SharkFest'23 US
San Diego, CA • June 10-15

#sf23us

"I wish Wireshark" - add the missing pieces with Lua

Chuck Craft
Wireshark Core

set_plugin_info(presenter_info)

```
local presenter_info =
{
    version = "SF23US San Diego",
    author = "Chuck Craft",
    description = "Wireshark Core",
    repository = "https://www.linkedin.com/in/cpu4coffee"
}
```

(yep - that's a valid version string. Try it in your Lua code.)

Example Lua tasks/solutions

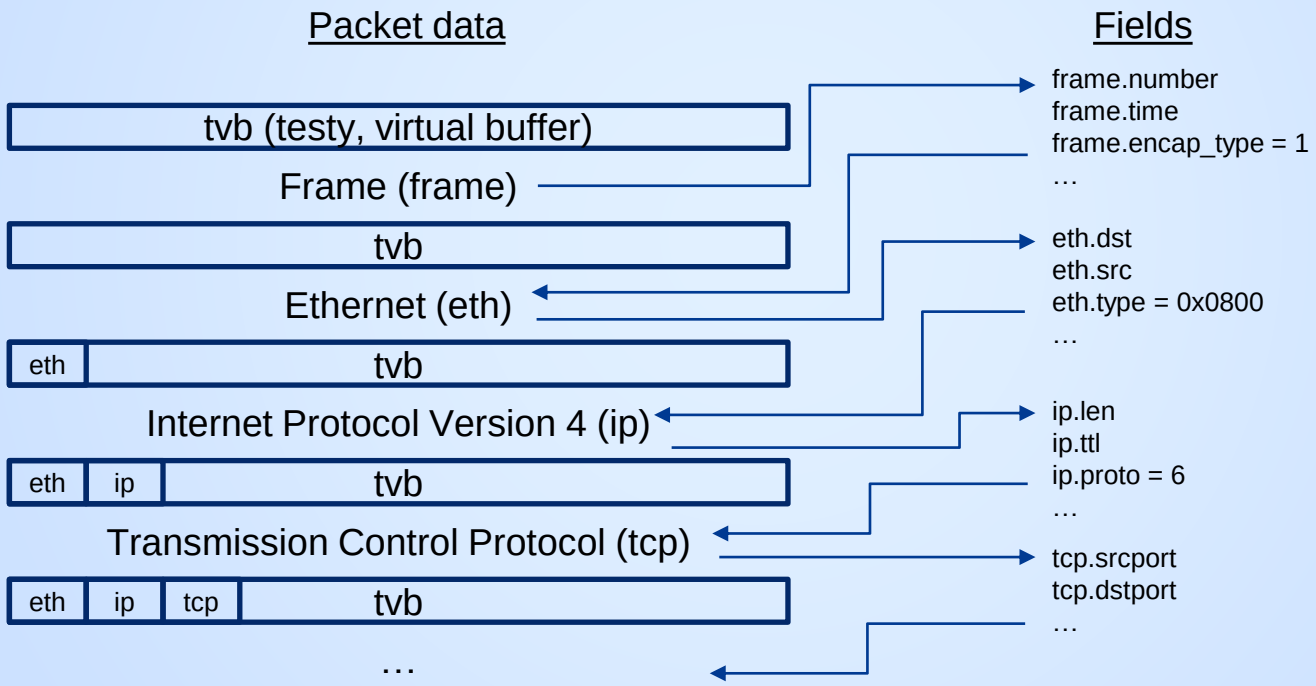
Dissectors

- An existing field in a different format
- New fields
- Dissecting an unsupported protocol

Taps/Listeners

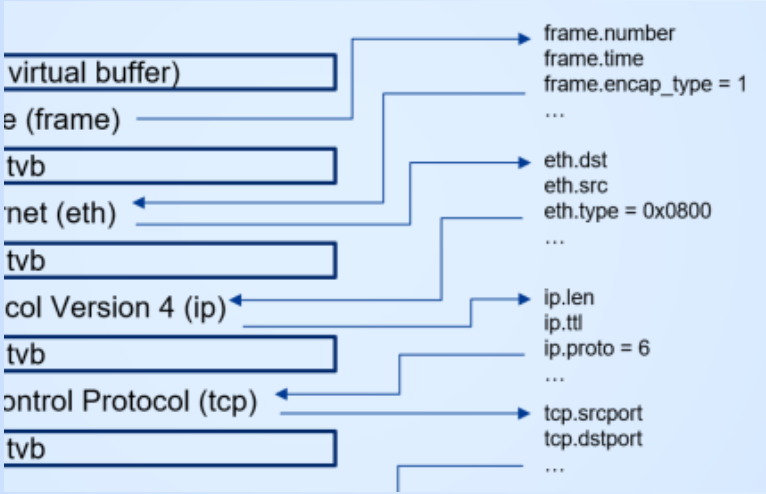
- Relate data across multiple packets
- Custom statistics
- Add menu items/utilities

Dissector handoff



DRAFT

Lua - Dissector



Dissectors are meant to analyze some part of a packet's data. Only get called when the packet matches or when the user forces it using "Decode As".

