Network Security
389.159 - SS 2018
Lab Exercise 3 & Lab Exercise 4

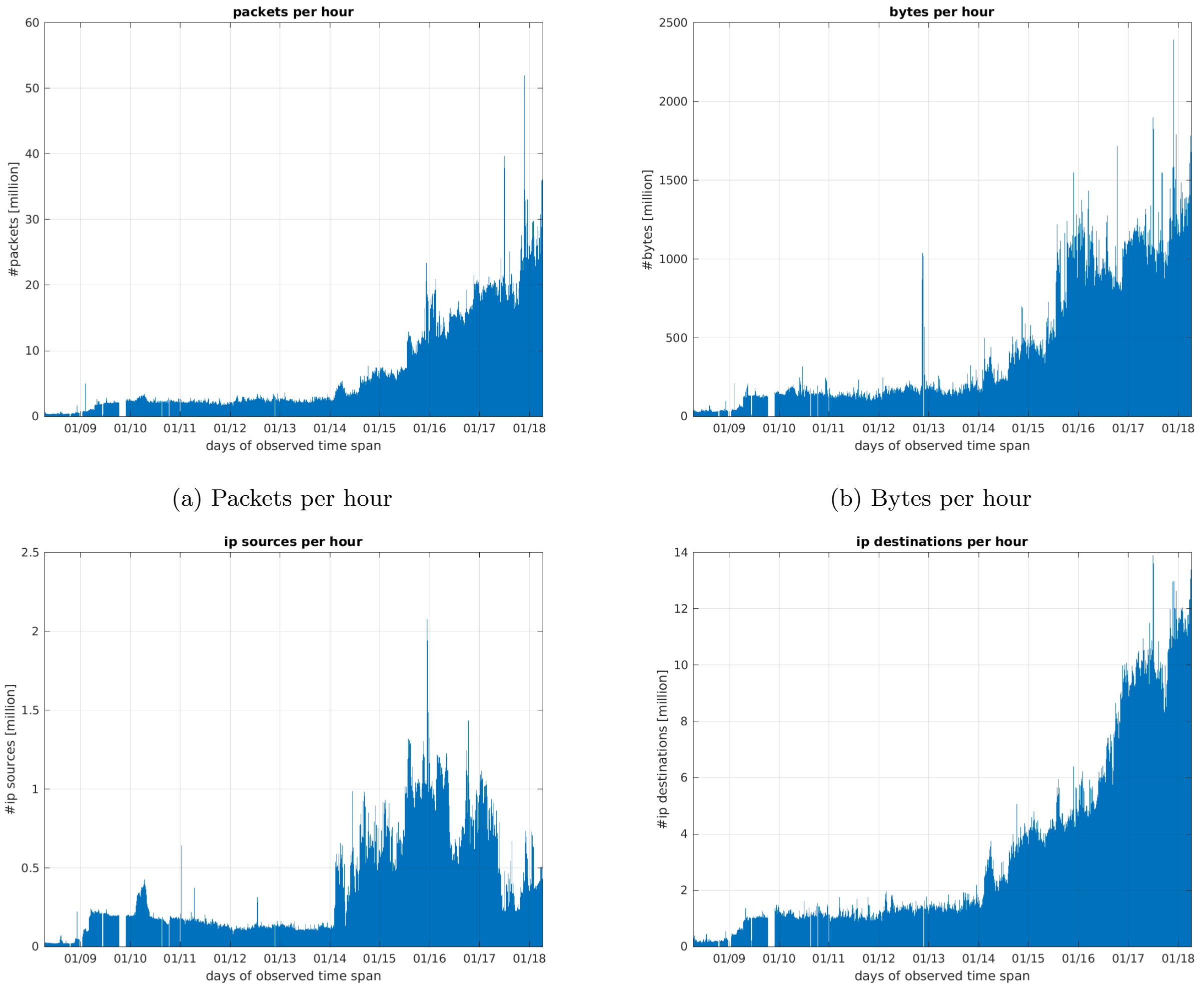
TEAM 02
Corentin Bergès (11741629) (066 506)
Christoph Echtinger-Sieghart (00304130) (066 938)

June 12, 2018

1 Lab Exercise 3

1.1 rep-10 → Matlab Code (Listing 2)

Figure 1 shows the stem plots for packets, bytes, unique IP sources and unique IP destinations per hour.



(a) Packets per hour

(b) Bytes per hour

(c) IP sources per hour

(d) IP destinations per hour

Figure 1

Optional Figure 2 shows all signals from Figure 1 combined, normalized and smoothed with a moving average filter.

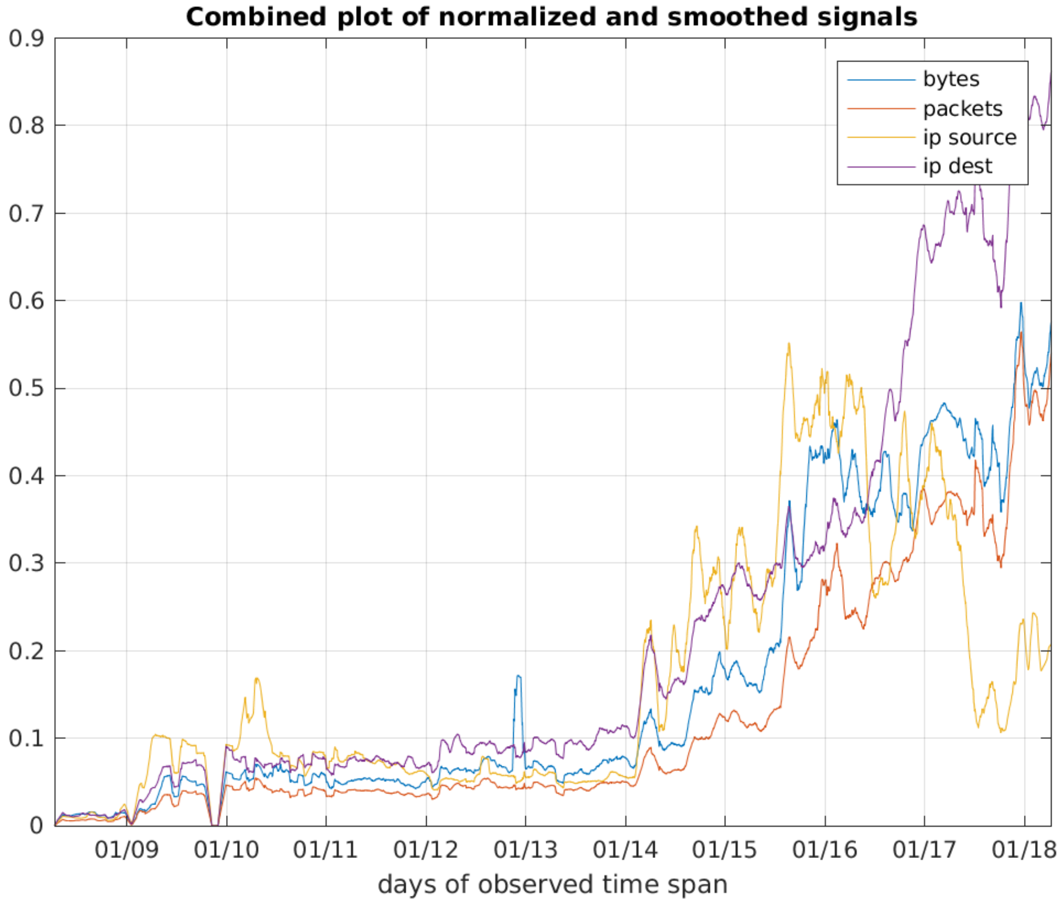


Figure 2: Combined, normalized and smoothed signals

1.2 rep-11 → Matlab Code (Listing 3)

The signal that shows the lower correlation to the other signals is **IP sources**. The minimum linear correlation coefficient is **0.5886** between the signals **IP sources** and **IP destinations**. See Table 1 for the raw data.

	Bytes	Packets	IP src	IP dst
Bytes	1	0.9655	0.7203	0.9340
Packets	0.9655	1	0.6105	0.9732
IP src	0.7203	0.6105	1	0.5886
IP dst	0.9340	0.9732	0.5886	1

Table 1: Correlation coefficients between signals

The reason for why the drop in unique IP sources does not cause a proportional drop in the other signals, could be that many small attackers (botnets), that did not contribute a lot to the other signals somehow stopped sending traffic.

1.3 rep-12 → Matlab Code (Listing 4)

The number of IP sources is bigger in average than the number of darkspace addresses receiving packets. There are on average around ten times more IP sources than IP destinations. This makes sense, because the darkspace is only a small part of the internet address-space, but the IP sources are taken from the whole address-space.

1.4 rep-13 → Matlab Code (Listing 5)

The main peak in IP sources starts on 14-Dec-2015 and lasts until 16-Dec-2015. See Table 2 for the detailed data.

