SIP Scenario Generator White Paper Part 2

Detailed Output Description and Theory of Operation

By: Ray Elliott
Email: ray.elliott@ipc.com
Web: http://www.ipc.com

Generated Files

The basic structure of the output information is split into four sections:

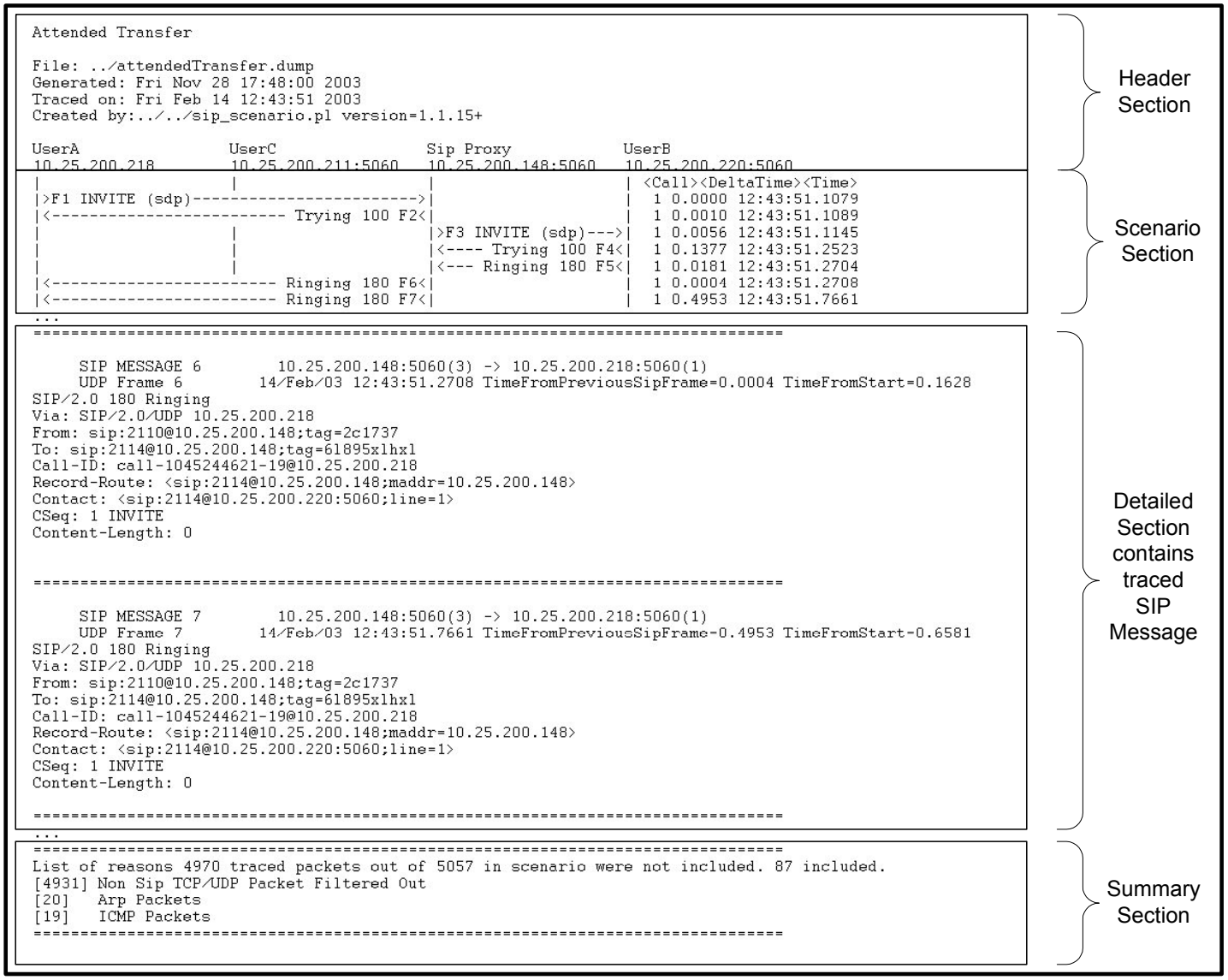
Header

SIP Scenario Body

SIP Message Details

Summary Information

Here is an example of the information.



The title in the header can be specified.

The summary section can be suppressed by a command option.

Related Command Arguments

-t[itile]:TITLE	Defines the Title in the header Section -title:Attended Transfer
-stat:Enable=1/Disable=0	Enables/Disable Statistics in the summary section -stat:1

There are three different formats produced two are html based and one is a text format. All three formats are generated by default.

Triple Frame HTML File

Single HTML File

Plain Text File

Triple Frame Html Files

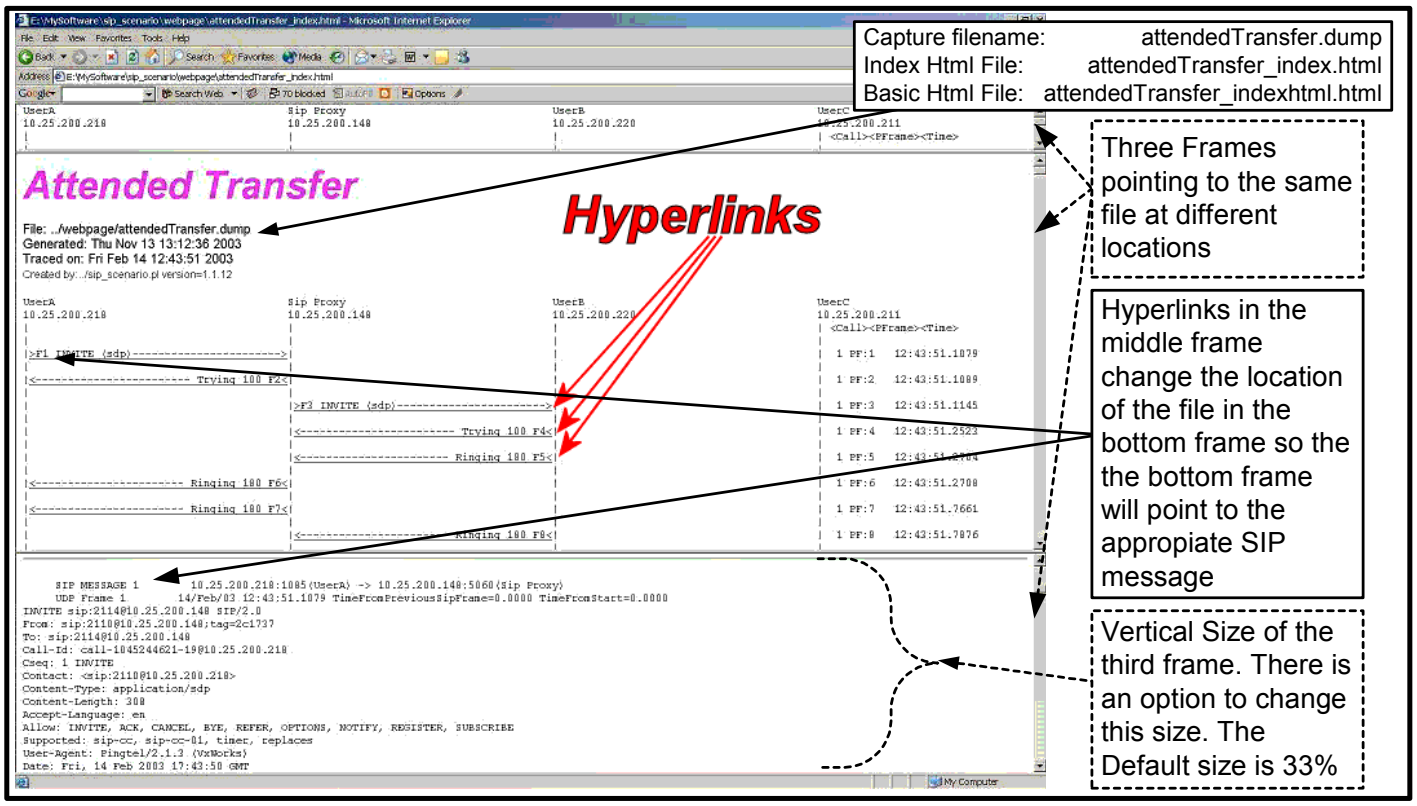
The triple frame html package contains two files.

