Convert Access Point Packet Dumps for Wireshark

Contents

Introduction

Prerequisites

Procedure

- Perform Packet Dump
- Output File Cleanup
- Cleanup Packet Summary Information
- Remove starting spaces and offset colons
- Correct packet offset
- Separate Packet Bytes
- Convert the Text File to PCAP
- Via Wireshark GUI
- Via command line

Troubleshooting

- Text File is Correct but Text2pcap Cannot Read Any Packets
- Inconsistent Offset

Introduction

This document describes how to convert a COS Access Point generated packet dump to PCAP format for Wireshark as a workaround to the size limitation.

Prerequisites

- Notepad++ - Available only on Windows
- Text2pcap installed - included on regular installations of Wireshark

Procedure

Perform Packet Dump

Capture an AP packet dump by running the command **debug traffic wired <multiple options> verbose** on the AP command line. You can choose between multiple filters and interfaces.

Log the session in the terminal.
Be careful to send the least amount of keystrokes when doing so, the more printable characters on the file that do not belong to the capture itself the more cleanup you need to do before conversion.
The easiest way to do it is a console session for the packet dump, replicate the issue, stop the dump and immediately end the session.

If you are performing the dump via ssh use a filter to only capture the traffic of interest. Otherwise the capture contains the ssh session packets.

Refer to Troubleshoot COS APs for complete instructions on how to configure the capture.

When you are done, stop the capture with the command **undebug all**. The resulting file looks like this:

```
AP-9105>en
Password:
AP-9105#debug traffic wired udp
  capture  capture packets in pcap file
  verbose  Verbose Output
  <cr>
AP-9105#debug traffic wired udp verbose
AP-9105#reading from file /dev/click_wired_log, link-type EN10MB (Ethernet)
22:35:17.1669188 IP CSCO-W-PF320YP6.lan.60354 > 239.255.255.250.3702: UDP, length 656
        0x0000:  0100 5e7f fffa 806d 971d a040 0800 4500
        0x0010:  02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
        0x0020:  fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
        0x0030:  7665 7273 696f 6e3d 2231 2e30 2220 656e
        0x0040:  636f 6469 6e67 3d22 7574 662d 3822 3f3e
<truncated>
undebug 0x0070:  444c 4e41 444f 432f 312e 3530 2050 6c61
        0x0080:  7469 6e75 6d2f 312e 302e 342e 320d 0a4d
        0x0090:  414e 3a20 2273 7364 703a 6469 7363 6f76
        0x00a0:  6572 220d 0a53 543a 2073 7364 703a 616c
all     0x00b0:  6c0d 0a4d 583a 2033 0d0a 0d0a
<truncated>
tcpdump: pcap_loop: error reading dump file: Interrupted system call
All possible debugging has been turned off
<end of file>
```

Output File Cleanup

Remove any information which is not part of the packet dump itself. Delete the lines containing the dump command, any prompt which contains the hostname (APname#) and any other unrelated syslog messages present in the file.

Pay special attention to the undebug command since it can be printed before a packet content as shown above. Afer the cleanup, the resulting file looks like this:

```
22:35:17.1669188 IP CSCO-W-PF320YP6.lan.60354 > 239.255.255.250.3702: UDP, length 656
        0x0000:  0100 5e7f fffa 806d 971d a040 0800 4500
        0x0010:  02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
        0x0020:  fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
        0x0030:  7665 7273 696f 6e3d 2231 2e30 2220 656e
        0x0040:  636f 6469 6e67 3d22 7574 662d 3822 3f3e
<truncated>
        0x0070:  444c 4e41 444f 432f 312e 3530 2050 6c61
        0x0080:  7469 6e75 6d2f 312e 302e 342e 320d 0a4d
        0x0090:  414e 3a20 2273 7364 703a 6469 7363 6f76
        0x00a0:  6572 220d 0a53 543a 2073 7364 703a 616c
```