Date	# IP sources
14-Dec-2015	2075358.074306
15-Dec-2015	1704892.012500
16-Dec-2015	1942072.404167

Table 2: Detailed data for peak in IP sources

Date	# Bytes
14-Nov-2012	870858582.136110
15-Nov-2012	1009586335.331900
16-Nov-2012	1038654926.456100
17-Nov-2012	1021464983.022200
18-Nov-2012	954193481.914190
20-Nov-2012	1005163238.508500
21-Nov-2012	1020526661.658000
22-Nov-2012	989613880.615110

Table 3: Detailed data for peak in Bytes

Optional → **Matlab Code (Listing 6)** The main peak in Bytes starts on 14-Nov-2012 and lasts until 22-Nov-2012. Note that on 19-Nov-2012 no data was available. See Table 3 for the detailed data.

1.5 rep-14 → Matlab Code (Listing 7)

Table 4 gives statistics for the data from `global_last10years.csv`. Table 5 gives statistics for the data from `Feb2017_gen.csv`.

	Sum	Mean	Median	StdDev
# Packets [millions]	146373.391	41.845	17.699	40.916
# Bytes [millions]	2381.003	0.681	0.263	0.735
# IP src [millions]	123.613	0.035	0.020	0.031
# IP dst [millions]	1150.796	0.329	0.142	0.330

Table 4: Statistics for daily data

	Sum	Mean	Median	StdDev
# Packets [millions]	76871.319	114.392	113.464	7.033
# Bytes [millions]	1272.998	1.894	1.890	0.097
# IP src [millions]	59.651	0.089	0.091	0.018
# IP dst [millions]	619.875	0.922	0.931	0.070

Table 5: Statistics for hourly data

1.6 rep-15

The statistical moments (mean, median, standard deviation) do not coincide. February 2017 seems to be a month that is not really representative for the data collected over a span of 10 years. It is interesting to note, that the relation in size between the four signals is roughly the same. The sum values for the 10 year span are around double the sum values for the data from February 2017.

Optional → **Matlab Code (Listing 8)** Figure 3 shows the boxplots for hourly and daily averaged data.

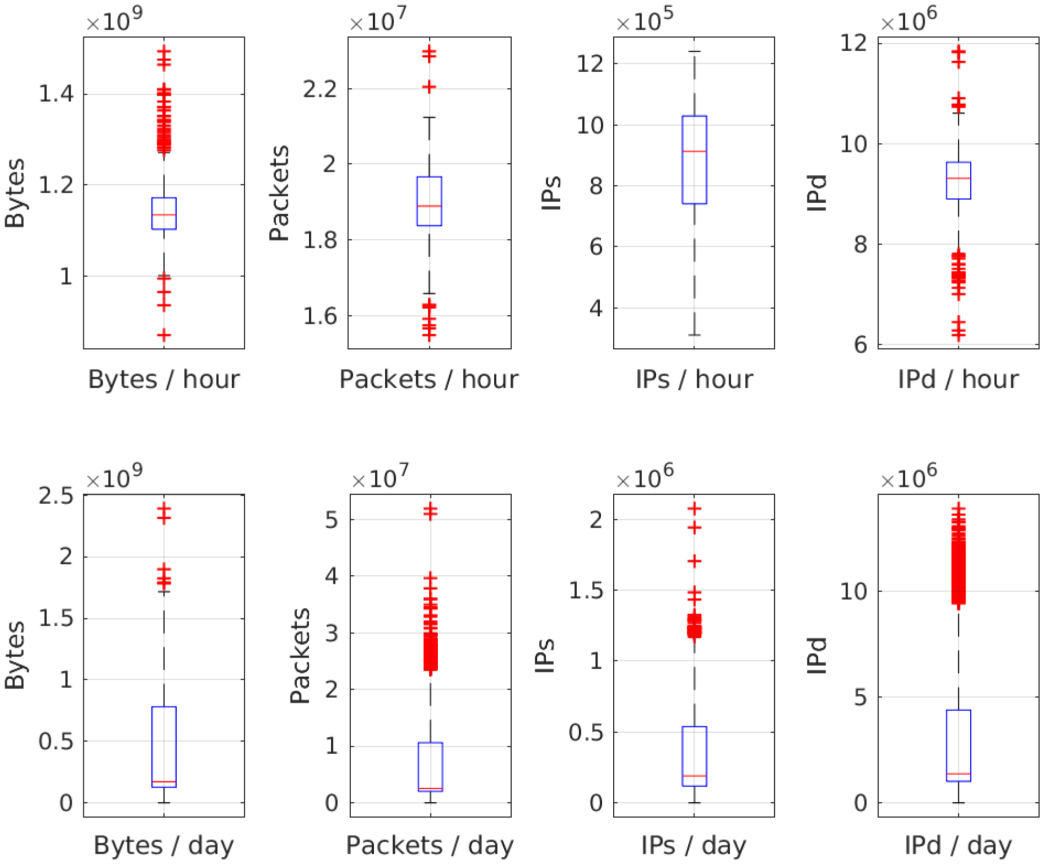


Figure 3: Boxplots for hourly (February 2017) and daily (last 10 years) averaged data

A difference between the box plots for the hourly and daily data are the positions of the medians in the plots. The medians in the daily averaged data show a clear tendency to be in the vicinity of the first quartile, whereas the medians in the hourly averaged data show no clear tendency. A second noticeable difference are the outliers. All outliers in the daily averaged data are above the whiskers, whereas the boxplots for the hourly averaged data show outliers below and above the whiskers. This is explained by the fact that the volume of packets, bytes and IP destinations had a major increase during the last three years (see Figure 2).

1.7 rep-16

We used `https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml` to look up the protocol numbers.

Protocol 6 (TCP) The Transmission Control Protocol is a connection oriented, reliable protocol that is widely used. A TCP connection is created by performing a three-way handshake (SYN, SYN/ACK, ACK). TCP uses port numbers to address applications.

Protocol 1 (ICMP) The Internet Control Message Protocol is part of the IP specification and used to exchange status and error messages concerning the IP protocol. ICMP is transported via IP. The popular `ping` and `traceroute` programs are applications that make use of ICMP.

Protocol 17 (UDP) The User Datagram Protocol is a connectionless, unreliable protocol. UDP does not perform a three-way handshake, but also uses port numbers to address applications.

1.8 rep-17 → Matlab Code (Listing 9)

Table 6 shows statistical information for the number of Packets per hour grouped by protocol. Table 7 shows statistical information for the number of IP sources per hour grouped by protocol. Table 8 shows statistical information for the number of IP destinations per hour grouped by protocol.

	Mean [%]	Median [%]	StdDev [%]
TCP	84.0	83.3	4.8
UDP	11.4	11.3	3.2
ICMP	4.30	5.4	2.6
Others	0.04	0.04	0.1

Table 6: Statistical information for Packets per hour

	Mean [%]	Median [%]	StdDev [%]
TCP	59.2	51.9	14.3
UDP	33.6	34.3	3.4
ICMP	17.5	22.5	10.6
Others	-10.3	-10.3	4.4

Table 7: Statistical information for IP sources per hour

	Mean [%]	Median [%]	StdDev [%]
TCP	89.0	88.3	3.9
UDP	15.8	15.5	4.5
ICMP	8.0	10.4	4.8
Others	-12.7	-14.2	4.2

Table 8: Statistical information for IP destinations per hour

Figure 4 shows boxplots for the data from `Feb2017_proto.csv` separated by protocol and signal.

Optional Figure 5 shows the various scatter plots.

