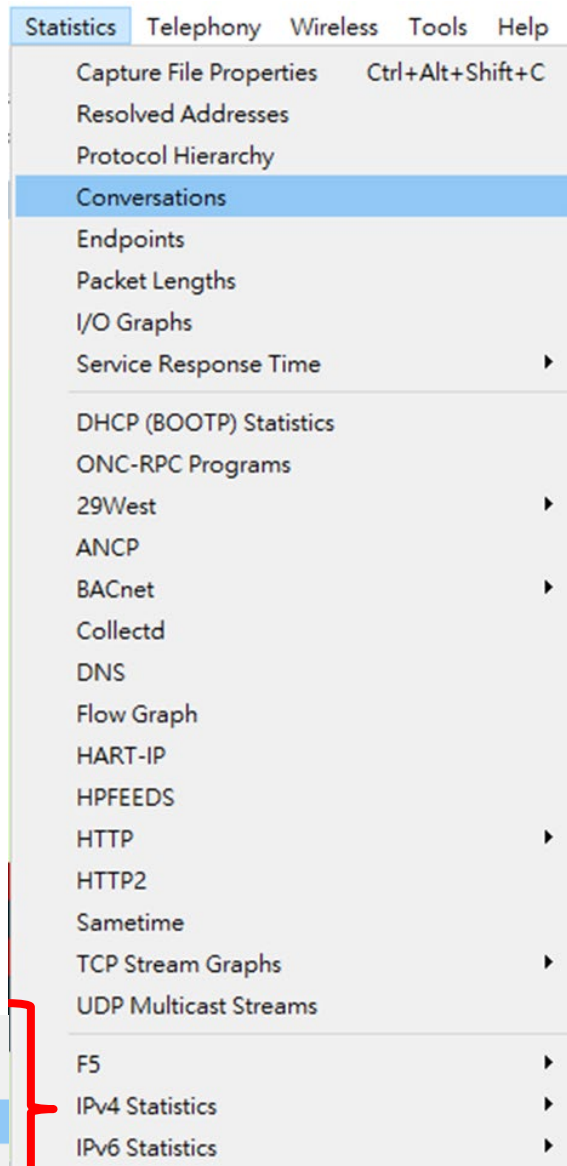
- Wireshark 提供基本的流量分析功能，例如協定階層(protocol hierarchy)、I/O圖(I/O graphs)以及 IPv4 與 IPv6 統計資訊等



可查看與所有關聯 IP 位址的統計資訊

可查看與所有目的 IP 位址與通訊埠的統計資訊

All Addresses
Destinations and Ports
IP Protocol Types
Source and Destination Addresses

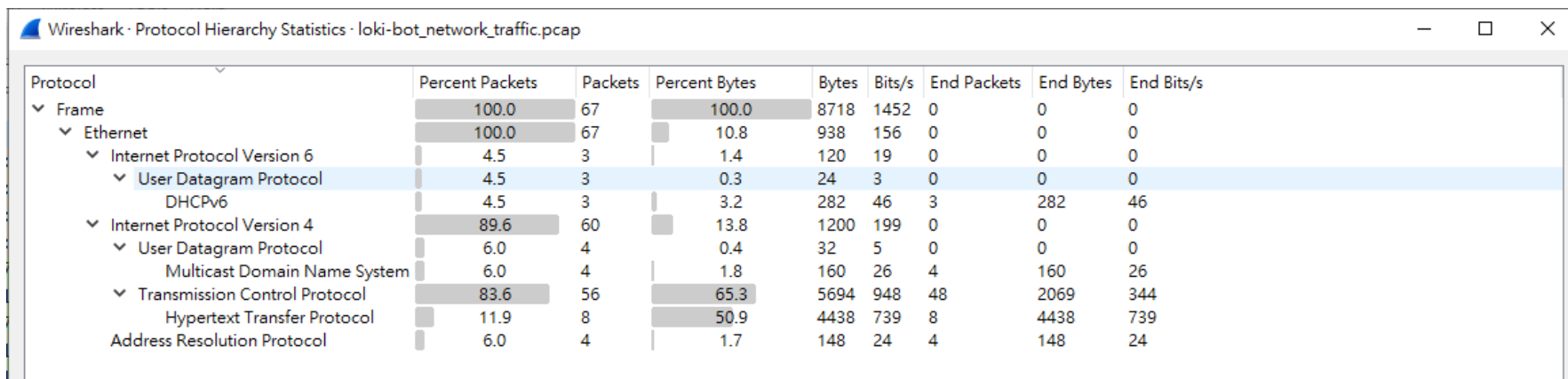


IPv6 Statistics也存在類似的選項

Wireshark常用技術：統計技術(2/4)

- 協定階層(protocol hierarchy)