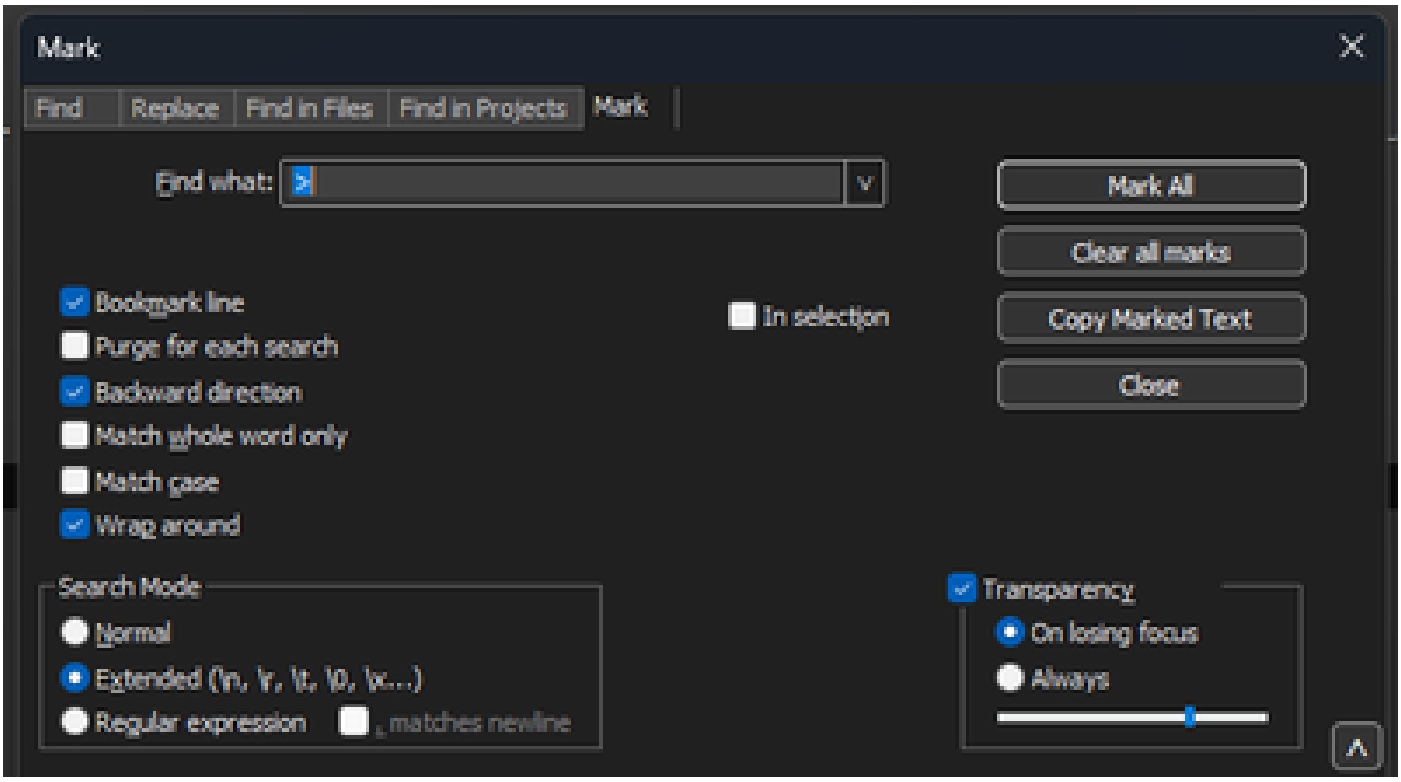
```
0x00b0:  6c0d 0a4d 583a 2033 0d0a 0d0a
```

Cleanup Packet Summary Information

The start of a new packet is detected when a new offset 000000 appears. Text2pcap can handle the summary information printed before each packet, to avoid issues is best to remove them.

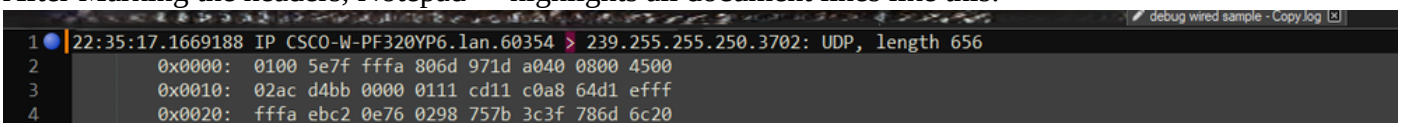
In Notepad++ navigate to **Search>Find** And select the **Mark** tab, ensure the **Search Mode** is **Extended**.

On the **Find what:** field enter the symbol > and click **Mark All.** This action bookmarks all lines containing the > symbol.



Notepad++ mark dialog box with Find what field with the chevron character inside.

After Marking the headers, Notepad++ highlights all document lines like this:



Packet dump snippet wiht highlighted line which contains the chevron.

