Wireshark用のアクセスポイントパケットダンプの変換

内容

はじめに

このドキュメントでは、サイズ制限の回避策として、COSアクセスポイント(AP)で生成されたパケットダンプをWireshark用のPCAP形式に変換する方法について説明します。

前提条件

- メモ帳++ - Windowsでのみ使用可能
- Text2pcapがインストール済み：Wiresharkの通常のインストールに含まれる

手順

パケットダンプの実行

APコマンドラインでdebug traffic wired <multiple options> verbose コマンドを実行して、APパケットダンプをキャプチャします。複数のフィルタとインターフェイスから選択できます。

端末にセッションを記録します。
このとき、最小限のキーストロークを送信するように注意してください。キャプチャ自体に属さないファイル上の印刷可能な文字が多いほど、変換前にクリーンアップが必要になります。

これを行う最も簡単な方法は、パケットダンプ用のコンソールセッションを使用して、問題を複製し、ダンプを停止して、すぐにセッションを終了することです。

ssh経由でダンプを実行している場合は、フィルタを使用して対象のトラフィックのみをキャプチャします。それ以外の場合、キャプチャにはsshセッションパケットが含まれます。

キャプチャの設定方法の詳細は、『COS APのトラブルシューティング』を参照してください。

作業が終了したら、undebug allコマンドを使用してキャプチャを停止します。結果のファイルは次のようになります。

```
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-PF32OYP6.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>