1.9 rep-18

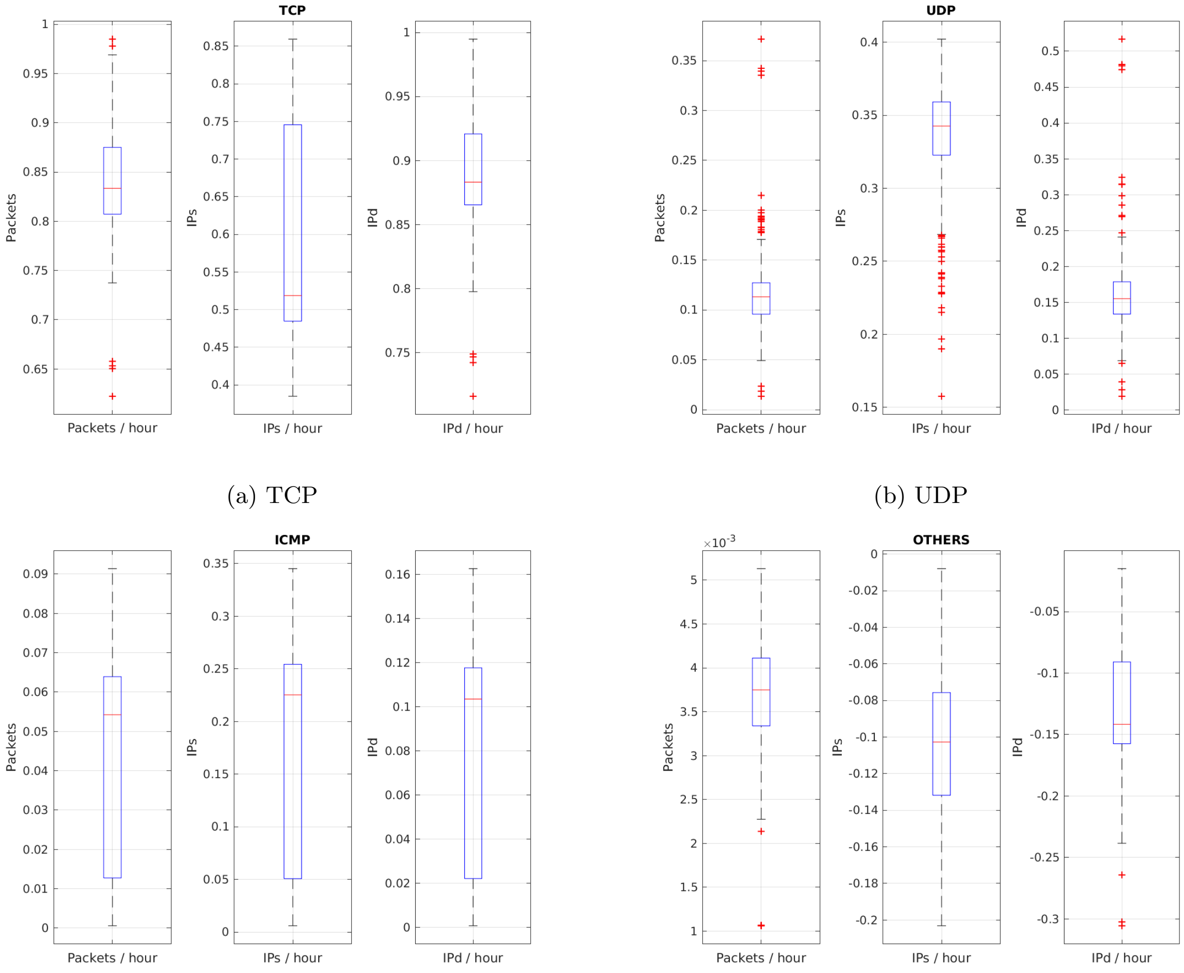
We obtained negative values for unique IP sources and destinations for "other" protocols because, in the combined data addresses are collapsed. The same address might get a TCP, a UDP and an ICMP packet, but will only be counted once.

1.10 rep-19 → Matlab Code (Listing 10)

The four most used destination ports in descending order are port 23, port 22, port 445 and port 80. Table 9 shows statistical information in absolute values for the four most used destination ports in the data file. Table 10 shows the statistical information in percentages.

Port 23 (Telnet) We see traffic to this port in the darkspace, because many devices on the internet have a telnet daemon listening on port 23 - often with minimal password protection. The connection attempts to port 23 are part of automated scanning for open telnet ports.

Port 22 (SSH) We see traffic to this port in the darkspace, because many host on the internet have an SSH daemon listening on port 22. The connection attempts to port 22 are part of automated scanning for open SSH ports - often followed by a dictionary based password guessing attack.



(a) TCP

(b) UDP

(c) ICMP

(d) Others

Figure 4: Boxplots separated by protocol and signal

Port 445 (Microsoft Directory Service) We see traffic to this port in the darkspace, because the "Microsoft Directory Service" is often exploited and various vulnerabilites in it where found in recent years.

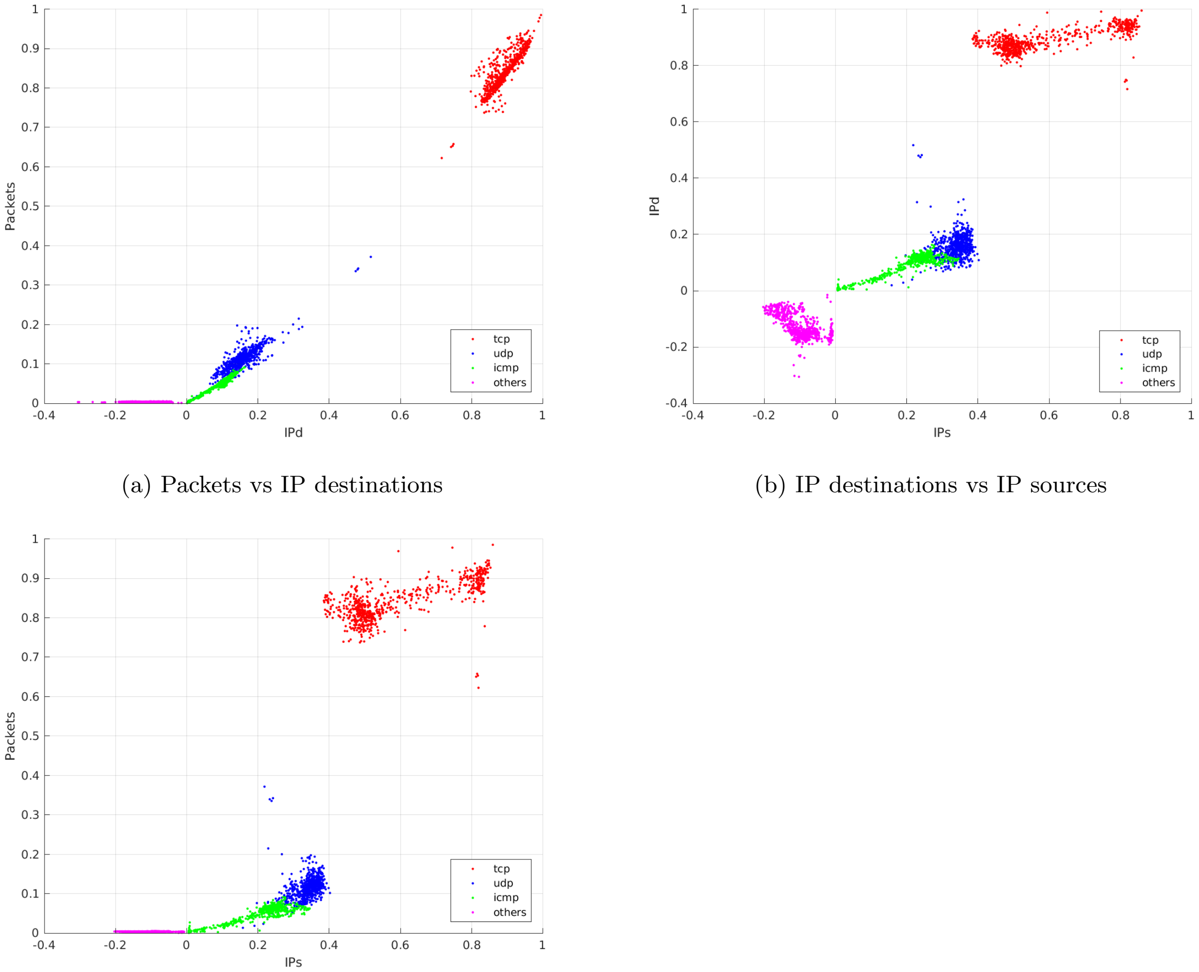
Port 80 (HTTP) We see traffic to this port, because many hosts on the internet have a webserver running on port 80. The connection attempts to port 80 are part of automated scanning for webservers.

	Port 23	port 22	Port 445	Port 80
Mean	0.627	0.049	0.026	0.015
StdDev	0.113	0.025	0.005	0.010

Table 9: Statistical information for TCP packets [in million]

	Port 23	port 22	Port 445	Port 80
Mean	39.7	3.1	1.6	0.9
StdDev	7.3	1.5	0.3	0.6

Table 10: Statistical information for TCP packets [in percent]



(a) Packets vs IP destinations

(b) IP destinations vs IP sources

(c) Packets vs IP sources

Figure 5: Scatter plots

1.11 rep-20 → Matlab Code (Listing 11)

Figure 6 shows the data for the ports 445 and 502. Data associated with port 445 is the data with the lowest relative difference between mean and median, data associated with port 502 is the data with the highest relative difference between mean and median.
The median better represents the average value of a dataset, because by definition 50 percent of the values in the distribution will be below the median, and the other 50 percent will be above the median. The median splits the distribution into half. An example for a distribution where the mean is not representative of the value distribution, would be an almost constant distribution with some extreme high or low outliers.

1.12 rep-21 → Matlab Code (Listing 12)

Figure 7 shows the time series plot for the number of packets per hour and the number of unique IP sources per hour. Figure 8 shows the amplitude spectra for the number of packets per hour and the number of unique IP sources per hour.

Periodicity Table 11 shows the maximum FFT value and associated information for Packets per hour and unique IP sources per hour. Table 12 shows further temporal patterns found in the signal for Packets per hour. Table 13 shows further temporal patterns found in the signal for unique IP sources per hour.