一點選【Statistics | Protocol hierarchy】，可找到協定與相關位元組、位元/秒、位元組百分比以及封包計數的詳細清單

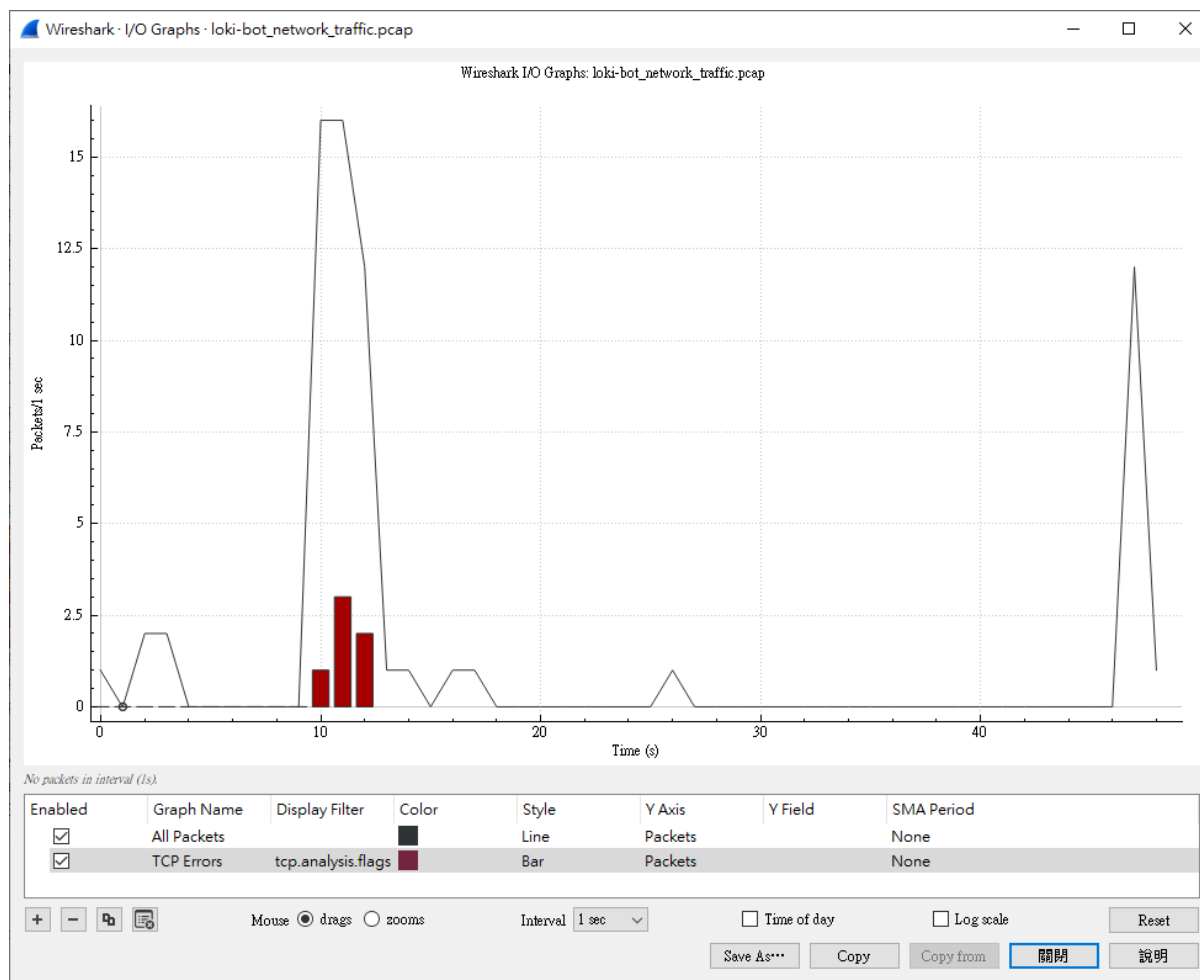


Wireshark · Protocol Hierarchy Statistics · loki-bot_network_traffic.pcap

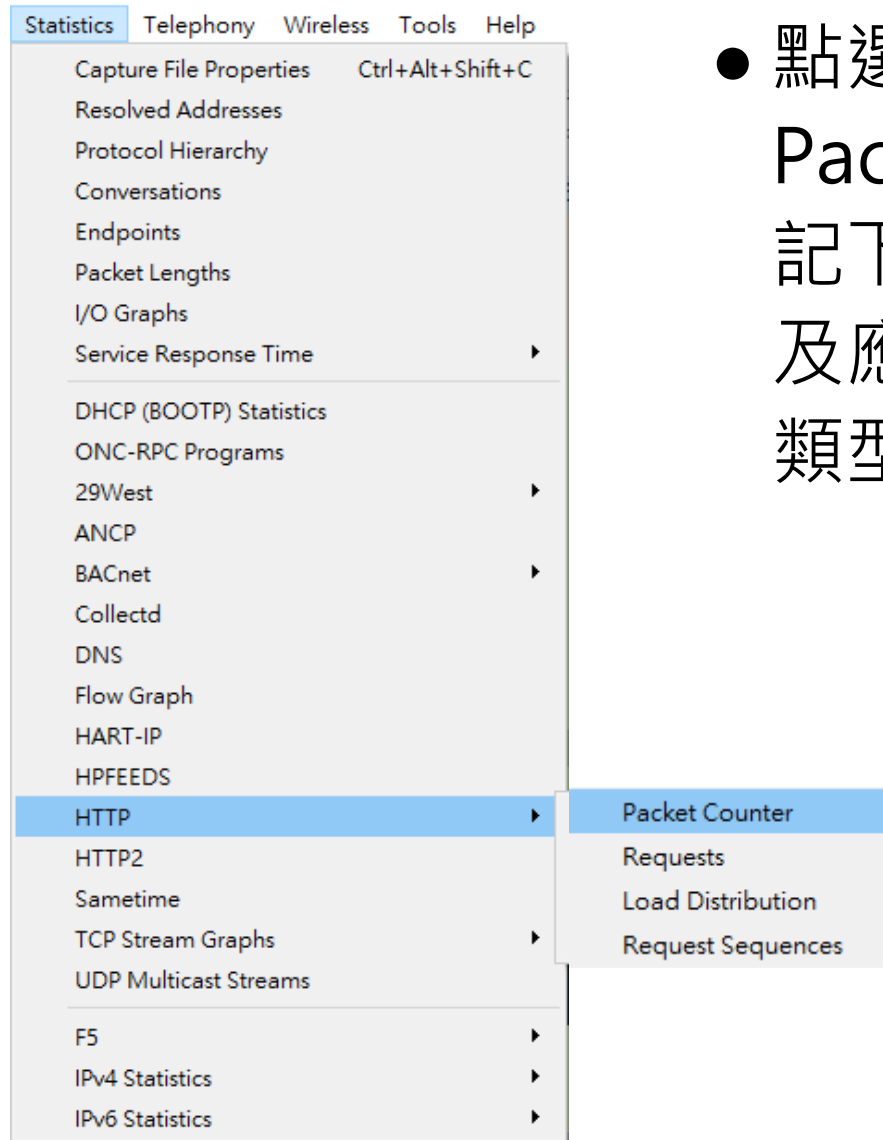
Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
▼ Frame	100.0	67	100.0	8718	1452	0	0	0
▼ Ethernet	100.0	67	10.8	938	156	0	0	0
▼ Internet Protocol Version 6	4.5	3	1.4	120	19	0	0	0
▼ User Datagram Protocol	4.5	3	0.3	24	3	0	0	0
DHCPv6	4.5	3	3.2	282	46	3	282	46
▼ Internet Protocol Version 4	89.6	60	13.8	1200	199	0	0	0
▼ User Datagram Protocol	6.0	4	0.4	32	5	0	0	0
Multicast Domain Name System	6.0	4	1.8	160	26	4	160	26
▼ Transmission Control Protocol	83.6	56	65.3	5694	948	48	2069	344
Hypertext Transfer Protocol	11.9	8	50.9	4438	739	8	4438	739
Address Resolution Protocol	6.0	4	1.7	148	24	4	148	24

