TraceWrangler

Table of contents

Getting Started

Welcome

TraceWrangler is a trace file toolkit running on Windows (and on Linux, using Wine) that supports the new PCAPng file format which is now the standard file format used by Wireshark. The most prominent use case for TraceWrangler is the easy sanitization/anonymization of packet trace files (sometimes called "capture files" or "packet captures"), removing or replacing sensitive data while being easy to use.

File and tasks

TraceWrangler generally works on a list of files. It doesn't matter if there's only one file in the list or hundreds. A couple of things can be performed using the pop up menu of the file list, but most things require creating a task and configuring its actions.

Windows only

The reason for being Windows only is that TraceWrangler is written in Delphi VCL, and the compiler can only produce Windows binaries at this time. The reason for choosing Delphi over some cross platform language is pretty simple: I could have spent years on learning a language well enough to be able to code a tool like TraceWrangler - or use those same years and start the project right away, with a language I already know. Well, know enough.

Also, you can always run TraceWrangler in a Windows VM if you have to. I agree that it's not as good as a native version, but right now there's not much I can do. I also heard that it runs without problems on Wine, but I haven't checked myself.

System requirements

There are no special requirements, except that TraceWrangler only runs on Windows. It can't hurt to have fast disks, at least 2GByte of RAM, and a fast CPU. Multiple CPU cores do not help at this time since TraceWrangler is single threaded for most of what it does. Having more RAM is a big help, because it allows working with a lot of large files at the same time.

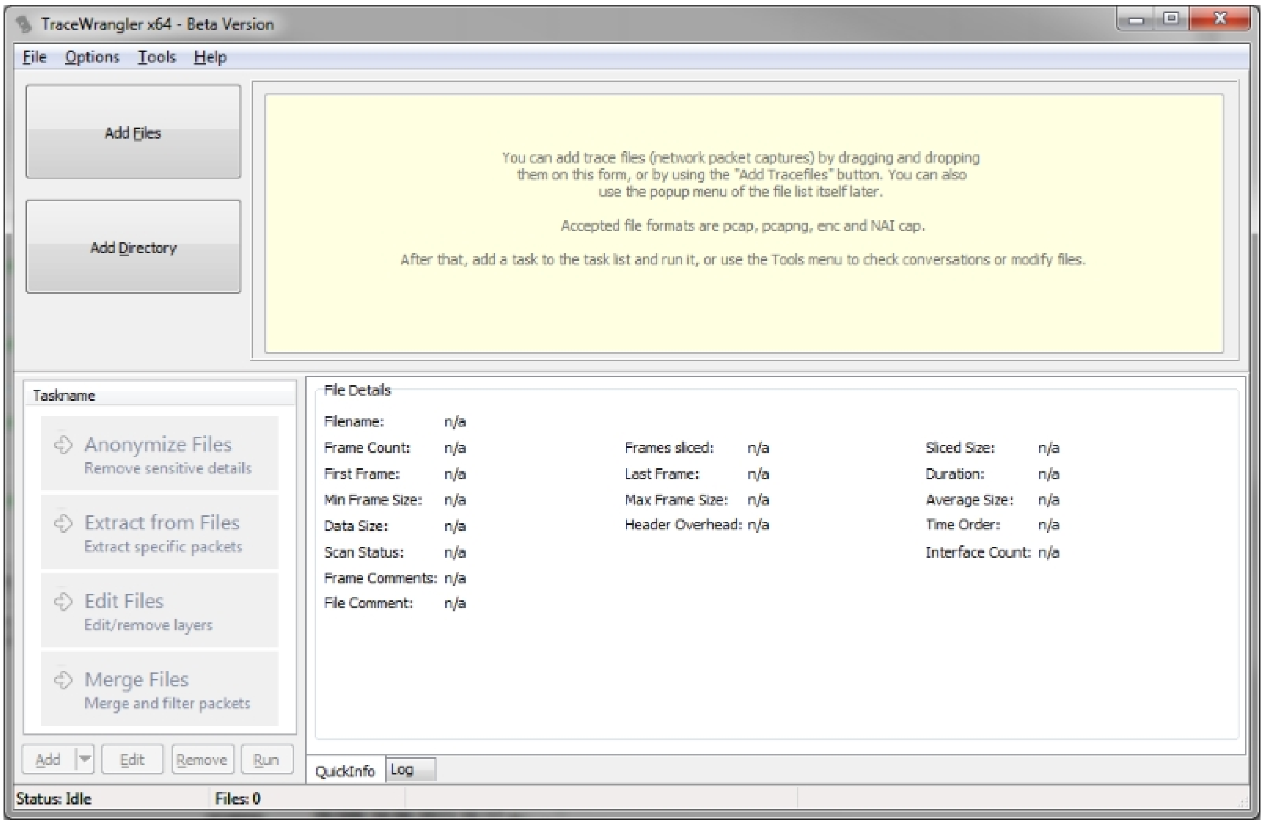
Getting help

There are a couple of resources that can help with using TraceWrangler:

1. First of all, it's this help file, or the online documentation at http://www.tracewrangler.com/documentation.
2. I did presentations about trace file anonymization at Sharkfest 2011 and 2013. The 2011 presentation is more basic and does not mention TraceWrangler as it didn't exist at the time. The 2013 presentation had a strong focus on using the anonymization task and is also available as a video recording on Youtube.
3. You can always send me an email to jasper@packet-foo.com. In case of bug reports please include a detailed problem description and (if possible) a trace file plus the task file if you still have it.

Starting TraceWrangler

When you start TraceWrangler it looks like this:



You can add trace files (network packet captures) by dragging and dropping them on this form, or by using the "Add Tracefiles" button. You can also use the popup menu of the file list itself later.

Accepted file formats are pcap, pcapng, enc and NAI cap.

After that, add a task to the task list and run it, or use the Tools menu to check conversations or modify files.

File Details

Filename:	n/a				
Frame Count:	n/a	Frames sliced:	n/a	Sliced Size:	n/a
First Frame:	n/a	Last Frame:	n/a	Duration:	n/a
Min Frame Size:	n/a	Max Frame Size:	n/a	Average Size:	n/a
Data Size:	n/a	Header Overhead:	n/a	Time Order:	n/a
Scan Status:	n/a			Interface Count:	n/a
Frame Comments:	n/a				
File Comment:	n/a				

Status: Idle   Files: 0

To get started, you should add trace files to the list of files. There are several ways do to that:

- Use the "Add Files" or "Add Directory" button
- Use the main menu to open a file dialog where you can select one or multiple files to be added
- Drag and drop files on to the main form
- Use the pop-up menu on the file list once it is shown
- Any parameter specified at the command line will be considered a capture file and added if it is a capture file

After adding a file it will be listed in the top half of the form. Informations shown include the size, the capture file type, the time stamp of the first frame found in the file, the duration of the capture (if available), the count of frames in the file (if available), and the current status.

Note: right now, TraceWrangler can only process files up to 2GB in size. The reason for that is the way files are read from disk, which does not support larger files without some code changes (which are on my ToDo list, but it can take some time - I'll probably have to rewrite some of the loader code). There will be a warning message if you try to add larger files.

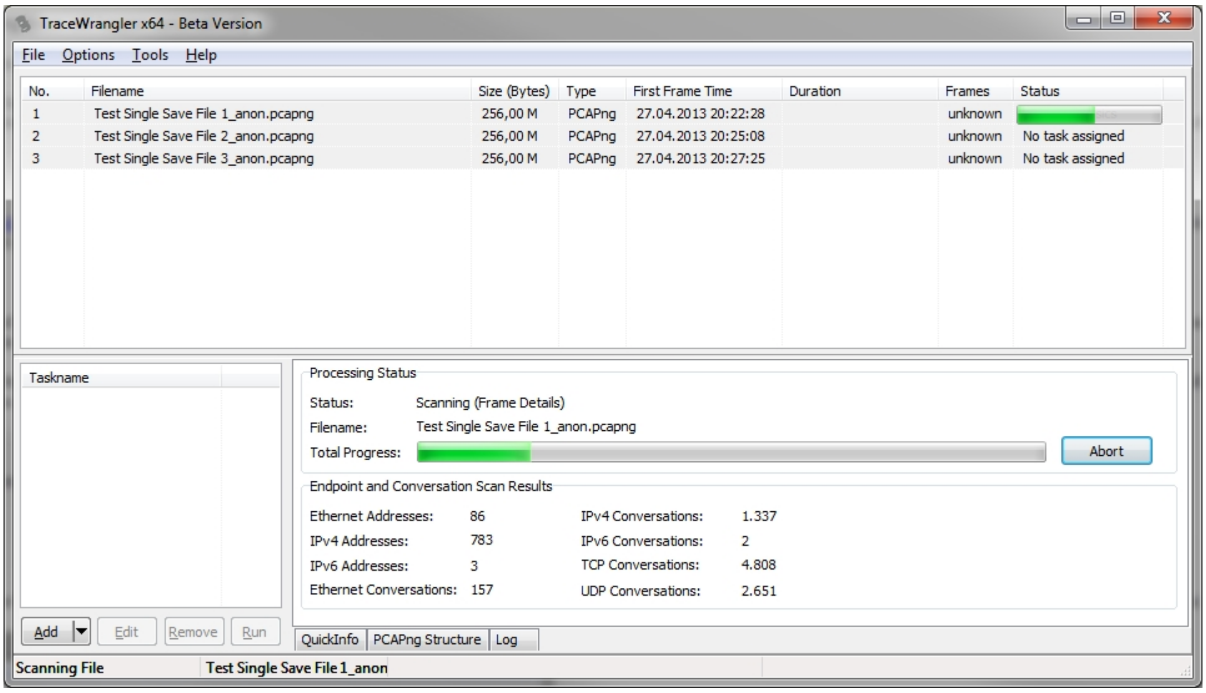
Scanning Files

When adding a capture file to TraceWrangler, the following steps will happen automatically:

1. TraceWrangler reads the first couple of bytes to determine if the file type is in fact a capture file by comparing those bytes to known values ("File Magic"). If a match is found the file is added to the list
2. the file size is be determined
3. the file is opened and the first frame is read to determine its absolute time stamp (down to the nanosecond)
4. in case the file type is PCAPng, the last frame is read to determine its absolute time stamp, too. This is possible because PCAPng allows reading blocks forward from the beginning or backwards from the end of the file.

Scanning files

If the file size is less than the AutoScan threshold defined in the preferences (or when the threshold is zero), the file will be scanned completely, unless it was scanned before. The scan reads every frame and gathers statistics for the file details pane. At the same time, all endpoints and conversations are identified. In case of PCAPng files, the block structure is recorded as well. The processing status pane shows the number of addresses and conversations found during the scan of the current file.



No.	Filename	Size (Bytes)	Type	First Frame Time	Duration	Frames	Status
1	Test Single Save File 1_anon.pcapng	256,00 M	PCAPng	27.04.2013 20:22:28		unknown	
2	Test Single Save File 2_anon.pcapng	256,00 M	PCAPng	27.04.2013 20:25:08		unknown	No task assigned
3	Test Single Save File 3_anon.pcapng	256,00 M	PCAPng	27.04.2013 20:27:25		unknown	No task assigned

Processing Status

Status:	Scanning (Frame Details)
Filename:	Test Single Save File 1_anon.pcapng
Total Progress:	

Endpoint and Conversation Scan Results

Ethernet Addresses:	86	IPv4 Conversations:	1.337
IPv4 Addresses:	783	IPv6 Conversations:	2
IPv6 Addresses:	3	TCP Conversations:	4.808
Ethernet Conversations:	157	UDP Conversations:	2.651

Scanning File   Test Single Save File 1_anon

Scanning files for endpoints and conversations is required for some of the advanced functionality of TraceWrangler. If the files are not scanned (because of being larger than the AutoScan threshold) you can't lookup IP addresses for anonymization tasks, and the endpoint/conversation table can't be displayed.

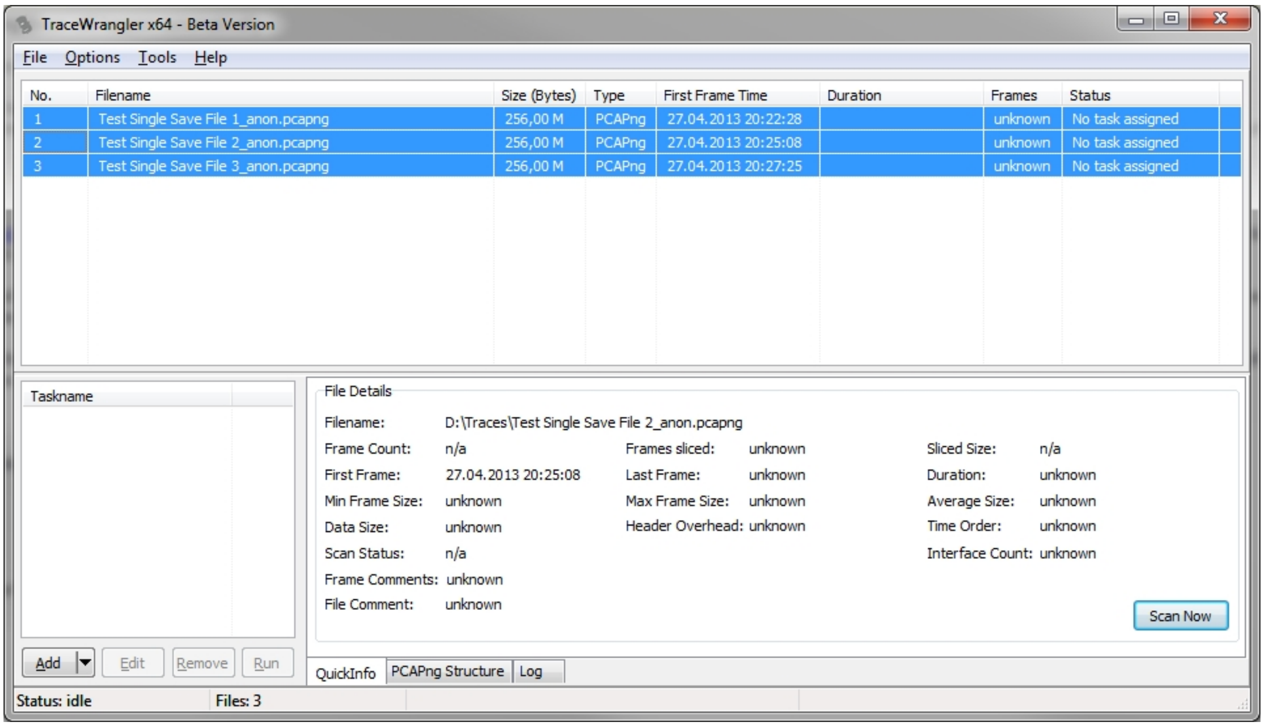
To avoid having to scan files again and again each time they are added to the list, TraceWrangler writes all scan results to a database called traceintel.db. When the same file is added to the file list again, TraceWrangler will check if there is a database entry for that file, and if so, read the details from the database instead.

In case a file isn't scanned you can do two things:

1. Raise the AutoScan threshold to allow scanning larger files, or setting it to 0 to ignore file size limits copmpletely
2. Forcing a manual scan of one or more files

Performing manual scans

To perform a manual scan, select one or more files in the file list. When TraceWrangler determines that at least one file hasn't been scanned yet it will display a "Scan Now" button in lower right corner of the file details pane:



No.	Filename	Size (Bytes)	Type	First Frame Time	Duration	Frames	Status
1	Test Single Save File 1_anon.pcapng	256,00 M	PCAPng	27.04.2013 20:22:28		unknown	No task assigned
2	Test Single Save File 2_anon.pcapng	256,00 M	PCAPng	27.04.2013 20:25:08		unknown	No task assigned
3	Test Single Save File 3_anon.pcapng	256,00 M	PCAPng	27.04.2013 20:27:25		unknown	No task assigned

File Details

Filename: D:\Traces\Test Single Save File 2_anon.pcapng

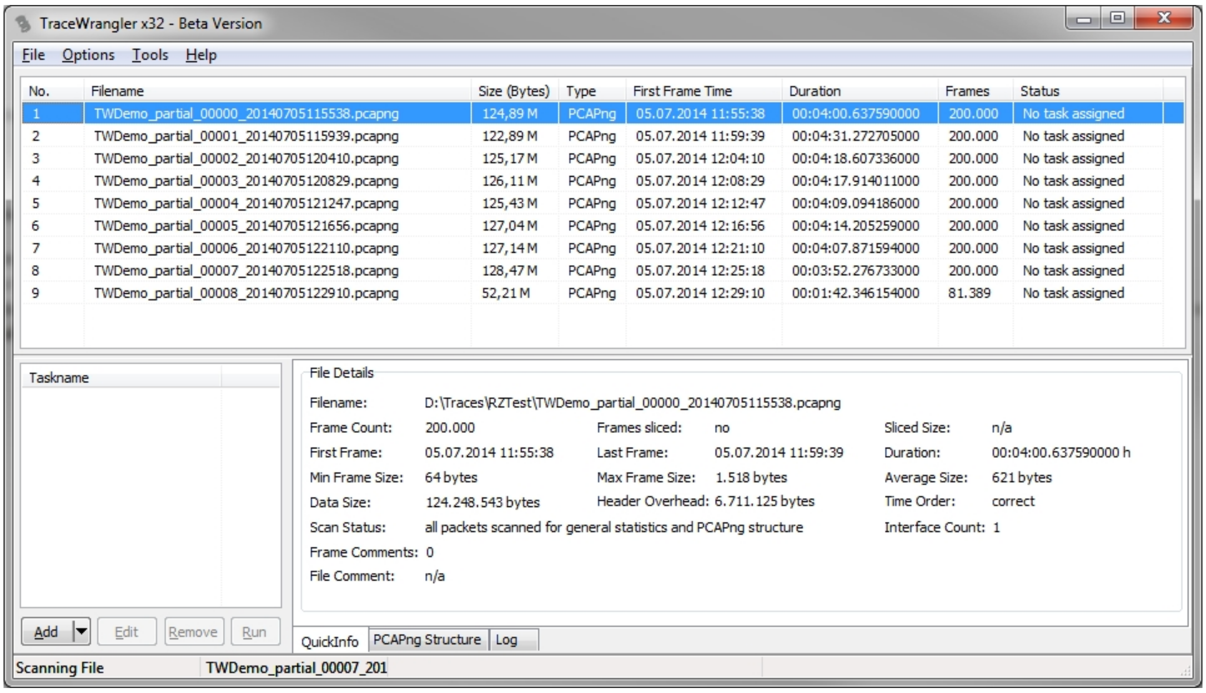
Frame Count:	n/a	Frames sliced:	unknown	Sliced Size:	n/a
First Frame:	27.04.2013 20:25:08	Last Frame:	unknown	Duration:	unknown
Min Frame Size:	unknown	Max Frame Size:	unknown	Average Size:	unknown
Data Size:	unknown	Header Overhead:	unknown	Time Order:	unknown
Scan Status:	n/a			Interface Count:	unknown
Frame Comments:	unknown				
File Comment:	unknown				

The Main Window

The file list

TraceWrangler performs tasks on the files added to the file list. There are also a couple of things you can do to any file (including a subset) in the list directly by selecting it and using the pop-up menu.

This is how the list looks like when a couple of files were added:


TraceWrangler x32 - Beta Version
File Options Tools Help
Taskname
Add Edit Remove Run
QuickInfo PCAPng Structure Log
Scanning File TWDemo_partial_00007_201

No.	Filename	Size (Bytes)	Type	First Frame Time	Duration	Frames	Status
1	TWDemo_partial_00000_20140705115538.pcapng	124,89 M	PCAPng	05.07.2014 11:55:38	00:04:00.637590000	200.000	No task assigned
2	TWDemo_partial_00001_20140705115939.pcapng	122,89 M	PCAPng	05.07.2014 11:59:39	00:04:31.272705000	200.000	No task assigned
3	TWDemo_partial_00002_20140705120410.pcapng	125,17 M	PCAPng	05.07.2014 12:04:10	00:04:18.607336000	200.000	No task assigned
4	TWDemo_partial_00003_20140705120829.pcapng	126,11 M	PCAPng	05.07.2014 12:08:29	00:04:17.914011000	200.000	No task assigned
5	TWDemo_partial_00004_20140705121247.pcapng	125,43 M	PCAPng	05.07.2014 12:12:47	00:04:09.094186000	200.000	No task assigned
6	TWDemo_partial_00005_20140705121656.pcapng	127,04 M	PCAPng	05.07.2014 12:16:56	00:04:14.205259000	200.000	No task assigned
7	TWDemo_partial_00006_20140705122110.pcapng	127,14 M	PCAPng	05.07.2014 12:21:10	00:04:07.871594000	200.000	No task assigned
8	TWDemo_partial_00007_20140705122518.pcapng	128,47 M	PCAPng	05.07.2014 12:25:18	00:03:52.276733000	200.000	No task assigned
9	TWDemo_partial_00008_20140705122910.pcapng	52,21 M	PCAPng	05.07.2014 12:29:10	00:01:42.346154000	81.389	No task assigned

File Details

Filename: D:\Traces\RZTest\TWDemo_partial_00000_20140705115538.pcapng

Frame Count:	200.000	Frames sliced:	no	Sliced Size:	n/a
First Frame:	05.07.2014 11:55:38	Last Frame:	05.07.2014 11:59:39	Duration:	00:04:00.637590000 h
Min Frame Size:	64 bytes	Max Frame Size:	1.518 bytes	Average Size:	621 bytes
Data Size:	124.248.543 bytes	Header Overhead:	6.711.125 bytes	Time Order:	correct
Scan Status:	all packets scanned for general statistics and PCAPng structure			Interface Count:	1
Frame Comments:	0				
File Comment:	n/a				

After adding a file it will be listed in the top half of the form. Informations shown include the size, the capture file type, the timestamp of the first frame found in the file, the duration of the capture (if available), the count of frames in the file (if available), and the current status. For some of the columns to be filled (like the number of frames) the file needs to be scanned first. There are a few things worth mentioning when it comes to the file list:

File Type

The file type is not determined by the file extension. TraceWrangler checks the file "magic" instead, which is a specific sequence of bytes at the beginning of each file. So it doesn't matter what extension a file has.

Sorting and processing order

TraceWrangler automatically sorts all files by the timestamp of the first frame in the file, and doesn't care about the file name. Processing the trace files will be performed in the same order, which is why the sorting is done by time stamp rather than the name. Some processing settings may require the frames to be read in chronological order whenever possible, so TraceWrangler adjusts the files accordingly. Even though the time column doesn't show more than the seconds of the first frame, the actual sorting is performed down to the nanosecond, if necessary.

Duration and Frame count

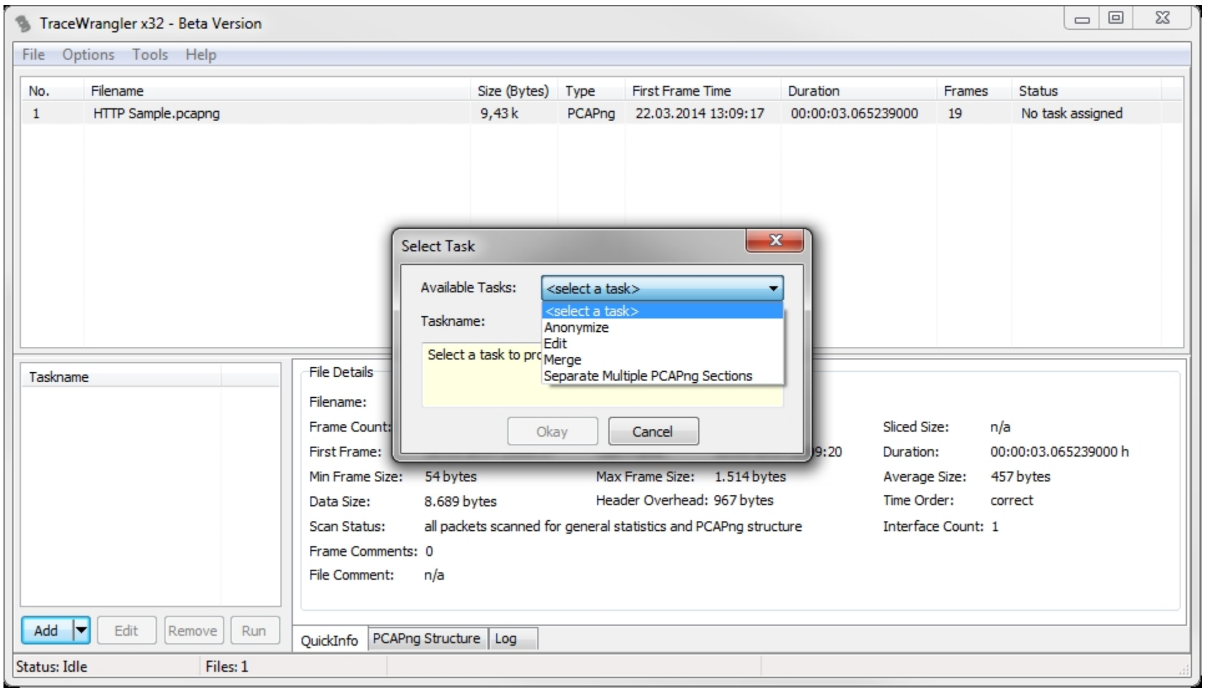
Duration and frame count require the whole file to be scanned first. TraceWrangler may do that automatically based on the size of the file and a setting in the preferences. If no scan was performed, the duration and frame count are not available. You can trigger a manual scan in that case, by using the according button at the bottom in the trace details pane.

Status

The status is primarily used when processing files and will show a progress bar while the current task is working with the trace file.

The task list

If you want to process the list of files you now need to add a task to perform. For that, click on the "Add" button at the bottom of the tasks pane. You can also use the popup menu of the task list to import existing .task files that you exported before, or simply drag them onto the form. As a quick way to add an "instant" task without having to type a name you can use the popup menu that shows when you select the triangle at the right of the Add button. You can also specify a task file as a command line parameter.



No.	Filename	Size (Bytes)	Type	First Frame Time	Duration	Frames	Status
1	HTTP Sample.pcapng	9,43 k	PCAPng	22.03.2014 13:09:17	00:00:03.065239000	19	No task assigned

Select a task to pro

File Details

Filename:					
Frame Count:				Sliced Size:	n/a
First Frame:			9:20	Duration:	00:00:03.065239000 h
Min Frame Size:	54 bytes	Max Frame Size:	1.514 bytes	Average Size:	457 bytes
Data Size:	8.689 bytes	Header Overhead:	967 bytes	Time Order:	correct
Scan Status:	all packets scanned for general statistics and PCAPng structure			Interface Count:	1
Frame Comments:	0				
File Comment:	n/a				

Status: Idle Files: 1

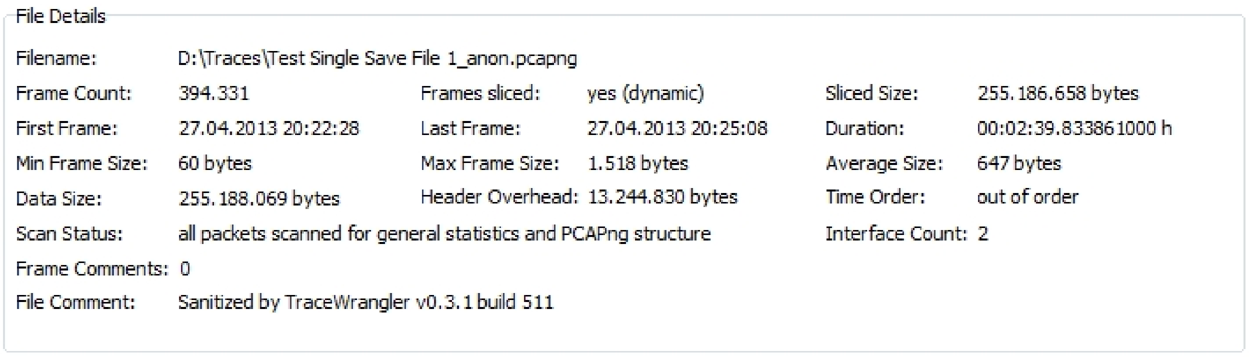
Task Settings

All task settings will be stored in an SQLite database file. You can find that find in the data directory that TraceWrangler uses, which is (unless configured otherwise in the preferences):

C:\users\<*username*>\AppData\Roaming\TraceWrangler\TaskData

The file details pane

The file details pane displays various statistics and other information about the currently selected trace file.


File Details

Filename:	D:\Traces\Test Single Save File 1_anon.pcapng				
Frame Count:	394.331	Frames sliced:	yes (dynamic)	Sliced Size:	255.186.658 bytes
First Frame:	27.04.2013 20:22:28	Last Frame:	27.04.2013 20:25:08	Duration:	00:02:39.833861000 h
Min Frame Size:	60 bytes	Max Frame Size:	1.518 bytes	Average Size:	647 bytes
Data Size:	255.188.069 bytes	Header Overhead:	13.244.830 bytes	Time Order:	out of order
Scan Status:	all packets scanned for general statistics and PCAPng structure			Interface Count:	2
Frame Comments:	0				
File Comment:	Sanitized by TraceWrangler v0.3.1 build 511				

A couple of the items may need some explanation, e.g.

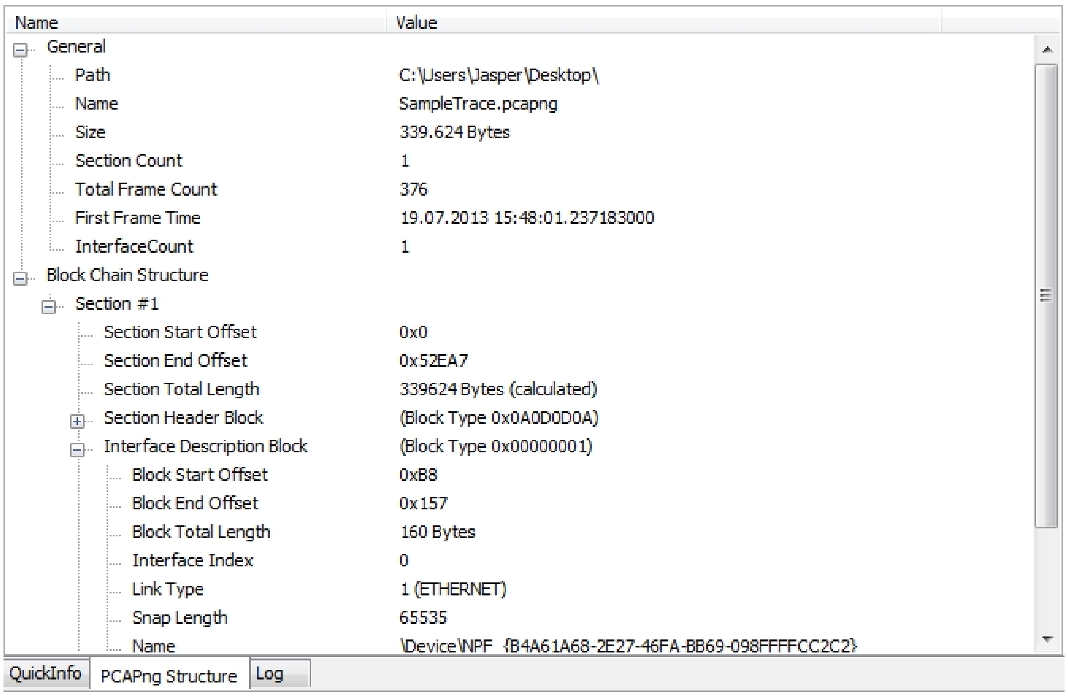
Frames sliced: if there are frames in the capture that were cut short during capture this will tell you at what offset the frames were cut. If the cut is dynamic (e.g. using TraceWrangler's anonymization functionality to cut after certain protocol layers) it will just say "dynamic".

