PACKET ANALYSIS

Session 2 – Network Security and Forensics

FUNDAMENTALS AND CHALLENGES

- Fundamentals
  - Protocol analysis
  - Packet analysis
  - Multipacket stream analysis
  - Stream reconstruction
- Challenges
  - Not always possible to recover all packets
  - Packet data may be corrupted or truncated
  - Contents may be encrypted
  - Undocumented protocol
  - Sheer volume of data

PROTOCOL ANALYSIS

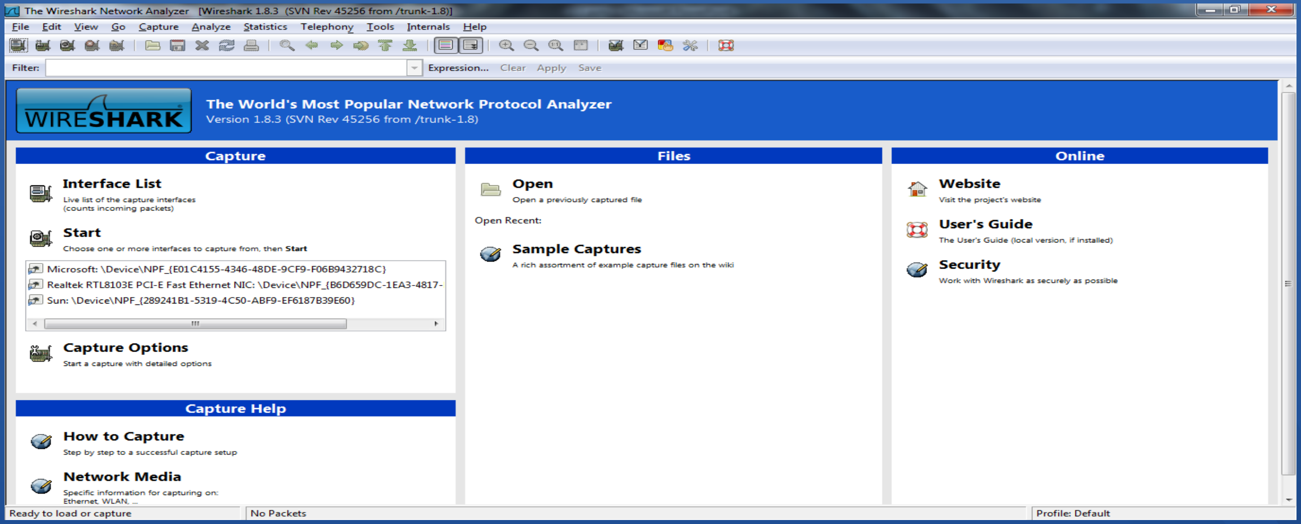
- Defined
    - “Examination of one or more fields within a protocol’s data structure. Protocol analysis is commonly conducted for the purposes of research (i.e., as in the case of an unpublished protocol specification) or network investigation.” (Davidoff & Ham, 2012)
- Best practice
    - Take cryptographic checksums of all data during collection
    - Work with an exact copy leaving original data intact

DOCUMENTATION

- Where to look
  - IETF - The Internet Engineering Task Force - http://www.ietf.org/
    - Large, public repository of documented protocols
    - RFCs – Requests for Comments - http://www.rfc-editor.org
      - Used to develop, communicate and define international standards for internetworking
  - IEEE-SA – Institute of Electrical and Electronics Standards Association
  - ISO – International Organization for Standardization
  - Vendors and researchers

PROTOCOL ANALYSIS TOOLS

- Packet Details Markup Language (PDML) and Packet Summary Markup Language (PSML)
- Wireshark
- Tshark
-



The Wireshark Network Analyzer [Wireshark 1.8.3 (SVN Rev 45256 from /trunk-1.8)]

The World's Most Popular Network Protocol Analyzer
Version 1.8.3 (SVN Rev 45256 from /trunk-1.8)