```

出力ファイルのクリーンアップ

パケットダンプ自体に含まれていない情報があれば削除します。dumpコマンドを含む行、ホスト名(APname#)を含むプロンプト、およびファイルに存在する他の無関係なsyslogメッセージを削除します。

undebugコマンドは、上記のようにパケットの内容の前に出力される可能性があるため、特に注意してください。クリーンアップ後、結果のファイルは次のようになります。

```
22:35:17.1669188 IP CSCO-W-PF32OYP6.lan.60354 > 239.255.255.250.3702: UDP, length 656
```

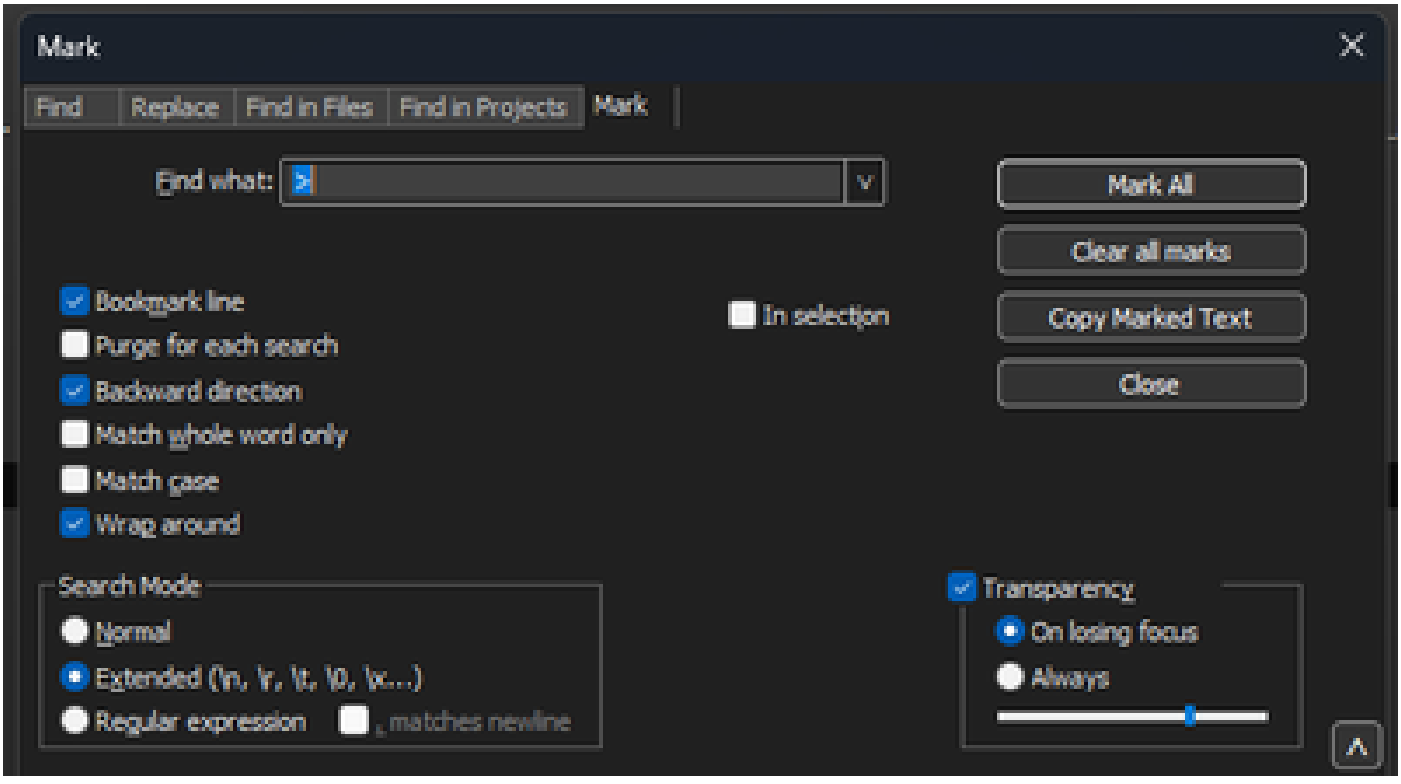
```
        0x0000:  0100 5e7f fffa 806d 971d a040 0800 4500
        0x0010:  02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
        0x0020:  fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
        0x0030:  7665 7273 696f 6e3d 2231 2e30 2220 656e
        0x0040:  636f 6469 6e67 3d22 7574 662d 3822 3f3e
<truncated>
        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
```

