


Processing Ethernet Flight Test Data with Open Source Tools

Paul Ferrill

CTO

Avionics Test and Analysis Corporation





Legacy Aircraft Flight Test – Avionics upgrade

- New data system utilizing IRIG 106 Chapter 10 recorder
- On-board real time display using IADS connecting to recorder over Ethernet
- ARINC-429, Ethernet, MIL-STD 1553, Video
- Documentation of all systems (LRUs) not initially provided

Data Interpretation

- Interface Control Document for data description
- Example: Encapsulated ARINC-429 data
- Data Format in Ethernet packet
 - 8 bytes consisting of 32-bit Data ID and 32-bit ARINC-429 data

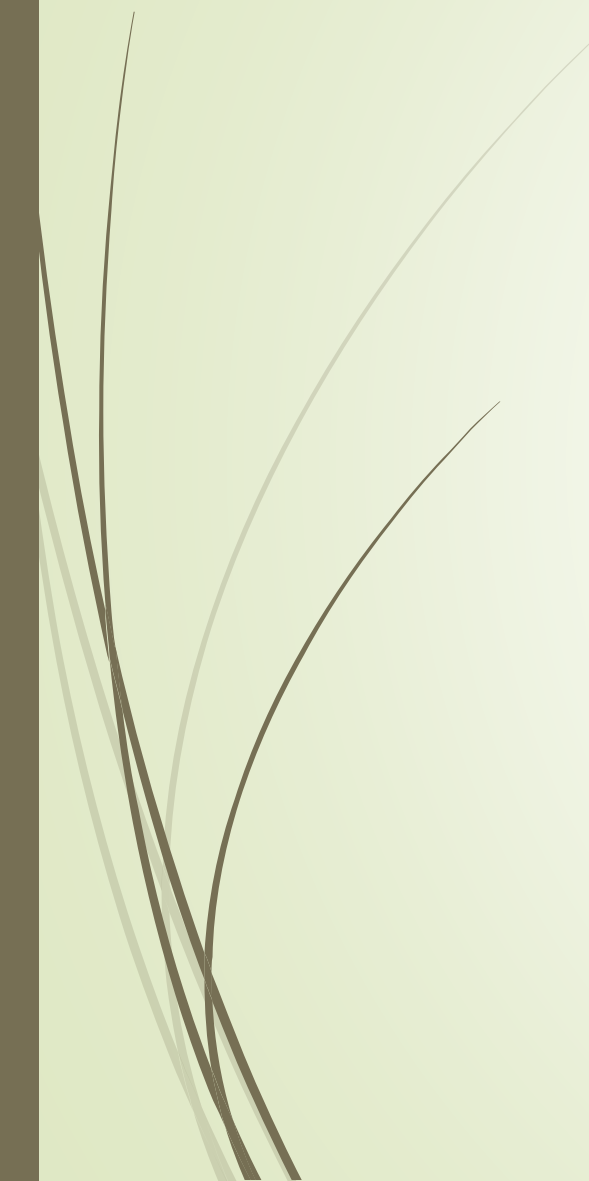
Mode	Label	DataID	Description
ALT	102	268d10	Altitude Reference
IAS	103	268d18	Indicated Airspeed Reference
VS	104	268d20	Vertical Speed Reference
MACH	106	268d30	Mach Reference

Label 103: FMS IAS Reference

Bit	Description
1-8	Octal Label=103
9-10	00 = All Call; 10 = Left Unit; 01 = Right Unit; 11 = Center Unit
11	Note [5]
12	Speed Reference Condition: 1=Manual Mode 0=Performance Mode
13	Caution Flag: 0=false 1=true (flag set)
14	0.062500 \
15	0.125 \
16	0.25 > Knots
17	0.5 /
18	1.0 /
28	1024.0 /
29	0 = Positive, 1 = Negative
30	0 Failure Warning 1 No Comp Data 0 Functional Test 1 Normal Op
31	0 0 1 1
32	Parity (Odd)



Wireshark for Ethernet Analysis

- Huge following with lots of tutorials
 - Customizable and programmable (lua)
 - Works in real-time monitor mode or offline processing of PCAP files
 - Cross platform (Windows, Mac, Linux)
 - Data filtering and export
- 