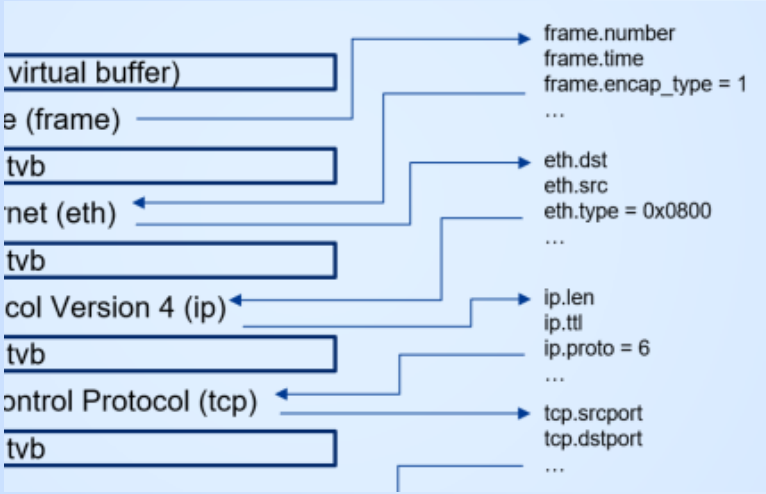
View->Internals->Dissector Tables

wtap_encap: 1 = Ethernet
ethertype: 0x0800 = IPv4
ip.proto: 6 = TCP

dissectortable:add(pattern, dissector)
proto:register_heuristic(listname, func)

DRAFT

Lua - Post-dissector



Post-dissector runs after all other dissectors

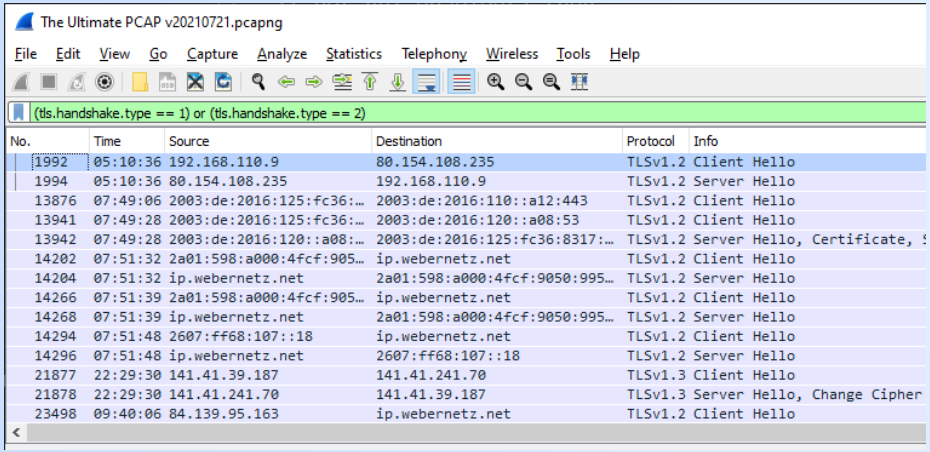
A dissector registered to be called after every other dissector has been called. These are handy as all protocol fields are available so they can be accessed, and they can add items to the dissection tree (Packet Details).

Not very efficient but easy to implement.

register_postdissector(proto, [allfields])

DRAFT

Lua - Listener (Tap)



local tap = Listener.new("tls", "(tls.handshake.type == 1) or (tls.handshake.type == 2)")

A `Listener` is called once for every packet that matches a certain filter or has a certain tap. It can read the tree, the packet's Tvb buffer as well as the tapped data, but it cannot add elements to the tree.

Called once every few seconds to redraw the GUI objects; in TShark this function is called only at the very end of the capture file.

DRAFT

Listener.new([tap], [filter], [allfields])

Existing field in a different format

- arp_host.lua
- EASYPOST.lua template

arp_host.lua

```
3 0.110617  cpe-24-166-172-1.kc.res.rr.com      Broadcast   ARP   60    24.166.173.161
4 0.211791  cpe-65-28-78-1.kc.res.rr.com        Broadcast   ARP   60    65.28.78.76

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface unknow
> Ethernet II, Src: Cisco251_af:f4:54 (00:07:0d:af:f4:54), Dst: Broadcast (ff:ff:ff:ff:f
v Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: cpe-65-28-78-1.kc.res.rr.com (00:07:0d:af:f4:54)
    Sender IP address: cpe-24-166-172-1.kc.res.rr.com (24.166.172.1)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: cpe-24-166-173-159.kc.res.rr.com (24.166.173.159)

Target IP address (arp.dst.proto_ipv4), 4 bytes
```

DRAFT

220703_arp-storm.pcapng (https://wiki.wireshark.org/SampleCaptures#arp-rarp)

arp_host.lua

"Is there a display filter that can be used to apply as column, the resolved or mapped host name for an ARP target IP address?