Wireshark常用技術：統計技術(3/4)

- 點選【Statistics | IO Graph】選項可查看特定時間間隔內是否有流量的突然上升情況



Wireshark常用技術：統計技術(4/4)



- 點選【 Statistics | HTTP | Packet Counter】選項可快速記下 Web 應用程式中的錯誤以及應用程式傳送給使用者的回應類型

The image shows the 'Wireshark - Packet Counter - loki-bot_network_traffic.pcap' window. It displays a table of HTTP statistics.

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
▼ Total HTTP Packets	4				0.0001	100%	0.0100	10.092
Other HTTP Packets	0				0.0000	0.00%	-	-
▼ HTTP Response Packets	0				0.0000	0.00%	-	-
??? : broken	0				0.0000	-	-	-
5xx: Server Error	0				0.0000	-	-	-
4xx: Client Error	0				0.0000	-	-	-
3xx: Redirection	0				0.0000	-	-	-
2xx: Success	0				0.0000	-	-	-
1xx: Informational	0				0.0000	-	-	-
▼ HTTP Request Packets	4				0.0001	100.00%	0.0100	10.092
POST	4				0.0001	100.00%	0.0100	10.092

Display filter: Apply

Copy Save as*** 關閉

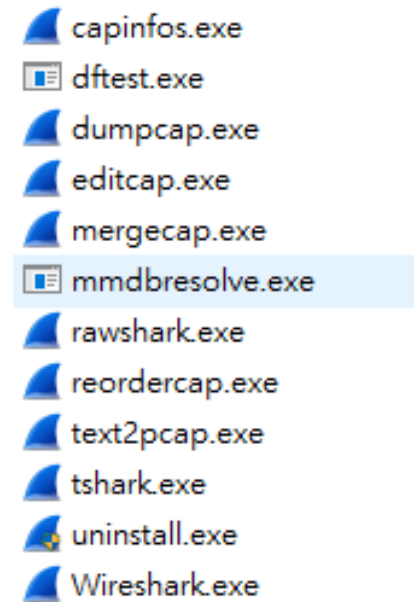
Wireshark命令列工具(1/2)

- Wireshark有許多命令列工具

- Capinfos：列印有關擷取文件的資訊
- Rawshark：轉儲與分析原始 pcap 數據
- Editcap：編輯|轉換pcap文件的格式
- Mergecap：將兩個或多個pcap合併為一個
- Reordercap：按時間戳記將輸入檔案重新排序到輸出檔案
- Text2pcap：從封包的 ASCII 十六進位轉儲產生pcap檔案
- 更多指令說明請參閱

- <https://www.wireshark.org/docs/man-pages/>

- https://www.wireshark.org/docs/wsug_html/附錄 **D. Related command line tools**



Wireshark命令列工具(2/2)

● Dumpcap

- 是一個網路流量轉儲工具
- 它從即時網路擷取封包資料並將封包寫入檔案
- Dumpcap 的原生擷取檔案格式是 pcapng，這也是 Wireshark 使用的格式
- 使用 Ctrl-C 可以隨時停止擷取

`dumpcap -i eth0 -a duration:60 -w output.pcapng`
從介面 `eth0` 擷取封包，共持續 60 秒 寫入到 `output.pcapng`

tshark分析技術(1/5)

- Tshark是Wireshark強大的命令列工具

- 可使用 tshark.exe -h 顯示可使用參數
- Tshark語法格式

```
tshark [ -i <擷取介面>|- ] [ -f <擷取過濾器等> ] [ -2 ] [ -r <輸入檔> ]  
[ -w <輸出檔>|- ] [選項] [ <過濾器> ]
```

– 範例1： sudo tshark -i eth0 -f "tcp"

➤ -i <擷取介面>

➤ -f <擷取過濾器等capture filter>

➤ 意義： 從etho介面擷取 只包含tcp封包

tshark分析技術(2/5)

● Tshark語法格式

```
tshark [ -i <擷取介面>|- ] [ -f <擷取過濾器> ] [ -2 ] [ -r <輸入檔> ]  
[ -w <輸出檔>|- ] [選項] [ <過濾器> ]
```

```
sudo tshark -r network.pcap -T json "http.request.method == GET"
```

輸入檔 使用json
輸出格式 顯示過濾器(display filter)
只顯示使用GET的HTTP 封包

● Tshark參數：

-i	--interface <擷取介面> 設定用於即時封包擷取的網路介面或管道的名稱
-D	列印TShark可以擷取的介面列表，然後退出
-f	-f <擷取過濾器> 用來設定擷取過濾器
-r	--讀取檔案 <infile> 從infile讀取封包數據
-Y	-Y --display-filter <顯示過濾器> 顯示封包的解碼形式 或 將封包寫入檔案之前應用指定的過濾器
-Z	-z <統計> 讓TShark收集各種類型的統計資訊，並在讀取pcap文件後顯示結果



tshark分析技術(3/5)