- The index html file was design as the target URL to be used by the browser. The index html file creates three frames, header (top) frame, scenario (middle) frame, and detail message (bottom) frame, where each frame references the same file, but at different locations. The default size of the third frame can be changed. The default size is 33%.
  "_index.html" is appended to the filename to create the index filename.

  Related Command Arguments

-percent:DefaultSize	Defines the default Spacing for the bottom HTML frame in percent. eg. –percent:33 for 33%.
-keep:bitValue	Bit 0 (value 1) Enables Triple Frame Output

- The basic html file is to be called from the index html file and contains all the SIP Scenario information about the capture file. The hyperlinks from the sip scenario section references the bottom frame (by target="bottom"). This results in clicking the hyperlinks in middle frame changes the location of the file in the bottom frame.
  Typically this file should have been in a subdirectory, but it was decided that creating subdirectories in user directories was not desired. If the basic html file is directly used by the browser (instead of the index file), then the hyperlinks will target a frame that is not defined. When the hyperlinks are clicked then sometimes no action take places or sometimes, the browser will create a new window where the SIP messages will be displayed.
  "_indexhtml.html" is appended to the filename to create the basic html filename.


Capture filename: attendedTransfer.dump
Index Html File: attendedTransfer_index.html
Basic Html File: attendedTransfer_indexhtml.html
Three Frames pointing to the same file at different locations
Hyperlinks in the middle frame change the location of the file in the bottom frame so the the bottom frame will point to the appropiate SIP message
Vertical Size of the third frame. There is an option to change this size. The Default size is 33%
Attended Transfer
Hyperlinks
File: ../webpage/attendedTransfer.dump
Generated: Thu Nov 13 13:12:36 2003
Traced on: Fri Feb 14 12:43:51 2003
UserA
Sip Proxy
UserB
UserC

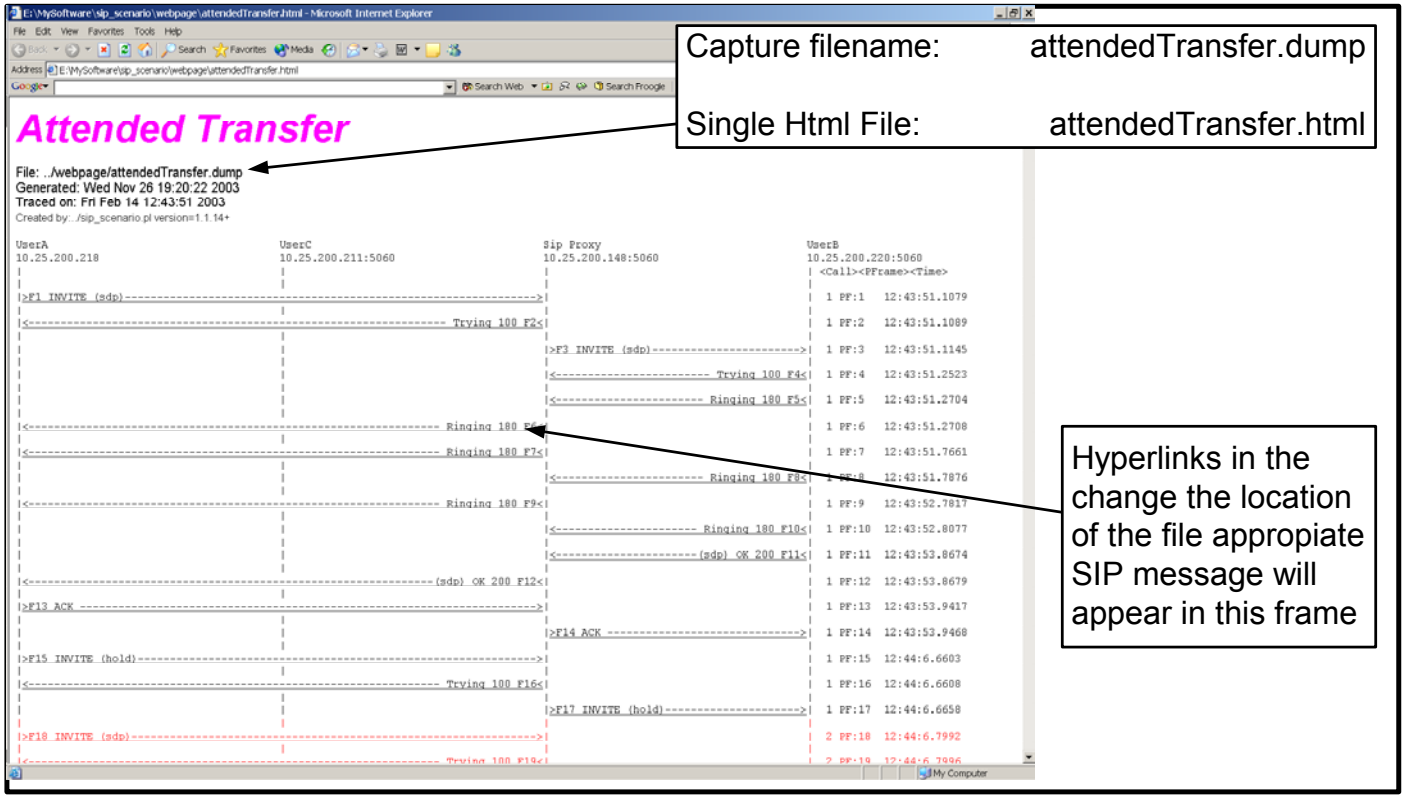
Single Html File

The single html file was design as the target URL to be used by the browser and is identical to basic html file with the exception that the references to the bottom frame (target="bottom") are removed.