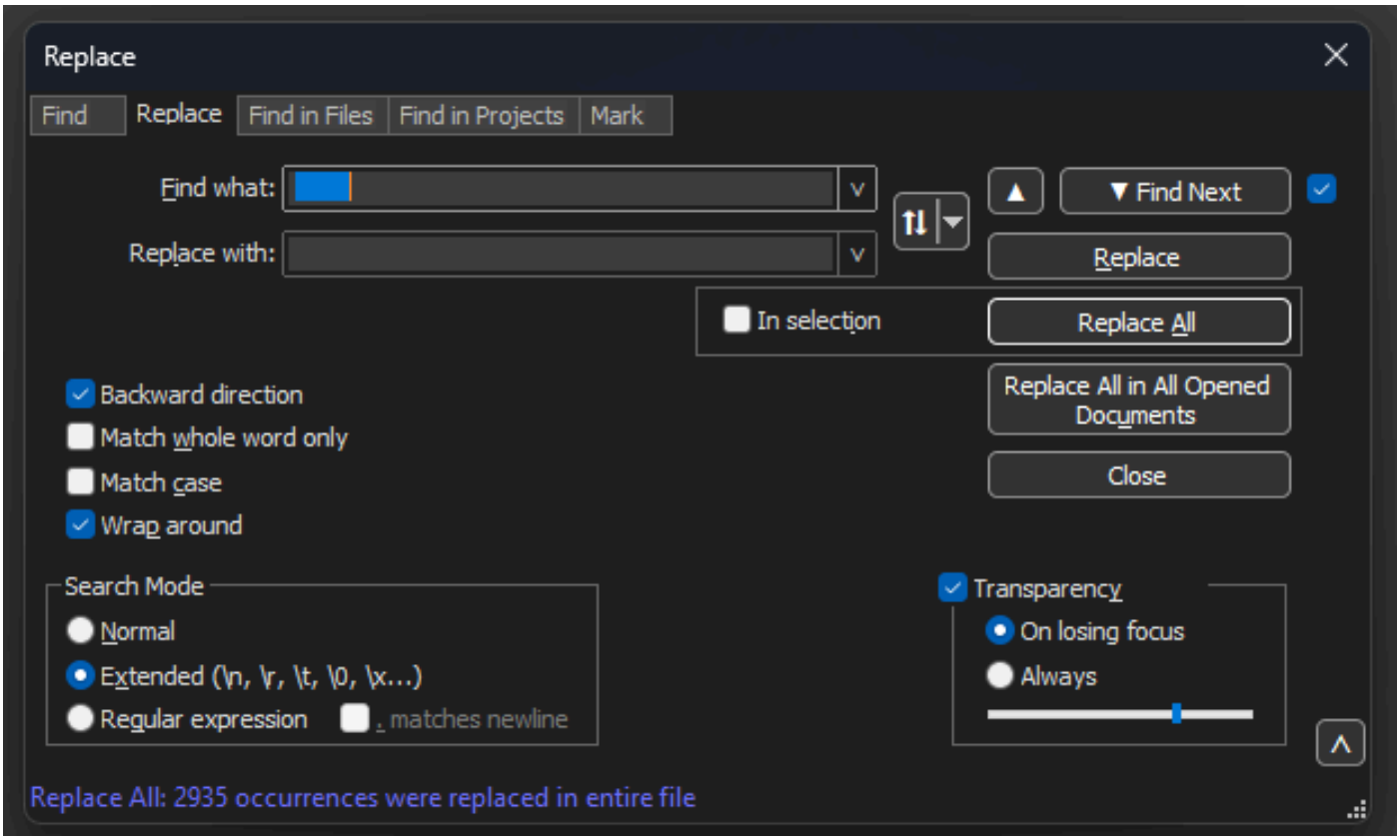
Navigate to **Search>Bookmark** and click on **Remove bookmarked lines**. After doing so, the file looks like this snippet:

```
0x0000:  0100 5e7f fffa 806d 971d a040 0800 4500
0x0010:  02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
0x0020:  fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
0x0030:  7665 7273 696f 6e3d 2231 2e30 2220 656e
```

Remove starting spaces and offset colons

Navigate to **Search>Find** And select the **Replace** Tab, ensure the Search Mode is **Extended**.

On the **Find what:** field enter **8 white spaces**. Leave the **Replace with:** field empty and click on **Replace all**. This replaces all 8 consecutive white spaces at the start of every line with nothing, effectively deleting them. The replace dialog looks like this image.