This string value is shown in the packet details window."

https://ask.wireshark.org/question/22016/resolved-or-mapped-arp-target-ip-address/

arp_host.lua

Source field:

```
local arp_target_f = Field.new("arp.dst.proto_ipv4")
```

Destination field:

```
target_host = ProtoField.string("arp_host.target",
                                "ARP target (resolved)")
```

Transformation:

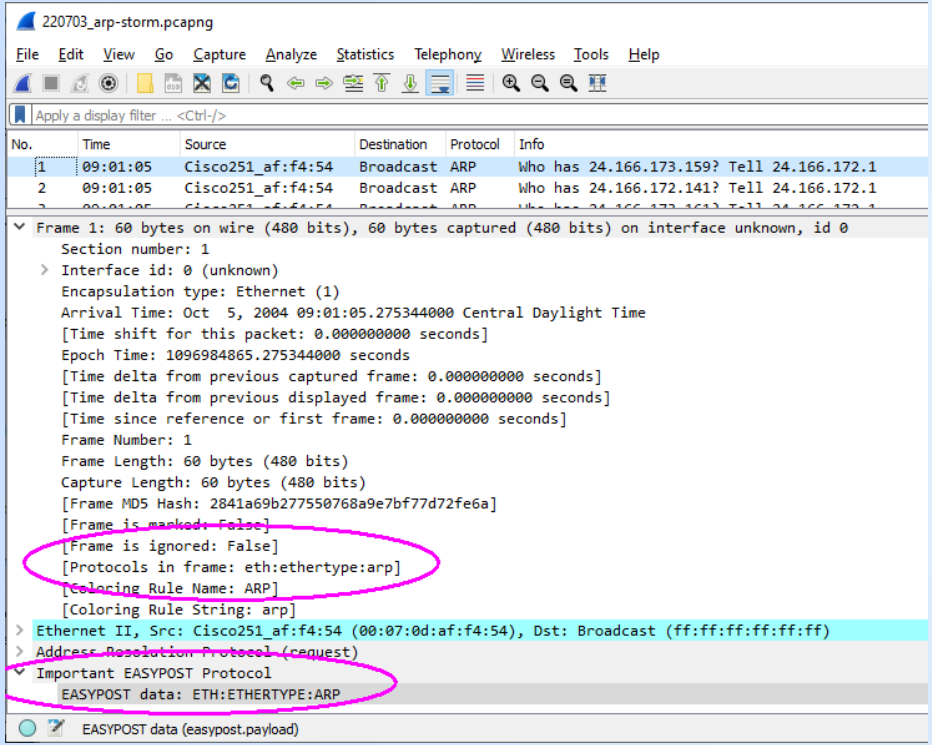
```
subtree:add(pf.target_host, v.display)
```

```
WSLUARM: fieldinfo.display
         "The string display of this field as seen in GUI."
```

DRAFT

Download: https://wiki.wireshark.org/Contrib -> Post-Dissectors

EASYPOST.lua template



`EASYPOST.lua` is a starting point (cookbook steps) for Lua plugins.

Out of the box, it builds a new field – uppercase of `frame.protocols`.

https://wiki.wireshark.org/lua#examples

1. Download / save to
   Personal Lua Plugins folder
2. Analyze -> Reload Lua Plugins

DRAFT

EASYPOST.lua template

Source field:

```
easypost_payload_f = Field.new("frame.protocols")
```

Destination field:

```
payload = ProtoField.string("easypost.payload", "EASYPOST data")
```

Transformation:

```
local field_data = string.format("%s", v):upper()
subtree:add(pf.payload, field_data)
```