● HTTP封包分析技術

- `sudo tshark -r packetFile.pcap -Y http -w packetFile-http.pcap`
 - 使用-Y設定顯示過濾器(display filter) 本例只顯示http封包
 - 使用-w設定輸出檔名
- `sudo tshark -r packetFile.pcap -Y http.request -T fields -e http.host -e http.user_agent`
 - 只顯示http.request封包
 - 使用-T fields 來設定要使用特定欄位，再加上使用-e 來設定這些特定欄位(可以有多個) 本例只顯示http.host 與http.user_agent
- `sudo tshark -nr test.pcap --export-objects smb,tmpfolder`
 - 使用--export-objects將pcap的檔案匯出
 - 本例是將smb流量匯出到tmpfolder目錄

tshark分析技術(4/5)

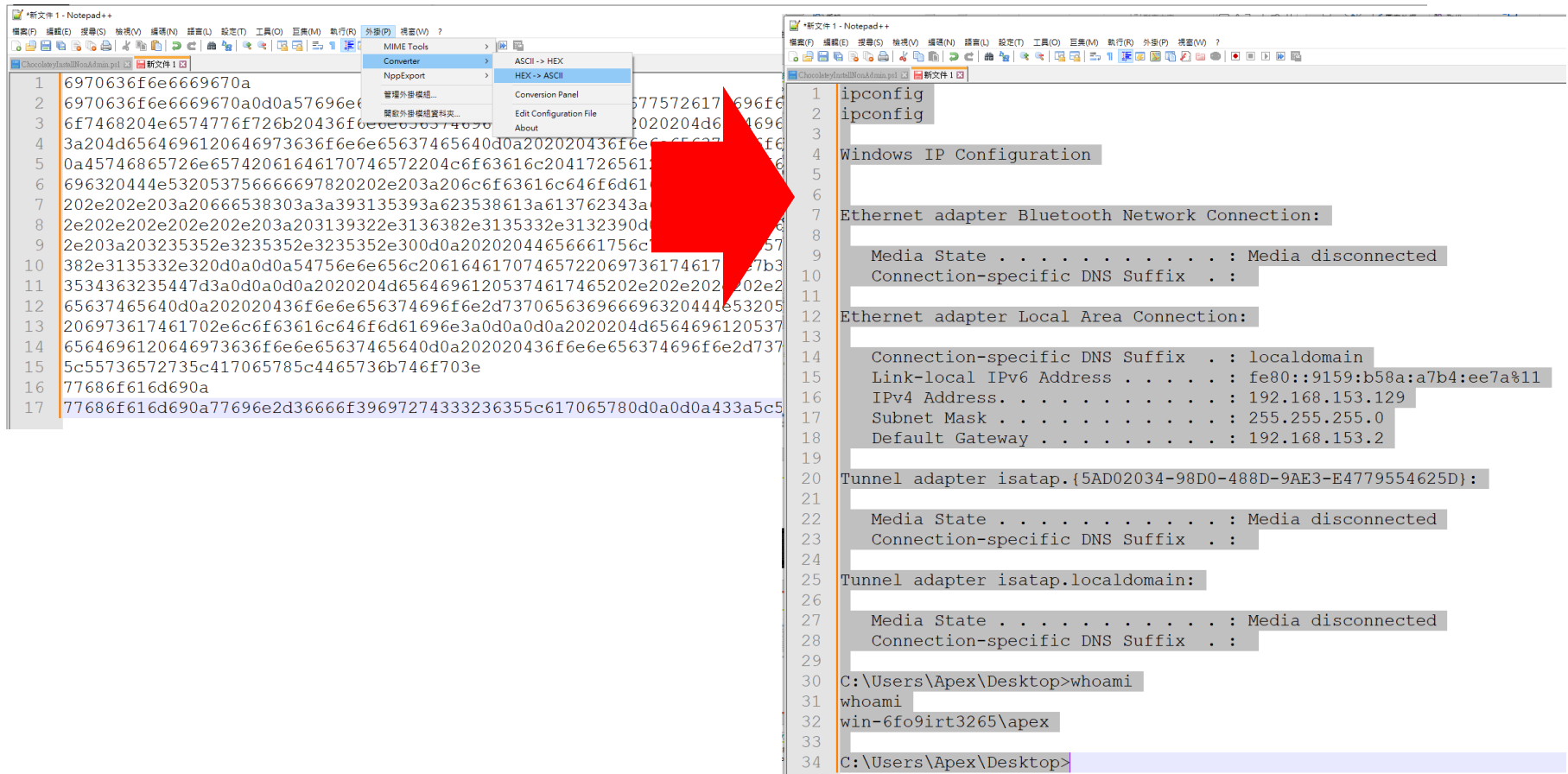
- 使用tshark命令列指令擷取流量中的資料
- 範例檔案下載
 - <https://github.com/nipunjaswal/networkforensics/tree/master/Ch3/ICMP%20Camp>

```
tshark.exe -Y data -r D:\NET\icmp_camp.pcapng -T fields -e data
```

[illegible]

tshark分析技術(5/5)