クリーンアップパケットの概要情報

新しいオフセット000000が表示されると、新しいパケットの開始が検出されます。Text2pcapは、各パケットの前に出力される要約情報を処理できるため、問題を回避するには、それらを削除するのが最善です。

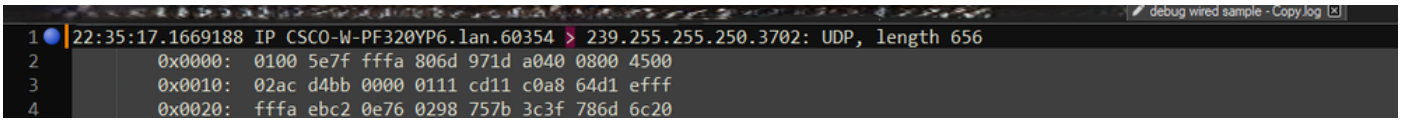
メモ帳で++Search>Findの順に移動し、Markタブを選択して、Search ModeがExtendedになっていることを確認します。

Find what:フィールドに記号>を入力し、Mark Allをクリックします。このアクションは、>記号を含むすべての行にブックマークを付けます。



Notepad++ markダイアログボックスのFind whatフィールドに山形の文字が表示されます。

ヘッダーにマークを付けた後、メモ帳++は次のようにすべてのドキュメント行を強調表示します。



シェブロンを含む強調表示された行を持つパケットダンプスニペット。

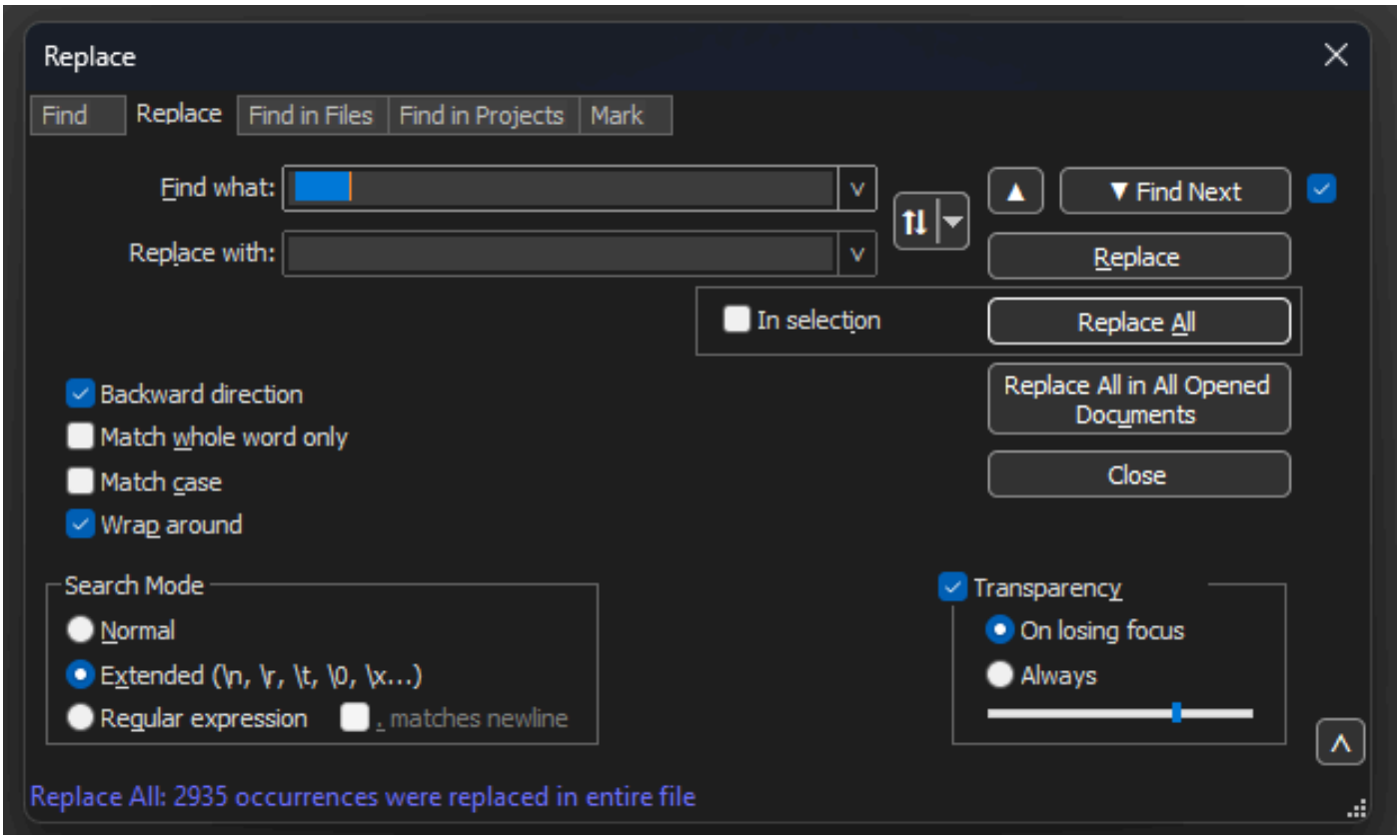
Search>Bookmarkに移動し、Remove bookmarked linesをクリックします。その後、ファイルは次のスニペットのようになります。

```
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
```

開始スペースとオフセットコロンを削除する

Search>Findに移動し、Replaceタブを選択して、検索モードがExtendedになっていることを確認します。

Find what:フィールドに、8つの空白を入力します。Replace with:フィールドを空のままにして、Replace allをクリックします。これにより、すべての行の先頭にある8つの連続する空白が何も置き換えられ、事実上それらを削除します。置き換えダイアログは次の図のようになります。

