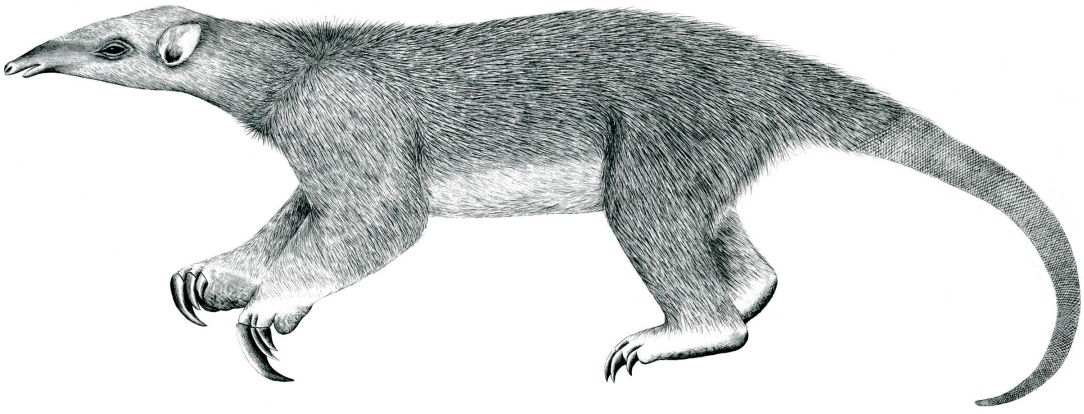
Tranalyzer2

pcapd

Create PCAP Files



Tranalyzer Development Team

Contents

1 pcapd

1.1 Description

The pcapd plugin can be used to create PCAP files based on some criteria such as flow indexes (Section 1.3.2) or alarms raised by other plugins (Section Section 1.3.3).

1.2 Dependencies

If `PD_MODE_OUT=1` (one pcap per flow), the libpcap version must be at least 1.7.2. (In this mode, the plugin uses the `pcap_dump_open_append()` function which was introduced in the libpcap in February 12, 2015.)

1.3 Configuration Flags

The following flags can be used to configure the plugin:

Variable	Default	Description	Flags
`PD_MODE_PKT`	0	0: all packets, 1: activate packet range selection	
`PD_STRTPKT`	1	Packet at which processing starts	`PD_MODE_PKT=1`
`PD_ENDPKT`	10	Packet at which processing ends (0: end of flow)	`PD_MODE_PKT=1`
`PD_MODE_IN`	0	0: if `-e` option was used, extract flows listed in `FILE`, otherwise, extract flows whose alarm bit is set 1: dump all packets	
`PD_EQ`	1	Save matching (1) or non-matching (0) flows	`PD_MODE_IN=0`
`PD_OPP`	0	1: extract also the opposite flow, 0: don't	`PD_MODE_IN=0`
`PD_DIRSEL`	0	2: extract A flow, 3: extract B flow, 0: off	`PD_MODE_IN=0`
`PD_MODE_OUT`	0	0: one pcap 1: one pcap per flow	
`PD_SPLIT`	1	Split the output file (Tranalyzer `-W` option)	
`PD_LBSRCH`	0	Search algorithm (`-e` option): 0: linear search 1: binary search	
`PD_TSHFT`	0	Time stamp shift in packets	
`PD_TTSFTS`	0	Time stamp increment seconds	`PD_TSHFT=1`
`PD_TTSFTNMS`	1	Time stamp increment micro/nano seconds (depends on `TSTAMP_PREC` in *tranalyzer.h*)	`PD_TSHFT=1`
`PD_MACSHFT`	0	MAC shift in packets	
`PD_MACSSHFT`	1	Src MAC increment in packets last byte	`PD_MACSHFT=1`
`PD_MACDSHFT`	1	Src MAC increment in packets last byte	`PD_MACSHFT=1`
`PD_VLNSHFT`	0	VLAN shift in packets	
`PD_VLNISHFT`	1	VLAN increment in packets	`PD_VLNSHFT=1`
`PD_IPSHFT`	0	IPv4/6 increment in packets	
`PD_IP4SHFT`	0x00000001	IPv4 shift 32 bit network order	`PD_IPSHFT=1`
`PD_IP6SHFT`	0x0000000000000001	IPv6 shift last 64 bit network order	`PD_IPSHFT=1`