Interface List
Live list of the capture interfaces (counts incoming packets)

Start
Choose one or more interfaces to capture from, then **Start**

Microsoft: \Device\NPF_{E01C4155-4346-48DE-9CF9-F06B9432718C}
Realtek RTL8103E PCI-E Fast Ethernet NIC: \Device\NPF_{B6D659DC-1EA3-4817-
Sun: \Device\NPF_{289241B1-5319-4C50-ABF9-EF6187B39E60}

Capture Options
Start a capture with detailed options

How to Capture
Step by step to a successful capture setup

Network Media
Specific information for capturing on: Ethernet, WLAN, ...

Open
Open a previously captured file

Open Recent:

Sample Captures
A rich assortment of example capture files on the wiki

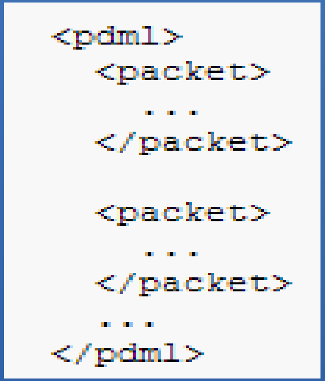
Website
Visit the project's website

User's Guide
The User's Guide (local version, if installed)

Security
Work with Wireshark as securely as possible

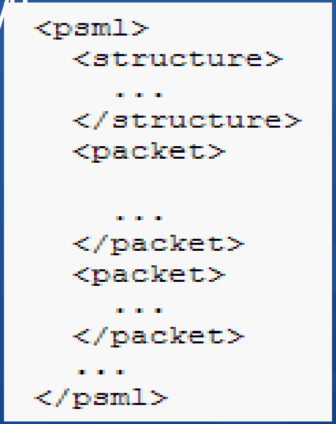
Ready to load or capture | No Packets | Profile: Default

PACKET DETAILS MARKUP LANGUAGE AND PACKET SUMMARY MARKUP LANGUAGE



```
<pdml>
  <packet>
    ...
  </packet>

  <packet>
    ...
  </packet>
  ...
</pdml>
```

- PDML
  - Expresses packet details for Layers 2-7 in an XML format
  - Example:
    - $ tshark -r capturefile.pcap -T pdml
- PSML
  - Used for most important details about a protocol also in XML
  - Example:
    - $ tshark -r capturefile.pcap -T psml
- Part of the NetBee library – support packet processing
- http://www.nbee.org/doku.php



