Statistical profiling

Generated by Doxygen 1.8.9.1

Wed Oct 26 2022 16:10:06

Contents

1 Statistical profiling 1

2 Namespace Index 3
2.1 Namespace List 3

3 File Index 5
3.1 File List 5

4 Namespace Documentation 7
4.1 detection Namespace Reference 7
4.1.1 Variable Documentation 7
4.1.1.1 args 7
4.1.1.2 input_file_name 7
4.1.1.3 outliers_by_char_dict 7
4.1.1.4 output 7
4.1.1.5 parser 7
4.1.1.6 profiles_dict 7
4.1.1.7 profiles_file_name 8
4.1.1.8 split_traffic_dict 8
4.1.1.9 time_window_size 8
4.1.1.10 traffic_dict 8
4.2 statistical_modeling Namespace Reference 8
4.2.1 Variable Documentation 8
4.2.1.1 args 8
4.2.1.2 best_split_point 8
4.2.1.3 candidate_split_points_dict 8
4.2.1.4 input_file_name 8
4.2.1.5 input_list 8
4.2.1.6 max_std 8
4.2.1.7 output 9
4.2.1.8 parser 9
4.2.1.9 profile_list 9
4.2.1.10 profiles_dict 9
4.2.1.11 split_points_results_dict 9
4.2.1.12 split_traffic_dict 9
4.2.1.13 time_window_size 9
4.2.1.14 traffic_dict 9
4.3 statistical_modeling_functions Namespace Reference 9
4.3.1 Function Documentation 10
4.3.1.1 add_delta_time_and_split_directions 10
4.3.1.2 delta_time_statistics 10
4.3.1.3 detect_all_outliers 10
4.3.1.4 detect_outliers 10
4.3.1.5 final_traffic_statistics 11
4.3.1.6 gather_number_of_packets 11
4.3.1.7 process_profiles_file 11

4.3.1.8 process_traffic_file 12
4.3.1.9 select_split_point 12
4.3.1.10 split_point_statistics 12

5 File Documentation **15**
5.1 detection.py File Reference 15
5.2 README.md File Reference 15
5.3 statistical_modeling.py File Reference 15
5.4 statistical_modeling_functions.py File Reference 16

Index **17**

Chapter 1

Statistical profiling

The tool consist of three files:

- `statistical_modeling_functions.py` - a module with functions used by other scripts.
- `statistical_modeling.py` - creates the statistical model(s) for the provided traffic.
- `detection.py` - detects the anomalies in the given traffic with the respect to the specified profile.

Requirements

- Python - version 3.9
- Pandas - version 1.2.4

Traffic model creation

The statistical model can be created with `statistical_modeling.py` script. The model consists of the profiles for each pair of IP adresses and for each direction. Profiles are printed to standard output, one per line.

Parameters:

`-f`: specifies the file with IEC104 data in csv format, required parameter \ `-t`: allows to specify the size of the time window in seconds, optional parametr, default value = 300 seconds

Example usage:

“‘bash python statistical_modeling.py -f datasets/mega104-17-12-18-ioa.csv > mega104-17-12-18-profile.csv “‘

Creates profiles of the communications captured in the file `datasets/mega104-17-12-18-ioa.csv`. For each pair of IP adresses and for each direction, one profile is derived. Profiles are stored one per line.

Anomalies detection:

Anomalies can be detected with the `detection.py` script.

Parameters:

`-f`: specify the file with IEC104 data in csv format, where anomalies should be found, required parameter \ `-p`: specify the file with communications profiles, that will be used to find the anomalies, required parametr \ `-t`: allows to specify the size of the time window in seconds, optinal parametr, default value = 300 seconds

Example usage:

"‘ python detection.py -f attacks/connection-loss.csv -p 17-12-18-profiles.csv "‘

The script compares the traffic captured in file `connection-loss.csv` against the profile stored in file `17-12-18-profiles.csv`. Time windows that do not fit into ranges defined in profiles are printed to standard output.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

detection 7
statistical_modeling 8
statistical_modeling_functions 9

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

detection.py 15
statistical_modeling.py 15
statistical_modeling_functions.py 16

Chapter 4

Namespace Documentation

4.1 detection Namespace Reference

Variables

- dictionary traffic_dict = {}

  dictionary of the lists that capture one-directinal communication
- dictionary split_traffic_dict = {}

  dictionary of the lists that capture one-directinal communication with added inter-arrival times
- dictionary profiles_dict = {}

  dictionary of statistical descriptions (one item per each one-directional communication)
- dictionary outliers_by_char_dict = {}

  dictionary of the anomaly time windows.
- tuple parser = ArgumentParser(description='The argument -f is required to specify the input file.')
- tuple args = parser.parse_args()
- input_file_name = args.input_file
- profiles_file_name = args.profiles_file
- time_window_size = args.time_window_size
- string output = key+": no profile available for the communication."