Variable	Default	Description	Flags
PD_TTLSHFT	0	0: no TTL change, 1: TTL shift, 2: random shift	
PD_TTL	8	sub value from TTL	PD_TTLSHFT=1
PD_TTLMOD	128	TTL modulo	PD_TTLSHFT>0
PD_CHKSUML3	0	Correct checksum in IPv4 header	
PD_MAX_FD	128	Max. number of simultaneously open file descriptors	PD_MODE_OUT=1
PD_SUFFIX	"_pcapd.pcap"	Suffix for output pcap file	

1.3.1 Environment Variable Configuration Flags

The following configuration flags can also be configured with environment variables (ENVCNTRL>0):

- PD_MAX_FD
- PD_SUFFIX

1.3.2 PD_MODE_IN=0, –e option used

The idea behind this mode (PD_MODE_IN=0 and Tranalyzer -e option used) is to use awk to extract flows of interest and then the pcapd plugin to create one or more PCAP with all those flows. The input file (-e option) consists of one column listing the flow indexes (one entry per row). Lines starting with '%', '#', a space or a tab are ignored, along with empty lines:

```
# This file is passed to tranalyzer via -e option: t2 -e file.txt ...
# Each row lists a different flow index to extract.
123   # Extract flow 123
456   # Extract flow 456
#789  # Do NOT extract flow 789
```

Flows whose index appears in the -e file will be dumped in a file named PREFIX_PD_SUFFIX, where PREFIX is the value given to Tranalyzer -e option. Note that if PD_EQ=0, then flows whose index does **not** appear in the file will be dumped.

1.3.3 PD_MODE_IN=0, –e option not used

In this mode (PD_MODE_IN=0 and Tranalyzer -e option **NOT** used), every flow whose status bit FL_ALARM=0x20000000 is set (PD_EQ=1) or not set (PD_EQ=0) will be dumped in a file named PREFIX_PD_SUFFIX, where PREFIX is the value given to Tranalyzer -w or -W option.

1.3.4 PD_MODE_IN=1

In this mode, all the packets are dumped into one or more PCAP files. If Tranalyzer -W option is used, then the pcap files will be split accordingly. For example, the following command will create PCAP files of 100MB each: tranalyzer -i eth0 -W out:100M

1.3.5 PD_MODE_OUT=1

In this mode, every flow will have its own PCAP file, whose name will end with the flow index.

1.4 Plugin Report Output

The following information is reported:

- Number of packets extracted

1.5 Additional Output

A PCAP file with suffix `PD_SUFFIX` will be created. The prefix and location of the file depends on the configuration of the plugin.

- If Tranalyzer `-e` option was used, the file is named according to the `-e` option.
- Otherwise the file is named according to the `-w` or `-W` option.

1.6 Examples

For the following examples, it is assumed that Tranalyzer was run as follows, with the `basicFlow` and `txtSink` plugins in their default configuration:

```
tranalyzer -r file.pcap -w out
```

The column numbers can be obtained by looking in the file `out_headers.txt` or by using `tawk` .

1.6.1 Extracting ICMP Flows

To create a PCAP file containing ICMP flows only, proceed as follows:

1. Identify the *"Layer 4 protocol"* column in `out_headers.txt` (column 14):
   `grep "Layer 4 protocol" out_headers.txt`
2. Extract all flow indexes whose protocol is ICMP (1):
   `awk -F'\t' '$14 == 1 { print $2 }' out_flows.txt > out_icmp.txt`
3. Configure pcapd.h as follows: `PD_MODE_IN=0, PD_EQ=1`
4. Build the pcapd plugin: `cd $T2HOME/pcapd/; ./autogen.sh`
5. Re-run Tranalyzer with the `-e` option:
   `tranalyzer -r file.pcap -w out -e out_icmp.txt`
6. The file `out_icmp.txt.pcap` now contains all the ICMP flows.

1.6.2 Extracting Non-ICMP Flows

To create a PCAP file containing non-ICMP flows only, use the same procedure as that of Section 1.6.1, but replace `PD_EQ=1` with `PD_EQ=0` in step 3. Alternatively, replace `$14==1` with `$14!=1` in step 2. Or if an entire flow file is preferred to the flow indexes only, set `PD_FORMAT=1` and replace `print $2` with `print $0` in step 2.