

You are here: [Home](#) » [Articles](#) » [Networking/Computing Tips/Tricks](#) » Finding Text Strings in Wireshark

Networking/Computing Tips/Tricks

Finding Text Strings in Wireshark Captures

🕒 Created: Thursday, 26 September 2019 10:19

📄 Hits: 28189



Rate this content:



5 of 5 - 2 votes

A common question regarding Wireshark packet analysis is "Can I find a text string in a capture?"

The answer is that it depends on where the text string is (like header vs. packet content) and if the packets contain encrypted data.

Usecase #1:

If you are looking for something like "password" in the contents of packets, and the user was on an HTTPS connection, then you will not find this string. However, if they are using HTTP or some other clear text protocol, then you will be able to find a string in the packet contents.

Usecase #2:

If you are looking for a string in the packet headers, it will depend on whether the header was inside or outside a VPN tunnel. Headers outside such a tunnel are always searchable and not encrypted. Anything in the tunnel will be encrypted and therefore not searchable.

Alright, let's talk about what tools come in Wireshark to find a string.

If you would like to see this - check out our video:

Check out these great

[Our custom profiles](#)

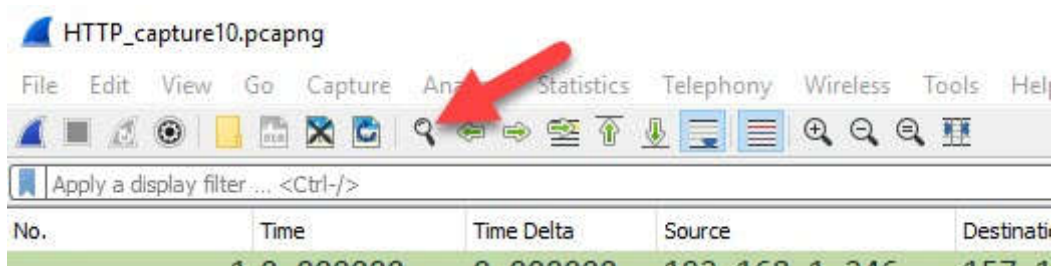
[Our Udemmy course](#)

[Our Udemmy course](#)

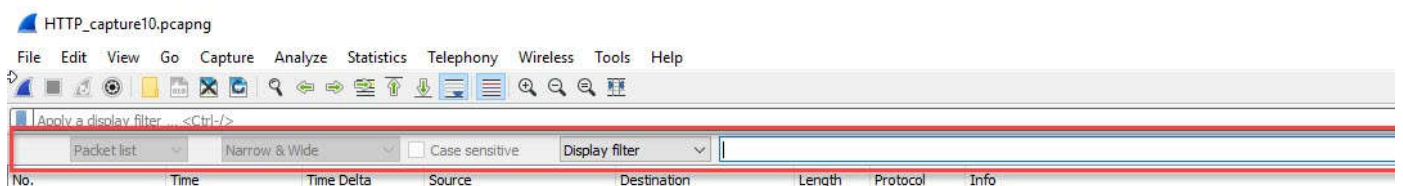


Option 1

First there is the generic find/search capability in Wireshark that is found here:

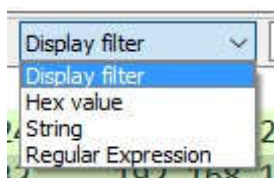


When you click on this looking glass button, or select Edit> Find Packet from the drop down menus, you will be present toolbar immediately below the display filter toolbar:



You will note the "Display filter" drop down just to the left of the string entry box.

The options are as follows:



To find a string, select string, and note that the two other drop down boxes are no longer greyed out.

Now select packet bytes if you want to look inside the packets, and then type the string you are looking for in the entry

HTTP_capture10.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl-/>

Packet bytes Narrow & Wide Case sensitive String BHI

No.	Time	Time Delta	Source	Destination	Length	Protocol	Info
236	1.084716	0.000056	192.168.1.246	157.140.2.32	54	TCP	64806 → 80 [ACK] Seq=2197 Ack=217069
237	1.084733	0.000017	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=217069 Ack=2197
238	1.084884	0.000151	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=218449 Ack=2197
239	1.084885	0.000001	157.140.2.32	192.168.1.246	126	HTTP	HTTP/1.1 200 OK (PNG)
240	1.084923	0.000038	192.168.1.246	157.140.2.32	54	TCP	64806 → 80 [ACK] Seq=2197 Ack=219901
241	1.107681	0.022758	192.168.1.246	157.140.2.32	860	HTTP	GET /sites/aba.myspecies.info/files/s
242	1.230243	0.122562	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=219901 Ack=3003
243	1.230564	0.000321	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=221281 Ack=3003
244	1.230633	0.000069	192.168.1.246	157.140.2.32	54	TCP	64806 → 80 [ACK] Seq=3003 Ack=222661
245	1.230676	0.000043	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=222661 Ack=3003
246	1.230919	0.000243	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=224041 Ack=3003
247	1.230919	0.000000	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=225421 Ack=3003

Acknowledgment number: 3003 (relative ack number)
 0101 = Header Length: 20 bytes (5)
 > Flags: 0x010 (ACK)
 Window size value: 279
 [Calculated window size: 35712]
 [Window size scaling factor: 128]
 Checksum: 0x0b6a [unverified]
 [Checksum Status: Unverified]
 Urgent pointer: 0
 > [SEQ/ACK analysis]
 > [Timestamps]
 TCP payload (1380 bytes)
Reassembled PDU in frame: 271
 TCP segment data (1380 bytes)

0330	e1 b7 ef ea bd fb 9e a7	ed e6 b4 14 42 48 49 10BHI
0340	7d cf 3b 57 1d da fd e3	b8 48 53 41 09 84 c8 a6	};W...HSA...
0350	40 11 02 68 3d 8c 52 b1	58 c8 04 54 3c 61 e8 f1	@.h=R.X.Tca
0360	d0 40 3e 97 4c 70 46 47	86 7b cf 3d bb 7a c3 b5	@>LpFG-{.=z..
0370	4b be f1 e9 b3 ee 79 cc	bb f3 9e b5 46 2a 26 5f	K...y...F*&
0380	c4 9f 5b 29 85 75 6d e6	94 e4 fa bd b5 50 68 15	..[]um...Ph
0390	1f 67 91 2e aa a5 76 db	28 8c 17 3f f5 6f 77 13	.g...v(-.?ow

A data segment used in reassembly of a lower-level protocol (tcp.segment_data), 1380 bytes

Packets: 285 · Displayed: 285 (100.0%)

Above, you can see I selected string, packet bytes, entered "BHI" as my string and then clicked find. Packet 246 has this highlighted. This was the first instance, and if I clicked find again, Wireshark will look further into the capture. Eventually, the capture ends and I have to reset the view to the first packet to initiate the search once again.

This search is case insensitive. Whether I had entered "bhi" or "bHi" or "bHI", the search will find the same packet.

Option 2

What if I just wanted to see the packets with "BHI" in them?

For this we need to use the Display Filter functionality of Wireshark. [A reference with details regarding my examples below](#)

Specifically, there is a display filter term called 'frame contains' and 'frame matches'. 'Contains' is fairly straightforward.

```
frame contains "BHI"
```

HTTP_capture10.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

frame contains "bhi"

No.	Time	Time Delta	Source	Destination	Length	Protocol	Info
246	1.230919	0.000000	157.140.2.32	192.168.1.246	1434	TCP	80 → 64806 [ACK] Seq=224041 Ack=3003

Acknowledgment number: 3003 (relative ack number)
 0101 = Header Length: 20 bytes (5)
 > Flags: 0x010 (ACK)
 Window size value: 279
 [Calculated window size: 35712]
 [Window size scaling factor: 128]
 Checksum: 0x0b6a [unverified]
 [Checksum Status: Unverified]
 Urgent pointer: 0
 > [SEQ/ACK analysis]
 > [Timestamps]
 TCP payload (1380 bytes)
Reassembled PDU in frame: 271
 TCP segment data (1380 bytes)

0030	01 17 0b 6a 00 00	a9 19 ab a7 6f e7 aa 73 c7 e6	...j... ..o..s..
0040	74 cf 3c 32 34 48 09 95	4a 69 1a 9d 54 84 26 97	t-<24H... ji..T.&.
0050	5f 25 15 e3 62 52 63 25	04 03 00 b8 98 14 65 95	%.bRc%e.
0060	10 52 02 85 30 3e 69 cc	12 52 6a 94 4c 5a b5 26	.R..0>i. .Rj.LZ.&
0070	4d 98 93 26 74 08 80 90	2f 4f 16 fd bf e4 fa 85	M..&t... /0.....
0080	11 fa c3 33 45 08 a1 ef	f9 67 9e 31 f7 6d 37 75	...3E... .g-1-m7u
0090	7e f1 13 6f 5a 7d ff af	1a 3d f5 89 3d c3 c5 e1	~..oZ}... ..=...=...

A data segment used in reassembly of a lower-level protocol (tcp.segment_data), 1380 bytes

Packets: 285 · Displayed: 1 (0.4%)

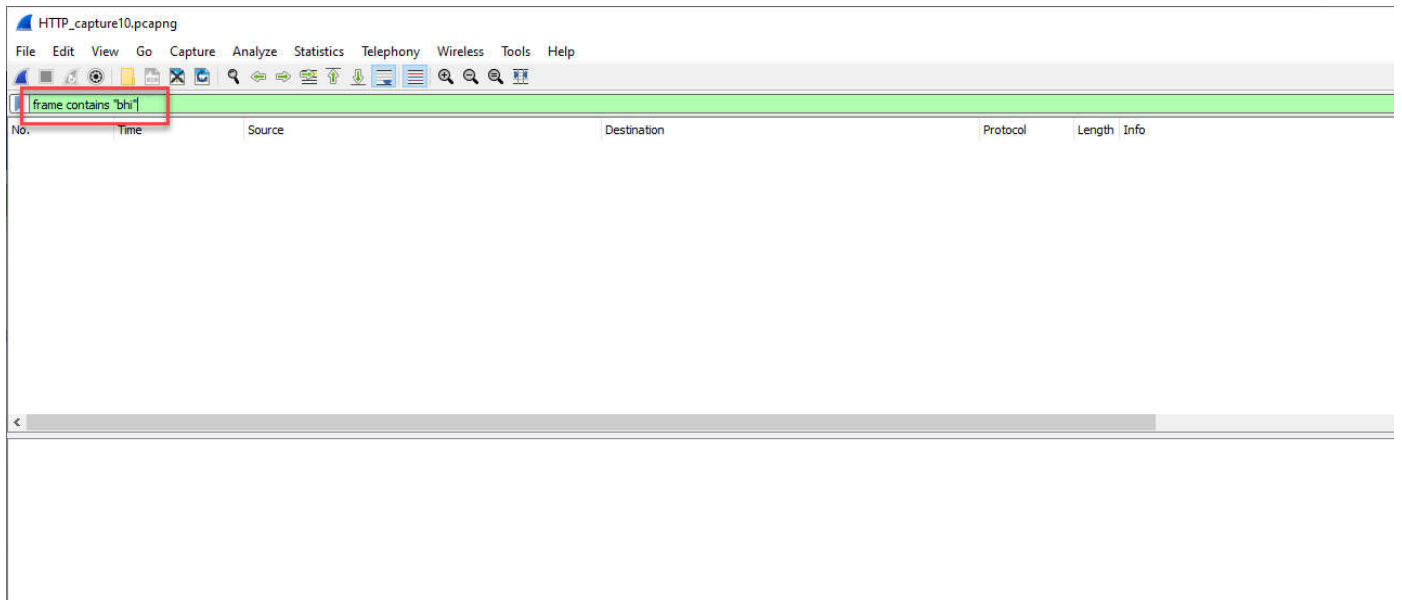
A couple of things here: you do not need to use quotes, and you cannot say something like an IP Address.

The 'frame matches' is a little different. This is a fairly flexible display filter and we will not cover all the options here. T "Regex next" to Wireshark - a Perl-compatible regular expression. [You can find a great cheat sheet for Regex here.](#) So details of this, we will provide and explain some examples commonly used. Hopefully from this you will understand it.

These 'frame contains' or 'frame matches' display filters are case sensitive. If I used:

```
frame contains "bhi"
```

I get no results:



So a common command I use when performing these types of searches is the (?i) which makes the search case insensi

So I could have used the following command:

```
frame matches "(?i)bhi"
```

I now find the packet again:

HTTP_capture10.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

frame matches "(?i)bhi"

No.	Time	Time Delta	Source	Destination	Length	Protocol	Info
246	1.230919	0.000000	157.140.2.32	192.168.1.246	1434	TCP	80 -> 64806 [ACK] Seq=224041 Ack=3003

Acknowledgment number: 3003 (relative ack number)
 0101 = Header Length: 20 bytes (5)
 > Flags: 0x010 (ACK)
 Window size value: 279
 [Calculated window size: 35712]
 [Window size scaling factor: 128]
 Checksum: 0x0b6a [unverified]
 [Checksum Status: Unverified]
 Urgent pointer: 0
 > [SEQ/ACK analysis]
 > [Timestamps]
 TCP payload (1380 bytes)
 Reassembled PDU in frame: 271
 TCP segment data (1380 bytes)

0030	01 17 0b 6a 00 00	a9 19 ab a7 6f e7 aa 73 c7 e6	...j... ..o..s..
0040	74 cf 3c 32 34 48 09 95	4a 69 1a 9d 54 84 26 97	t-<24H.. ..i..T.&..
0050	5f 25 15 e3 62 52 63 25	04 03 00 b8 98 14 65 95	%.bRc%e..
0060	10 52 02 85 30 3e 69 cc	12 52 6a 94 4c 5a b5 26	.R..0>i.. .Rj.LZ.&..
0070	4d 98 93 26 74 08 80 90	2f 4f 16 fd bf e4 fa 85	M..&t... /O.....
0080	11 fa c3 33 45 08 a1 ef	f9 67 9e 31 f7 6d 37 75	...3E... .g.1.m7u..
0090	7e f1 13 6f 5a 7d ff af	1a 3d f5 89 3d c3 c5 e1	...oZ}... ..=...=...

A data segment used in reassembly of a lower-level protocol (tcp.segment_data), 1380 bytes

Packets: 285 · Displayed: 1 (0.4%)

Let's say I wanted to find "BHI" or "BHS", I would use:

```
frame matches "(?i)bh[is]"
```

Let's say I wanted to find "BHI" and all others through "BHS", I would use:

```
frame matches "(?i)bh[i-s]"
```

Let's say I wanted to match "BHI" or "BHS", I would use:

```
frame matches "(?i)(bhi|bhs)"
```

As you can imagine, another common one here would be:

```
frame matches "(?i)(username|password)"
```

You can also use the '^', which means start of field, and '\$', which means end of field, commands.

Also the '\' means look for a dot.

So if you wanted to find any listed web sites in a capture, perhaps you would use:

```
frame matches "\.com$"
```

Or perhaps you would want to find possible files:

```
frame matches "\.(?i)(exe|zip|doc|xls|ppt|jar)"
```

Lets say you wanted to look for email addresses, you would use this:

```
frame matches "(?i)[A-Z0-9._%~]+@[A-Z0-9.-]+\.[A-Z\>]{2,4}"
```

So those are just some ideas.

I hope you find this article and its content helpful. Comments are welcomed below. If you would like to see more articles by clicking the [patron link](#) where you will receive free bonus access to courses and more, or simply [buying us a cup of coffee](#) comments are welcome!

Wireshark

[< Prev](#)

Add comment

Name required

E-mail required, but not visible

Title

Write your comment here...

1000 characters left

☐ Notify me of follow-up comments

☐ Accept privacy policy