Programming in Lua: (https://www.lua.org/pil/20.html)
20 – The String Library

Download: https://wiki.wireshark.org/lua#examples

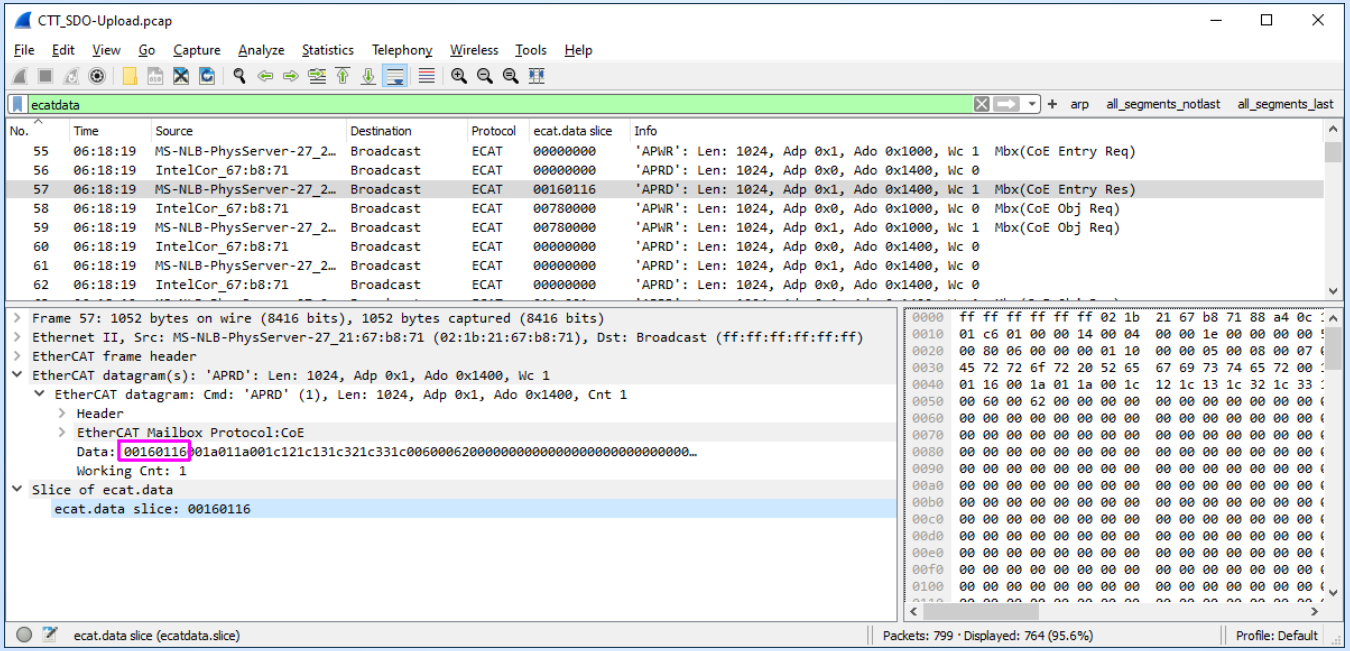
DRAFT

New fields

- ecatdata.lua - https://ask.wireshark.org/question/27207/how-to-display-slice-as-a-filter-in-column/
- filtcols -- https://wiki.wireshark.org/Lua/Examples/filtcols/

ecatdata.lua

#sf23us



DRAFT

CTT_SDO-Upload.pcap (https://gitlab.com/wireshark/wireshark/-/issues/11652)

ecatdata.lua

‘I want to add a column with displaying 4 bytes form ethercat data: eacat.data[0:4]

For some reason filter "ecat.data[0:4]" is not work.

I found that filter "ecat.data[0:4] & 0xff" is works, but only if I use it as a normal filter.

I can't set this filter as cloumn.’

https://ask.wireshark.org/question/27207/how-to-display-slice-as-a-filter-in-column/

ecatdata.lua

Source field:

```
ecatdata_f = Field.new("ecat.data")
```

Destination field:

```
payload = ProtoField.string("ecatdata.slice", “ecat.data slice")
```

Transformation:

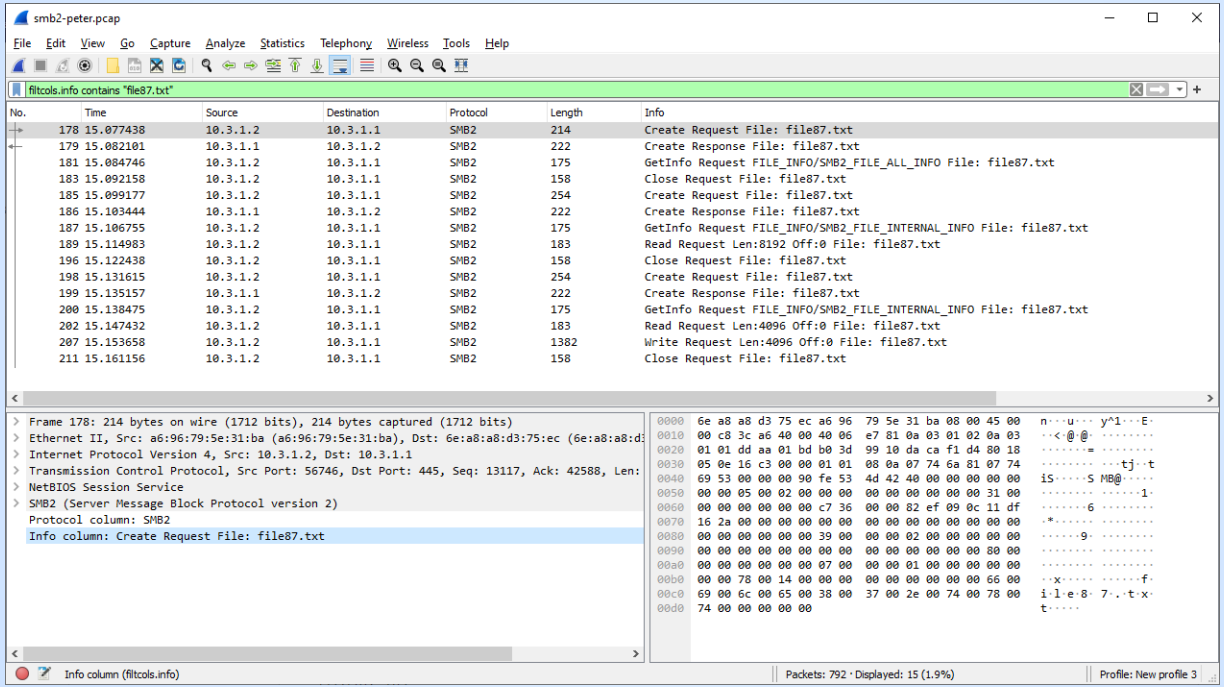
```
local slicelen = 4
if (v.len < slicelen) then
    slicelen = v.len
end
local field_data = string.format("%s", v.range(0,slicelen))
tree:add(pf.payload, field_data)
```

```
WSLUARM: fieldinfo.range
            “The TvbRange covering the bytes of this field in a Tvb
                        or nil if there is none.”
```