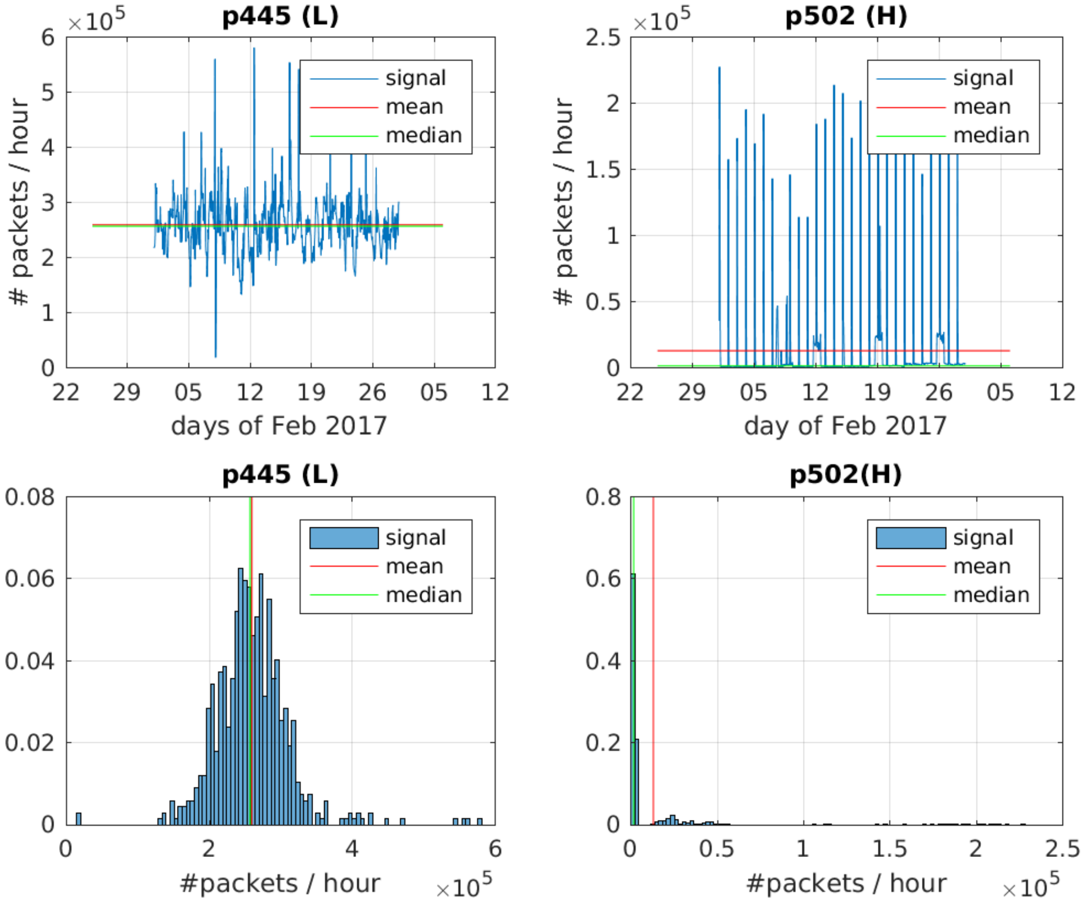


Figure 6: Data with highest and lowest relative difference between mean and median

	FFT maximum value	k	Period (days)	Period (hours)
Packets / hour	180308368.771	2	14	336
IP src / hour	21586333.529	1	28	672

Table 11: Maximum FFT value and futher information

FFT maximum value	Period (days)	Period (hours)	Comment
1.8031e+08	0.0418	1.0030	Hourly pattern
1.6480e+08	28	672	Fundamental frequency
1.6480e+08	0.0417	1.0015	Another hourly pattern
1.5489e+08	1.0370	24.8889	Daily pattern
1.5489e+08	0.0434	1.0419	Another hourly pattern
1.3649e+08	7	168	Weekly pattern
1.3649e+08	0.0419	1.0060	Another hourly pattern

Table 12: Temporal patterns in Packets per hour

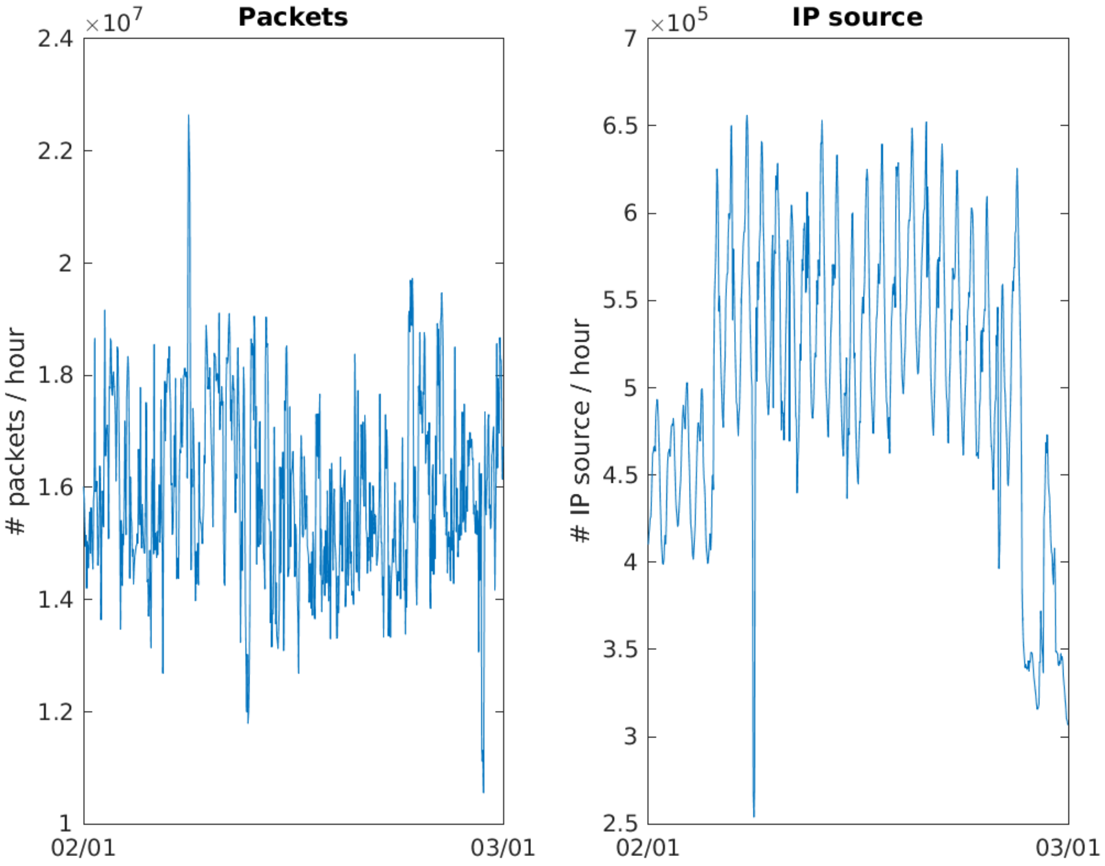


Figure 7: Time series plots for TCP for February 2017

FFT maximum value	Period (days)	Period (hours)	Comment
2.1586e+07	0.0417	1.0015	Hourly pattern
1.8748e+07	1	24	Daily pattern
1.8748e+07	0.0435	1.0435	Another hourly pattern
1.4339e+07	14	336	Bi-weekly pattern
1.4339e+07	0.0418	1.0030	Another hourly pattern

Table 13: Temporal patterns in unique IP sources per hour

1.13 rep-22 → Matlab Code (Listing 13)

Figure 9 shows an average day for the number of packets per hour and the number of unique IP sources per hour for February 2017. The size of the error bars represents the standard deviation.

1.14 rep-23 → Matlab Code (Listing 13)

Table 14 shows the correlation coefficients between the averaged signals for Packets per hour and unique IP sources per hour.