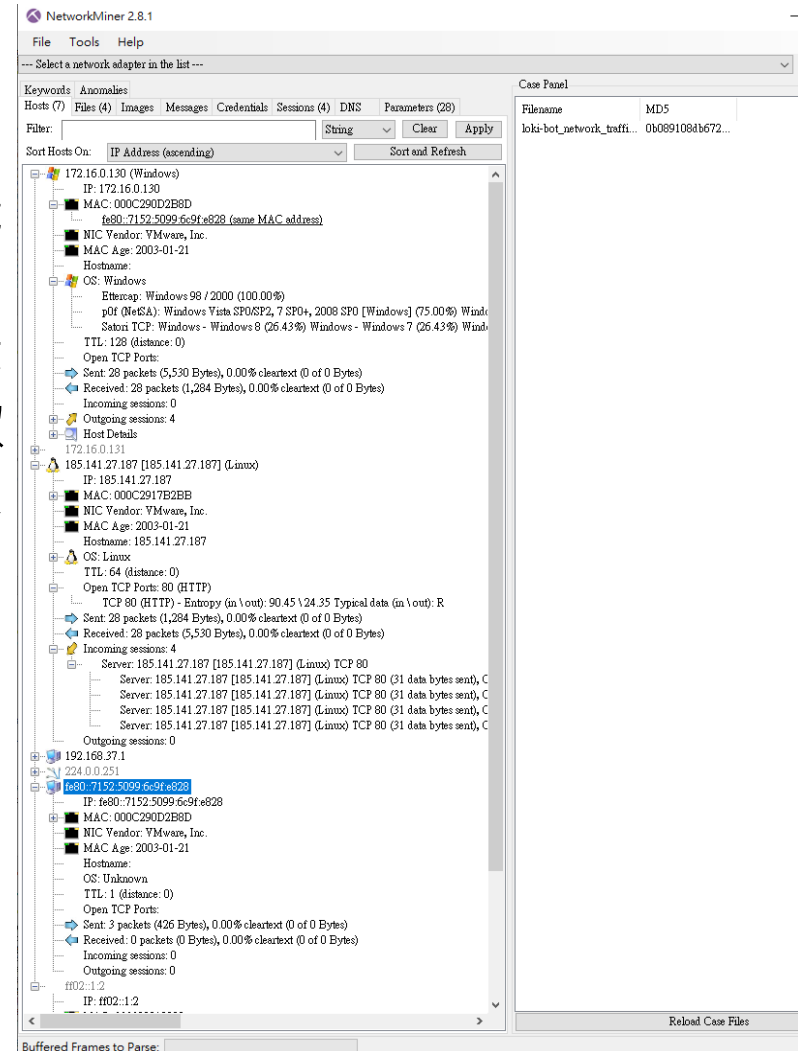
- 將顯示資料貼入notepad++
- 在notepad++點選【外掛 | converter | HEX -- > ASCII】



NetworkMiner網路鑑識(1/7)

● NetworkMiner

- 是一款功能強大的封包擷取與分析工具
- 會自動解析與分類文件中的資訊，使鑑識人員的分析變得更加容易
- 鑑識結果以更簡單且易於閱讀的格式顯示
- 最大優點：可簡單的從 PCAP 檔案中擷取網路流量中擷取檔案、影像、電子郵件及密碼等。
- 下載NetworkMiner_2-8-1.zip後解壓縮即可使用



下載點：<https://www.netresec.com/?page=NetworkMiner>

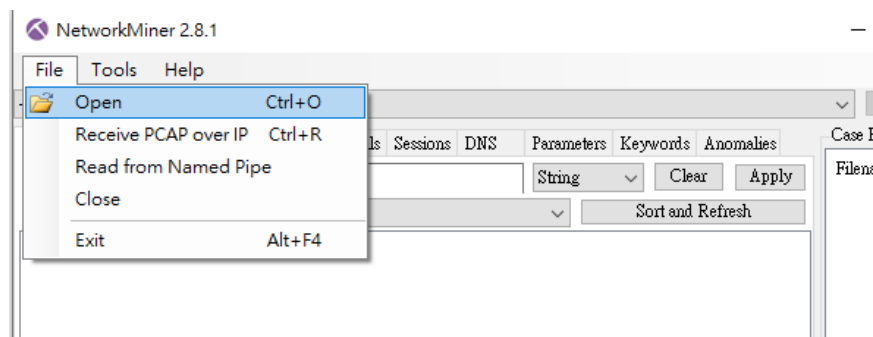
NetworkMiner網路鑑識(2/7)

● 講師展示教學

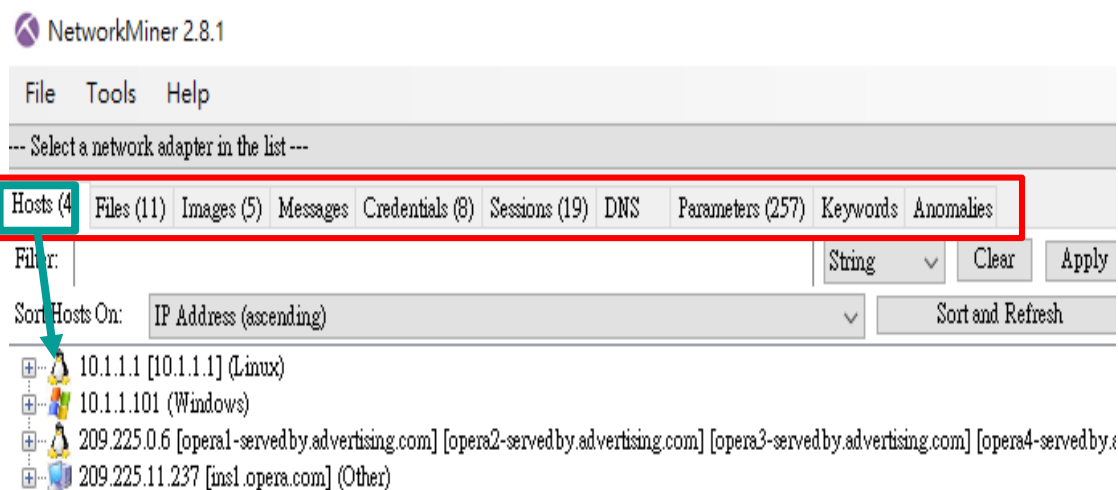
步驟一：下載底下測試PCAP並解壓縮

https://wiki.wireshark.org/uploads/__moin_import__/attachments/SampleCaptures/http_with_jpegs.cap.gz