".html" is appended to the filename for the single filename.

Related Command Arguments

-keep:bitValue — Bit 1 (value 2) Enables Single HTML Output



Plain Html File

The Plain text file is generated by removing all the html syntax from the file. The following is an extract from the plain text file.

".txt" is appended to the filename to create the plain text filename.

Related Command Arguments

-keep:bitValue

Bit 2 (value 4) Enables Plain Text Output

Bit 0 (value 1) Enables Triple Frame Output

Bit 1 (value 2) Enables Single HTML Output

Oring bit values together define the set of file that are generated. The default value of 7 generates all three output types.

```
Attended Transfer

File: ../webpage/attendedTransfer.dump
Generated: Wed Nov 26 19:20:22 2003
Traced on: Fri Feb 14 12:43:51 2003
Created by:../sip_scenario.pl version=1.1.14

UserA                          UserC                          Sip Proxy                      UserB
10.25.200.218                  10.25.200.211:5060             10.25.200.148:5060             10.25.200.220:5060
|                              |                              |                              | <Call><PFrame><Time>
|                              |                              |                              |
|>F1 INVITE (sdp)--------------------------------------------->|                              |  1 PF:1   12:43:51.1079
|                              |                              |                              |
|<---------------------------------------------- Trying 100 F2<|                              |  1 PF:2   12:43:51.1089
|                              |                              |                              |
|                              |                              |>F3 INVITE (sdp)------------->|  1 PF:3   12:43:51.1145
|                              |                              |                              |
|                              |                              |<------------ Trying 100 F4<|  1 PF:4   12:43:51.2523
|                              |                              |                              |
|                              |                              |<------------ Ringing 180 F5<|  1 PF:5   12:43:51.2704
|                              |                              |                              |
|<---------------------------------------------- Ringing 180 F6<|                              |  1 PF:6   12:43:51.2708
|                              |                              |                              |
|<---------------------------------------------- Ringing 180 F7<|                              |  1 PF:7   12:43:51.7661
|                              |                              |                              |
|                              |                              |<------------ Ringing 180 F8<|  1 PF:8   12:43:51.7876
|                              |                              |                              |
|<---------------------------------------------- Ringing 180 F9<|                              |  1 PF:9   12:43:52.7817
|                              |                              |                              |
```

UA Representation

The basic idea was that each IP Address represented a SIP UA, whether it was a proxy, gateway, or other SIP devices. Each column represents a SIP UA, at an IP address. The column will be label by the respective IP address. In addition to an IP address, an alias may be assigned to that IP address. This alias will also be displayed as a part of the column header.

Each is Sip message is indicated by arrow between two columns, a SIP UAC and a SIP UAS. The arrow will have a descriptor that contains the SIP request or the SIP response. Other information can be added to the description field such as SDP (indicating that an SDP was attached) or hold (if an attached SDP indicates hold). The length of the description field should fit between the two columns. If the description does not fit between the two columns then a warning message will indicate what the spacing (gap) between the columns should be so that the description field will fit.

Multiple SIP UAs on the same IP address

There have been several different SIP configurations that have multiple SIP UAs at a single IP address. This same situation occurs when SIP phones are on one side of a firewall and the proxy is on the other side. SIP message are traced at the SIP proxy. The NAT part of the firewall changed the IP address and ports numbers so that the two phone where at the same IP address, but at a different port. If SIP messages for different UAs at the same IP address can be correctly identified with the correct SIP UA, then this SIP UA will have a column identified by an IP address and received port number. An alias can also be associated with an IP address and port number.

Many SIP UAs, when sending UDP packets, do not send the UDP packet from the same port where UDP packet received. Typically all UDP packets are received on port 5060. Many UAs do not send UDP packets on port 5060 for some reason or another. It becomes very difficult when looking a packet trace to reliably associate the port number where are sent with the port number where packets are received. This also holds true for TCP packets.

Symmetric UDP Port Detection

If UDP packets are sent using the same port number (symmetric UDP ports) for a given UA, then there is a way to reliably determine the receive port related to a send SIP message – they are the same number.

The detection mechanism is very easy and conservative. A more extensive mechanism is possible, but the benefits are questionable. The rules are simple:

- All SIP MSGs from the same UA are sent on the same send/receive UDP ports.
- All SIP MSGs from all UAs are sent on UDP ports.

Or in other terms

There cannot exist

- A TCP packet to the IP address
- A UDP packet transmitted from a port, where the port has not received a UDP packet.

When a symmetric UDP port has been detected, then the column heading description will contain the IP address and the port number.

Symmetric UDP port identification can be suppressed globally or by IP address.

Related Command Arguments

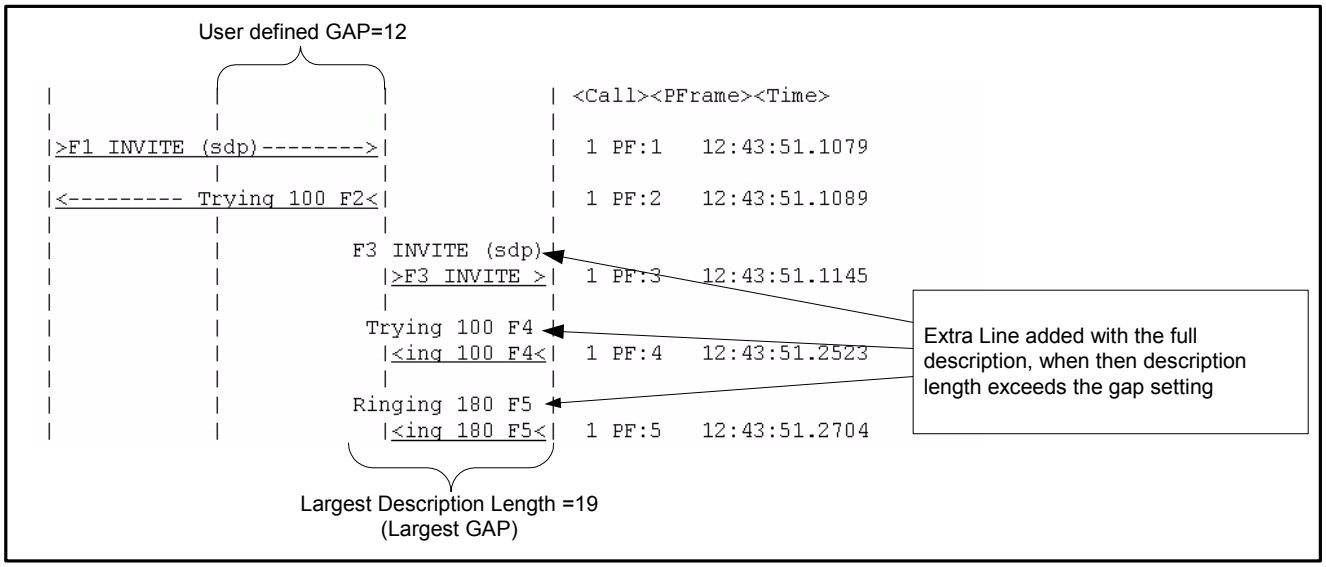
IPADDRESS[:PORT][/ALIAS][:COLUMN][:SINGLEUA] where