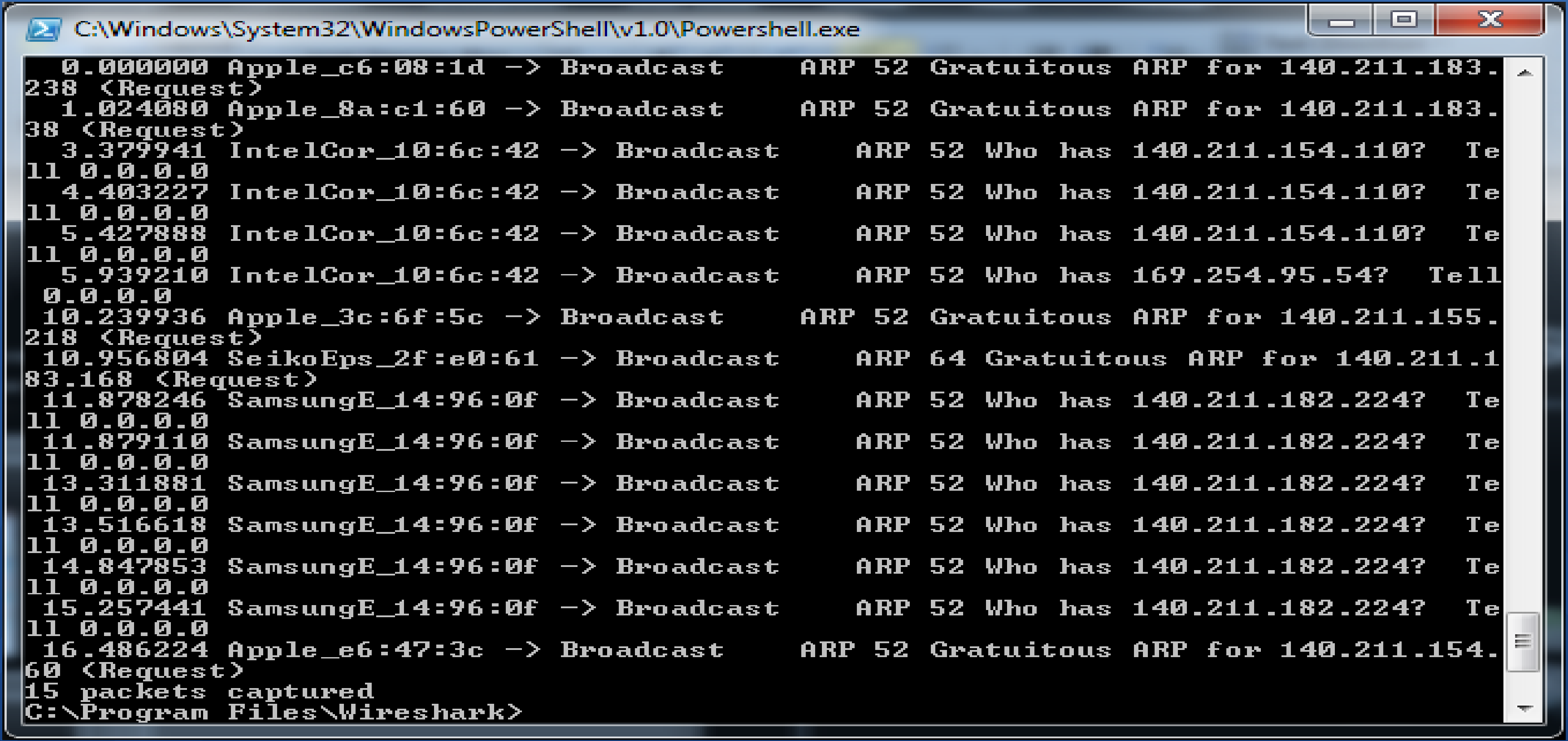
```
<psml>
  <structure>
    ...
  </structure>
  <packet>

    ...
  </packet>
  <packet>
    ...
  </packet>
  ...
</psml>
```

IMAGE/S CLIPPED FROM WORK CITED

TSHARK

- Same functionality as Wireshark using command-line interface
- Basic commands
  - $ tshark -r capturefile.pcap
    - Capture file
  - $ tshark -n -r capturefile.pcap
    - Disable network naming resolution to show IP addresses and port numbers, -n
  - $ tshark -r capturefile.pcap -T pdml
    - Select output format using t flag
  - $ tshark -r capturefile.pcap -T fields -e frame.number -e ip.addr -e udp
    - Prints a specific field, -e flag
  - $ tshark -r capturefile.pcap -d tcp.port ==29008 , http
    - Decode as, -d
  - $ tshark -r capturefile.pcap -R 'ip.addr == 192.168.1.1 '