步驟二：啟動MetworkMiner
點選File|Open將測試檔案載入

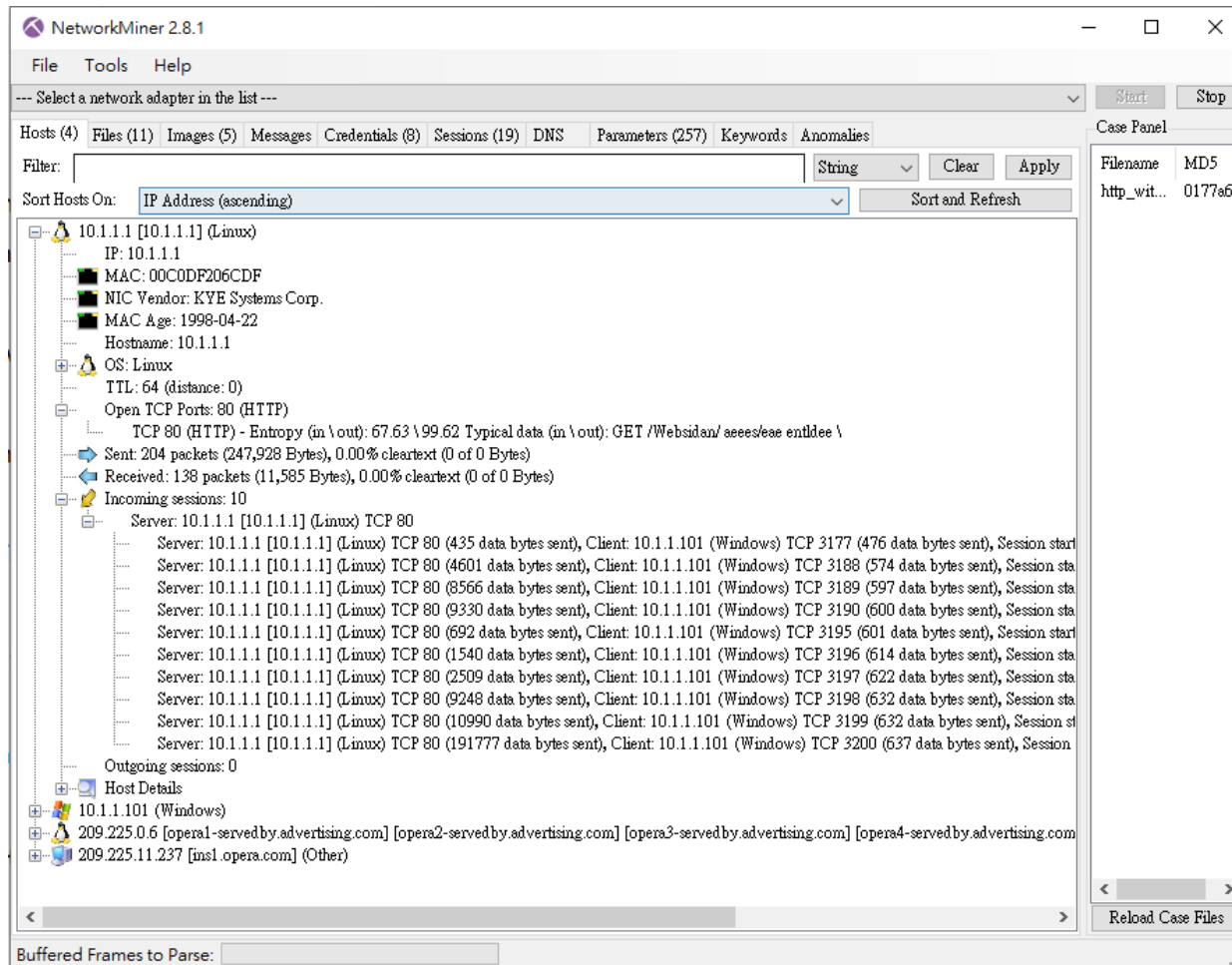


NetworkMiner會自動分析
將不同發現放置到不同Tab



NetworkMiner網路鑑識(3/7)

- Host欄位：呈現所有出現在封包的IP



NetworkMiner網路鑑識(4/7)

● File：檔案及檔案片段(未能完全重組)

The screenshot displays the NetworkMiner 2.8.1 application window. The 'Files (11)' tab is selected, showing a list of files. A red box highlights three files: DSC07858[1].JPG, DSC07859[1].JPG, and DSC07858[1].JPG. A red arrow points from this box to a 'File Details' window for 'DSC07858.JPG'. The 'File Details' window shows the file's metadata and a hex dump of its content. The hex dump shows a JFIF header and some data, but it is not a complete image file, as indicated by the text '未能完全重組' (Unable to fully reconstruct) and the red box around the file list.

Frame nr.	Filename	Extension	Size	Source host	S. port	Destination host	D. port	Protocol	Timestamp
4	index[1].html	html	160 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3177	HttpGetNormal	2004-11-19 22:29:14 UTC
16	xcms.asp.vnd[1].xaccp	xaccp	433 B	10.1.1.101 (Windows)	TCP 3179	209.225.11.237 [insl.opera.com] (Other)	TCP 80	HttpPostUpload	2004-11-19 22:29:14 UTC
31	index[1].html	html	4 323 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3188	HttpGetNormal	2004-11-19 22:29:15 UTC
48	bg2[1].jpg	jpg	8 281 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3189	HttpGetNormal	2004-11-19 22:29:15 UTC
50	sydney[1].jpg	jpg	9 045 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3190	HttpGetNormal	2004-11-19 22:29:15 UTC
157	dagbok[1].html	html	416 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3195	HttpGetNormal	2004-11-19 22:29:17 UTC
215	dagbok[1].html	html	1 263 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3196	HttpGetNormal	2004-11-19 22:29:19 UTC
227	dagbok[1].html	html	2 232 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3197	HttpGetNormal	2004-11-19 22:29:20 UTC
240	DSC07858[1].JPG	jpg	8 963 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3198	HttpGetNormal	2004-11-19 22:29:20 UTC
241	DSC07859[1].JPG	jpg	10 730 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3199	HttpGetNormal	2004-11-19 22:29:20 UTC
278	DSC07858[1].JPG	jpg	191 545 B	10.1.1.1 [10.1.1.1] (Linux)	TCP 80	10.1.1.101 (Windows)	TCP 3199	HttpGetNormal	2004-11-19 22:29:24 UTC