No.	Time	Source	Destination	Protocol	Length	Info
1	21:22:54.412	fe80::11e1:ffcc:ff02::1:ff93:b50	ff02::1:ff93:b50	ICMPv6	86	Neighbor solicitation for fe80::a478:7...
2	21:22:54.552	::	ff02::16	ICMPv6	130	Multicast Listener Report Message v2
3	21:22:54.552	::	ff02::16	ICMPv6	130	Multicast Listener Report Message v2
4	21:22:54.552	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0xfcda2...
5	21:22:54.555	100.45.155.1	255.255.255.255	DHCP	353	DHCP ACK - Transaction ID 0xfcda2...
6	21:22:54.563	fe80::1c7f:661a:ff02::2	ff02::2	ICMPv6	62	Router solicitation
7	21:22:54.799	100.45.155.126	111.221.77.172	TCP	62	2380→40021 [SYN] seq=0 win=8192 Len=0 M...
8	21:22:54.824	fe80::1c7f:661a:ff02::16	ff02::16	ICMPv6	110	Multicast Listener Report Message v2
9	21:22:54.845	100.45.155.126	157.56.100.135	TCP	62	2382→443 [SYN] seq=0 win=8192 Len=0 MS...
10	21:22:54.892	100.45.155.126	105.224.74.38	TCP	62	2383→33534 [SYN] seq=0 win=8192 Len=0 M...
11	21:22:54.955	100.45.155.126	157.56.96.207	TCP	62	2384→443 [SYN] seq=0 win=8192 Len=0 MS...
12	21:22:54.992	Portwell_49:4a:c	Broadcast	ARP	60	who has 100.45.155.55? Tell 100.45.15...
13	21:22:55.034	fe80::11d9:b450:ff02::1:2	ff02::1:2	DHCPv6	161	solicit XID: 0xf6a59f CID: 000100011b2...
14	21:22:55.046	fe80::1e99:4cff:ff02::16	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
15	21:22:55.066	2002:642d:9b7e::2002:c058:6301::	2002:c058:6301::	ICMPv6	82	Echo (ping) request id=0x0001, seq=6, l...
16	21:22:55.188	192.168.6.183	192.168.6.255	NBNS	92	Name query NB DRACEL-LAPTOP<1c>
17	21:22:55.283	100.45.155.126	134.170.104.176	TCP	62	2385→443 [SYN] seq=0 win=65535 Len=0 M...
18	21:22:55.283	Cisco_a9:97:0d	CDP/VTP/DTP/PaGp	CDP	456	Device ID: C08-IDF1B Port ID: FastEth...
19	21:22:55.356	Apple_e3:9a:38	Aironet_ff:ff:00	WLCCP	60	Ethernet II
20	21:22:55.392	100.45.155.126	24.249.194.6	TCP	62	2386→443 [SYN] seq=0 win=8192 Len=0 MS...
21	21:22:55.408	100.45.155.126	139.193.130.189	TCP	62	2387→61137 [SYN] seq=0 win=8192 Len=0 M...
22	21:22:55.423	::	ff02::1:ff43:6c6	ICMPv6	78	Neighbor solicitation for fe80::1c7f:6...
23	21:22:55.427	fe80::11e1:ffcc:ff02::1:ff93:b50	ff02::1:ff93:b50	ICMPv6	86	Neighbor solicitation for fe80::a478:7...
24	21:22:55.571	a8:66:7f:50:47:8	Broadcast	ARP	60	Gratuitous ARP for 100.45.155.36 (Requ...
25	21:22:55.577	a8:66:7f:50:47:8	Broadcast	ARP	60	who has 100.45.155.1? Tell 100.45.155...
26	21:22:55.577	a8:66:7f:50:47:8	Broadcast	ARP	60	who has 169.254.255.255? Tell 100.45.1...
27	21:22:55.597	::	ff02::16	ICMPv6	130	Multicast Listener Report Message v2

Wireshark 1.10.2 (SVN Rev 51934 from trunk-1.10.1)