Download: code in Ask question answer/comments

DRAFT

filtcols



https://wiki.wireshark.org/Lua/Examples/filtcols/

DRAFT

filtcols

Reading from columns is “wonky”.

Cache values for later use after dissector pass 2.

Why does column check fail and return "(%s)“ after first couple dissection passes?

https://gitlab.com/wireshark/wireshark/-/blob/master/epan/wslua/wslua_column.c#L122

https://gitlab.com/wireshark/wireshark/-/blob/master/epan/wslua/wslua.h#L359

filtcols

```
-- variables to persist across all packets
local pkt_data = {} -- indexed per packet

pkt_data.protocol = {}
pkt_data.info = {}
```

```
-- let's do it!
function filtcols_p.dissector(tvb,pinfo,tree)
    local cols_protocol = tostring(pinfo.cols.protocol)

    if  cols_protocol ~= "(protocol)" then
        pkt_data.protocol[pinfo.number] = cols_protocol
    end

    local pkt_proto = pkt_data.protocol[pinfo.number]

    if  pkt_proto ~= nil then
        tree:add(col_protocol_field, pkt_proto)
    end
```

DRAFT

Dissecting an unsupported protocol

- MYSTERY_PKT.lua
- Malfored_MyRoom.pcapng

DRAFT

MYSTERY_PKT.lua

```
4500 0034 8bfd 4000 8006 1068 c0a8 6e83
c0a8 6e8a 081a 01f6 41d2 eac6 e115 3ace
5018 fcc6 0032 0000 00d1 0000 0006 0103
0001 0001
```

Practical Packet Analysis, 3rd Edition - Chris Sanders
(https://nostarch.com/packetanalysis3)
Navigating a Mystery Packet 330

DRAFT

MYSTERY_PKT.lua

Add Ethernet header

No Offsets

```
45 00 00 34 8b fd 40 00 80 06 10 68 c0 a8 6e 83
c0 a8 6e 8a 08 1a 01 f6 41 d2 ea c6 e1 15 3a ce
50 18 fc c6 00 32 00 00 00 d1 00 00 00 06 01 03
00 01 00 01
```

Add spaces between bytes

Import from Hex Dump…
- Offsets: None
- Encapsulation Type: Ethernet
- Ethernet header - Ethertype (hex): 0800

DRAFT

MYSTERY_PKT.lua

```
45 00 00 34 8b fd 40 00 80 06 10 68 c0 a8 6e 83
c0 a8 6e 8a – IP header (20 bytes)

08 1a 01 f6 41 d2 ea c6 e1 15 3a ce 50 18 fc c6
00 32 00 00 – TCP header (20 bytes)

00 d1 00 00 00 06 01 03 00 01 00 01 – TCP payload
                                      (12 bytes)
```

MYSTERY_PKT.lua

Source field:

```
mystery_payload_f = Field.new("tcp.payload")
```

Destination fields:

```
local pf = {
        payload = ProtoField.string("mystery.payload", "Mystery Packet data"),
        tid = ProtoField.uint16("mystery.tid", "Mystery - Transaction Identifier", base.HEX),
        pid = ProtoField.uint16("mystery.pid", "Mystery - Protocol Identifier", base.HEX),
        length = ProtoField.uint16("mystery.length", "Mystery - Length", base.HEX),
        uid = ProtoField.uint8("mystery.uid", "Mystery - Unit Identifier", base.HEX),
        fcode = ProtoField.uint8("mystery.fcode", "Mystery - Function Code", base.HEX),
        variable = ProtoField.uint32("mystery.fcode", "Mystery - Remainder", base.HEX)
}
```

DRAFT

MYSTERY_PKT.lua

Transformation:

```
local field_data = string.format("%s", v):upper()
subtree:add(pf.payload,   v.range, field_data)
subtree:add(pf.tid,       v.range(0,2))
subtree:add(pf.pid,       v.range(2,2))
subtree:add(pf.length,    v.range(4,2))
subtree:add(pf.uid,       v.range(6,1))
subtree:add(pf.fcode,     v.range(7,1))
subtree:add(pf.variable,  v.range(8,4))
```

WSLUARM: fieldinfo.range

"The TvbRange covering the bytes of this field in a Tvb or nil if there is none."