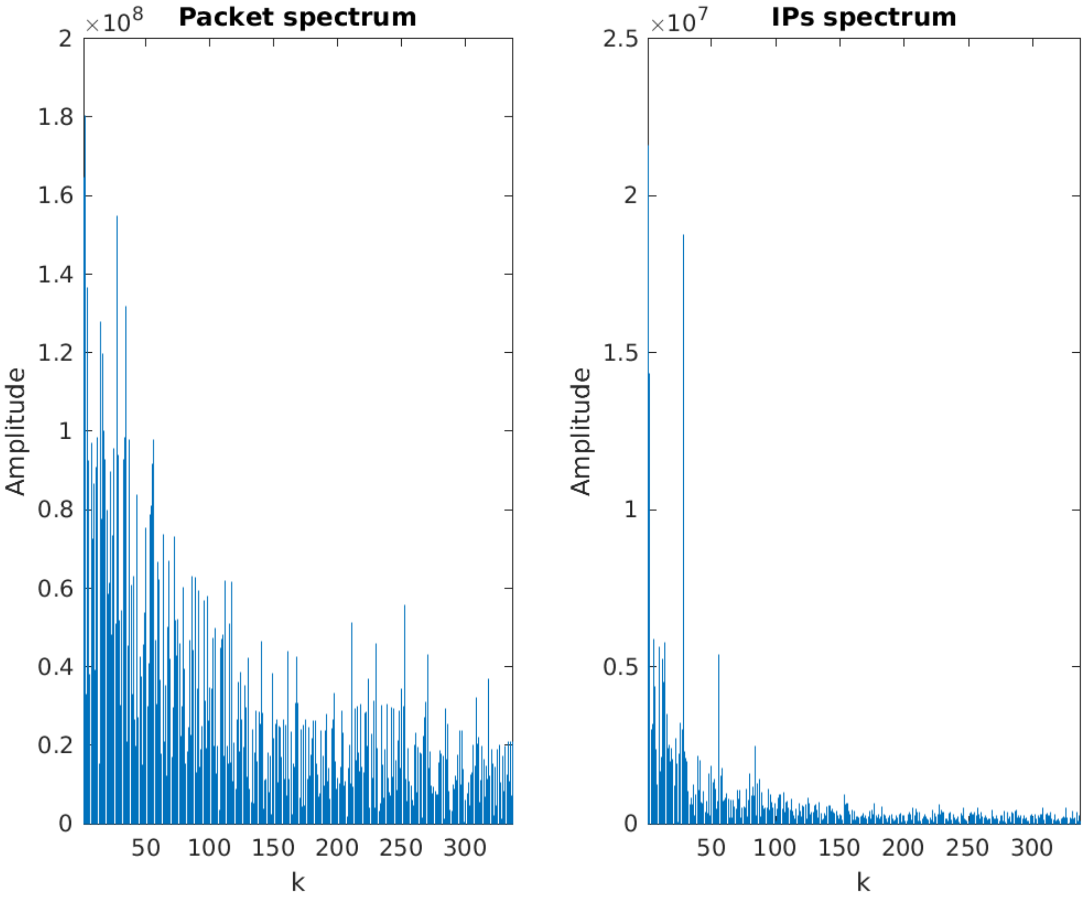
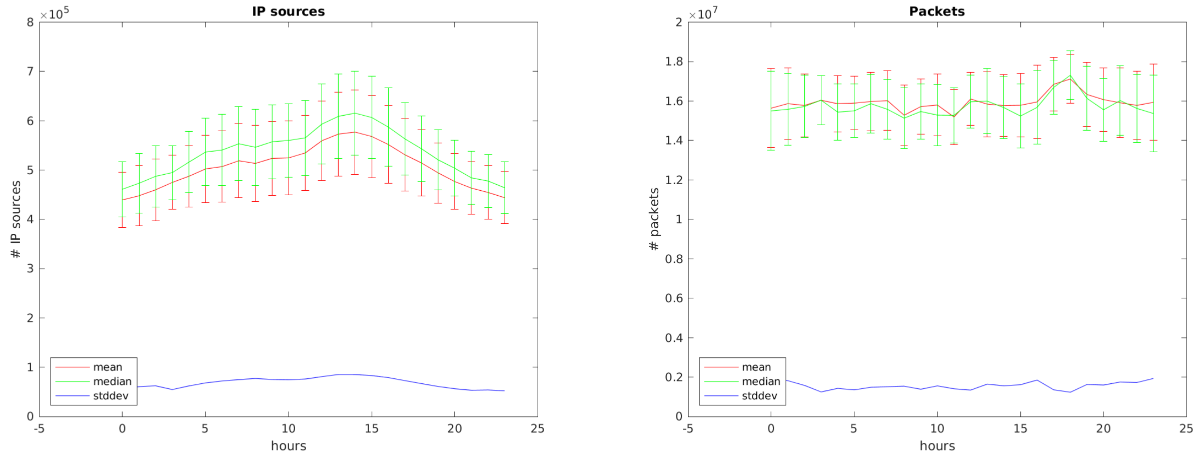


Figure 8: Amplitude spectra for TCP for February 2017

	Correlation coefficient
Mean	0.0348
Median	0.0485

Table 14: Correlation coefficients between Packets per hour and unique IP sources per hour

When averaging with the median the correlation coefficient of the signals is slightly higher. But the signals are not correlated. For the averaged signal for unique IP sources per hour a prominent peak at around 2 pm can be observed. For the averaged signal of Packets per hour no such prominent peak exists. The peak in the unique IP sources per hour could be explained by the fact that somehow more infected PCs are running around this time of the day.



(a) Average day for IP sources per hour

(b) Average day for packets per hour

Figure 9: Average days

2 Lab Exercise 4

2.1 rep-24

We converted the flowrecords into a CSV file using the command shown in Listing 1.

Listing 1: Command used to obtain CSV file

```
team02@pc01:~$ rwcut --num-recs=200000 --delimited=, \
    --fields=sIP,dIP,sPort,dPort,protocol,flags,ttl,bytes \
    ~/workfiles/team02.flowrecord.rw > team02_flowrecord.csv
```

2.2 rep-25

Table 15 shows the three most frequent destination ports. Table 16 shows the three most frequent source ports. Table 17 shows all used protocols.

Port	Rate of appearance (%)
80	21.6
0	3.8
25565	1.6

Table 15: Three most frequent source ports

Port	Rate of appearance (%)
445	41.0
10320	9.6
3072	2.8

Table 16: Three most frequent destination ports

2.3 rep-26

The rate of appearance concering the used protocols was roughly what we expected it to be. Port 445 as the top most used destination port was also no big surprise, but ports 10320 and 3072 where somewhat surprising, because as far as we know no popular applications use these two ports. Interesting to note are the two top most source ports 80 and 0. These ports are probably used to bypass not properly configured firewalls.

2.4 rep-27

Figure 10 shows the TTL frequency distribution. Note the two peaks around a TTL of 45 and a TTL of 107. The peaks in the TTL distribution might be due to the fact, that most of the sources are located “next to each other” (in terms of network connectivity) and therefore take the same number of hops to reach the darkspace.

Protocol	Rate of appearance (%)
TCP (6)	81.5
UDP (17)	15.1
ICMP (1)	3.4
IPv6 (41)	0.1
GRE (47)	0.0

Table 17: All used protocols

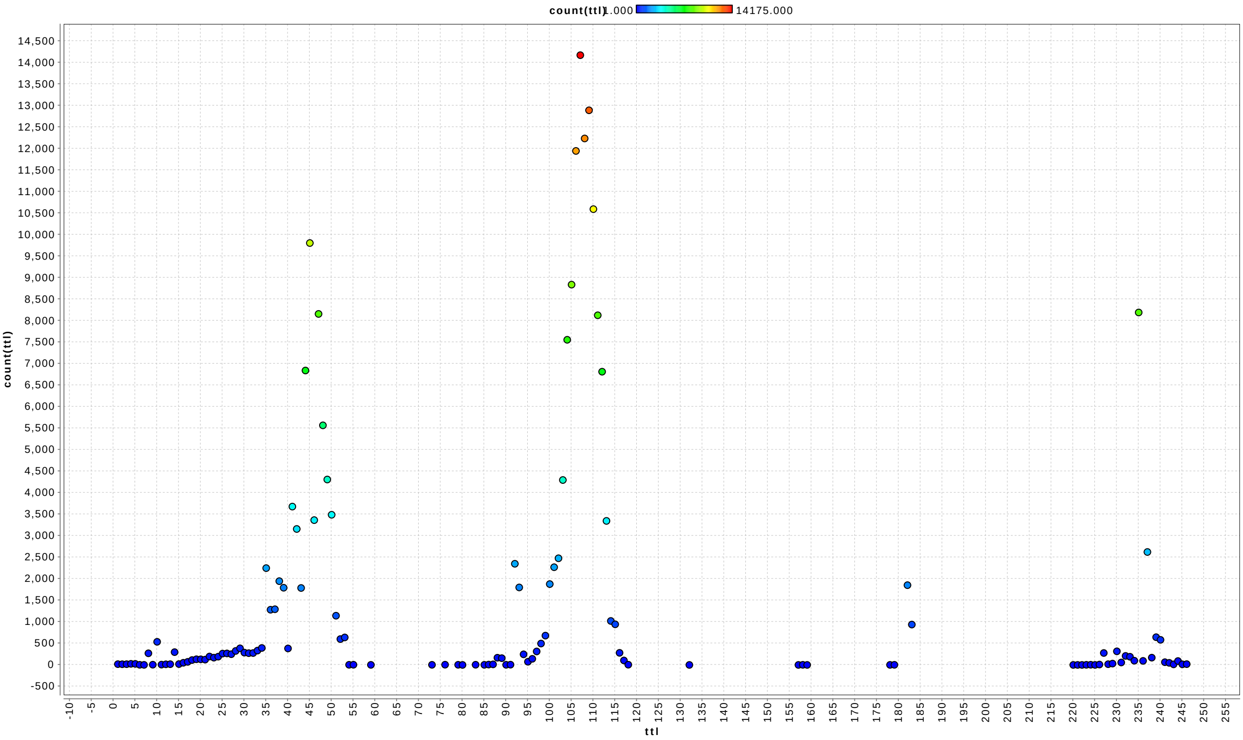


Figure 10: TTL frequency distribution

2.5 rep-28

The most recurring IP source in our dataset is the IP address `236.196.32.152`. It is exclusively using port `80` as a destination port and hitting ports `3072` and `1024` across a range of IP destinations. The source is performing horizontal port scans for ports 3072 and 1024.

2.6 rep-29

To arrive at the desired data we performed some aggregation and filtering instead of inspecting the scatter plot. The source IP that is connecting to the maximum number of different ports is `72.99.52.73`. Figure 11 shows a scatter plot of destination ports against IP destinations for the source IP `72.99.52.73`. The source seems to be performing some kind of combination between a vertical and a horizontal scan up to a specific port number.