4.1.1 Variable Documentation

4.1.1.1 tuple detection.args = parser.parse_args()

4.1.1.2 detection.input_file_name = args.input_file

4.1.1.3 dictionary detection.outliers_by_char_dict = {}

dictionary of the anomaly time windows.

4.1.1.4 tuple detection.output = key+": no profile available for the communication."

4.1.1.5 tuple detection.parser = ArgumentParser(description='The argument -f is required to specify the input file.')

4.1.1.6 dictionary detection.profiles_dict = {}

dictionary of statistical descriptions (one item per each one-directional communication)

4.1.1.7 detection.profiles_file_name = args.profiles_file

4.1.1.8 dictionary detection.split_traffic_dict = {}

dictionary of the lists that capture one-directinal communication with added inter-arrival times

4.1.1.9 detection.time_window_size = args.time_window_size

4.1.1.10 dictionary detection.traffic_dict = {}

dictionary of the lists that capture one-directinal communication

4.2 statistical_modeling Namespace Reference

Variables

- dictionary traffic_dict = {}

  dictionary of the lists that capture one-directinal communication
- dictionary split_traffic_dict = {}

  dictionary of the lists that capture one-directinal communication with added inter-arrival times
- dictionary candidate_split_points_dict = {}

  dictionary of candidate split-points (one item per each one-directional communication)
- dictionary profiles_dict = {}

  dictionary of final statistical descriptions (one item per each one-directional communication)
- int max_std = 0

  maximal standard deviation revealed during the best split point search
- tuple parser = ArgumentParser(description='The argument -f is required to specify the input file.')
- tuple args = parser.parse_args()
- input_file_name = args.input_file
- time_window_size = args.time_window_size
- list input_list = split_traffic_dict[key]
- dictionary split_points_results_dict = {}
- list profile_list = [ ]
- tuple best_split_point = smf.select_split_point(split_points_results_dict, max_std)
- string output = key+";"

4.2.1 Variable Documentation

4.2.1.1 tuple statistical_modeling.args = parser.parse_args()

4.2.1.2 tuple statistical_modeling.best_split_point = smf.select_split_point(split_points_results_dict, max_std)

4.2.1.3 dictionary statistical_modeling.candidate_split_points_dict = {}

dictionary of candidate split-points (one item per each one-directional communication)

4.2.1.4 statistical_modeling.input_file_name = args.input_file

4.2.1.5 list statistical_modeling.input_list = split_traffic_dict[key]

4.2.1.6 tuple statistical_modeling.max_std = 0

maximal standard deviation revealed during the best split point search

4.2.1.7 tuple statistical_modeling.output = key+";"

4.2.1.8 tuple statistical_modeling.parser = ArgumentParser(description='The argument -f is required to specify the input file.')

4.2.1.9 tuple statistical_modeling.profile_list = [ ]

4.2.1.10 dictionary statistical_modeling.profiles_dict = {}

dictionary of final statistical descriptions (one item per each one-directional communication)

4.2.1.11 dictionary statistical_modeling.split_points_results_dict = {}

4.2.1.12 dictionary statistical_modeling.split_traffic_dict = {}

dictionary of the lists that capture one-directinal communication with added inter-arrival times

4.2.1.13 statistical_modeling.time_window_size = args.time_window_size

4.2.1.14 dictionary statistical_modeling.traffic_dict = {}

dictionary of the lists that capture one-directinal communication

4.3 statistical_modeling_functions Namespace Reference

Functions

- def process_traffic_file (file_name, traffic_dict)

  Separates the communications from the input file to the traffic-dict.
- def process_profiles_file (file_name, profiles_dict)

  Separates the profiles that will be used for anomaly detection.
- def add_delta_time_and_split_directions (traffic_dict, split_traffic_dict)