Malfored_MyRoom.pcapng

#sf23us

*Malfored_MyRoom.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Payload	Info
1	0.000000	local_ip	remote_ip	UDP	77	000000000003df5900000001ddb9ad826f00000013000fef00a7c4f1234a4e8a201a0e	60652 → 13400 Len=35[Malformed Packet]
2	0.030947	remote_ip	local_ip	UDP	61	51000000130010ef00a7c4f1234a4e8a201a0e	13400 → 60652 Len=19[Malformed Packet]
3	6.004145	local_ip	remote_ip	UDP	77	000000000003df5900000001ddb9ad826f00000013000f0b5da74bf1234a4e8a203283	60652 → 13400 Len=35[Malformed Packet]
4	6.035878	remote_ip	local_ip	UDP	61	510000001300100b5da74bf1234a4e8a203283	13400 → 60652 Len=19[Malformed Packet]
5	12.006304	local_ip	remote_ip	UDP	77	000000000003df5900000001ddb9ad826f00000013000f2f043bb8f1234a4e8a202b15	60652 → 13400 Len=35[Malformed Packet]
6	12.035948	remote_ip	local_ip	UDP	61	510000001300102f043bb8f1234a4e8a202b15	13400 → 60652 Len=19[Malformed Packet]
7	18.004144	local_ip	remote_ip	UDP	77	000000000003df5900000001ddb9ad826f00000013000fd713b92ff1234a4e8a2043a0	60652 → 13400 Len=35[Malformed Packet]
8	18.032389	remote_ip	local_ip	UDP	61	51000000130010d713b92ff1234a4e8a2043a0	13400 → 60652 Len=19[Malformed Packet]
9	24.013709	local_ip	remote_ip	UDP	77	000000000003df5900000001ddb9ad826f00000013000f6e833640f1234a4e8a20743a	60652 → 13400 Len=35[Malformed Packet]
10	24.042595	remote_ip	local_ip	UDP	61	510000001300106e833640f1234a4e8a20743a	13400 → 60652 Len=19[Malformed Packet]

https://discord.com/channels/889214182837321788/1007722817736945795/1082900474707050556

(Wireshark Discord – pcap-help - 03/07/2023 11:39 PM

Malfored_MyRoom.pcapng

Chuckc 03/08/2023 10:24 AM

Looks like a six second heartbeat, echoing back the message sent by the client.

```
1    0.000000    000000000003df5900000001ddb9ad826f00000013000f  ef00a7c4f1234a4e8a201a0e
2    0.030947    51000000130010                                  ef00a7c4f1234a4e8a201a0e

3    6.004145    000000000003df5900000001ddb9ad826f00000013000f  0b5da74bf1234a4e8a203283
4    6.035878    51000000130010                                  0b5da74bf1234a4e8a203283

5    12.006304    000000000003df5900000001ddb9ad826f00000013000f  2f043bb8f1234a4e8a202b15
6    12.035948    51000000130010                                  2f043bb8f1234a4e8a202b15

7    18.004144    000000000003df5900000001ddb9ad826f00000013000f  d713b92ff1234a4e8a2043a0
8    18.032389    51000000130010                                  d713b92ff1234a4e8a2043a0

9    24.013709    000000000003df5900000001ddb9ad826f00000013000f  6e833640f1234a4e8a20743a
10    24.042595    51000000130010                                  6e833640f1234a4e8a20743a
```

https://discord.com/channels/889214182837321788/1007722817736945795/1083062717692260382

(Wireshark Discord - pcap-help - — 03/08/2023 10:24 AM

Tap/Listener vs Dissector

dissector:call(tvb, pinfo, tree)

A dissector can update the tree but has no corresponding draw() to update the GUI.

tap.packet(pinfo,tvb,tapinfo)

A listener can't update the tree but has a listener.draw() that will be called once every few seconds to redraw the GUI objects.

Relate data across multiple packets

- tls_conversations.lua

tls_conversations.lua

‘I see I can filter "tls.handshake.type == 1" for Client Hello and "tls.handshake.type == 2" for server hello.