2.7 rep-30

The port that is getting the most connections from different IP sources is port `445`. The source IP that is most frequently using this port is `187.154.74.45`. Figure 12 shows the frequency of connection attempts to port 445 from `187.154.74.45`. The source IP is scanning for port 445 over a range of IP destinations. Somehow one IP destination is more interesting than the others.

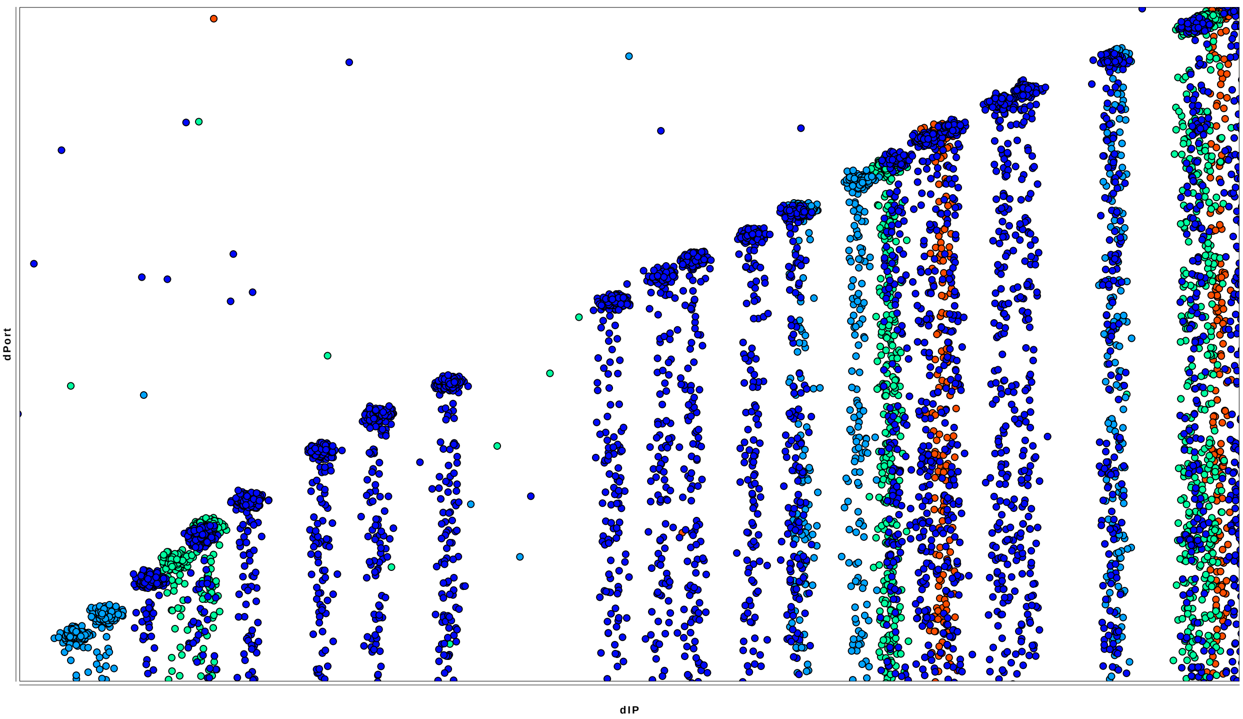


Figure 11: Scatter plot destination ports against IP destinations for source IP `72.99.52.73` (with jitter)

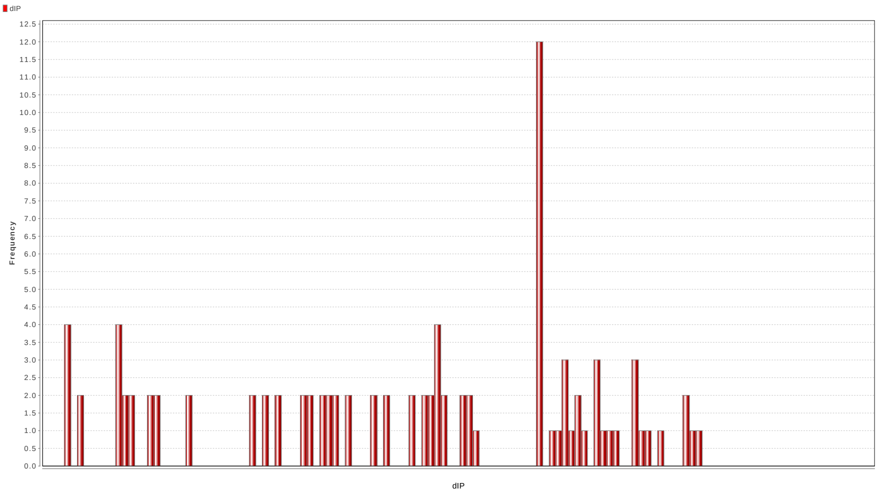


Figure 12: Frequency of connection attempts to port 445 from `187.154.74.45`

A Matlab Code

Listing 2: Matlab code to solve rep-10

```
function team02_rep10
% rep-10
    [timestamps, bytes, packets, ip_s, ip_d] = read_custom_csv('˜/workfiles/
        global_last10years.csv');

    function save_stem_plot(data, my_title, y_label, filename)
    % Do a stem plot of data in millions and write it to filename.png
        set (gca, 'fontname', 'Helvetica', 'fontsize', 20)
        figure
        stem(timestamps, data/10ˆ6, 'marker', 'none')
        datetick('x', 'mm/yy');
        xlabel('days␣of␣observed␣time␣span');
        ylabel(y_label);
        title(my_title);
        grid on
        set(gca, 'layer', 'top');
        xlim([min(timestamps) max(timestamps)]);
        saveas(gcf, filename, 'png')
    end

    save_stem_plot(bytes, 'bytes␣per␣hour', '#bytes␣[million]', 'plots/rep_10_2');
    save_stem_plot(packets, 'packets␣per␣hour', '#packets␣[million]', 'plots/rep_10_1');
    save_stem_plot(ip_s, 'ip␣sources␣per␣hour', '#ip␣sources␣[million]', 'plots/rep_10_3'
        );
    save_stem_plot(ip_d, 'ip␣destinations␣per␣hour', '#ip␣destinations␣[million]', 'plots
        /rep_10_4');

    % optional part

    function result = smooth_filter(data)
    % Moving averages filter for data
        window_size = 30;
        b = (1 / window_size) * ones(1, window_size);
        a = 1;
        % 1-D digital filter
        result = filter(b, a, data);
    end

    smooth_bytes = smooth_filter(bytes / unique(max(bytes)));
    smooth_packets = smooth_filter(packets / unique(max(packets)));
    smooth_ip_s = smooth_filter(ip_s / unique(max(ip_s)));
    smooth_ip_d = smooth_filter(ip_d / unique(max(ip_d)));

    figure
    plot(...
        timestamps, smooth_bytes, '-', ...
        timestamps, smooth_packets, '-', ...
        timestamps, smooth_ip_s, '-', ...
        timestamps, smooth_ip_d, '-' ...
    );
    legend('bytes', 'packets', 'ip␣source', 'ip␣dest');
    datetick('x', 'mm/yy');
    xlabel('days␣of␣observed␣time␣span');
    title('Combined␣plot␣of␣normalized␣and␣smoothed␣signals');
    grid on
    set(gca, 'layer', 'top');
    xlim([min(timestamps) max(timestamps)]);
    saveas(gcf, 'plots/rep_10_optional', 'png')
end
```

Listing 3: Matlab code to solve rep-11

```
function team02_rep11
% rep-11
    [~, bytes, packets, ip_s, ip_d] = read_custom_csv('~/workfiles/global_last10years.csv
        ');

    function result = correlation(a, b)
        result = unique(min(corrcoef(a, b)));
    end

    names = { ...
        'Bytes␣<->␣Packets', 'Bytes␣<->␣IPs', 'Bytes␣<->␣IPd', ...
        'Packets␣<->␣IPs', 'Packets␣<->␣IPd', 'IPs␣<->␣IPd' ...
    };
    correlations = [ ...
        correlation(bytes, packets), correlation(bytes, ip_s), ...
        correlation(bytes, ip_d), correlation(packets, ip_s), ...
        correlation(packets, ip_d), correlation(ip_s, ip_d) ...
    ];
    [minimum_coeff, idx] = min(correlations);
    fprintf('Minimum␣linear␣correlation␣coeff:␣%f␣(%s)\n', minimum_coeff, names{idx});

    names_signal = {'Bytes', 'Packets', 'IPs', 'IPd'};

    means = [ ...
        % Bytes          | Packets          | IPs              | IPd
        correlations(1), correlations(1), correlations(2), correlations(3); ...
        correlations(2), correlations(4), correlations(4), correlations(5); ...
        correlations(3), correlations(5), correlations(6), correlations(6)
    ];
    disp(names_signal);
    disp(means);
end
```

Listing 4: Matlab code to solve rep-12

```
function team02_rep12
    [~, ~, ~, ip_s, ip_d] = read_custom_csv('~/workfiles/global_last10years.csv');
    ip_s(ip_s==0) = NaN;
    ip_d(ip_d==0) = NaN;

    fprintf('Ratio␣IPs␣to␣IPd:␣%f\n', nanmean(ip_s) / nanmean(ip_d));
end
```