  Finds inter-arrival time for each packet and splits the communication into directions.
- def delta_time_statistics (input_dict, output_dict)

  Finds the quartiles and mean of inter-arrival times.
- def gather_number_of_packets (input_list, split_point, time_window_size, dict_with_windows)

  Counts the number of packets transmitted within all time windows of a given size.
- def split_point_statistics (input_list, boundary, time_window_size, results_dict, max_std)

  Finds mean and standard deviation of the series obtained for the given boundary (split-point) value.
- def final_traffic_statistics (input_list, split_point, time_window_size)

  Finds the final statistical profile of the given one-directional traffic.
- def select_split_point (statistics_dict, max_std)

  Select the best split point from the candidates.
- def detect_outliers (input_list, lower_boundary, upper_boundary)

  Detects outlier values in one characteristic of the traffic.
- def detect_all_outliers (input_list, boundaries_list, time_window_size, output_dict)

  Detects outliers in time windows series for all three characteristics.

4.3.1 Function Documentation

4.3.1.1 def statistical_modeling_functions.add_delta_time_and_split_directions (*traffic_dict, split_traffic_dict*)

Finds inter-arrival time for each packet and splits the communication into directions.

Inter-arrival times are computed from relative times in bidirectional traffic. Next, communications are divided by direction to the output dictionary.

Parameters

traffic_dict	dictionary of bidirectional traffic without inter-arrival times
split_traffic_dict	dictionary of one-directional traffic with inter-arrival times

Returns

split_traffic_dict

4.3.1.2 def statistical_modeling_functions.delta_time_statistics (*input_dict, output_dict*)

Finds the quartiles and mean of inter-arrival times.

For each item in input_dict (one-directional traffic), quartiles and mean are found and stored in output_dictinary.

Parameters

input_dict	should contain list of items for individual directions
output_dict	output parameter that returns values of medians and mean as a list of values for each direction

Returns

output_dict

4.3.1.3 def statistical_modeling_functions.detect_all_outliers (*input_list, boundaries_list, time_window_size, output_dict*)

Detects outliers in time windows series for all three characteristics.

The time-window representation of the traffic is found. For each characteristic the method detect_outliers() is called. The numbers of windows where anomaly occur are returned in output_dict.

Parameters

input_list	list of packets in one-directional traffic (with inter-arrival times)
boundaries_list	the list of values that represent tha statistical profile
time_window_↩ size	size of time window in seconds
output_dict	the dictionary with outliers (time windows with anomaly)

Returns

output_dict

4.3.1.4 def statistical_modeling_functions.detect_outliers (*input_list, lower_boundary, upper_boundary*)

Detects outlier values in one characteristic of the traffic.

Values from input_list are compared with boundaries. 3-value-detection method is used. If two out of three consecutive values are outside the specified range, an anomaly is reported (added to output_list and returned).

Parameters

input_list	list of values in which anomalies are searched for
lower_boundary	specifies the lower limit of the range of normal values
upper_boundary	specifies the upper limit of the range of normal values

Returns

output_list: list of anomalies

4.3.1.5 def statistical_modeling_functions.final_traffic_statistics (*input_list, split_point, time_window_size*)

Finds the final statistical profile of the given one-directional traffic.

The time-window representation of the traffic is found. Then outliers are removed (using 3-sigma rule). Statistical profile consisting of the split-point value and boundaries of the ranges of normal values for all three characteristics is found (using 3-sigma rule) and returned.

Parameters

input_list	list of packets in one-directional traffic (with inter-arrival times)
split_point	value used to separate transmitted packets to two groups according to inter-arrival time
time_window_↩ size	size of time window in seconds

Returns

output_list: list of values that compose the profile

4.3.1.6 def statistical_modeling_functions.gather_number_of_packets (*input_list, split_point, time_window_size, dict_with_windows*)

Counts the number of packets transmitted within all time windows of a given size.

Converts the time-series of packets into series of number of packets transmitted within consecutive time windows. Besides the total number of packets, also the number of packets with inter-arrival time smaller than split-point and the number of packets with inter-arrival time greater than or equal to split-point within each time window are found. Three resulting series of values are returned in dict_with_windows.