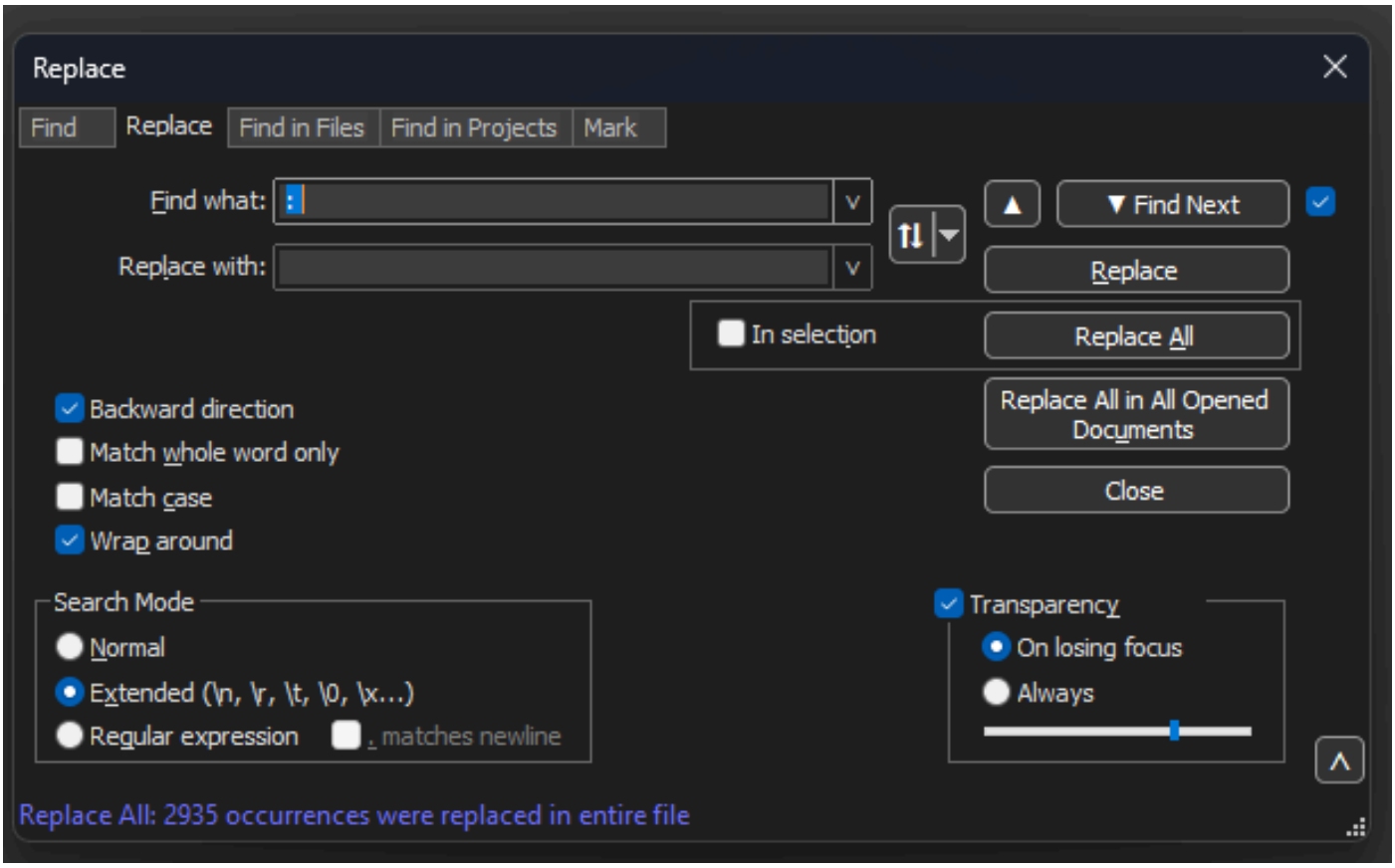


Notepad++ Replace dialog box with Find what field with 8 spaces.

The resulting file after this operation looks like this snippet:

```
0x0000:  0100 5e7f fffa 806d 971d a040 0800 4500
0x0010:  02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
0x0020:  fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
0x0030:  7665 7273 696f 6e3d 2231 2e30 2220 656e
0x0040:  636f 6469 6e67 3d22 7574 662d 3822 3f3e
0x0050:  3c73 6f61 703a 456e 7665 6c6f 7065 2078
0x0060:  6d6c 6e73 3a73 6f61 703d 2268 7474 703a
0x0070:  2f2f 7777 772e 7733 2e6f 7267 2f32 3030
```

Navigate to **Search>Find** And select the **Replace** Tab, ensure the **Search Mode** is **Extended.** Enter **:** (notice the blank space after the colon) on the **Find what:** field. Leave the **Replace with:** field empty and click on **Replace all.**
This replaces all colons and first spaces after the offset.



Notepad++ Replace dialog box with Find what field filled by a colon and a space.

After the previous operation, the resulting output file looks like this snippet:

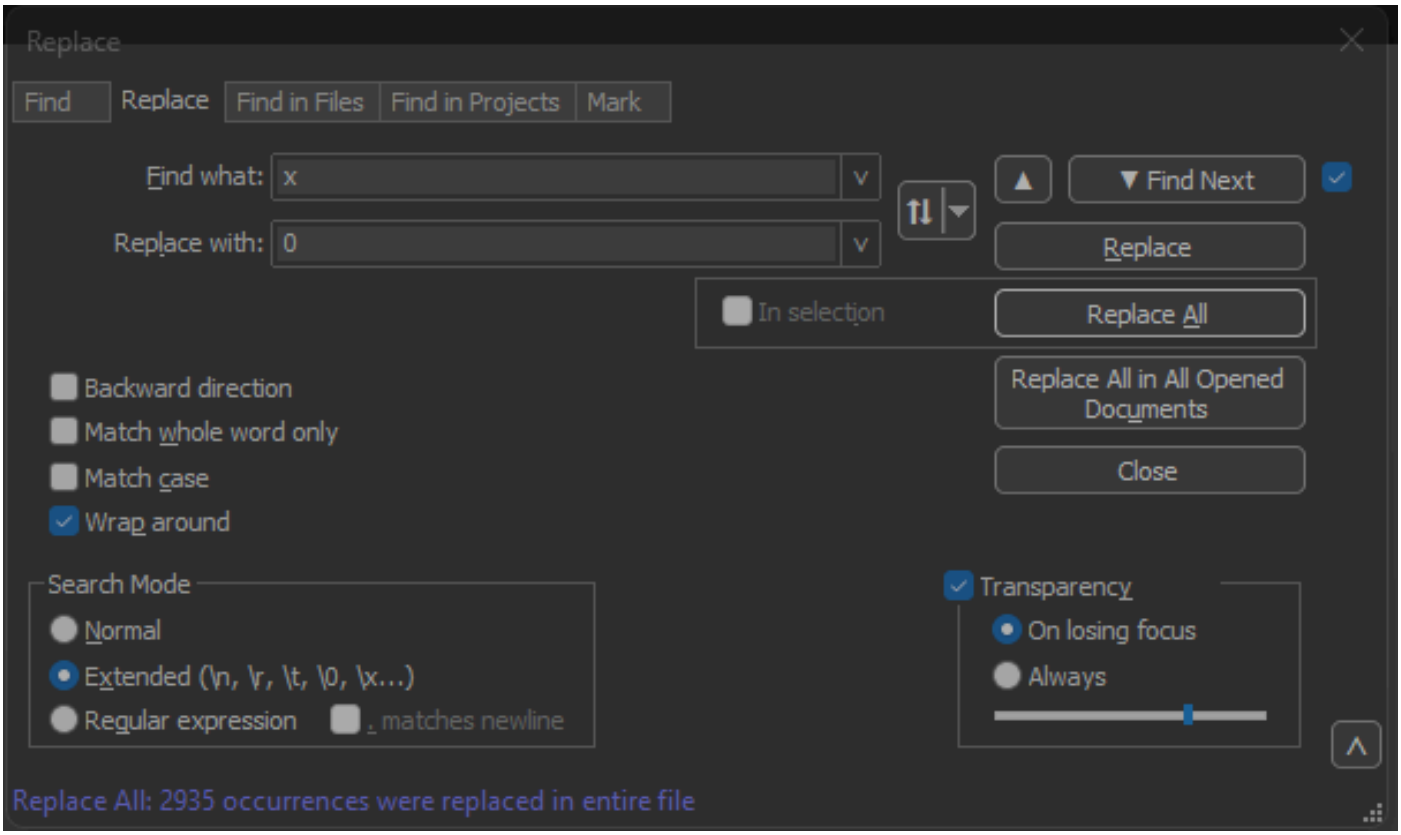
```
0x0000 0100 5e7f fffa 806d 971d a040 0800 4500
0x0010 02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
0x0020 fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
0x0030 7665 7273 696f 6e3d 2231 2e30 2220 656e
0x0040 636f 6469 6e67 3d22 7574 662d 3822 3f3e
0x0050 3c73 6f61 703a 456e 7665 6c6f 7065 2078
0x0060 6d6c 6e73 3a73 6f61 703d 2268 7474 703a
0x0070 2f2f 7777 772e 7733 2e6f 7267 2f32 3030
```

Correct packet offset

Text2pcap expects packet offset inside each packet as a 6 character hex string, but AP packet dumps use 0x to symbolize the offset instead. To correct it navigate to **Search>Find** And select the **Replace** Tab, ensure

the Search Mode is **Extended**.

Enter **x** on the **Find what:** field. Fill the **Replace with:** field with **0** and click on **Replace all.** This replaces all x inside the offset with 0 to match the expected offset format for Text2pcap.



Notepad++ Replace dialog box with Find what field filled with the character x and Replace field filled with the character 0.

After the previous operation, the resulting output file looks like this snippet:

```
000000 0100 5e7f fffa 806d 971d a040 0800 4500
000010 02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
000020 fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
000030 7665 7273 696f 6e3d 2231 2e30 2220 656e
000040 636f 6469 6e67 3d22 7574 662d 3822 3f3e
000050 3c73 6f61 703a 456e 7665 6c6f 7065 2078
```

Separate Packet Bytes

Text2pcap data format requires for each pair of hex values to be separated by a space, an incorrect format causes Text2pcap to read packet data as an offset and fail.