Time Order: if the trace contains frames with negative delta times (as can happen while capturing on 2 or more interfaces at the same time) this will say "out of order".

The PCAPng structure viewer

When selecting a single PCAPng trace file in the file list TraceWrangler will show an additional tab on the lower right pane, containing file format details. This can be helpful if you want to know what kind of sections and blocks a trace file contains, so if you're developing your own PCAPng code you can use TraceWrangler to verify your written files. Of course, this tab will only be shown for PCAPng files.

This screen shot shows an example:



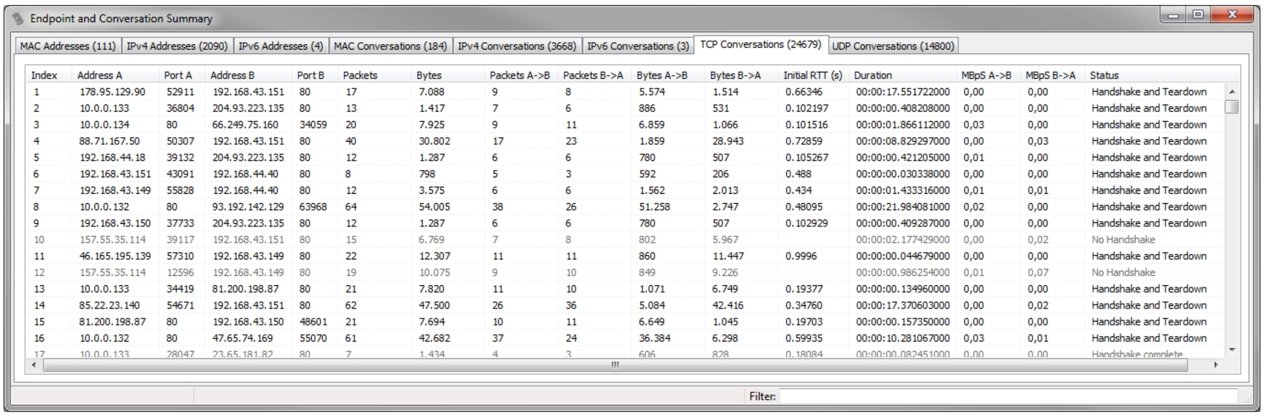
Name	Value
General	
Path	C:\Users\Jasper\Desktop\
Name	SampleTrace.pcapng
Size	339.624 Bytes
Section Count	1
Total Frame Count	376
First Frame Time	19.07.2013 15:48:01.237183000
InterfaceCount	1
Block Chain Structure	
Section #1	
Section Start Offset	0x0
Section End Offset	0x52EA7
Section Total Length	339624 Bytes (calculated)
Section Header Block	(Block Type 0x0A0D0D0A)
Interface Description Block	(Block Type 0x00000001)
Block Start Offset	0xB8
Block End Offset	0x157
Block Total Length	160 Bytes
Interface Index	0
Link Type	1 (ETHERNET)
Snap Length	65535
Name	\Device\NPF_{B4A61A68-2E27-46FA-BB69-098FFFFCC2C2}

Tools

Communication Details

The endpoint and conversation summary contains a list of all Ethernet, IPv4/IPv6 endpoints and conversations, plus TCP/UDP conversations. It is similar to the endpoint and conversations statistics of Wireshark, but all in one form and for all the files currently listed in the file list (that have been scanned first, of course). The number in brackets behind the header tab caption is the amount of lines the according list contains. All columns can be sorted by clicking on the column headers.

What is special about the communication summary is that it is created from the deep scans of all files in the current file list, so it shows an aggregated view over all files. This means that e.g if a TCP conversation starts in one file, continues over the next and ends in the third file it will only be listed **once**. All counters will reflect the totals of that conversations. So if you close the conversation summary, remove a file from the list and open the summary again, you'll see its lines have changed accordingly.



Index	Address A	Port A	Address B	Port B	Packets	Bytes	Packets A->B	Packets B->A	Bytes A->B	Bytes B->A	Initial RTT (s)	Duration	MBpS A->B	MBpS B->A	Status
1	178.95.129.90	52911	192.168.43.151	80	17	7.088	9	8	5.574	1.514	0.66346	00:00:17.551722000	0,00	0,00	Handshake and Teardown
2	10.0.0.133	36804	204.93.223.135	80	13	1.417	7	6	886	531	0.102197	00:00:00.408208000	0,00	0,00	Handshake and Teardown
3	10.0.0.134	80	66.249.75.160	34059	20	7.925	9	11	6.859	1.066	0.101516	00:00:01.866112000	0,03	0,00	Handshake and Teardown
4	88.71.167.50	50307	192.168.43.151	80	40	30.802	17	23	1.859	28.943	0.72859	00:00:08.829297000	0,00	0,03	Handshake and Teardown
5	192.168.44.18	39132	204.93.223.135	80	12	1.287	6	6	780	507	0.105267	00:00:00.421205000	0,01	0,00	Handshake and Teardown
6	192.168.43.151	43091	192.168.44.40	80	8	798	5	3	592	206	0.488	00:00:00.030338000	0,00	0,00	Handshake and Teardown
7	192.168.43.149	55828	192.168.44.40	80	12	3.575	6	6	1.562	2.013	0.434	00:00:01.433316000	0,01	0,01	Handshake and Teardown
8	10.0.0.132	80	93.192.142.129	63968	64	54.005	38	26	51.258	2.747	0.48095	00:00:21.984081000	0,02	0,00	Handshake and Teardown
9	192.168.43.150	37733	204.93.223.135	80	12	1.287	6	6	780	507	0.102929	00:00:00.409287000	0,00	0,00	Handshake and Teardown
10	157.55.35.114	39117	192.168.43.151	80	15	6.769	7	8	802	5.967		00:00:02.177429000	0,00	0,02	No Handshake
11	46.165.195.139	57310	192.168.43.149	80	22	12.307	11	11	860	11.447	0.9996	00:00:00.044679000	0,00	0,00	Handshake and Teardown
12	157.55.35.114	12596	192.168.43.149	80	19	10.075	9	10	849	9.226		00:00:00.986254000	0,01	0,07	No Handshake
13	10.0.0.133	34419	81.200.198.87	80	21	7.820	11	10	1.071	6.749	0.19377	00:00:00.134960000	0,00	0,00	Handshake and Teardown
14	85.22.23.140	54671	192.168.43.151	80	62	47.500	26	36	5.084	42.416	0.34760	00:00:17.370603000	0,00	0,02	Handshake and Teardown
15	81.200.198.87	80	192.168.43.150	48601	21	7.694	10	11	6.649	1.045	0.19703	00:00:00.157350000	0,00	0,00	Handshake and Teardown
16	10.0.0.132	80	47.65.74.169	55070	61	42.682	37	24	36.384	6.298	0.59935	00:00:10.281067000	0,03	0,01	Handshake and Teardown
17	10.0.0.133	28047	23.65.181.82	80	7	1.434	4	3	606	828	0.18084	00:00:00.082451000	0,00	0,00	Handshake complete

General attributes

While scanning the files for endpoints and conversations TraceWrangler automatically detects and records the following attributes:

1. Count and indexes of interfaces packets for the endpoint or conversation were captured on
2. Conversations that are only flowing in one direction (often indicating trouble with the capture setup)
3. Frame out-of-order, meaning that there are negative delta times between at least two packets of a conversation. For more information, take a look at my blog post here. If this is diagnosed, the status column will show a message telling you that there were out-of-order frames instead of TCP conversation status.
4. Conversations that span multiple files
5. Conversations where the Ethernet frame has trailing bytes after the normal TCP/UDP/IPv4/IPv6 payload
6. Hard slicing, meaning that the frame was only captured partially and the information about the original length is lost. This usually leads to a lot of TCP expert warnings in Wireshark because the TCP dissector thinks that segments are missing.

TCP Conversations

The TCP conversations are special, because they add two columns that are rarely seen in other tools: Initial RTT and Status.

Initial RTT: Initial RTT is the duration of the TCP three way handshake. It is quite important because it can be used to verify TCP behavior, especially when it comes to lost segments and retransmissions. It can also show conversations that have issues with latency or packet loss just by sorting for high values. The conversation list shows Initial RTT in seconds.

Status: TraceWrangler keeps track of the TCP connection setup and tear-down. There are a couple of possible values, most of them verified by checking sequence numbers, which will be shown in the status column:

1. **No Handshake, no teardown**: no SYN flags was seen for this connection, so neither SYN nor SYN/ACK were found. Also there was no FIN or RST at the end.
2. **SYN sent**: only a SYN packet was seen, but no SYN/ACK
3. **SYN/ACK sent**: a SYN/ACK packet was seen, but no SYN before that
4. **SYN Two Way**: both IPs have sent a SYN packet for the same conversation ("Simultaneous initiation"/"four way handshake", see page 32 of RFC 793), but there was no SYN/ACK from either side
5. **TCP half open**: SYN and SYN/ACK were seen and are correct, but no ACK was found. Handshake is incomplete.
6. **SYN rejected**: a SYN packet was seen but answered by a reset packet, refusing the connection
7. **Handshake complete**: a full three was handshake was observed, but no FIN or RST at the end of the connection, so parts are probably missing at the end
8. **Handshake complete after multi SYN retry:** same as "Handshake complete", but there were two or more SYN packets before the SYN/ACK was seen. This points to the server not accepting incoming connections right away, or packet loss of the initial SYN packet.
9. **Handshake complete after initial SYN reject**: this is a very unusual connection, because the server rejected at least one SYN packet with an RST packet before accepting the same connection for another SYN packet
10. **Handshake and Teardown**: a full three way handshake was observer, and a FIN or RST at the end. This may not be a complete FIN/ACK/FIN/ACK, as TraceWrangler just cares if there is at least one FIN. This status may also be seen with "multi SYN retry" or "initial SYN reject" attribute.
11. **Teardown**: when a handshake wasn't seen but packets of an active conversation followed by a FIN or RST at the end.

12. **New socket closed due to client inactivity**: this indicates that a TCP three way handshake was completed, but not data was transfered before tearing down the session again. The delta time between established session and teardown must be at least twice the iRTT.

Additional status messages may appear that aren't directly related to the TCP layer, but may be of interest or impact:

1. **Spans multiple files**: If TraceWrangler detected that a TCP conversation is spanning more than one file it will add a message about it to the status.
2. **Frames out-of-order:** In case a file has out-of-order frames (which usually only happens when capturing on multiple interfaces) it will show a message about that fact instead of the normal TCP status messages. Out-of-order frames prevent detecting the status correctly, so to be able to get the correct status the file needs to be reordered by absolute time stamp first.
3. **Has Ethernet trailer**: if bytes are found after the TCP payload (as indicated by the IPv4/IPv6 payload length) this message will be shown
4. **IP fragmented**: TCP should never be fragmented on the IP layer, but if it is, the status will tell you
5. **Port reused**: if two conversations are seen with the exact same IP and port combination the according list entry will be flagged

Filtering

A filter field is available in the lower right corner of the status bar. Anything you put in there will be used to match addresses against it for the currently visible list. In case of TCP conversations you can also look for specific status texts. When a filter is active, the current tab shows two numbers (displayed lines and total lines) instead of just the total count. To negate filters it can be specified with a starting "not " or "!". Pressing the **escape key** clears the filter instantly.

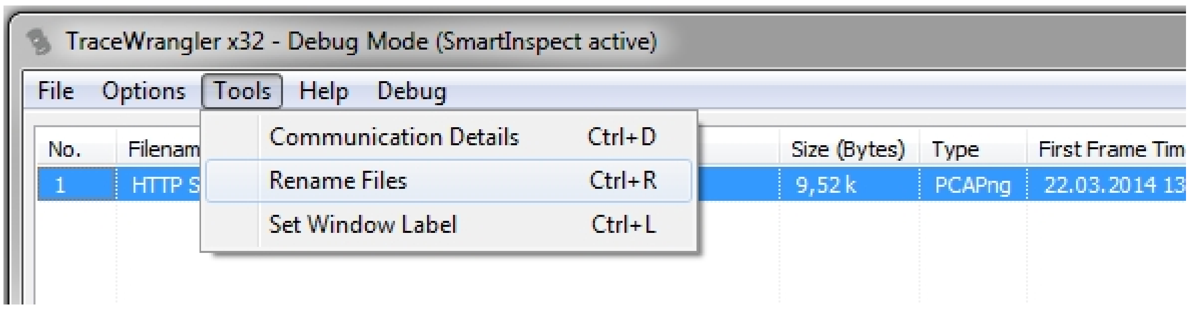
Copy as Shark Filter

A pop up menu is available for all lists, allowing you to copy a Wireshark/tshark display filter for any selected line to the clipboard. When multiple lines are selected a combination filter will be created, including all selected lines.

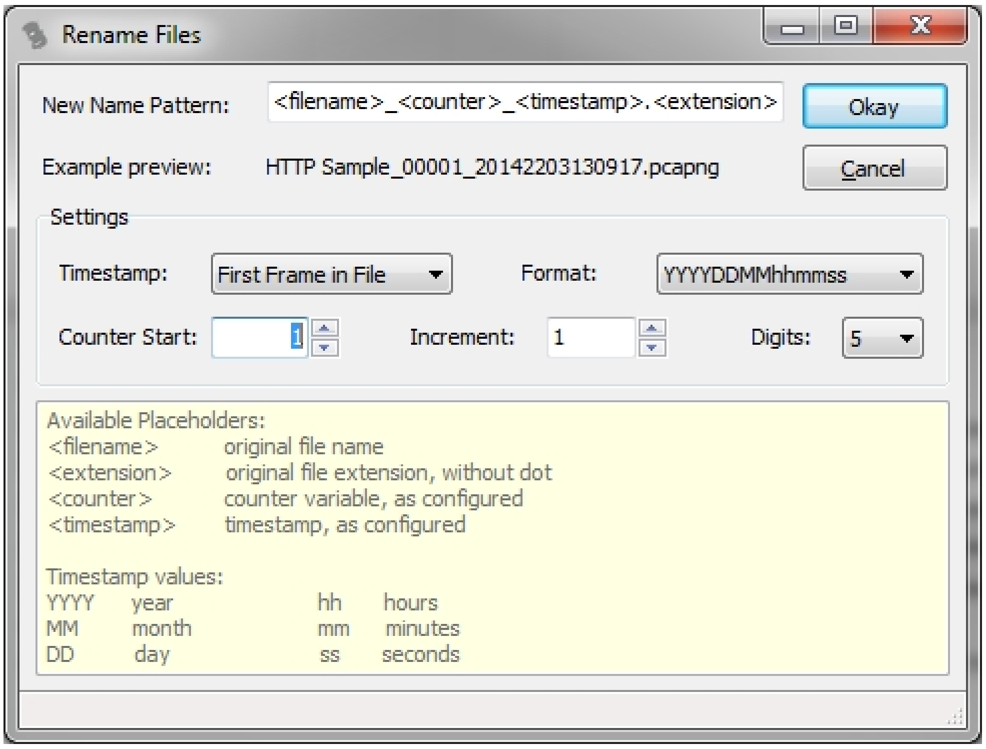
Renaming files

The "Rename Files" menu option allows renaming files with two primary goals in mind:
1. insert a timestamp into the file name to make it easier to determine the file you want
2. make the file naming scheme compatible with Wireshark ring buffer captures, so that the Wireshark "File Set" menu option works for them

The option is available from the main menu, but only if at least one file has been selected in the file list:


TraceWrangler x32 - Debug Mode (SmartInspect active)
File Options Tools Help Debug
Communication Details Ctrl+D
Rename Files Ctrl+R
Set Window Label Ctrl+L
No. Filenam Size (Bytes) Type First Frame Tim
1 HTTP S 9,52 k PCAPng 22.03.2014 13

Selecting the menu option will open a dialog box where the renaming scheme can be configured:



New Name Pattern:	<filename>_<counter>_<timestamp>.<extension>
Example preview:	HTTP Sample_00001_20142203130917.pcapng

Timestamp:	First Frame in File	Format:	YYYYDDMMhhmmss		
Counter Start:	1	Increment:	1	Digits:	5

Available Placeholders:

<filename>	original file name
<extension>	original file extension, without dot
<counter>	counter variable, as configured
<timestamp>	timestamp, as configured

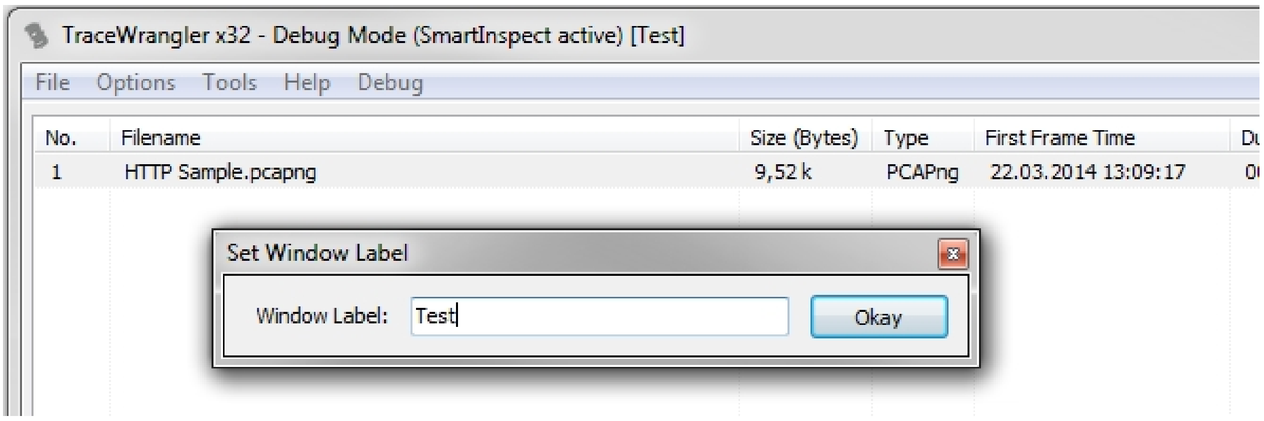
Timestamp values:

YYYY	year	hh	hours
MM	month	mm	minutes
DD	day	ss	seconds

There is a preview right below the pattern box to show how the files will look like.

Set Window Label

The option to set a window label is used to make multiple instances of TraceWrangler distinguishable.



No.	Filename	Size (Bytes)	Type	First Frame Time	Du
1	HTTP Sample.pcapng	9,52 k	PCAPng	22.03.2014 13:09:17	0

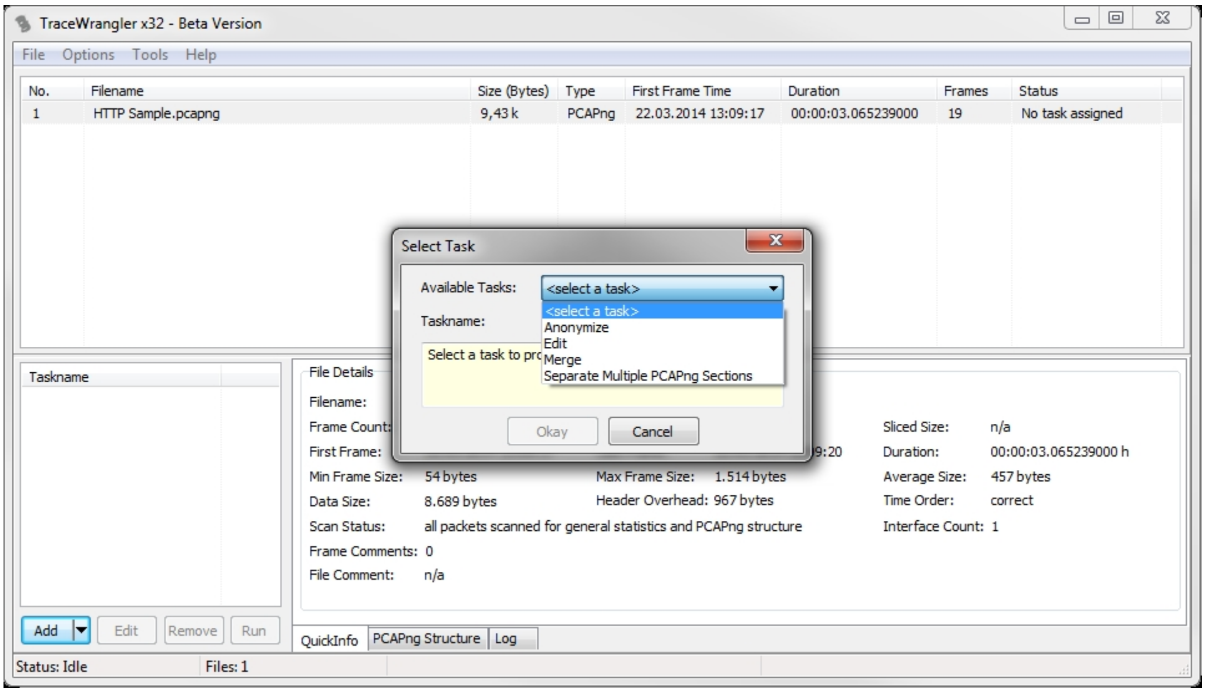
Window Label: Test

If you select the menu option a dialog box will open to allow setting a label, which will then be appended to the caption in square brackets.

Tasks

Working with tasks

If you want to process the list of files you now need to add a task to perform. For that, click on the "Add" button at the bottom of the tasks pane. You can also use the pop-up menu of the task list to import existing task files that you exported before, or simply drag them onto the form. As a quick way to add an "instant" task without having to type a name you can use the pop-up menu that shows when you select the triangle at the right of the Add button. You can also specify a task file as a command line parameter when starting TraceWrangler.


TraceWrangler x32 - Beta Version

File Options Tools Help

No.	Filename	Size (Bytes)	Type	First Frame Time	Duration	Frames	Status
1	HTTP Sample.pcapng	9,43 k	PCAPng	22.03.2014 13:09:17	00:00:03.065239000	19	No task assigned

Select Task

Available Tasks: <select a task>

Taskname:

Select a task to pro

<select a task>
Anonymize
Edit
Merge
Separate Multiple PCAPng Sections

Okay Cancel

Taskname

File Details

Filename:
Frame Count:
First Frame: 9:20
Min Frame Size: 54 bytes
Max Frame Size: 1.514 bytes
Data Size: 8.689 bytes
Header Overhead: 967 bytes
Scan Status: all packets scanned for general statistics and PCAPng structure
Frame Comments: 0
File Comment: n/a
Sliced Size: n/a
Duration: 00:00:03.065239000 h
Average Size: 457 bytes
Time Order: correct
Interface Count: 1

Add Edit Remove Run

QuickInfo PCAPng Structure Log

Status: Idle Files: 1

All task settings will be stored in an SQLite database file, one per task. You can find that file in the data directory that TraceWrangler uses, which is (unless configured otherwise in the preferences):

C:\users\<*username*>\AppData\Roaming\TraceWrangler\TaskData

Anonymization/Sanitization

Introduction

After selecting to add a new anonymization task the according settings dialog will automatically open up. By default a couple of settings are already configured, for example the randomization of IP addresses. This default setting is designed to provide reasonable sanitization without having to configure anything, at the price of all IP addresses being replaced by random addresses.

The section tree list on the left lists all layers or parts of a frame that you can anonymize or sanitize. When a specific section is set to do something it will be printed in bold letters so that it is easy to see which parts of a frame are going to be affected - or at least where the setting for the section isn't set to "Passthrough".

The "Tools" Button
If you want to start with a clean slate you can use the "Clear" option from the tool button menu, e.g when you only need to replace a few things and want to keep all the rest intact. The "Restore Factory Default" option replaces the currently set of options with the sanitization options TraceWrangler uses as a default when installed for the first time. You can set your own default set of options by selecting "Set as Default Profile", which can be helpful if you often sanitize capture files from a specific network and want to keep the replacements consistent.

Supported Protocols

This is the list of protocols and layers TraceWrangler currently supports when sanitizing files:

- Ethernet
- VLAN tags
- ARP/RARP
- Tunneling: AYIYA, GRE, GTP-U, VXLAN, Geneve (most are just passed through)

- IPv4
- IPv6, including Fragmentation Headers (more Extension Headers to come at a later time)
- TCP
- UDP
- ICMPv4
- ICMPv6
- DHCPv4
- NetFlow v5
- HSRP
- RTPS

If a protocol is not supported, TraceWrangler will do one of two things, depending on the payload settings you define on the General Settings section:

1. keep the unknown protocol intact, which may expose sensitive data
2. truncate the frame at the offset where the unknown protocol starts

The safe way of handling unknown protocols is to configure the sanitization task to remove them.

The Anonymization Process

The anonymization process is quite similar to what computers and other network nodes do when they send and receive packets and frames. TraceWrangler has a parsing engine that dissects the frame from Ethernet up using "parsers" (Ethernet being the only layer 2 protocol at the moment that a parser exists for). Each protocol or layer is processed by a specific parser that understands that protocol/layer and can extract information from it. It is important to know that parsers may not understand all aspects of a protocol; for example exotic TCP options are not parsed and thus the information of these options are not accessible. So basically, TraceWrangler will take a frame apart layer by layer, stopping only when there is nothing more to parse or when it doesn't have a parser for the next protocol. Right now, DHCPv4 is the only protocol that it can handle above layer 4, as you can tell from the list of supported protocols. When there is further data after the last layer that TraceWrangler could parse, it will be kept as payload in a byte buffer.

Other sanitization tools

The biggest difference between TraceWrangler and other trace file sanitization tools is that everything it does is always focused on keeping the result **as true to the original as possible**, in a way that you can still perform network analysis (and network forensics, hopefully) on it. E.g. it will not randomly replace an IP address like the Google DNS ("8.8.8.8") by blindly "rolling the dice", which may end up in something like "127.0.0.1" - which is never seen on a real network and would make any kind of analysis of the packets impossible. It will also try to keep the frame sizes as close to the original as possible, and it will not mess around with the timings. At all. If it does, it's a bug.

Parsers

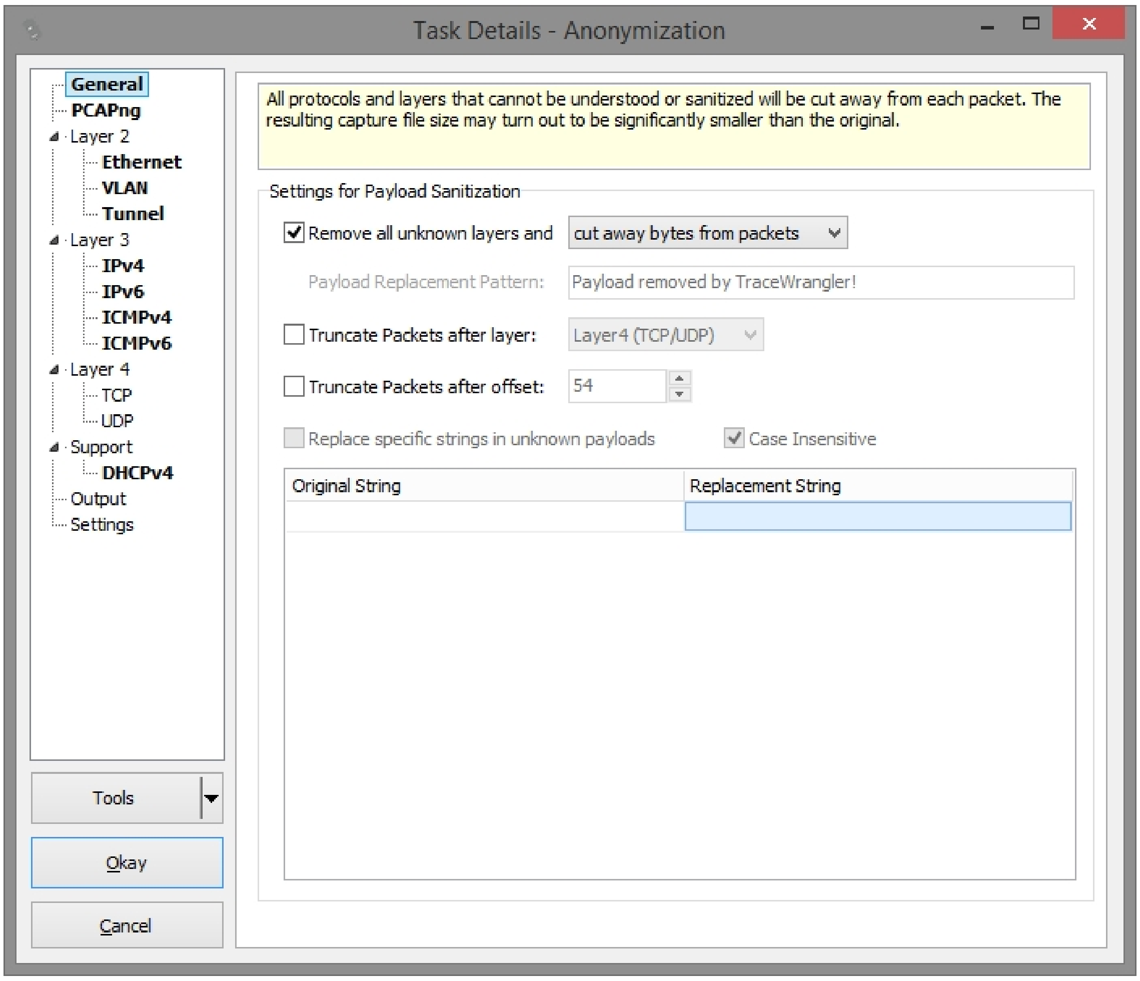
Parsers are the modules that dissect the frame content. I could have called them dissectors (like Wireshark does), but since they are a completely different code base I decided to call them parsers to avoid confusion. Wireshark is the king of dissectors, and I'll never get even close to what those can do :-)

After parsing the layers, TraceWrangler will reassemble the frame from top to button, going the other way than it was parsed. This is necessary because some layers depend on information of higher layers. E.g. TCP needs to calculate the CRC including the TCP payload. Also, this is the only way that layers can be left out (like VLAN tags), and to perform Defensive Transformation. That means that nothing that TraceWrangler did not understand while parsing a layer will be written to the new frame (unless you configured it to). E.g. in case of an exotic TCP option it will not make it into the sanitized frame because TraceWrangler can't be sure if it contains sensitive data.