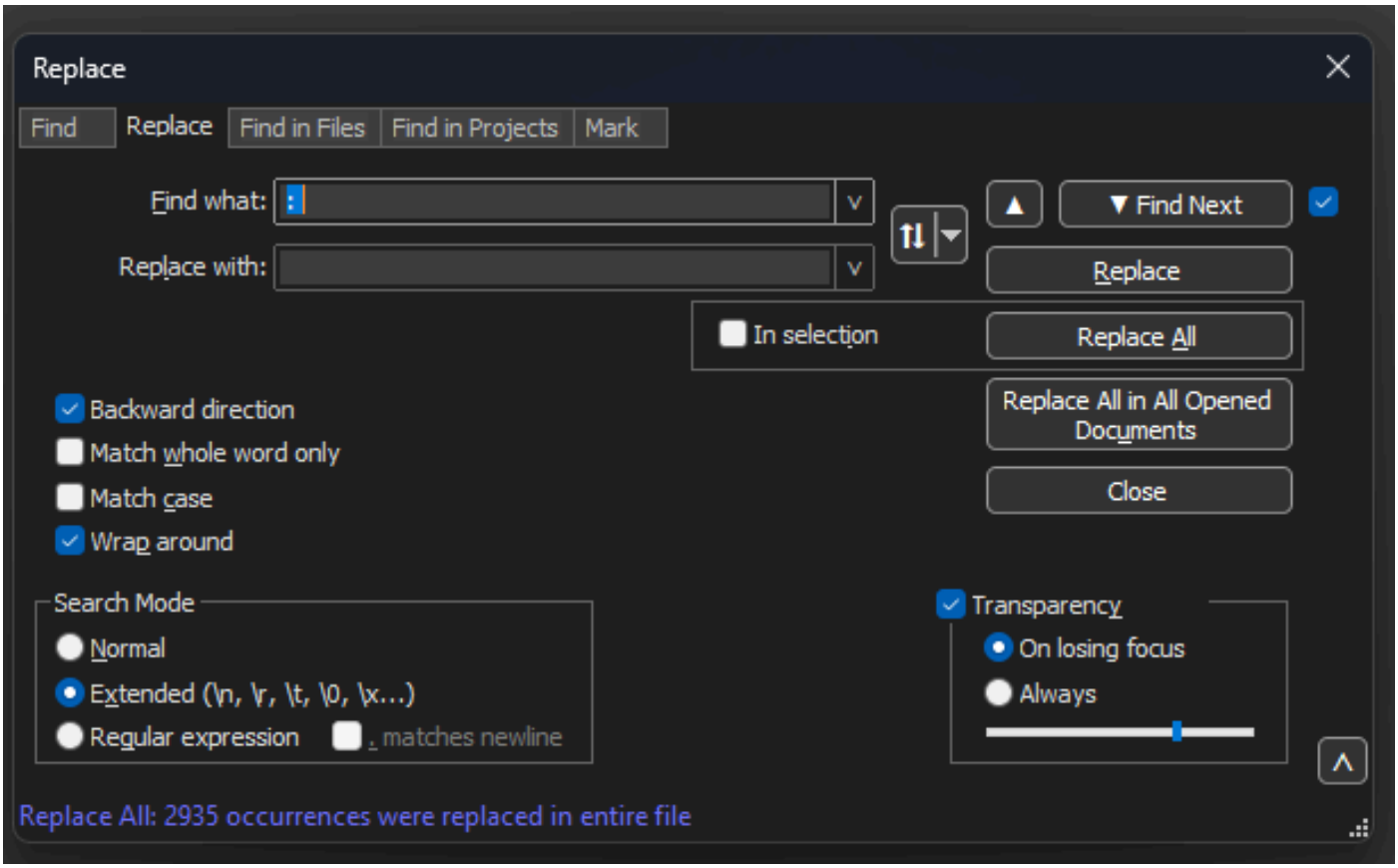


メモ帳++検索するフィールドを8つのスペースで置き換えます。

この操作後の結果ファイルは、次のスニペットのようになります。

```
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
```

Search>Findに移動し、Replaceタブを選択して、Search ModeがExtendedになっていることを確認します。Find what:フィールドに: (コロンの後の空白に注意してください) を入力します。
Replace with:フィールドを空のままにして、Replace allをクリックします。
これにより、オフセットの後のすべてのコロンと最初のスペースが置き換えられます。