Parameters

input_list	list of transmitted packets with relative time and inter-arrival time
split_point	value used to separate transmitted packets to two groups according to inter-arrival time
time_window_↩ size	window size in seconds
dict_with_↩ windows	output parameter that returns three series of values as a dictinary

Returns

dict_with_windows

4.3.1.7 def statistical_modeling_functions.process_profiles_file (*file_name, profiles_dict*)

Separates the profiles that will be used for anomaly detection.

Converts the information about comunnication profiles into dictionary. Split point and boundaries of individual characteristics are stored for each conversation.

Parameters

file_name	name of the file with statistical model
profiles_dict	output parameter, dictionary of statistical models

Returns

profiles_dict

4.3.1.8 def statistical_modeling_functions.process_traffic_file (*file_name, traffic_dict*)

Separates the communications from the input file to the traffic-dict.

Communication is identified from third and fourth column of the input file (IP adresses). For each pair of devices one item (with one key) in output dictionary is created. For each communication - relative time and the direction is stored for each packet. Directions are not distinguished yet.

Parameters

file_name	name of the csv file with network traffic (one line per packet)
traffic_dict	output parameter, dictionary of conversations, stores relative time and directions

Returns

traffic_dict

4.3.1.9 def statistical_modeling_functions.select_split_point (*statistics_dict, max_std*)

Select the best split point from the candidates.

For each candidate split-point (key) the resulting mean and standard deviation are stored in statistics_dict. The best split-point leads to the smallest standard deviation and defines the lower boundary of the final range of normal values greater then zero.

Parameters

statistics_dict	dictionary with means and standard deviations for each candidate
max_std	largest standard deviation found so far, used for initialization

Returns

best_boundary: value of the best split-point of inter-arrival times

4.3.1.10 def statistical_modeling_functions.split_point_statistics (*input_list, boundary, time_window_size, results_dict, max_std*)

Finds mean and standard deviation of the series obtained for the given boundary (split-point) value.

The packets from input_list are transformed to series of the number of packets. The mean and standard deviation are found for series 'range1' and 'range2'. They are used to find the most suitable split-point. They are returned in result_dict as values for the key 'boundary'.

Parameters

input_list	list of packets in one-directional traffic (with inter-arrival times)
boundary	used as a candidate split-point
time_window_↩ size	size of time window in seconds
results_dict	for each boundary (used as a key) contains the mean and standard deviation of resulted series serve as input-output parameter
max_std	largest standard deviation found so far

Returns

max_std, results_dict

Chapter 5

File Documentation

5.1 detection.py File Reference

Namespaces

- detection

Variables

- dictionary detection.traffic_dict = {}

  dictionary of the lists that capture one-directinal communication
- dictionary detection.split_traffic_dict = {}

  dictionary of the lists that capture one-directinal communication with added inter-arrival times
- dictionary detection.profiles_dict = {}

  dictionary of statistical descriptions (one item per each one-directional communication)
- dictionary detection.outliers_by_char_dict = {}

  dictionary of the anomaly time windows.
- tuple detection.parser = ArgumentParser(description='The argument -f is required to specify the input file.')
- tuple detection.args = parser.parse_args()
- detection.input_file_name = args.input_file
- detection.profiles_file_name = args.profiles_file
- detection.time_window_size = args.time_window_size
- string detection.output = key+": no profile available for the communication."

5.2 README.md File Reference

5.3 statistical_modeling.py File Reference

Namespaces

- statistical_modeling

Variables

- dictionary statistical_modeling.traffic_dict = {}

  dictionary of the lists that capture one-directinal communication

- dictionary statistical_modeling.split_traffic_dict = {}

  dictionary of the lists that capture one-directinal communication with added inter-arrival times
- dictionary statistical_modeling.candidate_split_points_dict = {}

  dictionary of candidate split-points (one item per each one-directional communication)
- dictionary statistical_modeling.profiles_dict = {}

  dictionary of final statistical descriptions (one item per each one-directional communication)
- int statistical_modeling.max_std = 0

  maximal standard deviation revealed during the best split point search
- tuple statistical_modeling.parser = ArgumentParser(description='The argument -f is required to specify the input file.')
- tuple statistical_modeling.args = parser.parse_args()
- statistical_modeling.input_file_name = args.input_file
- statistical_modeling.time_window_size = args.time_window_size
- list statistical_modeling.input_list = split_traffic_dict[key]
- dictionary statistical_modeling.split_points_results_dict = {}
- list statistical_modeling.profile_list = [ ]
- tuple statistical_modeling.best_split_point = smf.select_split_point(split_points_results_dict, max_std)
- string statistical_modeling.output = key+";"