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
4	17:13:59.995536	11.59.19.111	11.59.19.255	UDP	102	Source port: 54371 Destination port: wsm-server
5	17:13:59.999591	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=0, ID=3efc) [Reassembled in #15]
6	17:13:59.999606	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=1480, ID=3efc) [Reassembled in #15]
7	17:13:59.999612	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=2960, ID=3efc) [Reassembled in #15]
8	17:13:59.999618	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=4440, ID=3efc) [Reassembled in #15]
9	17:13:59.999624	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=5920, ID=3efc) [Reassembled in #15]
10	17:13:59.999633	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=7400, ID=3efc) [Reassembled in #15]
11	17:13:59.999639	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=8880, ID=3efc) [Reassembled in #15]
12	17:13:59.999645	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=10360, ID=3efc) [Reassembled in #15]
13	17:13:59.999651	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=11840, ID=3efc) [Reassembled in #15]
14	17:13:59.999657	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=13320, ID=3efc) [Reassembled in #15]
15	17:13:59.999663	11.59.19.111	11.59.19.255	UDP	882	Source port: 54371 Destination port: wsm-server
16	17:14:00.001684	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=0, ID=3efd) [Reassembled in #26]
17	17:14:00.001710	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=1480, ID=3efd) [Reassembled in #26]
18	17:14:00.001718	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=2960, ID=3efd) [Reassembled in #26]
19	17:14:00.001725	11.59.19.111	11.59.19.255	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=4440, ID=3efd) [Reassembled in #26]

Frame 15: 882 bytes on wire (7056 bits), 882 bytes captured (7056 bits)

Ethernet II, Src: Dell_44:37:4c (00:1d:09:44:37:4c), Dst: Broadcast (ff:ff:ff:ff:ff:ff)

Internet Protocol Version 4, Src: 11.59.19.111 (11.59.19.111), Dst: 11.59.19.255 (11.59.19.255)

- Version: 4
- Header length: 20 bytes
- Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
- Total Length: 868
- Identification: 0x3efc (16124)
- Flags: 0x00
- Fragment offset: 14800
- Time to live: 128
- Protocol: UDP (17)
- Header checksum: 0xb36f [correct]
- Source: 11.59.19.111 (11.59.19.111)
- Destination: 11.59.19.255 (11.59.19.255)
- [Source GeoIP: Unknown]
- [Destination GeoIP: Unknown]
- [11 IPv4 Fragments (15648 bytes): #5(1480), #6(1480), #7(1480), #8(1480), #9(1480), #10(1480), #11(1480), #12(1480), #13(1480), #14(1480), #15(848)]

User Datagram Protocol, Src Port: 54371 (54371), Dst Port: wsm-server (5006)

- Source port: 54371 (54371)
- Destination port: wsm-server (5006)
- Length: 15648

```

0000  d4 63 13 8e 3d 20 17 71 01 15 3e 00 25 eb 0c 00  .C..=.q .>.%..
0010  14 3d 00 00 f8 3c 00 00 02 12 03 40 b2 a4 34 c4  =...<...@..4.
0020  74 00 9c 20 00 00 00 00 00 47 15 21 e1 60 3a 51  t.. .... .G.! :Q
0030  63 a0 fb ff 35 a8 da b5 7b 5b 99 b6 b4 1c 6e 35  c...5... { [...n5
0040  bd f7 10 c6 41 c5 b7 51 52 10 37 65 04 33 a9 22  ...A.Q R.7e.3."
0050  ab d2 5b a8 7f 17 5d b5 eb 4b 4c 4f aa 81 72 4f  ..[...] .KLO..rO
0060  f3 cd 20 35 00 00 00 00 01 00 4b 18 40 96 e1 f3  .. 5.... .K.@...
0070  0f b9 a5 86 0e 59 f1 b9 10 6c 39 bf 54 60 08 35  ....Y...19.T.5
0080  d0 1b 71 10 fc a4 b0 41 4e fc 54 bb 28 0c a4 74  ..q...A N.T(.t
0090  35 30 b9 1d 38 5f 41 f6 3f 39 ad 70 82 16 52 62  50..8_A. ?9.p..Rb
  
```

Frame (882 bytes) Reassembled IPv4 (15648 bytes)