メモ帳++ [検索]ダイアログボックスの[検索]フィールドにコロンとスペースを入力します。

前の操作の後、結果の出力ファイルは次のようになります。

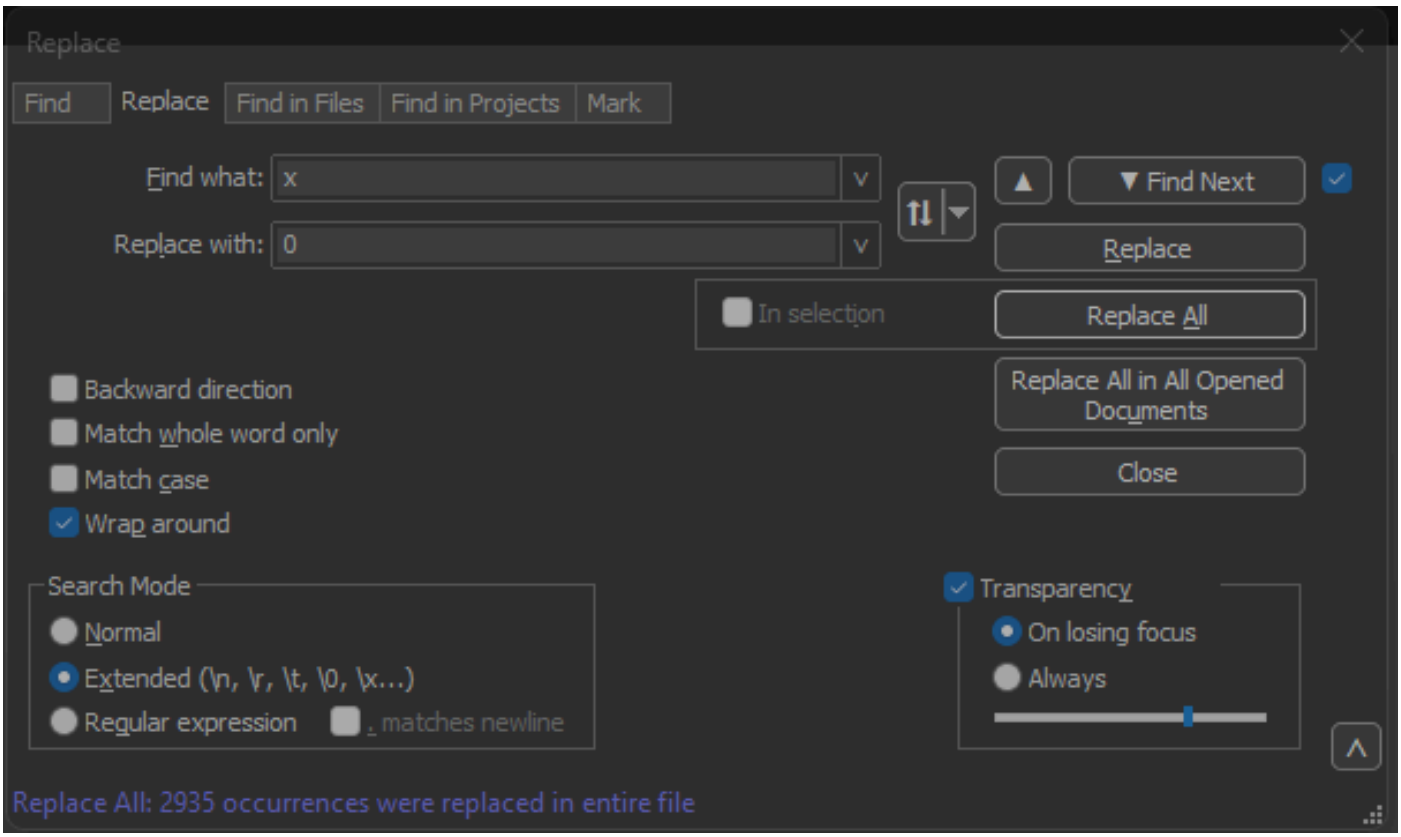
```
0x0000 0100 5e7f fffa 806d 971d a040 0800 4500
0x0010 02ac d4bb 0000 0111 cd11 c0a8 64d1 efff
0x0020 fffa ebc2 0e76 0298 757b 3c3f 786d 6c20
0x0030 7665 7273 696f 6e3d 2231 2e30 2220 656e
0x0040 636f 6469 6e67 3d22 7574 662d 3822 3f3e
0x0050 3c73 6f61 703a 456e 7665 6c6f 7065 2078
0x0060 6d6c 6e73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

正しいパケットオフセット

Text2pcapでは、各パケット内のパケットオフセットを6文字の16進数文字列として想定していますが、APパケットダンプではオフセットを示すために0xを使用しています。これを修正するには、Search> Findの順に移動し、Replaceタブを選択して、Search ModeがExtendedになっていることを確認します。

Find what:フィールドにxと入力します。Replace with:フィールドに0を入力し、Replace allをク

リックします。これにより、オフセット内のすべてのxが、Text2pcapの想定されるオフセット形式に一致するように0に置き換えられます。



メモ帳++ [置換]ダイアログボックスの[検索する文字列]フィールドに文字xを入力し、[置換]フィールドに文字0を入力します。

前の操作の後、結果の出力ファイルは次のようになります。