5.4 statistical_modeling_functions.py File Reference

Namespaces

- statistical_modeling_functions

Functions

- def statistical_modeling_functions.process_traffic_file (file_name, traffic_dict)

  Separates the communications from the input file to the traffic-dict.
- def statistical_modeling_functions.process_profiles_file (file_name, profiles_dict)

  Separates the profiles that will be used for anomaly detection.
- def statistical_modeling_functions.add_delta_time_and_split_directions (traffic_dict, split_traffic_dict)

  Finds inter-arrival time for each packet and splits the communication into directions.
- def statistical_modeling_functions.delta_time_statistics (input_dict, output_dict)

  Finds the quartiles and mean of inter-arrival times.
- def statistical_modeling_functions.gather_number_of_packets (input_list, split_point, time_window_size, dict_with_windows)

  Counts the number of packets transmitted within all time windows of a given size.
- def statistical_modeling_functions.split_point_statistics (input_list, boundary, time_window_size, results_dict, max_std)

  Finds mean and standard deviation of the series obtained for the given boundary (split-point) value.
- def statistical_modeling_functions.final_traffic_statistics (input_list, split_point, time_window_size)

  Finds the final statistical profile of the given one-directional traffic.
- def statistical_modeling_functions.select_split_point (statistics_dict, max_std)

  Select the best split point from the candidates.
- def statistical_modeling_functions.detect_outliers (input_list, lower_boundary, upper_boundary)

  Detects outlier values in one characteristic of the traffic.
- def statistical_modeling_functions.detect_all_outliers (input_list, boundaries_list, time_window_size, output↩ _dict)

  Detects outliers in time windows series for all three characteristics.

Index

add_delta_time_and_split_directions
    statistical_modeling_functions, 10
args
    detection, 7
    statistical_modeling, 8

best_split_point
    statistical_modeling, 8

candidate_split_points_dict
    statistical_modeling, 8

delta_time_statistics
    statistical_modeling_functions, 10
detect_all_outliers
    statistical_modeling_functions, 10
detect_outliers
    statistical_modeling_functions, 10
detection, 7
    args, 7
    input_file_name, 7
    outliers_by_char_dict, 7
    output, 7
    parser, 7
    profiles_dict, 7
    profiles_file_name, 7
    split_traffic_dict, 8
    time_window_size, 8
    traffic_dict, 8
detection.py, 15

final_traffic_statistics
    statistical_modeling_functions, 11

gather_number_of_packets
    statistical_modeling_functions, 11

input_file_name
    detection, 7
    statistical_modeling, 8
input_list
    statistical_modeling, 8

max_std
    statistical_modeling, 8

outliers_by_char_dict
    detection, 7
output
    detection, 7
    statistical_modeling, 8

parser
    detection, 7
    statistical_modeling, 9
process_profiles_file
    statistical_modeling_functions, 11
process_traffic_file
    statistical_modeling_functions, 12
profile_list
    statistical_modeling, 9
profiles_dict
    detection, 7
    statistical_modeling, 9
profiles_file_name
    detection, 7

README.md, 15

select_split_point
    statistical_modeling_functions, 12
split_point_statistics
    statistical_modeling_functions, 12
split_points_results_dict
    statistical_modeling, 9
split_traffic_dict
    detection, 8
    statistical_modeling, 9
statistical_modeling, 8
    args, 8
    best_split_point, 8
    candidate_split_points_dict, 8
    input_file_name, 8
    input_list, 8
    max_std, 8
    output, 8
    parser, 9
    profile_list, 9
    profiles_dict, 9
    split_points_results_dict, 9
    split_traffic_dict, 9
    time_window_size, 9
    traffic_dict, 9
statistical_modeling.py, 15
statistical_modeling_functions, 9
    add_delta_time_and_split_directions, 10
    delta_time_statistics, 10
    detect_all_outliers, 10
    detect_outliers, 10
    final_traffic_statistics, 11
    gather_number_of_packets, 11
    process_profiles_file, 11

process_traffic_file, 12
select_split_point, 12
split_point_statistics, 12
statistical_modeling_functions.py, 16

time_window_size
detection, 8
statistical_modeling, 9
traffic_dict
detection, 8
statistical_modeling, 9