IPADDRESS	An Ipv4 address in dotted notation. eg 192.168.10.3
PORT	The port number for the UA when there are more than one UA at an IP address.
ALIAS	Defines an ALIAS of the IP address that appears at the column heading.
COLUMN	Defines which column for the UA. Column numbers should be used in an increasing sequence.
SINGLEUA	Disable Multiple UA detection for that IP address
192.168.2.3/gateway:1:singleua	Forces a single column for ip address 192.168.2.3. Assigns alias “gateway” to IP address 192.168.2.3. which is forced into column 1.
192.168.2.4:5062/ATMgateway:2	Assigns alias “ATMgateway” to the UA at IP address 192.168.2.4. at port 5062 which is forced into column 2.
192.168.2.4:5060/PSTNgateway:3	Assigns alias “PSTNgateway” to the UA at IP address 192.168.2.4. at port 5060 which is forced into column 3.
-singleua	Disables Multiple UA detection for all IP addresses

Horizantile Spacing

The horizontal spacing (distance between vertical columns) has a default size of 18 characters. The maximum SIP message description length is calculated. If the maximum SIP message description length is greater than the default setting then the default setting is changed to the maximum SIP message description length and the SIP Scenario Diagram is regenerated.

If the gap is specified by the user and is less than a SIP message description length then an extra line containing the complete SIP message description is added. The adding of extra lines can be suppressed by a command.



Related Command Arguments

-g[ap]:gapSize	defines the horizontal spacing between columns
-de[scription]:Enable/Disable	Enable=1,Disable=0 Add Commplete Sip Message Description

Vertical Spacing

There are four vertical spacing options that are based on turning on or off vertical spacing features, which are:

- Bit 1: Arrows and the description are on the same line or not.
- Bit 0: Add an extra spacing line or not.

The default option is to have the arrow line and the description in the same line and to add an extra spacing line.

Related Command Arguments

-f[ormat]:v[ertical]:bit_value	Defines vertical spacing

The following diagrams illustrate the four different vertical spacing options. Each diagram contains 14 lines.

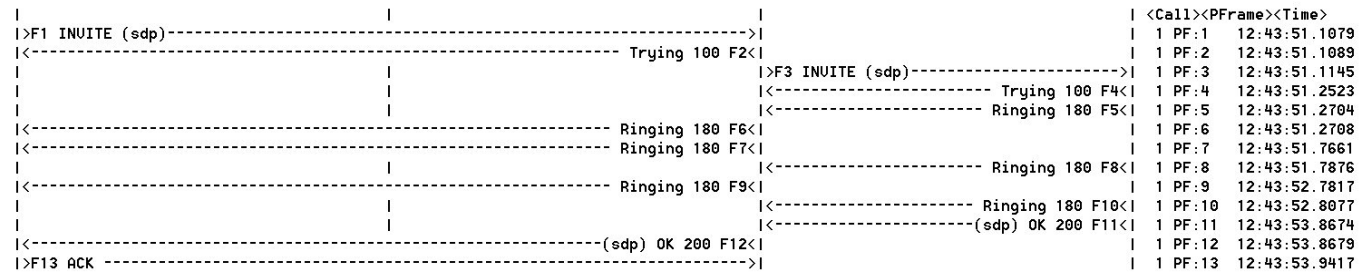


Figure 1: Vertical Compression mode 0

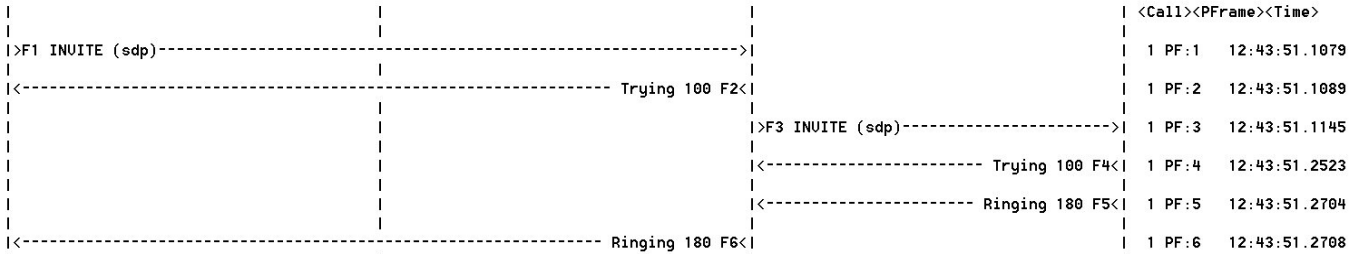


Figure 2: Default - Vertical Compression mode 1

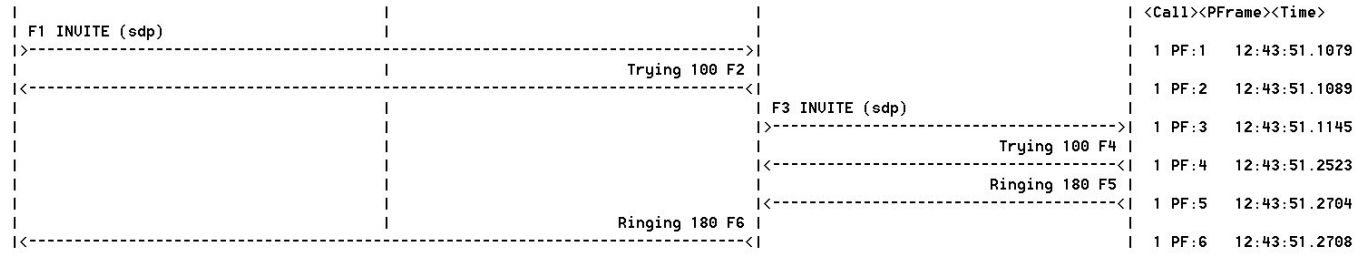


Figure 3: Vertical Compression mode 2

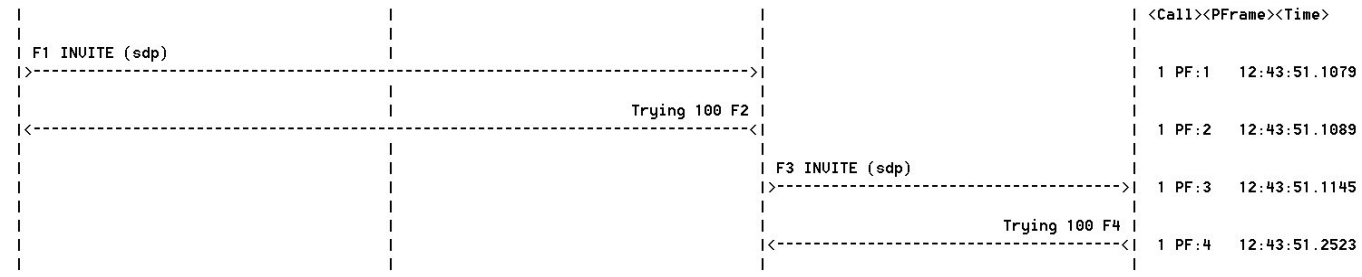


Figure 4: Vertical Compression mode 3

Packet Information

After each SIP scenario arrow line, information can be appended. This information includes