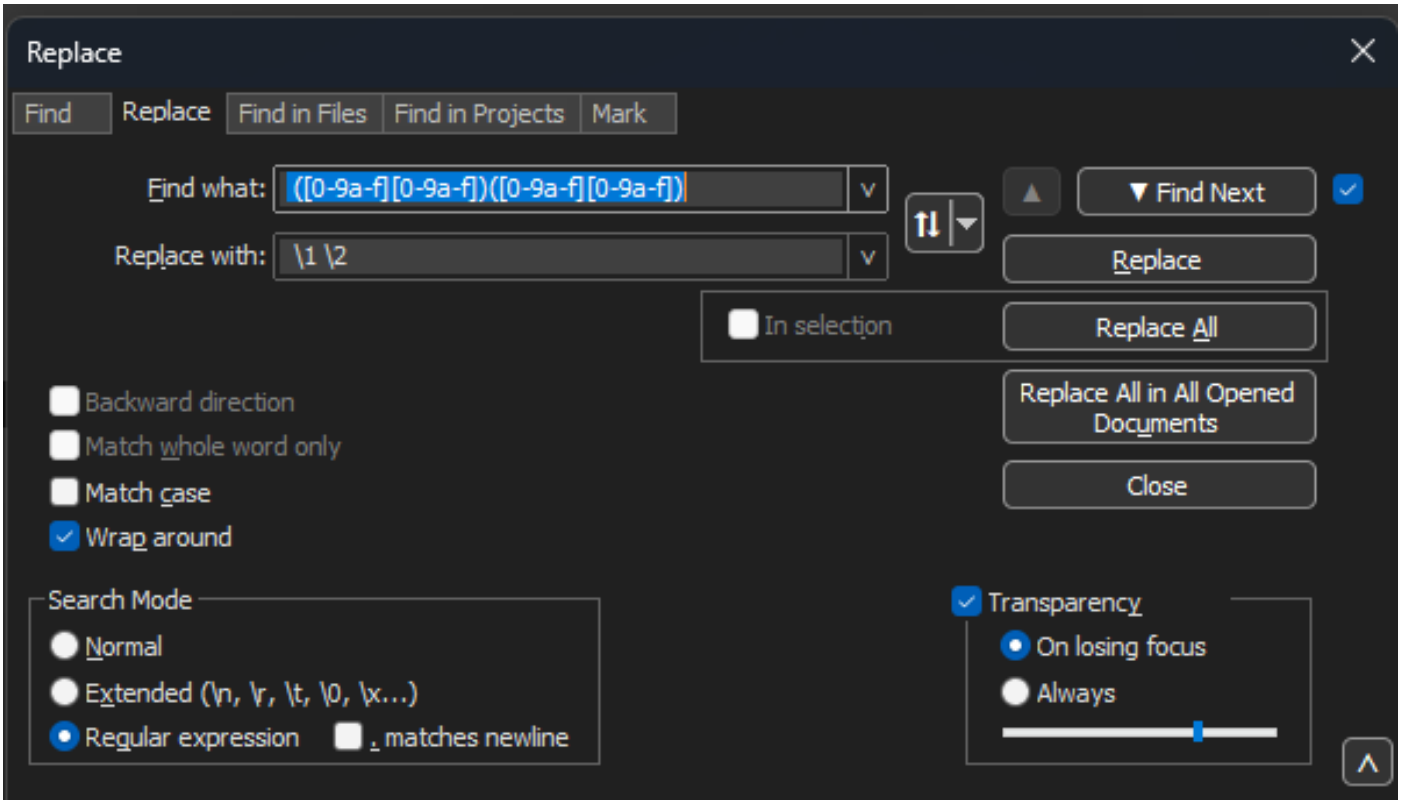
Navigate to **Search>Find** and select the **Replace** Tab, ensure the Search Mode is **Regular expression**.

Enter **([0-9a-f][0-9a-f])([0-9a-f][0-9a-f])** (notice the leading space) on the **Find what:** field.

Fill the **Replace with:** field with **\1 \2** (notice the leading space) and click on **Replace all**.

The replace operation finds the hex bytes of the packet and inserts a space between each pair. The regex matches a space followed by a hex digit pair, saves them on capture group 1, then takes the adjacent pair of hex digits, saves them on capture group 2. The replacement prints both required spaces as well as the content of each capture group.

It takes multiple seconds or minutes depending on the length of the file. It utilizes a lot of RAM while running If the file is large, be patient.



Notepad++ Replace dialog box with the find what filled with a regullar expression and the Replace field filled by another regular expression.

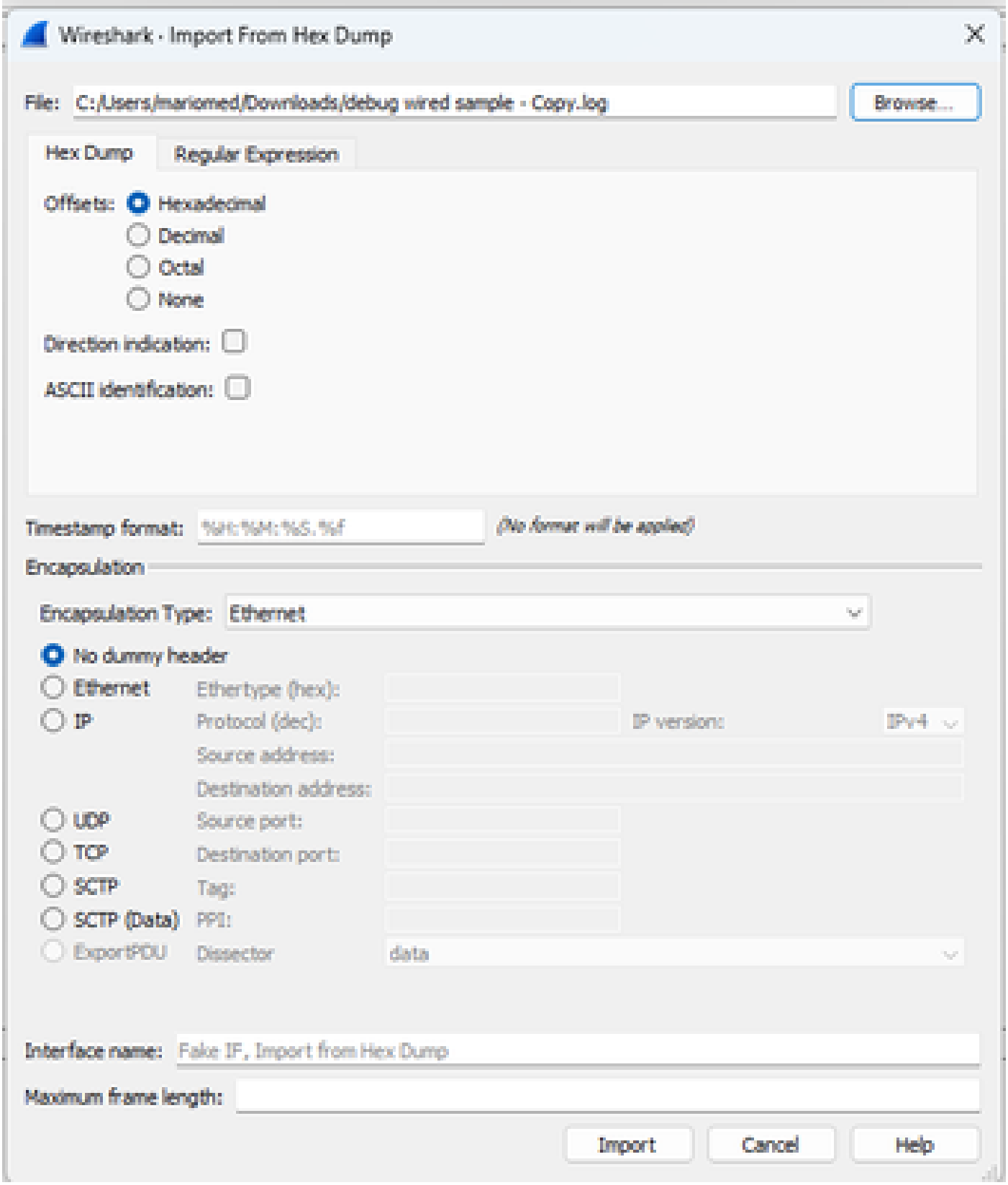
After the previous operation, the resulting output file looks like this snippet and is ready to be converted by Text2pcap.

```
000000 01 00 5e 7f ff fa 80 6d 97 1d a0 40 08 00 45 00
000010 02 ac d4 bb 00 00 01 11 cd 11 c0 a8 64 d1 ef ff
000020 ff fa eb c2 0e 76 02 98 75 7b 3c 3f 78 6d 6c 20
000030 76 65 72 73 69 6f 6e 3d 22 31 2e 30 22 20 65 6e
000040 63 6f 64 69 6e 67 3d 22 75 74 66 2d 38 22 3f 3e
000050 3c 73 6f 61 70 3a 45 6e 76 65 6c 6f 70 65 20 78
000060 6d 6c 6e 73 3a 73 6f 61 70 3d 22 68 74 74 70 3a
000070 2f 2f 77 77 77 2e 77 33 2e 6f 72 67 2f 32 30 30
000080 33 2f 30 35 2f 73 6f 61 70 2d 65 6e 76 65 6c 6f
000090 70 65 22 20 78 6d 6c 6e 73 3a 77 73 61 3d 22 68
```

Convert the Text File to PCAP

Via Wireshark GUI

To convert the complete file to pcap, open Wireshark and navigate to **File>Import from hex dump**, a dialog box appears.



Wireshark import dialog box

Click on the **Browse...** button and select the dump text file. Ensure the selected offset type is **Hexadecimal**, **Encapsulation type** is **Ethernet** and **No dummy header** is selected.

Click **Import** to start the conversion process.

Via command line

To convert a text file to a pcap file in the windows command line, run **<path to wireshark install folder>\text2pcap.exe <path to text file pcap> <output file path>.**

You can optionally add wireshark folder to your PATH otherwise you need to run text2pcap referencing the entire path to the text2pcap.exe every time you convert a file. Text2pcap.exe is located inside the wireshark install folder.

```
PS C:\Users\mariomed\Downloads> text2pcap "debug wired sample - Copy.log" final.pcap
Input from: debug wired sample - Copy.log
Output to: final.pcap
Output format: pcapng

--------------------------
Read 147 potential packets, wrote 147 packets (50904 bytes including overhead).
```

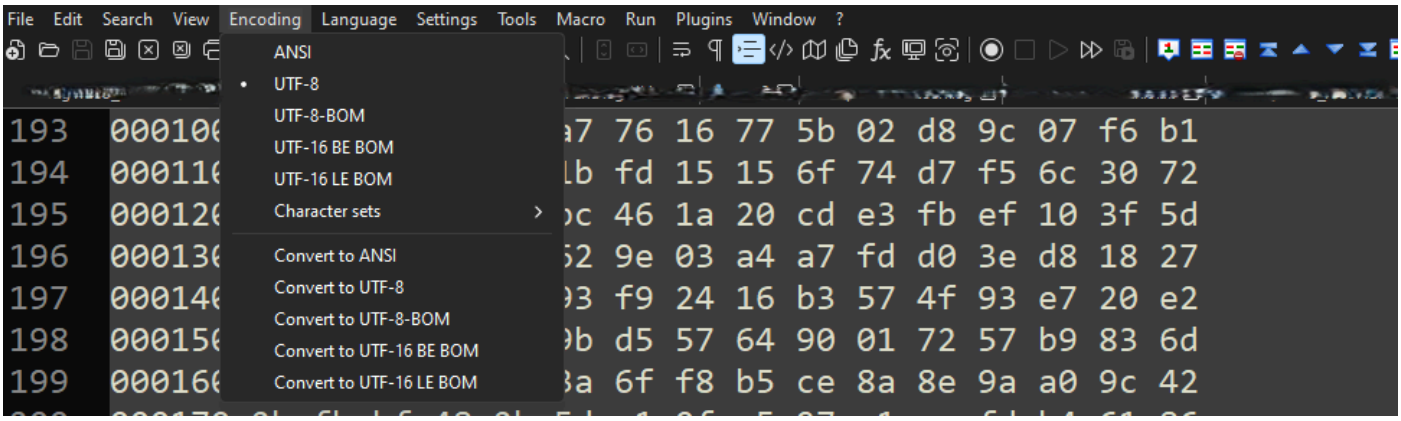
Windows command line output after the succesful packet dump conversion

Text2pcap also includes multiple regex options to pre-process the text file, please refer to the Text2pcap manual page for more information.

Troubleshooting

Text File is Correct but Text2pcap Cannot Read Any Packets

Text2pcap cannot read certain file encodings produced by terminal emulators commonly used (Secure CRT, Putty or others).
Change to an encoding readable by Text2pcap with Notepad++. Go to **Encoding>UTF-8** and save the file, then convert to pcap again.



Notepad++ encoding menu options.

Inconsistent Offset

This error appears when the bytes of the data portion on a packet are not correctly separated into pairs, this causes Text2pcap to assume the start of a new packet and fails to interpret.
Search for any packet bytes without separation or strings in the middle of a packet content such as the `undebug all` command.

```
C:\Users\mariomed>text2pcap "C:\Users\mariomed\Downloads\debug wired sample - Copy.log" output.pcap
Input from: C:\Users\mariomed\Downloads\debug wired sample - Copy.log
Output to: output.pcap
Output format: pcapng
 ** (text2pcap:81244) 10:30:46.781149 [(none) MESSAGE] -- Inconsistent offset. Expecting 75, got 80. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.781712 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.782136 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.782446 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.782599 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.782748 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.782891 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.783033 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.783169 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.783319 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
 ** (text2pcap:81244) 10:30:46.783456 [(none) MESSAGE] -- Inconsistent offset. Expecting 10, got 10. Ignoring rest of packet
```

Windows command line output after invalid file is attempted to convert. Inconsistent offset is printed to the terminal multiple times.