未能完全重組

DSC07858.JPG - File Details

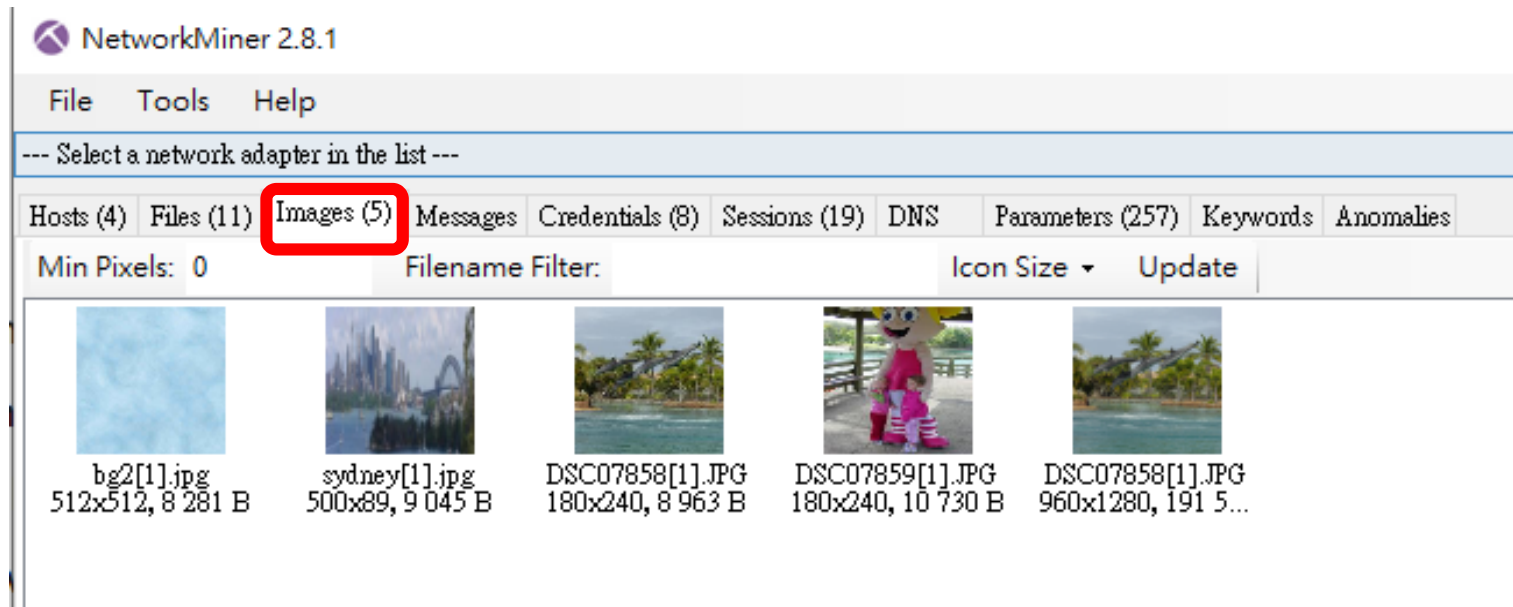
Name	Value
Name	DSC07858.JPG
MD5	835d8e7a12d31c4bbe4eff7b4b5ab3b
SHA1	32e02bbe5a1a6ec90af3f75b59af5942ae83949a
SHA256	ff9140064b9b70609962b4430ce3090be373e8f7e876e86497f01e9
Path	D:\WETNetworkMiner_2-8-1\NetworkMiner_2-8-1\AssembledFiles
Size	8963
LastWriteTime	2004/11/20 上午 06:29
Source	10.1.1.1 [10.1.1.1] (Linux)
Destination	10.1.1.101 (Windows)

Max bytes to read: 1024 Font size: 10 File type: JPG

```
FFD8FFE000104A464946000101010048      ????.JFIF....H
00480000FFFE00174372656174656420      .H...?.Created
77697468205468652047494050FFD800      with The GIMP??.
43000806060706050807070909080A      C.....
0C140D0C080B0C1912130F141D1A1F1E      .....
1D1A1C1C20242E2720222C231C1C2837      ....$.',#..(7
292C30313434341F27393D38323C2E33      ),01444.'9=82<.3
3432FFD80043010909090C0B0C180D0D      42??.C.....
1832211C213232323232323232323232      .2!..!2222222222
```

NetworkMiner網路鑑識(5/7)

- Images：重組成功的圖片
 - 可清晰識別其內容



NetworkMiner網路鑑識(6/7)