```
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
```

分離パケットバイト

Text2pcapデータ形式では、16進数値の各ペアをスペースで区切る必要があります。形式が正しくない場合、Text2pcapはパケットデータをオフセットとして読み取り、失敗します。

Search>Findに移動してReplaceタブを選択し、Search ModeがRegular expressionになっている

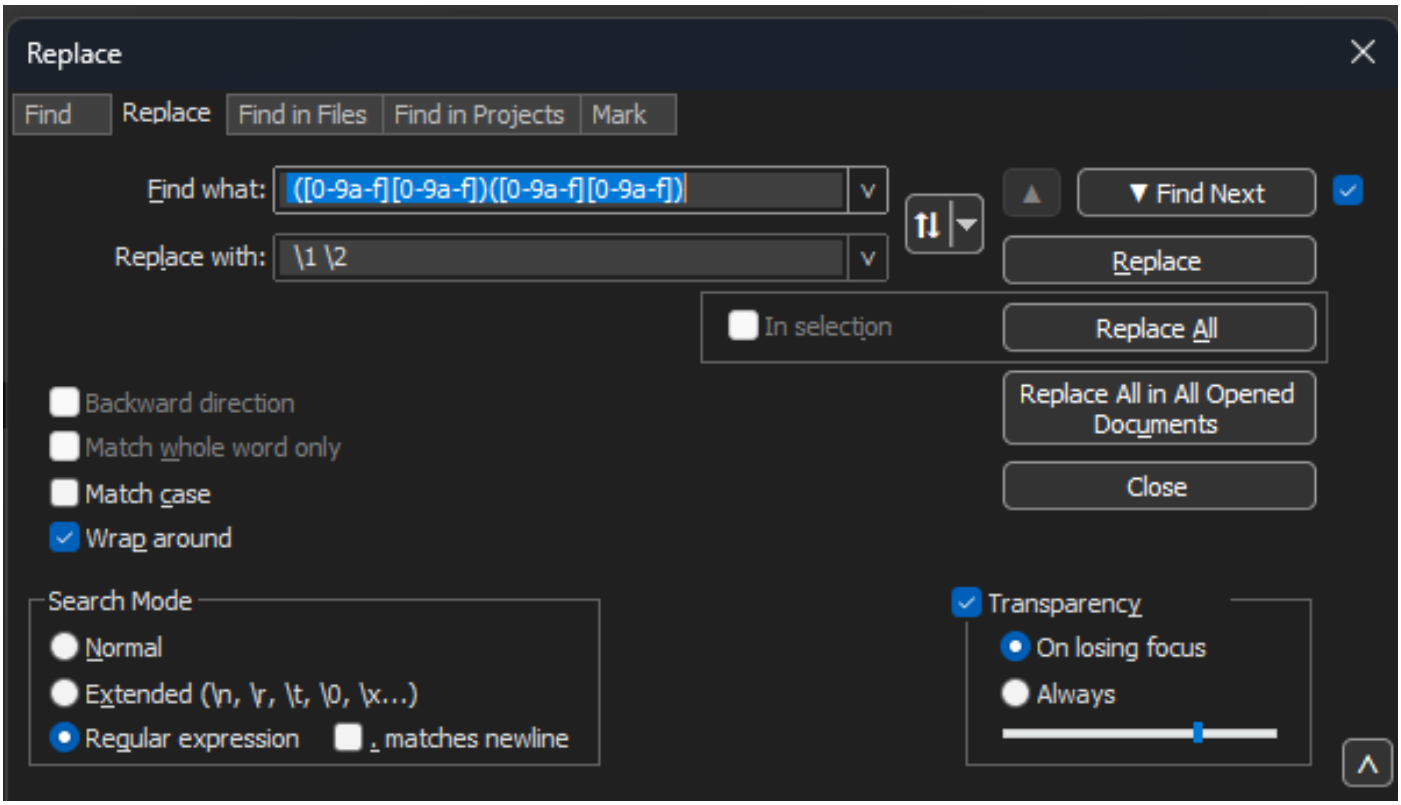
ことを確認します。

Find what:フィールドに、([0-9a-f][0-9a-f])([0-9a-f][0-9a-f])（先頭のスペースに注意）と入力します。

Replace with:フィールドに \1 \2（先頭のスペースに注意）を入力し、Replace allをクリックします。