I have server side capture and I want to filter all the TCP stream which has "Client Hello" but no "Server Hello" response back.’

https://ask.wireshark.org/question/26618/filter-tls-with-no-server-hello/

tls_conversations.lua

Wireshark · TLS Conversations

```
TCP_stream  Client_hello          Server_hello
---------------------------------------------
6           1992                  1994
91          13876                 *****
93          13941                 13942
113         14202                 14204
116         14266                 14268
117         14294                 14296

TCP stream #s missing handshake: 91
```

Highlight:

Close

DRAFT

The Ultimate PCAP v20221220.pcapng (https://weberblog.net/the-ultimate-pcap/)
Screenshot of export trimmed to first 20000 packets

tls_conversations.lua

Source fields:

```
    local tls_handshake_type_f = Field.new("tls.handshake.type")
    local tcp_stream_f = Field.new("tcp.stream")
```

Download: https://wiki.wireshark.org/Contrib -> Post-Dissectors

DRAFT

Custom statistics

- WSDG - Address Counter
- WSDG - Address Counter (Sorted)
- QA Cafe (CloudShark) –

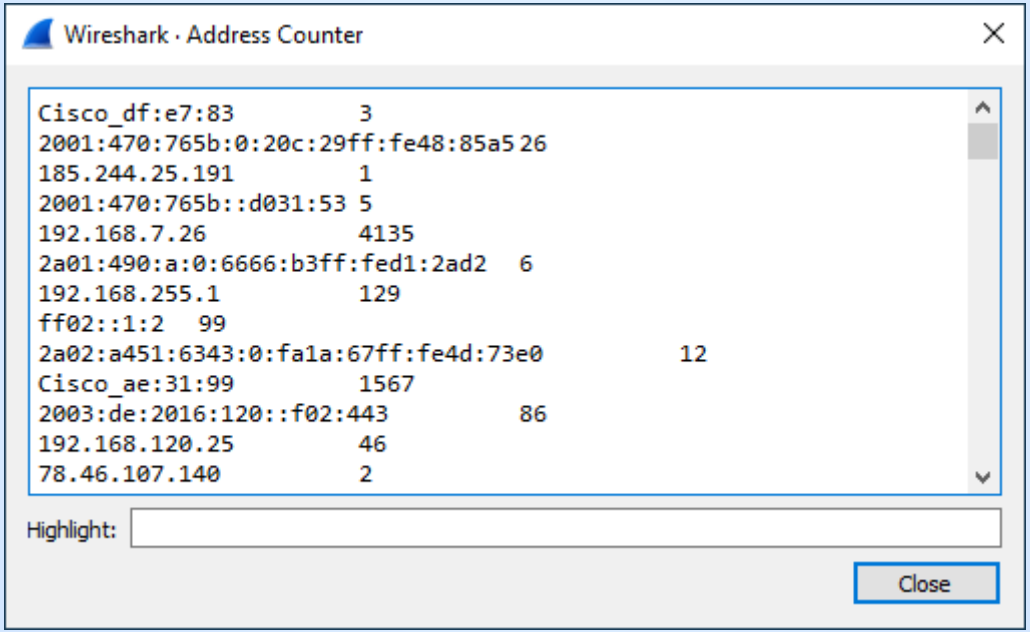
  "How to write a Wireshark tap plugin in Lua"

DRAFT

address_counter.lua

- WSDG - https://www.wireshark.org/docs/wsdg_html/#wslua_tap_example
- "This program will register a menu that will open a window with a count of occurrences of every address in the capture"

address_counter.lua



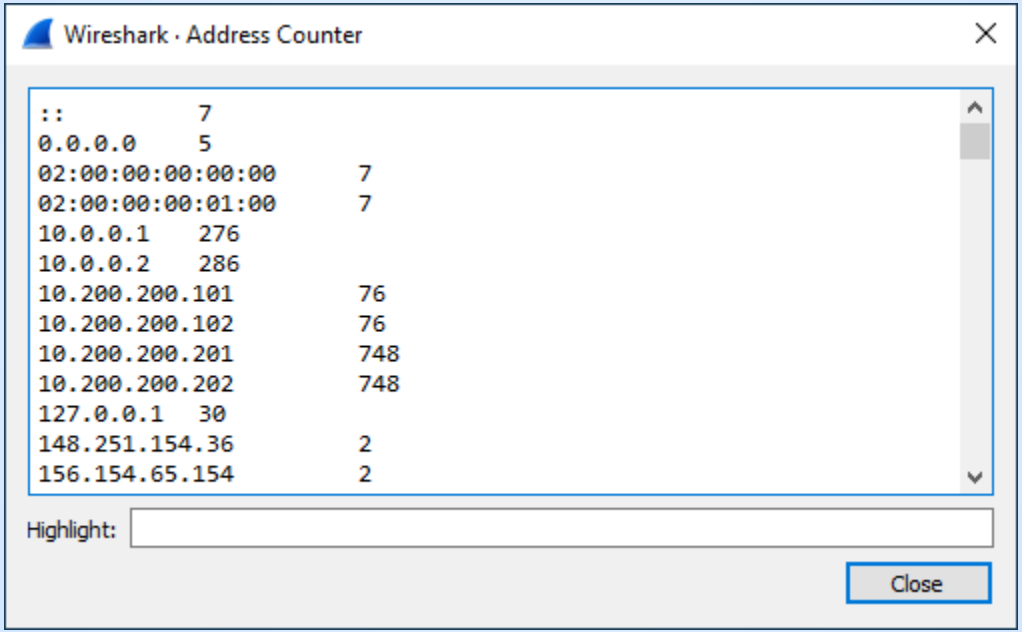
DRAFT

The Ultimate PCAP v20221220.pcapng (https://weberblog.net/the-ultimate-pcap/)
Screenshot of export trimmed to first 20000 packets