Data (data.data), 15640 bytes Packets: 80527 · Displayed: 80527 (100.0%) · Load time: 0:00.939 Profile: Classic

11 Fragments
reassembled in
Pkt #15

Follow UDP Stream

- Choose a packet first
- Launch from Analyze
- Default display Raw
- Save

The screenshot shows the 'Follow UDP Stream' window in Wireshark. The main area displays a hex dump of network data, with each line representing a packet's raw bytes. The hex values are shown in pairs, and the corresponding ASCII characters are shown to the right. The data appears to be a sequence of network-related information, possibly including IP addresses and port numbers, though the ASCII representation is mostly garbled. At the bottom of the window, there are buttons for 'Save As', 'Print', and radio buttons for selecting the display format: ASCII, EBCDIC, Hex Dump (selected), C Arrays, and Raw. There are also buttons for 'Filter Out This Stream' and 'Close'.

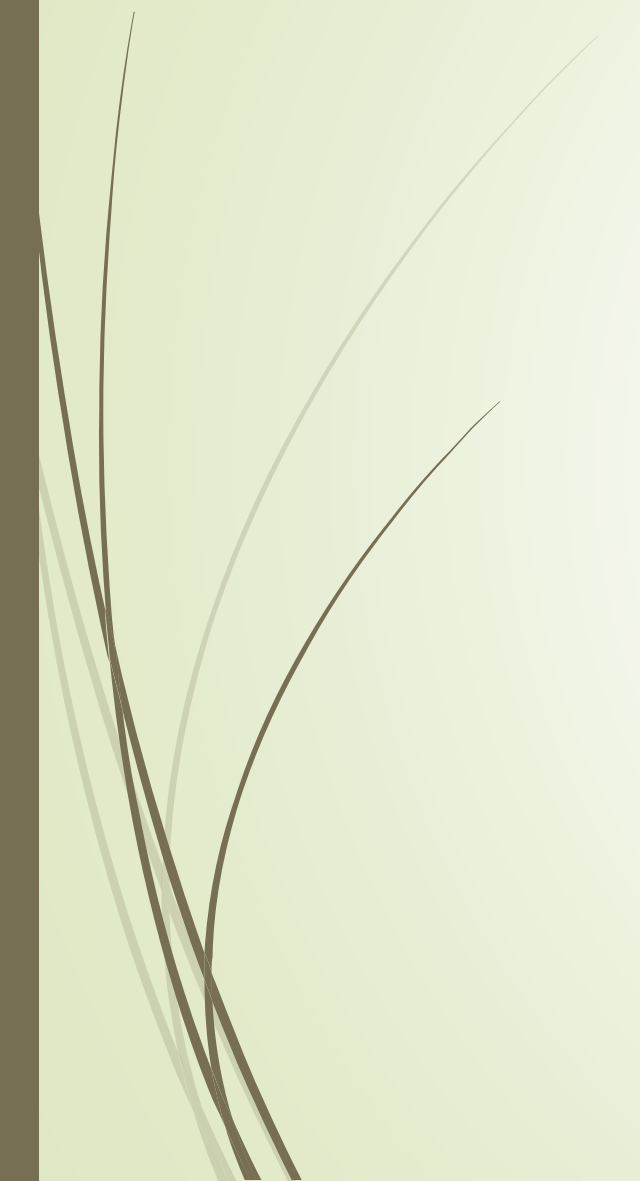
The screenshot shows the 'Filtering' dialog box in Wireshark. The title bar indicates the current filter: '3% of Filtering: (ip.addr eq 10.146.10.2 and ip.addr eq 239.1.15.62) and (udp.port eq 49...'. The main text area shows the active filter: 'Filtering: (ip.addr eq 10.146.10.2 and ip.addr eq 239.1.15.62) and (udp.port eq 49860 and udp.port eq 9006)'. Below this, the status is 'Status: 308576 of 7714479 frames'. The elapsed time is 'Elapsed Time: 00:09' and the time left is 'Time Left: --:--'. A progress bar shows 'Progress: 3%' with a green bar indicating the current progress. A 'Stop' button is located at the bottom right of the dialog.