置換操作では、パケットの16進数バイトが検索され、各ペアの間にスペースが挿入されます。正規表現は、スペースの後に16進数のペアが続く部分と一致する部分をキャプチャグループ1に保存し、その後に隣接する16進数のペアを取ってキャプチャグループ2に保存します。置き換えると、必要なスペースと各キャプチャグループの内容の両方が印刷されます。

ファイルの長さによっては、数秒または数分かかります。実行中に大量のRAMを使用しますファイルが大きい場合は、我慢してください。



メモ帳++ [置換]ダイアログボックスが開き、正規表現で入力された内容が検索され、 [置換]フィールドに別の正規表現が入力されます。

前の操作の後、結果の出力ファイルは次のスニペットのように表示され、Text2pcapで変換する準備が整います。

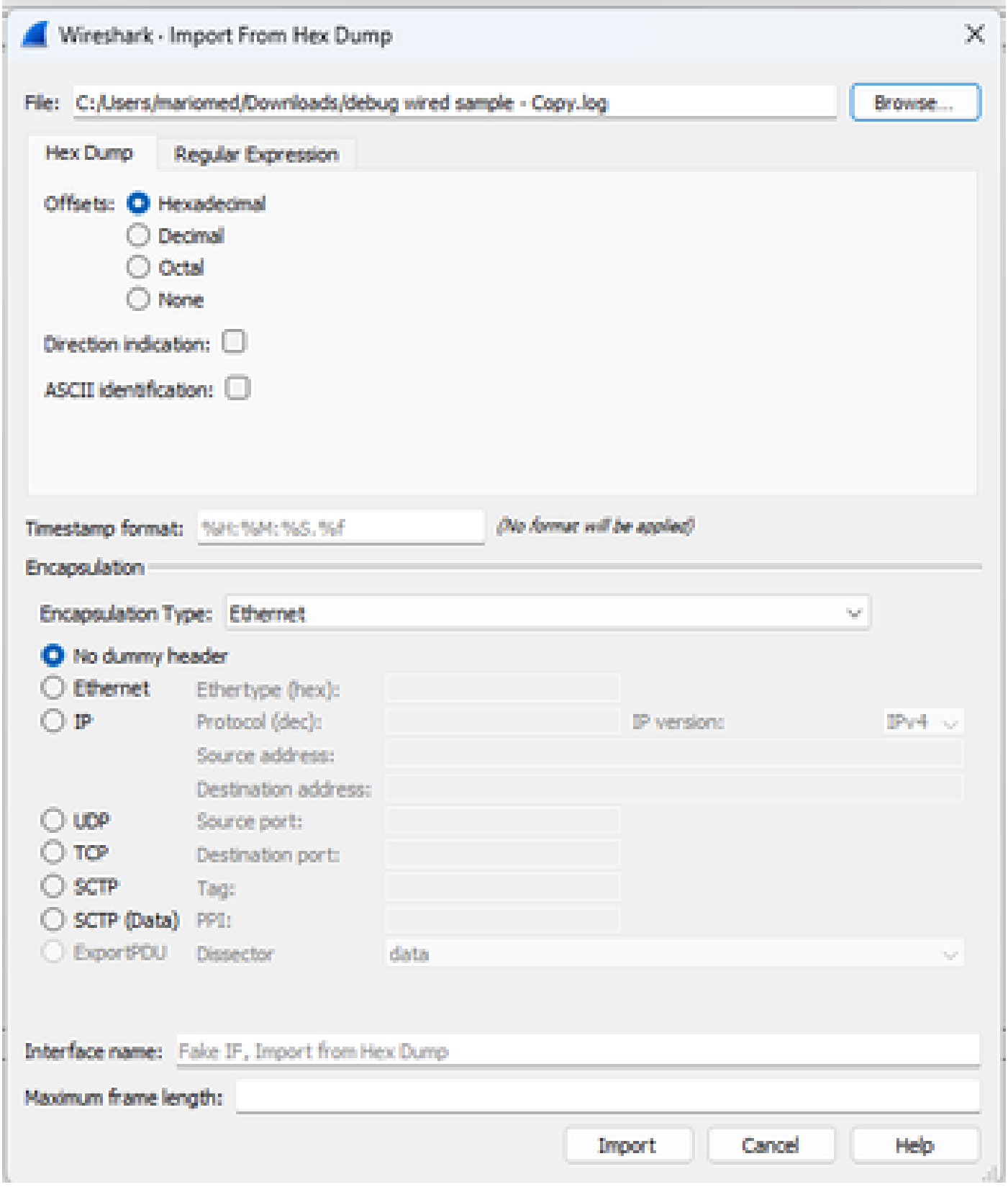
```
000000 01 00 5e 7f ff fa 80 6d 97 1d a0 40 08 00 45 00
```