TSHARK DISPLAY

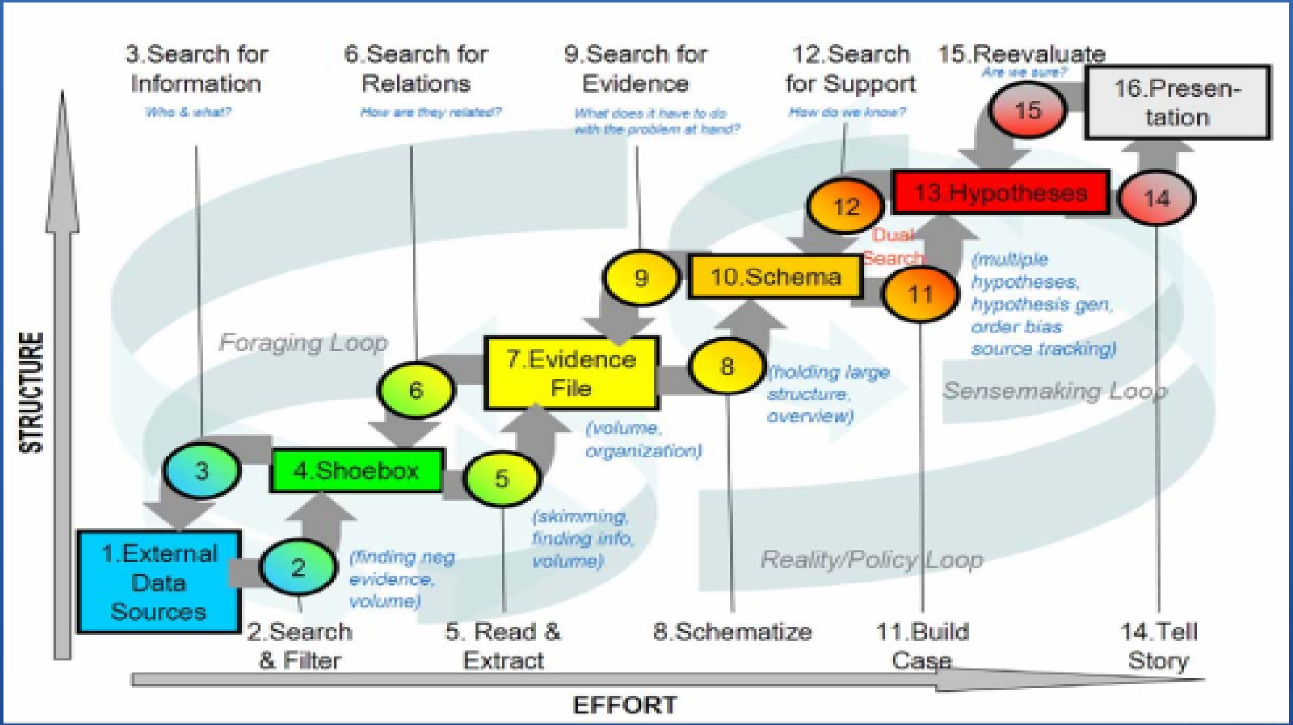


C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe

```
  0.000000 Apple_c6:08:1d -> Broadcast    ARP 52 Gratuitous ARP for 140.211.183.
238 (Request)
  1.024080 Apple_8a:c1:60 -> Broadcast    ARP 52 Gratuitous ARP for 140.211.183.
38 (Request)
  3.379941 IntelCor_10:6c:42 -> Broadcast    ARP 52 Who has 140.211.154.110?  Te
ll 0.0.0.0
  4.403227 IntelCor_10:6c:42 -> Broadcast    ARP 52 Who has 140.211.154.110?  Te
ll 0.0.0.0
  5.427888 IntelCor_10:6c:42 -> Broadcast    ARP 52 Who has 140.211.154.110?  Te
ll 0.0.0.0
  5.939210 IntelCor_10:6c:42 -> Broadcast    ARP 52 Who has 169.254.95.54?  Tell
 0.0.0.0
 10.239936 Apple_3c:6f:5c -> Broadcast    ARP 52 Gratuitous ARP for 140.211.155.
218 (Request)
 10.956804 SeikoEps_2f:e0:61 -> Broadcast    ARP 64 Gratuitous ARP for 140.211.1
83.168 (Request)
 11.878246 SamsungE_14:96:0f -> Broadcast    ARP 52 Who has 140.211.182.224?  Te
ll 0.0.0.0
 11.879110 SamsungE_14:96:0f -> Broadcast    ARP 52 Who has 140.211.182.224?  Te
ll 0.0.0.0
 13.311881 SamsungE_14:96:0f -> Broadcast    ARP 52 Who has 140.211.182.224?  Te
ll 0.0.0.0
 13.516618 SamsungE_14:96:0f -> Broadcast    ARP 52 Who has 140.211.182.224?  Te
ll 0.0.0.0
 14.847853 SamsungE_14:96:0f -> Broadcast    ARP 52 Who has 140.211.182.224?  Te
ll 0.0.0.0
 15.257441 SamsungE_14:96:0f -> Broadcast    ARP 52 Who has 140.211.182.224?  Te
ll 0.0.0.0
 16.486224 Apple_e6:47:3c -> Broadcast    ARP 52 Gratuitous ARP for 140.211.154.
60 (Request)
15 packets captured
C:\Program Files\Wireshark>
```