- Call Number
- Physical Frame Number
- Date
- Time
- Relative Time (from first packet)
- Delta Time (from previous packet)

Frame Number

opt#	Description
0	Not Display
1	Display (default)

Time: There are four different times that can be independantly display or not displayed. Each options is represented by a bit. Zero turns off the time and a one turns on the time. The sum of all the bit values produces the time display option. e.g. 15 displays all times while a 5 displays date and Delta time.

bit #	Value	Description
0	1	Delta Time from the previous frame
1	2	Relative Time from the first frame
2	4	Date
3	8	Local Time of Day

CALL Number

opt#	Description
0	Not Display
1	Display (default)
2	Full Display with prefix (call#:)

```
|                      |                      |                      | <Call><PFrame><DeltaTime><RelTime><Date><Time>
|>F1 INVITE (sdp)------------------------->|                      |  1 PF:1   0.0000 0.0000 14/Feb/03 12:43:51.1079
|<------------------------ Trying 100 F2<|                      |  1 PF:2   0.0010 0.0010 14/Feb/03 12:43:51.1089
|                      |                      |>F3 INVITE (sdp)--->|  1 PF:3   0.0056 0.0066 14/Feb/03 12:43:51.1145
|                      |                      |<---- Trying 100 F4<|  1 PF:4   0.1377 0.1443 14/Feb/03 12:43:51.2523
|                      |                      |<--- Ringing 180 F5<|  1 PF:5   0.0181 0.1624 14/Feb/03 12:43:51.2704
|<------------------------ Ringing 180 F6<|                      |  1 PF:6   0.0004 0.1628 14/Feb/03 12:43:51.2708
|<------------------------ Ringing 180 F7<|                      |  1 PF:7   0.4953 0.6581 14/Feb/03 12:43:51.7661
|                      |                      |<--- Ringing 180 F8<|  1 PF:8   0.0215 0.6797 14/Feb/03 12:43:51.7876
|<------------------------ Ringing 180 F9<|                      |  1 PF:9   0.9941 1.6737 14/Feb/03 12:43:52.7817
|                      |                      |<-- Ringing 180 F10<|  1 PF:10  0.0260 1.6997 14/Feb/03 12:43:52.8077
|                      |                      |<--(sdp) OK 200 F11<|  1 PF:11  1.0597 2.7595 14/Feb/03 12:43:53.8674
|<-----------------------(sdp) OK 200 F12<|                      |  1 PF:12  0.0005 2.7600 14/Feb/03 12:43:53.8679
|>F13 ACK ------------------------------->|                      |  1 PF:13  0.0738 2.8338 14/Feb/03 12:43:53.9417
|                      |                      |>F14 ACK ---------->|  1 PF:14  0.0051 2.8388 14/Feb/03 12:43:53.9468
|>F15 INVITE (hold)----------------------->|                      |  1 PF:15  12.7135 15.5524 14/Feb/03 12:44:6.6603
|<------------------------ Trying 100 F16<|                      |  1 PF:16  0.0005 15.5528 14/Feb/03 12:44:6.6608
|                      |                      |>F17 INVITE (hold)->|  1 PF:17  0.0050 15.5578 14/Feb/03 12:44:6.6658
```

OR

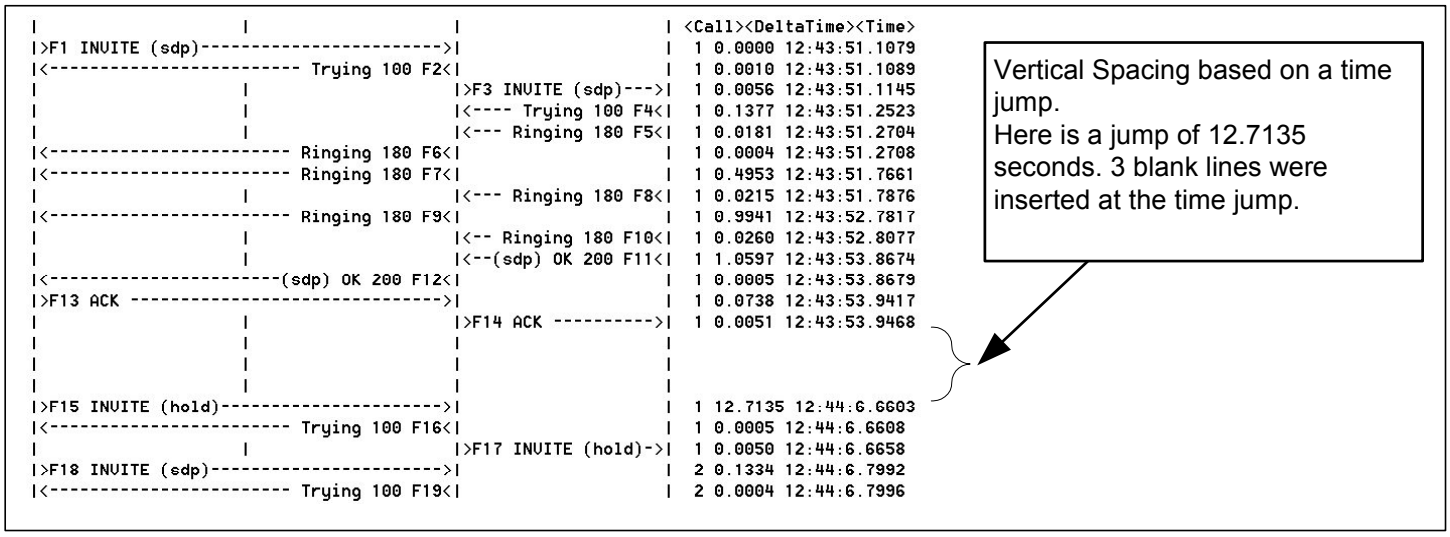
```
|                      |                      |                      | <Call><DeltaTime><Time>
|>F1 INVITE (sdp)------------------------->|                      |  Call#:1 0.0000 12:43:51.1079
|<------------------------ Trying 100 F2<|                      |  Call#:1 0.0010 12:43:51.1089
|                      |                      |>F3 INVITE (sdp)--->|  Call#:1 0.0056 12:43:51.1145
|                      |                      |<---- Trying 100 F4<|  Call#:1 0.1377 12:43:51.2523
|                      |                      |<--- Ringing 180 F5<|  Call#:1 0.0181 12:43:51.2704
|<------------------------ Ringing 180 F6<|                      |  Call#:1 0.0004 12:43:51.2708
|<------------------------ Ringing 180 F7<|                      |  Call#:1 0.4953 12:43:51.7661
|                      |                      |<--- Ringing 180 F8<|  Call#:1 0.0215 12:43:51.7876
|<------------------------ Ringing 180 F9<|                      |  Call#:1 0.9941 12:43:52.7817
|                      |                      |<-- Ringing 180 F10<|  Call#:1 0.0260 12:43:52.8077
```