Save file output

```
00000000 02 00 00 00 81 cb 5e 40 02 01 06 0b 83 52 60 60 .....^@ .....R` `
00000010 6d 18 90 0c 81 cb 5d 40 e0 1a e2 73 81 cb 58 40 m.....]@ ...s..X@
00000020 60 00 02 24 81 cb 60 40 80 00 01 6b 83 52 68 60 `..$.`@ ...k.Rh`
00000030 eb 59 90 8c 83 52 70 60 ee 1c 10 4c 83 52 78 60 .Y...Rp` ...L.Rx`
00000040 ec 58 10 cc .X..
00000044 02 00 00 00 81 cb 5c 40 7f e7 a2 b3 81 cb 59 40 ..... \@ .....Y@
00000054 e0 00 02 a4 81 cb 6e 40 60 1f 82 53 83 52 80 60 .....n@ `..S.R.`
00000064 ec 16 90 2c 83 52 88 60 6d 98 10 ac 83 57 dc 60 ..., .R.` m....W.`
00000074 00 00 00 fd 81 cb 69 40 80 00 02 1f .....i@ ....
00000080 02 00 00 00 81 ca b3 40 92 fc 02 ef 81 ca b3 40 .....@ .....@
00000090 83 00 02 ef 81 ca a8 40 00 00 00 83 81 ca a7 40 .....@ .....@
000000A0 00 20 00 03 81 ca 9c 40 9b 41 85 86 81 ca a0 40 . .....@ .A.....@
000000B0 e0 dc 00 49 81 ca b1 40 e0 00 00 af 83 59 08 60 ...I...@ .....Y.`
000000C0 e0 00 00 44 81 ca b5 40 65 8c 02 04 81 ca c5 40 ...D...@ e.....@
000000D0 66 00 02 06 81 ca cd 40 65 9a 02 05 81 ca d5 40 f.....@ e.....@
000000E0 e5 90 02 07 81 ca b6 40 e4 e4 02 84 81 ca c6 40 .....@ .....@
000000F0 64 e2 02 86 83 59 05 60 e0 60 04 46 81 ca ce 40 d....Y.` .`.F...@
00000100 64 de 02 85 81 ca d6 40 64 e6 02 87 83 59 06 60 d.....@ d....Y.`
```



Why Python

- Open source
 - Cross platform
 - Lots of tutorials – backed by Google
 - Huge number of libraries
 - Interpreted environment
 - Great first language
 - Object based
- 



Libraries

- dpkt

 - Google Code:

 - <https://code.google.com/p/dpkt/>

- socket

- binascii

- struct

- datetime



Data Manipulation



- Pandas

- <http://pandas.pydata.org/>

- Matplotlib

- <http://matplotlib.org/>

- Numpy

- <http://www.numpy.org/>

- Scientific Python

- <http://scipy.org/>

Examine Raw Packet Data

Example Data

```
0000  01 00 5e 01 0a 5b 02 00 00 01 02 08 08 00 45 00
0010  00 a0 bd 3f 00 00 40 11 b1 1c 0a 92 08 03 ef 01 - 10.145.8.3
0020  0a 5b 5d 8a 24 1e 00 8c 00 00 02 00 00 00 26 8f - 02 = data marker
0030  c0 40 00 00 00 19 26 8f 00 40 82 00 44 01 26 8f
0040  40 40 00 40 44 11 26 8f 80 40 92 0a 28 09 26 8f
0050  c8 40 80 00 00 99 26 8f 18 40 00 00 00 c1 26 91
0060  40 40 7f fb a0 13 26 91 48 40 7f fa fc 93 26 91
0070  00 40 7f f0 a0 03 26 91 08 40 ff f0 a0 83 26 91
0080  b8 40 e0 00 00 eb 26 8d 20 40 60 00 04 22 26 8d
0090  18 40 00 00 00 c2 26 8d 30 40 60 b4 00 62 26 8d - 268d10 NDO
00a0  10 40 6e a6 00 42 26 8d 07 40 7f f0 a0 26 78 d8 - 32 bits of data
00b0  1d e6
```



Walk thru Packets

```
pcapReader = dpkt.pcap.Reader(file(filename, 'rb'))  
for ts, data in pcapReader:  
    ether = dpkt.ethernet.Ethernet(data)  
    if ether.type == dpkt.ethernet.ETH_TYPE_IP:  
        ip = ether.data  
        tcp = ip.data  
        src = socket.inet_ntoa(ip.src)
```