PROTOCOL ANALYSIS TECHNIQUES

- Protocol Identification
- Protocol Decoding
- Exporting Fields



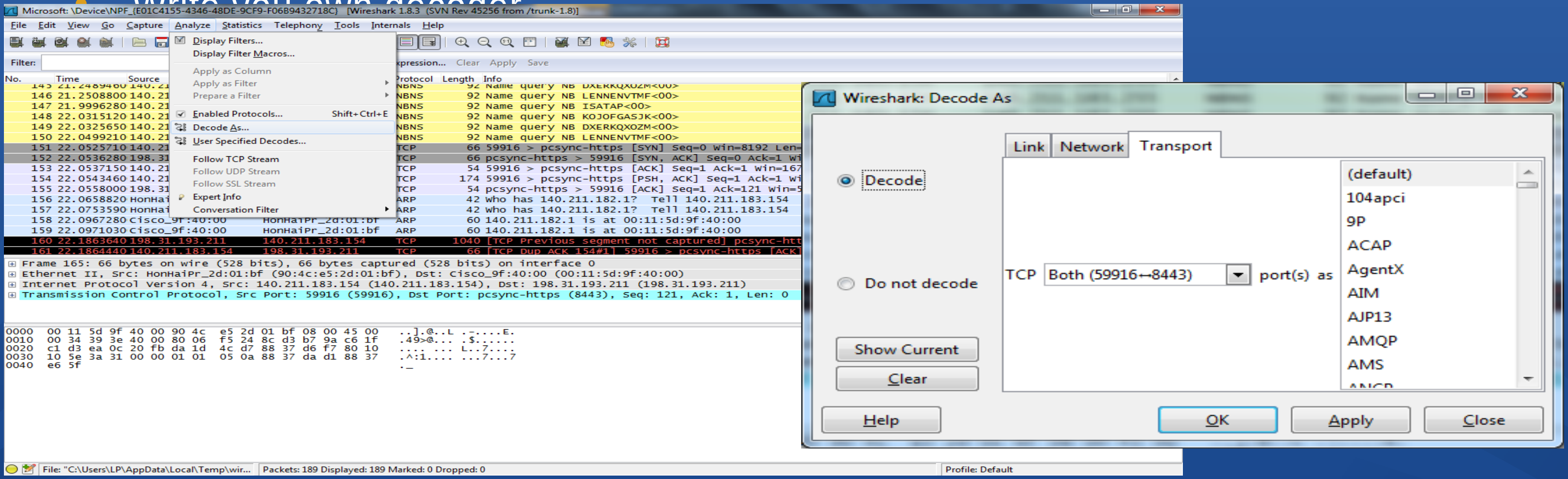
1. HTTP://RACHELSHADOAN.FILES.WORDPRESS.COM/2012/02/SENSEMAKING.PNG?W=490

PROTOCOL IDENTIFICATION

- Look for common binary/hex/ASCII values that are associated with specific protocols
  - Ex: 0x4500 marks the beginning of an IPv4 packet
- Use information in the encapsulating protocol
  - Ex: Byte 9 of the IP header indicates protocol, 0x06 corresponds with TCP
- Use port numbers for TCP/UDP
  - Ex: port 443 indicates TLS/SSL, check to see if packet is indeed encrypted
- Analyze the function of the src or dst server
  - Use IP address and do a WHOIS lookup
- Look for recognizable protocol structures
  - Refer to RFCs