address_counter_sorted.lua

- WSDG - https://www.wireshark.org/docs/wsdg_html/#wslua_tap_example
- "This program will register a menu that will open a window with a count of occurrences of every address in the capture"

address_counter_sorted.lua



DRAFT

The Ultimate PCAP v20221220.pcapng (https://weberblog.net/the-ultimate-pcap/)
Screenshot of export trimmed to first 20000 packets

address_counter_sorted.lua

Print sorted output:

```
function tap.draw(t)
    tw:clear()
    table.sort(ips)
    for ip,num in pairsByKeys(ips) do
        tw:append(ip .. "\t" .. num .. "\n");
    end
end
```

Programming in Lua: (https://www.lua.org/pil/19.3.html)
19.3 – Sort

```
function pairsByKeys (t, f)
```

DRAFT

wifi-networks.lua

- https://github.com/cloudshark/WiFi-Networks-Plugin
- https://www.qacafe.com/resources/how-to-write-a-wireshark-tap-plugin-in-lua/
- Gui'fy ???

DRAFT

wifi-networks.lua

Command Prompt

```
C:\ tshark -q  -r wifi-networks.pcapng
BSSID   SSID    Security        Vendor  Hidden  Signal  Noise   SNR     Channel
62:ab:eb:6c:fb:c0       "CloudShark"    WPA2-Personal   62:ab:eb:6c:fb:c0       false   -34     -74     40      11
4e:9d:08:53:d1:12       "ACME Corp"     WPA2-Personal   4e:9d:08:53:d1:12       false   -49     -74     25      11
de:dd:d7:51:ba:95       "The Neighbors" WPA2/3-Personal de:dd:d7:51:ba:95       false   -41     -73     32      11
de:4e:cb:fe:62:d4       "ISEEYOURPACKETS"       Open    de:4e:cb:fe:62:d4       false   -39     -76     37      11
3e:71:ce:4f:32:68       "Ye Olde Coffee Shop"   WPA3-Personal   3e:71:ce:4f:32:68       false   -41     -74     33      11

C:\
```

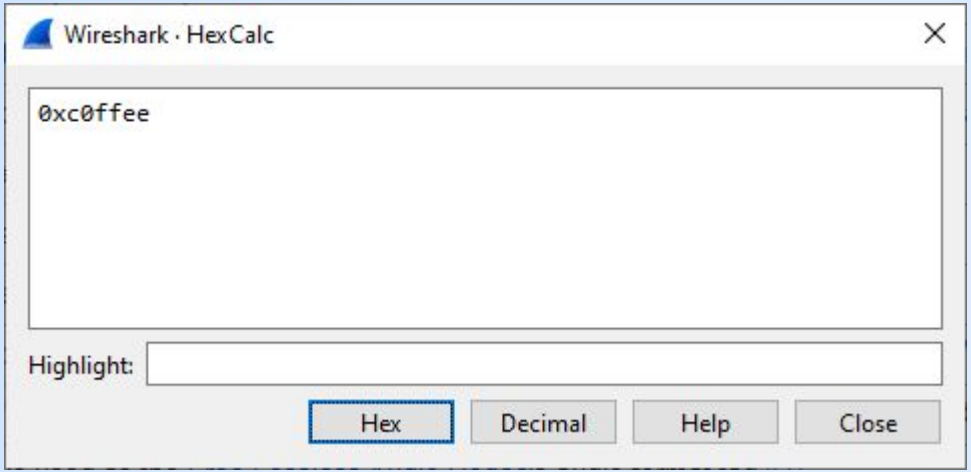
DRAFT

wifi-networks.pcapng - https://www.cloudshark.org/captures/6d72d13108b3

Add menu items/utilities

- hexcalc.lua

hexcalc.lua



Download: https://wiki.wireshark.org/Contrib -> Other

DRAFT

https://gitlab.com/wireshark/wireshark/-/issues/18386
funnel/lua: closing child window disconnects buttons of parent

hexcalc.lua

Document - `set_plugin_info(hexcalc_info)`

In GUI ? - `if not gui_enabled() then return end`

Main function - open window, read text, output value.

```
local function hexcalc()
    local win = TextWindow.new("HexCalc")
    win:set_editable(true)
```

Add buttons –

```
        win:add_button("Help", function()
```

Create the menu entry -

```
register_menu("Hex Calculator",hexcalc,MENU_TOOLS_UNSORTED)
```

DRAFT