Related Command Arguments

-f[ormat]:c[allid]:Full=2/On=1/Off=0	Defines call number formats
-f[ormat]:t[time]:BitValue	Enables Different time displays
-f[ormat]:p[hy]:Enable=1/Disable=0	Enables/Disables display of physical frame numbers

Vertical Spacing for a Time Jump

When there is a large amount of time between successive SIP messages, extra blank lines can be inserted. This option generates a vertical spacing when there is a jump in time. The amount of time and the number of blank lines inserted are defined by parameters to this display option. This option can make debugging easier and is turned off by default.



Related Command Arguments

-f[ormat]:s[pacetime]:Seconds/NumLines	Adds NumLines Extra Vertical Lines for a time jump of Seconds

Summary list

There is a special option that display's the list of IP address and aliases that are used by the scenario. This option displays the summary information to the display (Standard Output) or to the specified output file.

Related Command Arguments

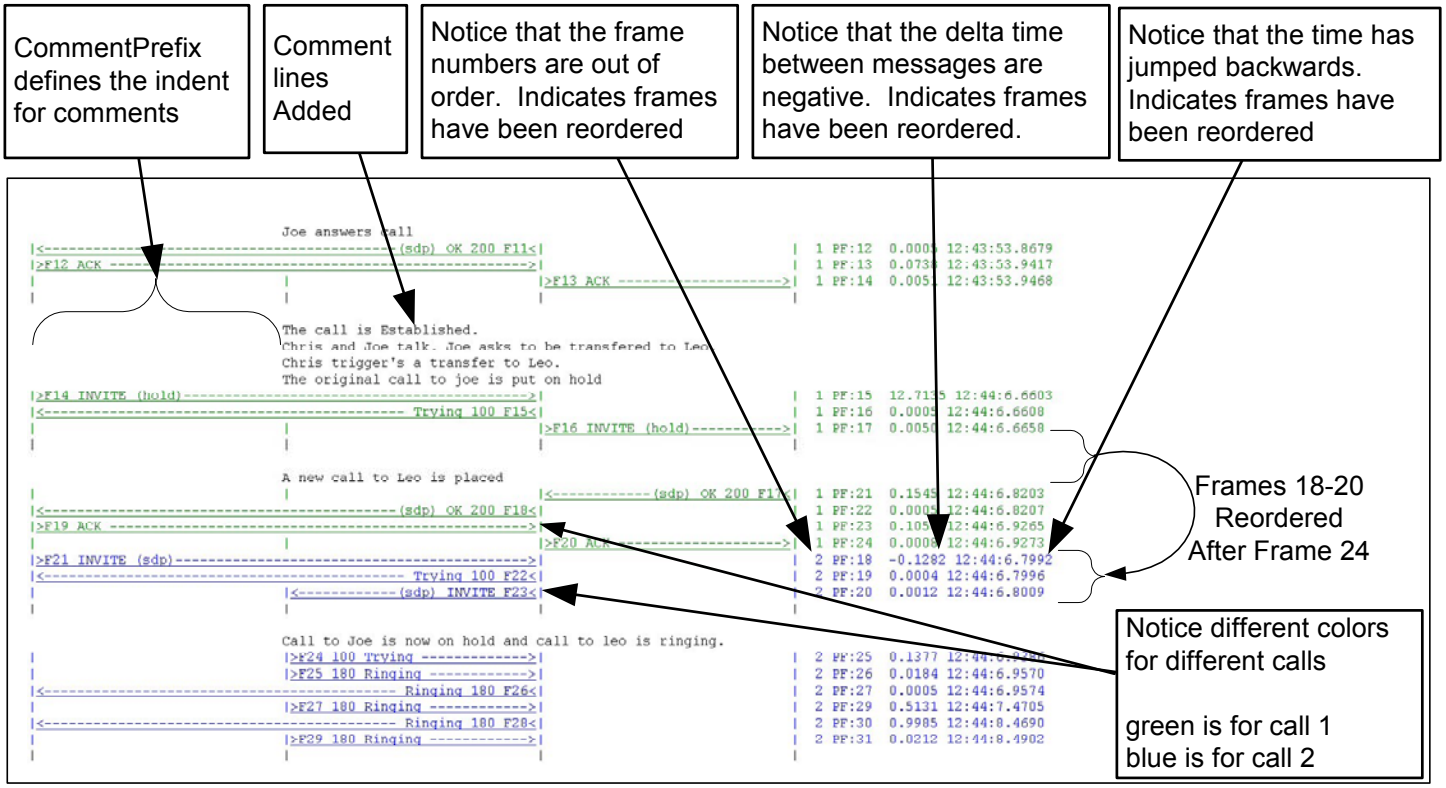
-summary[:FILENAME]	Displays a list of IP addresses and aliases defined. The output can be directed to a file.

Documentation Commands

There are commands that make the output look better and make it easier to understand.

There are commands to reorder (move) SIP messages within the file. Comment can also be added anywhere.

The colors that are used to indicate different calls, can be changed to any colors used by html.



Related Command Arguments

-re[order]:FRAMELIST:FRAME	Physical frames defined by FRAMELIST are displayed after FRAME. FRAMELIST is a comma separate list of ranges of physical frame numbers or individual physical frame numbers. eg. 1-5,2,4,10-12,3
-reorder:18-20:24	Displays frames 18-20 after frame 24.
-c[omment]:FRAME:COMMENT	Inserts the COMMENT after FRAME.
-comment:11:Joe Answers Call	Adds comment “Joe Answers Call” after frame 11.
-c[ommentprefix]:STRING:	pre-appends STRING before every comment. This is used to define the indent for comments. Typically the string is just a set of spaces.
-c:1234567890:	pre-appends 1234567890 before every comment.
-color:COLORLIST	A comma separated list of colors. Each color must be in HTML format for color.
COLORLIST	The set of html colors is from the following list Black,Green,Silver,Lime,Gray,Olive,White,Yello

	w,Maroon,Navy,Red,Blue,Purple,Teal,Fuchsia, Aqua or a color can be a user defined color in the form #RRGGBB where RR,GG,BB are each two hex digits e.g. #FF0080
-color:black,red,green,#FF2581	Assign the set of colors to be used for calls.

Call Filters

Call Filters reduce the quantity of ethernet frames by either including or excluding all the ethernet packets for a call. A include flag is maintained for each call (by unique callid) and has three values include, exclude and undefined. The default value for the include flag is undefined. Include call filters set the include flag to include. Exclude call filters set the include flag to exclude. Once the flag has set to exclude, it will not be changed.

When there exists an include call filter then the undefined value means exclude, else the undefined value means include.

- Call Number.
  Includes / Excludes a set of calls by the call number.
  This call filter is the first call filter to execute. It executes during the parsing of the command arguments. For each call referenced by the command argument, the respective include flag is set to either include or exclude.

Dynamic Call Filters

During the parsing of each SIP Message, dynamic call filters are executed. These filter either match or do not match the filter condition. If the filter matches the filter condition for any SIP message then the filter is applied. When the filter is applied then the include flag is set to include (for include filters) and to exclude (for exclude filters).

There can be multiple include filters and multiple exclude filters.

Each filter acts independently of other filters, resulting in include filter conditions being “ored” together and exclude filter conditions being “ored” together.

“Not” Option

There is a “not” option to dynamic filters. When a normal filter matches any SIP message then the filter is applied. The not option does the opposite, when the filter matches no SIP message in the SIP call, then the filter is applied. An include filter for condition A is the same as an exclude “not” filter for condition A. This becomes important when include filters need to be “anded” together so that include CONDTION A and include CONDITION B can be “anded” together instead “ored” together. So CONDITON A and CONDITION B can be achieved by

An Include CONDITION A statement and an exclude not CONDITION B statement.

- IP Address.
  Includes / Excludes any call that has the specified IP address in the IP header in any SIP message that is a part of the call.
  Each IP address filter has a value of either include or exclude.
  If the source IP address matches an IP address filter then that filter is applied to the respective

call.
If the destination IP address matches an IP address filter then that filter is applied to the respective call.

- SIP Request
  Includes/Excludes any call, where some SIP message has the SIP Request in the first header that matches
- Regular Expression Matching.
  Includes/Excludes any call, where some SIP message in that call has a header that matches the specified (Perl) regular expression.
  Each Perl Regular Expression filter has a value of either include or exclude.

  Each header in a SIP Message (not attached message bodies, like SDP) is matched against the Perl Regular Expression. If a match is found then the filter is applied to the respective call. All Regular Expression filters are executed in order that they are defined.

  Perl Expression matching is case insensitive.
  This filter could find all calls to me by
  EXPRESSION= "^INVITE.*ray.elliott@ipc.com" or
  EXPRESSION= "^to.*ray.elliott@ipc.com"
  or find REGISTER requests by
  EXPRESSION= "^REGISTER"

  See Perl documentation for details of matching and perl regular expressions
  http://www.perldoc.com/perl5.6/pod/perlre.html
  Here are some definitions of the Perl metacharacter characters.

^	Matches start of line
$	Matches end of line
|	Logical Or
\w	Match a "word" character (alphanumeric plus "_")
\W	Match a non-word character
\s	Match a whitespace character
\S	Match a non-whitespace character
\d	Match a digit character
\D	Match a non-digit character
\X	escapes next character. same as X e.g. \\ evaluates to \
\b	Matches a word boundary
.	Matches any character.
+	Preceding character or expression must occur one or more times
*	Preceding character or expression must occur zero or more times
?	Preceding character or expression must occur zero or one times
{N,M}	Preceding character or expression must occur at Least N times but not more than M times
()	Grouping
(?i)	Add to start of String if case-insensitive matching is desired.
\s*	Matches any white space string. Empty allowed.
\s+	Matches any white space string. Must have some white space.

\w+ Matches a word

Related Command Arguments

-I[nclude]:c[allid]:LIST	Calls defined by LIST are displayed. LIST is a comma separate list of ranges of call numbers or individual call numbers. eg. 1-5,2,4,10-12,3
-e[xclude]:c[allid]:LIST	Calls defined by LIST are not displayed. LIST is a comma separate list of ranges of call numbers or individual call numbers. eg. 1-5,2,4,10-12,3
-I[nclude][:not_option]:ip:IPADDRESS	Includes any call that has either the source of destination IP ADDRESS in the IP header equal to the specified IPADDRESS for some SIP message in that call.
-e[xclude] [:ip_options]:ip:IPADDRESS	Excludes any call that has either the source of destination IP ADDRESS in the IP header equal to the specified IPADDRESS for some SIP message in that call.
not_option := ! | not	the not_option is used to negate the matching condition. For example, -include:not:ip:192.168.10.11 will include any call that does not send/receive some SIP message to ip address 192.168.10.11. This seems to be identical to -exclude:ip:192.168.10.11. and not required. The difference becomes a necessity when trying to include SIP calls message to “ray.Elliott” that only contain “REFER request” Sip messages. Two includes would “or” these conditions together rather that “anding” them together. So to achieve the required results Include SIP messages to “Ray.Elliott” and exclude SIP calls that do not contain the REFER request.
-I[nclude][:not_option]:req[uest]:REQUEST	Includes any call that has some SIP Message with the SIP request matching REQUEST.
-e[xclude][:not_option]:req[uest]:REQUEST	Excludes any call that has some SIP Message with the SIP request matching REQUEST.
-I[nclude] [:exp_options]:m[atch]:STRING	Same as include:expression:PERLPATTERN.
-i[nclude] [:exp_options]:e[xpression]:PERLPATTERN	Includes any call that has a SIP message that has some header that matches the PERLPATTERN using Perl Pattern Matching.
-e[xclude] [:exp_options]:e[xpression]:PERLPATTERN	Excludes any call that has a SIP message that has some header that matches the PERLPATTERN using Perl Pattern Matching.

exp_option := ic | noic | not_option	The ic and noic option control case sensitive pattern matching. “ic” is ignore case and provides case in-sensitive matching. “noic” is no ignore case and provides case sensitive matching. The default is “ic” providing case in-sensitive matching.
exp_options := exp_option | [:exp_options]	Set of options for Perl expression pattern matching.
-e:request:REGISTER	Excludes any REGISTER requests transaction (calls).
-I:ip:192.168.2.3	Includes any call where some SIP message is sent or received by a UA at 192.168.2.3
-I:called:1-10	Include the first 10 calls.

Merge Utility

The merge utility merges two capture file together. It is assumed that the capture files have an overlapping period of time and at least one UDP SIP message in common. Typically, the capture files are generated at different point in the network at approximately the same time.