Process Packets

```
if src == addr:
    a = binascii.hexlify(str(tcp.data))
    if a[0:2] == '02':
        d10 = a.find('268d10')
        d10val = int(a[d10+9:d10+13],16)
        if d10val != lastd10:
            print "%s %s d10 = %s" % (secstosting(ts), src, d10val)
            lastd10 = d10val
        d18 = a.find('268d18')
        d18val = int(a[d18+9:d18+13],16) * 0.0625
```



Packet Utilities

▶ PCAP Replay

▶ PlayCap

▶ <http://www.signal11.us/oss/playcap/>

▶ TCP Replay

▶ <http://tcpreplay.synfin.net/>

▶ Scapy

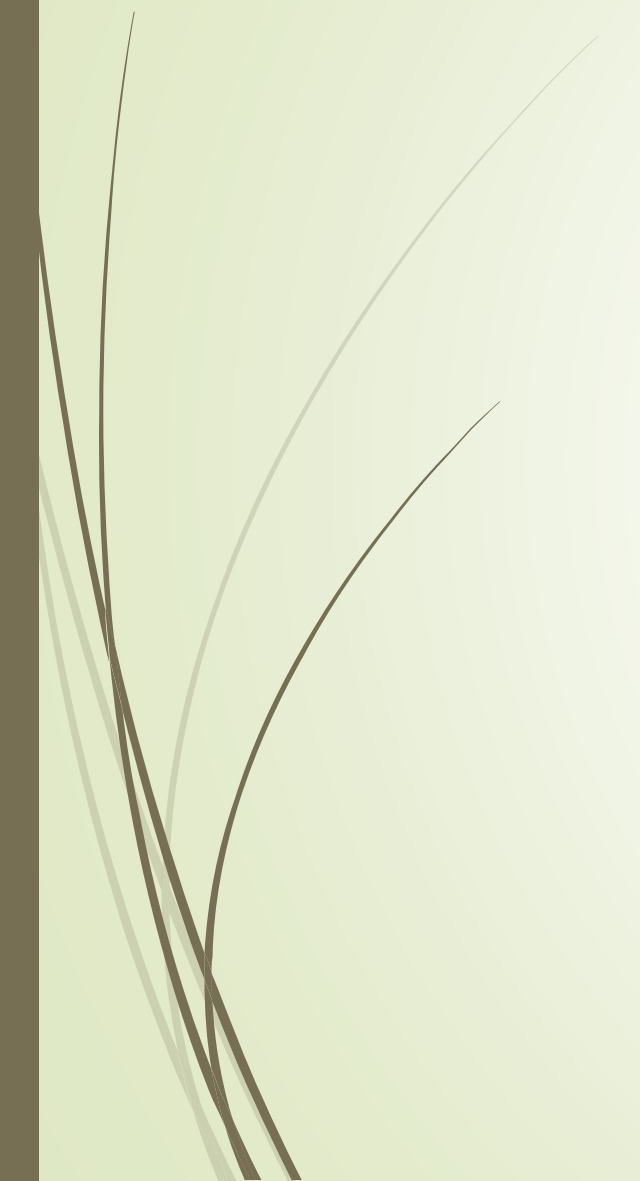
▶ <http://www.secdev.org/projects/scapy/>

▶ Ngrep

▶ <http://ngrep.sourceforge.net/>



Colasoft

- Offer a number of free tools
 - <http://www.colasoft.com/products/freeware.php>
 - Colasoft Packet Player
 - http://www.colasoft.com/packet_player/
 - Capsa Free
 - Colasoft MAC Scanner
 - Colasoft Ping Tool
 - Colasoft Packet Builder
- 



SolarWinds

- Lots of free and trial tools:
 - <http://www.solarwinds.com/downloads/>
- IP Address Manager
 - Free and trial version
- Subnet Calculator



Chapter 10 Tools

- IRIG106.org tools
- EMC Packet Viewer, Chapter 10 Validator
- Common Mission Debrief Program (CMDP)



Questions?