Listing 5: Matlab code to solve rep-13

```
function team02_rep13
    [timestamps, ~, ~, ip_s, ~] = read_custom_csv('~/workfiles/global_last10years.csv');
    % from visual inspection
    cutoff = 1.5*10^6;
    peak_locations = ip_s>cutoff;

    peak_timestamps = timestamps(peak_locations);
    peaks = ip_s(peak_locations);

    dates = arrayfun(@datestr, peak_timestamps, 'UniformOutput', false);
    result = dates';
    result(2,:) = num2cell(peaks);
    fprintf('%s:␣%f␣IPs\n', result{:});
end
```

Listing 6: Matlab code to solve rep-13 optional

```
function team02_rep13_optional
    [timestamps, bytes, ~, ~, ~] = read_custom_csv('~/workfiles/global_last10years.csv');
    % From visual inspection
    cutoff = 8*10^8;
    timestamps = timestamps(timestamps<=datenum('2014-01-01'));
    bytes = bytes(timestamps>0);

    peak_locations = bytes>cutoff;
    peak_timestamps = timestamps(peak_locations);
    peaks = bytes(peak_locations);

    dates = arrayfun(@datestr, peak_timestamps, 'UniformOutput', false);
    result = dates';
    result(2,:) = num2cell(peaks);
    fprintf('%s:␣%f␣Bytes\n', result{:});
    % NOTE: There is a gap because on 19-nov-2012 there was no data
end
```

Listing 7: Matlab code to solve rep-14

```
function team02_rep14

    function result = stats(data)
        data(data==0) = NaN;
        result = round([nansum(data), nanmean(data), nanmedian(data), nanstd(data)] ./ 10
            e6, 3);
    end

    disp('----␣Daily␣avg␣---');
    [~, bytes, packets, ip_s, ip_d] = read_custom_csv('~/workfiles/global_last10years.csv
       ');
    for col = horzcat(bytes, packets, ip_s, ip_d)
        fprintf('%.3f␣%.3f␣%.3f␣%.3f\n', stats(col));
    end

    disp('-----␣Hourly␣avg␣---');

    % WARNING: order is different
    [~, packets, bytes, ip_s, ip_d] = read_custom_csv('~/workfiles/Feb2017_gen.csv');
    for col = horzcat(bytes, packets, ip_s, ip_d)
        fprintf('%.3f␣%.3f␣%.3f␣%.3f\n', stats(col));
    end
end
```

Listing 8: Matlab code to solve rep-15 optional

```
function team02_rep15_optional
    [~, bytes_daily, packets_daily, ip_s_daily, ip_d_daily] = read_custom_csv('~/
       workfiles/global_last10years.csv');
    % WARNING order is different
    [~, packets_hourly, bytes_hourly, ip_s_hourly, ip_d_hourly] = read_custom_csv('~/
       workfiles/Feb2017_gen.csv');

    set (gca, 'fontname', 'Helvetica', 'fontsize', 20)

    ax1 = subplot(2,4,1);
    boxplot(ax1, bytes_hourly, 'Labels', {''})
    ylabel(ax1, 'Bytes');
    xlabel(ax1, 'Bytes␣/␣hour');
    grid on
    set(gca, 'layer', 'top');

    ax2 = subplot(2,4,2);
    boxplot(ax2, packets_hourly, 'Labels', {''})
    ylabel(ax2, 'Packets');
    xlabel(ax2, ’Packets␣/␣hour’);
    grid on
    set(gca, ’layer’, ’top’);

    ax3 = subplot(2,4,3);
    boxplot(ax3, ip_s_hourly, ’Labels’, {’’})
    ylabel(ax3, ’IPs’);
    xlabel(ax3, ’IPs␣/␣hour’);
    grid on
    set(gca, ’layer’, ’top’);

    ax4 = subplot(2,4,4);
    boxplot(ax4, ip_d_hourly, ’Labels’, {’’})
    ylabel(ax4, ’IPd’);
    xlabel(ax4, ’IPd␣/␣hour’);
    grid on
    set(gca, ’layer’, ’top’);

    ax5 = subplot(2,4,5);
    boxplot(ax5, bytes_daily, ’Labels’, {’’})
    ylabel(ax5, ’Bytes’);
    xlabel(ax5, ’Bytes␣/␣day’);
    grid on
    set(gca, ’layer’, ’top’);

    ax6 = subplot(2,4,6);
    boxplot(ax6, packets_daily, ’Labels’, {’’})
    ylabel(ax6, ’Packets’);
    xlabel(ax6, ’Packets␣/␣day’);
    grid on
    set(gca, ’layer’, ’top’);

    ax7 = subplot(2,4,7);
    boxplot(ax7, ip_s_daily, ’Labels’, {’’})
    ylabel(ax7, ’IPs’);
    xlabel(ax7, ’IPs␣/␣day’);
    grid on
    set(gca, ’layer’, ’top’);

    ax8 = subplot(2,4,8);
    boxplot(ax8, ip_d_daily, ’Labels’, {’’})
    ylabel(ax8, ’IPd’);
    xlabel(ax8, ’IPd␣/␣day’);
    grid on
    set(gca, ’layer’, ’top’);

    saveas(gcf, ’plots/rep_15_optional.png’, ’png’)
end
```

Listing 9: Matlab code to solve rep-17

```
function team02_rep17
    % WARNING: order is switched
    [˜, combined_packets, ˜, combined_ip_s, combined_ip_d] = read_custom_csv(’˜/workfiles
        /Feb2017_gen.csv’);
    [˜, tcp, udp, icmp] = read_custom_protocol_csv(’˜/workfiles/Feb2017_proto.csv’);
    % packets, ip_s, ip_d

   function result = packets(data)
        result = data(:,1);
    end

    function result = ip_s(data)
        result = data(:,2);
    end
```

```
function result = ip_d(data)
    result = data(:,3);
end

others_packets = combined_packets - packets(tcp) - packets(udp) - packets(icmp);
others_ip_s = combined_ip_s - ip_s(tcp) - ip_s(udp) - ip_s(icmp);
others_ip_d = combined_ip_d - ip_d(tcp) - ip_d(udp) - ip_d(icmp);
others = horzcat(others_packets, others_ip_s, others_ip_d);

function result = percentages(data)
    p = packets(data) ./ combined_packets;
    s = ip_s(data) ./ combined_ip_s;
    d = ip_d(data) ./ combined_ip_d;
    result = horzcat(p, s, d);
end

tcp_percentages = percentages(tcp);
udp_percentages = percentages(udp);
icmp_percentages = percentages(icmp);
others_percentages = percentages(others);

function table(t, u, i, o)
    fprintf(’%f␣%f␣%f\n’, mean(t), median(t), std(t));
    fprintf(’%f␣%f␣%f\n’, mean(u), median(u), std(u));
    fprintf(’%f␣%f␣%f\n’, mean(i), median(i), std(i));
    fprintf(’%f␣%f␣%f\n’, mean(o), median(o), std(o));
end

table(packets(tcp_percentages), packets(udp_percentages), packets(icmp_percentages),
    packets(others_percentages));
disp(’--’);
table(ip_s(tcp_percentages), ip_s(udp_percentages), ip_s(icmp_percentages), ip_s(
    others_percentages));
disp(’--’);
table(ip_d(tcp_percentages), ip_d(udp_percentages), ip_d(icmp_percentages), ip_d(
    others_percentages));

function stat_plot(data, my_title)
    figure
    ax1 = subplot(1, 3, 1);
    boxplot(ax1, packets(data), ’Labels’, {’’})
    ylabel(ax1, ’Packets’);
    xlabel(ax1, ’Packets␣/␣hour’);
    grid on
    set(gca, ’layer’, ’top’);

    ax2 = subplot(1, 3, 2);
    boxplot(ax2, ip_s(data), ’Labels’, {’’})
    ylabel(ax2, ’IPs’);
    xlabel(ax2, ’IPs␣/␣hour’);
    title(my_title)
    grid on
    set(gca, ’layer’, ’top’);

    ax3 = subplot(1, 3, 3);
    boxplot(ax3, ip_d(data), ’Labels’, {’’})
    ylabel(ax3, ’IPd’);
    xlabel(ax3, ’IPd␣/␣hour’);
    grid on
    set(gca, ’layer’, ’top’);

    saveas(gcf, strcat(’plots/rep_17_’, my_title, ’.png’), ’png’)
end

stat_plot(tcp_percentages, ’TCP’)
stat_plot(udp_percentages, ’UDP’)
```

```
    stat_plot(icmp_percentages, 'ICMP')
    stat_plot(others_percentages, 'OTHERS')

    % Optional part

    function plot_scatter(fun_x, fun_y, label_x, label_y)
        figure
        scatter(fun_x(tcp_percentages), fun_y(tcp_percentages), '.r');
        hold on
        scatter(fun_x(udp_percentages), fun_y(udp_percentages), '.b');
        scatter(fun_x(icmp_percentages), fun_y(icmp_percentages), '.g');
        scatter(fun_x(others_percentages), fun_y(others_percentages), '.m');
        legend({'tcp', 'udp', 'icmp', 'others'}, 'Location', 'southeast');
        ylabel(label_y);
        xlabel(label_x);
        grid on;
        set(gca, 'layer', 'top');
        saveas(gcf, strcat('plots/rep_17_optional_', label_x, label_y, '.png'), 'png')
    end

    plot_scatter(@ip_s, @ip_d, 'IPs', 'IPd');
    plot_scatter(@ip_s, @packets, 'IPs', 'Packets');
    plot_scatter(@ip_d, @packets, 'IPd', 'Packets');


end
```