● Credentials：各種解析成功的憑證與cookie

NetworkMiner 2.8.1

File Tools Help

--- Select a network adapter in the list ---

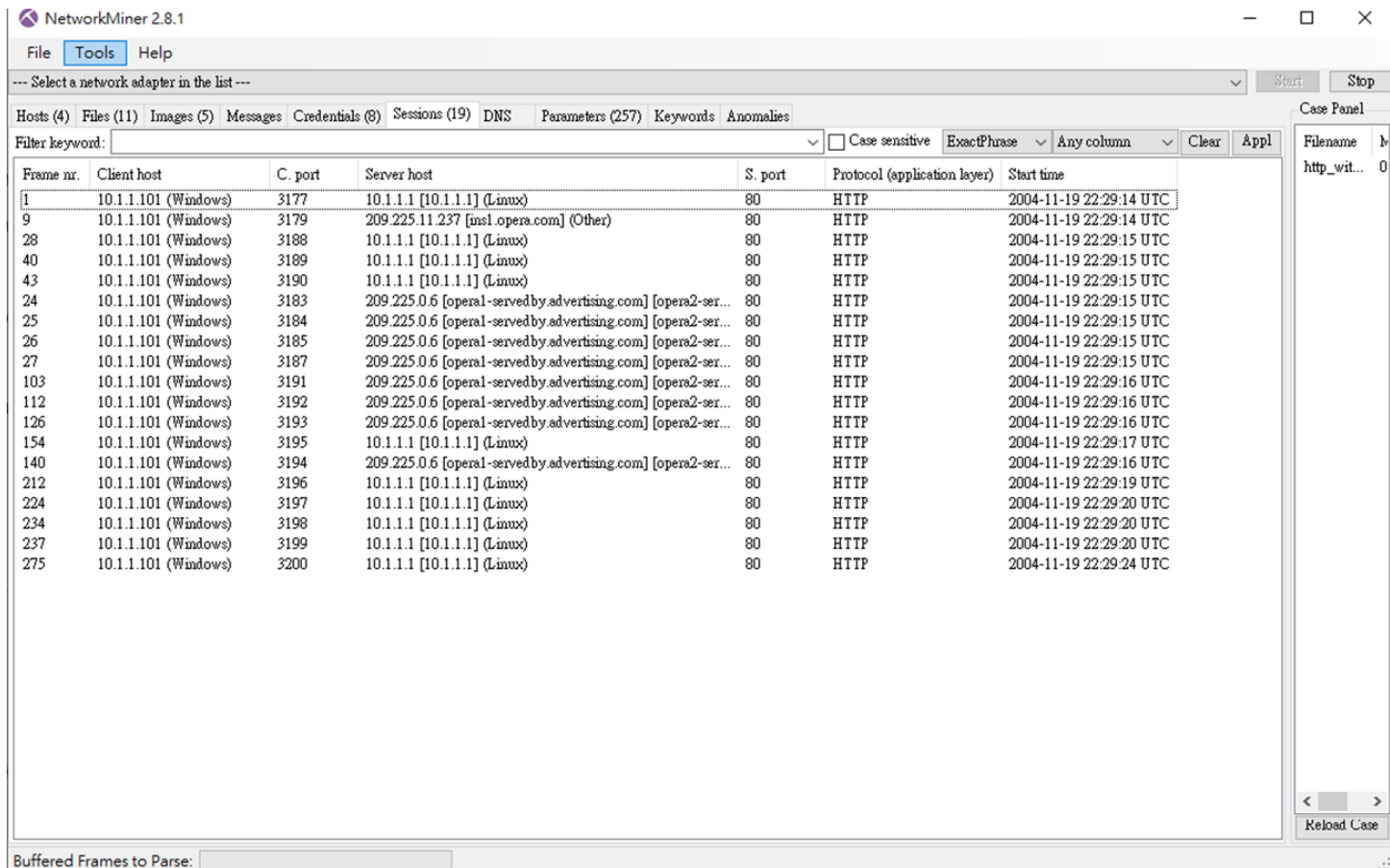
Hosts (4) Files (11) Images (5) Messages Credentials (8) Sessions (19) DNS Parameters (257) Keywords Anomalies

☒ Show Cookies ☒ Show NTLM challenge-response ☐ Mask Passwords

Client	Server	Protocol	Username	Password	Valid login	Login timestamp
10.1.1.101 (Windows)	209.225.0.6 [opera1-servedby.advertising.com]	HTTP Cookie	opera1= 4197c290,,126885^76592&127709^162766_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:15 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera2-servedby.advertising.com]	HTTP Cookie	opera2= 4197c290,,126885^76592&127709^162766_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:15 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera3-servedby.advertising.com]	HTTP Cookie	opera3= 4197c291,,126885^76592&127709^162766_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:15 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera4-servedby.advertising.com]	HTTP Cookie	opera4= 4197c291,,126885^76592&127709^162766_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:15 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera1-servedby.advertising.com]	HTTP Cookie	opera1= 419e70fb,,126885^76592&127709^162763_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:16 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera2-servedby.advertising.com]	HTTP Cookie	opera2= 419e70fb,,126885^76592&127709^162763_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:16 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera4-servedby.advertising.com]	HTTP Cookie	opera4= 419e70fc,,126885^76592&127709^162766_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:17 UTC
10.1.1.101 (Windows)	209.225.0.6 [opera3-servedby.advertising.com]	HTTP Cookie	opera3= 419e70fc,,126885^76592&127709^162763_ ; ACID=ee440...	N/A	Unknown	2004-11-19 22:29:17 UTC

NetworkMiner網路鑑識(7/7)

- Sessions：連線對話



NetworkMiner 2.8.1

File Tools Help

--- Select a network adapter in the list --- Start Stop

Hosts (4) Files (11) Images (5) Messages Credentials (8) Sessions (19) DNS Parameters (257) Keywords Anomalies

Filter keyword: Case sensitive ExactPhrase Any column Clear Appl

Frame nr.	Client host	C. port	Server host	S. port	Protocol (application layer)	Start time
1	10.1.1.101 (Windows)	3177	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:14 UTC
9	10.1.1.101 (Windows)	3179	209.225.11.237 [msl.opera.com] (Other)	80	HTTP	2004-11-19 22:29:14 UTC
28	10.1.1.101 (Windows)	3188	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:15 UTC
40	10.1.1.101 (Windows)	3189	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:15 UTC
43	10.1.1.101 (Windows)	3190	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:15 UTC
24	10.1.1.101 (Windows)	3183	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:15 UTC
25	10.1.1.101 (Windows)	3184	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:15 UTC
26	10.1.1.101 (Windows)	3185	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:15 UTC
27	10.1.1.101 (Windows)	3187	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:15 UTC
103	10.1.1.101 (Windows)	3191	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:16 UTC
112	10.1.1.101 (Windows)	3192	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:16 UTC
126	10.1.1.101 (Windows)	3193	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:16 UTC
154	10.1.1.101 (Windows)	3195	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:17 UTC
140	10.1.1.101 (Windows)	3194	209.225.0.6 [opera1-servedby.advertising.com] [opera2-ser...	80	HTTP	2004-11-19 22:29:16 UTC
212	10.1.1.101 (Windows)	3196	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:19 UTC
224	10.1.1.101 (Windows)	3197	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:20 UTC
234	10.1.1.101 (Windows)	3198	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:20 UTC
237	10.1.1.101 (Windows)	3199	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:20 UTC
275	10.1.1.101 (Windows)	3200	10.1.1.1 [10.1.1.1] (Linux)	80	HTTP	2004-11-19 22:29:24 UTC

Case Panel

Filename http_wit... 0

Reload Case

Buffered Frames to Parse: