Trace Parser Tool

Table of Contents

Restcomm jSS7 Trace Parser is a simple tool that can parse trace capture data files (of some formats) and log its content in some log file in a text form. This tool can be very handy to debug the flow of SS7 messages.

Running Restcomm jSS7 Trace Parser

The Restcomm jSS7 Stack comes with a Trace Parser module. You can install and run the module on any machine.

1. Change the working directory to the bin folder in the Simulator’s installation directory.

   ```
   downloads]$ cd mobicents-jss7-<version>/ss7/mobicents-ss7-traceparser/bin
   ```

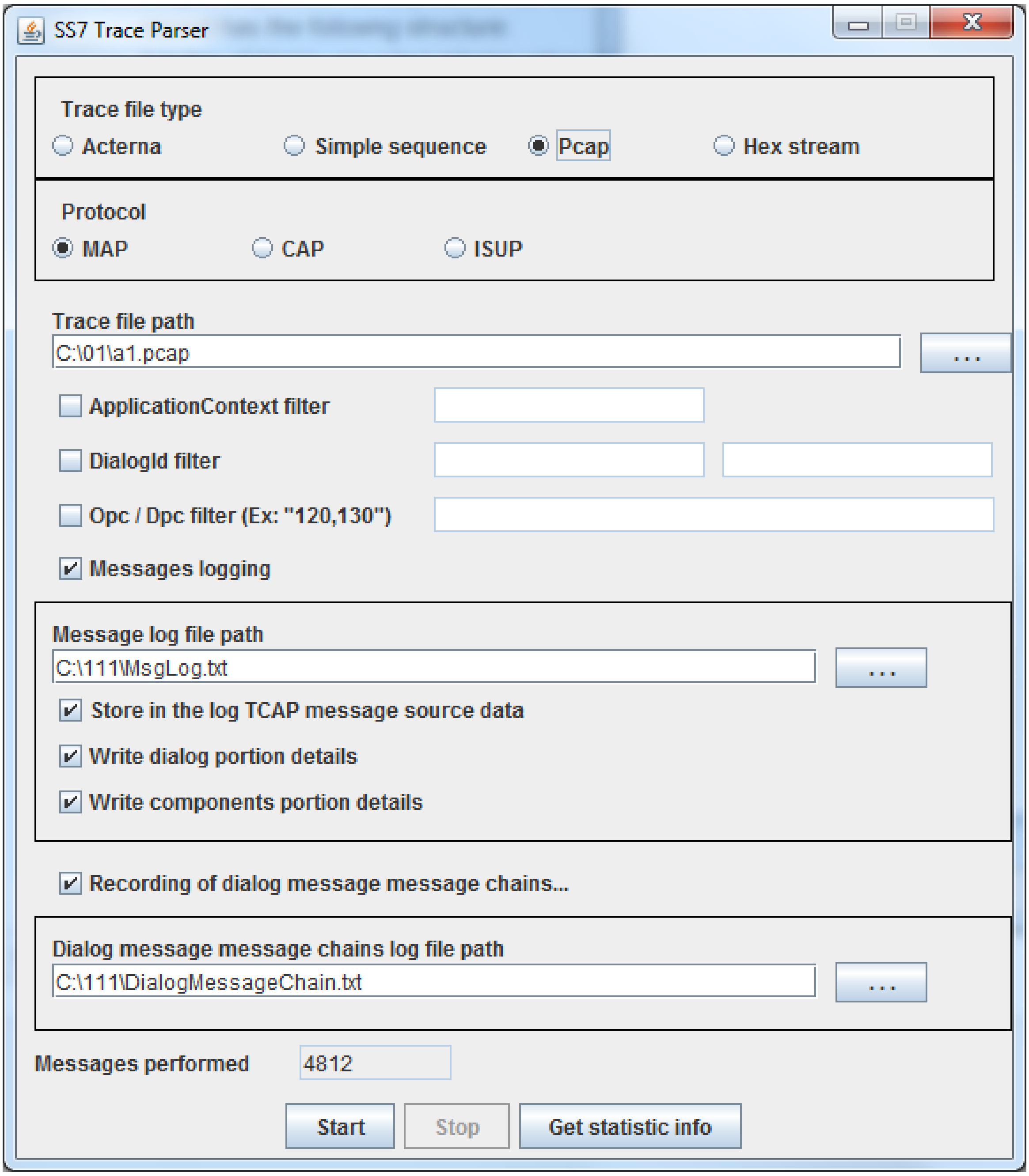
2. (Optional) Ensure that the *run.sh* start script is executable.

   ```
   bin$ chmod +x run.sh
   ```

3. Execute the *run.sh* Bourne shell script.

   ```
   bin$ ./run.sh
   ```

4. Result: This will launch the GUI application as shown in the figure below:



SS7 Trace Parser

Trace file type

○ Acterna ○ Simple sequence ● Pcap ○ Hex stream

Protocol

● MAP ○ CAP ○ ISUP

Trace file path: C:\01\a1.pcap

□ ApplicationContext filter

□ DialogId filter

□ Opc / Dpc filter (Ex: "120,130")

☑ Messages logging

Message log file path: C:\111\MsgLog.txt

☑ Store in the log TCAP message source data

☑ Write dialog portion details

☑ Write components portion details

☑ Recording of dialog message message chains...

Dialog message message chains log file path: C:\111\DialogMessageChain.txt

Messages performed: 4812

Figure 1. Running Trace Parser

Configuring Trace Parser

Once trace parser is started user can configure tool depending on requirements as explained below

- Trace file type : User can configure a trace file type for parsing. Possible values are:
    - Acterna - file format of logs of some Acterna family devices
    - Simple sequence - binary file format that content consists of several records. When you have some unsupported data format you can convert your data into this format. Each record has the following structure:
        - 2 bytes of binary encoded integer value with record length
        - without two length bytes (less significant byte is first) binary code of legacy SS7 message (Routing label and user data)

    - PCAP - pcap wireshark file format. DLT_EN10MB, DLT_LINUX_SLL and DLT_MTP3 network formats are supported now
    - Hex stream - special encoded hex text file format, it is not covered in this manaul.

- Protocol: MAP, CAP or ISUP - defines which protocol will be parsed.
- Trace file path: you have to select a file for parsing
- ApplicationContext filter - this value is used only for MAP protocol. It is integer value and used for message filtering. The value of this field is 7th number of MAP application context. For example for USSD services this value is 19. So can be filtered only all versions for a concrete application context.
- DialogId filter - for MAP and CAP protocols it is possible to filter one dialog chain. Two values in the integer form (as they are present in the text log of Trace Parser) can be used (to define origination and destination TCAP Transaction Id). Also can be only one Integer value used.
- OPC / DPC filter - this filter filters messages only for defined SS7 point codes. You can define several point codes, for example: "123,124,125". Use this filter when trace file contains a lot of data between several point codes and you want to select messages.
- Message logging - User have to define a file where text parsed data will be stored. User also needs to define which details will be saved into a log (TCAP message source data, MAP/CAP dialog portion details, components portion details)
- Recording of dialog message chains log - user can write another log that contains trace of which DPC / OPC nodes dialogs are sent. Look at log format for other details. If there are several dialogs with the same path they will be present only once in the log.

User can start parsing with the button "Start", stop with the button "Stop". After the parsing process is finished, user can get some statistic info like how many different TCAP primitives are exchanged, messages for separate Application contexts and operational codes.