Assemblers

The modules TraceWrangler uses to create the new, sanitized layers are called "assemblers". To be able to sanitize a protocol or a layer the according parser and assembler must exist. Right now there are some protocols where I have parsers for, but no assemblers yet, e.g. DNS. So until I find the time to code the according assembler, DNS cannot be sanitized.

Payload Settings



All protocols and layers that cannot be understood or sanitized will be cut away from each packet. The resulting capture file size may turn out to be significantly smaller than the original.

Settings for Payload Sanitization

- [✓] Remove all unknown layers and: cut away bytes from packets
- Payload Replacement Pattern: Payload removed by TraceWrangler!
- [ ] Truncate Packets after layer: Layer4 (TCP/UDP)
- [ ] Truncate Packets after offset: 54
- [ ] Replace specific strings in unknown payloads — [✓] Case Insensitive

Original String	Replacement String

Remove all unknown layers: when this setting is active, TraceWrangler will cut away all protocol layers of a frame that it does not understand. For example, there is no parser for SMTP at the moment, so if a packet is part of a SMTP conversation it will only be written to the sanitized file up to the TCP layer. That way no details will be kept that TraceWrangler does not know how to handle, which minimizes the risk of exposing sensitive information by mistake. You can choose one of three ways to handle unknown layers:

1. **cut away bytes from packets:** the unknown layers will be cut away and discarded. This will shorten the frame bytes, but TraceWrangler will keep the Wiresize value intact. That means that all calculations based on the original frame length will still be valid.
2. **replace bytes with zeroes:** in this mode, the packet length will stay the same, but all bytes of unknown layers will be replaced with zeroes
3. **replace bytes with pattern:** instead of cutting bytes away or replacing them with zeroes, TraceWrangler will use the value specified in the **Payload Replacement Pattern** field to overwrite unknown bytes. The pattern will be repeated as many times as required until all bytes are overwritten. Remaining pattern bytes will be discarded if the pattern is too long.

WARNING: Do **not** uncheck this setting unless you are fine with keeping all protocol layers, even those that cannot be sanitized!

Truncate packet after layer: this setting can be used to remove any packet content that is following the mentioned layer. For example, removing all payload after layer 4 (usually meaning TCP or UDP) will lead to all frames being written without any layer above TCP or UDP. TraceWrangler will determine where the layer to be kept ends and cut at the exact byte. This means that even when a TCP packet has options of different sizes it will keep the TCP header intact and remove the payload. The difference to doing this on a TAP is that TraceWrangler will keep the original frame length intact, e.g. for making it possible to keep working with TCP sequence numbers.

Truncate packet after offset: when this setting is active, TraceWrangler will truncate all packets after the offset that is specified.

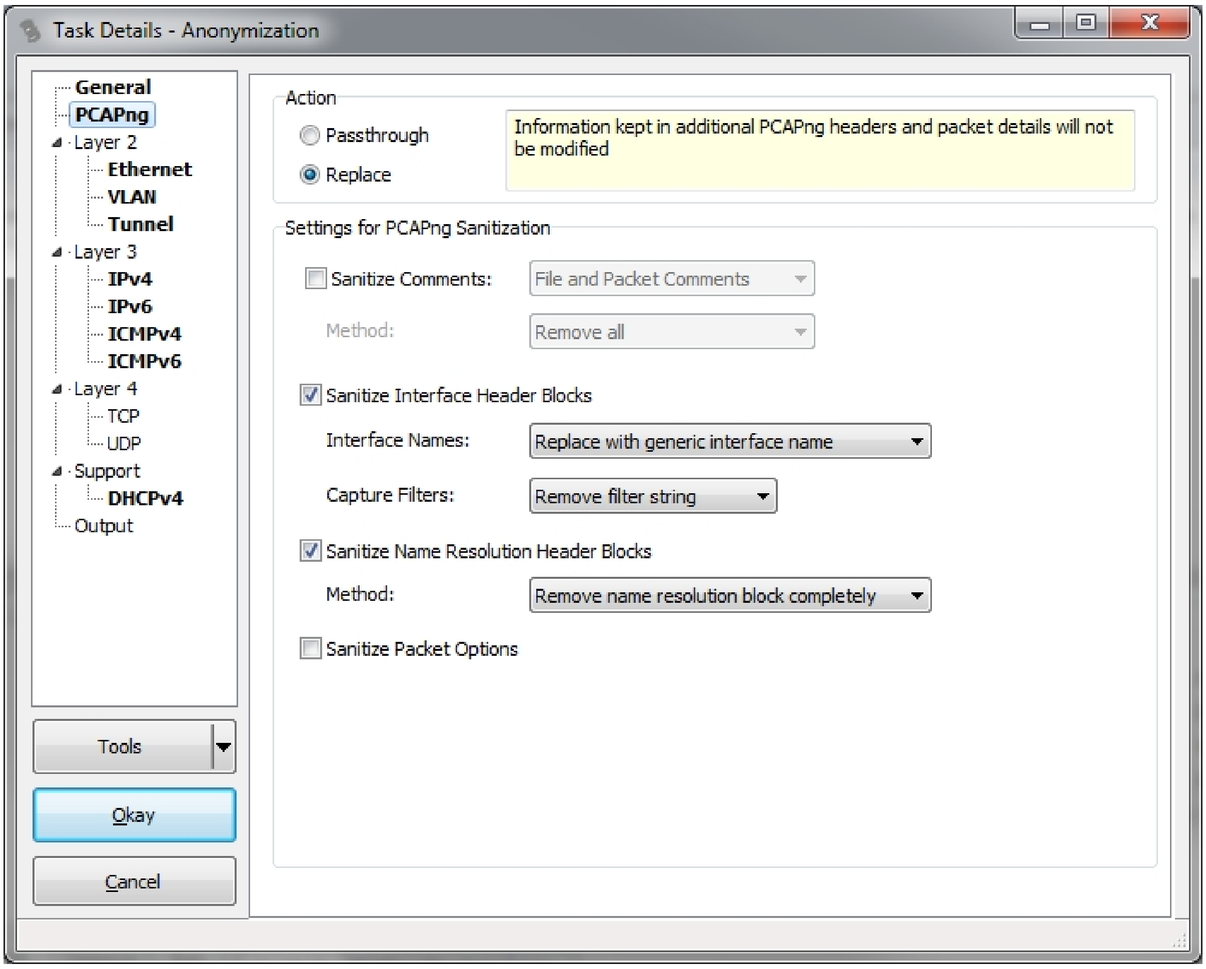
If more than one or even all settings are active, they will all be applied from top to bottom. This means that all unknown layers will be removed, followed by another truncate that will cut away anything beyond the layer specified, and finally the offset truncation will take place. So if for some reason you need a combination of these settings they will just work.

Replace specific strings in unknown payloads: If you want to replace strings in frames you can use this option to specify which string should be replacement with another string. There are a couple of important details if using this feature:

- This option is only available if **the packets are not truncated** after a layer, and if unknown payloads are kept in the sanitized file. Otherwise string replacement just makes no sense because there is no payload to look at, at least as long as TraceWrangler doesn't understand protocols transported by TCP.
- The replacement string needs to be the **same length** as the original. For that purpose a counter will appear in the upper right corner of the replacement list whenever a string replacement is added. You should try to make the replacement string the same length, otherwise it will be padded with spaces if too short, or truncated if too long.
- Adding more than one string is possible, but a little awkward at the moment: you need to press "**cursor down"** to add a new line.
- There is a pop up menu that allows **importing** and **exporting** strings, to make it easier to edit lots of strings without having to use the table editor on the form. The list needs to be comma separated, basically in the most simple from of "<key>,<value>". If you need to use commas in key or value they are escaped with a backslash.
- Strings can only be entered as numbers, characters and some extra characters - basically the lower 127 characters in the ASCII table.
- You can also use Snort style hexadecimal string syntax, embraced in two pipe characters, e.g. |00 01 02 03|. You can mix ASCII string elements with those hex elements if you wish. This allows the replacement of any type of byte string.
- Replacing payload content may lead to minor side effects of checksums being recalculated correctly even when the original frame had bad checksums. You can always take control of that behavior in the respective settings of each protocol layer.

PCAPng Settings

The settings in this section work on the extra header blocks that are part of each PCAPng file. Some of them are mandatory, others are optional. The official specification of the file format can be found here. The TraceWrangler setting is either "Passthrough" or "Replace". If "Passthrough" is selected, all settings in the lower pane are simply ignored. PCAPng settings only make sense when the file type is PCAPng, of course, and will be skipped in case of reading other file formats.


Task Details - Anonymization

Action
Passthrough [✓ Replace]

Information kept in additional PCAPng headers and packet details will not be modified

Settings for PCAPng Sanitization

☐ Sanitize Comments: File and Packet Comments
Method: Remove all

☑ Sanitize Interface Header Blocks
Interface Names: Replace with generic interface name
Capture Filters: Remove filter string

☑ Sanitize Name Resolution Header Blocks
Method: Remove name resolution block completely

☐ Sanitize Packet Options

Sanitize Comments: This setting allows removing comments from the file or the packets, or both.

Sanitize Interface Header Blocks: When a capture is taken by dumpcap and written in native PCAPng format, it will write the name of the network card into the mandatory interface description block that is required for each interface that was used in the capture. On Linux, this name is typically something like "eth0", which is pretty nondescript. On Windows, a GUID is created, which could be used to identify the exact interface at a later time because it is highly unique. The options for sanitization are to either keep it as it is, remove it completely by setting the name to an empty string, or to replace it with a generic interface name. The third option will have TraceWrangler create a new random GUID for any GUID it finds and replace the original with it.

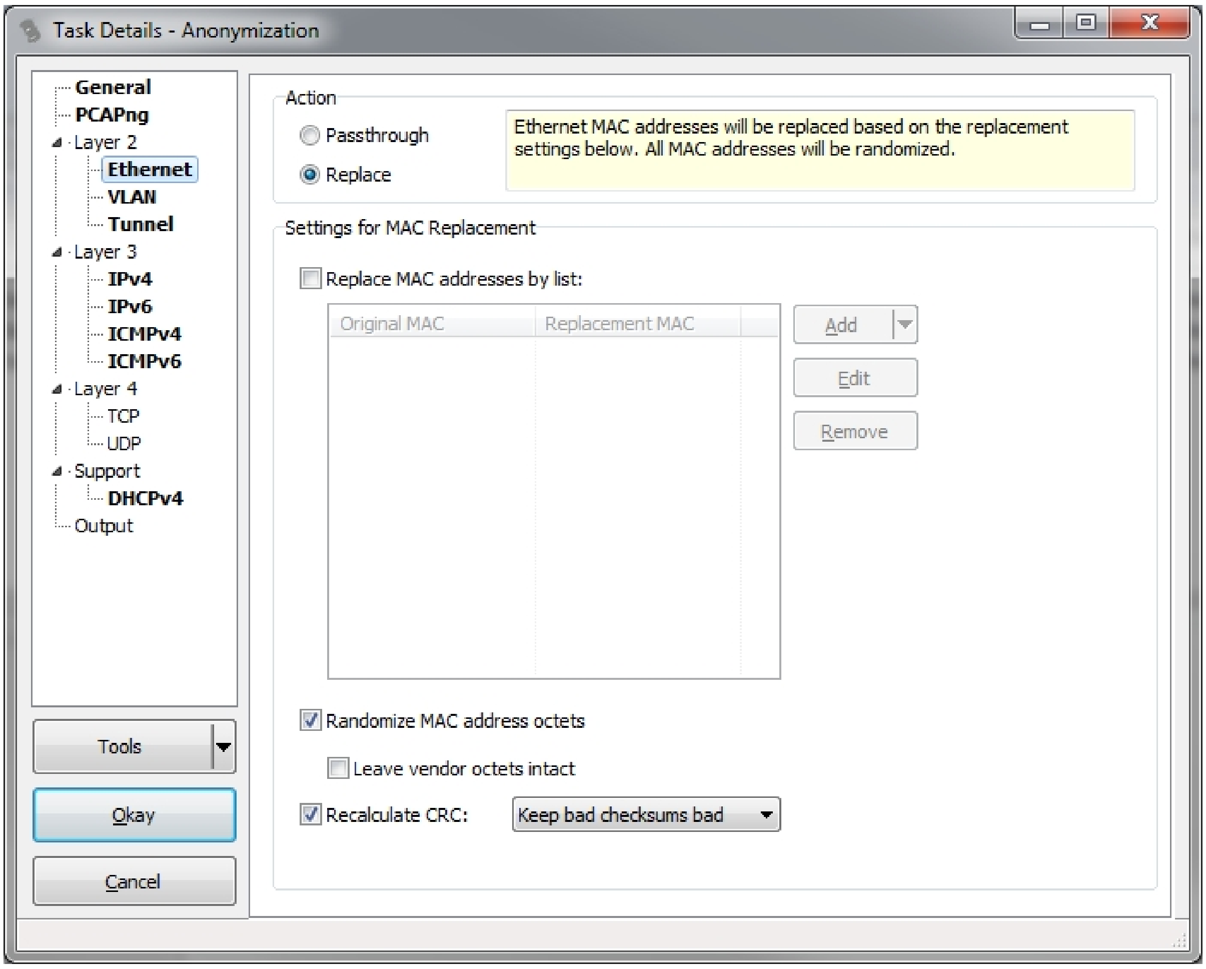
The other detail that is written to the Interface Header Block is the capture filter that was used when starting the capture. Since this field may often contain sensitive information like IP addresses that you don't want to expose, the default is to remove the filter string completely. It would be nice to have some sort of automatic replacement of any address found just like when dealing with an IP address in the IP layer, but parsing filter strings can become very complicated, leading to possibly failing to detect sensitive information. So the only options are to remove it or keep it.

Sanitize Name Resolution Header Blocks: PCAPng is capable of keeping a list of FQDNs with their associated IP addresses as they were determined when saving the capture file. This makes it easier to work with traces even when no DNS can provide the name resolution at the time of the analysis, because the file keeps the mappings in the Name Resolution Header Block. Of course, this could not only expose IP addresses you want to sanitize, but also FQDNs of critical systems that you do not want to distribute with the file. Right now, the only option is to completely remove the Name Resolution Block (or keep it, of course).

Sanitize Packet Options: There are a number of optional details that can be stored with each packet, e.g. an MD5 hash. If this setting is active, none of those optional values will be written to the sanitized file.

Ethernet

The Ethernet section offers settings that allow the sanitization of the Ethernet layer, which basically means MAC addresses. An important detail is the fact that when you chose to replace MAC addresses in this section it will also affect any other layer that contains that kind of address information. For example ARP frames contain MAC addresses simply because it works as a link layer resolution protocol. As you can see there is no ARP section in the section tree, but that doesn't mean that ARP frames will not be sanitized - they will just be sanitized according to the MAC address settings you specify for the Ethernet layer.



Action: Passthrough [✓ Replace]

Ethernet MAC addresses will be replaced based on the replacement settings below. All MAC addresses will be randomized.

Settings for MAC Replacement:

- [ ] Replace MAC addresses by list:
- [✓] Randomize MAC address octets
- [ ] Leave vendor octets intact
- [✓] Recalculate CRC: Keep bad checksums bad

Replace MAC addresses by list: When a MAC address is found in a frame, either as part of the Ethernet layer, or in an ARP frame, it will be compared to the "Original MAC" address in the list and replaced by the specified "Replacement MAC" if it was found. If you have a list of MAC addresses in a comma separated file you can import them by using the popup menu of the list. You can also export a list if you want to keep it for another task to avoid having to enter it all over again manually. When you add addresses the input dialog will expect the addresses in the format of "xx:xx:xx:xx:xx:xx" and only enable the "Add" buttons when the format is valid.

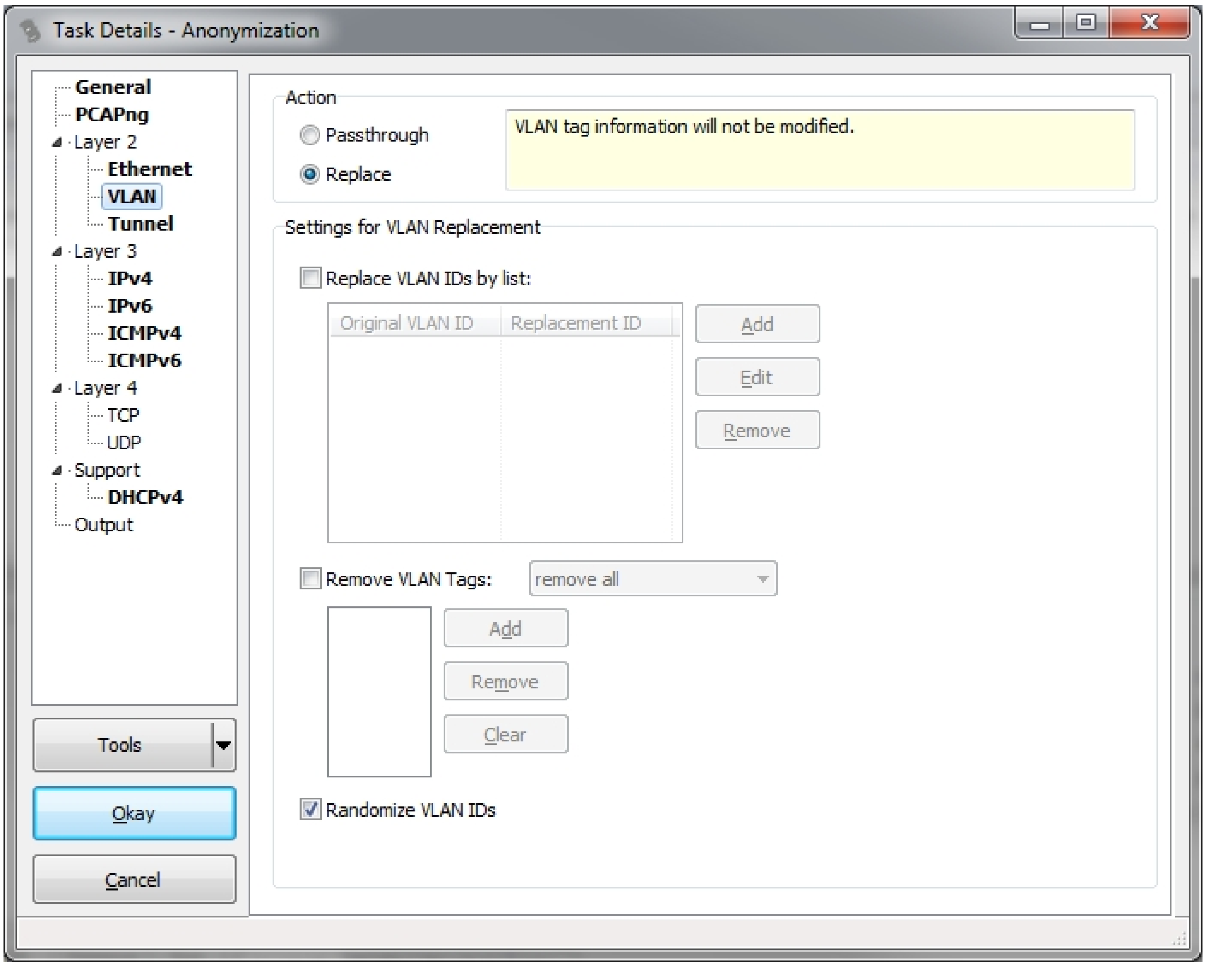
Randomize MAC address octets: When this setting is active and a MAC that had been found in a frame was not already processed by the MAC address list (either because it wasn't in it, or the setting wasn't enabled), it will be randomized. You can optionally keep the first three octets of the address, which are the vendor specific values that allows looking up who the MAC address range was assigned to. If you do not choose to keep the vendor informaton, the randomization process will always set the first octet to 0xf2, for three reasons: first of all, the "locally administered bit" is set to indicate that this MAC address is not coming from the official IEEE list. Second, there is no entry with that first byte in the Wireshark manuf file so it should not be mistaken for something that its not. And third, it will make it easier to see that it has been sanitized.

Recalculate CRC: Ethernet frames may or may not come with a Frame Check Sequence (FCS) in the trace file. Most analyzers do not or can not store the FCS, but if you have a capture which includes the FCS you can recalculate it after sanitization.

VLANs

Maybe there was a little surprise when you saw that VLANs had their own settings page, because it is usually considered to be part of the Ethernet layer. There are two reasons why VLANs got their own section:

1. TraceWrangler can read as many VLAN tags as there are in a given frame, because it will just follow the Ethertype values to determine what the next layer is. So it doesn't matter if a packet has just one VLAN layer, two (also known as "QinQ", three, or none at all.
2. The options for VLANs are too complex to fit them into the Ethernet section


Task Details - Anonymization

Action: [✓ Replace] Passthrough

VLAN tag information will not be modified.

Settings for VLAN Replacement

- [ ] Replace VLAN IDs by list:

Original VLAN ID	Replacement ID

- [ ] Remove VLAN Tags: remove all
- [✓] Randomize VLAN IDs

Replace VLAN IDs by list:This works pretty much like the Ethernet address list. If a VLAN tag is found and it contains a VLAN ID that listed in the replacement list it will be replaced by whatever value the list holds as the Replacement ID. You can import and export the list to a csv file if you want. Adding VLAN IDs to the list will verify that the ID is in the valid range between 0 and 4095 before enabling the "Add" button.

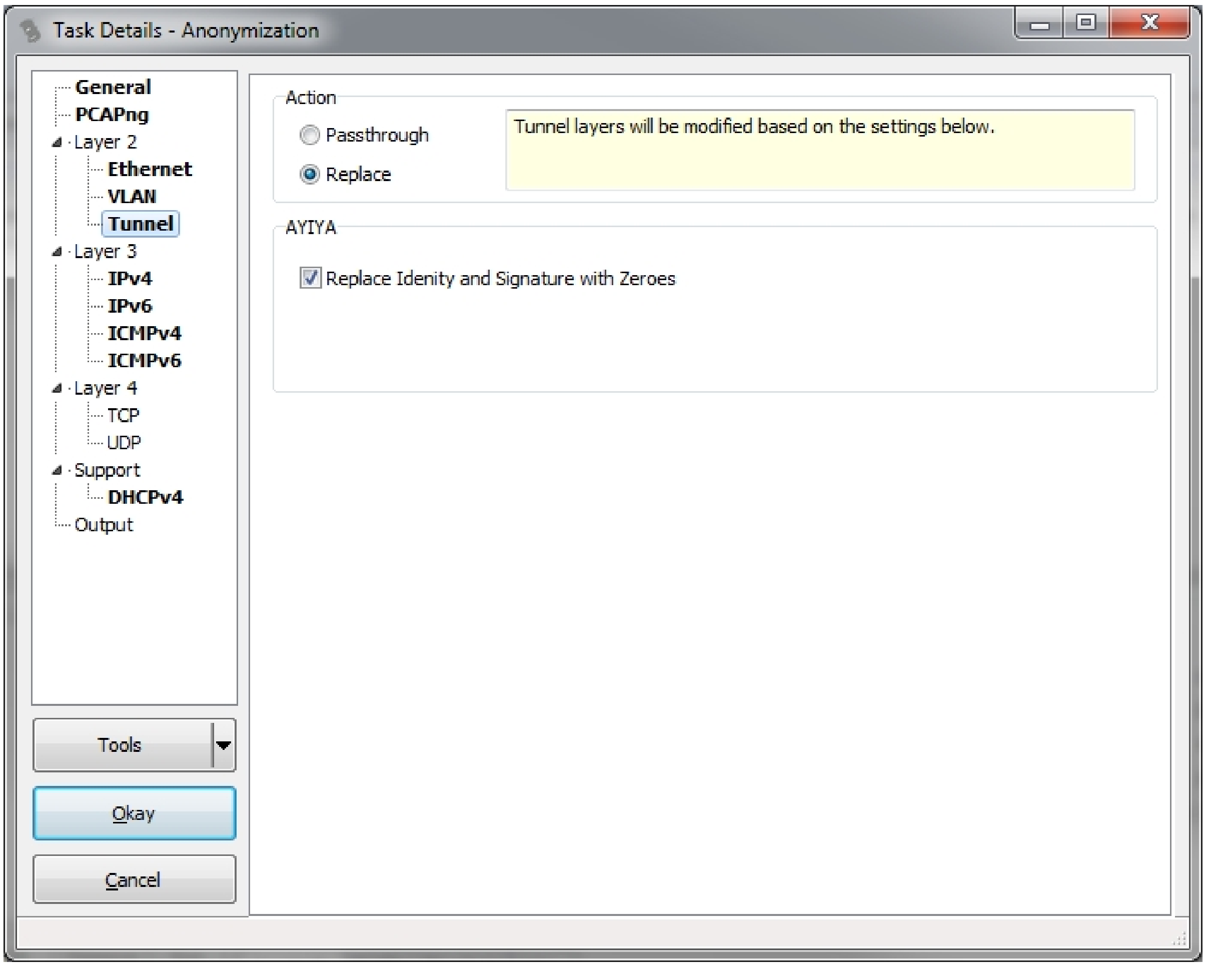
Remove VLAN tags: In case you want to completely remove some or all VLAN tags from the frames in your file(s) you can activate this setting and use the combo box to specify a list of VLAN IDs to remove or keep. Or simply remove them all. When a VLAN tag is removed, TraceWrangler will transfer the Ethertypes of the VLAN tag up to the parent layer to keep the frame intact.

Randomize VLAN IDs: If an ID has not yet been processed by the VLAN ID list (maybe simply because the list is not used), TraceWrangler will roll the dice and create a new ID.

Tunnels

This section is used to sanitize tunneling layers. At the moment, the only tunneling technique that is support is the AYIYA layer used by IPv6 over IPv4 tunnels provided by SIXXS. Their tunneling layer allows transporting IPv6 frames over UDP, and contains a identity and signature that are basically credentials for the tunnel connection.

The reason why this layer can be processed by TraceWrangler is pretty simple: I use SIXXS tunnels for IPv6 R&D as well as teaching classes. So I need that functionality myself, and that's the reason why I added a parser for this.

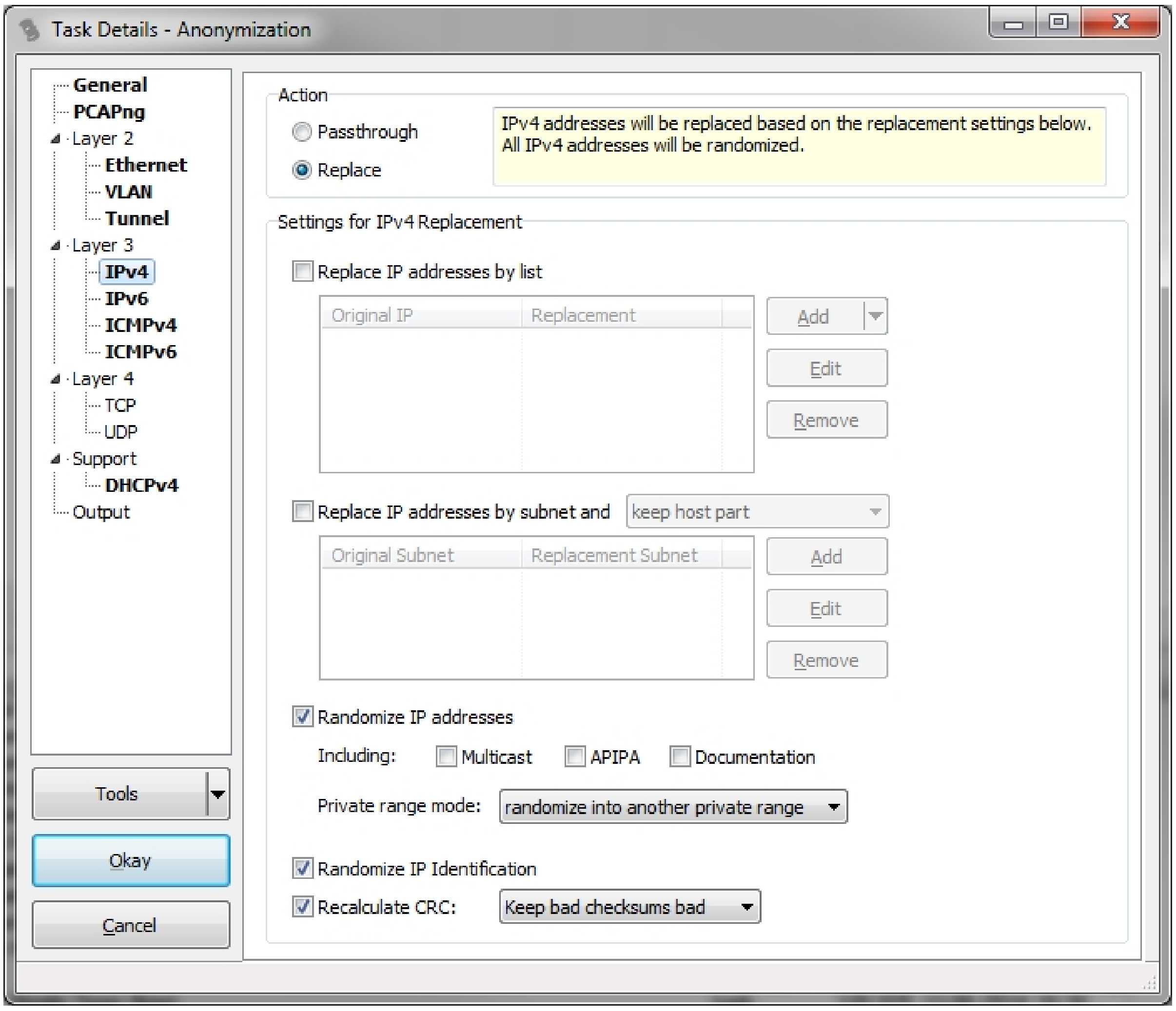

Task Details - Anonymization

Action
Passthrough [✓ Replace]
Tunnel layers will be modified based on the settings below.

AYIYA
[✓] Replace Idenity and Signature with Zeroes

IPv4

Replacing IP addresses is probably the most common thing that any anonymization tool is going to perform. TraceWrangler tries to make the process as easy to configure as possible, while being as powerful and versatile as possible at the same time. Keep in mind that any IPv4 address that is found in a protocol layer that TraceWrangler can parse and assemble will be anonymized just the same way. Of course that primarily means the IPv4 layer, but also ARP frames, DHCPv4 packets and so on. That is also the reason why there are no IPv4 settings for DHCPv4 and ICMPv4 in their respective sections.



Task Details - Anonymization

General
PCAPng
Layer 2
Ethernet
VLAN
Tunnel
Layer 3
IPv4
IPv6
ICMPv4
ICMPv6
Layer 4
TCP
UDP
Support
DHCPv4
Output

Tools
Okay
Cancel

Action
- [ ] Passthrough
- [✓] Replace

IPv4 addresses will be replaced based on the replacement settings below.
All IPv4 addresses will be randomized.

Settings for IPv4 Replacement

- [ ] Replace IP addresses by list

Original IP	Replacement

Add | Edit | Remove

- [ ] Replace IP addresses by subnet and: keep host part

Original Subnet	Replacement Subnet

Add | Edit | Remove

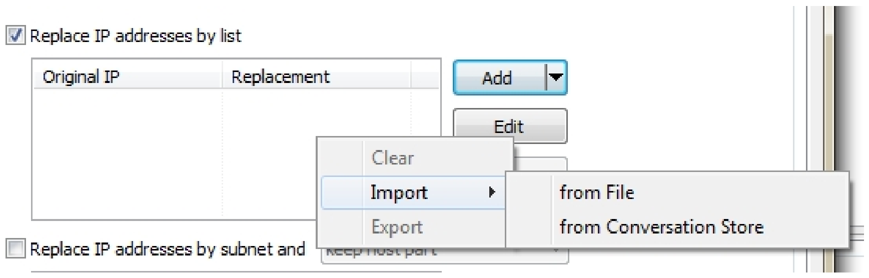
- [✓] Randomize IP addresses

Including: [ ] Multicast [ ] APIPA [ ] Documentation

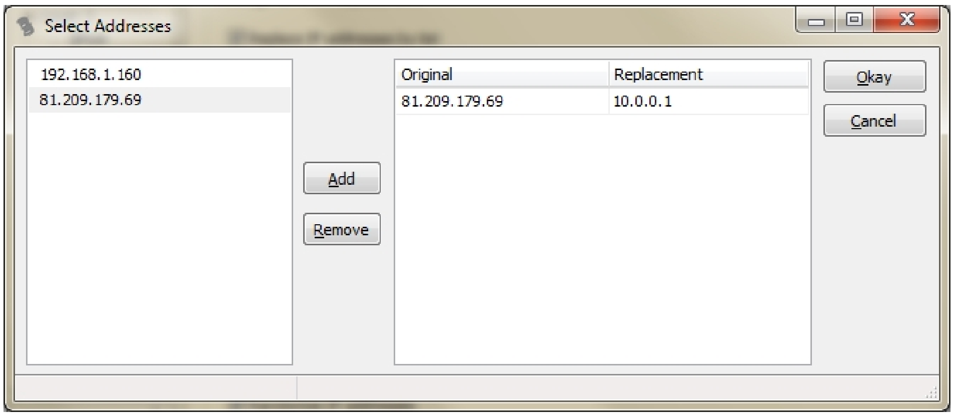
Private range mode: randomize into another private range

- [✓] Randomize IP Identification
- [✓] Recalculate CRC: Keep bad checksums bad

Replace IP addresses by list: if active, any address that is found in the trace that is listed in the IP address list on the "Original IP" column will be replaced wit the according address in the "Replacement IP" column. The "Add" button next to the list is used to add a new replacement pair. A click on the button will open a dialog that allows you to type in an original address to look for, and a second address that will replace the original address if found. There is also a pop up menu to help with adding address replacement pairs, which can be accessed by right-clicking the list:



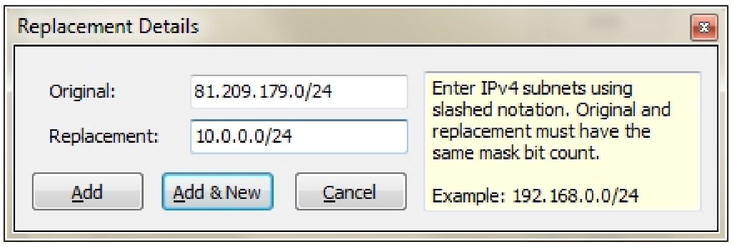
- it is possible to import and export comma separated lists from and to files, in case you want to keep and re-import a list of addresses at a later time. It is also helpful if you want to create replacement pairs in an text editor of your choice and import them.
- the Conversation Store is **available only** if the capture files have been scanned for conversation details before adding/editing the anonymization task. Otherwise the menu option will be grayed out. When selected, another dialog will open to allow selecting and editing replacement IP addresses:



192.168.1.160
81.209.179.69

Original	Replacement
81.209.179.69	10.0.0.1

Replace IP addresses by subnet: very often, the primary goal of anonymizing a trace is to hide subnet information, meaning that you do not want anyone to know to which network certain nodes belong. At the same time it can be helpful to keep the host part of each IP intact to preserve address relationships. For this, you can add subnets in CIDR notation (like "192.168.1.0/24"), but you need to enter the same mask bit count for the original network and the replacement network (otherwise the text in the field will be displayed in red).



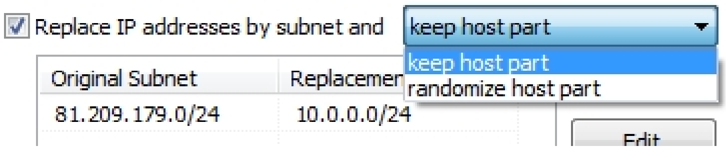
Original: 81.209.179.0/24

Replacement: 10.0.0.0/24

Enter IPv4 subnets using slashed notation. Original and replacement must have the same mask bit count.

Example: 192.168.0.0/24

If you choose to replace IP addresses by subnet you can specify what should happen to the host part of the original address. The combo box allows you to chose to keep the host part intact, or randomize it. Just like the IP address list you can use the pop up menu to import and export subnet lists to reuse them at a later time.



Replace IP addresses by subnet and: keep host part

Original Subnet	Replacement
81.209.179.0/24	10.0.0.0/24

Randomize IP addresses: if this setting is active, any address that has not been replaced by either of the lists described above will be randomized. As a sub setting you can force the randomization various special addresses, which will otherwise not be touched and written as their original value to the sanitized file. There are a couple of options to force randomization for specific address types if required:

- **Multicast Addresses**: Multicast addresses are often reserved for special purpose, which you might want to keep to allow troubleshooting the sanitized file. But even when you force the randomization of multicast addresses TraceWrangler will make sure that the result is once again a multicast address.
- **APIPA Addresses**: Maybe you've heard the name APIPA, which stands for Automatic Private IP Addressing. Even if you haven't heard that name, you've probably seen an APIPA address, which is in the range of 169.254.0.0/16. Those are usually assigned to network cards when getting an DHCP address fails. Usually they're not giving away any real network and are interesting when seen in a trace because it usually indicates some sort of trouble with DHCP. That's why they should be kept in most cases instead of randomizing them.
- **Documentation addresses**: There are three special ranges defined for documentation addresses, which means that they can be used as dummy IP addresses in documentation. The ranges are 192.0.2.0/24, 198.51.100.0/24, and 203.0.113.0/24. Since they are dummy addresses they usually do not expose any risk and should be kept intact.
- **Private IP Address ranges**: IPs in the range of 10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16 are

private ranges and are not routed over the internet, so exposing them doesn't tell a lot about the network topology. Still it may be desirable to keep the network range a secret if you are afraid that it helps a bad buy with his network reconnaissance. There are three settings for private IP ranges:

1. **Randomize into another private range**: in this case, TraceWrangler will generate a random address, but it will be in a private network range again. That way you keep the fact intact that the address belongs to a private network, without exposing exact details about the network and the IP
2. **Randomize over full address range**: TraceWrangler will randomize the IP address and it is not guaranteed that the IP is a private address afterwards.
3. **Do not randomize**: this setting will keep the original IP address intact.

Generally, randomization will **not allow** the random result to be any of the following addresses:

- 0.0.0.0
- 127.0.0.0/8
- 255.255.255.255
- the same as the original address
- the same as a replacement address for a different original address

If randomization creates an address from the listed address ranges the dice will be rerolled until it is not on the list anymore. The same will happen when a random result matches an existing replacement IP, because it would be bad if two different original addresses end up being replaced by the same replacement address. It may be very confusing to analyze a sanitized trace in that case, so TraceWrangler will verify each random result to be unique before accepting it.

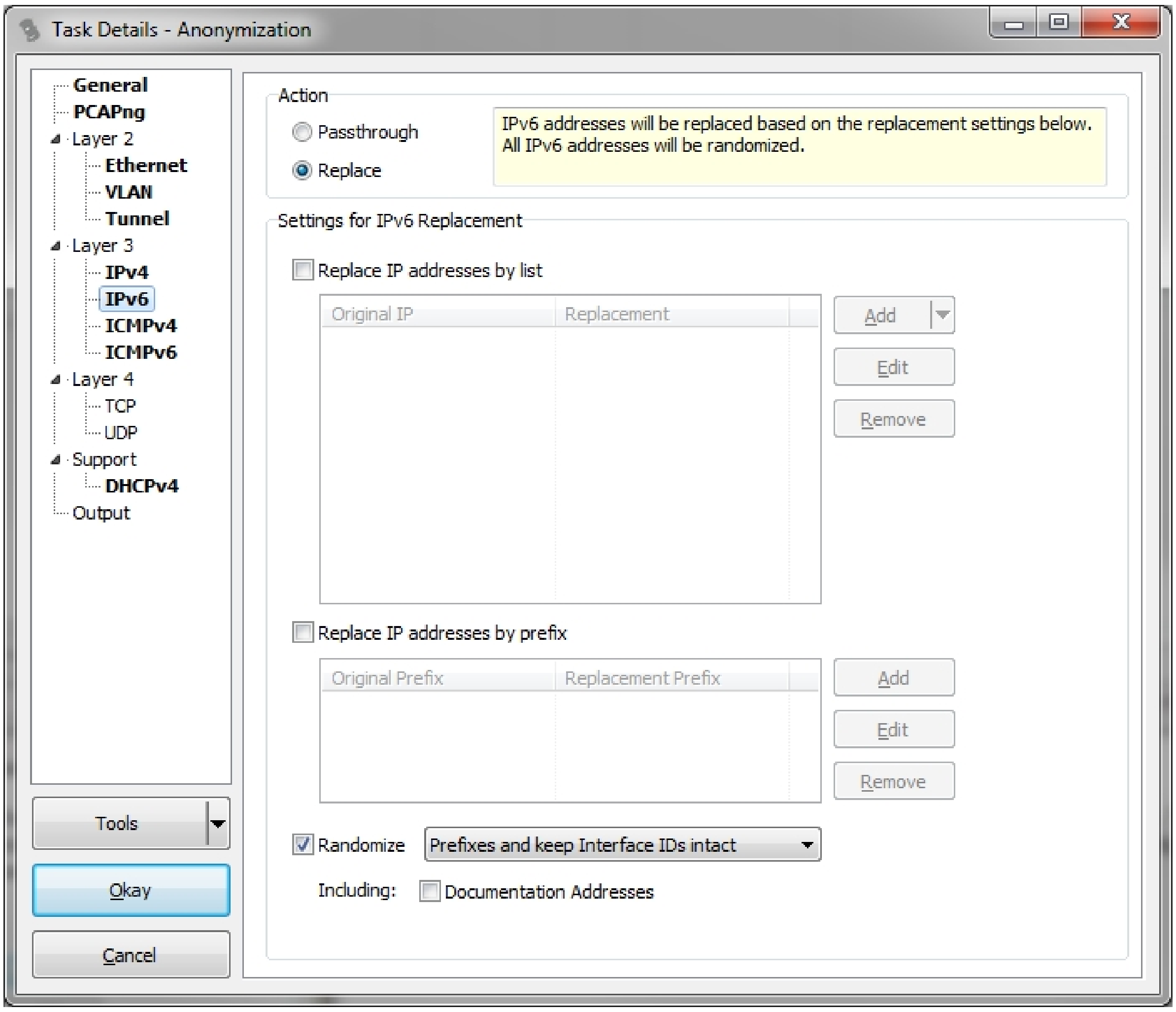
Randomize IP Identification: some people are concerned that the IP ID can expose information about system behavior, especially when it is counting up by one for each IP packet. With this setting each packet will be assigned a random new IP ID, except fragmented packets. Fragments will also get a new random IP ID, but all fragments belonging together will have their IP ID replaced by the same new random ID to keep their relationship intact.

Recalculate CRC: when the IP layer is changed, for example by replacing IP addresses, the original checksum will not be correct anymore. With this setting you can chose to recalculate the checksum. The combo box allows you to only recalculate checksums if the original checksum had been okay, which is the default. This is another setting that is trying to keep the facts about the original file intact, because it may be interesting for an analyst to know that the checksum had been bad in the original file as well.

To make sure that the checksum is incorrect TraceWrangler does not just use a random number, or the incorrect original, because both may incidentally turn out to be correct. Instead, the correct checksum will be calculated and then forcefully increment it by 1 (and wrapped, if necessary). That way the CRC is incorrect for certain, but you can also tell it was made up because it will always be just 1 higher than the correct would have been.

IPv6

IPv6 is still a pretty uncommon protocol (unfortunately), but TraceWrangler can sanitize it already, if not completely. Part of IPv6 are the so called "Extension Headers" which allow extending the way the protocol works while using a fixed size IPv6 base header. Right now, TraceWrangler can sanitize the base header as well as the Fragmentation Header, but none of the others (yet). Since the IP ID field that is part of the IPv4 header is only present in the Fragmentation Header which is only included when it is really needed, there is no IP ID sanitization setting for IPv6. It is just not necessary.

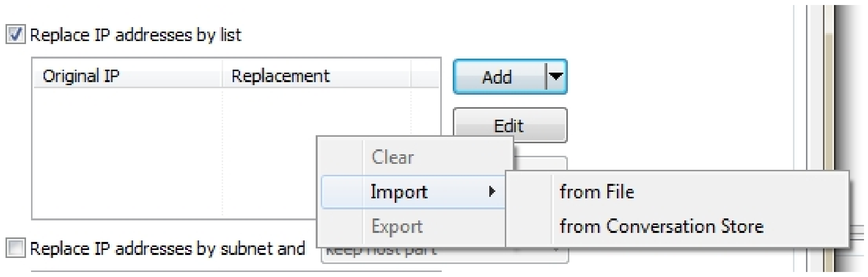


Action: [✓ Replace]

IPv6 addresses will be replaced based on the replacement settings below.
All IPv6 addresses will be randomized.

Randomize [✓]: Prefixes and keep Interface IDs intact

Replace IP addresses by list: Any IP found in the original trace that is listed in the Original IP column will be replaced by the IP in the Replacement IP column. You can use the pop up menu to import and export comma separated lists of IP addresses in case you want to store and retrieve them for later reuse. There is also a pop up menu to help with adding address replacement pairs, which can be accessed by right-clicking the list:



- it is possible to import and export comma separated lists from and to files, in case you want to keep and re-import a list of addresses at a later time. It is also helpful if you want to create replacement pairs in an text editor of your choice and import them.
- the Conversation Store is **available only** if the capture files have been scanned for conversation details before adding/editing the anonymization task. Otherwise the menu option will be grayed out. When selected, another dialog will open to allow selecting and editing replacement IP addresses.

Replace IP addresses by subnet: Any IP that is part of the subnet lists in the original columns will be put into the network that is listed in the according replacement column. When you add networks you need to use the same mask bit count for the replacement network as for the original network. Right now, there is no setting to randomize the host bits or keep the host part, because TraceWrangler will always keep the host part from the original. This may change at a later time.

Randomize IP addresses: If the sanitized IP address was not yet determined through the address or subnet lists, this setting will randomize the IP address if checked. There are three options for the randomization mode:

1. **Randomize Prefix and keep Interface IDs intact**: this setting keeps the interface ID intact and only randomizes the prefix. The new replacement prefix will be stored to the database and all further addresses with the same prefix will get the same replacement prefix. This setting should only be used if the interface IDs are random and do not expose any kind of attack vector by keeping them.
2. **Randomize both and synchronize prefixes**: in this mode both prefix and interface ID are randomized, if there is no existing replacement for the prefix or both. If a replacement for the prefix exists only the interface ID will be randomized. The prefix replacement is stored to the database, keeping all IPs of the same original network in the same replacement network automatically.
3. **Randomize both in stateless mode**: both prefix and interface ID will be randomized for each address. This means that most likely all IP addresses will end up in their own network since the prefix is not stored and synchronized. This usually only makes sense if used in combination with the prefix replacement list.

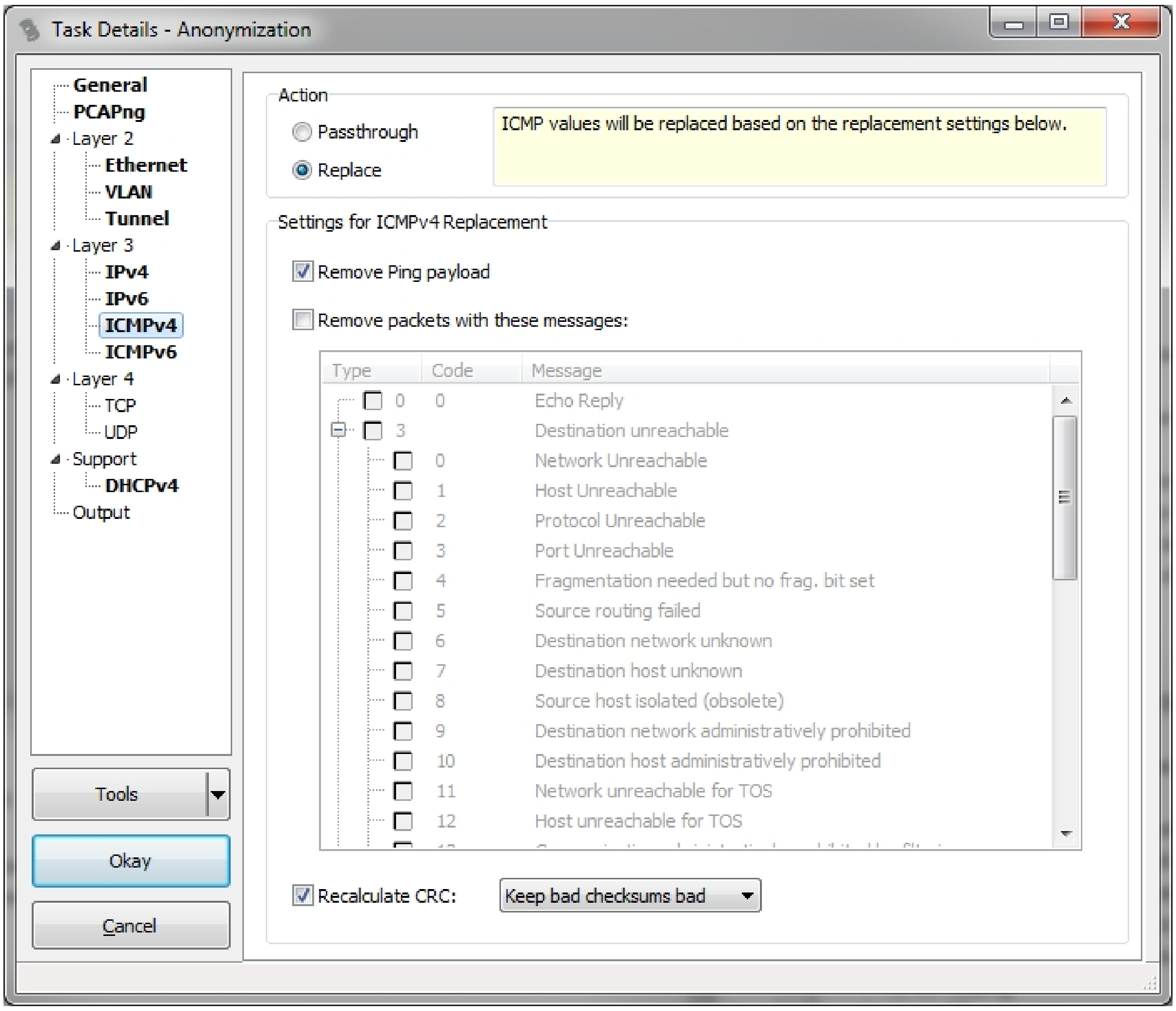
Randomization will generally do a couple of things automatically:

- It will make sure that original public addresses from the range of 2000://3 will end up with a replacement address in the same address block. This means that the random result will have a first 16 bit block with a value between 0x2000 and 0x3fff.
- Any address in the range of fc00::/8 will end up with a new random address, but in the same range. Same goes for addresses in the range of fd00://8
- Link local addresses will stay link local addresses, so they will have a prefix of fe80::/64. Only the host 64 bits will be randomized, since it would absolutely make no sense to randomize it into any other subnet.
- If the original IP address was based on a MAC address (EUI64), TraceWrangler will create a new address with the host part being EUI64 formatted as well. At the same time, a MAC replacement is created to match, unless the original base MAC was already replaced (in which case the new EUI64 address is based on that existing replacement automatically).
- Multicast addresses will not be changed by randomization. If you need to sanitize Multicast addresses use the address replacement list to specify the exact substitution address.

If you check the box for Documentation Addresses any address with a prefix of 2001:db8::/32 will stay in the same range, since it is the official range for documentation purposes and thus sanitized already by design.

ICMPv4

Since the IPv4 layer settings already take care of IP addresses that are present in the ICMPv4 layer (like for ICMP redirects), this section does not have any settings for this. Also, TraceWrangler will process quoted IPv4 packets (for Destination Unreachable etc.) just as another IPv4 layer, so the IPv4 parser will kick in again. This means that also quoted packets will be sanitized, of course.


Task Details - Anonymization

- General
- PCAPng
- Layer 2
  - Ethernet
  - VLAN
  - Tunnel
- Layer 3
  - IPv4
  - IPv6
  - ICMPv4
  - ICMPv6
- Layer 4
  - TCP
  - UDP
- Support
  - DHCPv4
- Output

Action

Passthrough [✓ Replace]

ICMP values will be replaced based on the replacement settings below.

Settings for ICMPv4 Replacement

[✓] Remove Ping payload

[ ] Remove packets with these messages:

Type	Code	Message
0	0	Echo Reply
3		Destination unreachable
	0	Network Unreachable
	1	Host Unreachable
	2	Protocol Unreachable
	3	Port Unreachable
	4	Fragmentation needed but no frag. bit set
	5	Source routing failed
	6	Destination network unknown
	7	Destination host unknown
	8	Source host isolated (obsolete)
	9	Destination network administratively prohibited
	10	Destination host administratively prohibited
	11	Network unreachable for TOS
	12	Host unreachable for TOS

[✓] Recalculate CRC: Keep bad checksums bad

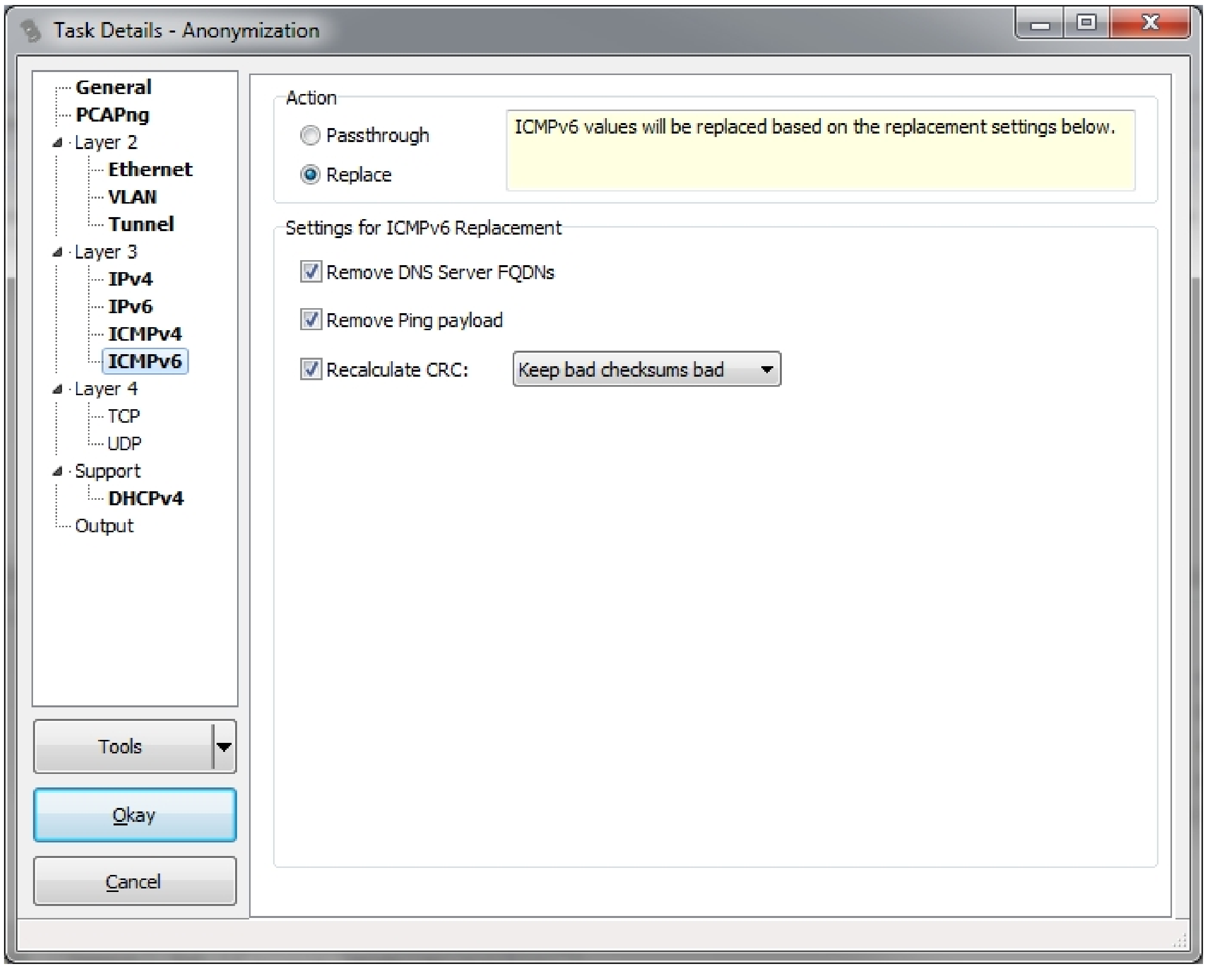
Remove Ping payload: if you want to remove the ping payload, check this option. Otherwise it is kept, even if you selected "Remove unknown payload" in the general section. Well, it isn't unknown anyway, since TraceWrangler knows what it is, right?

Remove Packets with these message: sometimes it may be a good idea to remove certain ICMP messages from the capture file completely, e.g. those that indicate that a firewall or ACL has blocked access to a resource.

Recalculate CRC: this works just like the IPv4 setting. Since sanitization may lead to a modified ICMP layer, the CRC needs to be recalculated. But in case you want to keep bad checksums bad, TraceWrangler do it the same way as for IPv4.

ICMPv6

Since the IPv6 layer settings already take care of IP addresses that are present in the ICMPv6 layer (like for ICMP redirects, and tons of IMCPv6 TLV options), this section does not have any settings for this. Also, TraceWrangler will process quoted IPv6 packets (for Destination Unreachable etc.) just as another IPv6 layer, so the IPv6 parser will kick in again. This means that also quoted packets will be sanitized, of course.



Action

Passthrough [✓ Replace]

ICMPv6 values will be replaced based on the replacement settings below.

Settings for ICMPv6 Replacement

- ✓ Remove DNS Server FQDNs
- ✓ Remove Ping payload
- ✓ Recalculate CRC: Keep bad checksums bad

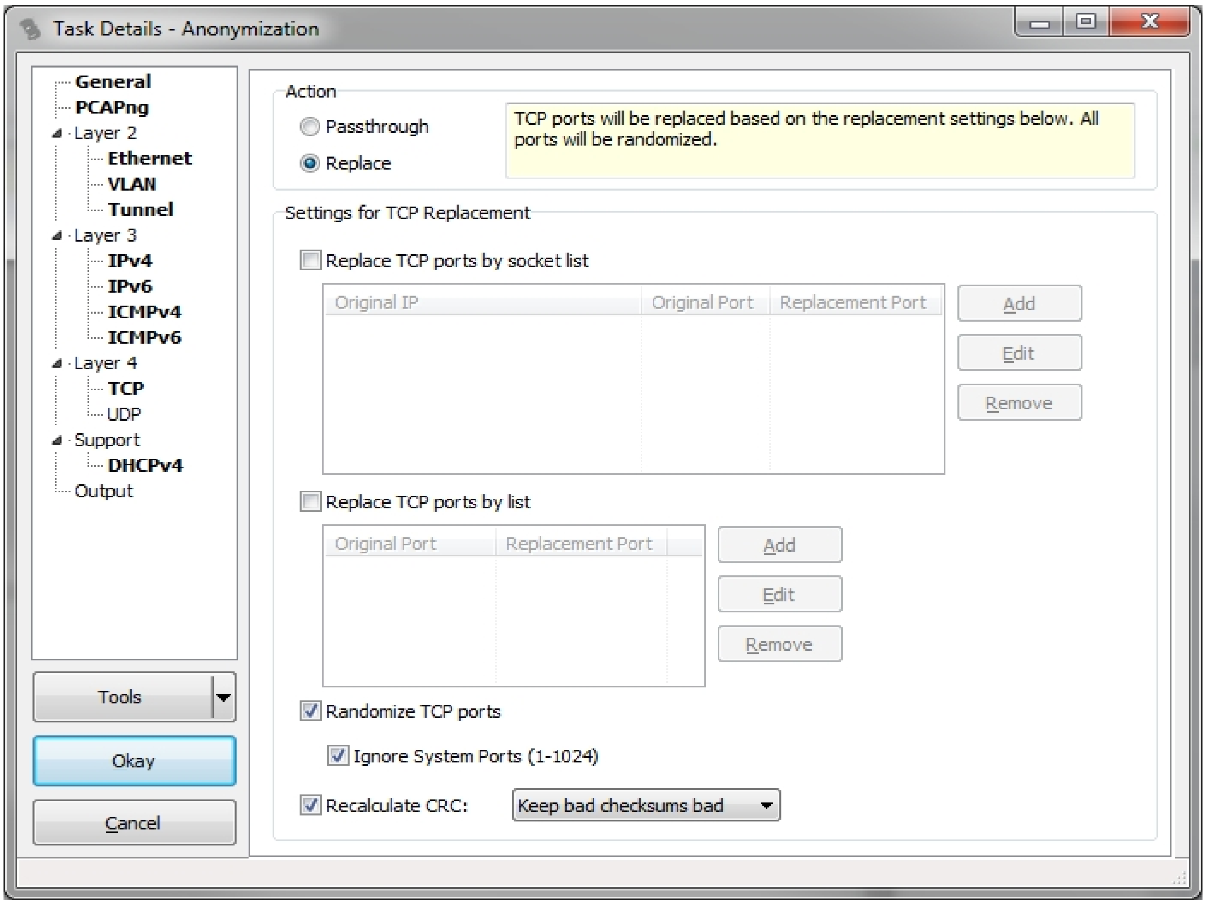
Remove DNS Server FQDNs: some options may carry FQDNs, e.g for DNS domains in more advanced Router Advertisements. With this setting, they will be sanitized by replacing all characters with an "x". E.g. a domain like "test.local" would end up as "xxxx.xxxxx".

Remove Ping payload: if you want to remove the ping payload, check this option. Otherwise it is kept, even if you selected "Remove unknown payload" in the general section. Well, it isn't unknown anyway, since TraceWrangler knows what it is, right?

Recalculate CRC: this works just like the IPv4 setting. Since sanitization may lead to a modified ICMP layer, the CRC needs to be recalculated. But in case you want to keep bad checksums bad, TraceWrangler do it the same way as for IPv4.

TCP

Right now, the only thing that TraceWrangler can sanitize is the TCP port number. Maybe there will be more options in the future if a need for additional sanitization comes up.



Task Details - Anonymization

- General
- PCAPng
- Layer 2
  - Ethernet
  - VLAN
  - Tunnel
- Layer 3
  - IPv4
  - IPv6
  - ICMPv4
  - ICMPv6
- Layer 4
  - TCP
  - UDP
- Support
  - DHCPv4
- Output

Action

( ) Passthrough
(●) Replace

TCP ports will be replaced based on the replacement settings below. All ports will be randomized.

Settings for TCP Replacement

[ ] Replace TCP ports by socket list

Original IP	Original Port	Replacement Port

[ ] Replace TCP ports by list

Original Port	Replacement Port

[✓] Randomize TCP ports

[✓] Ignore System Ports (1-1024)

[✓] Recalculate CRC: Keep bad checksums bad

Replace TCP ports by socket list: If a socket is found in a packet that matches the value in the original port column it will be replaced by the value in the replacement port column. This allows very specific port replacements, e.g. if you only want to replace ports for certain IP and port combinations. You can use the popup menu to import and export comma separated port lists to store and reuse ports at a later time.

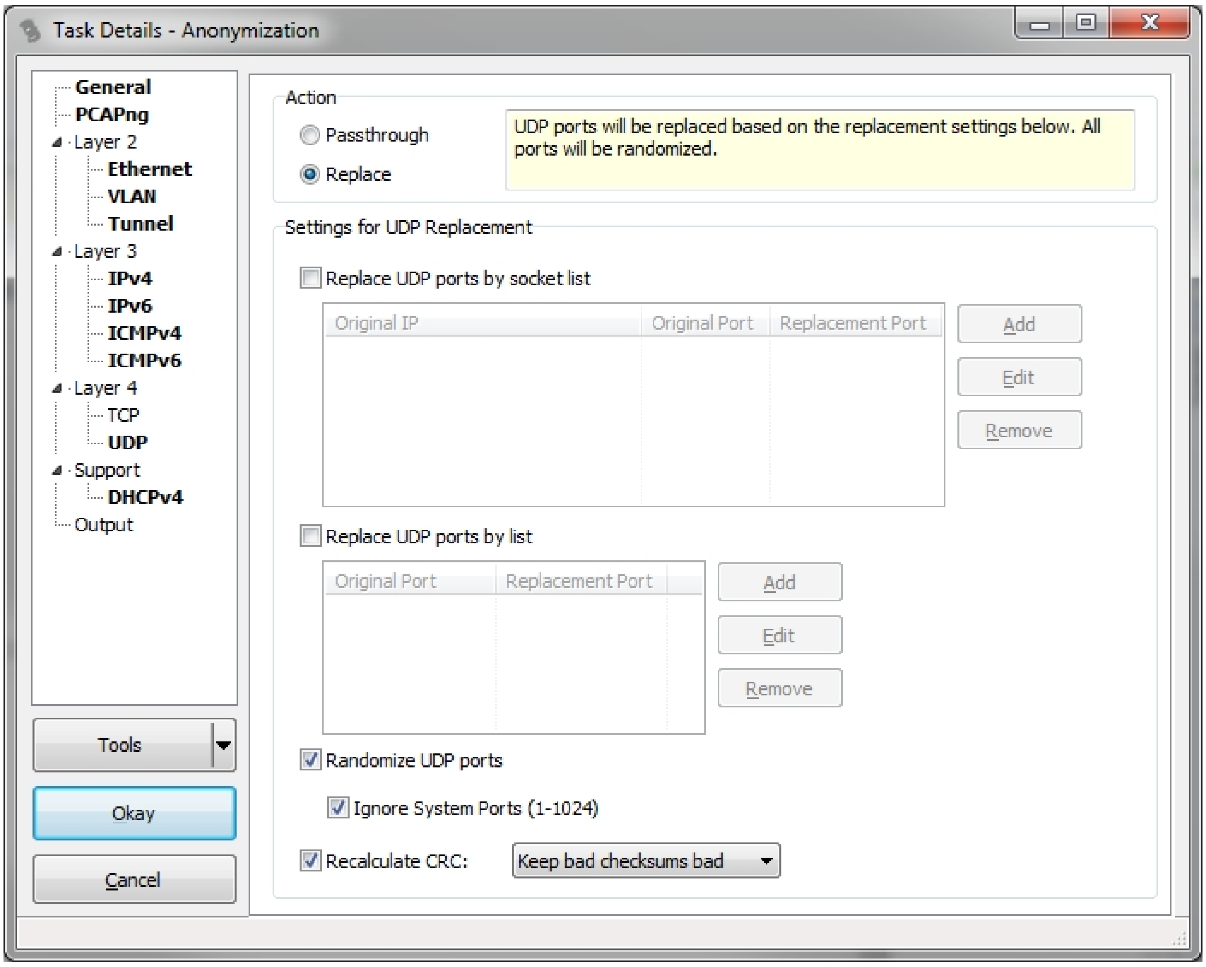
Replace TCP ports by list: If a port is found in a packet that matches the value in the original column it will be replaced by the value in the replacement port column. You can use the popup menu to import and export comma separated port lists to store and reuse ports at a later time.

Randomize TCP ports: If a port wasn't found in the list and this setting is active, a new random port will be assigned. You can use the checkbox for Ignore System ports (1-1024) to automatically keep any port unchanged that is in the range of system ports.

Recalculate CRC: this works just like the IPv4 setting. Since sanitization may lead to a modified TCP layer, the CRC needs to be recalculated. And in case you want to keep bad checksums bad, TraceWrangler do it the same way as for IPv4.

UDP

Right now, the only thing that TraceWrangler can sanitize is the UDP port number. Maybe there will be more options in the future if a need for additional sanitization comes up.



Action

Passthrough

[✓ Replace]

UDP ports will be replaced based on the replacement settings below. All ports will be randomized.

Settings for UDP Replacement

☐ Replace UDP ports by socket list

Original IP	Original Port	Replacement Port

☐ Replace UDP ports by list

Original Port	Replacement Port

☑ Randomize UDP ports

☑ Ignore System Ports (1-1024)

☑ Recalculate CRC: Keep bad checksums bad

Replace UDP ports by socket list: If a socket is found in a packet that matches the value in the original port column it will be replaced by the value in the replacement port column. This allows very specific port replacements, e.g. if you only want to replace ports for certain IP and port combinations. You can use the popup menu to import and export comma separated port lists to store and reuse ports at a later time.

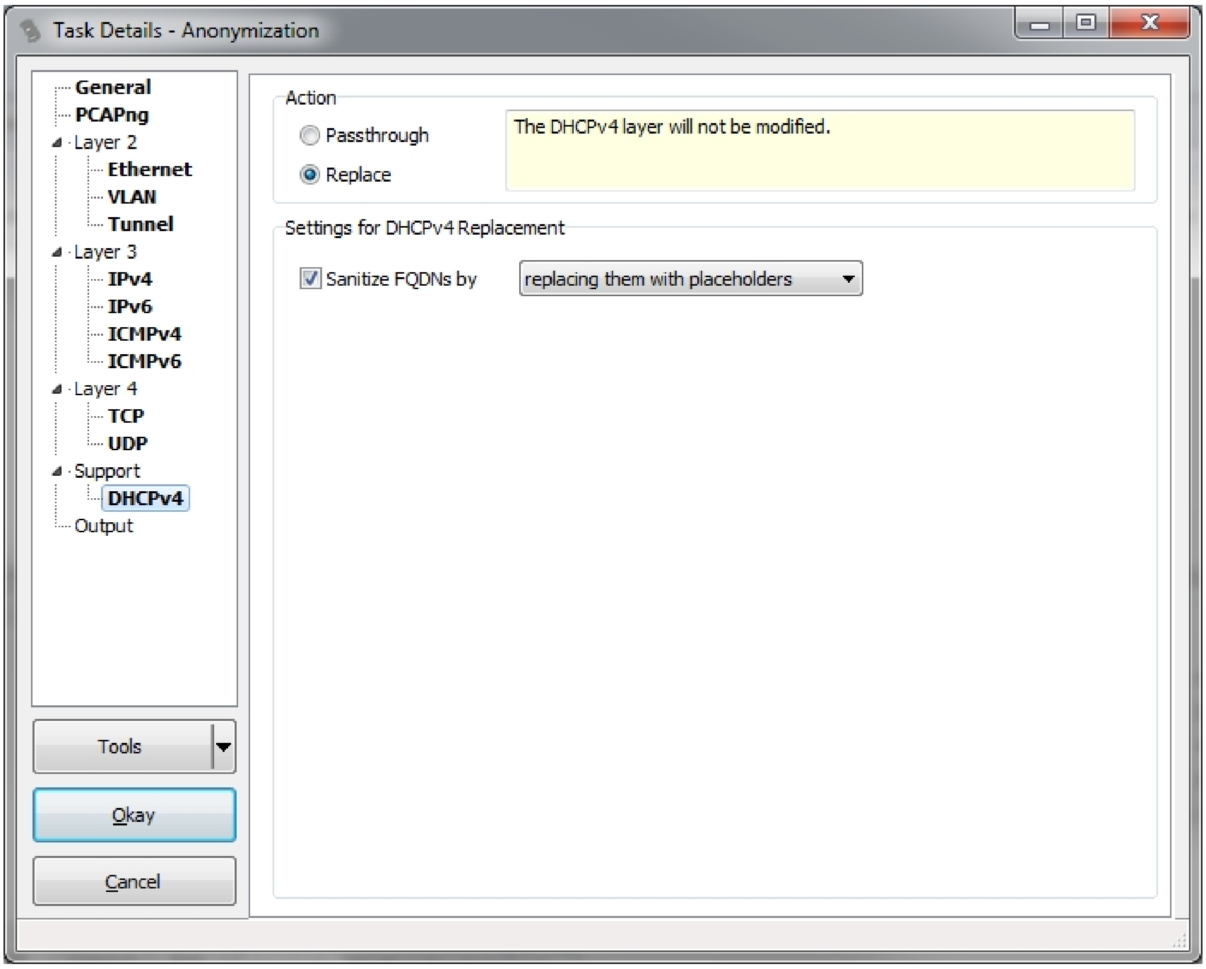
Replace UDP ports by list: If a port is found in a packet that matches the value in the original column it will be replaced by the value in the Replacement column. You can use the popup menu to import and export comma separated port lists to store and reuse ports at a later time.

Randomize UDP ports: If a port wasn't found in the list and this setting is active, a new random port will be assigned. You can use the checkbox for Ignore System ports (1-1024) to automatically keep any port unchanged that is in the range of system ports.

Recalculate CRC: this works just like the IPv4 setting. Since sanitization may lead to a modified UDP layer, the CRC needs to be recalculated. And in case you want to keep bad checksums bad, TraceWrangler do it the same way as for IPv4, except that for UDP, a checksum of zero will be kept at zero.

DHCPv4

DHCPv4 is the only upper layer protocol that TraceWrangler can sanitize at the moment. IP and MAC addresses will be replaced based on the according settings set in their respective sections. Also, since there are a lot of DHCPv4 options and values, TraceWrangler only parses the most commonly used ones at the moment. This means that any option that it does not understand will be missing in the sanitized frame.



Action

Passthrough [✓ Replace]

The DHCPv4 layer will not be modified.

Settings for DHCPv4 Replacement

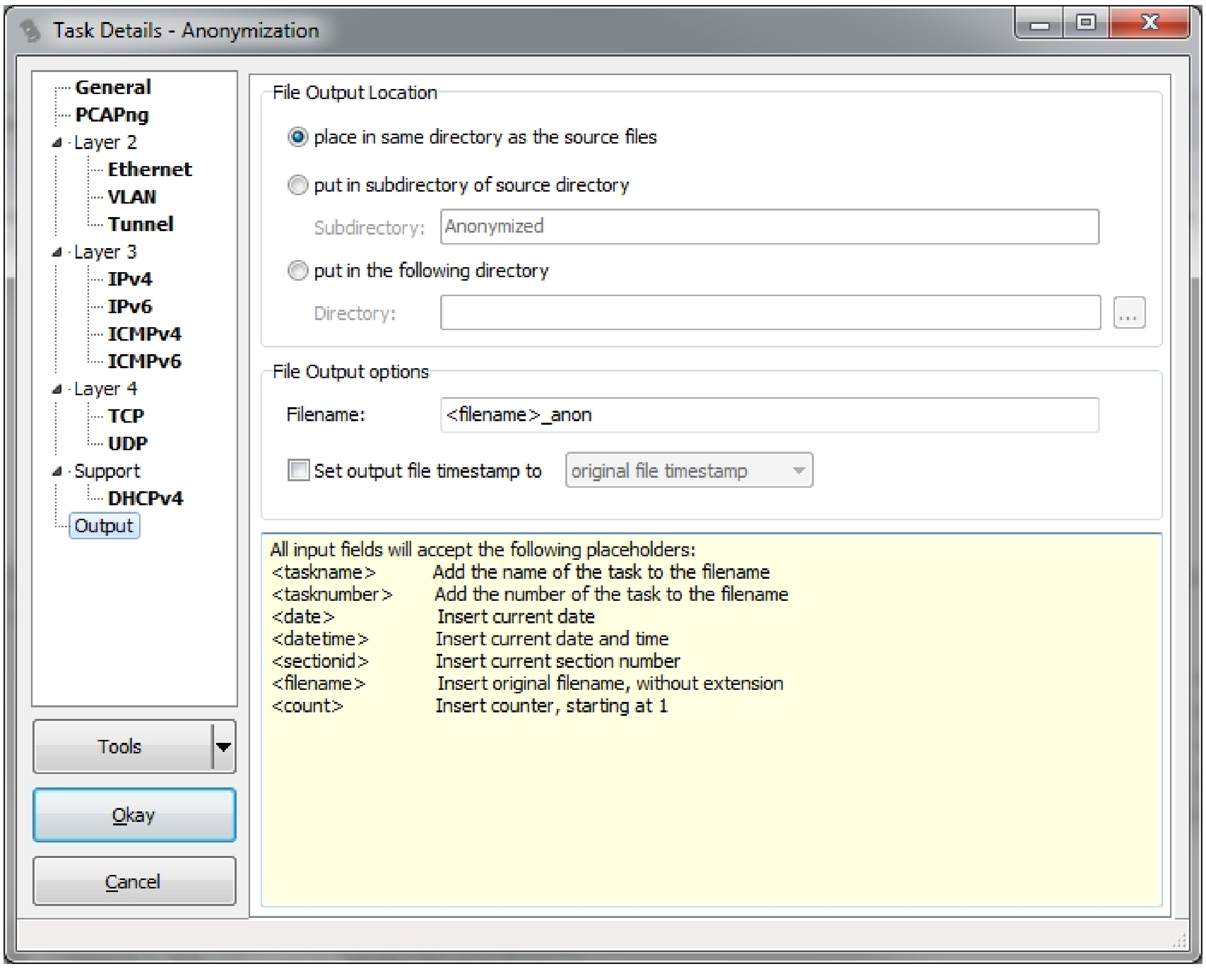
[✓] Sanitize FQDNs by: replacing them with placeholders

Sanitize FQDNs: right now, the only option to handle FQDNs is by replacing all characters with "x", as in the ICMPv6 section. This is necessary because the FQDN handling code does not yet exist for assembling frames. This will be part of the DNS assembler work that still needs to be done.

General

Output Settings

The Output settings are used to determine what the filename for the sanitized file will be. You can either put the file into the same directory as the original file, which is the default. Or you can put the sanitized files into a subdirecty or into a directory specified by absolute path. To make things more comfortable, TraceWrangler provides a couple of placeholders that can be put into the name of the directory or filename.



File Output Location

(●) place in same directory as the source files

( ) put in subdirectory of source directory

Subdirectory: Anonymized

( ) put in the following directory

Directory:

File Output options

Filename: <filename>_anon

[ ] Set output file timestamp to original file timestamp

All input fields will accept the following placeholders:

<taskname>	Add the name of the task to the filename
<tasknumber>	Add the number of the task to the filename
<date>	Insert current date
<datetime>	Insert current date and time
<sectionid>	Insert current section number
<filename>	Insert original filename, without extension
<count>	Insert counter, starting at 1

Extraction

Introduction

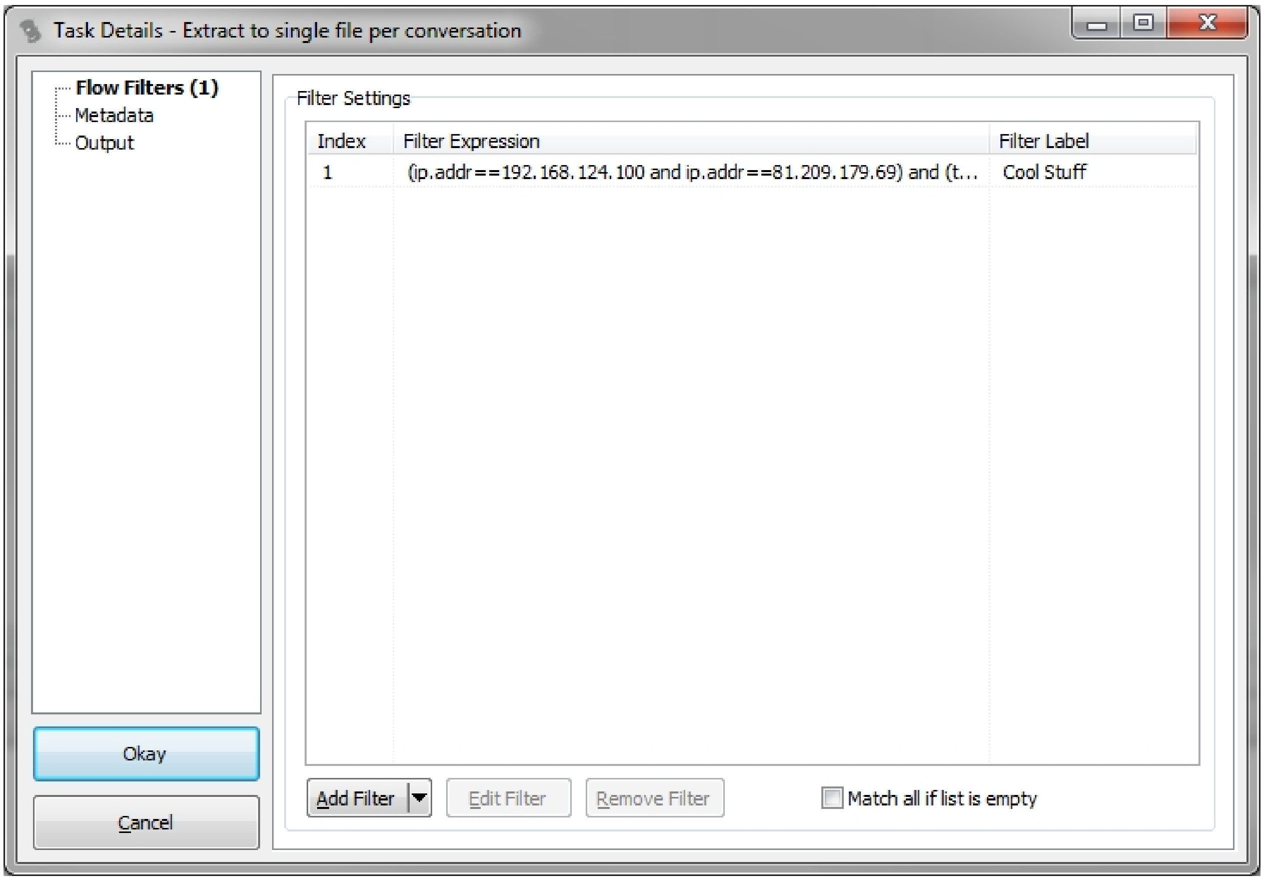
Extraction tasks are used to extract network communication flows into multiple files, usually one per flow. The basic principle is that frame filters are used to determine which frames are to be extracted, and then saving them to files. Right now there are two ways of doing flow extractions:

1. using the Conversation Summary, selecting rows for export and using the pop up menu to start the extraction process.
2. via an Extraction task, which allows to specify in greater detail how the extraction process should do its job

On a side note, double clicking on a row in the Conversation Summary dialog also extracts the according frames to a temporary file and opens the file in Wireshark (or any other tool that is configured to open PCAPng files)

Flow Filters

The flow filters page lists the frame filters that are going to be used to determine which frames to extract:



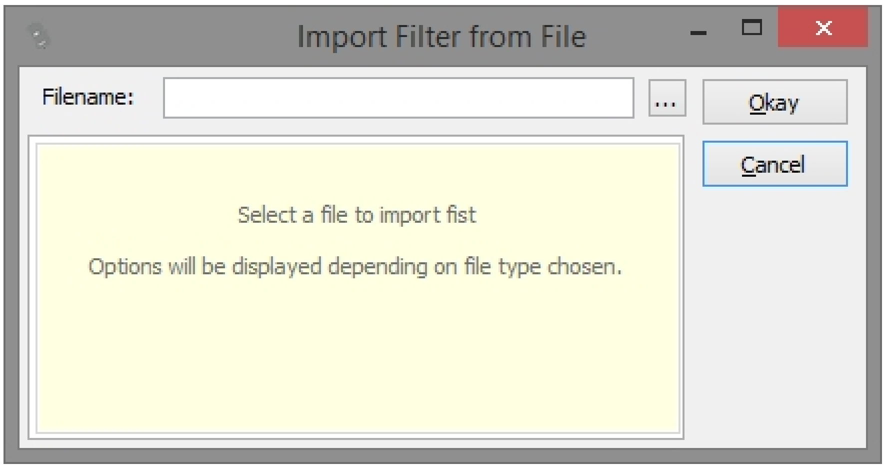
Index	Filter Expression	Filter Label
1	(ip.addr==192.168.124.100 and ip.addr==81.209.179.69) and (t...	Cool Stuff

Filter Expression: The filters are displayed in Wireshark display filter syntax, but this is only to make them easy to read for experienced Wireshark/tshark users. TraceWrangler itself only has a very limited filter engine and uses a special filter edit dialog to manually add and modify filters.

Filter Label: the filter label is an arbitrary text that can be added to a filter. The minor use case is to be able to add a small note to a filter, but the much more interesting thing is that the filter label can be used as a place holder in the output filename. This allows creating files or folders containing the filter label as part of the path and/or filename.

Match all if list is empty: usually unchecked, this will change the default behavior from not exporting anything that doesn't match a filter to exporting everything if there is no filter. Unless you really want to export everything you should not check this box.

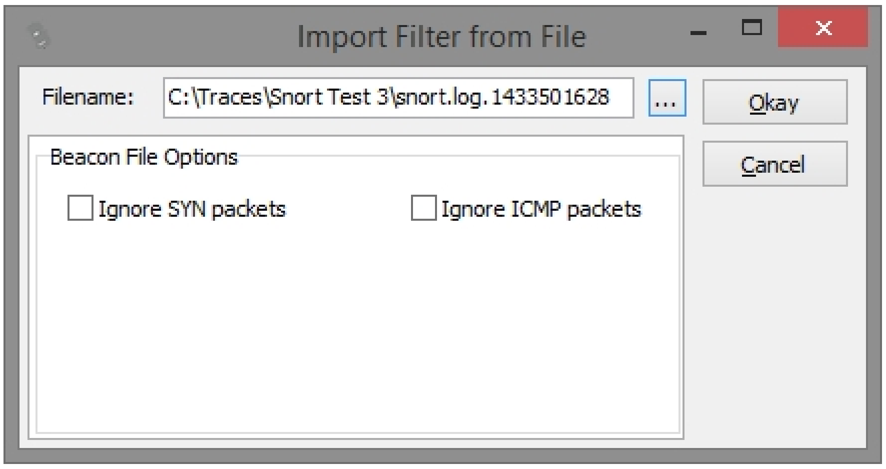
Add Filter: the "Add Filter" button is used to either open the filter editor for manually editing and adding a filter. It can also be used to open a popup menu which allows looking up conversations and create filters for them. Additionally, a couple of special file types can be imported to create filters from. If you select to import filters from a file, a dialog will be shown to allow you to select a file:



Select a file to import fist

Options will be displayed depending on file type chosen.

Enter a file name or select it via the "..." button. There are two types of files you can import filters from: a "beacon" file, and a "Snort alert" file. A beacon file is basically just a capture file containing network packets, so the usual four file formats apply: PCAPng, PCAP, ENC and CAP. Each packet or frame in such a file will be analyzed and a flow filter created from it. If you select a capture file you'll see the following options:



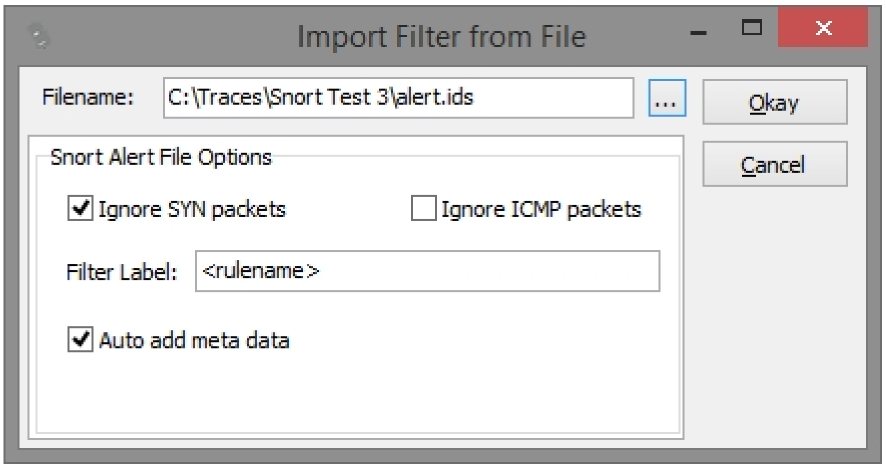
Filename: C:\Traces\Snort Test 3\snort.log.1433501628

Beacon File Options

- [ ] Ignore SYN packets
- [ ] Ignore ICMP packets

In case of Snort filter matches you may not be interested in hits on SYN packets (e.g. "Someone tried to access port 22 on IP 10.0.0.1", and you know it's firewalled anyway). Same goes for ICMP, because it may just not be of any interest to you.

If you select a snort alert file, you'll see these options:



Filename: C:\Traces\Snort Test 3\alert.ids

Snort Alert File Options

- [✓] Ignore SYN packets
- [ ] Ignore ICMP packets

Filter Label: <rulename>

- [✓] Auto add meta data

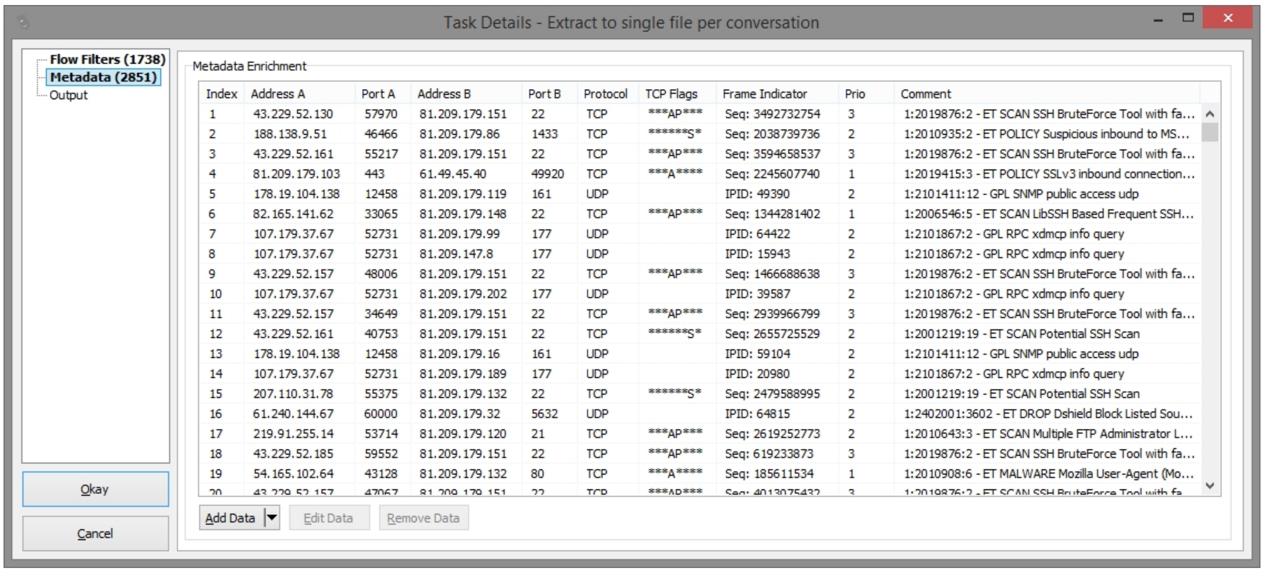
Again, you can choose to ignore SYN and ICMP packets. SYN packets are ignored by default here, because Snort alerts on those are just too annoying in most cases. Uncheck the option if you still want to extract those. The more interesting options are the filter label and the ability to automatically add meta data records.

Filter Label: If you choose to set a filter label, each filter generated from the alert file will have a label that contains the Snort rule name, classification or SID. Simply enter "<rulename>", "<classification>" or "<sid>" as placeholders, or any combination. Filter labels can then be used to generate an output file name for each extracted flow, e.g. if you want to use the Snort rule name as part of each filename, add "<rulesname>" to the filter label string.

Auto add meta data: if this option is checked, each Snort alert will be added to the meta data list. This list can be used to write comments into extracted packets, and the most common case is to use the meta data enhancement feature to add a comment to each frame where a Snort signature actually matched. That way you know which packet it really was that got hit.

Metadata Enhancement

Meta data enhancement is a feature that can be used to add comments to specific frames when they are extracted to a new file. The main reason for this is to be able to mark frames that have been hit by a IDS signature, e.g. Snort (actually, the Snort alert.ids file is the only format that can be imported for meta data enhancement at this time):



Index	Address A	Port A	Address B	Port B	Protocol	TCP Flags	Frame Indicator	Prio	Comment
1	43.229.52.130	57970	81.209.179.151	22	TCP	***AP***	Seq: 3492732754	3	1:2019876:2 - ET SCAN SSH BruteForce Tool with fa...
2	188.138.9.51	46466	81.209.179.86	1433	TCP	******S*	Seq: 2038739736	2	1:2010935:2 - ET POLICY Suspicious inbound to MS...
3	43.229.52.161	55217	81.209.179.151	22	TCP	***AP***	Seq: 3594658537	3	1:2019876:2 - ET SCAN SSH BruteForce Tool with fa...
4	81.209.179.103	443	61.49.45.40	49920	TCP	***A****	Seq: 2245607740	1	1:2019415:3 - ET POLICY SSLv3 inbound connection...
5	178.19.104.138	12458	81.209.179.119	161	UDP		IPID: 49390	2	1:2101411:12 - GPL SNMP public access udp
6	82.165.141.62	33065	81.209.179.148	22	TCP	***AP***	Seq: 1344281402	1	1:2006546:5 - ET SCAN LibSSH Based Frequent SSH...
7	107.179.37.67	52731	81.209.179.99	177	UDP		IPID: 64422	2	1:2101867:2 - GPL RPC xdmcp info query
8	107.179.37.67	52731	81.209.147.8	177	UDP		IPID: 15943	2	1:2101867:2 - GPL RPC xdmcp info query
9	43.229.52.157	48006	81.209.179.151	22	TCP	***AP***	Seq: 1466688638	3	1:2019876:2 - ET SCAN SSH BruteForce Tool with fa...
10	107.179.37.67	52731	81.209.179.202	177	UDP		IPID: 39587	2	1:2101867:2 - GPL RPC xdmcp info query
11	43.229.52.157	34649	81.209.179.151	22	TCP	***AP***	Seq: 2939966799	3	1:2019876:2 - ET SCAN SSH BruteForce Tool with fa...
12	43.229.52.161	40753	81.209.179.151	22	TCP	******S*	Seq: 2655725529	2	1:2001219:19 - ET SCAN Potential SSH Scan
13	178.19.104.138	12458	81.209.179.16	161	UDP		IPID: 59104	2	1:2101411:12 - GPL SNMP public access udp
14	107.179.37.67	52731	81.209.179.189	177	UDP		IPID: 20980	2	1:2101867:2 - GPL RPC xdmcp info query
15	207.110.31.78	55375	81.209.179.132	22	TCP	******S*	Seq: 2479588995	2	1:2001219:19 - ET SCAN Potential SSH Scan
16	61.240.144.67	60000	81.209.179.32	5632	UDP		IPID: 64815	2	1:2402001:3602 - ET DROP Dshield Block Listed Sou...
17	219.91.255.14	53714	81.209.179.120	21	TCP	***AP***	Seq: 2619252773	2	1:2010643:3 - ET SCAN Multiple FTP Administrator L...
18	43.229.52.185	59552	81.209.179.151	22	TCP	***AP***	Seq: 619233873	3	1:2019876:2 - ET SCAN SSH BruteForce Tool with fa...
19	54.165.102.64	43128	81.209.179.132	80	TCP	***A****	Seq: 185611534	1	1:2010908:6 - ET MALWARE Mozilla User-Agent (Mo...
20	43.229.52.157	47067	81.209.179.151	22	TCP	***AP***	Seq: 4013075432	3	1:2019876:2 - ET SCAN SSH BruteForce Tool with fa

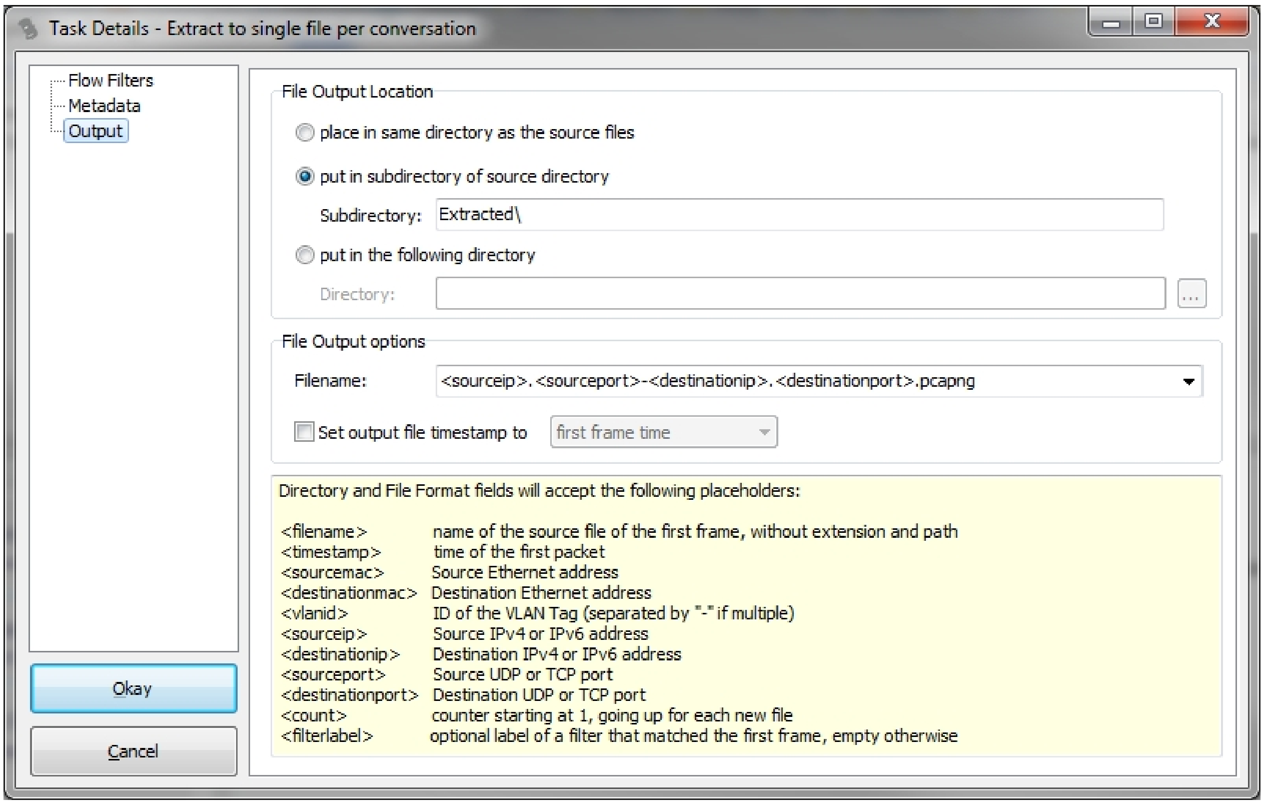
Frame Indicator: the frame indicator column contains the criteria by which the frame is specified that should be enhanced with the comment. For TCP this should be the sequence number, for everything else it is the IP ID (this is not going to work for IPv6, since IPv6 does not have the IP identification value anymore except in fragmentation headers).

Prio: this is the priority value as specified by the Snort rule

Comment: when importing Snort alerts from a file, this column contains the SID and the rule name of the signature that was detected.

Output settings

This dialog configures the way the exported frames are written to file. The default is designed to be compatible with the tcpflow file naming scheme, except that it is bidirectional:



File Output Location

- ( ) place in same directory as the source files
- (●) put in subdirectory of source directory

Subdirectory: Extracted\

- ( ) put in the following directory

Directory:

File Output options

Filename: <sourceip>.<sourceport>-<destinationip>.<destinationport>.pcapng

☐ Set output file timestamp to: first frame time

Directory and File Format fields will accept the following placeholders:

<filename>	name of the source file of the first frame, without extension and path
<timestamp>	time of the first packet
<sourcemac>	Source Ethernet address
<destinationmac>	Destination Ethernet address
<vlanid>	ID of the VLAN Tag (separated by "-" if multiple)
<sourceip>	Source IPv4 or IPv6 address
<destinationip>	Destination IPv4 or IPv6 address
<sourceport>	Source UDP or TCP port
<destinationport>	Destination UDP or TCP port
<count>	counter starting at 1, going up for each new file
<filterlabel>	optional label of a filter that matched the first frame, empty otherwise

The placeholders can be used in any of the input fields. Special fields are the **count** field that is increased for each new file, and the **filterlabel** field that takes the filter label specified in the flow filters table. The main use for this is to be able to put the snort signature name into the file name or path.

Merging

No content yet.

Introduction

After selecting to add a new merge task the according settings dialog will automatically open up. By default a couple of settings are already configured, for example the mapping of the interfaces.

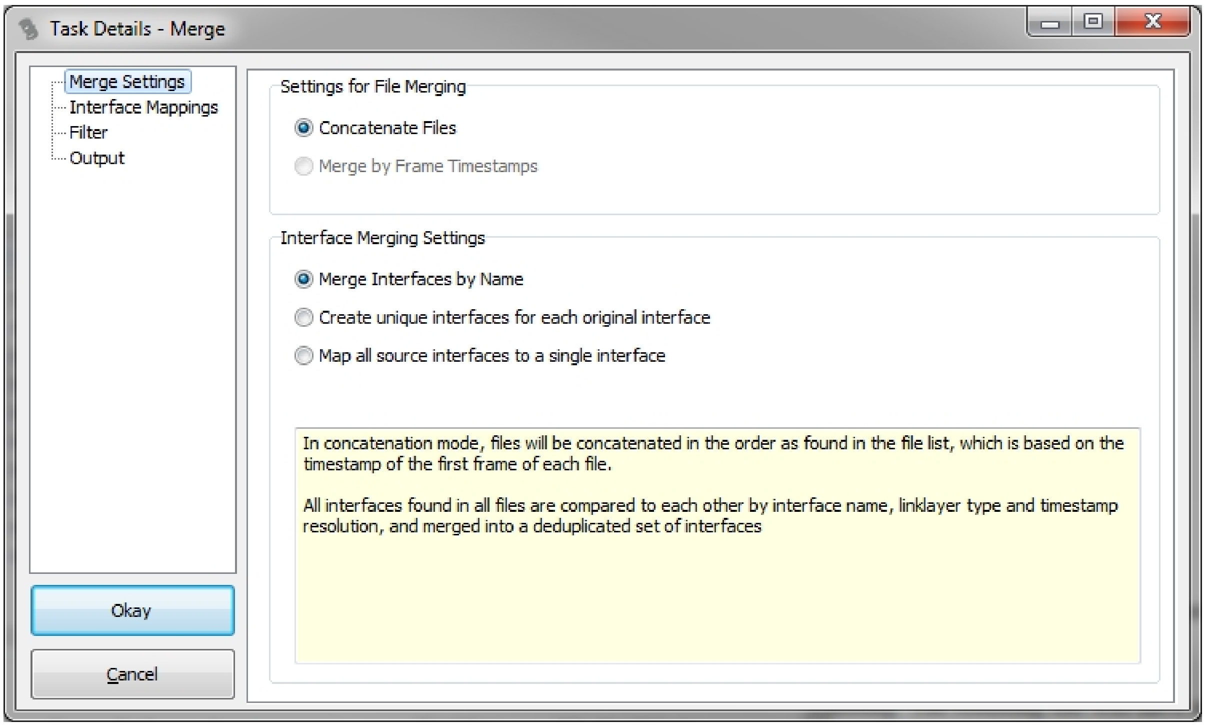
The merge process

The merge process deals with a a few issues that the current (as of June 2014) mergecap tool in the official Wireshark installation has:

- mergecap cannot merge PCAPng files that were captured on multiple interfaces. To be fair, it is quite hard to adjust settings for merging multiple interfaces with a command line tool.
- mergecap is limited to merging as many files as the command line allows to specify in the parameters. I have found that mergecap sometimes allows using wildcards, but not always (for whatever reason). This may not seem a big deal but it sometimes is a problem.
- merging by time stamp can produce incorrect results if the input files contains negative delta times between packets. This happens for captures on multiple interfaces, where frames are recorded into the capture file with out of order timestamps every now and then. To be fair again: TraceWrangler does not yet allow to merge by time stamp, either (because it's way more complicated than concatenation).

Merge Settings

The first page of the merge settings dialog allows you to set the basic parameters for the merge process.



Settings for File Merging

[✓ Concatenate Files]

Merge by Frame Timestamps

Interface Merging Settings

[✓ Merge Interfaces by Name]

Create unique interfaces for each original interface

Map all source interfaces to a single interface

In concatenation mode, files will be concatenated in the order as found in the file list, which is based on the timestamp of the first frame of each file.

All interfaces found in all files are compared to each other by interface name, linklayer type and timestamp resolution, and merged into a deduplicated set of interfaces

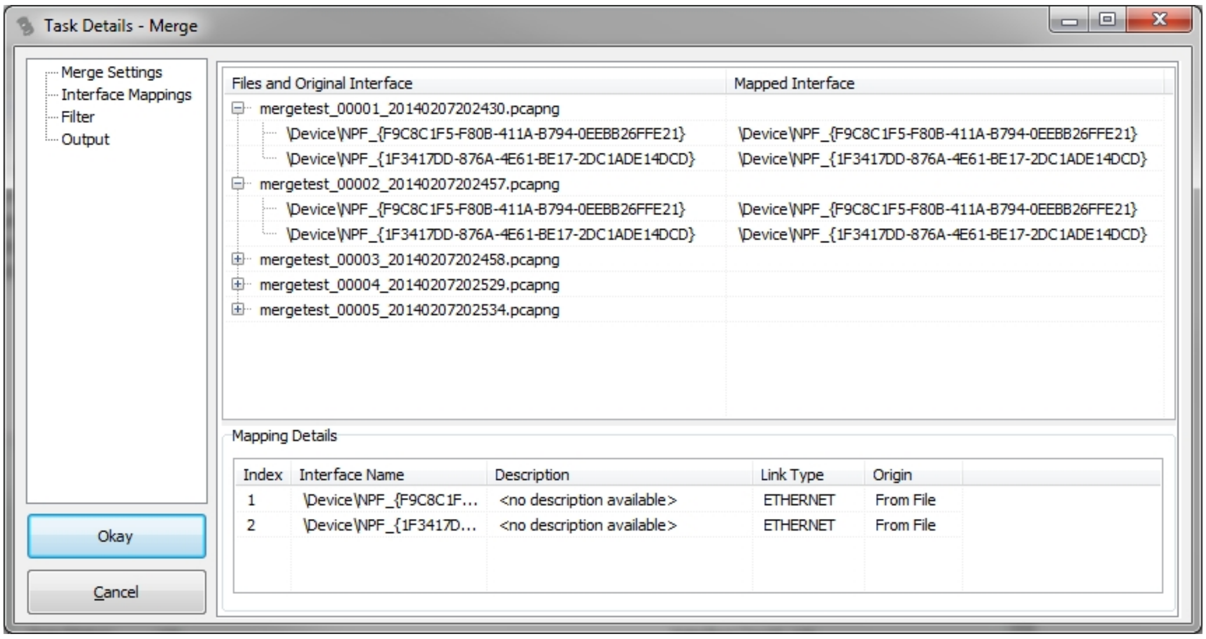
Concatenate files: This is currently the only mode available. The files in the file list will be concatenated as they are sorted in the list display, which is by first frame time stamp.

Merge Interfaces by Name: interfaces are merged by their name and link layer type, so if the same name and type are found in multiple files only one output interface will be created, and all frames matching that interface will be written with the new interface. This is the default setting because it should be the most common case.

Create unique interface for each original interface: if you would like to have a separate interface in the merge file for each interface found in the source files you can chose this setting. The resulting file will have as many interfaces as the total sum of all interfaces in all files. Map all source interfaces to a single interface: this setting will result in a target file that has only one capture interface, and all packets will be mapped to that interface.

Interface Mappings

This page allows you to fine tune the interface mapping for the merge process.



Files and Original Interface	Mapped Interface
mergetest_00001_20140207202430.pcapng	
\Device\NPF_{F9C8C1F5-F80B-411A-B794-0EEBB26FFE21}	\Device\NPF_{F9C8C1F5-F80B-411A-B794-0EEBB26FFE21}
\Device\NPF_{1F3417DD-876A-4E61-BE17-2DC1ADE14DCD}	\Device\NPF_{1F3417DD-876A-4E61-BE17-2DC1ADE14DCD}
mergetest_00002_20140207202457.pcapng	
\Device\NPF_{F9C8C1F5-F80B-411A-B794-0EEBB26FFE21}	\Device\NPF_{F9C8C1F5-F80B-411A-B794-0EEBB26FFE21}
\Device\NPF_{1F3417DD-876A-4E61-BE17-2DC1ADE14DCD}	\Device\NPF_{1F3417DD-876A-4E61-BE17-2DC1ADE14DCD}
mergetest_00003_20140207202458.pcapng	
mergetest_00004_20140207202529.pcapng	
mergetest_00005_20140207202534.pcapng	

Mapping Details

Index	Interface Name	Description	Link Type	Origin
1	\Device\NPF_{F9C8C1F…	<no description available>	ETHERNET	From File
2	\Device\NPF_{1F3417D…	<no description available>	ETHERNET	From File

The interfaces will be automatically pre-mapped by the selection on the previous page, with all available target interfaces listed at the bottom.

Adding interfaces

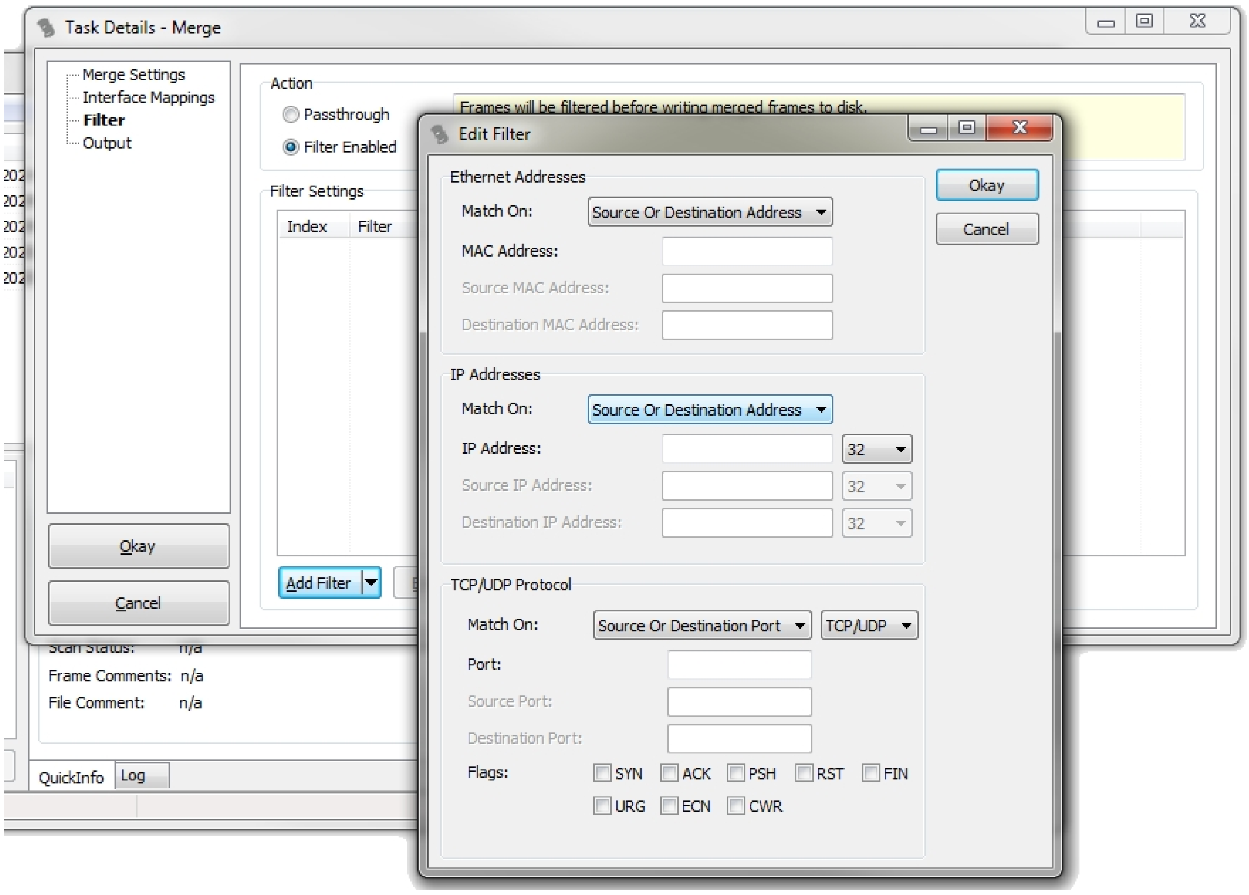
If you want you can use the pop-up menu on the target interface list to add arbitrary interfaces. Target interfaces can also be edited or removed from the list, unless it is an interface that was found while scanning the source files (when origin is "From File").

Manual mapping

Mapping interfaces is performed by selecting the existing interface in the upper list and then selecting a target interface in the lower list.

Filters

As an additional feature TraceWrangler can apply filters while merging files. The result is that only frames matching at least one filter will end up in the merged output file.



Task Details - Merge

- Merge Settings
- Interface Mappings
- **Filter**
- Output

Action: Passthrough [✓ Filter Enabled]

Frames will be filtered before writing merged frames to disk.

Filter Settings

Index	Filter

Scan Status: n/a
Frame Comments: n/a
File Comment: n/a

Edit Filter

Ethernet Addresses
Match On: Source Or Destination Address
MAC Address:
Source MAC Address:
Destination MAC Address:

IP Addresses
Match On: Source Or Destination Address
IP Address: 32
Source IP Address: 32
Destination IP Address: 32

TCP/UDP Protocol
Match On: Source Or Destination Port, TCP/UDP
Port:
Source Port:
Destination Port:
Flags: ☐ SYN ☐ ACK ☐ PSH ☐ RST ☐ FIN ☐ URG ☐ ECN ☐ CWR

The filter options are limited to simple address and port filters, with some additional TCP flags also being available. This feature is **not a replacement** for the powerful display filter engine of Wireshark/tshark, it is just a nice thing to have.

After adding a filter you'll see it listed in the filter list in Wireshark display filter syntax. This is only to allow seasoned Wireshark/tshark users to read it in a familiar way. You can't specify filters in that syntax manually.

Editing

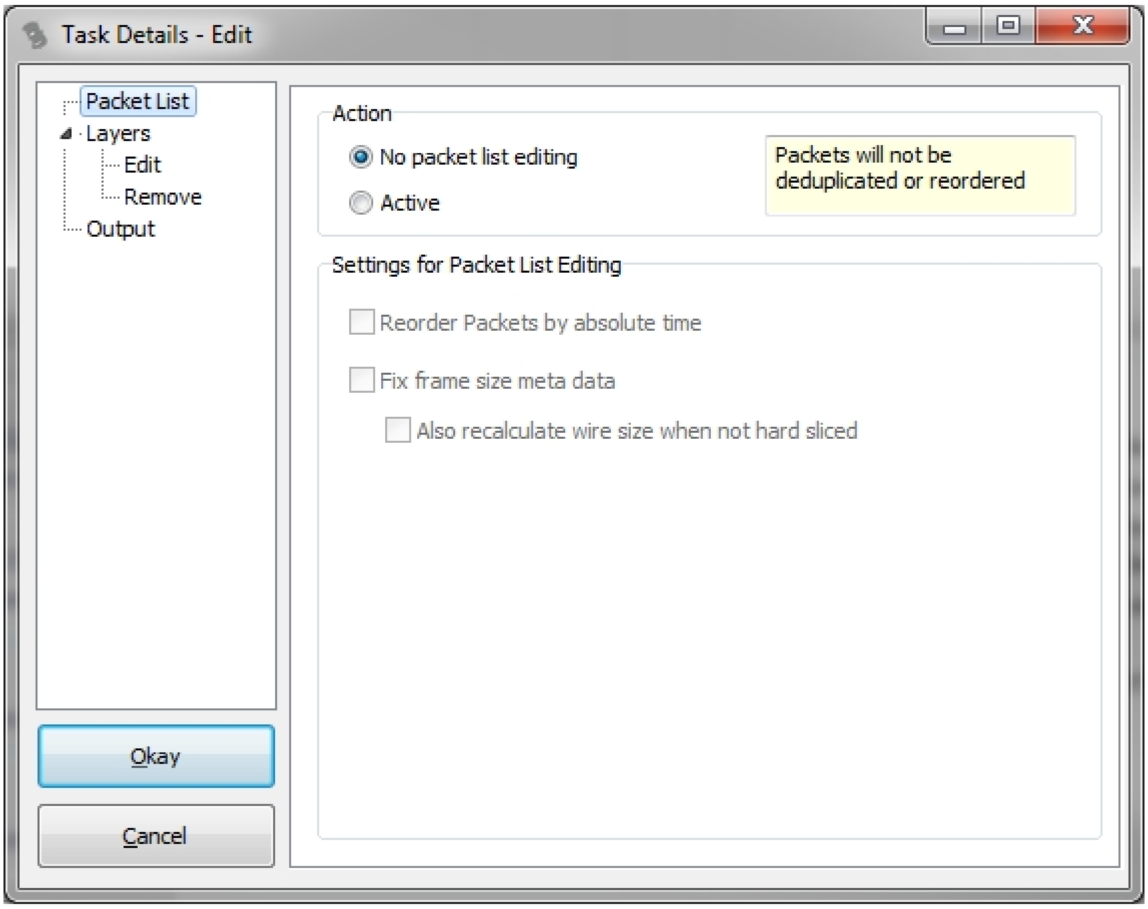
No content yet.

Introduction

Edit tasks are used to modify trace files in some way, either by working on the packet layers or by sorting or removing certain frames.

Packet List

Packet List options are functions that are performed on properties of the packet list, e.g. to reorder the packets.


Task Details - Edit

Action

[✓ No packet list editing] Active

Packets will not be deduplicated or reordered

Settings for Packet List Editing

☐ Reorder Packets by absolute time

☐ Fix frame size meta data

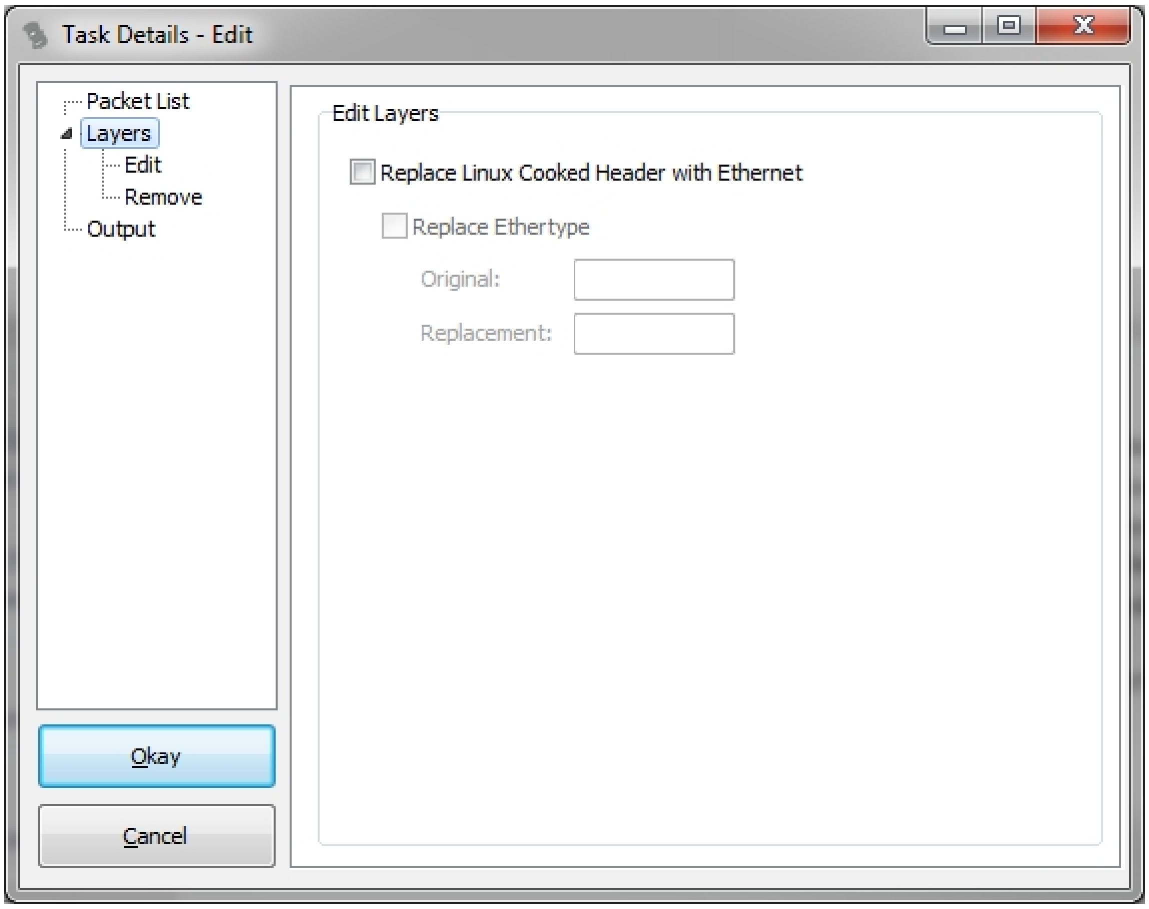
☐ Also recalculate wire size when not hard sliced

Reorder Packets by absolute time: when capturing traces using multiple interfaces in tshark/Wireshark/ dumpcap it is quite likely that some of the frames will be written to the file out of order. Those are easily spotted by looking at - or filtering for - frames with negative delta times. If this option is selected TraceWrangler will sort the frames by absolute time stamp when writing the file.

Fix frame size meta data: sometimes, frame lengths are stored incorrectly, with the frame size and capture size both set to sliced length (meaning, less bytes were stored to disk than were actually present on the wire). TraceWrangler can try to recalculate the correct wire size by looking at length fields of protocols like IPv4 and IPv6.

Also recalculate wire size when not hard sliced: normally, the meta data fixing process will only fix frames where capture size = wire size. This option forces also fixing frames when both are different, meaning that IP length information will be used in all cases

Edit Layers



Edit Layers

☐ Replace Linux Cooked Header with Ethernet

☐ Replace Ethertype

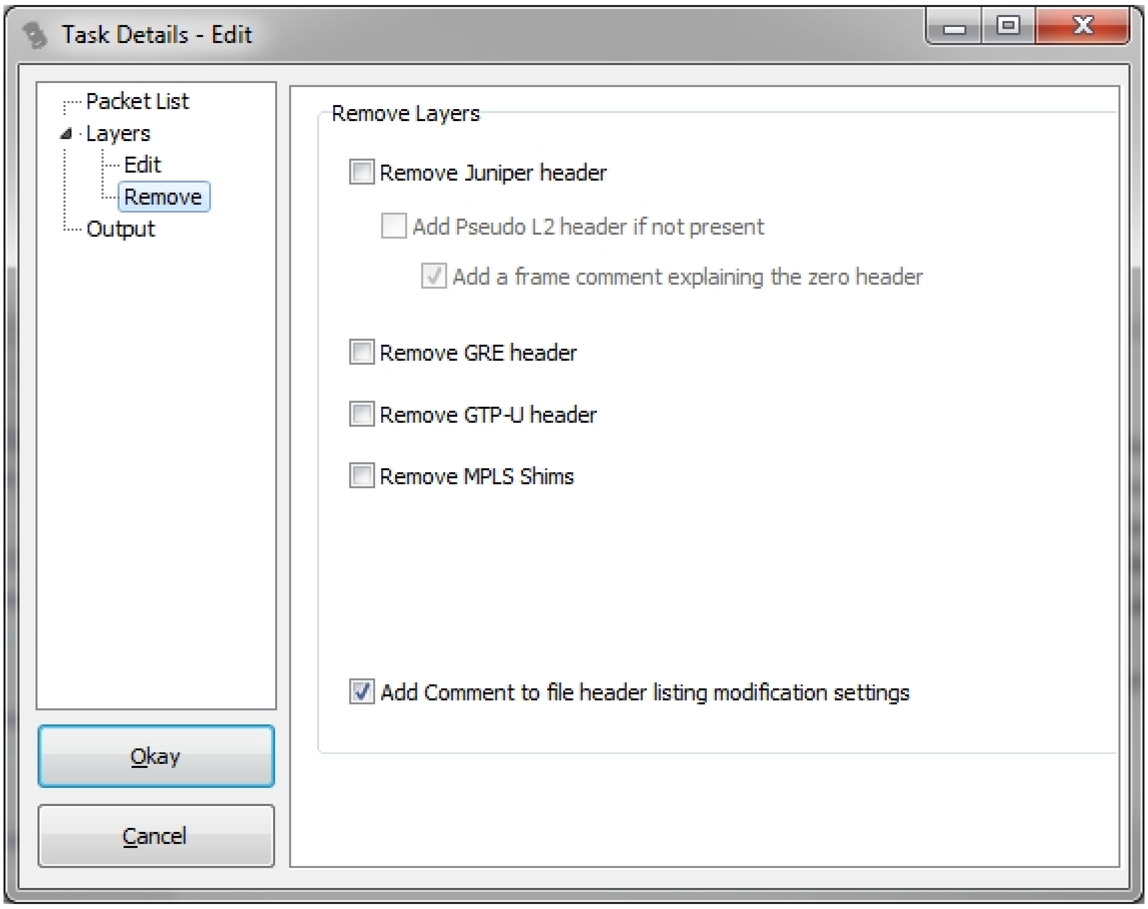
Original:

Replacement:

Replace Linux Cooked Header with Ethernet:if you capture on the "Any" interface on Linux you'll most likely end up with a trace file that contains pseudo headers instead of Ethernet headers. A new Ethernet header will be crafted, using as much information from the cooked header as possible, and written to the output file instead of the original header.

Replace Ethertype: in some cases you may want to force a specific Ethertype to be used instead of the original. If the value set in the "original" field is detected, it will be replaced with the replacement value.

Remove Layers


Task Details - Edit

Remove Layers

- [ ] Remove Juniper header
  - [ ] Add Pseudo L2 header if not present
    - [✓] Add a frame comment explaining the zero header
- [ ] Remove GRE header
- [ ] Remove GTP-U header
- [ ] Remove MPLS Shims
- [✓] Add Comment to file header listing modification settings

Remove Juniper header: trace files captured on certain Juniper devices may contain a special header preceding the Ethernet header, making it hard to open the file in a couple of network analysis tools that do not understand that header. This option will remove the Juniper header and change the link layer type with standard Ethernet.

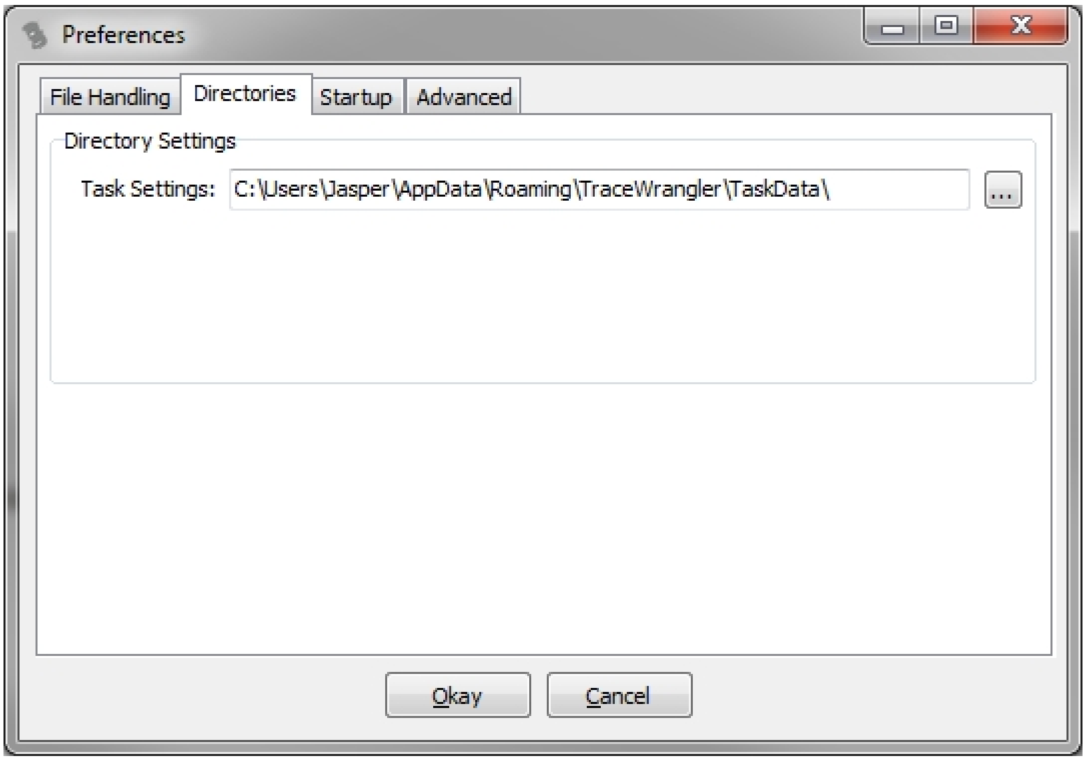
Add Pseudo L2 header if not present: in some cases there may not be a layer 2 header. With this option a generic header will be created and inserted, with zero values for the MAC addresses.

Add a frame comment explaining the zero header: adds a PCAPng frame comment explaining the modification

Remove GRE header: some packets may contain multiple IP layers when using GRE encapsulation. With this option TraceWrangler will remove everything between the Ethernet layer and the layer following the GRE layer.

Remove GTP-U header: GTP-U is used to tunnel IP over GPRS (mobile traffic). With this option you can cut away the layers between Ethernet and the tunneled IP layer.

Preferences

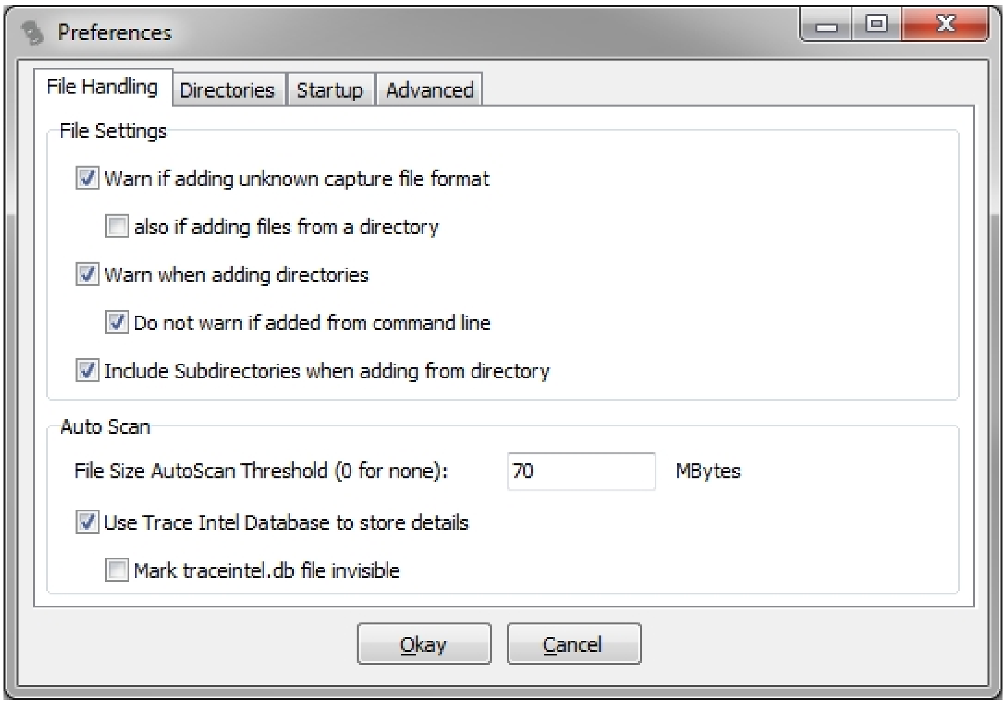


Directory Settings

Task Settings: C:\Users\Jasper\AppData\Roaming\TraceWrangler\TaskData\

File Handling

The file handling settings are mostly about how the GUI will interact with the user, and allow disabling (possibly annoying) messages. There is also a section to configure the auto scan feature.



File Settings

- [✓] Warn if adding unknown capture file format
  - [ ] also if adding files from a directory
- [✓] Warn when adding directories
  - [✓] Do not warn if added from command line
- [✓] Include Subdirectories when adding from directory

Auto Scan

File Size AutoScan Threshold (0 for none): 70 MBytes

- [✓] Use Trace Intel Database to store details
  - [ ] Mark traceintel.db file invisible

Warn if adding unknown capture file format: TraceWrangler checks the file format by reading the magic bytes which should be present in the first couple of bytes of the file. If no match is found this setting will warn the user that the format was unknown. There is one exception: there is never a warning if the traceintel.db file is added, because it can happen by adding directories or just marking all files in a directory. As an additional item it is possible to simply ignore unknown file formats that are added when selecting a directory.

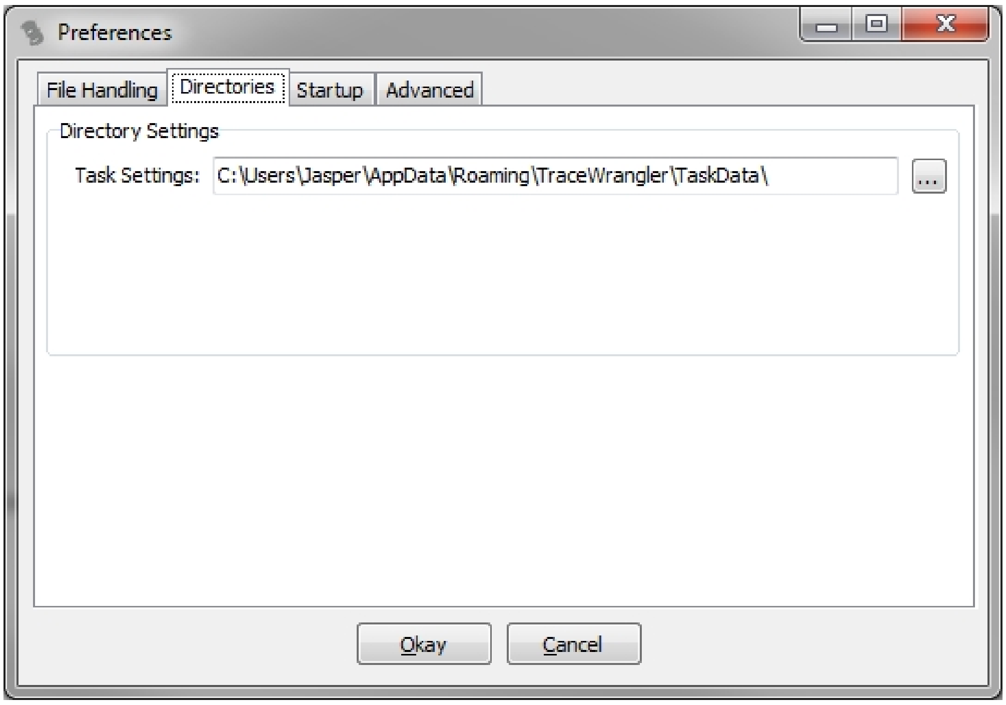
Warn when adding directories: since there may be a lot of files in a directory this warning will show up if you drag and drop a directory on the form. It will not appear if you select to add a directory from the main form button or the menu, of course.

Include subdirectories when adding from directory: if checked, all trace files found in a subdirectory will be added when adding directories.

File Size AutoScan Threshold: if a file is added which size is equal or less than the number of MBytes specified it will be scanned for statistics, endpoints and conversations automatically. Configuring larger numbers will lead to the adding process taking longer, so it is only set for 70 MBytes by default, which should be a quick scan. If you want to scan any file regardless of its size you can specify a size of 0 MBytes.

Use Trace Intel Database to store details: if this setting is selected, all scan results will be stored in an SQLite database file sitting in the same directory as the trace file that was scanned. See details here.

Directories

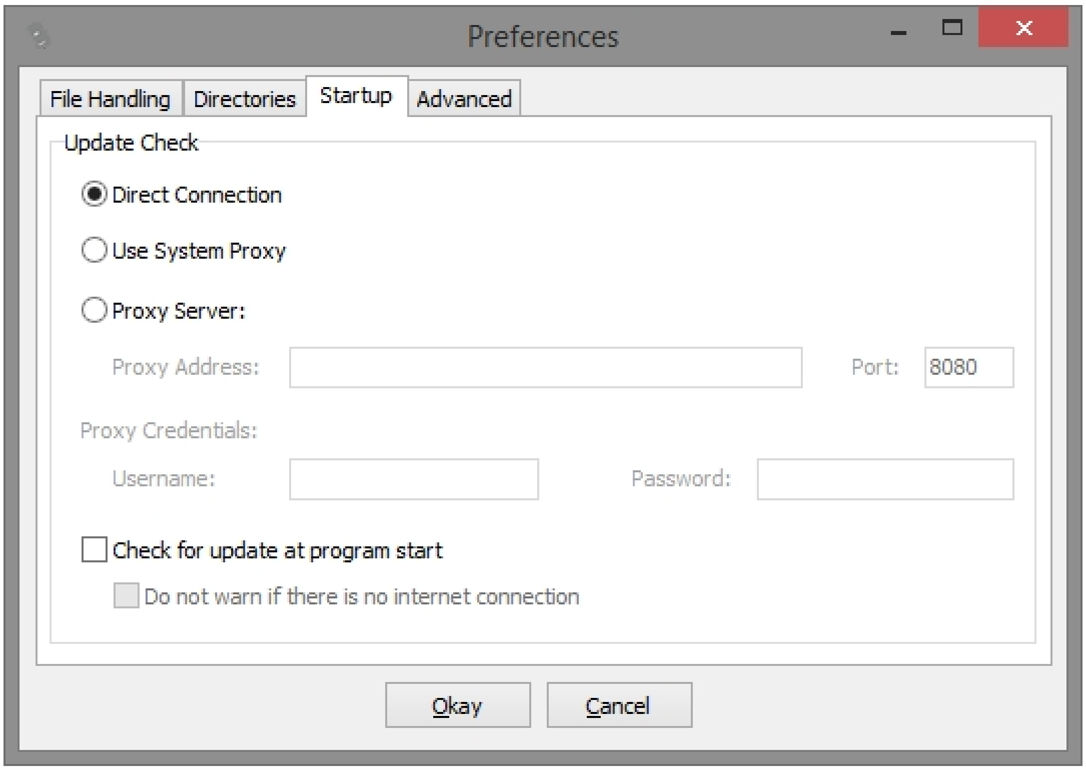


Task Settings: C:\Users\Jasper\AppData\Roaming\TraceWrangler\TaskData\

Task Settings: whenever you create a task, the settings for that task will be stored in a single SQLite database file, using the task name as the file name. This settings defines where the task files are stored.

Startup

TraceWrangler does not use communication except for update checks.



Update Check

(●) Direct Connection

( ) Use System Proxy

( ) Proxy Server:

Proxy Address: Port: 8080

Proxy Credentials:

Username: Password:

☐ Check for update at program start

☐ Do not warn if there is no internet connection

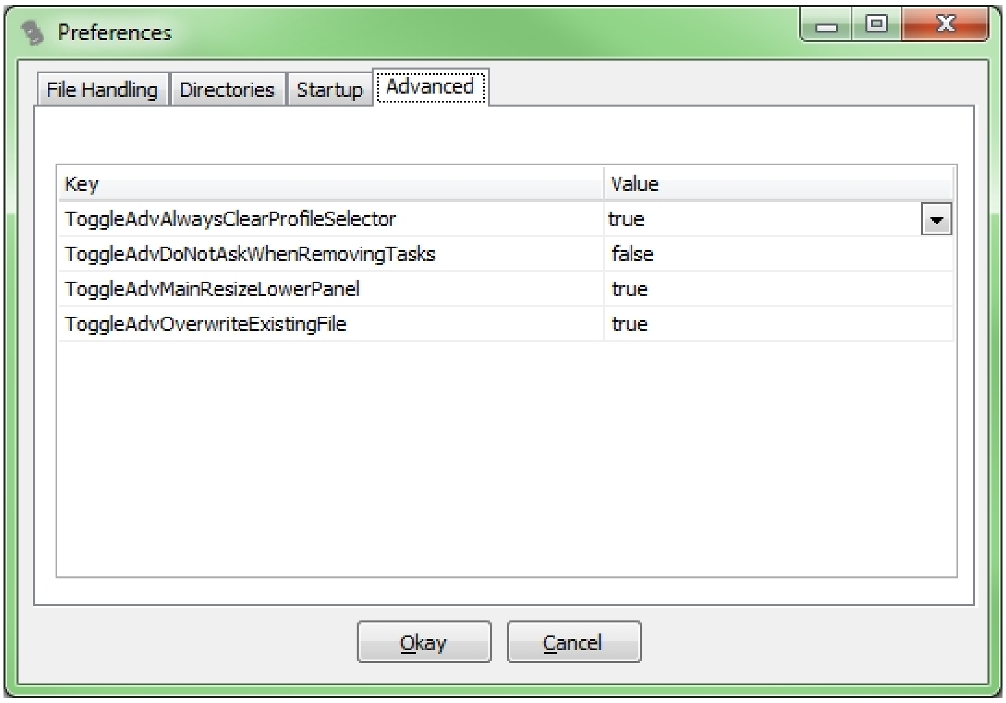
Use System Proxy: TraceWrangler will try to determine your system proxy setting and use it to check for new versions

Use Proxy Server: if checked, it will allow setting a HTTP proxy server that will be used to check for new versions.

Check for update at program start: if checked, TraceWrangler will check for a new version at each start. You can configure it to skip the connectivity warning if no internet connection is available.

Advanced

The advanced tab is used to store settings that should normally not needed to be touched. Sometimes you may see more lines than shown below, which is caused by older versions still keeping some values in the preferences file.



Key	Value
ToggleAdvAlwaysClearProfileSelector	true
ToggleAdvDoNotAskWhenRemovingTasks	false
ToggleAdvMainResizeLowerPanel	true
ToggleAdvOverwriteExistingFile	true

Always Clear Profile Selector: if checked, the profile selector will be cleared for each new task added. Otherwise it will show the same settings as the last time.

Do not ask when removing tasks: if set to true, no warnings will show up when removing tasks from the task list. Since this will also deleted the task settings file the default is to warn.

Main Resize lower panel: by default, the lower panel of the main window is resized when you resize the form, to make it easier to work with the lower right panel. Set this to false if you want the file list to be resized instead.

Overwrite existing file: if set to true, a warning will be shown when files are overwritten.

Internals

Command Line Parameters

There are a couple of command line parameters that you can use when starting TraceWrangler:

1. Any parameter that ends on ".task" will be considered as a task settings file. If the file exists the according task will be added to the task list right away, even if there is no file in the file list yet. This can be helpful if you want to create a standard task that you want to be in the task list whenever you start TraceWrangler.
2. Any other parameter will be checked if there is a file of that name, and if so, it will be added to the file list instantly. If it turns out to be a directory instead, all files in that directory will be added as long as they are one of the capture file formats TraceWrangler recognizes.

Example:

```
c:\> tracewrangler.exe d:\traces\test.pcapng d:\test\anon.task
```

This will add the file "test.pcapng" from the directory "d:\traces" to the file list (if the file exists, of course), and add the task "anon.task" to the task list (again, if it exists).

Trace Intel Database file

TraceWrangler uses an SQLite database to store details of trace files it processed. The idea is to only have to scan a file **once** and being able to perform multiple tasks depending on the scan results even if TraceWrangler is shut down and started again at a later time. The name of the database file is "traceintel.db", and it may be a hidden file, depending on the setting in the preferences dialog. It is located in the same directory as the files it contains information about. If you scan files in multiple directories each will have its own traceintel.db file.

Data kept in the database includes:

1. Details about the PCAPng block file structure in the file
2. General statistics for each file, including but not restricted to
   - the number of frames
   - min/max/average frame sizes
   - slicing information (when frames are stored with some of their payload truncated)
   - time order (if there are negative delta times or not)
   - time stamp of the last frame
   - total number of bytes
3. Conversation details, including Ethernet, IPv4, IPv6, TCP and UDP endpoints and conversations.

Whenever a file is added to the list, TraceWrangler will check if there is a database file in the same directory as the trace file. If there is, it will open the database and check if it can find the trace file, which is done by checking three things:

1. the Filename has to match, excluding the file path. This means that the file path can be different, in case the file has moved to another location, together with the database file, of course.
2. the file size has to match exactly to the byte for any file type except PCAPng
3. the time stamp of the first frame in the file has to match to the nanosecond
4. if the file type is PCAPng and the file size is different (which can happen to a file if comments are added or removed) the time stamp of the last frame in the file is checked as well. If it is the same as recorded in the database, the file is considered to be a match even though the file size has changed.

By requiring a match of the file size and the first frame time stamp TraceWrangler makes sure that the file wasn't overwritten with a newer capture using the same name, because even if the file size would match (which is already highly unlikely) the time stamp of the first frame in the file will not (unless doctored, of course). The only exception is PCAPng regarding the file size, as already mentioned.

Conversation Store

The Conversation Store is a memory structure that keeps records of all endpoints and conversations. Each file is scanned for endpoints and conversations for

1. Ethernet
2. IPv4 and IPv6
3. TCP
4. UDP

All conversation details are stored to the Trace Intel Database, if not disabled in the preferences.

Note: there are plans to also scan for FQDNs seen in a file, but that is not really a conversation and the DNS/DHCP parsers are still a little too unstable in some situations for this. My bad :-)

Additional tools

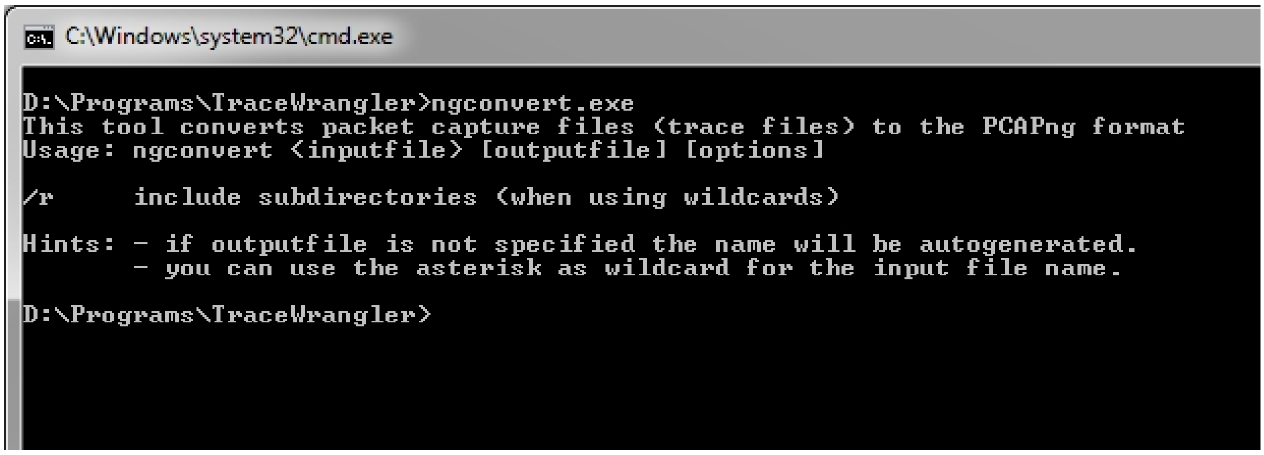
TraceWrangler is the main program, but there are a couple of helpers that perform tasks that may be easier to accomplish with a command line tool.

ngconvert

ngconvert is a command line tool that can be used to convert capture files to the PCAPng file format. It can read the same formats as TraceWrangler, which are

- libpcap (old Wireshark/dumpcap versions, tcpdump, etc.)
- Sniffer DOS format (.enc)
- Sniffer CAP format (.cap)

If no output file name is specified, or if multiple files are converted, ngconvert will use the original file name and just replace the file extension with "pcapng".



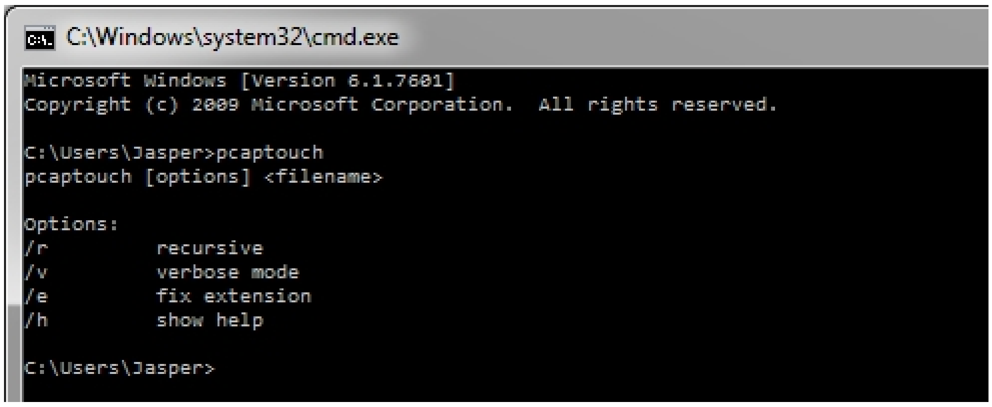
```
C:\Windows\system32\cmd.exe

D:\Programs\TraceWrangler>ngconvert.exe
This tool converts packet capture files (trace files) to the PCAPng format
Usage: ngconvert <inputfile> [outputfile] [options]

/r      include subdirectories (when using wildcards)

Hints: - if outputfile is not specified the name will be autogenerated.
       - you can use the asterisk as wildcard for the input file name.

D:\Programs\TraceWrangler>
```

It has two main features that are unique at the moment, which are:

- it allows the use of wildcards and is able to convert all files found in subdirectories
- when converting Sniffer CAP files it is able to preserve multiple capture interfaces

pcaptouch

pcaptouch is a command line tool that updates the file time stamp of capture files by reading the first frame and setting the same time for the file.



```
C:\Windows\system32\cmd.exe

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Jasper>pcaptouch
pcaptouch [options] <filename>

Options:
/r          recursive
/v          verbose mode
/e          fix extension
/h          show help

C:\Users\Jasper>
```

With the /r parameter pcaptouch will go through subdirectories recursively, while the /e parameter will also fix the file extension (setting it to pcap or pcapng based on the magic bytes seen in the file)

Appendix

Changelog

- Beta 0.6.0 (5th of July, 2016)
    - New Features:
        - Conversation Summary
            - lines can be marked now, and will stay visible even if not matching a filter if applied. Jumping from marked lines to the next/previous is possible
            - New highlight mode for TCP conversations to show conversations that start and end within the time of a selected conversation
            - New aggregated view for TCP conversations, grouping them by IP pair
            - Added copy-to-clipboard for the selected cell, plus external IP lookup when an IP address is selected
        - New protocols supported
            - RTPS (parsing and assembling, as far as required for sanitization)
            - ERSPAN parser added
        - Sanitization Tasks
            - GTPv1 can now be assembled, allowing sanitization of packets with a GTPv1 layer
            - Sanitization task can now reassemble IPv4 fragments to allow sanitizing reassembled payloads (RTPS sanitization forced this feature)
        - pcaptouch command line executable added, allowing setting the file time stamp to the time of the first frame in each file. It can also fix the file extension
    - Enhancements:
        - ngconvert now displaying the percentage of progress when converting a file
        - write speed when writing frames to PCAPng files improved for all tools
        - Main Window
            - Buttons added on top of task list to allow even faster adding of tasks if there isn't any yet
            - code added to allow ignoring file time stamp setting error messages
            - warning message added when files are not auto scanned due to threshold
            - window label added to be able to distinguish various TW instances
        - Preferences
            - Setting added to turn off warnings about files not being auto scanned
            - Proxy detection for performing update checks should work better now
        - Conversation Summary
            - pop-up menu added to column headers to allow hiding and showing columns (settings are not persistent yet)
            - displaying FlowCount per IP conversation entry plus average TCP RTT if available
            - displaying total TCP Flags per Conversation in TCPDump Format
            - enhanced the pop-up menus with shortcut keys
        - TraceIntel database
            - table structure changed to store more details
        - Sanitization Tasks
            - parameter added for keeping Ethernet trailer intact
            - setting added to allow re-padding of payloads
            - setting added for ignoring length differences in replacement strings
            - black marker settings added for addresses (replace all bytes with zeros)
            - parameters added for RTPS sanitization
            - setting added for preventing IP reassembly
            - now calculating delta between IP total length and wire size to see if the new IP total length needs to be adjusted. This can happen when encountering hard sliced frames and keeping the payload/replacing it with zeros/strings
            - functionality added to preserve Ethernet trailer

        - Edit Tasks
            - added detection of ERSPAN layers and removing them when GRE is cut away
            - IPv6 wasn't treated correctly for GTPv1 editing; now adjusting EtherType
            - now using the highest L2 header before the GTPv1 header as new L2 base
            - now able to handle SLL headers with malformed VLAN information injected into the header as if it were Ethernet. VLAN information is kept intact and added to the Pseudo Ethernet header
            - sub option added to meta editing to allow to force wire size modification even if wire size and capture size are not the same (as per request of Nathan Flowers)
        - Extraction Tasks
            - Exporting frames to multiple files now works based on the filename schema instead of what filters were used
            - displaying filters in grayed out mode if no frame will match, based on if the filter has a first frame location assigned (after looking conversations up in the conversation store for the filter)
            - status panel added to show process of creating filters from beacon file when the file is larger than a certain threshold
            - functionality added to import filters from CSV
            - VLANID placeholder added
            - replaced checkbox with "match all" hint
        - Merge Tasks
            - writing file comments is now configurable to avoid writing large source file name lists to the SHB comment field
            - feature added to write the filename of each source file for the first frame merged into the output file (only in concatenation mode)
        - ICMPv6 Parser
            - Handling of ICMP redirects added
        - IPv4 Assembler
            - Forcing payload length improved
    - Fixes:
        - Main Window
            - Update URL was still using port 80. Now changed to 443
            - Update check improved to hopefully work with most proxies now
            - Skipping over traceintel.db didn't work for all methods of adding files
        - Preferences
            - Storing values to disk may crash for items with quotation marks
        - Conversation Summary
            - RTT string output used ThousandSeparator instead of DecimalSeparator
            - CSV export now uses quotes for all fields
            - CSV export checks if the file exists
            - some changes to get low screen resolution to work
            - filter matching now case insensitive
            - some columns were not sortable
        - Sanitization Tasks
            - enabling MAC replacements but not setting any randomization lead to random MACs
            - determining a replacement EUI64 address failed when Ethernet sanitization was not active
            - now handling :: and ::1 correctly
            - UDP ports could be randomized even if not configured when UDP sanitization was enabled due to an uninitialized variable

    - now also checking the IPv4 original for being loopback or all-zero and returning it instead of a replacement except if a DB replacement was configured
    - storing VLAN replacements was in the wrong order, failing when hitting duplicates
    - IP total length modification was resulting in values too small when Ethernet trailer was present
    - replacing L4 ports by socket list wasn't working correctly; it requried to enter the replacement IP (which may not already be known) to do it's job. Now using the original IP
    - CRC handling for TCP & UDP was not always correct
    - various task form glitches fixed
- Extraction Tasks
    - crash fixed when trying to get an interface that isn't used by any output frame
    - adding frames from beacons failed if not a trace file
- TraceIntel database
    - memory leak when keeping creating new IntelDB instances even though running in persistent DB mode
    - major overhaul of the persistent IntelDB mechanism which was incomplete and leaking memory
    - avoid writing the database to disk unnecessarily
- Frame data structures
    - creating a new frame sometimes crashed by not copying the comment list
    - stupid zeroing loop was slowing down initialization
    - increased the maximum number of interfaces to 256 (from 32), because mergecap writes an ISB for each file merged
- Loaders
    - NAI CAP: offset of next packet was determined as wire size, not capture size, leading to crashes on sliced files
    - PCAP: Raw IP link layer type 101 added
    - PCAPng: added code to handle fractional time stamp calculation
    - determining last frame offset and index was completely broken (looks like copy&paste error from first frame)
- Filters
    - Reading filters from file wasn't initializing them correctly in some cases
    - Filtering gave wrong results in some situations, due to multiple bugs
- General parsing
    - parsing wasn't handling the case of an Ethernet trailer > 4
    - failed plausibly checks did not set the correct parse result status, leading to problems (e.g. with NetFlow when not parsing v5 types)
- ICMPv6 Parser
    - crash fixed when encountering unknown ICMP type
- MPLS Parser
    - MPLS shims with Ethernet following sometimes have an additional control word, which adds 4 more bytes which need to be skipped
- NetFlow Parser
    - plausibility check allowed all NetFlow versions; now reduced to 5 as we can't parse any other
- IPv4 Assembler
    - setting flag values failed in some cases because of not clearing all of them first
    - setting flags cleared fragment offset as well, always ending up as zero
    - fragment offset now correctly stored as group of 8 octets
- TCP Assembler

            - keeping CRC when copying values from TCP parser
        - UDP Assembler
            - merging UDP header and payload for checksum calculation was using the wrong size variable, leading to broken CRCs
            - setting keepCRC to true when copying data from parser
        - DNS Parser
            - some records weren't parsed because of an off-by-one check
- Beta 0.4.0 build 616 (17th of June, 2015)
    - New Features:
        - Conversation List:
            - Added the option to export rows into separate files
        - Main Window
            - Added rename option to tools menu, allowing to rename the files in the list based on a pattern
        - Tasks
            - Extraction task added to available tasks
            - Extraction task can create filters from capture files and Snort alert logs
        - New parsers added
            - VXLAN
        - New assemblers added:
            - VXLAN
            - GTP-u V1
    - Enhancements:
        - Main Window
            - task process details pane added; right now only used for extraction tasks
            - tasks are marked red when they can't be run for some reason
        - Conversation List
            - handshake seen and teardown seen are now separate TCP flow status messages
            - displaying TCP flow teardown status when no handshake was seen
            - calculating the total bytes and frames of all selected rows
            - displaying a hint for each row, listing the files the row was seen in
            - code added to allow double clicking rows to open them directly in the associated PCAPng file viewer
            - Code added to export list rows to CSV
            - detecting hard slicing and adding an attribute if found
            - simple TCP Port reuse tracking added (seeing a SYN after having a complete conversation)
            - added determining if a socket was refused or had inactivity timeouts
        - Anonymization
            - option to generate random IPs in the IPv4 private range if original is private
            - determining if an IPv4 address is a documentation address and ignoring it if option is set
            - SocketPortPair added to support defining TCP/UDP port replacements for specific original sockets only
            - code added to dermine Multicast base mac for an IPv6 address
            - setting an IPv6 socket now allows snort notation (without square brackets), needed for Snort alert log parsing
            - checks added to make sure a replacement is not the same as the original by chance
            - code added to randomize IPv6 addresses and prefixes stateless or stateful

            - function added to allow removing unwanted frames from the destination file, currently used for ICMPv4 type/code
            - Payload and general settings now have their own tabs
            - can now anonymize frames containing GTP-u tunneling layers
        - Editing
            - code added to allow replacing a specific Ethertype when converting Linux cooked captures to pseudo Ethernet
            - Juniper L2 header options updated
            - added the option to remove MPLS shims
            - option added to add a comment to a frame if the Juniper header has no L2 structure and a fake Ethernet header is built
        - Parsers
            - handling Ethernet over MPLS by looking at potential Ethertypes
        - Filters
            - Filters can now carry a label that can be used for file name generation
            - Filter editor now allows copy & paste
        - Preferences
            - Proxy settings can now be taken from system settings
        - General
            - code added to allow using a persistent TraceIntelDB for loading multiple files faster from the same directory
    - Bug Fixes:
        - Main Window
            - uses HTTPS URL for update checks now, not needing the redirect anymore
            - updating the file details pane didn't work when a previously selected file was reselected
            - displaying the comments for files wasn't working
        - Conversation List:
            - clearing the interface mapping list was missing, leading to problems when exporting files
            - Filter not always working correctly when negating the expression
            - Copy/Paste now working for filter field
            - crash when extracting flows when trying to access non existing file entries
            - TCP Handshake and teardown status wasn't tracked correctly
            - fixed loading IPv4 endpoint table twice and IPv6 endpoint table not at all
            - fixed determining if there is an Ethernet trailer failed when multiple Ethernet layers were present, e.g via VXLAN encapsulation
            - determining Ethernet trailer was wrong for IPv6
        - Anonymization
            - ICMPv6: Setting Source and Target link layer addresses wasn't reflected in the assembled frame
            - ICMPv6 checksums weren't calculated correctly
            - UDP checksum cannot be 0
            - determining if an IPv6 address is a solicited node multicast was broken
            - determining Pseudo Header was sometimes not working correctly (it ignored ICMPv6 and fragmentation headers)
            - crash fixed when looking for a GUID replacement
            - crash fixed when trying to store Solicited Node Multicast address replacements more than once
            - fixed missing path delimiter to replacement DB path when forced in settings
            - fixed crash when creating random fe80 addresses failed

            - fixed that MAC addresses weren't checked for being an IPv6 multicast carrier
            - processing Ethernet addresses even when Ethernet sanitization is not enabled but IPv6 requires it to
            - processing IPv6 neighbor solicitation/advertisement wasn't working correctly
            - processing IPv6 ND wasn't working correctly; now looking at the discovery target first
            - code added to allow replacing unknown payload bytes with zeros or a pattern
            - input validation of content replacement strings failed when no '|' where used
        - Editing
            - removing GTP-u layers now works also for frames with more than 2 IP layers
            - creating a pseudo Ethernet header was not really working
        - General
            - storing task details lost some information in certain situations
            - writing PCAPng interfaces was broken when the original interface was modified instead of the copy
            - setting wrong link type on write when it was changed for the new interface
            - not writing NRB anymore when copying over blocks from the loader and there are no records for IPv4 and IPv6
            - loader did not identify Juniper link layer type
            - file entries were duplicated under some circumstances
            - Intel DB version raised to 1.3 to void older dbs (structure has changed)
            - various minor bugs fixed
- Beta 0.3.6 build 583 (2nd of January, 2015)
    - New Features:
        - Main Window now shows total frames and bytes of all loaded files if available
        - Conversation List
            - supports "not" or "!" as an operator for the filter box
            - added a simple "Export to CSV" feature for TCP
            - shows total number of frames and bytes in the status bar of all selected rows
        - New parsers added
            - NetFlow v5
            - HSRP
            - Geneva
        - New assemblers added:
            - NetFlow v5
            - HSRP
        - Sanitization
            - new setting for IPv4 to allow replacing private IP addresses with a different random private address, or keep it, or fully randomize it
            - IPv4 documentation IPs can now be excluded from randomization
            - Selecting ICMPv4 type/code values to remove matching frames from the capture completely
            - TCP/UDP sanitization settings now allow replacing a port specifically for an IP/Port combination
    - Enhancements:
        - Conversation List
            - can report TCP Teardown without a initial handshake, and conversation that succeed only after multiple SYN retries/rejects
            - Keep the focus on the selected node when sorting
            - Shows more attributes, like Ethernet Trailer found, Hard Slicing

            - Exporting Conversations from the list will now skip over irrelevant files to speed up the extraction process
        - Sanitization
            - EUI64 based IPv6 addresses are now sanitized in synchronization with the MAC address they are based upon
            - Additional checks were added to make sure randomized replacements are not exactly the same as the original
        - PCAPng Loader will now auto correct irrelevant errors when loading files, e.g. capture size > wire size
    - Bug Fixes:
        - File details in main window weren't cleared when no files are selected anymore. Also they weren't redisplayed if a previously selected file was reselected.
        - New version check did not work with HTTPS version of the TraceWrangler home page
        - Conversation List did not display IPv6 based UDP conversations correctly
        - Conversation List did not show the list of conversations based on the order their first frames are found in the capture file
        - PCAPng loader would crash when Interface Description Blocks were not all written before the first packet block (Thanks to Pascal Quantin)
        - TCP parser could crash in some cases when parsing options
        - Writing to IntelDB could crash when Snap Length was too big
        - File entries were duplicated in the file table under some circumstances, so also deleting them failed
        - UDP port replacement crash fixed
        - GUID replacement crash fixed

- Beta 0.3.4 build 546 (5th of October, 2014)
    - New Features:
        - Anonymization tasks
            - accepts Snort style content byte strings for payload replacement
        - Offline help file added. Can be opened from the help menu of the main window
        - Conversation/Endpoint Statistics
            - Lists allows display filtering by using a text box in the status bar
            - one or more selected conversation can be extracted to a single file, even if spanning multiple input files
            - Conversations are now checked for additional attributes, e.g. one sided conversations, spanning files, etc.
            - Conversations are tracked for which interfaces their frames were captured on. This allows writing files with only interface blocks that are in use
    - Enhancements:
        - TraceIntelDB
            - stores more details about endpoints and conversations, including the time stamp of the last frame if available
            - determining if a file is the same as an entry in the database now checks first and last frame time stamp for PCAPng and ignores file size changes
            - database is now tracking if any changes need to be written to disk, so memory based databases are not written to disk anymore unless it is necessary, saving a lot of time
        - PCAPng loader
            - small errors in capture files are corrected if possible, e.g. if capture size > wire size
        - Parse engine
            - can now be told to stop parsing after Layer 2 and 3, e.g. when it is clear while filtering that no more depth is needed
    - Bug Fixes:
        - General

        - byte swap routines failed for the 64 bit version since the in-line assembler instructions needed to be improved for that
    - Loaders
        - NAI CAP loader crashed when reading certain files where the interface entries weren't parsed correctly
    - Anonymization tasks
        - Copy & Paste now working for Replacement String Cells. Will always copy whole cell and overwrite whole cell on paste
    - Main Window
        - no longer switch to file list even when adding files was canceled and the list is empty
        - file hints were not shown on mouse over events
        - conversation option in tool window was not disabled when the last file was removed from the file list
    - Conversation/Endpoint Statistics
        - Duration values did not calculate the minutes part correctly for values above 60 minutes
    - Parsers:
        - Ethernet parser did not return OSI layer value
        - some memory leaks fixed

– Beta 0.3.1 build 517
Fixed: ICMPv4 and ICMPv6 payload handling and checksum calculation were incorrect
Fixed: IPv4 header length was not calculated correctly
Fixed: IPv4 options were not copied correctly when assembling new IPv4 layer
Fixed: TCP options were not copied correctly when assembling new TCP layer
Fixed: maximum frame size was limited to 10k bytes, now increased to 64k bytes
Fixed: off-by-one error fixed when reserving frame buffer space
Fixed: calculating delta times failed in some situations
Fixed: calculating CRC32 for frame content was incorrect. CRC32 is necessary for calculation of Ethernet FCS
Fixed: using 16 bit values for frame size and capture length is not enough in some cases, increased to 32 bits
Fixed: problems occurred on systems that have no daylight savings time
Fixed: zero IPv6 address returned 0:0:0:0:0:0:0:0 instead of ::
Fixed: Greater than/Lesser than operators for IPv6 addresses where not working correctly
Fixed: stack overflow when returning a string for a tIPv6Socket type
Fixed: returning NIL if an interface cannot be found by index, fixing a bad crash
Fixed: CAPTimeToTicks routine to handle some new trace files made by Network General S6040
Fixed: reading Network General S6040 channels are now linked to correct interfaces
Fixed: reading the PCAPng Master blocks did not look at the BlockTotalLen correctly
Fixed: reading batches of frames could run into trouble sometimes because the remaining Filesize was not always checked correctly first
Fixed: nanoseconds where not assigned correctly when calculating frame timestamps while writing frames to PCAPng files
Fixed: calculation of the option length of PCAPng EnhancedBlocks was wrong when a filter type was set but the filter was empty
Fixed: writing PCAPng frames did not explicitly set the interface timestamp resolution
Fixed: WriteFrameBytes now using longword as FrameSize type when writing PCAPng files
Fixed: memory leak fixed in the input file scan routine
Fixed: IntelDB file was not disconnected correctly after writing statistics
Fixed: determining if frames where sliced or not wasn't working correctly, and still may need some more work for oversized frames
Fixed: frame time order was not determined correctly in some cases

Fixed: capture interfaces were not transfered to the interface list for ENC and PCAP files
Fixed: trace duration now calculated at the end of the scan
Fixed: Trace duration was not calculated after loading statistics from DB
Fixed: Block Chain Root was not assigned to PCAPng Statistics when loading from DB
Fixed: deleting the entries in the file list failed when more than one entry were in the list
Fixed: anonymization tasks loading IPv4 IP address replacements produced wrong results since the replacement address was not assigned
Fixed: parsing Dynamic Update records failed due to parsing Zone information at the wrong offset
Fixed: parsing records with more than 3 recursive pointers failed due to loop protection code, limit increased to 16 now
Fixed: parsing IP addresses from answer records failed for bogus answer lengths; now length checks ensure enough bytes are present and while issuing a warning the data will be parsed
Fixed: parsing DHCP for strings for ServerHostname and Boot Image failed when they were not empty
Fixed: parsing DHCP pointered FQDNs failed for TCP packets due to missing offset increase of 2 bytes
Fixed: EOL option didn't force aborting parsing of TCP options, so parser got stuck when hitting an arbitrary "option len" byte of 0
Fixed: parsing GTP headers other than the tunnel message type lead to a memory leak and a crash
Fixed: internal memory leak happened when querying the length of the remaining payload
Fixed: another memory leak fixed where the payload chunk was not freed
Fixed: adding new socket layers wasn't working when multiple IP layers were encountered in a frame
Fixed: truncated ICMPv4 payloads could lead to negative trailer length, which in turn lead to wrong packet length construction in sanitization tasks
Fixed: some indexes were off by one when parsing frames, leading to crashes
Fixed: determining if a protocol should be parsed or treated as payload was not working correctly
Fixed: IPv4 sanitization did not process the IPv4 option list
Fixed: Ipv4 sanitization got the IP payload length wrong sometimes
Fixed: truncating an already truncated original frame the IP length was not set correctly
Fixed: handling payload when cutting layers of was not working
Fixed: sanitizing ICMpv4 was not checking the ICMP type to see if it was an Echo Request/Reply
Fixed: memory leak fixed in GetEthernetAssembly caused by assembly.free not being called
Fixed: evil memory leak fixed caused by an object creating payload copies to determine its length
Fixed: two memory leaks fixed when discarding payloads due to sanitization parameter settings. No, it is not enough to set the length of the payload to 0. Free is a must.
Fixed: another memory leak fixed where the final payload to construct the sanitized frame wasn't freed
Fixed: Edit tasks Loader and Writer weren't properly closed at the end of the task
Fixed: crash in edit task when assigning LinkType when not running Cooked Header or Juniper Replacement
Fixed: crash in edit task when closing the writer twice
Fixed: creating replacements for empty FQDNs failed; now an empty FQDN is returned
Fixed: off-by-one error when creating replacement FQDNs
Fixed: replacement FQDNs were backwards. My bad, should have been tested :-)
Fixed: adding files to the file list was streamlined in all variants
Fixed: showing strange results in main window when no frames where found in a file
Fixed: making the PCAPng struct tab visible forced it to become active; now the current tab stays current
Fixed: new files were not shown in the list until every file in the batch had been added
Fixed: file status was not updated in some cases
Fixed: not showing the first frametime stamp if the file hadn't been deeply scanned
Fixed: manual scanning didn't work correctly

Changed: now using frametimestamps with nanosecond details for first/last frame timestamp to improve accuracy
Changed: when skipping checksum calculation for very large TCP payloads the next header type is now set to PAYLOAD to allow processing as undetermined payload
Changed: added preferences setting to allow disabling the download check
Changed: caption now declaring beta mode
Changed: displaying capture duration instead of interface count in file list, and file list size increased
Changed: scrolling every file node into view when active, not just when its really processed, to avoid sudden jumps
Added: linktype for IP Raw added
Added: added code to check and create the Ethernet FCS
Added: added code to handle Ethertypes for outdated/deprecated VLAN tags
Added: code to write a generic PCAPng header if the input file is not PCAPng
Added: code added to PCAPng loader to handle SysDig blocks
Added: TCP MultiPath options now being handled by TCP parser and assembler, which means that anonymization will work with those options
Added: functionality to export trace file header comment to descript.ion file (used by Total Commander, Take Command etc.)
Added: conversations are now scanned while scanning the file structure for general statistics
Added: setting IntelDB to invisible if parameter is given
Added: Functionality for address lookup from conversation storage now available for anonymization task settings
Added: replacement option to replace arbitrary strings in unknown payloads. The setting for this is still awkward to use, but it was a 15 minute implementation.
Added: option added to anonymization settings dialog to handle the Ethernet FCS
Added: button added to make it easy to clear the anonymization dialog settings
Added: button added to allow setting the current anonymization dialog settings as default
Added: button added to allow setting the form back to "Factory default"
Added: new Merge task added, which allows concatenation of multiple trace files, including those with more than one capture interface (which mergecap currently can't)
Added: Merge task allows filtering on certain packets so that the resulting merged file only contains frames matching the filters
Added: option added to set file timestamps of capture files to that of the input file, the first frame, or the last frame.
Added: trace file options added to main screen, which displays file statistics like capinfos
Added: SysDig block types added to PCAPng display
Added: functionality to remove tasks by popup
Added: code to handle the exit menu option
Added: code to help avoiding creating duplicate task names
Added: code to display conversation scan counters while scanning
Added: button to abort automatic and manual scans
Added: code added to toggle resizing upper or lower pane, can be set in preferences
Added: code to check filesize max amd warn if too big

– Alpha 0.1.3 build 320

Added: Added a new option to the editing task settings allowing the replacement of Linux Cooked headers with Ethernet headers
Fixed: Interface Description Block has IP and subnet values written without byteswapping them first (thanks ime-braun)
Fixed: There were multiple Memory leaks when handling frame content, editing/cutting frame content, and writing to disk.
Fixed: trace file formats other than PCAPng had a channel count of 0 in the file list
Fixed: editing tasks did not update the processing status bar in the file list while running

– Alpha 0.1.3 build 317

Fixed: The UDP checksum was set to "bad" values instead of keeping it at zero when it was zero in the original trace
Fixed: removing files from the file list didn't work when more than one file was selected
Fixed: writing Simple Packet Blocks in PCAPng files was using the wrong packet header (EPB)

Fixed: writing Interface Description Block with IPv4 option was broken, creating invalid files
Added: displaying MAC, IPv4 and IPv6 options of Interface Description Blocks in the PCAPng Structure Viewer

– Alpha 0.1.3 build 315

Fixed: Cutting GRE and GTP-U layers resulted in frames having a longer wiresize than capture size, because the wire length wasn't adjusted properly (Thanks to Chris Maynard)
Fixed: IPv4-in-IPv4 tunneling crashed on sanitization because I forgot to create the IPv4 pseudo header for the IP protocol type 4 (IPIP). Thanks to Chris Maynard for sending me the trace that crashed TW :-)
Fixed: forgot to handle optional checksum, key and sequence values in GRE parser and assembler
Fixed: Adding new tasks sometimes opened a form that held old settings. Now TraceWrangler has a folder with default settings for each kind of task and uses those files to initialize the forms (Thanks to Tony Fortunato)
Fixed: Cutting GRE and GTP-U layers failed when frames contained VLAN tags (Thanks to Herbert Grabmayer)

– Alpha 0.1.3 build 313

Fixed: Determining time bias for frame time stamp calculation crashed on systems with local timezone set to UTC (Thanks to George Gibat)
Fixed: Loading PCAP files failed for files with Link Layer Type set to Linux cooked capture (Thanks to Chris Maynard)
Fixed: Processing Edit tasks did not do anything for file formats other than PCAPng
Fixed: The Linklayer type wasn't handled correctly when reading file formats other than PCAPng, so the resulting PCAPng file would always have an ETHERNET linklayer (Thanks to Chris Maynard)
Fixed: the GRE parser calculated wrong offsets for the next parser when key bit or sequence number bit was set in the flags byte
Added: GTP-U removal now should be able to handle frames that have a Linux Cooked Frame header instead of an Ethernet header (could not test since I have no trace of that kind)

– Alpha 0.1.3 build 310

Fixed: DHCPv4 parser was skipped in release mode, so sanitizing DHCPv4 failed
Added: edit task has a new option to strip a GTP-U header from a frame, including the IP and UDP layer it is transfered by.

– Alpha 0.1.3 build 308

Fixed: Assembling packets containing IPv6 fragmentation headers crashed when trying to figure out the IPv6 pseudo header for CRC calculation
Fixed: Retrieving certain IPv4 addresses from the replacement table crashed when assigning an signed integer to an unsigned value. Similar issues were fixed for DHCPv4 options
Fixed: Anonymization task dialog UI problems fixed (Thanks to Tony Fortunato)
Fixed: Writing TCP SACK edge blocks had a constant TCP option length of 4. Now it is calculated correctly depending on the number of edges (Thanks to Stuart Kendrick)
Fixed: DHCPv4 option 15 (Domain Name) wasn't sanitized
Fixed: creating replacement FQDNs had a couple of flaws
Fixed: thousands separator for packet count in file list is now internationalized (Thanks to Tony Fortunato)
Fixed: UDP assembly didn't consider the size of payloads different than the unsanitized payload
Fixed: layers that had a different size after sanitization resulted in strange messages about truncated frames or load errors (Thanks to Tony Fortunato)
Fixed: PCAPng writing routines had a bug where options were written before the packet bytes, leading to corrupt files
Fixed: Parsing/Assembling RARP frames wasn't working (Thanks to Tony Fortunato)
Fixed: Parsing/Assembling 802.3 formatted Ethernet headers was broken (Thanks to Tony Fortunato)
Fixed: In some cases, remaining payload (a.k.a. bytes containing unkown protocol layers) wasn't handled correctly, leading to broken frames
Changed: the DHCPv4 parser zeroed the payload of all unknown options, which doesn't make much sense. Now unknown options are left out instead, so they will not make it into the sanitized frame
Changed: engine doing editing tasks completely rewritten

Added: handling DHCPv4 options 42 (NTP Servers), 44 (NetBIOS Nameservers), 46 (NetBIOS NodeType) and 120 (SIP Servers) (Thanks to Tony Fortunato for a sample trace)
Added: edit task can now be set to strip Juniper headers from frames in files written by Juniper devices. This will rewrite the link type, and transfer the packet direction to the metadata of the frame. (Thanks to Mike Canney for a sample trace)

– Alpha 0.1.3 build 294
Fixed: IPv6 did not read and write Version, Class and Flow Label correctly
Fixed: results of randomizing an IP address are no longer allowed to be 0.0.0.0 or in the range of 127.0.0.0/8
Fixed: IPv6 link local addresses will now stay link local addresses by forcing the prefix to be fe80::/64
Fixed: generating sanitized MAC addresses do no longer allow the all zero or broadcast address to be created, but after it has been processed by the address list
Fixed: storing/retrieving IPv4 addresses from database could crash for large 32 bit values because they weren't processed as unsigned values
Fixed: processing DHCPv4 host names wasn't working
Fixed: ICMPv4 did not sanitize the RedirectGwy for ICMP Redirect messages
Changed: FQDNs are now replaced with "x" characters as placeholders by default
Added: randomizing IPv6 addresses will keep the address in the according address range/class, e.g. Multicasts will stay Multicasts, private addresses will stay private, and public addresses will be in 2000::/3
Added: option to allow keeping IPv6 documentation addresses intact (prefix range of 2001:db8::/32)
Added: support for reading and writing MPLS added
Added: support for reading and writing GRE headers added
Added: support for ICMPv6 added to anonymization tasks
Added: option to keep IPv4 APIPA and Multicast addresses intact when randomizing IPs
Added: code to only randomize TCP/UDP ports for ports above 1024, if the setting for it is checked

– Alpha 0.1.2 build 281
Fixed: crash when parsing IPv6 packets containing UDP headers where IPv6 wasn't tunneled (thx, @packetlevel)
Fixed: status when processing files was finished wasn't correctly updated in the statusbar
Fixed: setting a subdirectory as output path failed
Fixed: routines calculating TCP and ICMP checksum were faulty, leading to bad CRCs in resulting files as well
Fixed: generating MAC addresses now checks for multicast and broadcast
Added: parser and assembler to handle the DHCPv4 layer
Added: anonymization task settings to allow clearing echo request/reply payloads
Added: storing FQDNs to replacement database
Added: parsing and writing Linux cooked headers
Added: generating new IPv4 addresses now checks if original addresses are Multicast or Broadcast. Also, generated addresses are now verified to be unique so that no two same replacements are made for different originals
Added: option to exclude IPv4 multicast addresses from randomization
Added: sanitity and debug code added to pinpoint problems in case trace samples cannot be provided
Changed: MAC address replacement routine now sets a default first octet of $F2 when the vendor octets aren't preserved
Changed: anonymization task now sanitizes the most important layer values by default

– Alpha 0.1.1 build 270
Fixed: protocol layers that failed to be parsed were not handled as generic payload correctly, leading to oversized frames that Wireshark can't read
Fixed: randomizing IP IDs wasn't working correctly for fragments since every fragment got a random ID. Now fragments belonging to the same original packet will have the same random ID.
Fixed: other minor issues were fixed
Added: parser/assembler for IPv6 fragmentation headers
Added: preference setting to check for update via http proxy
Added: preference setting to check for update on startup

– Alpha 0.1.1 build 263

Fixed: IPv4 fragments weren't handled correctly, which could lead to damaged anonymized frames because TraceWrangler did not realize that there were no further headers beyond the IP layer
Fixed: loading PCAPng files had a small problem with sanity checks for Interface Statistics Blocks
Fixed: ICMPv4 didn't handle Echo Request/Reply payloads correctly
Fixed: IPv4 Identification is now kept the same for fragments when randomization is configured. TraceWrangler will make sure that the IP ID stays unique per fragment group and roll the dice again in case of a collision with a former fragment group of the same IP pair.

-- Alpha 0.1.1 build 260
Fixed: ARP values weren't byte swapped on write
Fixed: loading settings could fail if the settings file had an older format
Changed: disabled Name Resolution Block sanitization options other than removing the block completely

-- Alpha 0.1.1 build 257
Fixed: setting frame.flags to 0 to avoid that the pcapng writer calculates block option lengths that aren't there, resulting in Wireshark refusing to load a file
Added: more sanity checking code when parsing block options

-- Alpha 0.1.1 build 255 --
Fixed: IPv6 traces crashed when trying to anonymize if not embedded in a IPv4 tunnel
Fixed: aborting a sanitization task didn't work
Fixed: file status is now set to fail when a task is aborted because of an error
Fixed: execute button was active when there was a task but no file
Added: hint texts in anonymization task editor are available for all layers now

-- Alpha 0.1.1 build 253 --
Fixed: nasty crash caused by copying "wiresize" number of bytes into a frame instead of "capture size", leading to memory corruption (thx, Landi)
Changed: TraceIntel database will not be used by default to avoid problems with processing different trace files with identical names

-- Alpha 0.1.1 build 251 --
Fixed: TCP checksum calculation was incorrect in some cases
Fixed: TCP header length could be wrong when options were written in a different way than in the original packet (thx, Landi)
Fixed: Preferences dialog would not include new advanced option strings when an older preferences file was loaded
Fixed: Timestamp calculation was incorrect in some cases, leading to out of order timestamps and the Wireshark TCP expert going crazy
Fixed: Trailing octets were not handled correctly in some cases, e.g. ICMP quotes of UDP packets
Added: Sanitization status is now updated visually in the trace file list while processing frames
Added: preference setting to write the ReplacementDB to disk for debugging purposes, instead of using a memory based DB that is discarded after the task has finished
Added: settings in anonymization task editor will now highlight the tree nodes on the left when a layer is set to modify something. This should make it easy to see what is going to be modified when the task is run.
Added: popup menu added for task list in main window; allows exporting and importing tasks
Added: drag & drop for ".task" files to add it to the list
Added: all files specified as a parameter when running TraceWrangler will be added as trace files except those ending on ".task", which will be added as tasks
Added: "About" dialog

Known Issues

TraceWrangler has some limitations at the moment (which may most likely last a little longer than just "a moment"):

- The maximum trace file size is less than 2GBytes per capture file added to the file list. This limitation is a result of the way the files are read by using Memory Mapped files. I'll work on this "when there is time"^{tm}.

- Capture files that contain truncated or damaged frames may not work under all circumstances. I recommend using captures that contain full sized frames. Reason for this is that I may have been lazy in some parts of the code where I do sanity checks against the length of the data available for processing, which is a great way to run some pointers into the great beyond. Ouch.
- IPv6 checksums may not be calculated correctly when extension headers (including fragmentation headers) are used in a frame.