Generate NetFlow-compatible data with labels (Instruction Manual)

SiLK, the System for Internet-Level Knowledge, is a collection of traffic analysis tools developed by the CERT Network Situational Awareness Team (CERT NetSA) to facilitate security analysis of large networks. The SiLK tool suite supports the systematic collection, storage, and analysis of network flow data, enabling network security analysts to rapidly query large historical traffic data sets. SiLK is ideally suited for analyzing traffic on the backbone or border of a large, distributed enterprise or mid- sized ISP.

Installation:

Download silk-3.16.0.tar.gz from https://tools.netsa.cert.org/silk/download.html

1. tar -xzvf silk-3.16.0.tar.gz
2. cd silk-3.16.0
3. ./configure --prefix=/usr/local
   or ./configure --prefix=/MY_PATH/silk-3.16.0
4. make
5. make install

Note: rwstats only outputs in seconds, not in milliseconds.
To fix this for millisecond, update src/rwstats/rwstatssetup.c for the two lines:

- line 293: SK_OPTION_TIMESTAMP_ALWAYS_MSEC
- line 301: static int bin_time_uses_msec = 1;

And, make and make install to complete.

Note: rwcut outputs in milliseconds.

To make flow data:

First download trace and log data:

- Packet trace download: http://mawi.wide.ad.jp/mawi/
- Intrusion log download: http://www.fukuda-lab.org/mawilab/

And, run rwptoflow and rwstats (or rwcut). rwstats makes summary output and reorders the entries. Rwcut preserves the output order.

1. rwptoflow captureFilepath --flow-out=FlowData.rw
2. rwstats FlowDataPath \
   --fileds=FieldsList \
   --values=Flows \

--output-path=yyyymmdd_result.data \
--percentage=0

where FlowDataPath is the output from rwptoflow (FlowData.rw)

3. rwcut FlowDataPath \
--fields=FieldsList \
--output-path=yyyymmdd_result.data

where FlowDataPath is the output from rwptoflow (FlowData.rw)

For example, download TCPdump file and anomalous_suspicious.csv

- 201807011400.pcap.gz (1426.45 MB)
  http://mawi.wide.ad.jp/mawi/samplepoint-F/2018/201807011400.html
- 20180701_anomalous_suspicious.csv
  http://www.fukuda-lab.org/mawilab/v1.1/2018/07/01/20180701.html
- Run the following commands (choose either rwstats or rwcut)
  1. rwptoflow 201807011400.pcap --flow-out=20180701.rw
  2. rwstats 20180701.rw \
     --fields=1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,20,21,25,26,27,28,29 \
     --values=Flows \
     --output-path=20180701_result.data \
     --percentage=0
  3. rwcut 20180701.rw \
     --fields=1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,20,21,25,26,27,28,29 \
     --output-path=20180701_result.data

Note: Below table shows the available fields to retrieve from the raw trace. The field numbers are specified in the "--fields=" option.

Labeling Flow Data:

The next step is to combine the flow data with the given anomaly information. We describe how to combine the flow record in the flow table with the anomalous traffic information provided by MAWILab.

For the flow data (15 minutes of a day), it is possible to download the associated attack information in a csv format. For example, a flow records data file (e.g. 20180701_result.data) and the associated csv file (e.g. 20180701_anomalous suspicious.csv).

To combine those two files, we implemented a new Python3 program of "flowlabeling.py", which is based on the old internal program combineFlow_1003.py. The program is executed based on priority rules defined below. Option "--sec" is when the records are in seconds. Without source modifications, outputs from rwstats are in seconds. By default, it assumes records in milliseconds.

Priority	matches	sIP	sPort	dIP	dPort

41	4	match	match	match	match
34	3	match	null	match	match
33	3	match	match	match	null
32	3	null	match	match	match
31	3	match	match	null	match
26	2	match	null	match	null
25	2	null	null	match	match
24	2	null	match	match	null
23	2	match	null	null	match
22	2	match	match	null	null
21	2	null	match	null	match
14	1	null	null	match	null
13	1	match	null	null	null
12	1	null	null	null	match
11	1	null	match	null	null
0	0				

With the assumption that the flow and attack files are located in the same directory, the program is being executed with the following command:

% python flowlabeling.py -t YYYYMMDD

e.g. python flowlabeling.py -t 20180701
python flowlabelling.py -t 20180701
python flowlabelling.py -t 20180701 --sec
python flowlabelling.py -i ./20180701_result.data
python flowlabelling.py -i ./20180701_result.data -c 20180701_anomalous suspicious.csv
python flowlabelling.py -i ./20180701_result.data -o output5

% python flowlabeling.py -h

```
usage: flowlabeling.py [-h] [-i INPUTFILE] [-c CLASSIFIER] [-o OUTPUTDIR] [-t DATESTR] [--sec]
        Tool to combine the flow and classifier
        optional arguments:
         -h, --help            show this help message and exit
         -i INPUTFILE, --input INPUTFILE   input file path. e.g. *_result.data
         -c CLASSIFIER, --classifier CLASSIFIER  input classifier file path. e.g. *_anomalous_suspicious.csv
         -o OUTPUTDIR, --output OUTPUTDIR  output directory path
         -t DATESTR, --time DATESTR       datetime of the file.  When used, -i and -o are ignored.
         --sec   flow times in seconds, rather than milliseconds. default False.
```

To break the outputs into multiple files with designated time windows, use the Python3 program “flowsplitter.py”.