PROTOCOL DECODING

- A way to interpret frame data based on known frame structure
- To use specific protocol specs
    - Use publically available automated decoders and tools
    - Manually decode traffic with publically available documentation
    - Write you own decoder



Microsoft: \Device\NPF_{E01C4155-4346-48DE-9CF9-F06B9432718C} [Wireshark 1.8.3 (SVN Rev 45256 from /trunk-1.8)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Display Filters...
Display Filter Macros...
Apply as Column
Apply as Filter
Prepare a Filter
Enabled Protocols... Shift+Ctrl+E
Decode As...
User Specified Decodes...
Follow TCP Stream
Follow UDP Stream
Follow SSL Stream
Expert Info
Conversation Filter
Filter:
Expression... Clear Apply Save
Wireshark: Decode As
Link
Network
Transport
Show Current
Clear
Help
OK
Apply
Close

```
No.    Time        Source          Protocol Length Info
146 21.2508800 140.21           NBNS       92 Name query NB LENNENVTMF<00>
147 21.9996280 140.21           NBNS       92 Name query NB ISATAP<00>
148 22.0315120 140.21           NBNS       92 Name query NB KOJOFGASJK<00>
149 22.0325650 140.21           NBNS       92 Name query NB DXERKQXOZM<00>
150 22.0499210 140.21           NBNS       92 Name query NB LENNENVTMF<00>
151 22.0525710 140.21           TCP        66 59916 > pcsync-https [SYN] Seq=0 Win=8192 Len=
152 22.0536280 198.31           TCP        66 pcsync-https > 59916 [SYN, ACK] Seq=0 Ack=1 Wi
153 22.0537150 140.21           TCP        54 59916 > pcsync-https [ACK] Seq=1 Ack=1 Win=167
154 22.0543460 140.21           TCP       174 59916 > pcsync-https [PSH, ACK] Seq=1 Ack=1 Wi
155 22.0558000 198.31           TCP        54 pcsync-https > 59916 [ACK] Seq=1 Ack=121 Win=5
156 22.0658820 HonHai           ARP        42 Who has 140.211.182.1?  Tell 140.211.183.154
157 22.0753590 HonHai           ARP        42 Who has 140.211.182.1?  Tell 140.211.183.154
158 22.0967280 Cisco_9f:40:00   HonHaiPr_2d:01:bf   ARP   60 140.211.182.1 is at 00:11:5d:9f:40:00
159 22.0971030 Cisco_9f:40:00   HonHaiPr_2d:01:bf   ARP   60 140.211.182.1 is at 00:11:5d:9f:40:00
160 22.1863640 198.31.193.211   140.211.183.154     TCP 1040 [TCP Previous segment not captured] pcsync-htt
161 22.1864440 140.211.183.154  198.31.193.211      TCP   66 [TCP Dup ACK 154#1] 59916 > pcsync-https [ACK]

⊞ Frame 165: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
⊞ Ethernet II, Src: HonHaiPr_2d:01:bf (90:4c:e5:2d:01:bf), Dst: Cisco_9f:40:00 (00:11:5d:9f:40:00)
⊞ Internet Protocol Version 4, Src: 140.211.183.154 (140.211.183.154), Dst: 198.31.193.211 (198.31.193.211)
⊞ Transmission Control Protocol, Src Port: 59916 (59916), Dst Port: pcsync-https (8443), Seq: 121, Ack: 1, Len: 0

0000  00 11 5d 9f 40 00 90 4c  e5 2d 01 bf 08 00 45 00   ..].@..L .-....E.
0010  00 34 39 3e 40 00 80 06  f5 24 8c d3 b7 9a c6 1f   .49>@... .$......
0020  c1 d3 ea 0c 20 fb da 1d  4c d7 88 37 d6 f7 80 10   .... ... L..7....
0030  10 5e 3a 31 00 00 01 01  05 0a 88 37 da d1 88 37   .^:1.... ...7...7
0040  e6 5f                                              ._

File: "C:\Users\LP\AppData\Local\Temp\wir...   Packets: 189 Displayed: 189 Marked: 0 Dropped: 0   Profile: Default
```