No NAT devices

There must NOT be a NAT device in the network between the two capture locations. The NAT device will change the SIP packets (especially the IP header) and the Merge utility will not find common sip messages, since the IP header is different.

The merge utility keeps only TCP packets and SIP UDP packets. All other captured packets will be discarded. The two capture files are assumed to contain some common SIP messages. This common SIP message is used to synchronize time stamps between the two capture files.

The first capture file is be the master capture file.

The master file is read into memory, discarding the appropriate packets and keeping track of identical UDP SIP messages. For example, SIP 180 ringing messages for a given call are all identical with the exception that they are sent at different times.

The second file is then is read into memory, discarding the appropriate packets. The first UDP SIP message that is identical to a unique UDP SIP message in the first file is used to establish the time synchronization. When successive UDP SIP messages match identically to a unique USP SIP message in the first file then time is re-synchronized.

When a UDP SIP message from the 2nd file matches identically to a SIP message in the first file within +- .35 seconds then the UDP SIP message from the 2nd file is discarded, since there is a duplicate SIP meesage in the 1st file.

TCP packets are just mergedfs, updating the timestamp. The TCP implementation will then discard duplicate TCP messages.

Related Command Arguments

-Merge[:FILENAME]	Defines a second input capture file. The second file is merged into the first file. The output with have the base output file name appended with “.new.dump”

Theory of Operation

Argument Processing

SIP Scenario first parses all the command arguments. Some of the arguments are stored in an array and are processed later on.

Include Files

Any command option (switch) can be place in a include file, (or multiple include files). There is a command that includes an include file. There is a default include file called sip_scenario.ini. This file is automatically included when the program is started before any other command arguments are parsed. Each line in the file is a command argument. Blank lines and lines that start with white space "#" or just "#" are ignored.

The sip_scenario.ini can be setup to contain your personal defaults.

Related Command Arguments

-I[nclude][:skip]:INCLUDEFILE	Specifies a file that should be included. Skip specifies the number of line to ignore at the beginning of the include file.

Handling the Capture File

Next the libpcap header file is read, determining which version of the libpcap is being used. There is a version for big-endian vs little-endian, a version for ethernet, and versions for other transport mediums. SIP Scenario will handle big-endian and little-endian versions, but SIP Scenario will only handle ethernet transport medium.

The libpcap formatted has a header for each captured packet. This header contains 3 pieces of information:

- The time of day (seconds and micro seconds) GMT that the packet was traced.
- The length of the frame capture from the transport medium (frame length)
- The length of the captured data stored in the file (capture length).

There are two types of errors that can occur that are based on the capture length.

1. There is not enough data stored to process the next header.
2. Capture Length is not the same as the frame length for SIP messages

The time of day (local time) for the first packet will be displayed in the output header as the "Traced On" time.

The frame (ethernet packet and all its associated information) is stored in a "Perl Hash" array, which emulates a "C" structure. We will call this structure the "Packet Information Structure".

Statistics are kept each time a frame is eliminated from being processed. These statistics are normally displayed at the end of processing. The number of packets eliminated for a specific reason will be displayed.

There are two include filter that are applied.

The first filter includes frames based on the frame number. A set of ranges (pairs of numbers consisting of a start frame numbers and a end frame numbers) is sorted and defines the set of frames to include.

The next filter is the time filter. There is a single start time and a single end time. Frames outside the start time and end time will be discarded.

Related Command Arguments

-range:RANGELIST	Specifies the set of frames that will be included.
RANGELIST	A comma separate list of ranges of physical frame numbers or individual physical frame numbers. eg. 1-5,2,4,10-12,3 There can be multiple –range arguments each having their own rangelist.
-I[nclude]:l[ine]:RANGELIST	Alternate form of the range argument
-I[nclude][:t[ime]]: STARTTIME-ENDTIME	Specifies the time range (start time and end time) to include capture frames. Frames outside this range will be discarded. There can be only a single –include:time argument.
TIME:= [[[YEAR/]MM/]DD/]HH:MM	Defines Local time Where
Year	Number. Eq 2003.
MM	Month. Range 1-12
DD	Day of Month. 1-31
HH	Hour. 0-23
MM	Minute. 0-59.
STARTTIME:= TIME	Defines STARTTIME
ENDTIME:= TIME	Defines ENDTIME Default values for Year,MM (month), and DD (Day) are the same as the first packet traced. StartTime must be before EndTime.

Parsing a Captured Frame

The frame is then parsed for ethernet information, which is stored in the Packet Information Structure.

Protocol filtering takes place next. All frames, which are not TCP or UDP, are eliminated from further processing. When a packet is eliminated by protocol filtering, the description will indicate the protocol name (if known).

Next the frame is parsed by either the TCP or UDP parser, which stores parsed information into the Packet Information Structure. There is an option that allows Kerberos packets to be parsed.

Kerberos

During testing of Kerberos, I added the display of Kerberos messages mixed with SIP messages. Kerberos messages will be display when the Kerberos option is turned on. The Kerberos is turned off by default.

Related Command Arguments

-kerberos:Enable=1/Disable=0	Enables/Disables Expansion of Kerberos packets.

TCP Parsing

The transmit IP address, the transmit TCP port, the receive IP address and the receive TCP port together from a unique TCP channel ID. TCP packets are queued onto the TCP channel packet queue. The TCP packet headers are parsed to find the TCP sequence numbers. The TCP sequence numbers are the basis for TCP operation. The sequence numbers identify the position of the transmitted/received data in the byte stream. If data has been received, that has already been accepted, then it will be discarded. If there is missed data, then that data will be queue until the missed data is received. When the correct data is available then it is accepted and passed on. This mechanism follows the same basic principles as a TCP stack.

SIP Message Detection

SIP messages are detected by having the first line in the packet to be in one of two formats, where

- X is a set of decimal digits
- DESCRIPTION is a set of displayable ASCII characters.
- REQUEST is a set of alpha-numeric character
- RESPONSE is a set of decimal digits

SIPX.X RESPONSE DESCRIPTION

REQUEST DESCRIPTION SIPX.X

SIP Message Parsing

Sip Message parsing is based on stream processing for TCP. First the SIP message is found by searching for cr lf cr lf. When the SIP message is found then the SIP message is parsed.

The first line is parsed for request / response values. The CallId header value is checked if has already been used. If not then a call number is assigned to the called Header value.

The called, the call number, request /response values are stored in the Packet Information Structure.

If there is a content length with a non-zero value then the attached message body is extracted.

End of File Processing

When the input file finished parsing the last message, certain actions must be taken. TCP message queue must be flushed to insure that SIP message are not lost.

Next certain command arguments that were put into the delayed execution list are now executed.

Next, SIP message can be included/excluded or reordered. Also. At this time, comment records are added as special “SIP messages”.