% python flowsplitter.py -t YYYYMMDD

e.g. python flowsplitter.py -t 2018070101 -n 5
python flowsplitter.py -t 2018070101 -n 5 --sec
python flowsplitter.py -t 2018070101 -n 15
python flowsplitter.py -t 2018070101 -n 30

```
python flowsplitter.py -i ./2018070101_result/2018070101_mawilab_flow.csv -n 5
python flowsplitter.py -i ./2018070101_result/2018070101_mawilab_flow.csv -o output5 -n 5

% python flowsplitter.py -h
  usage: flowsplitter.py [-h] [-i INPUTFILE] [-o OUTPUTDIR] [-t DATESTR] [-n SPLITSEC]
        Tool to split the flow files in timed order
        optional arguments:
         -h, --help   (show this help message and exit)
         -i INPUTFILE, --input INPUTFILE  (input flow file path. e.g. *_mawilab_flow.csv)
         -o OUTPUTDIR, --output OUTPUTDIR (output directory path)
         -t DATESTR, --time DATESTR (datetime of the file. When used, -i and -o are ignored.)
         -n SPLITSEC  (time separation in seconds. default 5 sec.)
         --sec  (flow times from rwstats in seconds, rather than milliseconds. default False.)
```

The output:

Note that feature #26 class is the label for anomaly detection.

Feature #	Field# (in Silk)	Feature	NetFlow field	Description
1	1	sIP	IPV4_SRC_ADDR	Source IP
2	2	dIP	IPV4_DST_ADDR	Dest IP
3	3	sPort	L4_SRC_PORT	Source Port
4	4	dPort	L4_DST_PORT	Dest port
5	5	proto	PROTOCOL	IP protocol
6	6	packets	IN_BYTES	Packet count
7	7	bytes	IN_PKTS	Byte count
8	8	flags	TCP_FLAGS	Bit-wise or of TCP flags over all packets
9	9	sTime	UNIX_Seconds	Starting time of flow (in sec)
10	10	durat		Duration of flow (in sec)
11	11	eTime		End time of flow (in sec)
12	12	sen	FLOW_SAMPLER_ID	Name or ID of the sensor
13	13	in	SRC_VLAN	Router SNMP input interface
14	14	out	DST_VLAN	Router SNMP output interface
15	15	nhIP	IPV4_NEXT_HOP	Router next hop ID
16	16	sType	SRC_TOS	Type of source IP address (pmap required)
17	17	dType	DST_TOS	Type of destination IP address (pmap required)
18	20	senClass		Class of sensor that collected flow (SiLK-specific)
19	21	typeFlow		Type of flow for this sensor class (SiLK-specific)
20	--	iType	ICMP_TYPE	ICMP type value for ICMP flows
21	--	iCode		ICMP code value
22	26	initialF		TCP flags on first packet in flow
23	27	sessionF		Bit-wise OR of TCP flags over all packets except the first in the flow
24	28	attribut		Flow attributes set by the flow generator
25	29	appli		Guess as to the content of the flow
26	--	class		{Normal, Anomaly, Unsure} -- Records labeled Unsure may be excluded for anomaly detection experiments

27	--	taxonomy		category assigned to the anomaly using the taxonomy for backbone traffic anomalies 1. *Unknown* are labels starting with the prefixes "unk" and "empty" 2. *Other* are labels starting with the prefixes "ttl_error","hostout","netout", and "icmp_error" 3. *HTTP* are labels starting with the prefixes "alphflHTTP","ptmpHTTP","mptpHTTP","ptmplaHTTP" and "mptplaHTTP" 4. *Multi. points* are labels starting with the prefixes "ptmp","mptp" and "mptmp" 5. *Alpha flow* are labels starting with the prefixes "alphfl","malphfl","salphfl","point_to_point" and "heavy_hitter" 6. *IPv6 tunneling* are labels starting with the prefixes "ipv4gretun" and "ipv46tun" 7. *Port scan* are labels starting with the prefixes "posca" and "ptpposca" 8. *Network scan ICMP* are labels starting with the prefixes "ntscIC" and "dntscIC" 9. *Network scan UDP* are labels starting with the prefixes "ntscUDP" and "ptpposcaUDP" 10. *Network scan TCP* are labels starting with the prefixes "ntscACK","ntscSYN","sntscSYN","ntscTCP","ntscnull","ntscXmas","ntscFIN" and "dntscSYN" 11. *DoS* are labels starting with the prefixes "DoS","distributed_dos","ptpDoS","sptpDoS","DDoS" and "rflat"
28	--	label		• The label *anomalous* is assigned to all abnormal traffic and should be identified by any efficient anomaly detector. • The label *suspicious* is assigned to all traffic that is probably anomalous but not clearly identified by our method. • The label *notice* is assigned to all traffic that is not identified

				anomalous by our method but that has been reported by at least one anomaly detector. This traffic should not be identified by any anomaly detector, we do not label them as benign in order to trace all the alarms reported by the combined detectors.
29	--	heuristic		code assigned to the anomaly using simple heuristic based on port number, TCP flags and ICMP code
30	--	distance		difference *Dn-Da*
31	--	nbDetectors		number of configurations (detector and parameter tuning) that reported the anomaly