[✓ Decode] Do not decode

TCP Both (59916↔8443) port(s) as

(default)
104apci
9P
ACAP
AgentX
AIM
AJP13
AMQP
AMS

EXPORTING FIELDS

- Wireshark
  - “Export Selected Packet Bytes”
- Tshark
  - Example:
    - $ tshark -r evidence01.pcap -X lua_script:oft -tsk.lua -R "oft" -n -R frame. number ==112 -T pdml
    - $ tshark -r evidence.pcap -X lua_script:oft -tsk.lua -R "oft" -n -T fields –e "oft.filename" -e oft.totsize -R frame.number ==112
      - -e –T flags will show only specific fields

PACKET ANALYSIS

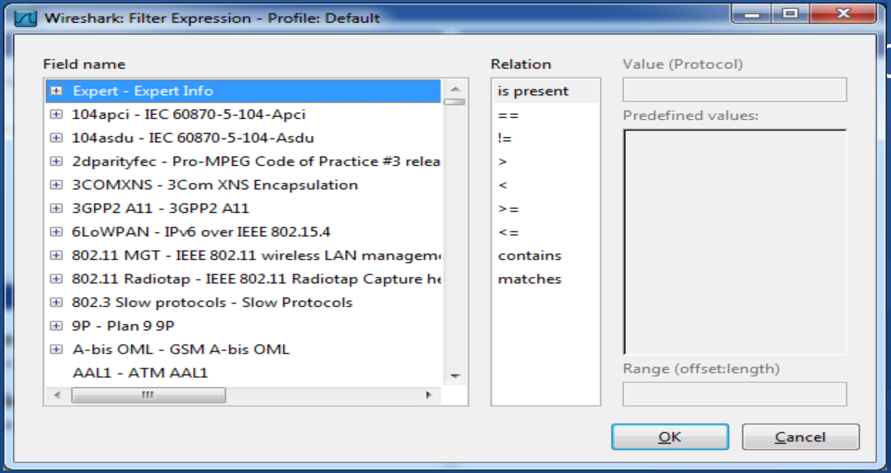
- Defined
    - “Packet Analysis—Examination of contents and/or metadata of one or more packets. Packet analysis is typically conducted in order to identify packets of interest and develop a strategy for flow analysis and content reconstruction.” (Davidoff & Ham, 2012)
-

PACKET ANALYSIS TOOLS

- Wireshark And Tshark Display Filters
- Ngreg
- Hex Editors

WIRESHARK AND TSHARK DISPLAY FILTERS

- Over 105,000 display filters
- Supports open plugin architecture
    - Build your own protocol parser
- “Expressions” button to build a filter of your choice
- Tshark uses –R for filters
    - Example:
        - $ tshark -r capturefile.pcap 0“
          28. “
-


Wireshark: Filter Expression - Profile: Default

Field name
- Expert - Expert Info
- 104apci - IEC 60870-5-104-Apci
- 104asdu - IEC 60870-5-104-Asdu
- 2dparityfec - Pro-MPEG Code of Practice #3 relea
- 3COMXNS - 3Com XNS Encapsulation
- 3GPP2 A11 - 3GPP2 A11
- 6LoWPAN - IPv6 over IEEE 802.15.4
- 802.11 MGT - IEEE 802.11 wireless LAN managem
- 802.11 Radiotap - IEEE 802.11 Radiotap Capture he
- 802.3 Slow protocols - Slow Protocols
- 9P - Plan 9 9P
- A-bis OML - GSM A-bis OML
- AAL1 - ATM AAL1