Listing 10: Matlab code to solve rep-19

```
function team02_rep19()
    [port_data, column_names] = read_tcp_ports_csv('~/workfiles/Feb2017_TCPdstport.csv');
    [~, tcp, ~, ~] = read_custom_protocol_csv('~/workfiles/Feb2017_proto.csv');

    tcp_packets = tcp(:,1);
    % for element-wise division later on
    tcp_packets = horzcat(tcp_packets, tcp_packets, tcp_packets, tcp_packets);

    % We don't want the timestamp
    port_data = port_data(:,2:end);
    column_names = column_names(:,2:end);

    % Sum over the columns and sort descending
    [~, idx] = sort(sum(port_data), 'descend');

    port_data = port_data(:,idx);
    port_data = port_data(:,1:4);

    column_names = column_names(idx);
    fprintf('Most␣used␣ports:␣%s␣%s␣%s␣%s\n', column_names{:,1:4});

    % Absolute values
    disp('Absolute');
    fprintf('mean␣%.3f␣%.3f␣%.3f␣%.3f\n', (mean(port_data) ./ 10e6));
    fprintf('std␣␣%.3f␣%.3f␣%.3f␣%.3f\n', (std(port_data) ./ 10e6));

    disp('Perc');
    fprintf('mean␣%.1f␣%.1f␣%.1f␣%.1f\n', mean(port_data ./ tcp_packets) * 100);
    fprintf('std␣␣%.1f␣%.1f␣%.1f␣%.1f\n', std(port_data ./ tcp_packets) * 100);
end
```

Listing 11: Matlab code to solve rep-20

```
function team02_rep20()
    [port_data, column_names] = read_tcp_ports_csv('~/workfiles/Feb2017_TCPdstport.csv');

    ts = epoch_to_date(port_data(:,1));
```

```
% We don't want to analyze the timestamp
port_data = port_data(:,2:end);
column_names = column_names(:,2:end);

means = mean(port_data);
medians = median(port_data);

function result = difference(means_, medians_)
    result = abs((means_ - medians_) ./ medians_);
end

[~, idx] = sort(bsxfun(@difference, means, medians));
column_names = column_names(idx);
means = means(idx);
medians = medians(idx);
port_data = port_data(:,idx);

L_name = column_names(1);
L_mean = means(1);
L_median = medians(1);
L = port_data(:,1);

H_name = column_names(end);
H_mean = means(end);
H_median = medians(end);
H = port_data(:,end);

set(gca, 'fontname', 'Helvetica', 'fontsize', 20)

figure
subplot(2, 2, 1);
plot(ts, L);
hold on;
plot(xlim, [L_mean, L_mean], 'r');
plot(xlim, [L_median, L_median], 'g');
legend('signal', 'mean', 'median');
title(strcat(L_name, '␣(L)'));
ylabel('#␣packets␣/␣hour');
datetick('x', 'dd');
xlabel('days␣of␣Feb␣2017');
grid on
set(gca, 'layer', 'top');

subplot(2, 2, 2);
plot(ts, H);
hold on;
plot(xlim, [H_mean, H_mean], 'r');
plot(xlim, [H_median, H_median], 'g');
legend('signal', 'mean', 'median');
title(strcat(H_name, '␣(H)'));
ylabel('#␣packets␣/␣hour');
datetick('x', 'dd');
xlabel('day␣of␣Feb␣2017');
grid on
set(gca, 'layer', 'top');

subplot(2, 2, 3);
histogram(L, 100, 'Normalization', 'probability');
hold on;
line([L_mean, L_mean], ylim, 'Color', 'r');
line([L_median, L_median], ylim, 'Color', 'g');
legend('signal', 'mean', 'median');
title(strcat(L_name, '␣(L)'));
grid on
xlabel('#packets␣/␣hour');
set(gca, 'layer', 'top');
```

```
    subplot(2,2,4);
    histogram(H, 100, 'Normalization', 'probability');
    line([H_mean, H_mean], ylim, 'Color', 'r');
    line([H_median, H_median], ylim, 'Color', 'g');
    legend('signal', 'mean', 'median');
    title(strcat(H_name, '(H)'));
    grid on
    xlabel('#packets␣/␣hour');
    set(gca, 'layer', 'top');

    saveas(gcf, 'plots/rep_20.png', 'png')
end
```

Listing 12: Matlab code to solve rep-21

```
function team02_rep21()
    [ts, tcp, ~, ~] = read_custom_protocol_csv('~/workfiles/Feb2017_proto.csv');

    N = length(ts);
    N2 = floor(N/2);

    tcp_packets = tcp(:,1);
    tcp_ip_s = tcp(:,2);

    % (a)

    figure
    subplot(1, 2, 1);
    plot(epoch_to_date(ts), tcp_packets);
    title('Packets');
    ylabel('#␣packets␣/␣hour');
    datetick('x', 'mm/dd');


    subplot(1, 2, 2);
    plot(epoch_to_date(ts), tcp_ip_s);
    title('IP␣source');
    ylabel('#␣IP␣source␣/␣hour');
    datetick('x', 'mm/dd');

    saveas(gcf, 'plots/rep_21_a.png', 'png')

    function plot_spectrum(amplitudes)
        k = (1:(N2+1));
        stem(k, amplitudes(1:(N2 + 1)), 'marker', 'none');
        xlim([1 N2+1]);
        xlabel('k');
        ylabel('Amplitude');
    end

    function [a, offset] = amplitudes(data)
        data(data == 0) = median(data);
        data_fft = fft(data);
        data_abs = abs(data_fft);
        a = data_abs(2:end);
        offset = data(1);
    end

    function f = freq(k)
        f = k / N;
    end

    function p = period(k)
        p = N / k;
    end
    packet_amp = amplitudes(tcp_packets);
    ip_s_amp = amplitudes(tcp_ip_s);

    % (b)

    figure
    subplot(1, 2, 1);
    plot_spectrum(packet_amp);
    title('Packet␣spectrum');
    subplot(1, 2, 2);
    plot_spectrum(ip_s_amp);
    title('IPs␣spectrum');

    saveas(gcf, 'plots/rep_21_b.png', 'png')

    %[v, k] = max(a(2:end));
    %freq(k)
    %period(k)

    function max_fft_info(amplitudes)
        [v, k] = max(amplitudes);
        fprintf('max:␣%.3f␣max_k:␣%d␣period:␣%.3f␣hours␣(%.3f␣days)\n', v, k, period(k),
            period(k) / 24);
    end

    % (c)
    max_fft_info(packet_amp);
    max_fft_info(ip_s_amp);

    [max_amps max_ks] = sort(packet_amp, 'descend');

    function report_fft(max_amps, max_ks)
        for i=1:20
            disp(max_amps(i));
            disp(period(max_ks(i)) / 24);
            disp(period(max_ks(i)));
            disp('--');
        end
    end

    disp('packets');
    [a, k] = sort(packet_amp, 'descend');
    report_fft(a, k);

    disp('ip_s');
    [a, k] = sort(ip_s_amp, 'descend');
    report_fft(a, k);


end
```

Listing 13: Matlab code to solve rep-22 and rep-23

```
function team02_rep22
    [ts, tcp, ~, ~] = read_custom_protocol_csv('~/workfiles/Feb2017_proto.csv');

    num_days = length(ts) / 24;

    tcp_packets = tcp(:,1);
    tcp_ip_s = tcp(:,2);

    function [mean_day, median_day, std_day] = average_day(data)
        days = reshape(data, [24, num_days]);
        % ... over the rows
        mean_day = mean(days, 2);
        median_day = median(days, 2);
        std_day = std(days, 0, 2);
    end

    function plot_day(data)
        [mean_day, median_day, std_day] = average_day(data);
        hours = 0:23;
        figure
        errorbar(hours, mean_day, std_day, 'r-')
        hold on
        errorbar(hours, median_day, std_day, 'g-')
        plot(hours, std_day, 'b-');
        legend({'mean', 'median', 'stddev'}, 'Location', 'southwest');
    end

    plot_day(tcp_packets)
    title('Packets');
    xlabel('hours');
    ylabel('#␣packets');
    saveas(gcf, 'plots/rep_22_packets.png', 'png')

    plot_day(tcp_ip_s)
    xlabel('hours');
    title('IP␣sources');
    ylabel('#␣IP␣sources');
    saveas(gcf, 'plots/rep_22_ip_s.png', 'png')

    % rep-23

    [packets_mean, packets_median] = average_day(tcp_packets);
    [ip_s_mean, ip_s_median] = average_day(tcp_ip_s);

    disp('mean');
    disp(corrcoef(packets_mean, ip_s_mean));
    disp('median');
    disp(corrcoef(packets_median, ip_s_median));
end
```