```
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
```

テキストファイルをPCAPに変換する

Wireshark GUIを使用

ファイル全体をpcapに変換するには、Wiresharkを開いてFile > Import from hex dumpの順に選択すると、ダイアログボックスが表示されます。



Wiresharkインポートダイアログボックス

Browse...ボタンをクリックして、ダンプテキストファイルを選択します。選択したオフセットタイプが16進数、カプセル化タイプがイーサネット、ダミーヘッダーが選択されていないことを確

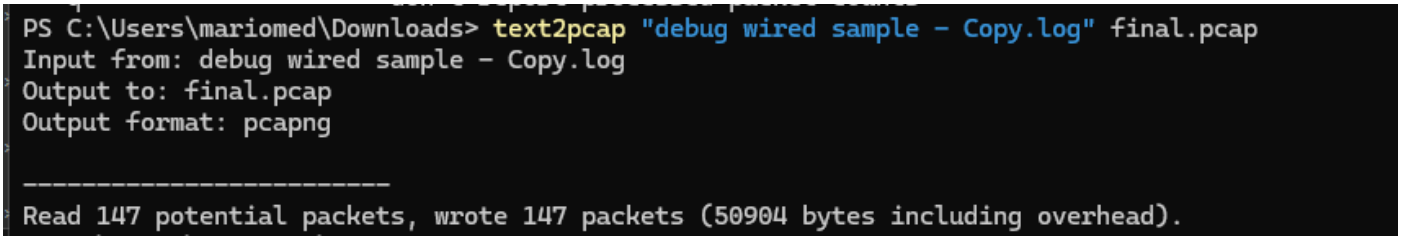
認します。

Importをクリックして、変換プロセスを開始します。

コマンドラインを使用

テキストファイルをWindowsコマンドラインでpcapファイルに変換するには、<path to wireshark install folder>\text2pcap.exe <path to text file pcap> <output file path>を実行します。

オプションで、PATHにwiresharkフォルダを追加できます。追加しない場合は、ファイルを変換するたびに、text2pcap.exeへのパス全体を参照するtext2pcapを実行する必要があります。Text2pcap.exeは、wiresharkのインストールフォルダ内にあります。



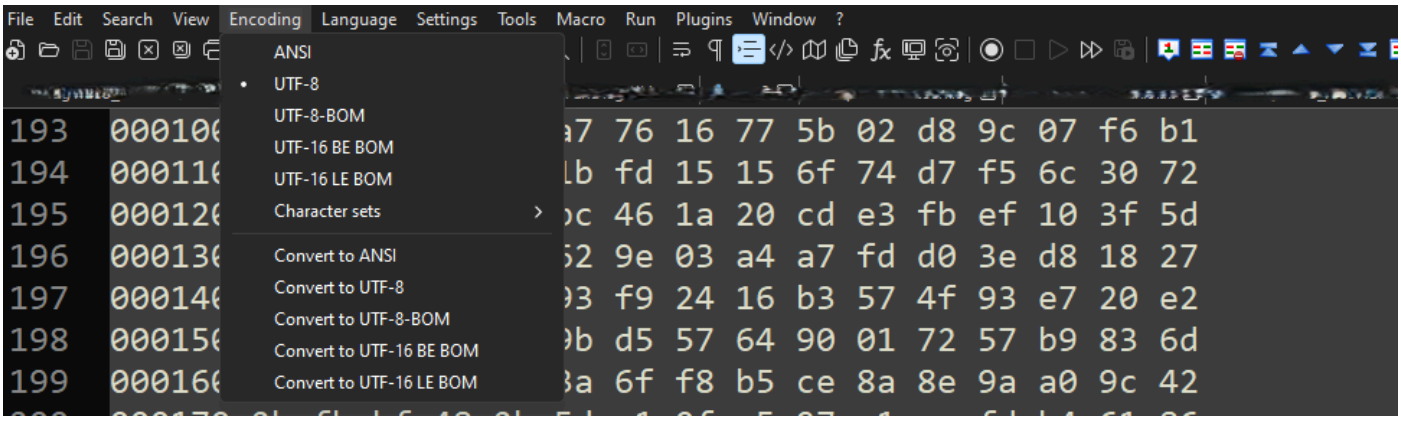
パケットダンプ変換が成功した後のWindowsコマンドライン出力

text2pcapには、テキストファイルを前処理するための複数のregexオプションも含まれています。詳細は、『Text2pcapのマニュアルページ』を参照してください。

トラブルシューティング

テキストファイルは正しいにも関わらず、Text2pcapがパケットを読み取れない

Text2pcapは、よく使われるターミナルエミュレータ（Secure CRT, Puttyなど）で生成された特定のファイルエンコーディングを読み込めません。
Text2pcapで読み取り可能なエンコーディングにNotepad++で変更します。Encoding>UTF-8に移動してファイルを保存し、再度pcapに変換します。



メモ帳++のエンコーディングメニューオプション

矛盾したオフセット

このエラーは、パケットのデータ部分のバイトが正しくペアに分割されていない場合に表示されます。この場合、Text2pcapは新しいパケットの先頭を引き継ぎ、解釈に失敗します。

undebug all コマンドなど、パケットコンテンツの途中で区切りや文字列が含まれていないパケットバイトがないかどうかを検索します。

```
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
```

無効なファイルを変換しようとした後の*Windows*コマンドライン出力。矛盾したオフセットが端子に複数回出力されます。

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。