Relation
- is present
- ==
- !=
- >
- <
- >=
- <=
- contains
- matches

Value (Protocol)

Predefined values:

Range (offset:length)

NGREP

- Looks for packets based on particular string, binary sequences or patterns within the packet
- Recognizes common protocols: IP, TCP, UDP, and ICMP
- No flow reconstruction
    - Will not detect if data spans multiple packets
    - Detects matching packet not matching flow
- Example:
    - `$ ngrep -I capturefile.pcap "string to search for“`
    - `$ ngrep -I capturefile.pcap "string to search for" 'src host 192.168.1.20 and dst port 80'`

HEX EDITORS

- View and manipulate raw bits of data
- Indispensable for isolation of specific packet fragments and file carving
- Sometimes regular tools are not equipped to handle data
    - Example:
        - Loki tunneling protocol is often not recognized by tools like Wireshark
        - Most tools will not see inside compressed files
- Bless, Winhex, FTK Imager
-

PACKET ANALYSIS TECHNIQUES

- Pattern Matching
- Parsing Protocol Fields
- Packet Filtering

PATTERN MATCHING

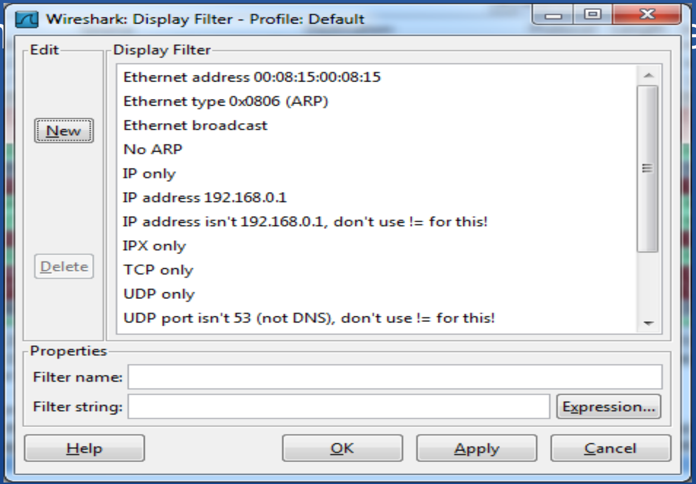
- “dirty word search”
  - List of strings, names, patterns that are related to suspect activity
- ngrep is the best tool for these searches
  - Example:
    - $ ngrep -I evidence01.pcap ‘words|search|for‘
    -

PARSING PROTOCOL FIELDS

- Application of extracting the contents of protocol fields within packets of interest.
- Example:
    - $ tshark -r evidence01.pcap -d tcp.port ==443 , aim -T fields -n -e "aim. messageblock.message“
- Good tshark reference
    - http://www.packetlevel.ch/html/tshark/tshark.html

PACKET FILTERING

- "…the art of separating packets based on the values of fields in protocol metadata or payload." (Davidoff & Ham, 2012)
- Use tcpdump with a BPF filter to dump out suspicious converstions
  - Example using IP addresses
    - $ tcpdump -s 0 -r evidence01.pcap -w evidence01 -talkers.pcap 'host 64.12.24.50 and host 192.168.1.158 ' Reading from file eviden et)
- Use Wireshark
  -


Wireshark: Display Filter - Profile: Default

Edit
New
Delete

Display Filter
Ethernet address 00:08:15:00:08:15
Ethernet type 0x0806 (ARP)
Ethernet broadcast
No ARP
IP only
IP address 192.168.0.1
IP address isn't 192.168.0.1, don't use != for this!
IPX only
TCP only
UDP only
UDP port isn't 53 (not DNS), don't use != for this!

Properties
Filter name:
Filter string:
Expression...

Help OK Apply Cancel

Works Cited

Davidoff, S., & Ham, J. (2012). *Network Forensics Tracking Hackers Through Cyberspace.* Boston: Prentice Hall.